

Internationaler Studiengang Medieninformatik (IMI) – Master

**Masterarbeit**

Computergestützte Identifizierung von Pflanzen anhand ihrer Blattmerkmale

Wintersemester 2016/2017

Student: Christoph Franke

549642

Betreuer: Prof. Dr. Klaus Jung

Nico Hezel

**Inhaltsverzeichnis**

[1. Einleitung 1](#_Toc475297866)

[2. Grundlagen 1](#_Toc475297867)

[2.1. Botanik 1](#_Toc475297868)

[2.1.1. Relevante Merkmale zur Bestimmung von Pflanzen 1](#_Toc475297869)

[2.1.2. Blattmerkmale 1](#_Toc475297870)

[2.1.2.1. Blattform 1](#_Toc475297871)

[2.1.2.2. Blattfarbe 2](#_Toc475297872)

[2.1.2.3. Blattadern 2](#_Toc475297873)

[2.2. PreProcessing 2](#_Toc475297874)

[2.2.1. Binarisierung 2](#_Toc475297875)

[2.2.2. Segmentierung 2](#_Toc475297876)

[2.3. Deskriptoren 2](#_Toc475297877)

[2.3.1. Featurevektoren 2](#_Toc475297878)

[2.3.2. Momentenbasierte geometrische Merkmale 2](#_Toc475297879)

[2.3.3. Fourier-Transformation 2](#_Toc475297880)

[2.3.4. Hauptkomponentenanalyse 2](#_Toc475297881)

[2.4. Maschinelle Lernverfahren 2](#_Toc475297882)

[2.4.1. Klassifikatoren 3](#_Toc475297883)

[2.4.1.1. MMC Hypersphere??? 3](#_Toc475297884)

[2.4.1.2. Support Vector Machine 3](#_Toc475297885)

[2.4.1.3. k-nearest neighbours 3](#_Toc475297886)

[2.4.2. Künstliche Neuronale Netze (teilweise auch Klassifikatoren?) 3](#_Toc475297887)

[2.4.2.1. MultiLayerPerceptrons 3](#_Toc475297888)

[2.4.2.2. PNN 3](#_Toc475297889)

[2.4.2.3. PCNN 3](#_Toc475297890)

[3. Konzeption 3](#_Toc475297891)

[3.1. Anforderungsanalyse 3](#_Toc475297892)

[3.1.1. Kriterien des geplanten Produkts 3](#_Toc475297893)

[3.1.2. Produkteinsatz 3](#_Toc475297894)

[3.1.3. Produktumgebung 3](#_Toc475297895)

[3.2. Vorhandene Lösungen 3](#_Toc475297896)

[3.2.1. Überblick 4](#_Toc475297897)

[3.2.2. Form 5](#_Toc475297898)

[3.2.3. Blatttextur 5](#_Toc475297899)

[3.2.4. Fourier-Transformation 5](#_Toc475297900)

[3.3. Lösungsansatz 6](#_Toc475297901)

[3.3.1. Datensatz 6](#_Toc475297902)

[3.3.2. Segmentierung der Blätter 7](#_Toc475297903)

[3.4. Wahl und Extraktion der Features 8](#_Toc475297904)

[3.4.1. Blattform 8](#_Toc475297905)

[3.4.1.1. Momente 9](#_Toc475297906)

[3.4.1.2. Shape Context 9](#_Toc475297907)

[3.4.1.3. Blattadern 9](#_Toc475297908)

[3.4.2. Blattadern 9](#_Toc475297909)

[3.4.2.1. Local binary pattern 9](#_Toc475297910)

[3.4.3. Blattadern 9](#_Toc475297911)

[3.4.4. PCNN? 10](#_Toc475297912)

[3.4.5. Klassifizierung 11](#_Toc475297913)

[3.4.6. Metrik zur Messung der Erkennungsgenauigkeit 11](#_Toc475297914)

[3.5. Systementwurf 11](#_Toc475297915)

[3.5.1. Systemkomponenten 11](#_Toc475297916)

[3.5.2. Datenbank? 11](#_Toc475297917)

[4. Implementierung 12](#_Toc475297918)

[4.1. Verwendete Bibliotheken 12](#_Toc475297919)

[4.2. Testumgebung 12](#_Toc475297920)

[4.3. allgemein 12](#_Toc475297921)

[4.4. Segmentierung 13](#_Toc475297922)

[4.5. Inner Distance Shape Context 15](#_Toc475297923)

[4.6. Gaussian Pyramid Inner Distance Shape Context 16](#_Toc475297924)

[4.6.1. Codebooks 17](#_Toc475297925)

[4.6.2. Multilayer Perceptron 18](#_Toc475297926)

[4.6.3. Support Vector Machine 18](#_Toc475297927)

[4.6.4. Datenhaltung 18](#_Toc475297928)

[5. Ergebnis 18](#_Toc475297929)

[5.1. Genauigkeit (welche Metrik?) 18](#_Toc475297930)

[5.2. Aufwandsanalyse? 18](#_Toc475297931)

[6. Zusammenfassung 18](#_Toc475297932)

[6.1. Ausblick 18](#_Toc475297933)

[I. Abbildungsverzeichnis 19](#_Toc475297934)

[II. Tabellenverzeichnis 19](#_Toc475297935)

[III. Codelistings 19](#_Toc475297936)

[a. Installationsanleitung 22](#_Toc475297937)

# Einleitung

Erkennung von Pflanzen herausforderndes Problem der computer vision, unregelmäßige Formen, hochvariable Texturen(?)

# Grundlagen

## 

Kürzeste Wege?

## Botanik

### Relevante Merkmale zur Bestimmung von Pflanzen

Blätter, Speicherorgane, Blüten, Früchte

[books.google.de/books?id=e1H\_DAAAQBAJ&pg=PA135&lpg=PA135&dq=W.+Rothmaler+(2002):+Exkursionsflora+von+Deutschland+-+Band+2:+Gef%C3%A4%C3%9Fpflanzen&source=bl&ots=C1OQBgV0tM&sig=4w4Q9PdbJRfdvjrbL4bpsHxAL-s&hl=de&sa=X&ved=0ahUKEwiw1InHrqjQAhVFtxQKHenTDIs4ChDoAQgoMAM#v=onepage&q=W.%20Rothmaler%20(2002)%3A%20Exkursionsflora%20von%20Deutschland%20-%20Band%202%3A%20Gef%C3%A4%C3%9Fpflanzen&f=false](https://books.google.de/books?id=e1H_DAAAQBAJ&pg=PA135&lpg=PA135&dq=W.+Rothmaler+(2002):+Exkursionsflora+von+Deutschland+-+Band+2:+Gef%C3%A4%C3%9Fpflanzen&source=bl&ots=C1OQBgV0tM&sig=4w4Q9PdbJRfdvjrbL4bpsHxAL-s&hl=de&sa=X&ved=0ahUKEwiw1InHrqjQAhVFtxQKHenTDIs4ChDoAQgoMAM#v=onepage&q=W.%20Rothmaler%20(2002)%3A%20Exkursionsflora%20von%20Deutschland%20-%20Band%202%3A%20Gef%C3%A4%C3%9Fpflanzen&f=false)

### Blattmerkmale

http://www.spektrum.de/lexikon/biologie/blatt/9145

#### Blattform

Grundform, Formen von Blattbasis und Blattspitze, Ränder, Anordnung von Spreitenabschnitten bzw. von Blättchen

Seitenquerschnitt

Blattstellung

Größe

#### Blattfarbe

Farbe, Panaschierung … (Leafnet hat Farbschema an Seite)

#### Blattadern

aus Zentrum oder von Mittelader abgehend

Haare

## PreProcessing

### Binarisierung

### Segmentierung

## Deskriptoren

### Featurevektoren

### Momentenbasierte geometrische Merkmale

### Fourier-Transformation

### Hauptkomponentenanalyse

## Maschinelle Lernverfahren

oder das hier Klassifizierung nennen?

Support vector machine?? Pulse-coupled neural network?? [Leaf recognition based on PCNN]

Features einfach reinschmeissen? (zB Blattadern extrahieren und rein damit)

### Klassifikatoren

#### MMC Hypersphere???

#### Support Vector Machine

#### k-nearest neighbours

### Künstliche Neuronale Netze (teilweise auch Klassifikatoren?)

wäre gut, das von den anderen Klassifizierern zu trennen

#### MultiLayerPerceptrons

#### PNN

#### PCNN

# Konzeption

## Anforderungsanalyse

### Kriterien des geplanten Produkts

Muss/Kann/Abgrenzungskriterien

### Produkteinsatz

Möglichkeiten der Anwendung

Zielgruppe

### Produktumgebung

technische Anforderungen

## Vorhandene Lösungen

kollidiert mit Grundlagen!!!

### Überblick

Wie in Kapitel 2.2 bereits erwähnt, sind verschiedene Arbeitsschritte notwendig, um die Ausgangsdaten für die vorzubereiten.

Um die Blätter einheitlich beschreiben zu können, müssen die Deskriptoren unabhängig von Skalierung, Rotation und ? beschreiben. Chaki, Parekh und Bhattacharya (2015) schlagen vor das Blatt zuvor zu rotieren.

Datenbanken für Blätter

Einen interessanten Ansatz bietet

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Autoren | preprocessing | Deskriptoren | Klassifikatoren | Datensatz | Schluss |
| (Chaki, Parekh, & Bhattacharya, 2015) | - Graustufen, Binarisierung, Kontur  - Blatt rotiert, so dass Hauptachse ausgerichtet zu horizontaler Achse  - Beschneidung auf BB-Größe, Skalierung auf einheitl. Auflösung (7 „slots“ mit max. 300 px) | -Textur: Gabor-Filter, Gray level co-occurrencematrix  -Form: Curvelet-Transform, Hu-Momente  - komibinierter Featurevektor aus beiden | -MLP  -neuro fuzzy classifier | -Flavia |  |
| (Du, Wang, & Zhang, n.d.) | Graustufen, Binarisierung, Kontur, Region of Interest | geometrische features, invariante Features (Hu-Momente) | moving median center hyperspheres (MMC), KNN | ? |  |
| (Han, Chee, Chan, Wilkin, & Remagnino, n.d.) | ? | unüberwachte Features ermittelt durch convolutional neural networks (CNN) | multi-layered deconvolutional network | Royal Botanic Gardens Kew |  |
| (Kalyoncu & Toygar, 2015) | Segmentierung, noise reduction, Kontur, Glättung and corner region detection | invariante Momente, Multi-scale distance matrix, margin distance (Distanz zwischen Punkten des Original-Blattrands und des geglätteten Rands) | Linear Discriminant Classifier | Flavia, Leafsnap |  |
| (Kumar et al., 2012) | Farbbasierte Segmentierung, | Curvature-based Shape Features, | Nearest Neighbours | Leafsnap |  |
| (Wang, Sun, Zhang, Ying, & Ma, 2016) | ? | PCNN, Hu-Momente, Zernike-Momente | SVM | Intelligent Computing Lab dataset |  |
| (Reyes, Caicedo, & Camargo, n.d.) | ? | fine-tuned CNN | CNN | ImageCLEF |  |
| (Satti, Satya, & Sharma, 2013) | Binarisierung, Kontur, Glättung | Farbe  Form: Geometrie, Morphologische Features (Adern, Glättungsfaktor …), Zahnung (über Winkel von Randpunkten) | KNN, ANN (keine Angabe welches) | Flavia | ANN schneller und genauer als KNN |
| (Cao, Wang, & Brown, 2016) | Binarisierung, Kontur, Beschreibung durch gleichmäßig verteilte Punkte | R-Angles (Skalierungsinv. über 7 Stufen) | L1-Norm, Punkt zu Punkt Dynamic Programming | Flavia | hohe Genauigkeit (speziell mit DP) |
|  |  |  |  |  |  |

### Form

R-Angles [Similarity based leaf image retrieval using multiscale R-angle description]

FFT

### Blatttextur

-zB Entropy, Kontrast etc. (Shabanzade, Zahedi, & Amin Aghvami, 2011) Extraktion der Blattadern

### Fourier-Transformation

## Lösungsansatz

### Datensatz

Es sind Datenbanken verfügbar, die unter freier Lizenz zum Training für die automatische Bestimmung genutzt werden können. Unter anderem veröffentlichten Wu, Bao, Xu, Wang, Chang und Xiang (2007) den Flavia-Datensatz, den sie im Rahmen der Entwicklung eines eigenen Algorithmus zur Blatterkennung erstellten. Der Datensatz beinhaltet 1800 Bilder von Blättern 32 unterschiedlicher, hauptsächlich chinesischer Arten. Die Bilder liegen vorverarbeitet vor. Die Blätter sind bereits freigestellt und die Blattstiele wurden entfernt (siehe Abbildung 10a).

Der sogenannte Leafsnap-Datensatz entstand in Zusammenarbeit der Columbia University, der University of Maryland und der Smithsonian Institution. Kumar, Belhumeur, Biswas, Jacobs, Kress, Lopez und Soares (2012) schufen die Grundlagen zur Entwicklung eines elektronischen Feldführers für Smartphones. Die zugrundeliegenden Trainingsdaten umfassen Fotografien von nach eigener Aussage allen 185 in den nordöstlichen USA beheimateten Baumarten. Die Bilder sind sowohl in ihrer ursprünglichen Form, als auch in segmentierter, binarisierter Form erhältlich (siehe Abbildung 10b).



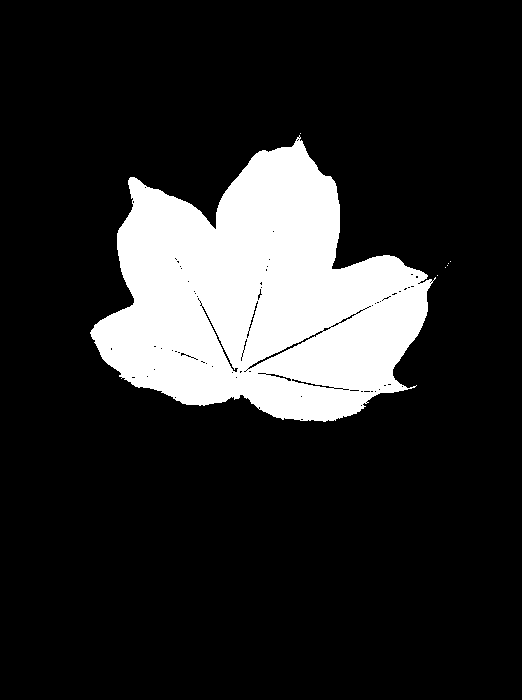


Abb. 10a: Beispielfoto aus der Flavia-Datenbank Abb. 10b: unbearbeitetes (links) und segmentier-

(Wu, Bao, Xu, Wang, Chang & Xiang, 2007) tes Blatt (rechts) aus dem Leafsnap-Datensatz

(Kumar, Belhumeur, Biswas, Jacobs, Kress, Lopez & Soares, 2012)

Den beiden Datensätzen ist gemein, dass nur Fotografien von den Blattoberseiten aufgenommen wurden. Es soll in dieser Arbeit aber auch geprüft werden, ob sich die Blattunterseite besser eignet, um bestimmte Merkmale, im Besonderen die Blattadern, erkennen und trainieren zu können.

Daher wurde für die Bestimmung zusätzlich ein eigener Datensatz erstellt. Er enthält Daten 25 verschiedener Spezies mit jeweils 40 Fotografien je Ober- und Unterseite. Bei den Arten handelt es sich hauptsächlich um in Deutschland beheimatete Pflanzen. In Abbildung 11 ist ein Auszug aus den Bilddaten zu sehen. *(verweisen auf Anhang, Tabelle der Spezies)*



Abb. 11: Ober- (links) und Unterseite (rechts) eines Klarapfels (Malus Domestica)

Alle genannten Datensätze sollen beim Training des zu implementierenden Algorithmus Verwendung finden und die Ergebnisse der Erkennung verglichen werden.

AUFLÖSUNG! mehr Details

### Segmentierung der Blätter

Die Blätter des selbsterstellten Datensatzes müssen zunächst vom Hintergrund freigestellt werden. Auch die Bilder des Leafsnap-Algorithmus von Kumar et. al. (2012) sollten vor der weiteren Verwendung erneut segmentiert werden. Neben den nicht segmentierten Bildern sind lediglich binarisierte Daten vorhanden (siehe Abbildung 10b). Diesen fehlen notwendige Informationen der Blatttextur, da sich die Autoren auf die Blattkontur konzentrieren und die Textur dafür nicht benötigen.

Kumar et. al. (2012) benutzen in ihrer Arbeit einen eigenen Algorithmus einer farbbasierten Segmentierung. Sie führen an, dass kanten- und regionsbasierte Verfahren gegenüber ihrem Verfahren langsamer arbeiten und Details der Blattränder verloren gehen können.

Wie in Abbildung 11 beispielhaft zu sehen, ist der Hintergrund im selbsterstellten Bildmaterial aber hell und gleichmäßig ausgeleuchtet, so dass sich die Blätter deutlich von ihm abheben. Dies bedeutet, dass sich auch in einem umgewandelten Graustufenbild zwei deutlich voneinander getrennte Maxima zwischen Vorder- und Hintergrund im Histogramm ergeben. Es wird angenommen, dass deshalb ein Schwellwert-Algorithmus basierend auf den Graustufen genügt, um das Blatt freizustellen ohne Details der Kontur zu verlieren. Zu diesem Zweck wird die Otsu-Methode verwendet, die mit bimodaler Graustufenverteilung gute automatisierte Schwellwerte liefert *(Quelle!)*.

Es muss allerdings getestet werden, ob diese Methode auch mit den Leafsnap-Bildern fehlerfrei arbeitet, da der Datensatz Bilder mit ungenügender Ausleuchtung und somit geringem Kontrast zwischen Vorder- und Hintergrund enthält *(Beispielbilder?)*.

### Wahl und Extraktion der Features

invariant weil?

rotation

scale - verschiedene Winkel

Form und Textur möglichst getrennt betrachten

### Blattform

Wie in Kapitel 2.1.2.1 angeführt, ist die Form der Blätter in der Botanik ein wichtiges Unterscheidungsmerkmal für die Bestimmung einer Pflanze. Daher soll sie auch bei der automatisierten Bestimmung eine besondere Beachtung finden.

Zur Beschreibung der Geometrie der Blätter eignen sich Momente. Es kann nicht davon ausgegangen werden kann, dass die als Ausgangsmaterial dienenden Blätter stets aus dem gleichen Winkel mit derselben Ausrichtung und Entfernung fotografiert werden. Daher müssen die beschreibenden Momente rotations-, skalierungs- und translationsinvariant gewählt werden. Dazu kommen die in Kapitel 2.3.2 erwähnten Hu-Momente in Betracht.

Daneben gibt es weitere invariante Momente, die zur Beschreibung in Frage kommen. Zulkifli (2009) vergleicht in seiner Arbeit Zernike-Momente, Legendre-Momente und Tchebichef-Momente als Eingangsdaten für ein neuronales Netz *(Spezialfall erwähnen: GENERAL REGRESSION NEURAL NETWORK*)*.* Er kommt zu dem Schluss, dass von den drei Verfahren die Tchebichef-Momente die geringste Fehlerrate bei der Erkennung aufweisen.

Aus diesem Grund werden sowohl Hu-Momente, als auch Tchebichef-Momente zum Einsatz kommen und deren Erkennungsgenauigkeit verglichen werden.

*… (Kontur?)*

#### Momente

Hu, Zernike

Zernike moments are not a texture feature, but rather a global measure of how the mass is distributed

#### Shape Context

#### Blattadern

### Blattadern

Texturbeschreibung

#### Local binary pattern

#### Blattadern

Als Schwerpunkt dieser Arbeit soll die Eignung der Struktur der Blattadern zur Erkennung der Pflanzenspezies überprüft werden.

Zur Klassifikation von Leguminosen nutzen Larese, Bayá, Craviotto, Arango, Gallo und Granitto (2014) in ihrer Abhandlung ausschliesslich die Blattadern. Als Besonderheit können sie durch ihre angebrachte Methode auch verschiedene Variationen innerhalb einer Art unterscheiden. Zur Segmentierung der Adern nutzen sie eine erweiterte Form der Hit or Miss Transform *(erklären!)* auf den Graustufenbildern. Sie wird für verschiedene Skalierungen durchgeführt, um eine skalierungsinvariante Beschreibung zu erhalten. Als Features dienen unter anderen die Anzahl der Kanten, Knoten und die Winkel der Kanten. Allerdings sind speziell die Winkel nicht rotationsinvariant. Aus diesem Grund müssen die Blätter bereits in der Vorverarbeitung entlang ihrer Längsachse gleich ausgerichtet werden.

Alternativ können die Blattadern auch durch einen Gabor-Filter segmentiert werden, wie es Katyal (2012) vorschlägt. *(mehr)* Seine Arbeit konzentriert sich auf die Freistellung der Adern, beinhaltet aber keine Methode zur Extraktion etwaiger Features.

Vorschlag für Features:

* Zahl der starken Kanten (Adern mit größter Breite, wie differenzieren?)
* Zahl der Areolen
* Volumen der Adern (Anzahl Pixel im Verhältnis Volumen Blatt?)
* Länge (Pixelzahl im Verhältnis Kontur?)
* wie Winkel rotationsunabhängig beschreiben? im Verhältnis zur Hauptachse? (ist das nicht immer gleich?)

(Larese et al., 2014)

* Feature: 100 x 100 Patch für jede Skalierung (um Shape-Einfluss rauszunehmen)

52 traits (Tabelle) VMeO/VmO/VMO Nicht rotationsinvariant!

* sKlassifikation: Random Forests, Support Vector Machines with Gaussian kernel und Penalized Discriminant Analysis (linear nach Fisher)
* Recursive Feature Elimination (RFE)

Röntgenbilder stehen nicht zur Verfügung

* Nur Farbbild, Röntgenbilder besser, wird auch teilweise so angewendet (Areolenbestimmung)

### PCNN?

pcnn - pulse-coupled neural networks (Wang et al., 2016)

* 2D-neuronales Netz basierend auf visuellem Cortex von Katzen
* selbstständiges Finden von relevanten Features
* jedes Neuron repräsentiert Farbinfo eines Pixels
* Neuronen sind mit Nachbarn verbunden
* Puls setzt Stimulus zurück

### Klassifizierung

probabilistisches neuronales Netz (MLP)

wenn zu langsam PNN

wenn nicht zufriedenstellend irgendwas anderes

support vector machine

### Metrik zur Messung der Erkennungsgenauigkeit

## Systementwurf

### Systemkomponenten

keine blosse Abfolge von Skripten, sonder strukturiert, damit austauschbare Komponenten (Interface, abstrakte Klassen)

### Datenbank?

Warum Datenbank?

* zentraler Zugriff
  + vereinfachter Zugriff über Netzwerk
  + Übersicht (besser als lauter Dateien irgendwo)
  + besser als HDF5, da immer ganz gelesen wird
* Option: pandas, LMDB

„Sacred“ to keep track? <http://sacred.readthedocs.io/>

HDF5:

* effiziente Serialisierung/Speicherung von numpy arrays
* cross languages
* common dataformat in science.
* einfacher physischer Austausch der Daten
* Nachteil: nur Arrays(?)

hdf5 vs pickle (Serialisierung): - pickle kann nur aus python gelesen werden, kein zukunftssicherer Standard(?), kleinere Dateien,

warum SQL

* strukturiert (besser für Auswertung der Ergebnisse, übersichtlicher)
* Datenmenge überschaubar (große Datenmengen -> besser NOSQL)
* Daten gleichförmig?

NOSQL

* Skalierbarkeit horizontal (auf Rechner verteilen)

# Implementierung

### Verwendete Bibliotheken

Unittests, Oberfläch, nicht Java sondern Python

OpenCV Konturpunkte, Binarisierung

skimage und opnecv, eigentlich alle benutzen numpy arrays als represäntation von Bildern

### Testumgebung

Oberfläche mit PyQt

### allgemein

alle Features erwarten segmentiertes Bild, alle Segmentierer liefern segmentiertes Bild (Vereinheitlichung)

Segmentierte Bilder, Hintergrund weiß (abgeguckt von allen anderen)

einfache Binarisierung: da Hintergrund mit weiß codiert, gleich schwarz setzen (Binärbild Codierung!) und Rest auf 1 bzw. 255

Achtung: jpg macht Probleme! (irgendwo anders hier)

Feature Klasse übernimmt Vorbereitung des Bildes, delegiert an Unterklassen, wo eigentliche Berechnungen stattfinden

### Segmentierung

Ausgang immer maskiertes Bild

Die Segmentierung mittels Otsu-Thresholding wurde mit Hilfe der OpenCV-Bibliothek umgesetzt. Zunächst wird das Ausgangsbild im RGB-Farbraum eingelesen und in ein Graustufenbild umgewandelt. ??? Allerdings war das Ergebnis entgegen der Vermutung aus Kapitel unzureichend für eine anschließende Weiterverarbeitung. In einigen Bildern verfälschen lange Schatten die Kontur des binarisierten Blattes. Da bei der Aufnahme des Testdatensatzes teilweise ein Blitzlicht verwendet wurde, sind die vom Blatt geworfenen Schatten sehr dunkel. Da auch das Blatt dunkler ist als der Hintergrund, werden die Pixel der Schatten dem Vordergrund zugeordnet. (siehe Bilder)

Da die Fotos der Blätter vor einem neutralen farbarmen Hintergrund aufgenommen wurden, lassen sich die Blätter vor allem in der Sättigung von diesem unterscheiden, der eigentliche Farbwert ist vernachlässigbar. Somit ist der HSV-Farbraum, der die Helligkeit von der Farbsättigung und dem Farbwert trennt, für die Freistellung der farbigen Blätter besser geeignet als der RGB-Farbraum (siehe Abbildung). Ein Otsu-Thresholding des Sättigungsanteils trennt in einer Vielzahl der Bilder

Warum kmeans? Bilder raussuchen

Es ist also notwendig

Zudem fransen die schattigen Ränder der Blätter aus.

Löcher schließen, Löcher sollen Merkmal der Textur, aber nicht der Form sein

Hintergrund eingefärbt vom Umgebungslicht. Einige Fotos in Dämmerung geschossen, farbiges Licht

Auch dies ergab

value dazu

Bestehendes Problem: farbige Schatten. Die Schatten sind hauptächlich grau mit einer geringen Sättigung. zwei Lichtquellen: Umgebungsbeleuchtung, Blitz

Da aber auf einigen Es verbleiben dennoch farbige Schatten,

Generell erfolgt vor allen Segmentierungen, bei denen Faltungen mit festen Kernelgrößen vorgenommen werden, zunächst eine Skalierung des Bildes auf eine vordefinierte Größe. So wird verhindert, dass die strukturierenden Elemente unterschiedlichen Einfluss haben (Erosion, Dilation).

kein Gauss, damit Kontur scharf bleibt (-> Segmentation verursacht deswegen Löcher)

DSC\_6982.JPG, DSC\_6813.JPG - Beispiel dafür, dass einfaches Otsu und k-means nicht funktionieren

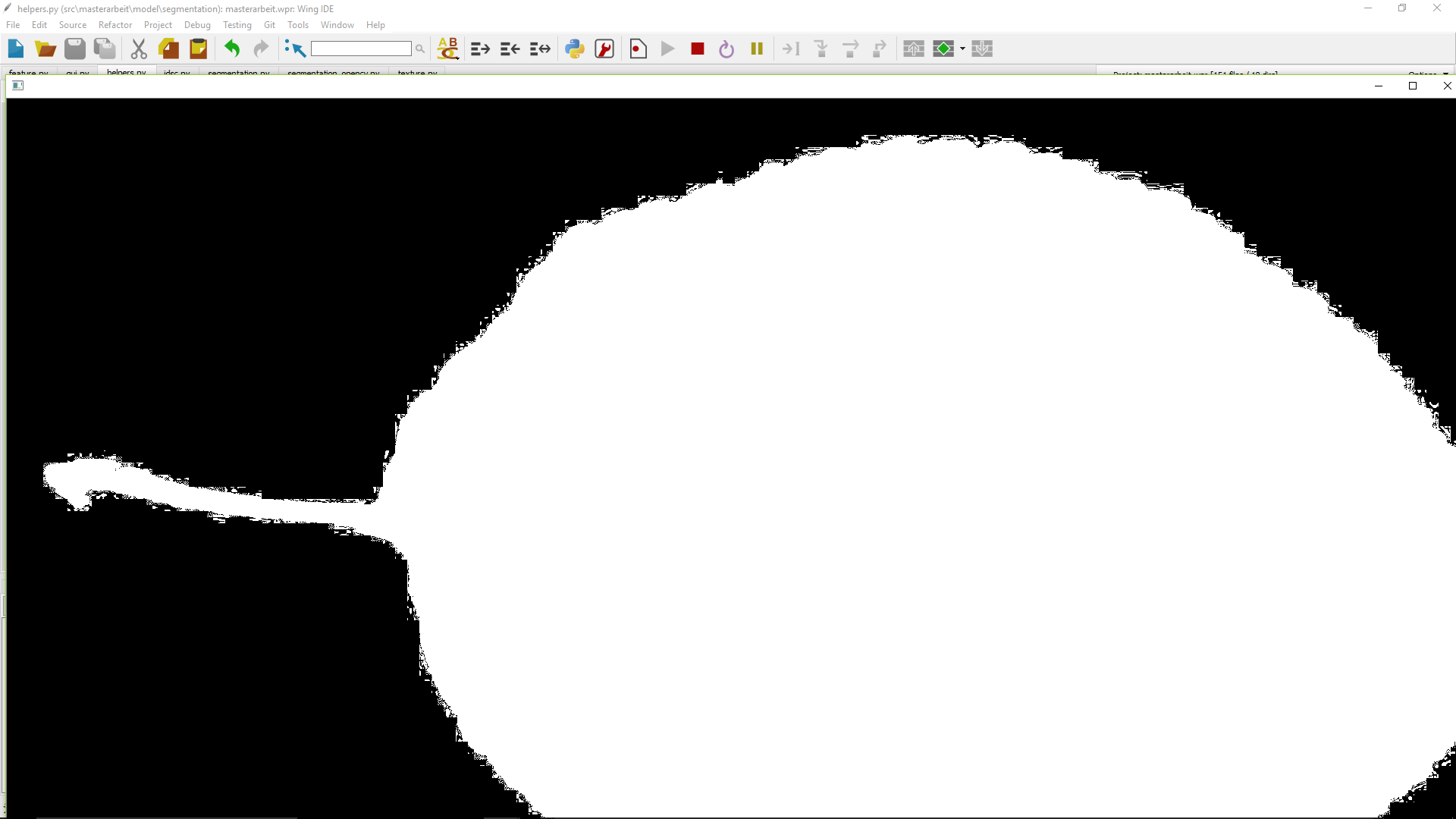
H - viel noise

S + V - wenig noise, bisschen harte schatten

nur S - weniger schatten, noisy, da fragmente der schatten

entfernung stiel entfernt (vielleicht bezug auf Plant identification using leaf shapes—A pattern counting approach)

normalisieren, da nur erhältnis der werte zueinander entscheidend, nicht höhe der werte (wie oft taucht pattern x auf im verh. zu allen anderen)



Shape allgemein: immer längste kontur nehmen, falls mehrere Konturen, kleine Objekte bei Segmentierung zurückbleiben

### Inner Distance Shape Context

Für Python sind keine Bibliotheken verfügbar, die Methoden zur Extraktion des Inner Distance Shape Context bereitstellen. Daher wird der Algorithmus von ??? selbst implementiert.

Ausgangspunkt für die Berechnung jedweder Shape Contexts ist die aus einem Binärbild ermittelte Kontur. Dieser wird eine vordefinierte Anzahl an Punkten, die in regelmäßigen Abständen verteilt sind, entnommen. Keine besonderen Punkte

Die euklidischen Distanzen zwischen allen entnommenen Punkten werden berechnet und in einer mit Nullen initialisierten Distanzmatrix gehalten. Abweichend vom ursprünglichen Konzept des Shape Contexts, werden allerdings nur innere Distanzen berücksichtigt. Dies bedeutet, dass die direkte Linie zwischen zwei betrachteten Punkten komplett innerhalb der Form liegen muss, damit sie innerhalb der Form als verbunden gelten können und ihre Distanz eingetragen wird. Da der Hintergrund im Binärbild mit Nullen codiert ist, wird dazu programmatisch überprüft, ob Zwischenpunkte auf dieser Linie nur Werte größer Null annehmen. Ist dies der Fall, wird die Distanz eingetragen. Die verbleibenden Nullen geben an, dass keine Verbindung zwischen den jeweilig indizierten Punkten besteht.

Die Distanzmatrix bildet den Eingangsgraphen für die Ermittlung der kürzesten Wege zwischen den Punkten der Kontur, wobei die euklidische Distanz die Kantenlänge angibt. Über den Floyd-Warshall-Algorithmus wird eine Matrix erzeugt (scipy), die paarweise die kürzesten Wege enthält. Jeder kürzeste Weg ist die minimale Summe der Länge der Kanten im ungerichteten Graph, die abgelaufen werden müssen, um den Punkt zu erreichen. Gibt es keine Kante zwischen zwei Punkten, ist der kürzeste Weg, wie in der Distanzmatrix, gleich Null, beziehungsweise unendlich.

Pseudocode

Für jeden aus der Kontur entnommenen Punkte wird anschließend ein Histogramm erstellt. Das Histogramm beschreibt das Verhältnis zu allen anderen Punkten im Hinblick auf die relative Orientierung und die Länge des kürzesten Weges zu ihnen. Die relative Orientierung wird als Winkel zwischen der an der Kontur anliegenden Tangente am betrachteten Punkt und dem Zielpunkt definiert.

Für die Zuordnung der Punkte in das Histogramm werden die Wertebereiche der Winkel und Wegelängen logarithmisch in jeweils acht Klassen unterteilt. Der Wertebereich der Winkel liegt zwischen 0 und 2 • π, der der Wegelängen zwischen 0 und dem Logarithmus der maximalen Distanz zur Basis zwei. Als maximale Distanz wird die Diagonale des Ausgangsbildes angenommen. entspricht der Vorlage

Jeder nicht erreichbare Punkt wird der Histogramm-Klasse, zugeordnet, die alle Punkte enthält, die im Hinblick auf kürzeste Wege den größten Abstand zum betrachteten Punkt haben. Somit ist das Histogramm jedes Punktes vollständig beschreibend, die Summe seiner Werte entspricht immer der Anzahl der entnommenen Punkte minus den betrachteten Punkt. Um den Wertebereich des Histogramms unabhängig von der Anzahl der Punkte zu halten, wird die L1-Norm, also die Summennorm, darauf angewendet.

Histogrammformel hier hin

Um die Konturen von Objekten anhand ihrer ermittelten Inner Distance Shape Contexts zu vergleichen und zu klassifizieren, verwenden bla et al, dynamische Programmierung. Da die Histogramme nicht geordnet, irgendwas Dies eignet sich nicht für die angestrebte Lösung, die Klassifizierung mit neuronalen Netzen oder Support Vector Machines durchzuführen.

Die erzeugten Histogramme können aber als sich wiederholende Muster gleicher Dimension angenommen werden. Diese Muster können als einzelne Fragmente für ein Bag-of-words-Modell dienen und zu visuellen Wörtern zusammengefügt werden, die wiederum die Form beschreiben. Diese Aufgabe übernehmen die in Kapitel ??? beschriebenen Codebooks.

Die einzelnen Histogramme mit einer Dimension von 8 x 8 werden konkateniert, so dass das Endergebnis der Feature-Extraktion eine Matrix der Dimension Anzahl der Punkte x 8 x 8 ist.

### Gaussian Pyramid Inner Distance Shape Context

Angelehnt an die Idee des Independent Inner Distance Shape Context von ??? wurde der im vorigen Kapitel beschriebene Algorithmus erweitert, aus der Motivation heraus, die feinen Details der Konturen unabhängig von der Gesamtkontur beschreiben zu können.

Wie auch beim Independent Inner Distance Shape Context, wird zunächst eine Gauß-Pyramide des binarisierten Bildes gebildet. Auf der niedrigsten, der feinsten Stufe, bleibt das Ausgangsbild erhalten, um alle Details abzubilden. Auf den weiteren Stufen wird das Ausgangsbild mit einem Gauß-Filter *g(x,y)* wie in (1) gefaltet, und das Bild somit geglättet. Zwischen den Stufen steigt *σ* stetig an, so dass auch der Grad der Glättung stetig ansteigt und die Form des Objektes so an Details verliert. Konkret wird *σ*, beginnend mit *σ = 8* auf der zweiten Stufe, von Stufe zu Stufe verachtfacht. Die Auflösung der geglätteten Bilder bleibt auf allen Stufen ist identisch, damit auch die maximalen Distanzen vergleichbar bleiben.

(1)

mean

Die maximalen Distanzen bestimmen hier zusätzlich die Größe der Nachbarschaft eines Punktes in Bezug auf die maximale Länge der kürzesten Wege. Auf dem feinsten Detailgrad, also der niedrigsten Stufe der Gauß-Pyramide, wird nur die unmittelbare Nachbarschaft betrachtet, die maximale Distanz muss dementsprechend klein sein. Auf dem gröbsten Detailgrad sollen alle Punkte der Kontur als benachbart angenommen werden, die maximale Distanz ist am größten.

Die geglätteten Konturen der Bilder der Gauß-Pyramide dienen als Eingangspunkte des Algorithmus des Inner Distance Shape Context (Kapitel ..). Der Algorithmus wurde dahingehend modifiziert, dass Punkte außerhalb der maximalen Distanz ignoriert werden, anstatt sie der letzten Histogramm-Klasse zuzuordnen. Das Histogramm ist somit nicht vollständig beschreibend, sondern beschreibt nur die Nachbarschaft der Punkte.

Abweichend vom Independent Inner Distance Shape Context von … werden die Stufen der Gauß-Pyramide unabhängig voneinander behandelt. Die Konturpunkte zwischen den Stufen sind nicht deckungsgleich. Stattdessen werden mehr Punkte betrachtet, je feiner der Detailgrad ist. Der Grund dafür ist, dass die Gesamtzahl an Konturpunkten erhöht werden muss, damit für eine detaillierte Beschreibung einer kleinen Nachbarschaft genug Punkte in diesem Ausschnitt der Kontur zur Verfügung stehen.

Die resultierenden Histogramme der verschiedenen Stufen sind als visuelle Wortfragmente, aufgrund der verschieden definierten Nachbarschaften, nicht direkt miteinander kombinierbar. Daher wird für jede Stufe ein eigenes Codebook benötigt.

### Codebooks

Die Klassifizierer benötigen als Eingangsdaten Feature-Vektoren, die statistische Eigenschaften eines Objektes in einer festen Reihenfolge beschreiben. Um ungeordnete Features zu beschreiben, können diese als Fragmente visueller Wörter aufgefasst werden. Über die Häufigkeit des Auftretens der visuellen Wörter, können Histogramme gebildet werden, welche dann wiederum als Feature-Vektoren nutzbar sind. Die Aufgabe der Bestimmung geeigneter visueller Codewörter übernehmen die im Folgenden beschriebenen Codebooks.

### Sparse dictionary learning

Speziell für die Umwandlung der Histogramme des Inner Distance Shape Context wurde ein Wörterbuch für das sogenannte „sparse dictionary learning“ umgesetzt. Die Aufgabe des Wörterbuchs ist es, die Menge der extrahierten Shape-Context-Vektoren zu reduzieren, indem diese in ihre Atome zerlegt werden. Mit Atomen werden die Linearkombinationen bezeichnet, aus denen sich die Vektoren komponieren lassen.

Löst das Problem der Formel

Vorteil: online dictionary (wird nicht benutzt), Dictionary kann erweitert werden

Angelehnt an die Arbeit von Zhao et al. werden die Atome als sich wiederholende Muster angenommen und in einem Histogramm gezählt. Abweichend nicht darauf geachtet, dass nur bestimmte Zahl auftauchen darf.

Die Repräsentation ist übererfüllt,

Dictionary. KMeans zusätzlich

erwartete Dimension

### Multilayer Perceptron

### Support Vector Machine

### Datenhaltung

# Ergebnis

local binary pattern keine Aussagekraft, auch in Kombination nicht

weder

Annahme: Kombination besser, da jedes für sich nicht umfassend beschreibend

## Genauigkeit (welche Metrik?)

Vergleich verschiedene Blattformen

Vergleich steigende Anzahl an Klassen

Vergleich Vorder/Rückseite

Vergleich steigende Anzahl an Trainingsblättern (Erkennungsrate)

Kombination der Features

## Aufwandsanalyse?

# Zusammenfassung

## Ausblick

* Gleiche Spezies untersuchen – verschiedene Sorten (möglicherweise gleiche Blattform, aber Unterschiede in Blattadern), im Datenset nur Apfel (?)
* bugs/usability in Oberfläche (war für Zwecke hier egal9
* Smartphone App

Optimierung IDSC: Punkte auf Linie immer 10, kein check für unique Punkte(wenn 10 zuviel)

Performance generell (in C schreiben, python zu langsam, opencv interface?)

Textur integrieren in IDSC (SPT)

Trainingsfotos: Beleuchtung, eine diffuse, unfarbige Lichtquelle, DSC\_5821

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Codelistings

1. Literaturverzeichnis

Cao, J., Wang, B., & Brown, D. (2016). Similarity based leaf image retrieval using multiscale R-angle description. *Information Sciences*, *374*, 51–64. https://doi.org/10.1016/j.ins.2016.09.023

Chaki, J., Parekh, R., & Bhattacharya, S. (2015). Plant leaf recognition using texture and shape features with neural classifiers. *Pattern Recognition Letters*, *58*, 61–68. https://doi.org/10.1016/j.patrec.2015.02.010

Du, J.-X., Wang, X.-F., & Zhang, G.-J. (n.d.). Leaf shape based plant species recognition. https://doi.org/10.1016/j.amc.2006.07.072

Han, S., Chee, L., Chan, S., Wilkin, P., & Remagnino, P. (n.d.). DEEP-PLANT: PLANT IDENTIFICATION WITH CONVOLUTIONAL NEURAL NETWORKS.

Kalyoncu, C., & Toygar, Ö. (2015). Geometric leaf classification. *Computer Vision and Image Understanding*, *133*, 102–109. https://doi.org/10.1016/j.cviu.2014.11.001

Katyal, V. (2012). *Leaf vein segmentation using Odd Gabor filters and morphological operations*. Amity University Indien.

Kumar, N., Belhumeur, P. N., Biswas, A., Jacobs, D. W., Kress, W. J., Lopez, I., & Soares, J. V. B. (2012). Leafsnap: A Computer Vision System for Automatic Plant Species Identification. *Proceedings of the 12th European Conference on Computer Vision (ECCV)*.

Larese, M. G., Bayá, A. E., Craviotto, R. M., Arango, M. R., Gallo, C., & Granitto, P. M. (2014). Multiscale recognition of legume varieties based on leaf venation images. *Expert Systems with Applications*, *41*(10), 4638–4647. https://doi.org/10.1016/j.eswa.2014.01.029

Reyes, A. K., Caicedo, J. C., & Camargo, J. E. (n.d.). Fine-tuning Deep Convolutional Networks for Plant Recognition.

Satti, V., Satya, A., & Sharma, S. (2013). AN AUTOMATIC LEAF RECOGNITION SYSTEM FOR PLANT IDENTIFICATION USING MACHINE VISION TECHNOLOGY. *International Journal of Engineering Science and Technology*, *5*(4), 975–5462.

Shabanzade, M., Zahedi, M., & Amin Aghvami, S. (2011). Combination of Local Descriptors and Global Features for Leaf Recognition. *Signal & Image Processing : An International Journal*, *2*(3), 23–31. https://doi.org/10.5121/sipij.2011.2303

Wang, Z., Sun, X., Zhang, Y., Ying, Z., & Ma, Y. (2016). Leaf recognition based on PCNN. *Neural Computing and Applications*. https://doi.org/10.1007/s00521-015-1904-1

Wu, S. G., Bao, F. S., Xu, E. Y., Wang, Y. X., Chang, Y. F., & Xiang, Q. L. (2007). A leaf recognition algorithm for plant classification using probabilistic neural network. *ISSPIT 2007 - 2007 IEEE International Symposium on Signal Processing and Information Technology*, (December), 11–16. https://doi.org/10.1109/ISSPIT.2007.4458016

Zhao, C., Chan, S. S. F., Cham, W.-K., & Chu, L. M. (2015). Plant identification using leaf shapes—A pattern counting approach. *Pattern Recognition*, *48*(10), 3203–3215. https://doi.org/10.1016/j.patcog.2015.04.004

Zulkifli, Z. B. (2009). *PLANT LEAF IDENTIFICATION USING MOMENT INVARIANTS & GENERAL REGRESSION NEURAL NETWORK*. Faculty of Computer Science and Information Systems.

1. Anhang

## Installationsanleitung

-theano.orc (Windows: unter %USERPROFILE%):

[*global]*

*device =* gpu

- CUDA Toolkit installieren

u.U in theano.orc (wenn cl.exe nicht gefunden oder Value '2008' is not defined for option 'cl-version')

[*nvcc]*

*compiler\_bindir=E:\Programme\Microsoft Visual Studio 12.0\VC\bin*

utf-8 not supported: in theano.compat ändern (fett):

def decode\_iter(itr):

for x in itr:

yield x.decode(**'gbk'**)

header-Datei fehlt: in das angegebene Verzeichnis kopieren