

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](#_Toc475659187)

[2. Grundlagen 1](#_Toc475659188)

[2.1. Morphologie von Pflanzen 1](#_Toc475659189)

[2.1.1. Bestimmungsmerkmale höherer Pflanzen 1](#_Toc475659190)

[2.1.2. Morphologie des Blattes 2](#_Toc475659191)

[2.1.2.1. Funktion und Aufbau des Blattes 2](#_Toc475659192)

[2.1.2.2. Blattformen 3](#_Toc475659193)

[2.1.2.3. Blattnervatur 3](#_Toc475659194)

[2.2. Digitale Bildverarbeitung 4](#_Toc475659195)

[2.2.1. Diskrete Bildrepräsentation 4](#_Toc475659196)

[2.2.2. Binarisierung 5](#_Toc475659197)

[2.2.3. Filter 7](#_Toc475659198)

[2.2.3.1. Allgemeine Definition 7](#_Toc475659199)

[2.2.3.2. Lineare Filter 8](#_Toc475659200)

[2.2.3.3. Kantenerkennung 10](#_Toc475659201)

[2.2.3.4. Morphologische Filter 10](#_Toc475659202)

[2.2.4. Filter 10](#_Toc475659203)

[2.3. Maschinelles Lernen 10](#_Toc475659204)

[2.3.1. Grundlagen 10](#_Toc475659205)

[2.3.2. Features 10](#_Toc475659206)

[2.3.2.1. Momentenbasierte geometrische Merkmale 10](#_Toc475659207)

[2.3.3. Fourier-Transformation 10](#_Toc475659208)

[2.3.4. Hauptkomponentenanalyse 10](#_Toc475659209)

[2.4. Maschinelle Lernverfahren 10](#_Toc475659210)

[2.4.1. Klassifikatoren 10](#_Toc475659211)

[2.4.1.1. MMC Hypersphere??? 10](#_Toc475659212)

[2.4.1.2. Support Vector Machine 10](#_Toc475659213)

[2.4.1.3. k-nearest neighbours 10](#_Toc475659214)

[2.4.2. Künstliche Neuronale Netze (teilweise auch Klassifikatoren?) 11](#_Toc475659215)

[2.4.2.1. MultiLayerPerceptrons 11](#_Toc475659216)

[2.4.2.2. PNN 11](#_Toc475659217)

[2.4.2.3. PCNN 11](#_Toc475659218)

[3. Konzeption 11](#_Toc475659219)

[3.1. Zielstellung 11](#_Toc475659220)

[3.2. Testumgebung 11](#_Toc475659221)

[3.3. Vorhandene Lösungen (oder Einordnung in Kontext oder so) 12](#_Toc475659222)

[3.3.1. Überblick 12](#_Toc475659223)

[3.3.2. Form 13](#_Toc475659224)

[3.3.3. Blatttextur 13](#_Toc475659225)

[3.3.4. Fourier-Transformation 13](#_Toc475659226)

[3.4. Lösungsansatz 14](#_Toc475659227)

[3.4.1. Datensatz 14](#_Toc475659228)

[3.4.2. Segmentierung der Blätter 15](#_Toc475659229)

[3.5. Wahl und Extraktion der Features 16](#_Toc475659230)

[3.5.1. Blattform 17](#_Toc475659231)

[3.5.1.1. Momente 17](#_Toc475659232)

[3.5.1.2. Shape Context 17](#_Toc475659233)

[3.5.1.3. Blattadern 17](#_Toc475659234)

[3.5.2. Blattadern 17](#_Toc475659235)

[3.5.2.1. Local binary pattern 18](#_Toc475659236)

[3.5.3. Blattadern 18](#_Toc475659237)

[3.5.4. Lernverfahren 19](#_Toc475659238)

[3.5.5. Klassifizierung 19](#_Toc475659239)

[3.5.6. Metriken zur Messung der Erkennungsgenauigkeit 19](#_Toc475659240)

[3.6. Systementwurf 19](#_Toc475659241)

[3.6.1. Systemkomponenten 19](#_Toc475659242)

[3.6.2. Datenbank? 20](#_Toc475659243)

[4. Implementierung 20](#_Toc475659244)

[4.1. Verwendete Bibliotheken 20](#_Toc475659245)

[4.2. allgemein 21](#_Toc475659246)

[4.3. Segmentierung 21](#_Toc475659247)

[4.4. Inner Distance Shape Context 23](#_Toc475659248)

[4.5. Multilevel Inner Distance Shape Context 25](#_Toc475659249)

[4.6. Codebooks 26](#_Toc475659250)

[4.6.1. Sparse dictionary learning 26](#_Toc475659251)

[4.7. Multilayer Perceptron 27](#_Toc475659252)

[4.8. Support Vector Machine 27](#_Toc475659253)

[4.9. Datenhaltung 27](#_Toc475659254)

[5. Ergebnis 27](#_Toc475659255)

[5.1. Genauigkeit (welche Metrik?) 27](#_Toc475659256)

[5.2. Aufwandsanalyse? 27](#_Toc475659257)

[6. Zusammenfassung 28](#_Toc475659258)

[6.1. Ausblick 28](#_Toc475659259)

[I. Abbildungsverzeichnis 28](#_Toc475659260)

[II. Tabellenverzeichnis 28](#_Toc475659261)

[III. Codelistings 29](#_Toc475659262)

[a. Installationsanleitung 32](#_Toc475659263)

# Einleitung

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

# Grundlagen

Kürzeste Wege?

## Morphologie von Pflanzen

### Bestimmungsmerkmale höherer Pflanzen

Die als Kormophyten bezeichneten höhere Pflanzen weisen einen Kormus genannten Pflanzenkörper auf, der sich aus den Grundorganen Laubblatt, Wurzel und Sprossachse zusammensetzt. Die Gestalt der Organe steht in enger Wechselbeziehung mit ihrer Funktion. (Welle, 2014, S. 16)

Abhängig von den Lebensumständen des Organismus, variiert die Beschaffenheit der Organe zwischen den Arten. Anhand des Erscheinungsbildes ihrer Organe, können Pflanzen bestimmt und zugeordnet werden. Mit dieser Thematik beschäftigt sich die Morphologie, die Lehre von der äußeren Gestalt der Pflanzen. (Welle, 2014, S. 13)

Die Wurzel hat die Befestigung der Pflanze im Boden und die Aufnahme von Wasser und Nährsalzen zur Aufgabe. Die Gestalt des Wurzelsystems wird durch Standortfaktoren wie die Bodenbeschaffenheit, bestimmt (Welle, 2014, S. 18). Wurzeln können neben den ursprünglichen Aufgaben weitere Funktionen übernehmen, wie die Speicherung von Nährstoffen als Rüben oder Wurzelknollen. Die Anpassung an die zusätzliche Aufgabenstellung hat eine Veränderung der Grundform, das heißt eine Metamorphose zur Folge (Welle, 2014, S. 42).

Hauptaufgabe der Sprossachse ist die Laubblätter zu tragen und sie mit der Wurzel zu verbinden, um Wasser und Nährsalze zwischen diesen beiden Organen zu transportieren (Welle, 2014, S. 22). Wichtige Unterscheidungsmerkmale von Sprossachsen sind ihre Stellung, ihre Beschaffenheit und die Form ihres Achsenquerschnitts (Rothmaler, 1967, S. X). Metamorphosen der Sprossachse können, unter anderen, Sprossdornen, Knollen und Zwiebeln sein (Welle, 2014, S. 48, 50).

Bei Samenpflanzen können zusätzlich Früchte und Samen als Bestimmungsmerkmal dienen. Diese bilden sich nach der Befruchtung aus dem Fruchtknoten mit der eingeschlossenen Samenanlage. (Welle, 2014, S. 140)

### Morphologie des Blattes

#### Funktion und Aufbau des Blattes

Laubblätter haben in ihrer Grundform die Funktionen des Gasaustauschs, der Transpiration und der Photosynthese inne. Die Photosynthese dient dem Aufbau von energiereichen Kohlenhydraten aus dem energieärmeren Kohlendioxid und Wasser unter dem Einfluss des Sonnenlichtes und der Anwesenheit von Chlorophyll. Die Konzentration von Letzterem bestimmt die Intensität der Grünfärbung der Blätter. (Welle, 2014, S. 26)

In Anpassung an bestimmte Gegebenheiten können Blätter als Metamorphosen die Gestalt von Blattranken als Halte- und Kletterorgane, von Blattdornen zum Schutz vor Tierfraß und von Fangblättern bei fleischfressenden Pflanzen annehmen. (Welle, 2014, S. 54)

Die sich aus metaphorisierten Laubblättern zusammensetzenden Blüten sind ein weiteres wichtiges Unterscheidungsmerkmal von sich geschlechtlich fortpflanzenden Samenpflanzen. Die Blütenhülle wird aus den umschließenden schützenden Kelchblättern und den gefärbten Blumen- oder Kronblättern gebildet. Die Staubblätter stellen das männliche und die Fruchtblätter das weibliche Geschlechtsorgan der Pflanze dar. (Welle, 2014, S. 118)

Abbildung aus Welle S. 27 links

Die Fläche des Laubblattes wird als Blattspreite bezeichnet (Abb. 1(1)). Das Ende des Blattes bezeichnet man als Blattspitze (Abb. 1(2)). Abgegrenzt wird das Blatt vom Blattrand (Abb. 1(3)). Die Blattfläche wird von Blattnerven (Abb. 1(4)) durchzogen. Über den Blattstiel (Abb. 1(8)) ist das Blatt mit der Sprossachse verbunden. Der Blattgrund (Abb. 1(7)) bezeichnet den direkt an die Sprossachse angrenzenden Teil des Blattes. (Welle, 2014, S. 26)

Im Folgenden wird betrachtet, anhand welcher Merkmale man Laubblätter morphologisch unterscheiden kann.

#### Blattformen

Die Form der Blattspreite ist sehr variantenreich und laut Welle (2014, S. 32) ein wichtiges Merkmal zur Bestimmung der Pflanzen. Ein Auszug aus den möglichen Formen ist in Abbildung X zu sehen.

Grundsätzlich kann die Form unterschieden werden in einfache ungeteilte Blätter (Abb. X a-c), bei denen der Rand nicht oder nur leicht eingeschnitten oder gelappt ist, und geteilte Laubblätter (Abb X d-e), deren Blattspreite durch mehr oder weniger tiefe Einschnitte geteilt sind (Welle, 2014, S. 32, 34). Besteht die Blattspreite aus mehreren selbstständigen Blattteilen, spricht man von zusammengesetzten Blättern (Rothmaler, 1967, S. XIV). Die Anordnung der Blättchen genannten Teile ist eine weitere Möglichkeit der Differenzierung (Abb. X f-).

Zur Variation innerhalb der Blattformen tragen auch die Form des Spreitengrundes und der Spreitenspitze bei, welche in Abbildung X zu sehen sind.

Der Blattrand kann als weiteres Unterscheidungsmerkmal dienen. Die Gliederung nach Rothmaler (1967) ist in Abb. X dargestellt.

Nicht alle Blätter weisen, abhängig von der Art der Anheftung an die Sprossachse, einen Stiel auf. Bei gestielten Blättern ist der Blattstiel deutlich ausgebildet, während stengelumfassende, durchwachsene oder sitzende Blätter keinen Stiel ausbilden. (Rothmaler, 1967, S. XII)

Weiter kann man Blätter anhand ihrer Anordnung an der Sprossachse in wechselständige, gegenständige, quirlständige und bodenständige Blätter differenzieren (Rothmaler, 1967, S. XIII). Ferner dienen der Unterscheidung die Beschaffenheit des Blattes, die sich in ledrig, krautig, fleischig und häutig unterteilen lässt (Rothmaler, 1967, S. XVII) und der Spreitenquerschnitt, der die Art der Rollung oder Faltung des Blattes beschreibt (Rothmaler, 2011, S. 887).

#### Blattnervatur

Die Blattnerven, oder synonym Blattadern, sind das Blatt durchziehende Leitbündel. Ihre Aufgabe ist der Transport von Stoffen und die Wahrung der Festigkeit der Blattspreite (Welle, 2014, S. 86). Die Anordnung der Nerven kann zwischen den Arten variieren.

Nach Rothmaler (1967, S. XIII) unterscheidet man nach dem Verlauf der Blattnerven Blätter in:

* streifennervig: gleich starke Nerven verlaufen vom Blattgrund ohne Verzweigung nebeneinander zur Blattspitze
* netznervig: von einer oder mehreren Hauptnerven gehen kleinere Seitennerven ab, die in ein engmaschiges Adernetz übergehen. Weiter untergliedert man netznervige Blätter in:
  + fiedernervig: die Seitennerven gehen zu beiden Seiten von einem Hauptnerv ab
  + fingernervig: die größeren Nerven gehen strahlenförmig vom Blattstiel aus

## Digitale Bildverarbeitung

### Diskrete Bildrepräsentation

Da Computer keine kontinuierlichen analogen Bilder verarbeiten können, werden diese im einfachsten Fall in ein zweidimensionales Gitter überführt. Die meist mit der Matrixnotation angegebene Position innerhalb des Gitters deckt eine rechteckige Region des Bildes ab, die von einem Pixel repräsentiert wird. Der Wert des Pixels gibt die mittlere Intensität der Bestrahlung in dieser Bildregion wieder. (Jähne, 2012, S. 110) Ist das verständlich?

Die Auflösung des diskreten Bildes bestimmt die Anzahl der Pixel und somit den Detailgrad der Darstellung. Laut Jähne (2012, S. 111) „sollte die Pixelgröße kleiner sein als die kleinsten Objekte, die untersucht werden sollen“. Dagegen steht, dass der Aufwand der Bildverarbeitung mit steigender Pixelzahl zunimmt. (Jähne, 2012, S. 111).

Ein weiterer Aspekt der Auflösung ist die Tiefe. Diese bezieht sich auf die Anzahl der Bits, die einem Pixel zur Verfügung stehen, um den Intensitätswert zu speichern. Bei 8-Bit Tiefe lassen sich beispielsweise 28 = 256 verschiedene Intensitäts- oder Farbwerte abbilden. (Burdick, 1997, S. 17)

Mit einem Wert je Pixel können lediglich Graustufen dargestellt werden. Für die Repräsentation von Farben werden jedoch mehr Dimensionen benötigt. Bei den Farbräumen handelt es sich um dreidimensionale orthogonale Räume, deren Achsen senkrecht zueinander stehen. Im Falle des RGB-Farbraums bilden die Achsen die Intensitäten der Farben Rot, Grün und Blau. Der RGB-Farbraum ist additiv, das heißt alle Farben werden durch Addition der einzelnen Farbwerte abgeleitet, beginnend bei Schwarz, wenn kein Farbwert gesetzt ist. In der Bildverarbeitung werden häufig RGB-Bilder verwendet, da Computermonitore ihre Darstellung nativ, also direkt unterstützen. (Burdick, 1997, S. 22)

|  |  |
| --- | --- |
| Abbildung 1 - RGB-Farbraum | C:\Users\chris\AppData\Local\Microsoft\Windows\INetCache\Content.Word\1280px-HSV_color_solid_cylinder_alpha_lowgamma.png  Abbildung 2 - HSV-Farbraum als Zylinder (Wikipedia, 2017) |

Das RGB-System entspricht allerdings nicht der menschlichen Farbwahrnehmung. Eine natürlichere Beschreibung von Farben ergibt sich durch ihre Unterteilung in Farbton (Hue), Sättigung (Saturation) und Intensität (Intensity) im sogenannten HSI-Raum, der ein Polarkoordinatensystem verwendet. (Jähne, 2012, S. 54)

Es gibt viele Methoden, um Bilder vom RGB-Raum in den HSI-Raum und zurück zu transformieren. Eine spezielle Implementierung ist das HSV-Farbmodell, dessen Farbraum in Abbildung 2 zylindrisch dargestellt ist. Das Modell generiert den Farbton zwischen 0° und 360°, beginnend mit Rot. Die Sättigung liegt im Wertebereich von 0 bis 1, wobei 0 die Abwesenheit von Farbe codiert, also Grau, in Abbildung 2 entlang der zentralen Achse zu sehen. Der Value ist eine Variation der Intensität und liegt zwischen 0 (schwarz) und 1 (weiß). (Burdick, 1997, S. 29)

### Binarisierung

Binärbilder sind Bilder mit nur einem Bit je Pixel. Jeder Pixel kann folglich nur einen von zwei Werten annehmen, schwarz oder weiß. In der Bildverarbeitung werden die beiden Werte auch häufig als Vorder- beziehungsweise Hintergrund bezeichnet. (Burger & Burge, 2015, S. 223)

Anwendung finden Binärbilder bei morphologischen Filtern (Abschnitt 2.2.3.4), zur Bestimmung momentbasierter geometrischer Eigenschaften und zur Extraktion von Konturen im Bild (Abschnitt??).

Um ein Bild, meist ein Graustufenbild, zu binarisieren, also in ein Binärbild zu überführen, wendet man Schwellwertoperationen an. Die Pixel werden, anhand ihrer Intensität *α*, in einer Punktoperation in zwei Klassen unterteilt. Die Klasse dient der Zuweisung von einem von zwei festen Werten, α0 und α1. Das heißt, dass Pixel mit einer Intensität unterhalb eines gewählten Schwellwertes *q* den Wert von *α0* erhalten, darüber den Wert *α1* (Burger & Burge, 2015, S. 61):

(1)

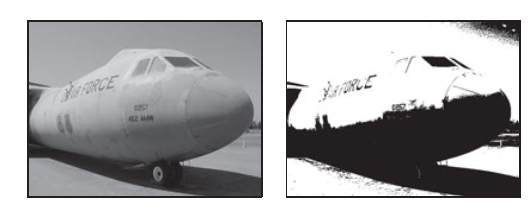
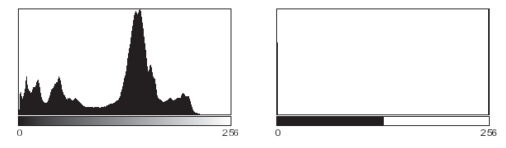
 

Abbildung 3 - Schwellwertoperation angewendet auf ein Graustufenbild (links oben) mit resultierendem Binärbild (rechts oben), Histogramme des Originalbildes (links unten) und des Binärbildes (rechts unten) (Burger & Burge, 2015, S. 62)

Die Findung eines geeigneten Schwellwertes ist die Aufgabe von automatischen Schwellwertoperationen. Viele Verfahren zur Schwellwertberechnung werten dazu die Histogramme des Ausgangsbildes aus. Histogramme sind Darstellungen von Häufigkeitsverteilungen bestimmter Merkmale. Im Falle von Bildern, zeigen Histogramme die Häufigkeit der auftretenden Intensitätswerte an (Burger & Burge, 2015, S. 42). Beispielhaft ist in Abbildung 3 ein 8-Bit Graustufenbild mit zugehörigem Histogramm (links) und das Binärbild mit Histogramm nach einer Schwellwertoperation (rechts) zu sehen.

Bei Verfahren zur automatischen Schwellwertbestimmung wird zwischen globalen und adaptiven Methoden unterschieden. Globale Schwellwertmethoden bestimmen für ein Graustufenbild einen einzigen optimalen Schwellwert, anhand dessen die Pixel Vorder- oder Hintergrund zugeordnet werden (Burger & Burge, 2015, S. 268). Globale Schwellwerte können bei ungleichmäßiger Beleuchtung im Bild unerwünschte Ergebnisse erzielen. Bei adaptiven Methoden wird daher für jeden einzelnen Bildpunkt ein angepasster Schwellwert bestimmt (Burger & Burge, 2015, S. 291).

Otsu ist ein Beispiel für eine globale, histogrammbasierte Schwellwertbestimmung. Das Verfahren versucht die Varianz, also die Streuung der Grauwerte, innerhalb der beiden zu bildenden Klassen zu minimieren. Die Varianz kann direkt aus dem Histogramm des Bildes abgeleitet werden. Gleichzeitig soll die Trennung zwischen den Klassen möglichst groß sein, Dazu müssen ihre Mittelwerte möglichst weit auseinander liegen. (Burger & Burge, 2015, S. 275)

Laut Burger & Burge (2015, S. 278) arbeitet die Otsu-Methode „sehr effizient“ und liefert in der Praxis „recht brauchbare Ergebnisse“.

### Filter

#### Allgemeine Definition

Filter, auch Nachbarschaftsoperationen genannt, betrachten, im Gegensatz zu Punktoperationen, nicht nur einen Pixel, sondern auch eine kleine Region um ihn herum, die Nachbarschaft. Sie kombinieren die benachbarten Pixel in einer bestimmten Weise und liefern so ein geändertes Bild. Wichtige Aufgaben, die Filter übernehmen können, sind beispielsweise die Unterdrückung von Rauschen, die Korrektur von Störungen oder die Detektion und Unterscheidung lokaler Strukturen im Bild, wie Kanten oder Linien. (Jähne, 2012, S. 296, 297)

Die Größe der Filterregion, also die Größe der Nachbarschaft, ist ein wichtiger Parameter des Filters. Sie bestimmt, wie viele umgebende Pixel zur Berechnung des neuen Wertes für den betrachteten Pixel herangezogen werden. Mit unterschiedlichen Größen erzielt man unter Umständen auch unterschiedlich starke Effekte. (Burger & Burge, 2015, S. 94)

Die Anwendung des Filters erfolgt für jeden Bildpunkt, durch beispielsweise zeilenweises Ablaufen des Bildes, wie in Abbildung 3 zu sehen. Die grauen Bereiche in der Abbildung stellen bereits betrachtete und geänderte Pixel dar. Wichtig ist, dass für die Betrachtung der Umgebung die ursprünglichen Werte des Bildes *I* herangezogen werden. Die Änderungen werden deshalb separat in einem Ergebnisbild *I‘* gespeichert werden. (Jähne, 2012, S. 303)

|  |  |
| --- | --- |
| Abbildung 4 - Anwendung eines Filters durch zeilenweises Verschieben der Filtermaske (Jähne, 2012, S. 303) | Abbildung 5 - Koordinatensystem der Filtermatrix *H* mit „Hot Spot“ genanntem Ursprung in der Mitte (Burger & Burge, 2015, S. 96) |

#### Lineare Filter

Lineare Filter verknüpfen die Werte der Pixel in der Umgebung in linearer Form durch eine gewichtete Summation. Die Filterfunktion wird durch eine Matrix abgebildet, in der die Gewichtungen der Pixelwerte der Umgebung gespeichert sind. Die Dimension der Matrix gibt die Größe der betrachteten Umgebung an. (Burger & Burge, 2015, S. 95)

Die Filtermatrix besitzt ein eigenes Koordinatensystem, dessen Ursprung üblicherweise in ihrer Mitte liegt (Abbildung 4). Bei Anwendung der Filtermatrix *H*, wird sie mit ihrem Ursprung über den aktuell betrachteten Bildpunkt *I(u, v)* positioniert. Darauf folgt eine Multiplikation aller gewichteten Elemente *H(i, j)* innerhalb der Matrixregion *R* mit den darunter liegenden Bildpunkten *I(u+i, v+j)* und eine anschließende Summation aller Produkte. Das Ergebnis wird im Ergebnisbild *I‘(u, v)* gespeichert, an derselben Position wie im Ausgangsbild *I(u, v)*. (Burger & Burge, 2015, S. 96)

Auf alle Bildpunkte angewendet, lässt sich die Operation wie folgt formulieren:

|  |  |
| --- | --- |
|  | (2) |

Die durchgeführte Operation entspricht einer Faltung *I‘ = I \* H* mit dem Faltungsoperator (\*). In der Mathematik beschreibt die Faltung einen Operator, der zwei Funktionen gleicher Dimension miteinander verknüpft. (Burger & Burge, 2015, S. 105)

Wichtige mathematische Eigenschaften von linearen Faltungen sind die Kommutativität *I \* H = H \* I*, die Linearität und die Assoziativität *(I \* H1) \* H2 = I \* (H1 \* H2)*. Auf Filter bezogen bedeutet die Assoziativität, dass man das gleiche Ergebnisbild erhält, auch wenn die Reihenfolge der Anwendung verschiedener Filter verändert wird, oder sie zu durch paarweise Faltung zu neuen Filtern zusammengefasst werden. (Burger & Burge, 2015, S. 107)

Aus der Assoziativität folgt auch die Separierbarkeit eines einzelnen Filters in kleinere Filter, die nacheinander ausgeführt werden können. Voraussetzung ist, dass sich der zweidimensionale Filter als Produkt von zwei eindimensionalen Filtern darstellen lässt. (Burger & Burge, 2015, S. 108)

Ein Beispiel für einen solchen separierbaren Filter ist der Gaußfilter *Gσ(x, y)*, der sich in zwei einzelne Funktionen separieren lässt (Burger & Burge, 2015, S. 109):

|  |  |
| --- | --- |
|  | (3) |

Die Gaußfunktion ist glockenförmig, dargestellt in Abbildung 5 links. Die Standardabweichung σ bestimmt ihren Radius und somit auch die Größe der Filtermatrix. Der Ursprung der Matrix erhält das maximale Gewicht, die Größe der übrigen Koeffizienten nimmt zu den Rändern hin kontinuierlich ab (Abbildung 5 rechts).

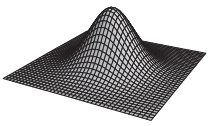
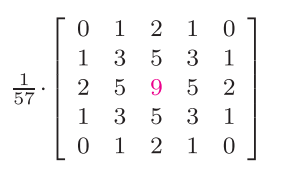
 

Abbildung 6 - Beispiel für einen Gaußfilter, Darstellung der Filterfunktion *H* als 3D-Plot (links) und als Filtermatrix (rechts) (Burger & Burge, 2015, S. 103)

Der Gaußfilter ist ein sogenannter Glättungsfilter, oder auch Tiefpassfilter. In der Bildverarbeitung werden durch Glättung die Intensitätsunterschiede benachbarter Bildelemente verringert. In der menschlichen Wahrnehmung wirken Bilder dadurch unschärfer. (Burger & Burge, 2015, S. 104)

Kanten erhaltend?

zu Canny überleiten

#### Kantenerkennung

#### Morphologische Filter

### Filter

## Maschinelles Lernen

### Grundlagen

### Features

#### Momentenbasierte geometrische Merkmale

#### KMeans

### 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

## Zielstellung

abbilden, wie Mensch Blätter unterscheidet -> möglichst viel „Domänenwissen“ in features packen

keine Aussage über Größe möglich

Konzentration auf Form der Blattspreite, da sehr variabel

Blätter einzeln betrachten, Anordnung fällt somit weg. Aber: zusammengesetzte Blätter auch als solche betrachten (nicht Blättchen einzeln)

keine Metamorphosen

Fotos einzelstehender Blätter

## Testumgebung

aus Oberfläche heraus Segmentierung testen, Features extrahieren/kombinieren, Training starten

Unittests

## Vorhandene Lösungen (oder Einordnung in Kontext oder so)

### Überblick

Wie in Kapitel **Fehler! Verweisquelle konnte nicht gefunden werden.** 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, o. J.) | Graustufen, Binarisierung, Kontur, Region of Interest | geometrische features, invariante Features (Hu-Momente) | moving median center hyperspheres (MMC), KNN | ? |  |
| (Han, Chee, Chan, Wilkin, & Remagnino, o. J.) | ? | 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 u. a., 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, o. J.) | ? | 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).

Warum die beiden?

Farbe, Panaschierung … (Leafnet hat Farbschema an Seite)



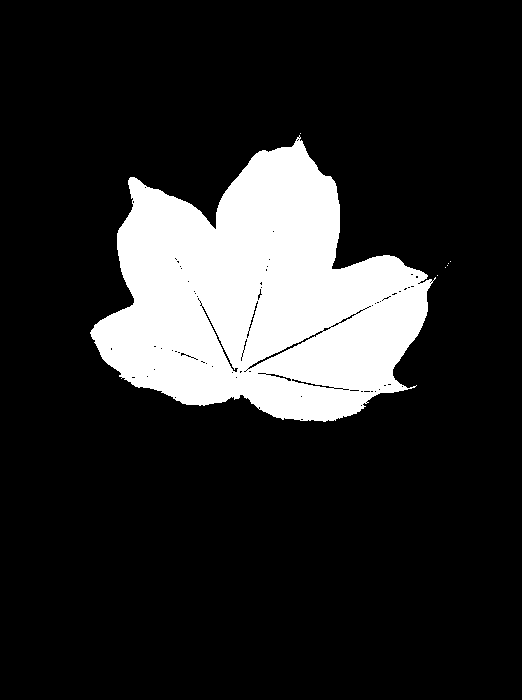


Abb. 10a: Beispielfoto aus der Flavia-Datenbank Abb. 10b: unbearbeitetes (links) und segmentier-

(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 falsch (ungeordnet bei Leafsnap). Es tauchen Bilder von Blattunterseiten zwischen denen von Blattoberseiten auf. 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. geordnet, gleiche Menge an Ober- und Unterseiten, bessere Möglichkeit zu vergleichen

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

alles gestielte Blätter, netzadrige …

keine nadelförmigen Blätter von Nadelbäumen, wenig Textur-Informationen

Bild mit zusammengesetzter Blattform

### 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!)*.

da hintergrund heller -> invers (definition vorder/hintergrund)

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

Wie aus den Grundlagen der Morph. von Blättern hervorgeht, sind Blätter sehr formenreich

Bezug auf Blattmerkmale, was nehmen, was nicht

zweidimensionale Abbildung

3D theoretisch möglich mit Bildserien aus verschiedenen Blickwinkeln, zu vernachlässigende Merkmale, Rollung fällt somit weg

invariant weil?

rotation

scale - verschiedene Winkel

Form und Textur möglichst getrennt betrachten

keine Farbe, weil Blätter zumeist grün, keine Blüttenblätter, die bunt sein können und bei denen es eine wichtige Rolle spielen würde

mit oder ohne Blattstiel? laut Grundlagen (Anheftung) auch Aussagekraft

### Blattform

Wie in Kapitel 2.1.2.2 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 2.3.2.1 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 u. a., 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)

### Lernverfahren

SVM, MLP

Deep Learning in letzter Zeit zwar der neueste heiße Scheiß, aber hier „Domänenwissen“ verwenden

pcnn - pulse-coupled neural networks (Wang u. a., 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

### Metriken 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

in Kapitel 3 rein!

Binärbilder trotzdem in 8Bit, definition: 0 Hintergrund >0 (1 oder 255) Vordergrund, opencv liefert beim thresholding 255

nicht Java sondern Python

OpenCV Konturpunkte, Binarisierung

skimage und opnecv, eigentlich alle benutzen numpy arrays als represäntation von Bildern, benutzt: 8bit farbrepräsentation

versuch schleifen zu vermeiden mit numpy, indizierung über mehrere Indizes

### 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

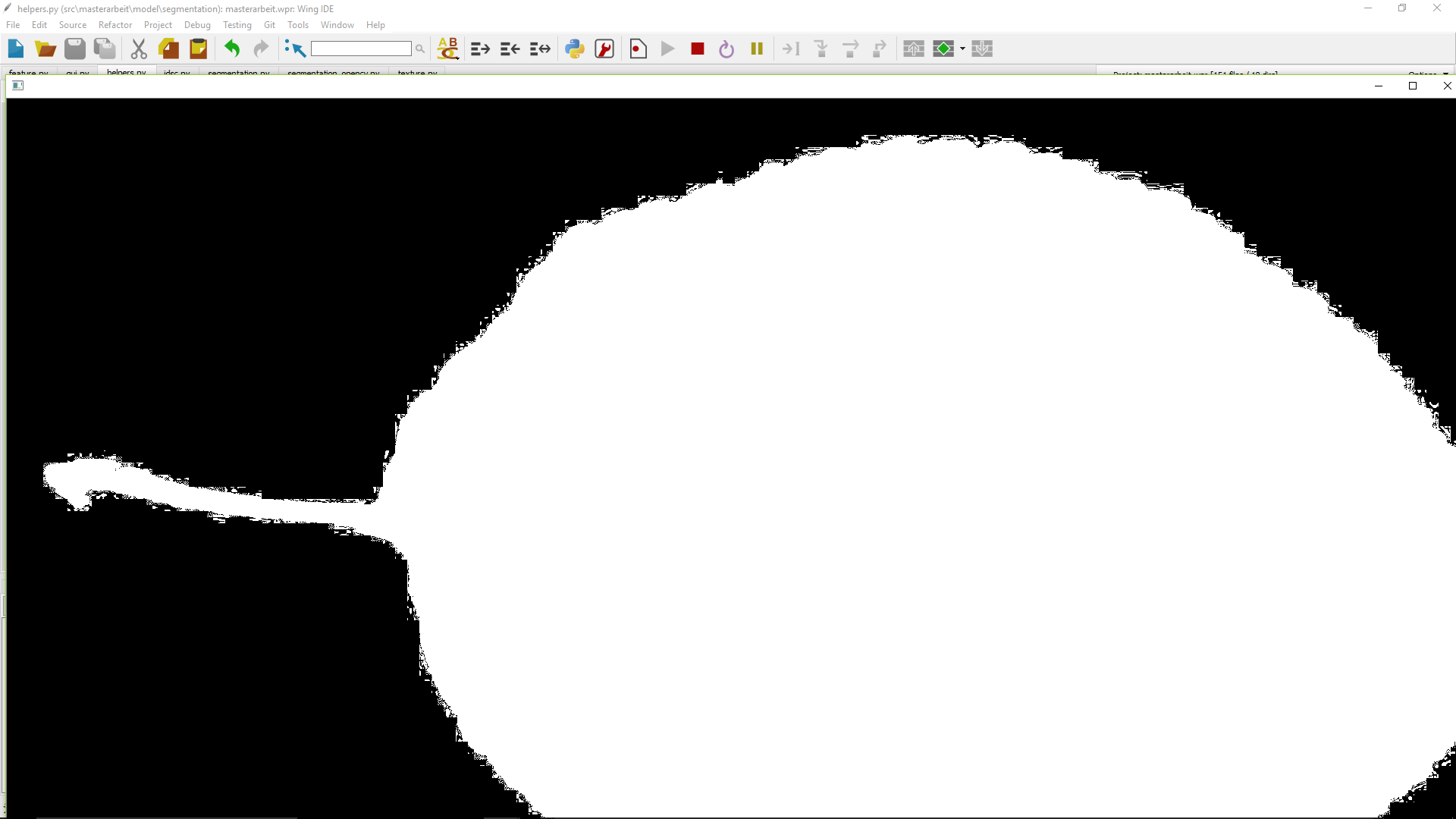
nur S - weniger schatten, noisy, da fragmente der schatten

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

Stiel wichtig (Grundlage), Details von Blattgrund und Blattspitze kann verloren gehen

Entscheidung: Stiel bleibt dran

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 hierhin

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 Winkel und Wegelängen als Polarkoordinaten in jeweils acht Klassen unterteilt. Der Wertebereich der Winkel liegt zwischen 0 und 2π, der der Wegelängen zwischen 0 und dem Logarithmus der maximalen Distanz. 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.

### Multilevel 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. Anderer Name, da Umsetzung von dem vorgeschlagenen Algorithmus abweicht.

Wie auch beim Independent Inner Distance Shape Context, wird zunächst mehrere Stufen des binarisierten Bildes gebildet. Auf der niedrigsten, der feinsten Stufe, bleibt das Ausgangsbild erhalten, um alle Details der Kontur abzubilden. Auf den weiteren Stufen wird das Ausgangsbild mit einem Gauß-Filter *Gσ(x,y)* gefaltet, und das Bild somit geglättet (siehe 2.2.3.2). Zwischen den Stufen steigt die Standardabweichung *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. Thresholding!

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 Basiselemente, aufgrund der verschieden definierten Nachbarschaften, nicht direkt miteinander kombinierbar. Daher wird für jede Stufe ein eigenes Codebook benötigt.

### Codebooks

Die Klassifizierer benötigen als Eingangsdaten eindimensionale 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 eindimensionale geordnete Feature-Vektoren nutzbar sind. Die Aufgabe der Bestimmung geeigneter visueller Codewörter übernehmen die im Folgenden beschriebenen Codebooks. (kommt in 3 rein)

### 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. sind die Kombinationen der Atome des Wörterbuchs sich wiederholende Muster und können somit als visuelle Wörter betrachtet werden. und in einem Histogramm gezählt. Abweichend nicht darauf geachtet, dass nur bestimmte Zahl auftauchen darf.

Atom als visuelles Wort

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

bekannte Fehler: gelegentlicher hdf5 error beim Zugriff auf Datei

„H5F\_open - unable to open file“

# Abbildungsverzeichnis

[Abbildung 1 - RGB-Farbraum 5](#_Toc475643765)

[Abbildung 2 - HSV-Farbraum als Zylinder (Wikipedia, 2017) 5](#_Toc475643766)

[Abbildung 3 - Anwendung eines Filters durch zeilenweises Verschieben der Filtermaske (Jähne, 2012, S. 303) 6](#_Toc475643767)

[Abbildung 4 - Koordinatensystem der Filtermatrix *H* mit „Hot Spot“ genanntem Ursprung in der Mitte (Burger & Burge, 2015, S. 96) 6](#_Toc475643768)

# Tabellenverzeichnis

**Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.**

# Codelistings

**Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.**

1. Literaturverzeichnis

Burdick, H. E. (1997). *Digital Imaging- Theory and Applications*. New York: McGraw-Hill.

Burger, W., & Burge, M. J. (2015). *Digitale Bildverarbeitung: eine algorithmische Einführung mit Java* (3. Aufl.). Berlin, Heidelberg: Springer-Verlag. https://doi.org/10.1007/978-3-642-04604-9

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. (o. J.). 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. (o. J.). DEEP-PLANT: PLANT IDENTIFICATION WITH CONVOLUTIONAL NEURAL NETWORKS.

Jähne, B. (2012). *Digitale Bildverarbeitung: und Bildgewinnung (German Edition)* (7. Aufl.). Heidelberg: Springer Verlag.

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. (o. J.). Fine-tuning Deep Convolutional Networks for Plant Recognition.

Rothmaler, W. (1967). *Exkursionsflora von Deutschland - Gefäßpflanzen* (6. Aufl.). Berlin: Volk und Wissen volkseigener Verlag.

Rothmaler, W. (2011). *Exkursionsflora von Deutschland - Gefäßpflanzen* (20. Aufl.). Heidelberg: Spektrum Akademischer Verlag.

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

Welle, E. F. (2014). *Kleines Repetitorium der Botanik* (16. Aufl.). Hamburg: Verlag Dr. Felix Büchner - Handwerk und Technik.

Wikipedia, C. (2017). HSL and HSV. Abgerufen 22. Februar 2017, von https://en.wikipedia.org/w/index.php?title=HSL\_and\_HSV&oldid=765756890

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