

Projektarbeit

Automatische Vogelzug-Erfassung mittels Radarbildern

Georg Kuschk

15. Juli 2007



Martin-Luther-Universität Halle-Wittenberg
Institut für Informatik

Inhaltsverzeichnis

1	Einleitung	1
1.1	Aufgabenstellung	1
1.2	Existierende Software	3
2	Methoden und Algorithmen	5
2.1	RGB-IHS - Transformation	5
2.2	Radar-Hilfslinien entfernen	6
2.3	Bildstörungen, Aussortieren gestörter Bilder	7
2.4	Binarisierung	9
2.5	Regionen-Segmentierung	9
2.6	Statische Signale, Region-Of-Interest-Maske	9
3	Klassifikation	11
3.1	Objektklassen	11
3.2	Aufbau des Klassifikators	11
3.3	Klassifikationsgüte	12
3.4	Merkmale	13
3.5	SVM-Klassifikator	16
3.5.1	Theorie	16
3.5.2	Wahl der SVM-Parameter	18
3.5.3	Unausgeglichenheit der Daten	20
3.6	Flugzeugsignale	21
4	Systemaufbau	24
4.1	Systemüberblick	24
4.1.1	Trainingsphase	24
4.1.2	Anwendungsphase	27
4.2	Programmausgabe	28
4.3	Zusätzliche Programmfunktionalitäten	30
5	Implementierung	31
5.1	Verwendete Software	31
6	Ergebnisse	33
6.1	Gütequalität	33
6.2	Testlauf	33
6.3	Zusammenfassung und Ausblick	36
	Literatur	38

1 Einleitung

1.1 Aufgabenstellung

Die Verhütung ungewollter Zusammenstöße von Vögeln und Flugzeugen im Luftverkehr ist eine sowohl wirtschaftlich, als auch sicherheitstechnisch nicht zu unterschätzende Problematik, sowohl im zivilen als auch im militärischen Gebiet. Die wichtigste Voraussetzung für sinnvolle Maßnahmen zur Vogelschlagverhütung ist eine exakte Vogelschlagstatistik.

In diesem Rahmen ist die GmbH Flughafen Leipzig/Halle über den DAVVL (Deutscher Ausschuss zur Verhütung von Vogelschlägen im Luftverkehr) an Mitarbeiter des zoologischen Instituts der Martin-Luther-Universität Halle mit der Beauftragung eines Vogelzug-Gutachtens herangetreten. Aufgrund des enormen Umfangs der auszuwertenden Daten wurde der Fachbereich Informatik kontaktiert, mit der Zielstellung, ein automatisiertes Analyseverfahren zu entwickeln. Die vorliegende Projektarbeit ist das Resultat dieser Zusammenarbeit. Die auszuwertenden Daten bestehen hierbei aus Radarbildern, welche im 60 Sekunden Takt über 2 Jahre aufgezeichnet werden. Dies entspricht 10080 auszuwertenden Bildern pro Woche. Die gewünschte Ausgabe der Bildanalysen ist eine Auflistung aller erkannten Vogel-Signale mit zugehöriger Beobachtungszeit, Flughöhe, sowie Distanz zum Radar. Zur Anwendung kommt ein Schiffs-Radar der Firma Furuno Deutschland GmbH (FAR-2127BB), welches allerdings nicht flächig über dem Boden, sondern um 90 Grad gekippt arbeitet (siehe Abbildung 1).

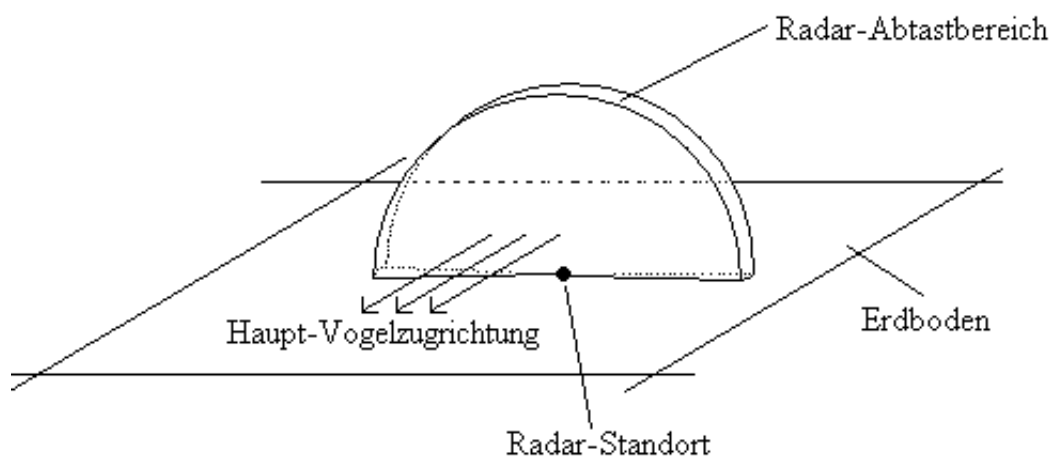


Abb. 1: Schematischer Aufbau der Messanlage

Auf diese Weise wird ein halbkreisförmiger, senkrecht nach oben ragender Ausschnitt des Himmels betrachtet. (Die 'Dicke' dieses Halbkreises entspricht den Ausdehnungen der entsprechenden Radarkeule, deren Berücksichtigung jedoch nicht

Aufgabe der Zählung der Vogelsignale ist, sondern erst seitens der zoologischen Mitarbeiter in die Statistik-Berechnungen integriert wird.) Die Ausdehnung, d.h. Reichweite des Radarbereiches ist variabel einstellbar, wurde jedoch für die Aufgabenstellung auf 1.5km festgelegt. Desweiteren erfolgt die Ausrichtung des Radars senkrecht zur Haupt-Vogelzugrichtung, so dass Vögel den Radar-Abtastbereich hauptsächlich frontal durchfliegen und selten seitwärts. Die Ausgabe dieses Radargerätes sind Bilder vom Format 1280x1024 mit 24Bit Farbtiefe (RGB) (siehe Abbildung 2).

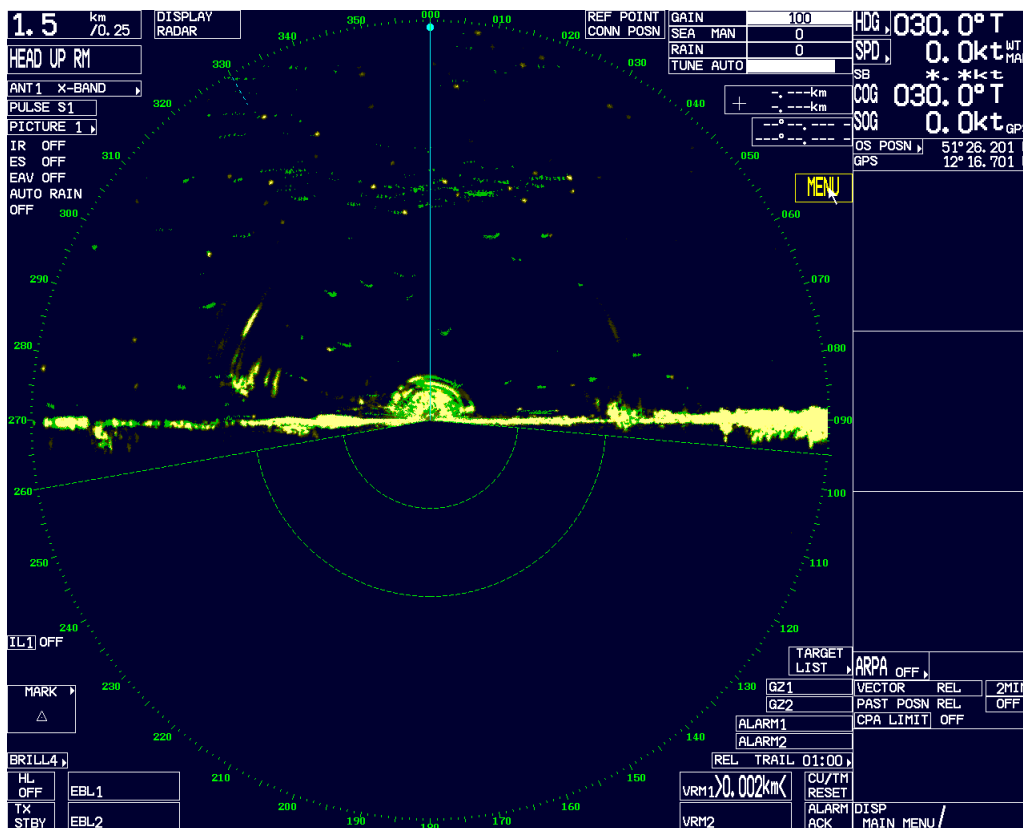


Abb. 2: Beispiel eines auszuwertenden Bildes : Der Erdboden ist in der Bildmitte durch die horizontale, gelbe, linienartige Struktur erkennbar, entlang welcher Störungen aufgrund von Reflexionen auftreten. Oberhalb sind gelbe Regionen zu sehen, welche den, durch das Radar erkannten, Flugobjekten entsprechen. Die grünen Strukturen kennzeichnen Bewegungen dieser Signale. (= Tracking-Informationen - Verändert ein Signal im Radarbereich seine Position oder verlässt diesen, so 'hallt' das Signal noch eine bestimmte Zeit lang mit der sogenannten Tracking-Farbe eingefärbt nach.) Der Text am Rand stellt einige Parameter und Positionsangaben der Radaranlage dar.

Diese Bilder stellen die auszuwertenden Daten dar, d.h. in ihnen sollen die vorkommenden Vogelsignale erkannt werden. Um dem späteren Sprachgebrauch vorzugrei-

fen, sei hier erwähnt, dass diese RGB-Bilder im allerersten Vorverarbeitungsschritt in den IHS-Farbraum (Intensity-Hue-Saturation) konvertiert werden, und für die weiteren Verarbeitungsschritte nur noch der Intensity-Kanal, sowie der Hue-Kanal, d.h. Intensität und Farbwert, benutzt werden (siehe Abschnitt 2.1)

Die detailliertere Aufgabenstellung besteht nun darin, aus den Einzelbildern die potentiellen Vogelsignale (gelbe Regionen) zu extrahieren und aus diesen charakterisierende Merkmale zu ermitteln. Anhand dieser Merkmale werden alle extrahierten Signale klassifiziert und die Ergebnisse in, für Statistikprogramme einfach zu importierende, Dateien ausgegeben.

1.2 Existierende Software

Für die Aufgabenstellung existiert keine bekannte Software. Bis zum jetzigen Zeitpunkt erfolgte die Auswertung der Radarbilder manuell. Einzig zum manuellen Vermessen von Objektgrößen und -geschwindigkeiten kam das folgende Programm zum Einsatz :

- **Image Tool**

(University of Texas, 2002)

Hierbei handelt es sich um eine (für den medizinischen Bereich entwickelte) Algorithmensammlung, um manuell diverse Bildverarbeitungsoperationen auf Bildern auszuführen. Als Beispiel seien Schwellwertoperationen, Histogrammanalysen, Größen- und Distanz-Vermessungen, sowie Filteroperationen genannt. Diese sind auch mittels einer Skriptsprache zu Mehrbild-Analysen verknüpfbar. Die implementierten Funktionen sind jedoch für allgemein gehaltene Aufgabenstellungen vorgesehen und für spezielle Analysen nicht geeignet, zumal das Programm aufgrund fehlerhafter Speicher-Allozierungen mitunter abstürzt.

Hervorzuheben sei aber an dieser Stelle, dass es auch eine Klassifikations-Option gibt. Diese binarisiert Grauwertbilder entsprechend eines Schwellwertes und führt eine Regionen-Segmentierung durch. Für eine Auswahl an 15 vorgegebenen Parametern (z.B. Fläche, Grauwert-Intensität, Kompaktheit, etc.) können Maximalwerte eingegeben werden, anhand derer Objekte, welche in diese Merkmals-Intervalle fallen, als 'positiv' oder 'negativ' klassifiziert werden.

Gegen den Einsatz dieser Klassifikationsmethode zur Lösung der vorliegenden Aufgabenstellung sprechen jedoch :

- Die Operation ist nur auf Grauwert-Bildern möglich, und für deren Konvertierung gibt es lediglich einen 'Color-to-Grayscale' - Operator, welcher ohne Eingabe von Parametern oder Informationen zur Vorgehensweise, aus einem RGB-Bild ein Grauwertbild berechnet. D.h. der Farbinformationsgehalt geht zum Teil verloren, da die drei Farbkanäle auf nicht kommentierte Weise zu einem Kanal 'verrechnet' werden.
- Es können nur Maximalwerte und keine Minimalwerte für die einzelnen Merkmale angegeben werden, d.h. es wird nicht einmal die Leistungsfähigkeit eines Hyperquader-Klassifikators erreicht.
- Für jede Klasse ist ein bestimmtes Bild einzeln zu analysieren.
- Aufgrund der vorigen zwei Schritte kann ein bestimmtes Objekt somit im Maximalfall zu allen Klassen gehören.
- Der Benutzer muss für die Operationen entsprechende Grundverständnisse der Bildverarbeitung besitzen und die dem Programm zugehörige Skriptsprache erlernen.

2 Methoden und Algorithmen

In Abbildung 3 sind die groben Verarbeitungsschritte dargestellt, welche verwendet werden, um aus den Radarbildern die gewünschten Ergebnisse zu berechnen. Als Ergebnisse seien hierbei die erkannten Objekte mit zugehöriger Klassifikationsinformation verstanden.

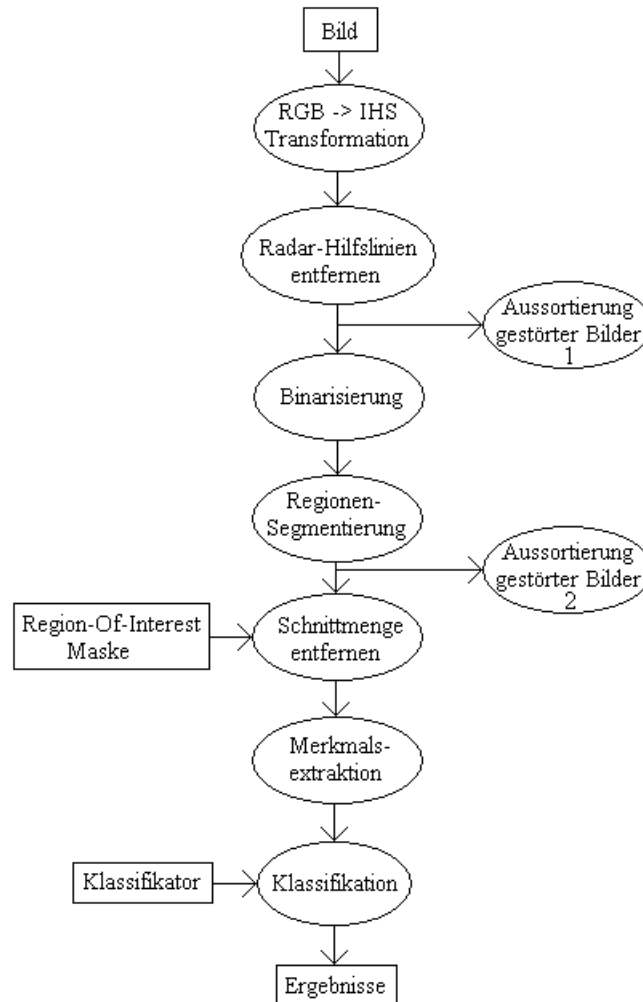


Abb. 3: Ablaufplan der automatischen Bildanalyse

2.1 RGB-IHS - Transformation

Die RGB-Bilder werden im ersten Schritt in den IHS-Farbraum konvertiert, wobei IHS für Intensity, Hue und Saturation steht. Für die weiteren Verarbeitungsschritte werden nur noch der Intensity-Kanal, sowie der Hue-Kanal, d.h. Intensität und Farbwert, benutzt. Da die Radarsignale alle in gelber Farbe und unterschiedlicher Helligkeit/Intensität dargestellt werden, bietet es sich an, diese Farbraum-

Transformation durchzuführen. Ein weiterer Vorteil ist, daß die nachfolgenden Operationen nur noch auf zwei statt der ursprünglichen drei Bildkanäle erfolgen. Der Algorithmus wurde auf effiziente Weise nach [Burger,Burge 2005] implementiert. Nach der Konvertierung gilt : $I, H, S \in [0, 1]$.

Listing 1: Algorithmus RGBtoIHS

MAX = max(R,G,B)	1
MIN = min(R,G,B)	2
if (MAX = MIN) then H = 0	3
else	4
{	5
if (R = MAX) then H = (G - B)/(MAX - MIN)	6
if (G = MAX) then H = (B - R)/(MAX - MIN)+2	7
if (B = MAX) then H = (R - G)/(MAX - MIN)+4	8
if (H<0) then H = H+6	9
}	10
H = H/6	11
	12
if (MAX = 0) then S = 0	13
else S = (MAX - MIN)/MAX	14
	15
I = MAX/255	16

2.2 Radar-Hilfslinien entfernen

Es kann nicht ausgeschlossen werden, daß in den Radarbildern Hilfslinien, wie z.B. Höhenlinien, auftreten, und diese vom Radargerät 'über' die zu klassifizierenden Objekte gezeichnet werden. Ein Beispiel hierfür ist in 4 zu sehen. Es kann also vorkommen, dass solche Hilfslinien Objekte 'durchtrennen'. Daher wurde eine Methode implementiert, um bei Auftreten solcher Hilfslinien eventuell durchtrennte Objekte wieder zusammenzufügen. Da die Linienstärke immer genau einen Pixel beträgt, fällt der zugehörige Algorithmus denkbar einfach aus :

Es wird für jeden Bildpunkt (x, y) überprüft, ob seine Farbe einer (im Programm manuell definierbaren) „Hilfslinienfarbe“ entspricht. Falls ja, so werden nacheinander seine Nachbar-Bildpunkte $(x, y + 1)$, $(x + 1, y)$, $(x, y - 1)$ und $(x - 1, y)$ überprüft, ob diese eine Nicht-Hilfslinienfarbe besitzen. Ist dies der Fall, so bekommt der Bildpunkt (x, y) die Farbe des gefundenen Bildpunktes zugewiesen. Die gewählte Reihenfolge der Überprüfung stellt sicher, daß bei horizontalen bzw. vertikalen Hilfslinien nicht einfach der Anfangspunkt vor dieser Linien sukzessive deren

Bildpunkte ersetzt (da das Bild zeilenweise von links oben beginnend durchlaufen wird).

Listing 2: Algorithmus ReplaceColors

```

for all (x,y) of RGB_Image do 1
{ 2
  if RGB_Image(x,y) has "Hilfslinienfarbe" then 3
  { 4
    if lower pixel RGB_Image(x,y+1) has valid color then 5
    RGB_Image(x,y) = RGB_Image(x,y+1) 6
    else if right pixel RGB_Image(x+1,y) has valid color then 7
    RGB_Image(x,y) = RGB_Image(x+1,y) 8
    else if upper pixel RGB_Image(x,y-1) has valid color then 9
    RGB_Image(x,y) = RGB_Image(x,y-1) 10
    else if left pixel RGB_Image(x-1,y) has valid color then 11
    RGB_Image(x,y) = RGB_Image(x-1,y) 12
    else 13
    RGB_Image(x,y) = BackgroundColor 14
  } 15
} 16

```

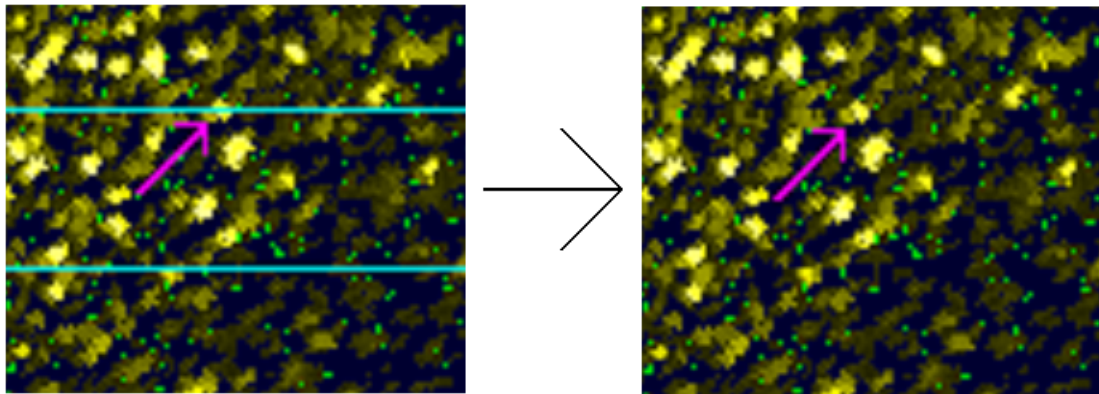


Abb. 4: Beispiel für Hilfslinien-Entfernung : Zu sehen sind viele Radarsignale in gelb, sowie eingeblendete Höhenlinien des Radars in blau. Der Pfeil deutet auf eine veranschaulichende Position.

2.3 Bildstörungen, Aussortieren gestörter Bilder

Besondere Umwelteinflüsse wie z.B. Regen oder große Insektenschwärme sind ebenfalls zu berücksichtigen, da diese gegebenenfalls ein komplettes Radarbild für die

Auswertung unbrauchbar machen. Ein Beispiel für Regeneinflüsse auf Radarbilder ist in Abbildung 5 dargestellt.

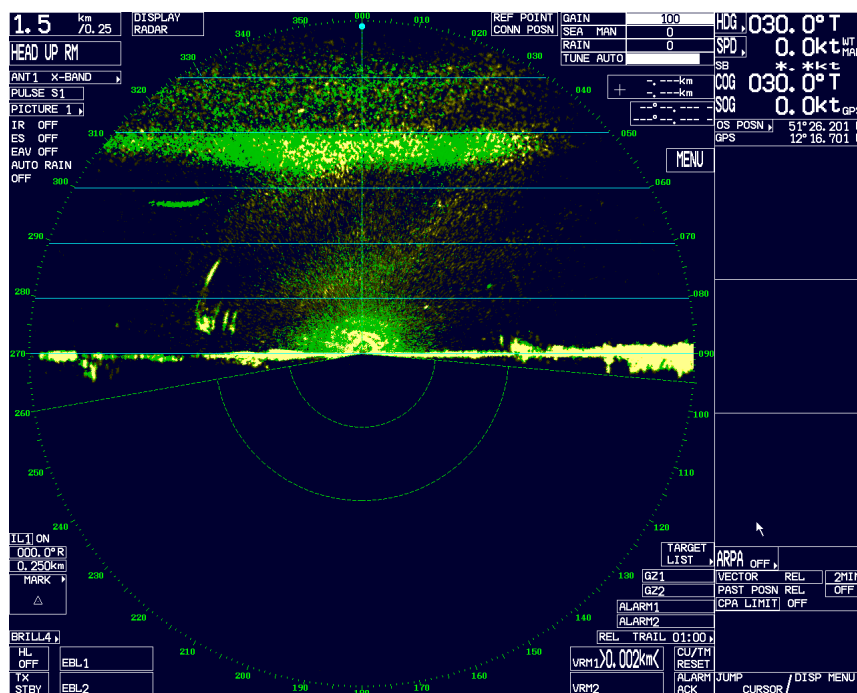


Abb. 5: Beispiel für Regen : Die vertikal verlaufende Anhäufung von Radarsignalen im oberen Bildbereich entspricht einer Regenwolke

Für das Herausfiltern dieser Bilder wurden zwei, vom Benutzer festzulegende Kriterien implementiert :

Das erste Kriterium ist eine Obergrenze von erkannten Regionen in einem Bild (*Maximal number of objects*). Werden in einem Bild also z.B. 400 Regionen erkannt (segmentiert) und liegt die Obergrenze z.B. bei 80 Regionen, so wird die weitere Analyse des Bildes abgebrochen und das Bild als gestört gekennzeichnet.

Das zweite Kriterium ist eine Obergrenze für diejenigen Bildpunkte, welche die Farben der Radarsignale, oder die der Tracking-Informationen annehmen (*Maximal Interference-Percentage*). Liegt diese Obergrenze z.B. bei 5% der Bildfläche, die Radarsignale und Tracking-Informationen nehmen aber 20% der Bildfläche ein, so wird auch hier die weitere Analyse des Bildes abgebrochen und das Bild als gestört gekennzeichnet.

Die Informationen über diese Entscheidungen werden in die Endergebnisse mit aufgenommen, so daß der Benutzer diese Entscheidung gegebenenfalls revidieren kann. Das erste Kriterium wird in Abbildung 3 im Schritt „Aussortierung gestörter Bilder 1“ berechnet, das zweite Kriterium im Schritt „Aussortierung gestörter Bilder 2“. Die Werte für die jeweiligen Obergrenzen werden im Programm während der

Trainingsphase (siehe Abschnitt 4.1.1) vom Benutzer festgelegt.

2.4 Binarisierung

Entsprechend der während der Trainingsphase (siehe Abschnitt 4.1.1) definierten Intervalle für den Farbwert der Radarsignale und deren Intensität wird das 2-Kanal-Bild nun binarisiert.

Listing 3: Algorithmus Binarisierung

for all (x,y) of IH_Image do	1
{	2
if (Intensity(x,y)−ObjectColorIntensity ≤ I_Tolerance) and	3
(Hue(x,y)−ObjectColorHue ≤ H_Tolerance) then	4
binaryImage(x,y) = 1	5
else	6
binaryImage(x,y) = 0	7
}	8

2.5 Regionen-Segmentierung

Um im Binärbild die zusammenhängenden Regionen zu berechnen, wird nun eine Regionen-Segmentierung durchgeführt. Für diese wurde ein sehr effizienter Algorithmus aus [Burger,Burge 2005] entnommen. Dieser ist eine Kombination aus sequentieller und rekursiver Berechnung und hat den Vorteil, daß auch die Konturen der Regionen gleich mit berechnet werden.

2.6 Statische Signale, Region-Of-Interest-Maske

Wie in Abbildung 2 und 5 zu erkennen ist, gibt es vor allem in Bodennähe erhebliche (unerwünschte) Radar-Echos. Die Bodennähe ist in den zwei angegebenen Bildern als horizontale, gelbe, linienartige Struktur in der Bildmitte erkennbar. Da diese Signale in jedem Bild mit großer Konstanz auftreten, seien diese Signale im folgenden als statische Signale bezeichnet. Sie resultieren von Reflexionen des Bodens, Bäumen, Straßen, etc. und sind im Nahbereich des Radarstandortes besonders stark. Der Radarstandort befindet sich in den Bildern in der Mitte des von Gradzahlen umrahmten Abtastbereiches. Eine sogenannte Region-Of-Interest (ROI) Maske wird verwendet, um diese statischen Signale auszublenden. Diese Maske ist ein editierbares Schwarz-Weiß-Bild, welches vom Benutzer für den jeweiligen Radar-Standort angefertigt werden sollte. Ein Beispiel für eine solche ROI-Maske ist in Abbildung 6 gegeben. Hierbei ist vom Benutzer darauf zu achten, daß die

auszublenenden Bereiche möglichst klein gewählt werden, um nicht unnötig viele Vogelsignale mit auszublenden. Im Programm werden nun alle segmentierten Regionen aus dem Auswertungsprozeß entfernt, die sich mit ausgeblendeten Bereichen der ROI-Maske überschneiden. Dieser Ansatz wurde gewählt, da sich bei einer AND-Verknüpfung der Maske mit dem binarisierten Radarbild die ursprüngliche Form und Größe von Objekten, welche teilweise in diese statischen Regionen hineinragen, nicht mehr rekonstruieren läßt, und diese Rest-Objekte somit nicht mehr verlässlich auswertbar sind.



Abb. 6: Beispiel für eine ROI-Maske

3 Klassifikation

Nachdem die zusammenhängenden Regionen im Bild berechnet wurden, gilt es nun aus diesen Merkmale zu gewinnen, anhand derer die Regionen anschließend bestimmten Objektklassen zu geordnet werden. Diesen Prozeß bezeichnet man als Klassifikation.

3.1 Objektklassen

Als zu unterscheidende Objektklassen existieren im Programm

$$\{ \text{'Bird'}, \text{'Plane'}, \text{'NearPlaneObject'}, \text{'SomethingElse'} \},$$

wobei der Benutzer allerdings selbst entscheiden kann, welche dieser Klassen er überhaupt benutzen will. Für die vorliegenden Daten ist es sinnvoll, alle diese Klassen zu verwenden und alle Güte-Betrachtungen in dieser Arbeit wurden unter Einbeziehung all dieser Klassen erstellt. Die Elemente der Klasse 'NearPlaneObject' werden nicht zur Klasse 'SomethingElse' gezählt, um eine Abgrenzung der durch Flugzeuge erzeugten Störsignale mit den anderen restlichen Signalen zu erhalten.

3.2 Aufbau des Klassifikators

Die Klassifikation erfolgt mittels eines 2-stufigen Klassifikators (siehe Abbildung 7).

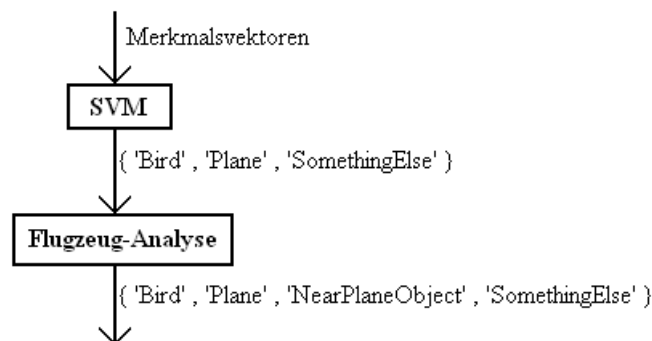


Abb. 7: 2-stufiger Klassifikator

Dieser 2-stufige Aufbau resultiert aus der Art und Weise, wie das Radargerät Flugzeug-Signale darstellt. Auf diese wird in Abschnitt 3.6 gesondert eingegangen.

3.3 Klassifikationsgüte

Um die Klassifikationsgüten zu berechnen, werden vier Gütemaße herangezogen, die im folgenden definiert werden :

Zunächst bezeichne gemäß der gebräuchlichen Verwendung für jede Klasse :

- TP (*true positives*)
die Anzahl korrekt klassifizierter positiver Elemente
- TN (*true negatives*)
die Anzahl korrekt klassifizierter negativer Elemente
- FP (*false positives*)
die Anzahl inkorrekt klassifizierter positiver Elemente
- FN (*false negatives*)
die Anzahl inkorrekt klassifizierter negativer Elemente

Hieraus ergeben sich für jede Klasse die zu berechnenden Gütemaße :

- Sensitivität sn (Rate korrekt klassifizierter positiver Elemente)

$$sn = \frac{TP}{TP + FN} \quad , \quad sn \in [0, 1]$$

- Spezifität sp (Rate korrekt klassifizierter negativer Elemente)

$$sp = \frac{TN}{TN + FP} \quad , \quad sp \in [0, 1]$$

- G-Mean gm (geometrisches Mittel aus Sensitivität und Spezifität)

$$gm = \sqrt{sn \cdot sp} \quad , \quad gm \in [0, 1]$$

- Korrelationskoeffizient cc (hier binärer Spezialfall - siehe [Haufe 2005])

$$cc = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FN)(TP + FP)(TN + FP)(TN + FN)}} \quad , \quad cc \in [-1, 1]$$

Im weiteren Verlauf werden jedoch nur noch die auf $[0, 1]$ normierten Werte betrachtet, um einheitliche Intervalle zu erhalten :

$$cc_N = \frac{cc + 1}{2}$$

3.4 Merkmale

Um aus den Bilddaten möglichst niedrigdimensionale, jedoch aussagekräftige Merkmalsvektoren zu gewinnen, werden für jedes erkannte Objekt die folgenden 7 Merkmale berechnet :

1. Flächeninhalt

Summe der zu einer Region gehörenden Bildpunkte.

$$area = \sum_{(x,y) \in Region} 1$$

2. Mittlere Helligkeit

Intensitäts-Aufsummierung über alle Bildpunkte der Region, und Mittelung über die Fläche.

$$\overline{intensity} = \frac{\sum_{(x,y) \in Region} intensity(x,y)}{area}$$

3. Distanz zum Radarstandort

Die Distanz *dist* entspricht dem euklidischen Abstand zwischen Objekt-Schwerpunkt (m_x, m_y) und dem in der Trainingsphase (siehe Abschnitt 4.1.1) definierten Radarstandort (r_x, r_y) .

$$dist = \sqrt{(m_x - r_x)^2 + (m_y - r_y)^2}$$

4. Kompaktheit

$$compactness = \frac{4 \cdot \pi \cdot area}{circumference^2}$$

Mit *circumference* = Länge der äußeren Kontur in Pixel-Einheiten.

5. Objektschärfe

Aufsummierung der Anzahl von Bildpunkten der angrenzenden Umgebung eines Objekts, welche im selben Farbwert-Toleranzbereich liegen, deren Helligkeit jedoch unterhalb der Helligkeits-Toleranz für die Objekte liegt. Anschließende Mittelung über die Länge des äußeren Objektrandes (äußere Kontur). Hierzu sei zunächst die angrenzende Umgebung eines Objektes als all jene Bildpunkte definiert, deren (euklidische) Distanz zum äußeren Rand des Objekts $\leq d \in \mathbb{R}$ ist. Die Berechnung dieser Objekt-Umgebung erfolgt durch Dilatation des binären Objektes und anschließender XOR Verknüpfung des

dilatierten und des Original-Objektes. Ein erklärendes Beispiel ist in Abbildung 8 zu sehen. Die Idee hierbei war, daß Vögel im Vergleich zu Insektenschwärmen oder Wolken ein sehr 'scharfes' Signal erzeugen sollten, und Insektenschwärme, bzw. Wolken am Rand eher 'ausfransen'.

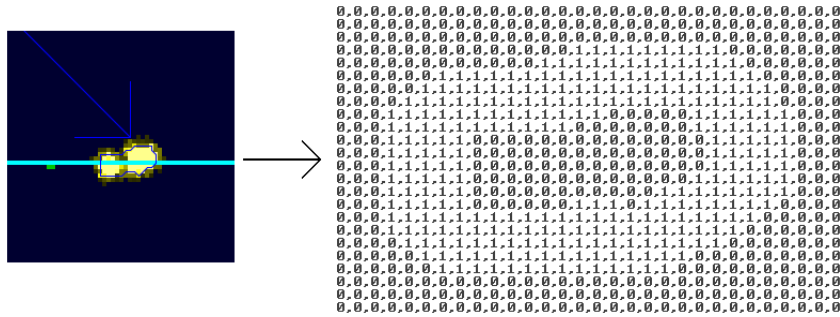


Abb. 8: Beispiel einer Objekt-Umgebung ($1 \hat{=}$ Umgebung)

6. Umgebungs-Objekte

Aufsummierung der Bildpunkte der angrenzenden Umgebung eines Objekts, welche zu anderen erkannten Objekten gehören. Dieser Wert stellt ein Maß für die Nähe und Anzahl anderer Objekte zum betrachteten Objekt in unmittelbarer Umgebung dar. Die Zählung der Bildpunkte erfolgt ebenfalls in der oben definierten Objekt-Umgebung und wird auch über die Länge des äußeren Objektrandes gemittelt.

7. Umgebungs-Trackinginformation

Aufsummierung der Bildpunkte der angrenzenden Umgebung eines Objekts, welche die Farbe der Trackinginformationen besitzen. Dieser Wert stellt ein Maß für die Tracking-Intensität in unmittelbarer Umgebung dar. Die Zählung der Bildpunkte erfolgt ebenfalls in der oben definierten Objekt-Umgebung und wird auch über die Länge des äußeren Objektrandes gemittelt. (Zur Definition der 'Trackinginformationen' siehe Abschnitt 1.1.)

Merkmalsauswahl :

Um zu ermitteln, welche der obigen Merkmale für das Klassifikationsproblem am besten geeignet sind, wurde von einer Stichprobe mit 576 Radarbildern ausgehend, für alle möglichen Kombinationen der Merkmale ein Klassifikator trainiert und die resultierenden Gütemaße ermittelt. Bei den obigen 7 Merkmalen ergeben sich somit $2^7 - 1 = 127$ Möglichkeiten. Dieses Durchtesten der Kombinationsmöglichkeiten erfolgt automatisiert und wird in Abschnitt 4.1.1 näher beschrieben. Um aus den 16 Gütemaßen pro Kombination (je 4 Gütemaße für 4 Klassen) aussagekräftige und

vergleichbare Werte zu gewinnen, wurden nur die zwei wichtigsten, da mit Abstand am häufigsten auftretenden, Klassen 'Bird' und 'SomethingElse' betrachtet. Von diesen wiederum wurde für das Gütemaß $G\text{-Mean}$ und für die $Korrelationskoeffizienten$ das geometrische Mittel gebildet.

$$\begin{aligned} \text{geom_mean}(G - \text{Mean}) &= \sqrt{\text{gm}(\text{Bird}) \cdot \text{gm}(\text{SomethingElse})} \\ \text{geom_mean}(CC) &= \sqrt{\text{cc}(\text{Bird}) \cdot \text{cc}(\text{SomethingElse})} \end{aligned}$$

Um die eingangs erwähnten 127 Kombinationsmöglichkeiten der Merkmale zu vergleichen, werden die zwei geometrischen Mittel für jede dieser Kombinationen berechnet. In Tabelle 1 sind die 7 besten Werte sortiert nach $\text{geom_mean}(G - \text{Mean})$ und in Tabelle 2 die 7 besten Werte sortiert nach $\text{geom_mean}(CC)$ aufgetragen.

Merkmalskombination	$\text{geom_mean}(G - \text{Mean})$	$\text{geom_mean}(CC)$
1010101	0.846851886	0.827523421
1011101	0.846487087	0.827497245
1010111	0.840891307	0.824302723
1011100	0.839249664	0.826112734
1010110	0.837832172	0.819957850
1110101	0.836955290	0.821462497
1011111	0.836935757	0.822991555

Tab. 1: Merkmalsauswahl: Die besten 7 Merkmalskombinationen bezüglich $\text{geom_mean}(G - \text{Mean})$ absteigend sortiert. Die binäre Darstellung der Merkmalskombinationen ist so zu verstehen, daß eine 1 dem Verwenden eines Merkmals entspricht und eine 0 bedeutet, daß dieses Merkmal nicht verwendet wird. Die Sortierung der Merkmale erfolgt hierbei gemäß der Definitions-Reihenfolge am Anfang dieses Abschnittes. Die am weitesten links stehende Binärziffer entspricht dem Merkmal *Flächeninhalt* und die am weitesten rechts stehende Binärziffer entspricht dem Merkmal *Umgebungs-Trackinginformation*.

Merkmalskombination	$\text{geom_mean}(G - \text{Mean})$	$\text{geom_mean}(CC)$
1010101	0.846851886	0.827523421
1011101	0.846487087	0.827497245
1011100	0.839249664	0.826112734
1010111	0.840891307	0.824302723
1011111	0.836935757	0.822991555
1110101	0.836955290	0.821462497
1110111	0.832044586	0.821448450

Tab. 2: Merkmalsauswahl: Die besten 7 Merkmalskombinationen bezüglich $\text{geom_mean}(CC)$ absteigend sortiert.

Zu erkennen ist, daß die ersten zwei Zeilen der beiden Tabellen identisch sind. Die zwei entsprechenden Merkmalskombinationen liefern also bei beiden Gütemaßen die besten Werte.

Aufgrund dieser Betrachtungen wird für die vorliegende Aufgabenstellung die Empfehlung ausgegeben, die Klassifikatoren unter Verwendung der Merkmale *Flächeninhalt*, *Distanz zum Radarstandort*, *Objektschärfe* und *Umgebungs-Trackinginformation* zu trainieren und zu verwenden.

Die Merkmale können aber natürlich je nach Aufgabenstellung unterschiedlich gewählt werden - siehe Abschnitt 4.1.1.

Desweiteren besitzen die einzelnen Merkmale unterschiedliche Wertebereiche und werden daher vor dem Training des Klassifikators jeweils auf das Intervall $[0, 1]$ normiert. Die zugehörigen Normierungsinformationen werden gespeichert, um beim Klassifizieren mittels eben dieses Klassifikators die Komponenten der zu klassifizierenden Merkmalsvektoren ebenfalls im gleichen Maße zu normieren.

3.5 SVM-Klassifikator

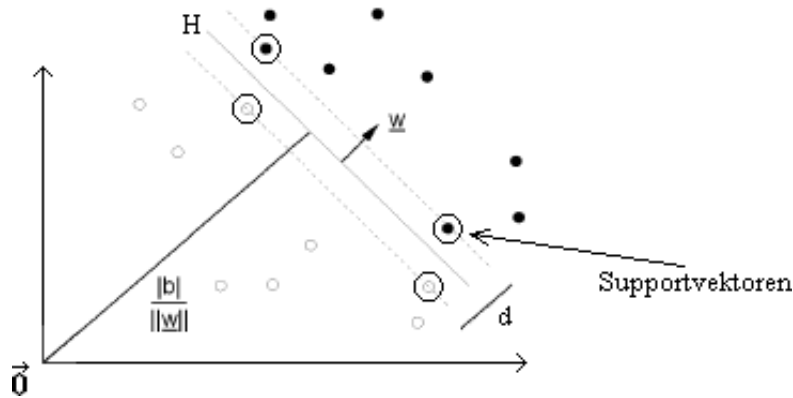
3.5.1 Theorie

Für die Klassifikation der aus den Radarbildern gewonnenen Merkmalsvektoren wurde eine Support-Vektor-Maschine (SVM) [Abe 2005], [Theodoridis 1999] gewählt. Der Grund für die Wahl dieser statistischen Lernmaschinen liegt zum einen in der guten Klassifikationsleistung gegenüber anderen Klassifikatoren und zum anderen im Vorhandensein einer frei verfügbaren, einfach zu implementierenden und leistungsfähigen Open-Source-Bibliothek : *libsvm* [Chang, Lin 2001]. Mit Verweis auf die oben angeführte Literatur werden an dieser Stelle die Hintergründe der SVM nur grob erläutert :

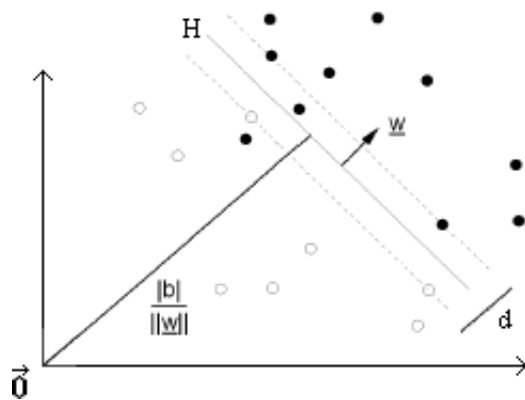
Liegt ein linear separables Zweiklassenproblem vor (siehe Abbildung 9), so ist eine Hyperebene $H(\underline{w}, b)$ mit $\underline{w}^T \underline{x} + b = 0$ gesucht, welche diese zwei Klassen trennt. Für alle Merkmalsvektoren x_k mit zugehöriger Klasse $y_k \in \{-1, +1\}$ gilt

$$\begin{aligned} \underline{w}^T \underline{x}_k + b &\geq 0 \quad \forall y_k = +1 \\ \underline{w}^T \underline{x}_k + b &< 0 \quad \forall y_k = -1 \end{aligned}$$

Vereinheitlicht geschrieben also : $y_k(\underline{w}^T \underline{x}_k + b) \geq 0$.

Abb. 9: Linear separables Zweiklassenproblem im \mathbb{R}^2

Da unendlich vieler solcher Hyperebenen existieren, ist das Problem nicht eindeutig lösbar. Der Ansatz ist nun, diejenige trennende Hyperebene auszuwählen, welche die minimale quadratische Norm $\|w\|_2^2$ besitzt, so daß gleichzeitig die Nebenbedingungen $y_k(w^T x_k + b) \geq 1$ erfüllt sind (korrekte Klassifizierung und leere Randzone). In die Berechnung von w geht meist nur eine Teilmenge der x_k ein, nämlich diejenigen x_k , die genau auf der Grenze der Randzone liegen, d.h. deren kürzester Abstand zur Hyperebene $\frac{2}{\|w\|}$ beträgt (siehe eingekreiste x_k in Abbildung 9). Dies sind die sogenannten Supportvektoren. Im allgemeinen wird natürlich selten ein linear separables Problem vorliegen (siehe Abbildung 10).

Abb. 10: Linear nichtseparables Zweiklassenproblem im \mathbb{R}^2

In diesem Fall werden die Nebenbedingungen durch Einführung von Schlupfvariablen $\xi_k \geq 0$ gelockert :

$$y_k(w^T x_k + b) \geq 1 - \xi_k$$

Das ursprüngliche Minimierungsproblem wird nun durch einen Strafterm modifiziert, mit dem Hintergrund, die Randverletzungen unter obigen Nebenbedingungen

zu minimieren :

$$\|w\|^2 + C \cdot \sum \xi_k$$

Hierbei ist C eine a-priori vorgegebene Gewichtskonstante.

Der oben beschriebene Algorithmus klassifiziert die Daten mit Hilfe einer linearen Funktion. In vielen Anwendungen ist dies aber unzureichend, da oft nichtlineare Klassengrenzen gewünscht sind. Bildet man die Daten in einen Raum höherer Dimension ab,

$$\Phi : \mathbb{R}^{d_1} \rightarrow \mathbb{R}^{d_2} \quad , \quad d_1 < d_2$$

so erhöht sich nach dem Theorem von Cover [Schölkopf, Smola 2001] die Anzahl möglicher linearer Trennungen. Um die Berechnungskomplexität hierbei in Grenzen zu halten, nutzt man aus, daß die Datenpunkte x_k im dualen Optimierungsproblem nur in Skalarprodukten eingehen. Durch Verwendung spezieller, sogenannter *Kernel*funktionen lassen sich diese Skalarprodukte eines hochdimensionalen Raumes implizit und somit schnell berechnen.

3.5.2 Wahl der SVM-Parameter

In dieser Arbeit kam die Bibliothek *libsvm* (siehe Abschnitt 5.1) zu Verwendung. Als *Kernel*funktion wurde der RBF-Kernel (Radiale Basisfunktion bzw. Gauß-Kernel) gewählt :

$$K(\underline{x}_i, \underline{x}_j) = \exp\left(-\frac{\|\underline{x}_i - \underline{x}_j\|^2}{\sigma^2}\right)$$

Dadurch ergab sich lediglich ein freier Parameter : die Standardabweichung σ . In der *libsvm*-Bibliothek erscheint dieser Parameter unter der Bezeichnung *gamma* mit der Beziehung

$$gamma = \frac{1}{\sigma^2}$$

Ein (für die Aufgabenstellung) geeigneter Wert dieses Parameters wurde experimentell ermittelt (siehe Abbildung 11 und 12). Hierzu wurden von einer Stichprobe mit 576 Radarbildern ausgehend Klassifikatoren mit den Werten $gamma = 2^{-10}, 2^{-9}, 2^{-8}, \dots, 2^7, 2^8$ trainiert, validiert und die entsprechenden Gütemaße aufgetragen. Die Validierung erfolgte, indem die Stichprobe halbiert wurde (in Trainings- und Testdaten), nur auf den Trainingsdaten trainiert wurde und auf den Testdaten validiert wurde. Um einen möglichst guten Wert für *gamma* zu ermitteln, wurde von den zwei aussagekräftigsten Gütemaßen *G-Mean* und *Korrelationskoeffizient* jeweils das geometrische Mittel über die Klassen 'Bird' und 'SomethingElse' gebildet und in Abbildung 12 aufgetragen.

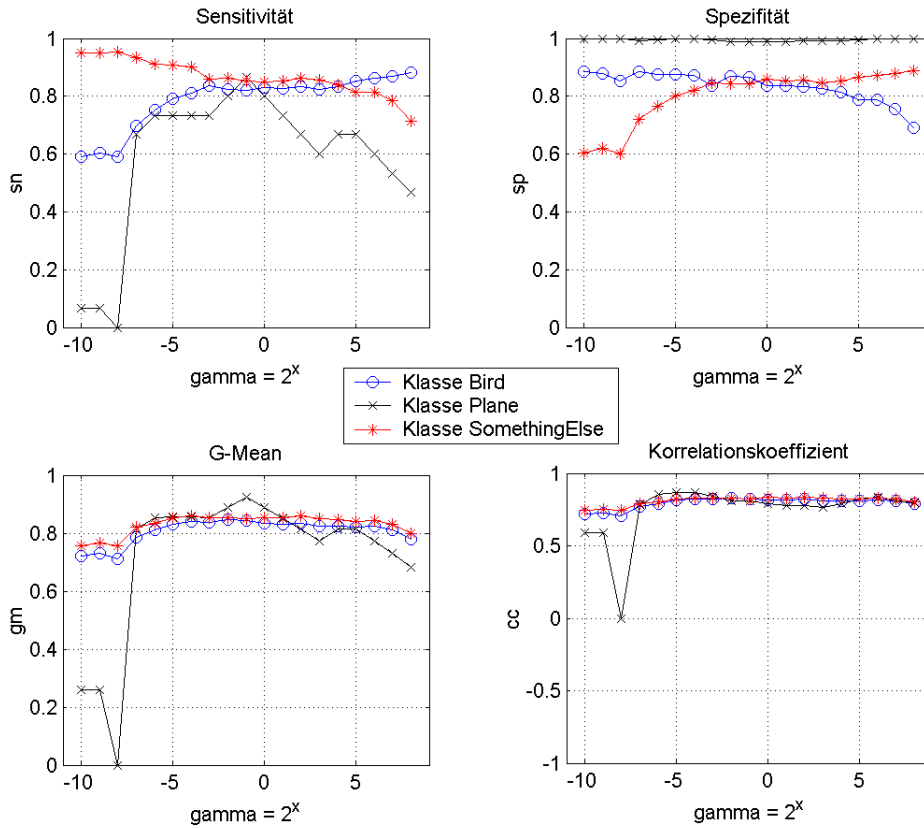


Abb. 11: Gütemaße der Klassifikation bei Variation von gamma , aufgetragen für die Klassen 'Bird', 'Plane' und 'SomethingElse'.

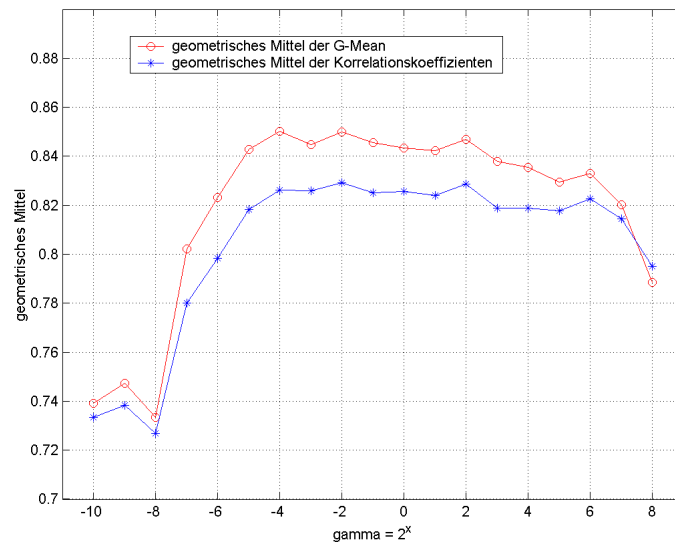


Abb. 12: Geometrische Mittel der Gütemaße $G\text{-Mean}$ und $Korrelationskoeffizient$ von den Klassen 'Bird' und 'SomethingElse'

In die Berechnungen wurden nur die zwei Klassen 'Bird' und 'SomethingElse' aufgenommen, da bei der Interpretation der Abbildungen 11 zu beachten ist, daß hinter dem Kurvenverlauf der Klasse 'Plane' lediglich 15 Flugzeug-Objekte stehen. Im Vergleich dazu enthielt die Stichprobe jedoch 896 Objekte der Klasse 'Bird' und 350 Objekte der Klasse 'SomethingElse'. Die Klasse 'NearPlaneObject' ist nicht mit aufgetragen, da sie nicht durch den SVM-Klassifikator, sondern erst in der zweiten Stufe erkannt wird.

geometrisches Mittel der G-Mean	zugehöriges γ	geometrisches Mittel der Korrelationskoeffizienten	zugehöriges γ
0.8501	2^{-4}	0.8291	2^{-2}
0.8498	2^{-2}	0.8287	2^2
0.8469	2^2	0.8263	2^{-4}

Tab. 3: Die jeweils drei besten Werte für die geometrischen Mittel der Gütemaße *G-Mean* und *Korrelationskoeffizient*

In Tabelle 3 wurden die drei besten Werte der geometrischen Mittel der Gütemaße zusammengestellt. Für das Gütemaß *G-Mean* ergab sich das Maximum bei $\gamma = 2^{-4}$ und für das Gütemaß *Korrelationskoeffizient* bei $\gamma = 2^{-2}$. Hieraus wurde letztlich $\gamma = 2^{-2} = 0.25$ ($\hat{=} \sigma = 2$) als Parameter für den RBF-Kernel gewählt.

Die *libsvm*-Bibliothek unterstützt desweiteren Multi-Klassen, d.h. sie ist nicht auf binäre Klassifikationsprobleme beschränkt. Dies ist auch nötig, da der SVM-Klassifikator 3 Klassen differenzieren soll ('Bird', 'Plane', 'SomethingElse'). Die bereits implementierten Methoden kombinieren mehrere binäre SVM miteinander (*one-against-one*-Methode) und sind in [Chang, Lin 2001] näher beschrieben.

3.5.3 Unausgeglichenheit der Daten

Ein weiteres Problem war die Unausgeglichenheit der Daten, da in den Radarbildern Flugzeug-Objekte viel seltener auftreten als z.B. Vogel-Objekte. Eine Abschätzung der Häufigkeit von diesen sogenannten Flugzeug-Bildern ist schlecht anzugeben, da das Radar-Erscheinungsbild starken Jahreszeit-Schwankungen, Wochentags-Schwankungen und Uhrzeit-Schwankungen unterliegt. Nach manueller Durchsicht wird die Häufigkeit von Flugzeugbildern auf ungefähr 0.5%–1% geschätzt. Aufgrund dieser Unausgeglichenheit der Daten verschiebt der SVM-Klassifikator die Klassengrenzen zuungunsten der unterrepräsentierten Klasse (siehe [Haufe 2005]). Um dem entgegenzuwirken, wurden die Merkmalsvektoren der unterrepräsentierten Klassen mittels *Oversampling* künstlich auf die Mächtigkeit der am häufigsten

vorkommenden Klasse vervielfältigt. Hierbei werden Elemente der unterrepräsentierten Klassen dupliziert, bis das gewünschte Stichprobenverhältnis erreicht ist.

3.6 Flugzeugsignale

Flugzeuge erzeugen auf dem Radarschirm ein im Vergleich zu Vögeln sehr großes Radarsignal, welches jedoch in Form- und Größe äußerst vielfältig und nicht zwingend zusammenhängend ist. Durchfliegen Flugzeuge den Radarbereich nicht frontal, sondern seitlich, so erzeugen sie aufgrund der Geschwindigkeit und Beschaffenheit viele, an Größe schwankende Radarsignale entlang der Flugbahn (siehe Abbildung 13).

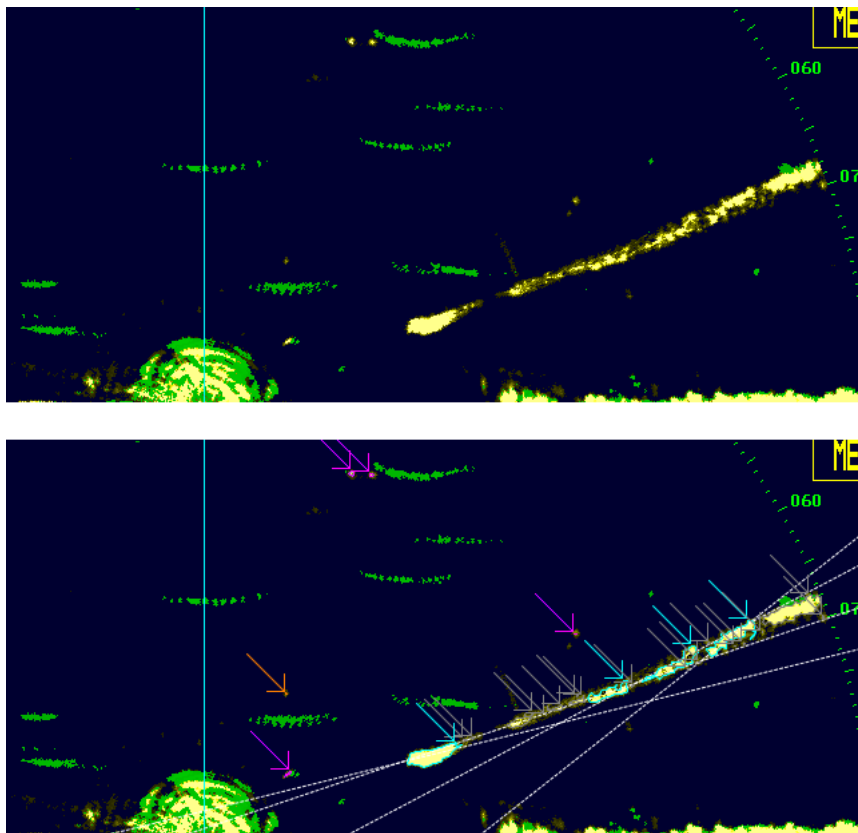


Abb. 13: Oben : Beispiel eines Bildes mit Flugzeug-Signal ; Unten : Dasselbe Bild mit Visualisierung der Klassifikations-Ergebnisse. Zu sehen ist, dass alle der zusätzlich von dem Flugzeug erzeugten Signale grob auf einer Geraden liegen und somit durch den oben besprochenen Algorithmus als 'NearPlaneObject' (grau) klassifiziert werden - wenn nicht schon als Flugzeug klassifiziert. Die eingezeichneten Linien entsprechen den Orientierungen der Flugzeug-Objekte und die Pfeile geben die Klassenzugehörigkeit an (Farblegende : 'Bird'=lila , 'Plane'=hellblau , 'NearPlaneObject'=grau , 'SomethingElse'=orange). Die Farblegende entspricht aus Darstellungsgründen nicht der des Programms.

Das stärkste Radarsignal (bzw. auch das darauf folgende erste Echo) kann aufgrund der Merkmale (siehe Abschnitt 3.4) in den meisten Fällen eindeutig der Klasse 'Plane' zugeordnet werden. Die darauf folgenden Radarsignale sind in ihren Merkmalen jedoch mehrdeutig interpretierbar und würden in den seltensten Fällen ebenfalls als 'Plane' klassifiziert werden. Der hier verwendete Lösungsansatz sieht vor, die Flugrichtung der Flugzeug-Objekte zu berechnen und entlang dieser die vom Flugzeug zusätzlich erzeugten Signale auszusortieren. Da die Flugzeuge zumeist ein längliche Form besitzen wurde als Ansatz die Berechnung die Orientierung ϕ der Flugzeug-Objekte durch sogenannte Momente gewählt.

Es gilt :

$$\tan\phi = 2 \frac{\mu_{11}}{\mu_{20} - \mu_{02}}$$

mit den zentralen Momenten

$$\mu_{ij} = \sum_{(x,y) \in Region} (x - \bar{x})^i \cdot (y - \bar{y})^j \cdot b(x, y)$$

und (\bar{x}, \bar{y}) = Schwerpunkt der Region, sowie $b(x, y)$ = Binärwert des zugehörigen Binärbildes im Bildpunkt (x, y) .

Anhand dieses Winkels ϕ wird nun eine Gerade durch den Flugzeug-Schwerpunkt (m_x, m_y) gelegt, welche die Flugbahn des Flugzeuges approximiert. Dieser Ansatz geht von Flugzeugen mit gerader Flugbahn aus. Tatsächlich ist es so, daß aufgrund der geringen Einsatz-Reichweite des Radars auf den Radarbildern ausschließlich startende oder landende Flugzeuge auftreten, deren Flugbahn in dieser Höhe annähernd eine Gerade ist. Nach dem Klassifikationsprozeß der SVM (siehe Abschnitt 3.5), werden nun alle Objekte, die einen bestimmten Minimalabstand $dist$ zu dieser Flugbahn unterschreiten, der gesonderten Klasse 'NearPlaneObject' (flugzeugnahe Objekte) zugeordnet (siehe Abbildung 14). Dieser Minimalabstand $dist$ kann vom Benutzer selbst gewählt werden und ist zusammen mit allen anderen wählbaren Parametern in 4.1.1 näher erläutert.

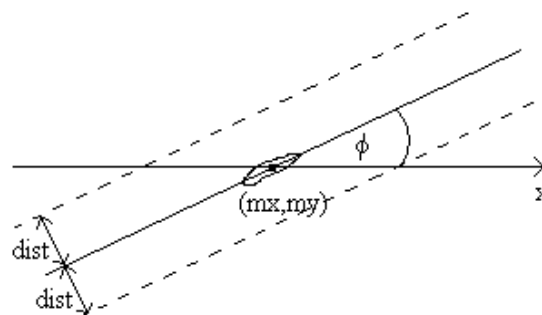


Abb. 14: Beispiel zur Flugbahnberechnung eines Flugzeug-Objektes

Diese Besonderheit ist auch im Validierungsprozeß (siehe Abschnitt 4.1.1) berücksichtigt. Objekte der Klasse 'NearPlaneObject' werden nicht in den SVM-Klassifikator aufgenommen, sondern nach dessen Klassifikationsprozeß beim Vorhandensein eines Flugzeuges (Klasse 'Plane') durch obiges Verfahren ermittelt. Ein komplettes Beispiel ist in Abbildung 13 dargestellt.

4 Systemaufbau

Als Ziel galt die Entwicklung eines möglichst vielseitigen und nicht nur auf die vorgegebenen Daten anwendbaren, sowie leicht und intuitiv zu bedienendem Programms.

4.1 Systemüberblick

Äußerst grob läßt sich das Programm in Trainings- und Anwendungsphase unterteilen. In der Trainingsphase werden vom Benutzer sämtliche Parameter des Programms festgelegt und Klassifikatoren anhand von Stichproben trainiert. In der Anwendungsphase analysiert das Programm Radarbilder mittels eines trainierten Klassifikators und gibt die Klassifikationsergebnisse aus.

4.1.1 Trainingsphase

Es sei zunächst die Trainingsphase näher erläutert: Diese läßt sich grob in folgende 6 Schritte unterteilen, durch welche der Benutzer sukzessive geleitet wird. Damit soll gewährleistet sein, daß das Training wohl durchdacht verläuft und keine zu treffenden Einstellungen vergessen werden. Auch kann der Trainingsprozeß an jeder beliebigen Stelle zur späteren Weiterbearbeitung zwischengespeichert werden.

In Abbildung 15 ist die Trainingsphase kurz gefasst dargestellt.

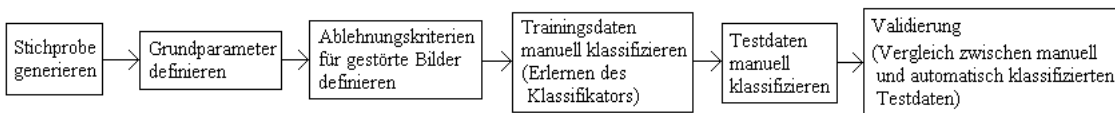


Abb. 15: Übersicht Trainingsphase

1. Generierung der Stichprobe

Der Benutzer wird aufgefordert einen möglichst umfangreichen Datensatz an Radarbildern anzugeben, aus welchem eine bestimmte Anzahl an Bildern (ebenfalls vom Benutzer wählbar) zufällig ausgewählt und in die Stichprobe übernommen wird (empfohlen 200 aufwärts, mindestens jedoch 40). Diese Stichprobe wird anschließend halbierend aufgeteilt in Trainings- und Testbilder.

An dieser Stelle sei schon einmal auf eine Besonderheit der Aufgabenstellung hingewiesen: Da bestimmte Arten von Bildern sehr selten vorkommen, aber dennoch in das Training mit aufgenommen werden müssen, muss der

Benutzer von diesen zwei Bilder-Arten (Bilder mit Flugzeugsignalen und Regen) je einen Datensatz zusammenstellen, so dass während der Stichproben-Generierung, zusätzlich zu den zufällig ausgewählten Bildern, auch aus diesen zwei extra Datensätzen eine bestimmte Menge an Bildern (siehe unten) zufällig ausgewählt und der Stichprobe hinzugefügt wird. So ist gewährleistet, dass auch diese selten vorkommenden Bild-Arten in der Trainings-Stichprobe vorhanden sind.

Für die Implementierung wurde festgelegt, daß 5% der Bilder der Stichprobe aus dem Datensatz der Bilder mit Flugzeugsignalen stammen (mindestens jedoch 4 Stück) und 10% aus dem Datensatz der Bilder mit Regen (mindestens jedoch 10 Stück). Diese Bilder werden ebenfalls halbiert auf Trainings- und Testdaten verteilt. Eine nähere Erklärung dieser zwei Bild-Arten findet sich unter Abschnitt 3.6, sowie Abschnitt 2.3.

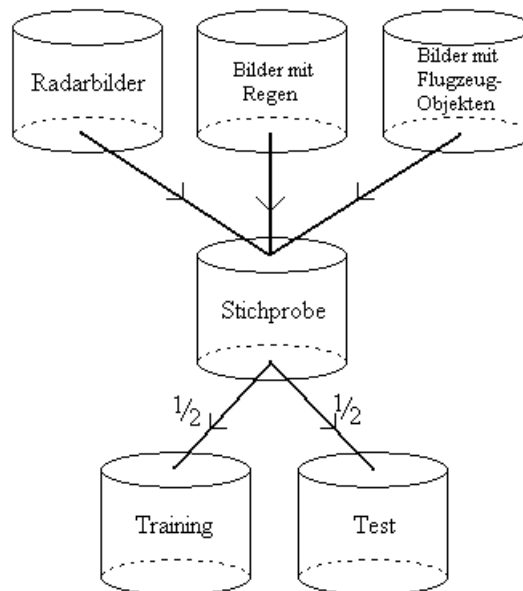


Abb. 16: Stichproben-Generierung

2. Grundparameter festlegen

In Schritt 2 definiert der Benutzer anhand der Trainingsbilder die grundlegenden Parameter :

- Farbwert und Intensität der zu analysierenden Objekte
- Farbe des Bildhintergrundes
- Farbe der Tracking-Informationen

- Toleranzbereiche in denen dieser Farbwert und Intensität variieren kann
- Hilfslinienfarbe des Radargerätes
Da nicht ausgeschlossen werden kann, dass in den Radarbildern Hilfslinien, wie z.B. Höhenlinien, auftreten, und diese vom Radargerät leider 'über' die Objekte gezeichnet werden, kann es somit vorkommen, dass solche Hilfslinien Objekte 'durchtrennen'. In Abschnitt 2.2 ist die verwendete Methode erläutert, um diese 'durchtrennten' Objekte wieder zusammenzufügen.
- Region-Of-Interest Maske (wird als Bilddatei ausgewählt) - siehe Abschnitt 2.6. Diese Maske wird außerhalb des Programms als Schwarz-Weiß-Bild in einem beliebigen Grafikprogramm erstellt, bzw. bearbeitet.
- Radarstandort im Bild (dieser ist über den gesamten Beobachtungszeitraum fix), sowie Entfernungsinformation (Verhältnis von Streckenlänge im Bild zu Streckenlänge in realer Welt). Der Benutzer definiert hierzu den Radarstandort in den Radarbildern und gibt zu einer wählbaren Strecke im Bild die zugehörige Entfernung in der realen Welt an. Hierdurch kann einer beliebigen Strecke im Bild eine Entfernung in der realen Welt zugeordnet werden.
- Minimalabstand (*Minimal distance*) von Objekten zu Flugzeug-Flugbahnen (siehe Abschnitt 3.6)
- Definition von Kriterien für schwierig zu entscheidende Bilder - nicht zu verwechseln mit dem Aussortieren gestörter Bilder in Abschnitt 2.3. Diese schwierig zu entscheidende Bilder werden in der Ergebnis-Ausgabe als „unsicher“ markiert, so daß der Benutzer diese manuell nachkontrollieren kann. Die 4 Kriterien hierbei sind : Maximalzahl an Objekten (*Small-ObjectsMaxNumber*), die eine festlegbare Größe unterschreiten (*Small-ObjectsSize*), sowie Maximalzahl an Objekten (*NearObjectsMaxNumber*) innerhalb eines festlegbaren Radius (*NearObjectsRadius*) um ein Objekt.
- Begrenzungs-Zeichenkette (*Delimiter*) der einzelnen Spalten für die Ausgabedateien (siehe Abschnitt 4.2)

3. Ablehnungskriterien für gestörte Bilder

In diesem Schritt legt der Benutzer die in Abschnitt 2.3 eingeführten Kriterien fest, anhand derer die Radarbilder auf Brauchbarkeit für die Auswertung untersucht werden (Obergrenze von erkannten Regionen in einem Bild und Obergrenze für diejenigen Bildpunkte, welche die Farben der Radarsignale, oder die der Tracking-Informationen annehmen). Dieser Schritt wird separat

vom vorigen Schritt behandelt, da er Informationen über die gegebenenfalls vorliegende ROI-Maske voraussetzt.

4. Markieren der Trainingsdaten

Im folgenden wird der Benutzer aufgefordert, die, mittels der in den vorigen Schritten bestimmten Parametern, erkannten Objekte in den Trainingsdaten per Hand zu klassifizieren. Mit diesen Objekten (bzw. deren Merkmalen) wird anschließend der SVM-Klassifikator (siehe Abschnitt 3.5) trainiert. Der Prozeß, mit welchem ausgehend von den auszuwertenden Bilddaten des Radargerätes die Objekte erkannt werden, wurde in Abschnitt 2 näher erläutert.

5. Markieren der Testdaten

Um die Güte des Klassifikators berechnen zu können, wird der Benutzer nun aufgefordert auch sämtliche in den Testdaten erkannte Objekte manuell zu klassifizieren. Diese Klassifizierungen werden für den späteren Vergleich mit den Ergebnissen des SVM-Klassifikators gespeichert.

6. Validierung

Im letzten Schritt der Trainingsphase gibt der Benutzer an, welche Merkmale für das Klassifikationsproblem verwendet werden sollen. Hierfür besteht die Option, sämtliche Kombinationsmöglichkeiten der Merkmale automatisiert zu testen und die resultierenden Klassifikationsgüten zur Betrachtung mittels eines Statistikprogramms zu speichern. Wurde schließlich eine Merkmalskombination durch Aktivieren der entsprechenden Kontrollkästchen aktiviert, so wird der erlernte Klassifikator auf die unklassifizierten Testdaten angewandt und die Klassifikationsergebnisse mit den manuell zugeordneten Klassenzugehörigkeiten der Testdaten aus dem vorigen Schritt verglichen. Aus diesem Vergleich werden die Verwechslungstabelle und die Gesamt-Klassifikationsrate (d.h. die Rate korrekt klassifizierter Objekte) berechnet und zur genaueren Analyse auch für jede Klasse die in Abschnitt 3.3 erwähnten Gütemaße. Schlußendlich wird dem Benutzer die Möglichkeit geboten, diesen soeben validierten Klassifikator für den späteren Anwendungsgebrauch zu speichern. Die Validierungsinformationen werden dabei für eventuelle nachträgliche Betrachtungen ebenfalls gespeichert.

4.1.2 Anwendungsphase

In der Anwendungsphase wählt der Benutzer einen bereits trainierten Klassifikator aus, sowie ein Verzeichnis mit den zu analysierenden Bildern. Desweiteren ist die Eingabe der Aufnahmezeit des ersten Bildes, sowie des Zeitintervalls zwischen

den einzelnen Aufnahmen vonnöten, um ausgehend von der Aufzeichnungszeit des ersten Radarbildes jedes weitere Radarbild einem exakten Zeitpunkt zuordnen zu können. Mit der Bezeichnung 'erstes Bild' ist an dieser Stelle das Bild mit dem lexikographisch kleinsten Dateinamen des zu analysierenden Verzeichnisses gemeint. Diese lexikographische Ordnung entspricht der tatsächlichen zeitlichen Bildabfolge, wenn die Bildaufnahme wie im vorliegenden Fall mit fortlaufender Nummerierung der Bilder erfolgt. Nach Beendigung der automatisch ablaufenden Bildauswertung werden die Ergebnisse tabellarisch aufgelistet und können gegebenenfalls manuell durchgesehen und überarbeitet werden. Die Ergebnisspeicherung erfolgt nach dem Schema aus dem folgenden Abschnitt 4.2. Dabei werden für jedes erkannte Objekt folgende Informationen gespeichert: Klassenzugehörigkeit, Datum, Uhrzeit, Flughöhe, Entfernung zum Radar, Helligkeit, Fläche, Koordinaten des Objektes im Radarbild, der Dateiname des zugehörigen Radarbildes und drei optionale Informationen : Tracking-Länge, Tracking-Winkel und Tracking-Geschwindigkeit. Diese letzten drei Werte können vom Benutzer anhand der Tracking-Informationen eines Objektes manuell im Programm ausgemessen werden. Die Flughöhe und die Entfernung zum Radar lassen sich durch die in Abschnitt 4.1.1 definierten Grundparameter berechnen.

4.2 Programmausgabe

Die Ergebnisse werden in einem Dateiformat gespeichert, welches ein einfaches Importieren in weiterführende Statistik-, bzw. Tabellenprogramme ermöglicht. Die Ausgabe erfolgt für alle analysierten Radarbilder zusammengefaßt in zwei Dateien: Die erste Datei (*Dateiname_Statistics.txt*) gibt für alle analysierten Radarbilder (und somit für eine bestimmte Uhrzeit) die Anzahl der erkannten Vögel, die Gesamtzahl der Objekte, etc. aus. Die zweite Datei (*Dateiname_SingleObjects.txt*) listet jedes erkannte Objekt mitsamt der zugehörigen Informationen auf - siehe Abschnitt 4.1.2. Die Dateiformate dieser zwei Dateien sind im folgenden grob in BNF (Backus-Naur-Form) aufgeführt :

```

< Datei_Statistics > ::= < Dateikopf > { < Daten > } * '#' < Anhang >
< Dateikopf >       ::= '#' < Verzeichnis > [ '#' < Abbruchwarnung > ]
                    '#' < Spaltenkoepfe >
< Daten >          ::= < Datum > ';' < Uhrzeit > ';' < Anzahl_Voegel > ';'
                    < Gesamtzahl_Objekte > ';' < Bildstoerung > ';'
                    < Dateiname >

```

Beispiel Dateiname_Statistics.txt :

```
#Image-Directory : TestImages\
#Warning !! User has aborted computation at 30.04.2007 , 16:21:13
#Date; Time; NumberOfBirdSignals; NumberOfObjectsTotal; Image Dismissed (due to interferences); Filename
30.04.2007 ; 15:31:05 ; 4 ; 13 ; No ; cap_00001581.png
30.04.2007 ; 15:32:05 ; 1 ; 3 ; No ; cap_00003725.png
30.04.2007 ; 16:19:13 ; 0 ; 11 ; No ; cap_00004906.png
30.04.2007 ; 16:20:13 ; 2 ; 5 ; No ; cap_00005028.png
30.04.2007 ; 16:21:13 ; 6 ; 7 ; No ; cap_00005610.png
#Number of Bird-Signals from 30.04.2007 , 16:17:13 to 30.04.2007 , 16:21:13 = 13
```

```
< Datei_SingleObjects > ::= < Dateikopf > { < Daten > } *
< Dateikopf > ::= '# ' < Verzeichnis > '# ' < Spaltenkoepfe >
< Daten > ::= < Objektklasse > '!' < Datum > '!' < Uhrzeit > '!'
           < Flughoehe > '!' < Entfernung > '!' < Helligkeit > '!'
           < Flaeche > '!' < x - Koordinate > '!'
           < y - Koordinate > '!' < Dateiname > '!'
           [ < Zusatzinfo > ]
[ < Zusatzinfo > ] ::= < Tracking - Laenge > '!' < Tracking - Winkel > '!'
                  < Tracking - Geschwindigkeit >
```

Beispiel Dateiname_SingleObjects.txt :

```
#Image-Directory : TestImages2\
#Objectclass ; Date ; Time ; Height ; Distance ; Intensity ; Area ; x-Coordinate ; y-Coordinate ; Filename ;
Track-Length ; Track-Angle (in deg) ; Track-Velocity (in /s)
SomethingElse ; 30.04.2007 ; 15:28:01 ; 266.9217 ; 270.7716 ; 195.0 ; 3.0 ; 543 ; 421 ; img20070408120843.png ;
Bird ; 30.04.2007 ; 15:28:01 ; 245.6893 ; 607.0192 ; 215.6 ; 30.0 ; 711 ; 428 ; img20070408120843.png ;
Bird ; 30.04.2007 ; 15:28:01 ; 245.6893 ; 702.56433 ; 249.96 ; 25.0 ; 745 ; 428 ; img20070408120843.png ; 351.85184
; 90.0 ; 5.8641973
Bird ; 30.04.2007 ; 15:28:01 ; 245.6893 ; 582.1613 ; 199.0 ; 4.0 ; 702 ; 428 ; img20070408120843.png ;
```

4.3 Zusätzliche Programmfunktionalitäten

Als sekundäre Aufgabenstellung galt es, ein älteres Auswerteprogramm für die Daten eines ASR (Air Surveillance Radar), aufgrund der zunehmenden Inkompatibilität mit heutigen Betriebssystemen, neu zu implementieren, bzw. die Funktionalität zu übernehmen und zu erweitern. In Abbildung 17 ist ein solches ASR-Bild zu sehen. Die implementierte Funktionalität umfasst die automatisierte Einschätzung der Zug-Intensität durch Schwarz/Weiß-Zählung der binär vorliegenden Bild-Pixel, sowie die Möglichkeit zur manuellen Kommentierung einzelner Bilder. Als zukünftige Aufgabenstellung könnte eine Automatisierung der Zugrichtungs-Bestimmung gelten, wobei dabei allerdings ernsthaft zu hinterfragen ist, ob die Qualität auf der Hardware-Seite, sprich der zu analysierenden Daten, hierfür überhaupt ausreichend ist.

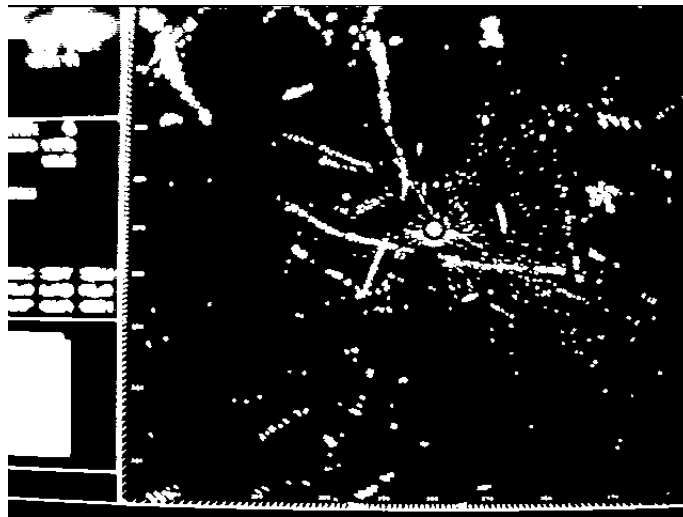


Abb. 17: Beispiel eines ASR-Bildes (640x480 Pixel Binärbild)

5 Implementierung

5.1 Verwendete Software

Die Implementierung erfolgte in Java (Java SE Development Kit 6) mittels der Entwicklungsumgebung Eclipse (SDK 3.2.2). Zur Verwendung kamen folgende zusätzliche Funktions-Bibliotheken :

1. **JAI (Java Advanced Imaging)** Version 1.1.3
[JAI] Ein vielseitiges Java-Paket mit diversen Datenstrukturen und Operationen zur Bildverarbeitung. Da sich diese Routinen im Verlauf der Arbeit jedoch als relativ langsam herausstellten, wurden Sie durch eigene, direkt auf Daten-Arrays arbeitenden Routinen ersetzt. Somit wird die JAI Bibliothek nur noch zum Einlesen der Bilder und zur Darstellung verwendet.
2. **LIBSVM (Library for Support Vector Machines)** Version 2.84
[Chang, Lin 2001] Eine Bibliothek von Java Funktionen um Klassifikationsprobleme mittels Support-Vektor-Maschinen zu lösen (siehe Abschnitt 3.5).

In Abbildung 18 ist die Programmstruktur des implementierten Java-Programms aufgelistet. Viele der aufgelisteten Klassen sind lediglich graphische Oberflächen. Die wichtigsten Klassen sind :

- *ThreadMultiImageAnalysis*
Dieser Thread wird bei der Analyse eines Datei-Verzeichnisses angestoßen und analysiert alle darin befindlichen Bild-Dateien.
- *RegionInfos*
Datenstruktur, welche alle Objekte eines Bildes speichert.
- *RegionFactory*
Beinhaltet die bildverarbeitenden Prozesse und extrahiert die Merkmalsvektoren aus den einzelnen Bildern.
- *ClassifierSVM*
Klasse mit Methoden für z.B. Merkmals-Skalierung, Oversampling, etc. Die vorverarbeiteten Merkmalsvektoren werden dann an die Klasse *svm* im package *libsvm* weitergegeben, welche die eigentliche SVM-Klassifikation übernimmt.
- *ContourTracer*
Beinhaltet den ComponentLabeling-Algorithmus.

- *BirdWatch*

Das Hauptfenster mit Main-Funktion.

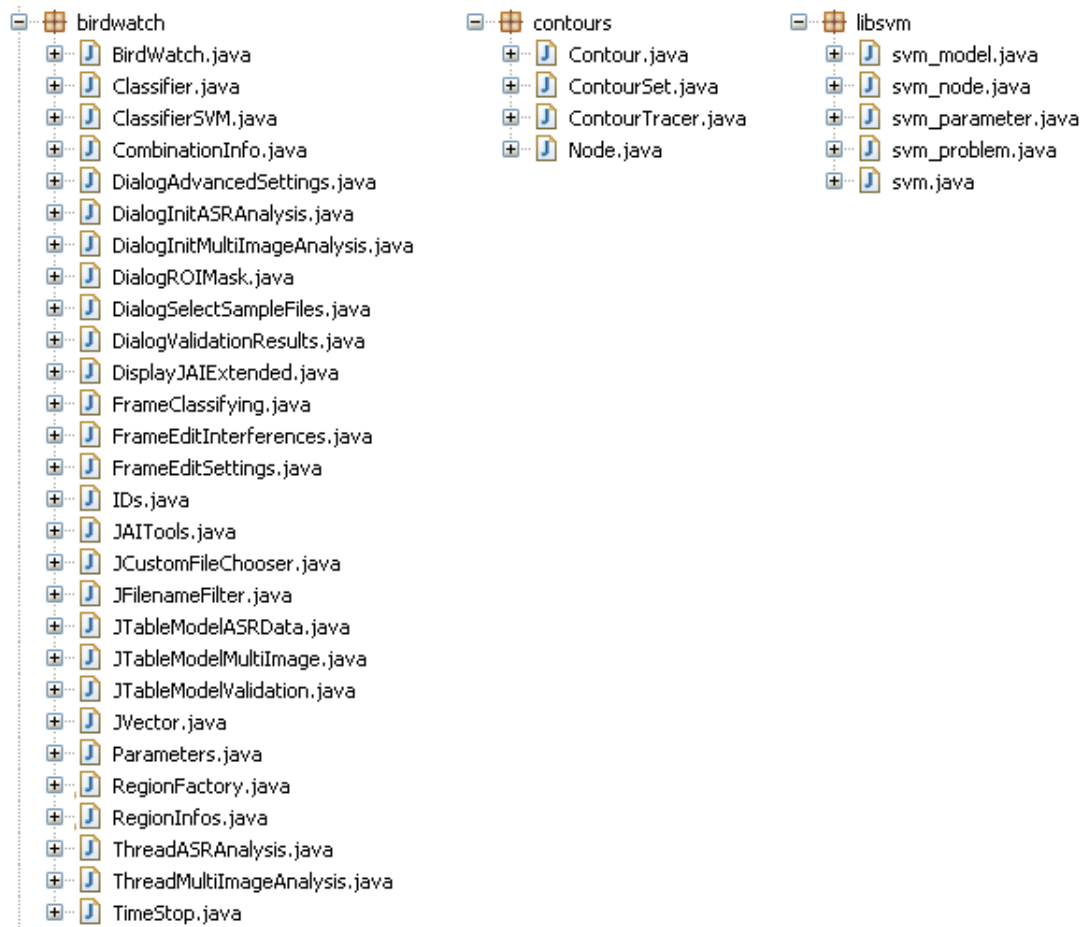


Abb. 18: Auflistung der Klassenstruktur des Programms

6 Ergebnisse

6.1 Gütequalität

Es wurde ein Experiment mit einem Stichprobenumfang von 576 Radarbildern durchgeführt. Diese Radarbilder wurden aus dem Aufzeichnungszeitraum vom 01.03.2007 bis 10.05.2007 zufällig ausgewählt und beinhalten bereits den in Abschnitt 4.1.1 erwähnten „Pflichtanteil“ an Bildern mit Regen und mit Flugzeugsignalen. Die Stichprobe wurde in Test- und Trainingsdaten halbiert und manuell klassifiziert. Anhand der klassifizierten Trainingsdaten wurde ein SVM-Klassifikator trainiert und auf die Testdaten angewandt. Die Ergebnisse wurden im Anschluß mit den manuell klassifizierten Testdaten verglichen und daraus vier Gütemaße berechnet (Definitionen siehe Abschnitt 4.1.1). In Tabelle 4 sind diese dargestellt. In Tabelle 5 ist die zugehörige Verwechslungstabelle angegeben.

	Bird	SomethingElse	Plane	NearPlaneObject
sn	0.825	0.863	0.800	0.575
sp	0.869	0.843	0.990	0.992
gm	0.847	0.853	0.890	0.755
cc	0.658	0.658	0.614	0.623

Tab. 4: Gütemaße des Experiments

Soll \ Ist	Bird	SomethingElse	Plane	NearPlaneObject
Bird	739	146	4	7
SomethingElse	44	302	2	2
Plane	2	0	12	1
NearPlaneObject	7	3	7	23

Tab. 5: Verwechslungstabelle des Experiments

Zu sehen ist, daß die Klassifikationsgüte der Klassen 'Bird', 'Plane' und 'SomethingElse' ungefähr gleich gut ist und die Klasse 'NearPlaneObject' im Vergleich dazu schlechtere Werte liefert. Aus Tabelle 5 ist erkennbar, daß die meisten Fehlklassifikationen zwischen den beiden Klassen 'Bird' und 'SomethingElse' stattfinden.

6.2 Testlauf

Eine Beispiel-Auswertung von 5 Tagen Aufzeichnungsmaterial (1.März 2007 - 5.März 2007, 60 Sekunden Bild-Intervall) ist in den folgenden Tabellen 6, 7, 8, 9, 10 dargestellt :

(Dabei gilt : 'tags' $\hat{=}$ Sonnenaufgang bis Sonnenuntergang , 'nachts' dementsprechend Sonnenuntergang bis Sonnenaufgang.)

Datum	Bird	SomethingElse	Plane	NearPlaneObject	Gesamt
01.03.2007	3038	607	45	36	3726
02.03.2007	2685	779	30	27	3521
03.03.2007	1335	407	11	7	1760
04.03.2007	4615	1214	44	66	5939
05.03.2007	5218	1034	39	62	6353
Summe	16891	4041	169	198	21299

Tab. 6: Beispiel-Auswertung: absolute Häufigkeiten der Objektklassen

Datum	Bird	SomethingElse	Plane	NearPlaneObject	Gesamt
01.03.2007	507	210	27	23	767
02.03.2007	554	258	15	14	841
03.03.2007	346	113	8	6	473
04.03.2007	1343	523	36	65	1967
05.03.2007	1203	391	23	51	1668
Summe	3953	1495	109	159	5716

Tab. 7: Beispiel-Auswertung: absolute Häufigkeiten der Objektklassen tags

Datum	Bird	SomethingElse	Plane	NearPlaneObject	Gesamt
01.03.2007	2531	397	18	13	2959
02.03.2007	2131	521	15	13	2680
03.03.2007	989	294	3	1	1287
04.03.2007	3272	691	8	1	3972
05.03.2007	4015	643	16	11	4685
Summe	12938	2546	60	39	15583

Tab. 8: Beispiel-Auswertung: absolute Häufigkeiten der Objektklassen nachts

	Anteil in % (Gesamt-Tag)	% tags	% nachts
'Bird'	79.30	69.16	83.03
'SomethingElse'	18.97	26.15	16.34
'Plane'	0.79	1.91	0.39
'NearPlaneObject'	0.93	2.78	0.25

Tab. 9: Beispiel-Auswertung: relative Häufigkeiten der Objektklassen

Auffällig an den Werten ist, dass die Anzahl der Objekte am 03.03.2007 sehr niedrig ausfällt, dafür an den folgenden zwei Tage im Vergleich sehr hoch ist. Betrachtet man den zugehörigen Anteil an Störungsbildern (siehe Tabelle 10), so sieht man, daß dieser Störungs-Anteil am 03.03. sehr hoch, und in den folgenden zwei Tagen relativ niedrig ist. Dies lässt vermuten, daß es am 01.03. und 02.03. geregnet hat, am 03.03. dann stark geregnet hat und am 04.03. und 05.03. kaum geregnet hat.

Datum	Anteil Störungsbilder in %
01.03.2007	19.82
02.03.2007	14.05
03.03.2007	58.66
04.03.2007	3.48
05.03.2007	6.75

Tab. 10: Beispiel-Auswertung : Anteil der Störungsbilder (z.B. Regen)

In Abbildung 19 ist die Niederschlagsmenge am Radarstandort in diesen Tagen zu sehen. Deutlich erkennbar ist die Niederschlagsspitze am 03.03.

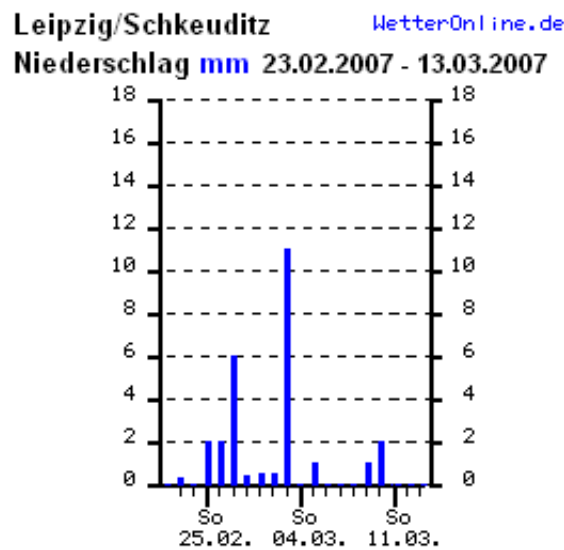


Abb. 19: Niederschlagsmenge am Radarstandort an den Tagen vom 23.02.2007 - 13.03.2007 (Quelle : <http://www.wetteronline.de>)

Laufzeit

Die Laufzeit-Analyse fand auf einem 1.4 GHz AMD Prozessor (mit 256MB RAM und Windows XP) statt. Die Bearbeitung und Auswertung eines Radarbildes benötigte durchschnittlich 1.1s. Grob 80% dieser Zeit entfielen auf das Laden und Dekodieren der auf einer Festplatte vorliegenden Bilder im PNG-Format (1280x1024, 24Bit RGB). Der Rest (ca. 0.2s) entfiel auf die bildverarbeitenden Algorithmen und die Klassifikation.

6.3 Zusammenfassung und Ausblick

Mit Hilfe des Programms soll ermöglicht werden, das Vogelaufkommen über dem Luftraum zu quantifizieren. Dabei spielen auch die Höhenverteilung und das saisonale Auftreten der Vogelsignale eine wichtige Rolle. Bei der Dauer des Projektes und somit der Menge an Daten liegt es auf der Hand, dass eine manuelle und selbst semi-automatische Behandlung der Daten für diese Zielvorgaben nicht erfüllbar ist. Zudem wird angestrebt, reproduzierbare Ergebnisse zu erhalten, die einer wissenschaftlichen Bewertung standhalten. Da diese Untersuchung im möglichen Trend zu strengeren Auflagen bezüglich der Untersuchungen zu Vogelaufkommen an Flughäfen im Rahmen von Planfeststellungsbeschlüssen liegen könnte, erscheint eine weitere Anwendung des Programms möglich. Auch im gutachterlichen Tätigkeitsfeld - insbesondere bei der Beurteilung von potentiellen Standorten von Windkraftanlagen - werden solche Schiffsradargeräte benutzt, so daß hier ebenfalls große Datenmengen vorliegen.

Es hat sich gezeigt, daß die Auswahl der Merkmale einen bedeutenden Beitrag zur Klassifikationsgüte leisten. Von den 127 Kombinationsmöglichkeiten der betrachteten 7 Merkmale liegt z.B. die Möglichkeit, alle Merkmale zur Klassifikation heranzuziehen, bei den betrachteten Gütemaßen $geom_mean(G - Mean)$ und $geom_mean(CC)$ (siehe Abschnitt 3.4) nur auf Platz 33 bzw. 23. Durch eine problemangepasste Merkmalsauswahl kann die Klassifikationsgüte somit signifikant gesteigert werden.

Mit den in Abschnitt 3.4 getroffenen Entscheidungen konnten für das dortige Experiment Sensitivitätswerte von 0.825 für die Klasse 'Bird', 0.863 für die Klasse 'SomethingElse' und 0.8 für die Klasse 'Plane' erzielt werden. Die Spezifitätswerte erreichten 0.869 für die Klasse 'Bird', 0.843 für die Klasse 'SomethingElse' und 0.99 für die Klasse 'Plane'.

Auch das Erkennen und Behandeln von Störungseinflüssen hat Auswirkungen auf die Endresultate, da hierbei unerwünschte Signale herausgefiltert werden müssen. Desweiteren musste aufgrund des Einsatzgebietes an einem Flughafen die Behandlung der Flugzeugsignale implementiert werden, welche in flughafenfernen Gebiete

vernachlässigbar wären. Zur Klassifikationshilfe mag die Idee einer Objektverfolgung auf den ersten Blick naheliegend erscheinen, da die Bilder jedoch in einem Intervall von 60 Sekunden aufgenommen wurden, ist der Versuch einer Objektverfolgung relativ zwecklos, da sich die Bilder in diesen 60 Sekunden stark verändern. Bei einzelnen seitlich fliegenden Objekte mag dies zwar möglich sein, da der Hauptteil der Vögel den Radar-Bereich jedoch senkrecht durch die Betrachtungsebene durchquert, ist deren Verweildauer im Radar-Bereich sehr gering. Da die Objektverfolgung anhand dieser Bilder somit nicht möglich ist, entfallen sämtliche auf Geschwindigkeit und Bewegungsmuster/-richtung der Objekte bezogene Merkmale. Und leider sind gerade dies (wahrscheinlich) sehr gute Merkmale, um Vögel von z.B. Flugzeugen, Insekten oder Regenwolken zu unterscheiden. Eine zukünftige Entwicklungsrichtung wäre eventuell eine 'online'-Analyse eines Live-Video-Streams, da hierbei oben genannte Merkmale aus den Daten gewonnen werden können. Allerdings besteht dann die Frage, ob die Rohdaten (das Video-Material) überhaupt noch für eventuelle andere Analysen gespeichert werden sollen. Mit heutiger (2007) Speichertechnologie wäre die Speicherung des Video-Materials (bei einer Aufzeichnungszeit von 2-3 Jahren) in akzeptabler Qualität jedenfalls äußerst aufwendig und teuer.

Literatur

- [Abe 2005] Shigeo Abe : Support Vector Machines for Pattern Recognition, Springer, 2005
- [Burger,Burge 2005] Wilhelm Burger, Mark James Burge : Digitale Bildverarbeitung, Springer, 2005
- [Chang, Lin 2001] Chih-Chung Chang, Chih-Jen Lin : LIBSVM - A Library for Support Vector Machines, National Taiwan University, 2001, <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>
- [Haufe 2005] Stefan Haufe : Detektion von Transkriptionsfaktorbindestellen mit Support-Vektor-Maschinen, Diplomarbeit Universität Halle, 2005
- [JAI] „jai-core Project Home Page“, <https://jai-core.dev.java.net/>
- [Schölkopf, Smola 2001] Schölkopf, Smola : Learning with Kernels, MIT Press, 2001
- [Theodoridis 1999] S. Theodoridis, K. Koutroumbas : Pattern Recognition, Academic Press, 1999
- [Vapnik 1998] Vladimir N. Vapnik : Statistical Learning Theory, Wiley, 1998