



INTRODUCTION TO IMAGE PROCESSING AND COMPUTER VISION

LABORATORY PROJECT 2 (LABORATORIES 3 & 4)

Rafał Józwiak

R. Jozwiak@mini.pw.edu.pl

Faculty of Mathematics and Information Science







Warsaw University of Technology

REALIZATION

- algorithms elaborated with OpenCV library
 - OpenCV (C++)
 - SimleCV/OpenCV (Python)
 - EmugCV (C#)
- usage of other libraries (for texture characterization and ML) is also allowed
- solution for the laboratory task should contain:
 - source code with description (GUI is not obligatory)
 - documentation (description of solution, testing procedure, results and comments)
- solution should be send up to 24.01.2022

LEAFSNAP DATASET

- input: images of leaves with segmentation masks
- *LeafSnap* – **public dataset**
- 23147 Lab images, consisting of high-quality images taken of pressed leaves, from the Smithsonian collection. These images appear in controlled backlit and front-lit versions, with several samples per species
- The dataset currently covers all 185 tree species from the Northeastern United States
- dataset contains original images (field and lab), segmented images, text index of images – **use lab version and appropriate segmentation masks**

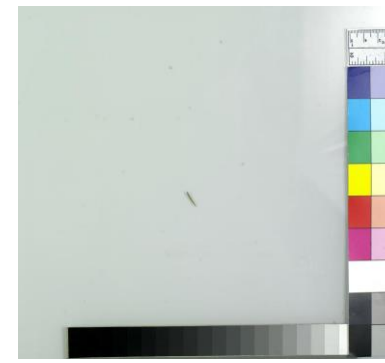
<u>Species</u>	<u>Example lab image</u>	<u>Example lab segmentation</u>
<i>Halesia tetraptera</i>		
<i>Ostrya virginiana</i>		
<i>Nyssa sylvatica</i>		

TREE SPECIES RECOGNITION

- aim: elaboration of discriminative feature vector (for classification purpose), to assess quality of different features (assuming different research scenario)
- input: samples (about 800 x 800)
- output: sample class -> classification accuracy -> features quality
- try to use different features (shape features, texture features, LBP, local features descriptors, etc.)
- for the purpose of classification use any classifier (usage of different environments is allowed e.g. R, Python, Matlab etc.; try to use quite simply classifier) – DO NOT CHANGE DURING EXPERIMENTS!!!
- try to prepare analysis for:
 - different types of texture features (initially independently) e.g. statistical, GLCM, LBP, etc.
 - combining different groups of features together
- for learning and classification evaluation use:
 - divide into training and testing set (80% training / 20% testing)
- usage of CNN (as a magic black box) is NOT ALLOWED !!!

YOUR DATASET

- select your own dataset for experiments (subset of LeafSnap)
- select min. 10 – 12 different classes of tree species
- try to use diversified species (both completely different and quite similar)
- avoid small and difficult classes / species



cedrus_atlantica (80)



quercus_falcata (7)



pinus_cembra (120) ???



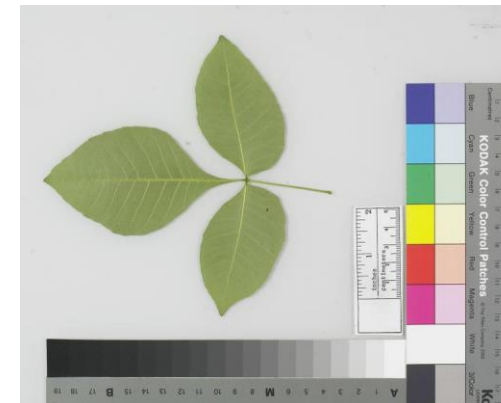
betula_jacquemontii (120)



betula_alleghaniensis (120)



eucommia_ulmoides (120)



ptelea_trifoliata (220)

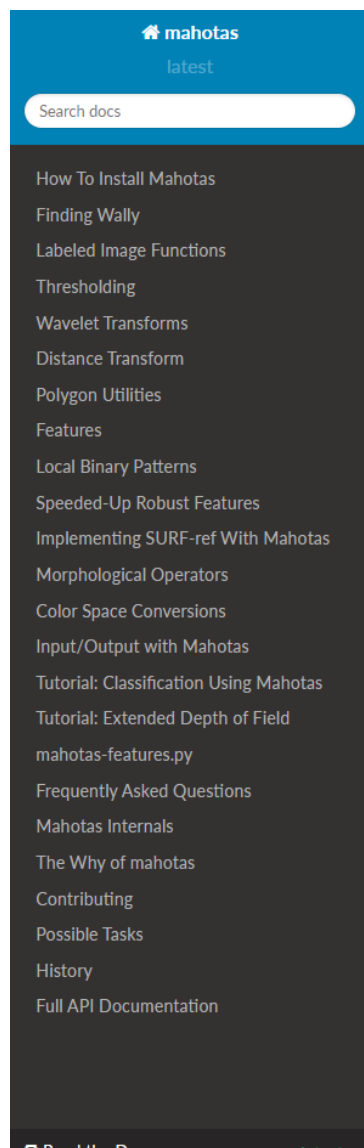


quercus_nigra (120)

DOCUMENTATION

- documentation should contain:
 - introduction, problem definition (task description)
 - your data set preparation and description
 - description of solution (algorithm, step by step explanation and visualization, proposed improvements)
- results:
 - mean results for different tree species / groups of species / whole data set
 - mean results for different types / groups of texture features
 - examples of good and bad segmentation results
- comments and conclusions

TOOLS



Docs » Mahotas: Computer Vision in Python

[Edit on GitHub](#)

Mahotas: Computer Vision in Python

Note

If you are using mahotas in a scientific publication, please cite:

Coelho, L.P. 2013. Mahotas: Open source software for scriptable computer vision. Journal of Open Research Software 1(1):e3, DOI: <http://dx.doi.org/10.5334/jors.ac>

Mahotas is a computer vision and image processing library for Python.

It includes many algorithms implemented in C++ for speed while operating in numpy arrays and with a very clean Python interface.

Mahotas currently has over 100 functions for image processing and computer vision and it keeps growing. Some examples of mahotas functionality:

- [watershed](#)
- convex points calculations.
- [hit & miss](#), [thinning](#)
- Zernike & Haralick, [local binary patterns](#), and TAS features.
- [morphological processing](#)
- [Speeded-Up Robust Features \(SURF\)](#), a form of local features
- [thresholding](#)
- convolution.
- Sobel edge detection.

The release schedule is roughly one release every few months and each release brings new functionality and improved performance. The interface is very stable, though, and code written using a version of mahotas from years back will work just fine in the current version, except it will be faster (some interfaces are deprecated and will be removed after a few years, but in the meanwhile, you only get a warning).

Bug reports with test cases typically get fixed in 24 hours.

<https://mahotas.readthedocs.io/en/latest/index.html>

EXAMPLES

Image Classification using Python and Scikit-learn

Global Feature Descriptors

These are the feature descriptors that quantifies an image globally. These don't have the concept of interest points and thus, takes in the entire image for processing. Some of the commonly used global feature descriptors are

- **Color** – Color Channel Statistics (Mean, Standard Deviation) and Color Histogram
- **Shape** – Hu Moments, Zernike Moments
- **Texture** – Haralick Texture, Local Binary Patterns (LBP)
- **Others** – Histogram of Oriented Gradients (HOG), Threshold Adjacency Statistics (TAS)

Local Feature Descriptors

These are the feature descriptors that quantifies local regions of an image. Interest points are determined in the entire image and image patches/regions surrounding those interest points are considered for analysis. Some of the commonly used local feature descriptors are

- SIFT (Scale Invariant Feature Transform)
- SURF (Speeded Up Robust Features)
- ORB (Oriented Fast and Rotated BRIEF)
- BRIEF (Binary Robust Independent Elementary Features)

Combining Global Features

There are two popular ways to combine these feature vectors.

- For global feature vectors, we just concatenate each feature vector to form a single global feature vector. This is the approach we will be using in this tutorial.
- For local feature vectors as well as combination of global and local feature vectors, we need something called as Bag of Visual Words (BOVW). This approach is not discussed in this tutorial, but there are lots of resources to learn this technique. Normally, it uses Vocabulary builder, K-Means clustering, Linear SVM, and Td-Idf vectorization.

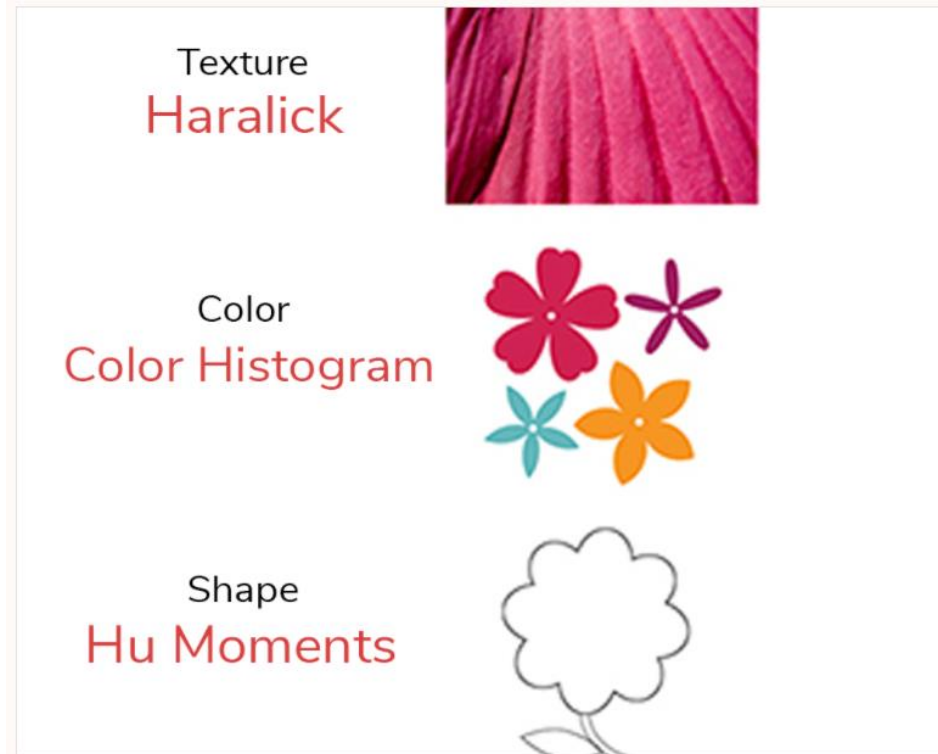


Figure 4. Global Features to quantify a flower image.

ASSESSMENT

- source code with description (idea, meritorical (content-related) assessment) – 10 pts
- documentation – 20 pts
 - introduction, problem definition – 2 pts
 - dataset preparation and description – 5 pts
 - description of solution (different experiments, explanation) – 5 pts
 - results (effectiveness and presentation layer) – 6 pts
 - comments and conclusions – 2 pts