Analyse der Klassifizierbarkeit von Vögeln anhand ihrer projizierten Flugbahn

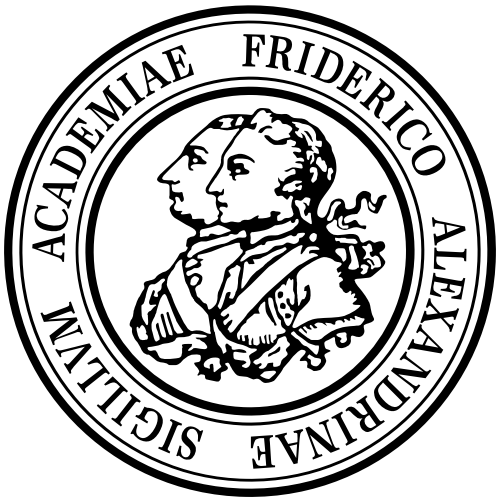
Bachelorarbeit aus der Physik

Vorgelegt von

Matthias Späth

14.03.2014

Lehrstuhl für Physikalisch-Medizinische Technik
Friedrich-Alexander-Universität Erlangen-Nürnberg



Betreuung: Dr. Daniel Zitterbart
und Prof. Dr. Ben Fabry

Inhaltsverzeichnis

Zusammenfassung

Im Rahmen der Arbeit werden projizierte Flugbahnen von Vögeln analysiert und dabei untersucht, inwieweit die verschiedenen Eigenschaften der Flugbahnen eine Trennung in Klassen - und damit vielleicht in Arten - ermöglichen. Die Daten dafür stammen von einer Kamera auf dem Forschungsschiff Polarstern, aus deren Infrarotbildern die Flugbahnen zunächst durch ein Tracking mittels Bildanalyse und Bewegungsmodell gewonnen werden. Dabei werden unvermeidlich neben den Vogelflugbahnen auch weitere Tracks auf beispielsweise Reflexionen des Sonnenlichts gebildet. Die weitergehende Analyse zieht Parallelen zu den Eigenschaften von Correlated Random Walks und zeigt auf, dass die Vogeldaten mittels Support Vector Machine aus den verschiedenen Tracks in den Datensätzen extrahiert werden können. Untersuchungen am Modell des Correlated Random Walks deuten an, dass eine Unterteilung der Vögel in verschiedene Klassen ebenfalls möglich ist.

1 Einleitung

> ≫Seabirds are considered excellent indicators of the state of the marine environment and are used throughout the world to provide signals from the marine environment.≪[1]

So lautet es auf einer Homepage des *Department of the Environment* der Australischen Regierung [1]. Und dieser Eindruck wird noch weiter verstärkt, wenn man sich mit dem Einfluss des Klimawandels auf die Meere befasst.
Mit einer Erwärmung von 6 °C in den letzten 50 Jahren ist die westliche Antarktische Halbinsel die am stärksten vom Klimawandel betroffene Region der Welt [2]. Dies führt zu einem Rückgang des Meereises und damit der Seealgen, die unter dem Eis wachsen. Der Einfluss setzt sich fort beim Krill, Schrimps ähnlichen Krustentieren die Algen fressen, und führt über Fische, die sich wiederum vom Krill ernähren, zu den Seevögeln. Beginnend mit dem Schmelzen des Eises, kommt es durch diese lange Nahrungskette zu einem Rückgang der Beute für die Vögel [3]. Durch unterschiedlich stark ausgeprägte Anpassungsfähigkeit können manche Vogelarten besser oder schlechter mit dieser Entwicklung umgehen. Sturmvögel sind aufgrund ihrer Langlebigkeit in der Lage ein zu warmes Jahr, und damit ein Jahr mit weniger Beute, beim Brüten auszulassen. Dies beeinflusst den Fortbestand der Art nur geringfügig [3]. Dadurch ergibt sich aus dem Klimawandel ein Vorteil für die Sturmvögel gegenüber anderen, was das Verhältnis der

[1] Übersetzung: Meeresvögel werden für herausragende Indikatoren für den Zustand der marinen Umwelt gehalten und werden über die ganze Welt verwendet, um Signale der marinen Umwelt darzustellen.

Populationen beeinflusst. Ein weiteres offensichtliches Beispiel für den Effekt des Klimawandels auf das Leben der Seevögel findet sich am anderen Ende der Welt: Die Elfenbeinmöwe. In der Arktis und in Kanada verbringt sie den größten Teil ihres Lebens auf dem Eis. Die Möwe brütet auf dem felsigen Kliff und jagt die meiste Zeit durch einzelne Löcher im naheliegenden Eis nach Fischen. Es wird geschätzt, dass sich die Population der Elfenbeinmöwe in den letzten 20 Jahren bedingt durch den Rückgang des Eises um 90% verringert hat [4]. Beide Beispiele zeigen deutlich, dass das Leben der Seevögel eng mit den Gegebenheiten im Meer und dem Klimawandel verbunden ist. Die Populationen der Seevögel stellen demnach direkte Indikatoren für ihre Umwelt und den Einfluss des Klimawandels dar. Deshalb bietet es sich für Rückschlüsse auf die Effekte des Klimawandels an, Änderungen der Populationsgrößen und deren Lokalisierung als Indikatoren zu betrachten.

Für die Erfassung dieser Daten sind zwei Möglichkeiten verbreitet: Die Beobachtung von Tieren mittels GPS Sendern, welche vorher direkt an die Tiere angebracht werden müssen, und das Beobachten durch den Mensch bzw. durch Sichtung von Videomaterial. Bei der Verwendung einer Kamera gibt es die Möglichkeit eine Infrarotkamera zu verwenden, um die Tiere durch ihre Körperwärme in den Aufnahmen erkennen zu können. Zählungen der Populationen von Mexikanischen Bulldoggfledermäusen sind ein Beispiel für diese Anwendung [5]. Eine Infrarotkamera hat dabei den Vorteil, dass in der Nacht, wenn andere Kamerasysteme mit dem geringen Licht zu kämpfen haben, trotzdem Aufnahmen möglich sind [6]. Die steigende Anzahl an Offshore Windparks im Rahmen der Energiewende gibt einen aktuellen Anlass für ein weiteres Beispiel: Untersuchungen der Häufigkeit von, meist tödlichen, Zusammenstößen zwischen Meeresvögeln und Windturbinen, wobei auch ein Zusammenhang zwischen Vogelgröße und Entfernung für einige Vogelarten analysiert wurde [7]. Durch die Verwendung von zwei Kameras kann eine extrem genaue Lokalisierung (Auflösungen von mehr als 3 cm bei einer Entfernung von 50 m) erfolgen. Das bietet Möglichkeiten für die Analyse des genauen Verhaltens der Vögel zueinander und für eine präzise Größenbestimmung [8].

Alle genannten Methoden mittels Videoaufnahme sind dadurch eingeschränkt, dass die Flugdauer der Vögel im Bild der Kamera maximal einige Sekunden beträgt. Außerdem ist die Kamera dabei an einer festen Position, wodurch das Aufnehmen an mehreren Orten einen erheblichen Mehraufwand bedeutet. Dies ließe sich durch eine Kamera lösen, die selbst während der Aufnahme mobil ist: zum Beispiel die Infrarotkamera auf dem Forschungsschiff Polarstern des Alfred-Wegener-Instituts. Die Kamera ist bereits seit 2009 auf dem Schiff montiert und wird dort beispielsweise auf Fahrten von Bremerhaven

nach Longyearbyen in der Grönlandsee oder zur Neumayer Station in der Antarktis zur Detektion von Meeressäugern verwendet [9]. Die Kamera nimmt dabei bis zum Horizont auf, wobei auch ein Großteil des Flugraumes für Vögel erfasst wird, und somit finden sich in den umfassenden Datensätzen von vielen Monaten eine große Anzahl an Vögeln. Dies stellt eine außergewöhnliche Möglichkeit dar, Daten aus abgelegenen Gebieten zu erhalten, wenn die Infrarotaufnahmen ausreichend Informationen für Rückschlüsse auf die Vögel enthalten. Diese Möglichkeit zu untersuchen ist Aufgabe dieser Arbeit. Da die Vögel in den Aufnahmen sehr klein sind und jegliche Farbinformation fehlt muss die Information in der Flugbahn der Vögel gefunden werden. Um diese aus den Rohdaten zu extrahieren ist ein Tracking der Vögel in den Bildern notwendig, wobei auf Basis von einzelnen Detektionen plausible Tracks generiert werden. Jedoch stellt sich das Erkennen der Vögel in den Aufnahmen als recht anspruchsvoll heraus: Neben den Vögeln enthalten die Bilder auch eine Vielzahl von Reflexionen des Infrarot Spektrums der Sonne auf der Wasseroberfläche. Aus diesen Grund kommen für die Detektion verschiedene Filter und für das Tracking ein Hilfsmittel für Interpolation und Vorhersage zum Einsatz. Um eine anschließende Klassifizierung der Flugpfade - und damit Rückschlüsse auf die Vögel - zu ermöglichen, müssen die gesammelten Tracks analysiert werden. Eine Herausforderung dabei ist, dass in den Daten keine 3D-Information enthalten ist, sondern lediglich die projizierten Informationen der Flugbahn ermittelt werden können.

2 Theoretischer Hintergrund

Im Folgenden werden zunächst die mathematischen Grundlagen des Kalman Filter, der zur Messung der Trajektorien dient (Vorhersage der folgenden Zeitschritte und Interpolation), betrachtet. Anschließend werden Correlated Random Walks vorgestellt, welche als Modell für die Flugbahnen und deren Analyse geeignet sind.

2.1 Kalman Filter

Für das Tracking ist die Verwendung eines Algorithmus zur Vorhersage für die nächste Position grundlegend, um eine korrekte Zuordnung von Track und Detektion zu ermöglichen. Dafür bietet sich der Kalman Filter an, der in der Literatur häufig verwendet wird und für persistente Bewegungen eine gängiges Tool für das Tracking ist.

2.1.1 Grundlagen

Der Kalman Filter ist ein bayesscher Filter zur Vorhersage von Punkten eines Tracks und hat seinen Ursprung bei der Interpolation und Mittelung von verrauschten Messwerten, beispielsweise zur Verbesserung bei GPS Positionen. Die hier vorgestellte theoretische Beschreibung des Kalman Filters basiert größtenteils auf dem Buch *Propabilistic Robotics* von Sebastian Thrun, Wolfram Burgard und Dieter Fox [10]. Für den Kalman Filter wird eine lineare Dynamik für den Zustandsvektor x_t angenommen, das heißt, dass

$$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t \tag{1}$$

den Übergang von x_{t-1} zu x_t beschreibt. Hierbei ist A_t eine $n \times n$ Matrix, wobei n die Dimension des Vektors x ist. Im Fall des Tracking eines Vogels im zweidimensionalen Bild ist die Dimension folglich zwei. u_t ist ein Kontrollvektor mit Dimension m, somit ist B_t dementsprechend eine $n \times m$ Matrix. Der Kontrollvektor wird dabei beispielsweise beim Tracking von Autos verwendet, bei denen der Beobachter aktiv Einfluss auf die Bewegung nehmen kann (Beschleunigung des Autos) und ist deshalb in der Anwendung hier gleich $u_t = 0$ zu setzen. Beschleunigung oder ein Drehen des Schiffes ist dabei so langsam, dass dies vernachlässigt werden kann. Mit einer mehrdimensionalen Gauß Verteilung kann dann die Wahrscheinlichkeit mittels $p(x_t \mid u_t, x_{t-1})$ für den Schritt von x_{t-1} zu x_t mit der Kovarianz R_t angegeben werden:

$$p(x_t \mid u_t, x_{t-1}) = det(2\pi R_t)^{\frac{1}{2}} exp[-\frac{1}{2}(x_t - A_t x_{t-1} + B_t u_t)^T R_t^{-1}(x_t - A_t x_{t-1} + B_t u_t)] \tag{2}$$

Weitere Voraussetzung ist, dass die Messwahrscheinlichkeit $p(z_t \mid x_t)$, also dass z_t gemessen wird wenn x_t vorliegt, linear $z_t = C_t x_t + \delta_t$ ist und damit ebenfalls als Gaußkurve mit Kovarianz Q_t dargestellt werden kann.

$$p(z_t \mid x_t) = det(2\pi Q_t)^{-\frac{1}{2}} exp[-\frac{1}{2}(z_t - C_t x_t)^T Q_t^{-1}(z_t - C_t x_t)] \tag{3}$$

Das gleiche gilt für die Annahme (*believe*) zu den Startbedingungen des Tracks $bel(x_0) = p(x_0)$ mit Mittelwert μ_0 und Kovarianz Σ_0. Die Bewegung des Vogels, der mit dem Kalman Filter verfolgt werden soll, kann als *Hidden Markov Model* angesehen werden. Der nächste Zeitschritt hängt immer nur direkt mit dem Vorherigen mittels einer Übergangswahrscheinlichkeit zusammen. Deshalb muss jeder Zustand vollständig sein. Daraus folgt, dass die Zukunft auch mit zusätzlicher Information nicht genauer vorhergesagt werden kann als nur mit diesem einen Zustand. Das *Hidden* im Namen bedeutet, dass nicht der Zustand direkt gemessen wird, sondern die zugehörigen Ausgabewerte. Dies entspricht dem Fall, dass der Vogel im teilweise schlecht erkennbaren Zustand im Bild gesucht wird oder manchmal nicht erkennbar ist.

2.1.2 Mathematische Begründung

Nach oben gegebenen Grundlagen ist die Wahrscheinlichkeit für einen neuen Zustand x_t nur abhängig vom vorherigen Zustand x_{t-1} und der Kontrollvariable u_t: $p(x_t \mid x_{t-1}, u_t)$. Diese Wahrscheinlichkeit wird *State Transition Probability* genannt. Die Messung z_t für den neuen Zustand ist wiederum nur abhängig vom Zustand x_t: $p(z_t \mid x_t)$ (*Measurement Probability*). Für den nächsten Zustand kann dann ein vermuteter Wert (*State Of Knowledge*) angesetzt werden, welcher abhängig von den vorherigen Messungen und Kontrollvariablen ist: $bel(x_t) = p(x_t \mid z_{1:t}, u_{1:t})$. Der $bel(x)$ ergibt sich aus dem $\overline{bel}(x_t) = p(x_t \mid z_{1:t-1}, u_{1:t})$ durch die Messung. Um diese Werte zu berechnen wird der Bayes Filter verwendet:

$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1} \tag{4}$$

$$bel(x_t) = \eta p(z_t \mid x_t) \overline{bel}(x_t) \tag{5}$$

η ist hierbei nur ein Faktor für die Normalisierung. Für die Herleitung wird die Formel von Bayes und die Bedingung, dass der jeweilige Zustand vollständig ist verwendet. Berechnet man $\overline{bel}$ nach Gleichung (4) für den Kalman Filter erhält man wiederum eine Gaußkurve. Die Berechnung ist möglich, indem man den Exponenten im Integral geschickt aufteilt, in Teile abhängig von x_{t-1}, x_t und Teile abhängig nur von x_t, um

dx_{t-1} ausführen zu können. Dadurch ergibt sich für x_t die erstaunlich einfache Formel

$$\bar{\mu}_t = \overline{bel}(x_t) = B_t u_t + A_t \mu_{t-1} \tag{6}$$

und mittels Kehrwert der zweiten Ableitung an der Stelle x_t die Kovarianz

$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t. \tag{7}$$

Der nächste Schritt ist nun die Integration der Messung um von $\overline{bel}(x_t)$ zu $bel(x_t)$ zu kommen. Da $p(z_t \mid x_t)$ nach den Voraussetzungen ebenso wie $\overline{bel}(x_t)$ eine Gaußkurve ist, ergibt sich für $bel(x_t)$ ebenfalls eine Gaußkurve. Auch hier wird durch erste und zweite Ableitung Mittelwert und Kovarianz bestimmt, wobei der sogenannte Kalman Gain $K_t = \Sigma_t C_t^T Q_t^{-1}$ eingeführt wird:

$$\mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t) \tag{8}$$

Der Kalman Gain wird durch geschickte Erweiterungen mit Einheitsmatrizen auf

$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1} \tag{9}$$

umgeformt. Dadurch lässt sich die Kovarianz für den neuen Punkt durch den Kalman Gain ausdrücken:

$$\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \tag{10}$$

Damit stehen alle Formeln für die Berechnung von μ_t und Σ_t aus μ_{t-1}, Σ_{t-1}, u_t und z_t:

$$\bar{\mu}_t = B_t u_t + A_t \mu_{t-1} \tag{11}$$

$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \tag{12}$$

$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1} \tag{13}$$

$$\mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t) \tag{14}$$

$$\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \tag{15}$$

2.2 Random Walks

Für die Analyse der Flugbahnen der Vögel kann eine Analogie zu Correlated Random Walks herangezogen werden. Deshalb ist ein Verständnis für die Grundlagen von Random Walks und deren Eigenschaften hilfreich. Ein Random Walk ist eine Bewegung, die sich aus einer zufälligen Aufeinanderfolge von Schritten zusammensetzt. Ein Spezialfall mit breiterem Anwendungsspektrum ist der Correlated Random Walk.

2.2.1 Beschreibung eines Correlated Random Walk

Korreliert (*correlated*) heißt bei einem Random Walk, dass nicht jeder neue Schritt eine völlig zufällige Richtung hat, sondern dass dieser mindestens vom vorherigen Schritt statistisch abhängig ist. Dies kann bedeuten, dass die Richtung sich nur in einem gewissen Winkelbereich ändern kann, in diesem Bereich am wahrscheinlichsten ist oder auch die Geschwindigkeit, also die Länge des Schrittes, sich nur begrenzt ändern kann. Eine häufige Verwendung von korrelierten Random Walks ist die Darstellung von Zellbewegungen [11], jedoch ist auch die Anwendung auf Bewegungen von Insekten (beispielsweise des Borkenkäfer [12]) oder andere Tiere möglich. Da Vögel eine gezielte aktive Bewegung ausführen ist für die Beschreibung eines Vogelflug ein stark korrelierter Random Walk notwendig. Im weiteren Verlauf bezieht sich der Ausdruck Random Walk immer auf einen Correlated Random Walk, wenn nicht anders angegeben. Die Korrelation erhält man beispielsweise, wenn die neue Schrittlänge immer wieder direkt aus der vorherigen berechnet wird und dabei eine zufällige Änderung eingefügt wird. Das hier verwendete Modell beruht dabei auf dem AR(1)-Prozess (Autoregressiver Prozess 1. Ordnung [13]) und damit auf folgendem Zusammenhang für die Schrittlängen:

$$\Delta x_k = q\Delta x_{k-1} + \sigma\eta \tag{16}$$

Hierbei ist Δx_k die neue, Δx_{k-1} die vorherige Schrittlänge, q ein Parameter zwischen -1 und 1, der ein Maß für den Grad der Korrelation angibt, σ ein Maß für die Geschwindigkeit und η eine Normalverteilung mit Mittelwert 0 und Varianz 1. Für Vögel, die durchaus einen hohen Grad an Korrelation aufweisen, liegen geeignete Parameter für q im Bereich von 0.8 bis 1. Bei diesem Modell geht diese direkt einher mit Persistenz, das heißt eine hohe Korrelation bedingt auch eine hohe Persistenz, wie sie bei den Flugpfaden gegeben ist. Im Rahmen der Arbeit wurden auch andere Methoden des Random Walk betrachtet, beispielsweise das Zusammensetzen von einzelnen Zeitschritten, die jeweils aus dem vorherigen Zeitschritt mittels einer Winkeländerung in sphärischen Koordinaten

mit verschiedensten Verteilungen errechnet werden. Keine dieser teilweise sehr aufwendigen Konstruktionen kommt dabei allerdings optisch näher an einen Vogelflug als das recht einfache Modell des AR(1)-Prozess.

2.2.2 Analyse eines Random Walks

Hat man durch die Anwendung des Kalman Filters und verschiedener Filter zur Bildbearbeitung die Tracks aus der Videosequenz extrahiert, ist es entscheidend, die relevanten Vogeltracks von denen von Reflexionen auf Wellen oder Ähnlichem zu trennen. Verdeutlicht wird diese Notwendigkeit an der Darstellung in Abb. 1. Hier sind alle Tracks aus einem Datensatz von 300 Bildern abgebildet, wobei nur ein wirklicher Vogel bei insgesamt 236 Tracks dabei ist. Für diese Analyse kann man den Pfad wieder vereinfachend als Correlated Random Walk annehmen. Die Schrittlänge des Random Walk ist hierbei dann stark variabel, da der Vogel je nach Entfernung zur Kamera, Flugrichtung und Fluggeschwindigkeit unterschiedlichste Distanzen im Zeitraum *dt* überwindet. Der Winkel kann ebenfalls stark variieren, wobei man davon ausgehen kann, dass ein Vogel meist relativ gerade und nur selten sehr enge Kurven fliegt. Es folgt eine kurze Vorstellung der dabei verwendeten Kriterien.

Mean Squared Displacement

Die Mean Squared Displacement (mittlere quadratische Verschiebung, kurz MSD) ist eine Methode zur Analyse der räumlichen Ausbreitung einer Zufallsbewegung und wird häufig in der Zellbiologie zur Beschreibung von Zellbewegungen verwendet (beispielsweise [11], [14]). Obwohl sich Vögel sicherlich gezielter fortbewegen als Zellen, lässt sich die MSD auch zur Analyse der Flugbahnen von Vögeln verwenden. Die MSD ist dabei der Mittelwert der quadratischen Verschiebung des Vogels in der Zeit τ:

$$MSD(\tau) =< \Delta\mathbf{r}(\tau)^2 >=< [\mathbf{r}(t+\tau) - \mathbf{r}(t)]^2 > \tag{17}$$

Die MSD kann, logarithmisch aufgetragen, durch eine Gerade genähert werden, deren Steigung die Klassifizierung der Bewegung zulässt. Dies steht nach [11] in Zusammenhang mit der Diffusionskonstante D über die Relation

$$< \Delta\mathbf{r}(\tau)^2 > \propto D \left(\frac{\tau}{t_0}\right)^{\beta}. \tag{18}$$

Dabei ist die Klassifizierung der Bewegung für $\beta = 1$ diffusiv, $\beta < 1$ subdiffusiv, $\beta > 1$ superdiffusiv und für $\beta = 2$ ballistisch.

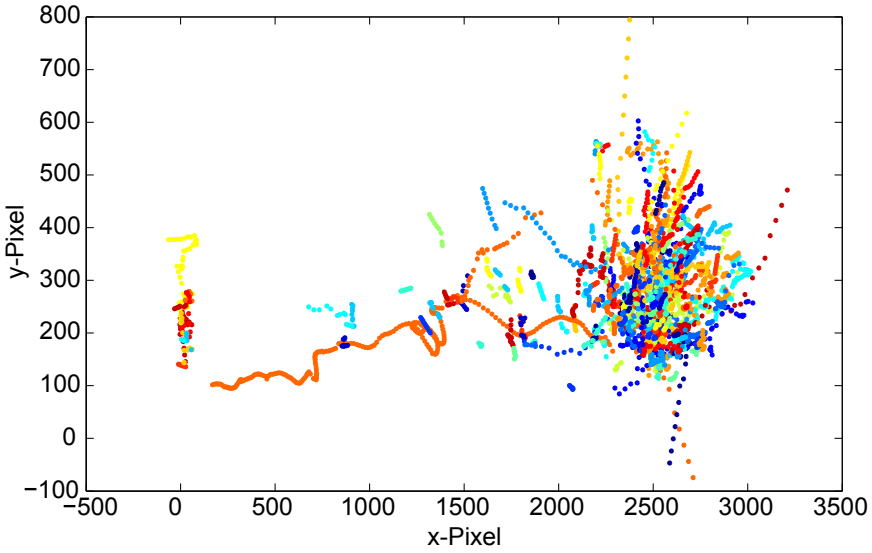


Abbildung 1: Alle Tracks in einem Datensatz von einer Minute. Der einzige darin vorhandene Vogel findet sich im Bereich von y = ca. 200 bis ca. 2200 Pixel und x = ca. 100 bis ca. 300 Pixel in oranger Farbe. Gesamtzahl der Tracks: 236 (Negative Positionen für Pixel in der Darstellung können durch Interpolationen bis außerhalb des Bildes auftreten.)

Straightness Index

Eine schnelle und einfache Möglichkeit einen Random Walk zu analysieren ist der Straightness Index [14]. Dieser gibt ein Verhältnis von absolut zurückgelegter Strecke zur dafür benötigten Pfadlänge an.

$$I_{\text{Straightness}} = \frac{|(x_{\text{end}} - x_{\text{start}}, y_{\text{end}} - y_{\text{start}})|}{\sum_{t=0}^{n} l_j} \tag{19}$$

Hier ist x_{end} der Endpunkt und x_{start} der Anfangspunkt des Pfades in x und y analog. l_t ist die Länge des Schrittes im jeweiligen Zeitschritt t, wobei n die Anzahl der Zeitschritte des Pfades ist. Somit ist also $I_{\text{Straightness}} = 1$ eine vollkommen gerade Flugbahn und $I_{\text{Straightness}} = 0$ eine Flugbahn die wieder zum Ausgangspunkt zurückkehrt. Durch die einfache Berechnung eignet sich der Straightness Index hervorragend, um bereits einen Teil der verfolgten Wellen auszusortieren, da deren Straightness Index meist deutlich unter dem einer Vogelflugbahn liegt. Die Grenzen des Straightness Index liegen beim Vergleich von ähnlichen Flugbahnen von Vögeln: Der Straightness Index ist abhängig von der Anzahl an Zeitschritten. Dadurch können Pfade unterschiedlicher zeitlicher Länge

eigentlich nicht miteinander verglichen werden. Jedoch reicht für die Unterscheidung Welle-Vogel der Unterschied trotz unterschiedlicher Pfadlängen aus.

Turning Angle Distribution

Die für die Arbeit wichtigste Möglichkeit der Analyse von Random Walks ist die Turning Angle Distribution (Drehwinkel-Verteilung) [11]. Bei der Turning Angle Distribution wird jeder Punkt des Pfades durch einen Vektor mit einem Punkt n Schritte in der Zeit voraus $\Delta\vec{r_+} = \vec{r}(t + \Delta t) - \vec{r}(t)$ und mit dem Punkt n Schritte in der Zeit zurück $\Delta\vec{r_-} = \vec{r}(t) - \vec{r}(t - \Delta t)$ verbunden. Der Winkel zwischen den beiden Vektoren wird gemessen und zu diesem Punkt gespeichert:

$$\Delta\Phi(t, \Delta t) = \angle(\Delta\vec{r_+}, \Delta\vec{r_-}) \tag{20}$$

Der gemessene Winkel kann dann mittels Histogramm dargestellt und beispielsweise anhand von Mittelwert, Varianz oder Kurtosis analysiert werden. Darin sind Informationen über die Persistenz der Flugbahn und damit der Häufigkeit von Kurven, die Steilheit der Kurven und der bevorzugten Richtung der Kurven enthalten. Eine große Schwierigkeit bei der Analyse ergibt sich dadurch, dass die Informationen im Bild nur die projizierte Flugbahn der Vögel ist. Die Turning Angle Distribution ist skaleninvariant und umgeht damit den einen Nachteil der fehlenden Information zur Entfernung. Allerdings entsteht mangels 3D-Information auch das Problem, dass bei einer Bewegung auf die Kamera zu oder von der Kamera weg im Bild keine Bewegung vorhanden ist. Dadurch entsteht die Möglichkeit, dass ein Vogel, der eigentlich eine weite Kurve fliegt, dabei allerdings teilweise auf die Kamera zufliegt, in der Projektion einen direkten Knick macht. Dieses Beispiel wird folglich auch in der Turning Angle Distribution falsch aufgefasst und erschwert die Interpretation der Daten.

3 Methode

3.1 Kamera

Die für die Arbeit verwendeten Aufnahmen kommen von einer FirstNavy IR Kamera (hergestellt von der Rheinmetall Defence Eletronics GmbH), die ursprünglich für militärische Zwecke, zur Abwehr von Granaten, gedacht war. Die Kamera ist bereits seit 2009 auf dem Forschungsschiff Polarstern angebracht, um im Umkreis befindliche Wale zu detektieren. Durch den großen Blickwinkel und die Ausrichtung der Kamera filmt diese bis zum Horizont, wodurch von der Höhe im Krähennest (28.5 m) auch ein großer Teil des Luftraums gefilmt wird. In diesem befinden sich eine Vielzahl von Vögeln, die durch ihre Körperwärme im Infrarotbereich aufgelöst werden können. Die Kamera erfasst dabei in 360° 7200 auf 576 Pixel bei einer Wiederholrate von 5 Hz. Zur Stabilisierung gegenüber dem schwankenden Schiff ist die Kamera auf einer gyrostabilisierten Plattform angebracht. Gemessen werden keine absoluten Temperaturen sondern Temperaturdifferenzen, was jedoch durch den großen Kontrast der Körpertemperatur zum verhältnismäßig kaltem Meerwasser (in polaren Regionen) kein Problem darstellt. Der große Vorteil einer Kamera auf dem bewegten Schiff ist, dass Vögel teilweise auch über längere Zeit dem Schiff folgen und somit länger beobachtet werden können. Außerdem werden so Daten von verschiedenen Orten gesammelt, was bei der Verwendung von stationären Kameras wesentlich aufwendiger wäre.

3.2 Bereinigen der Rohdaten

Die Rohdaten, welche durch die spezielle Form der Aufnahme um 360° nicht direkt verwendet werden können, müssen zunächst mit der Software Tashtego [15] vorbereitet werden. Die Aufnahmen stehen nach dieser Vorbereitung wie beispielsweise weiter unten in Abb. 2 (a) zur Verfügung.

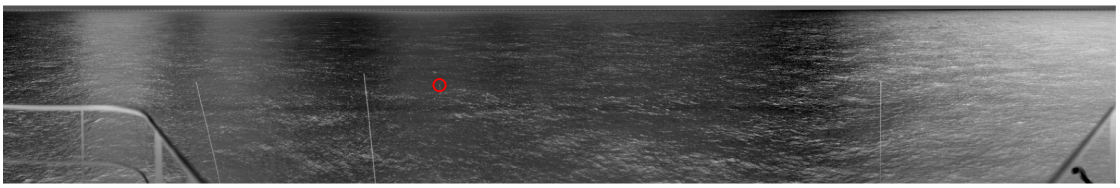
3.3 Detektion

Um mit dem Tracking der Vögel beginnen zu können, müssen diese zunächst im Bild detektiert werden. Erste Gedanken dazu sind, dass Vögel in der Regel zu den hellsten Punkten in der jeweiligen Aufnahme gehören und zudem auch nicht zu groß (wie beispielsweise Eisschollen) sein können.
Für das Detektieren und Verfolgen der Vögel wird im Weiteren immer mit einem Binärbild gearbeitet. Dies wird mit einem 99% Perzentil bestimmt. Die Angabe 99% bedeutet da-

bei, dass nur das hellste Prozent des Bildes als 1 (in dieser Darstellung weiß) im Bild bestehen bleibt und alle anderen Punkte 0 (schwarz) gesetzt werden.

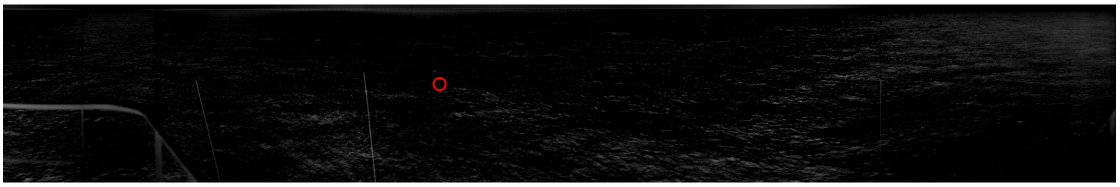
3.3.1 Vorbereiten des Bildes mittels Filtern

Ohne vorherige Anwendung von Filtern ist es selbst mit dem menschlichen Auge in den meisten Bildern äußerst schwierig bis unmöglich, einen Vogel als solchen zu erkennen. Aus diesem Grund muss das jeweilige Bild zuerst bearbeitet werden. In Bild (a) der Abbildung 2 ist deutlich zu erkennen, dass das reflektierte Sonnenlicht auf dem Wasser im Bild grundsätzlich in vertikal verlaufenden Flächen stärker oder schwächer auftritt. Da im Weiteren nur das hellste Perzentil verwendet wird und das Perzentil über das gesamte Bild berechnet wird, bleiben fast nur noch Punkte im Bereich der hellen Reflexionen übrig. Dies ist in (b) deutlich zu erkennen. Aus diesem Grund muss das Bild zunächst gleichmäßig hell werden. Dies läuft so ab, dass mit einem Filter (in meinem Code `colThreshold.m`) je in vertikalen Spalten der Mittelwert der Pixelwerte berechnet wird. Anschließend wird von der jeweiligen Spalte der Mittelwert abgezogen. Durch diesen Prozess wird die Intensität spaltenweise normiert und die Reflexionen der Sonne werden herausgerechnet. Ein Beispiel für die berechneten Mittelwerte der Helligkeit in den Spalten findet sich in Abb. 3. Vergleicht man dies mit Bild (a) in Abb. 2 sind deutlich die Bereiche starker und schwacher Reflexion erkennbar. Wird nun das Binärbild (ebenfalls das obere 1% Perzentil) erstellt, bleiben deutlich mehr Punkte über das Bild verteilt erhalten und die hell beleuchteten Flächen verlieren ihre Sonderstellung als hellster Bereich im Bild. Die Vögel heben sich in den dunkleren Bereichen des Bildes durch den dynamischen Grenzwert deutlicher vom Hintergrund ab und sind in diesem Bereichen leichter zu detektieren und zu tracken. Der Filter ermöglicht es dann, dass diese Vögel auch wirklich Teil der gewählten Perzentile sind und nicht hinter dem Sonnenlicht in einem anderen Bereich des Bildes verschwinden. Ein Vogel, der sich im hell beleuchteten Teil des Bildes befindet, bleibt jedoch, aufgrund der geringen Intensitätsdifferenz, eine Herausforderung. Im gezeigten Beispiel ist der Effekt erkennbar, dass im ungefilterten Bild der Vogel im Binärbild verschwindet, während er im Binärbild nach dem Filter noch zu erkennen und somit zu detektieren und tracken ist. Abbildung 4 zeigt diesen Effekt in Vergrößerung. Um das Tracking mit den gegebenen Bildern weiter zu vereinfachen ist es notwendig, große weiße Bereiche und sehr kleine weiße Bereiche aus dem Binärbild zu entfernen. Große Bereiche können beispielsweise die Reling, eine der Antennen des Schiffs oder Eisschollen sein, während sehr kleine Punkte größtenteils helle Reflexionspunkte auf den Wellen sind. Diesem Verfahren liegt der Gedankengang zu-



(a) Ausgangsbild

(b) Ausgangsbild Binär



(c) Gefiltertes Bild

(d) Gefiltertes Bild Binär

Abbildung 2: Beispiel für die Filterung des Bildes: In (a) erkennt man das Ausgangsbild für das Tracking aus dem mittels 99% Perzentile Binärbild (b) errechnet wird. Bild (c) wurde mittels Filter bearbeitet und mit gleicher Perzentile in Binärbild (d) umgerechnet. Es ist deutlich zu sehen, wie der Filter die von der Sonne hell beleuchteten Bereiche gegenüber den anderen Bereichen abschwächt. Der Vogel in diesem Bild wird von dem roten Kreis eingeschlossen, kann hier aber mit dem Auge nicht erkannt werden, weshalb eine genaue Darstellung in Abb. 4 gegeben ist.

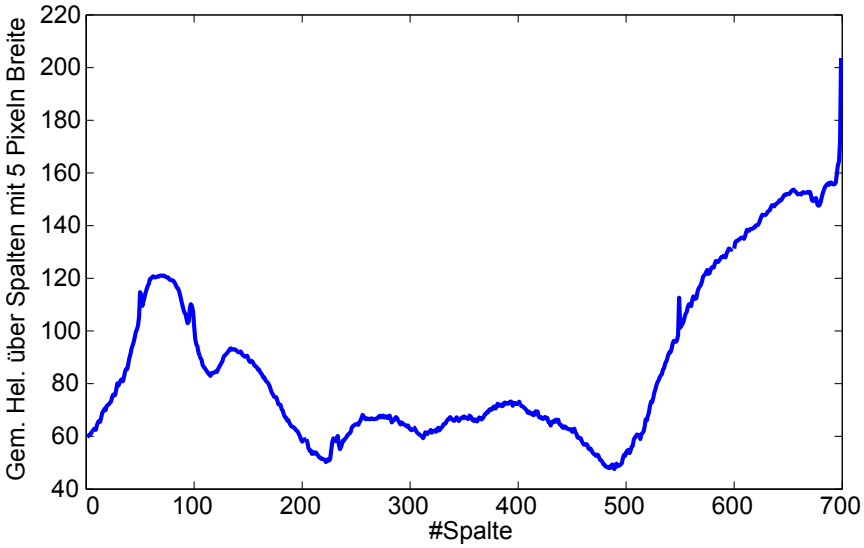
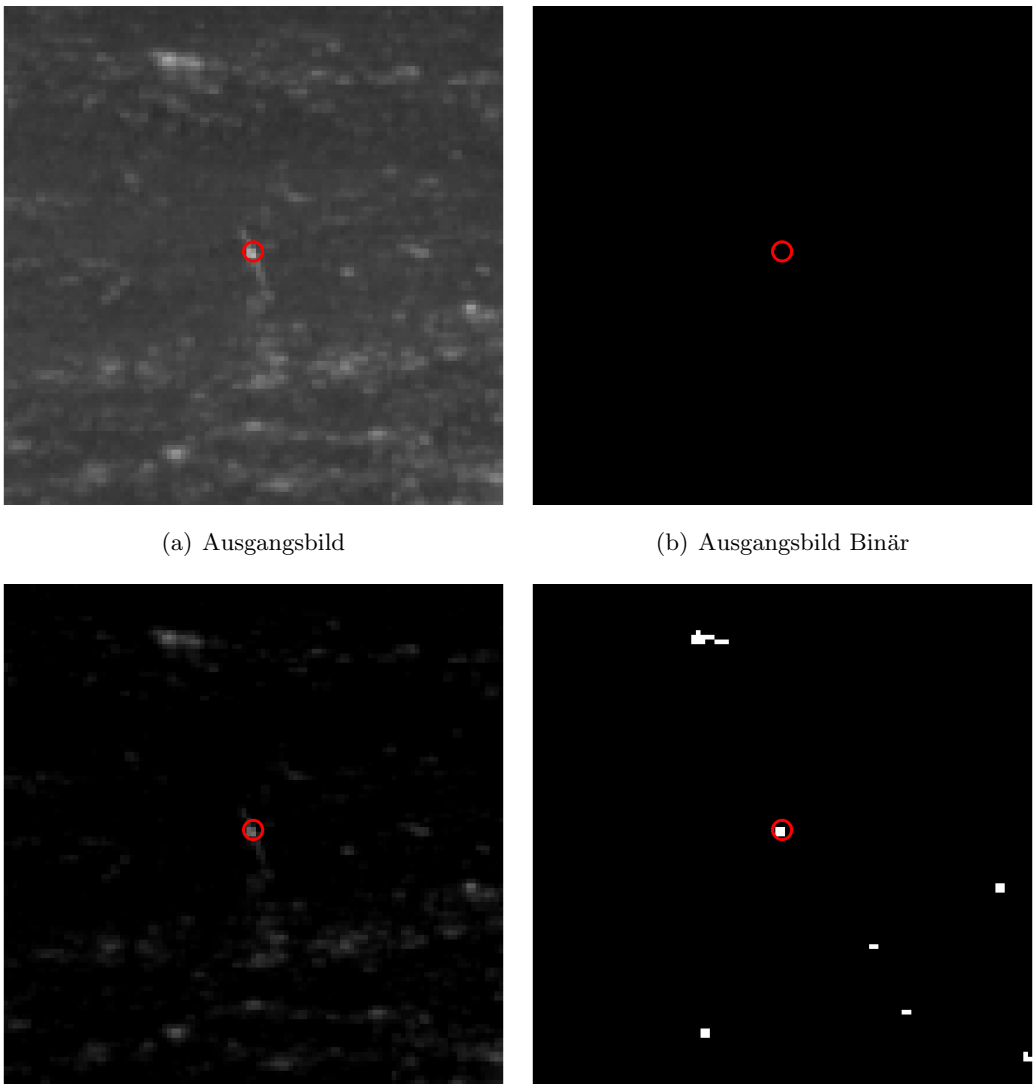


Abbildung 3: Errechnete Helligkeitsmittelwerte über die einzelnen Spalten des Bildes (hier Spalten mit je 5 Pixel). Der Effekt der ungleichmäßigen Reflexion des Sonnenlichts, welcher im Bild erkennbar ist, wird hier durch mittlere Helligkeit in den Spalten quantifiziert.

grunde, dass die Größe des Vogels selten bis nie in diesen Größenordnungen liegt und der Vogel im kleinsten Fall sowieso kaum sinnvoll verfolgbar wäre. Da alle im Bild befindlichen Punkte später mögliche Kandidaten für das Tracking darstellen, macht es Sinn, hier möglichst viele unpassende Kandidaten bereits zu entfernen. Im Laufe der Arbeit hat sich allerdings gezeigt, dass diese Grenzen sehr streng gesetzt werden müssen und deshalb wirklich nur kleinste und größte Flächen entfernt werden können, da sonst die Gefahr besteht, dass der Vogel dadurch verloren geht. Abb. 5 zeigt ein Beispiel für diesen Filter, wobei nur ein Ausschnitt (300 auf 300 Pixel) des Bildes dargestellt ist. In Matlab werden zunächst mittels `bwareaopen` im Bild die kleinsten Flächen aus dem Bild entfernt (Abb. 5 (d) zeigt diese kleinen Punkte). Anschließend wird ein Bild erzeugt, bei dem mit Hilfe von `bwareaopen` nur noch die größten Flächen enthalten sind (Abb. 5 (c)). Dieses wird dann vom ursprünglichen Bild abgezogen, wodurch man das bereinigte Bild erhält (Abb. 5 (b)). Diese beiden Filter sind in der Funktion `createTrackingImg.m` zusammengefasst. Im Rahmen der Entwicklung des Tracking Algorithmus wurden weitere Filtermethoden für die Rohdaten untersucht, die jedoch keine Verbesserung für das Tracking darstellten. Auf den ersten Blick erscheint eigentlich eine Hintergrundkompen-



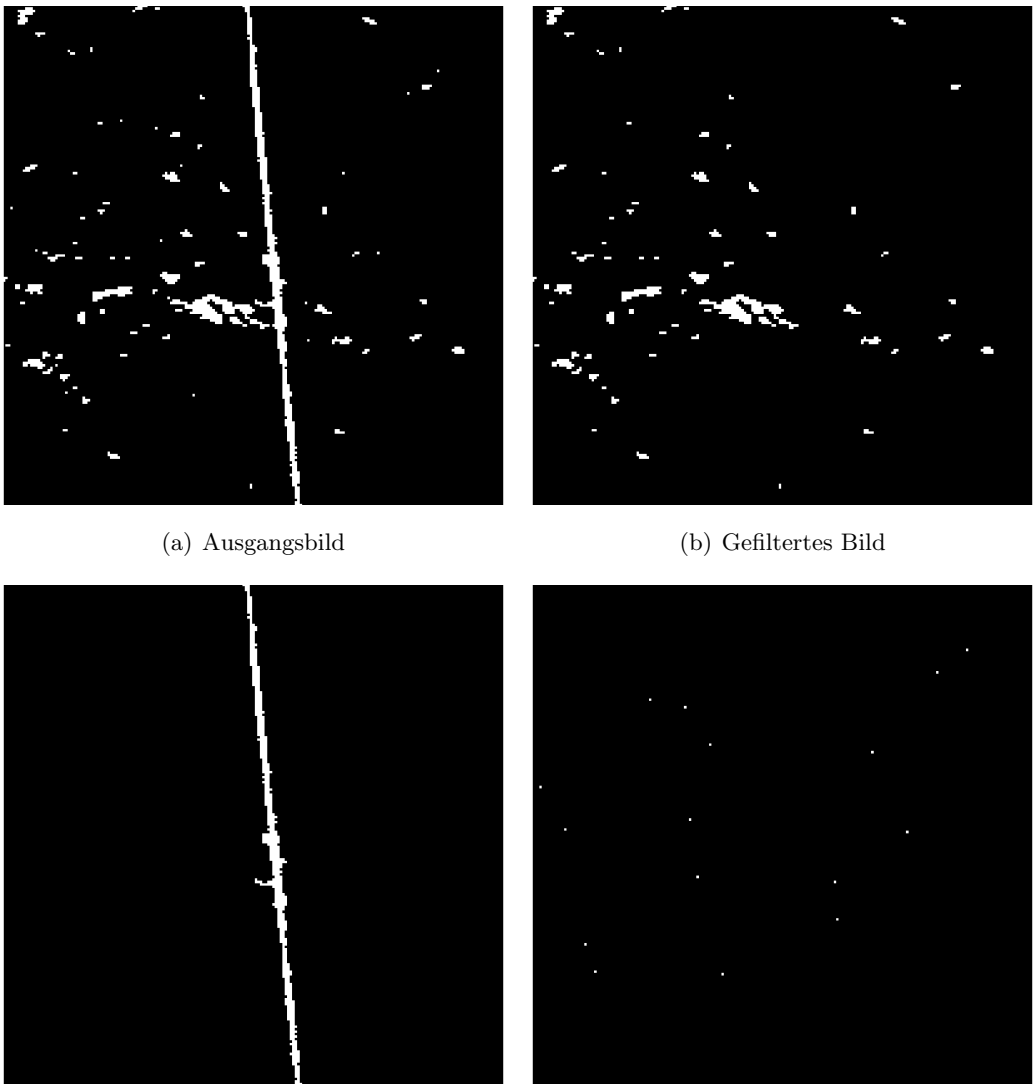
(a) Ausgangsbild

(b) Ausgangsbild Binär

(c) Gefiltertes Bild

(d) Gefiltertes Bild Binär

Abbildung 4: Beispiel für die Anwendung des Filters zur Reduzierung des Sonnenreflexion mit Zoom auf dem Vogel: Analog zu Abbildung 2 wird hier der Übergang vom Ausgangsbild (a) zu Binärbild (b) und gefiltertem Bild (c) zu Binärbild (d) gezeigt. In dieser Darstellung ist der Ausschnitt um den Vögel vergrößert, wodurch zu erkennen ist, dass beim Binarisieren der Vogel im Ausgangsbild verschwindet, während er beim gefilterten Bild als Punkt erhalten bleibt, obwohl in beiden Fällen die gleiche Perzentile verwendet wird.



(a) Ausgangsbild

(b) Gefiltertes Bild

(c) Bild mit großen Flächen

(d) Bild mit kleinen Flächen

Abbildung 5: Von Ausgangsbild (a) werden die sehr großen Flächen im Bild (c) (Flächen größer oder gleich 200 Pixel) und die sehr kleinen Punkte im Bild (d) (Flächen kleiner zwei Pixel) abgezogen, wodurch man das gefilterte Bild (b) erhält.

sation (*Background Substraction*) sinnvoll. Dabei wird das Bild über die letzten Schritte gemittelt und vom aktuellen Bild abgezogen, damit alle Teile des Bildes, welche sich in der Zeit nicht verändert haben, nahezu verschwinden. Bei den untersuchten Datensätzen zeigt dies keine Vorteile für das Tracking, da im Bild, durch die Fahrt des Schiffes und durch die Wellenbewegungen der Reflexionen, zu wenig unbewegter Hintergrund vorhanden ist. Alle untersuchten Varianten der *Background Substraction* (Mittelung über 1, 5 und 15 letzte Bilder) verschlechterten das Tracking in den getesteten Datensätzen.

3.3.2 Detektion der Vögel im gefilterten Bild

In dem in Abb. 2 (d) gezeigten Bild, also dem Bild aus dem bereits nicht relevante Objekte entfernt sind, befinden sich noch 1759 Flächen. Aufgrund dieser hohen Zahl und einer begrenzten Rechenleistung ist es notwendig die Auswahl für die Neudetektion eines Vogel weiter einzuschränken. Mit Neudetektion ist hier ein Objekt gemeint, dass noch nicht Teil eines Tracks ist und ab diesem Zeitpunkt verfolgt werden soll. Diese Auswahl wiederum beschränkt sich auf die wirklich hellsten Punkte im Bild (strenger als das 1% Perzentil). Diesem Vorgehen liegt die Überlegung zugrunde, dass ein Vogel, der getrackt werden soll, wenigstens einmal im Verlauf seiner Flugbahn hell genug ist, um das Tracking auszulösen. Ansonsten ist der Vogel einfach zu weit entfernt oder zu klein. Je nach gesetztem Grenzwert für diese Schwelle ist es möglich, die Anzahl der mitverfolgten Wellen zu erhöhen und gleichzeitig die Sicherheit, einen Vogel zu detektieren, zu erhöhen oder beides zu verringern.
In Matlab funktioniert dies mittels `regionprops(filteredImgBin,filteredImg, 'MaxIntensity')`, wobei hier `regionprops` die Matlab Funktion, `filteredImgBin` das Binärbild des gefilterten Bilds, `filteredImg` das gefilterte Bild in normaler Darstellung und `MaxIntensity` der Parameter ist. `regionprops` ist eine von Matlab zur Verfügung gestellte Funktion, die verschiedene Eigenschaften von Bildern analysieren kann. Durch die Kombination aus Binärbild und normalem Bild hat die Funktion bei diesem Aufruf Zugriff auf sowohl Informationen zu Helligkeitswerten wie auch zu den durch die Binärwerte festgelegten Flächen. `MaxIntensity` liefert dann den Helligkeitswert des hellsten Pixels einer Fläche zurück. Um dann den zugehörigen Mittelpunkt zur Fläcke zu erhalten wird `regionprops(filteredImgBin, 'Centroid')` verwendet, welches den Mittelpunkt einer Fläche (ohne Berücksichtigung der Helligkeit) zurückliefert. Die zu den `MaxIntensity`-Flächen gehörenden Mittelpunkte werden dann nach dem Wert von `MaxIntensity` sortiert und für die Detektion nur die hellsten verwendet. Wichtig zu erwähnen ist hierbei noch, dass die für das Verfolgen von vorhandenen Tracks verwende-

ten Mittelpunkte bereits aus dem Bild entfernt sind und keinen neuen Track beginnen. Für jeden so detektierten Mittelpunkt wird eine neue Instanz der Kalman Filter Klasse, ein konkretes Objekt mit den Eigenschaften der Klasse, zum Tracking angelegt.

3.4 Verfolgung der detektieren Punkte

Die Verfolgung der Tracks erfolgt mittels Kalman Filter wie in der Theorie bereits beschrieben. Dieser wurde in enger Zusammenarbeit mit Sebastian Richter als Klasse implementiert.

3.4.1 Eigenschaften des Kalman Filters

Für die Klasse des Kalman Filter werden die im Theorieteil gezeigten mathematischen Vorgaben direkt umgesetzt. Der Konstruktor setzt sich dabei folgendermaßen zusammen: `kf(numberOfKalmanFilter) = fx_kalman(id,frame,x_pos, y_pos,dt,u,w,v_x,v_y)`, wobei mit `id` die Id, `frame` das beginnende Bild, `x_pos`, `y_pos` die x- und y-Koordinate des ersten Punkts, `dt` die Zeiteinheit, `u` die Größe der Beschleunigung, `w` das Rauschen im Prozess und `v_x`, `v_y` das Meßrauschen des Kalman Filters übergeben werden. Versuche an einzelnen Tracks haben gezeigt, dass für diese Variablen die Werte `dt = 1`, `u = 0`, `w = .5`, `v_x = .1`, `v_y = .1` am besten geeignet sind. Hat man alle Variablen definiert erfolgt der Update Schritt analog zur mathematischen Beschreibung:

```
function obj = predict(obj,x,y)
    % predict next state
    obj.x_est = obj.A * obj.x_est + obj.B * obj.u;
    obj.pred_state = [obj.pred_state obj.x_est(1:4)];

    % predict covariance
    obj.P = obj.A * obj.P * obj.A' + obj.Ex;

    % calculate Kalman Gain
    obj.K = obj.P * obj.C' / ( obj.C * obj.P * obj.C' + obj.Ez);
```

`obj.x_est` entspricht dabei $\bar{\mu}_t$, `obj.P` entspricht Σ_T, `obj.Ex` entspricht R_t und `obj.C` direkt C_t. In `obj.pred_state` werden alle Werte von `obj.x_est` gespeichert, um den Track am Stück zu speichern. Für die Vorhersage des Punkts mit Messung muss zuerst unterschieden werden, ob im nächsten Bild ein sinnvolles Objekt als Vogel identifiziert werden kann oder nicht:

```
if(~isnan(x) && ~isnan(y))
    obj.x_est = obj.x_est + obj.K * ([x;y] - obj.C * obj.x_est);
```

Für den Fall, dass für x und y ein Wert gegeben ist, wird wieder direkt nach der mathematischen Beschreibung das `obj.x_est` mit dem Messwert aktualisiert. Andernfalls ist das bereits vorher berechnete `obj.x_est` der beste Vorhersagewert und es kann nichts weiter verbessert werden.

```
% update covariance estimation
obj.P = (eye(4) - obj.K * obj.C ) * obj.P;
```

Anschließend wird die neue Kovarianz berechnet und die Methode für die Vorhersage beendet. Im originalen Code finden sich noch weitere Variablen und Zähler, die beispielsweise mitzählen, wie oft ein Messupdate gemacht werden konnte und wann nicht, um später beurteilen zu können, wie zuverlässig der vorhandene Track ist. Weitere für meine Arbeit verwendete Variablen werden dann an gegebener Stelle erklärt. Eine weitere Methode der Klasse, `getPrediction(obj)`, erzeugt eine Vorhersage ohne `obj.pred_state` zu ändern, um abzuschätzen, wo die nächste Detektion liegen sollte, um wiederum eine erleichterte Suche nach der Detektion zu ermöglichen.

3.4.2 Anlegen eines Kalman Filters

Da die detektierten Punkte, die es durch die Auswahl geschafft haben, potenzielle Kandidaten für einen Vogel sind, wird jeweils ein neue Instanz (siehe 3.3.2) des Kalman Filter für diese angelegt. Das geschieht durch das Erhöhen der Anzahl an Kalman Filter und anschließendes Aufrufen des Konstruktor der Klasse mit der neuen Nummer. Dabei wird der x- und y-Wert des jeweiligen Mittelpunkts der Fläche, die aktuelle Id entsprechend der Nummer des Kalman Filters, die Nummer des aktuellen Bildes zur zeitlichen Zuordnung und `dt,u,w,v_x,v_y`, wie oben angegeben, zum Aufruf verwendet. Vor dem Anlegen des neuen Filter wird noch `if (kalman_x_pos ~= 0 || kalman_y_pos ~= 0)` überprüft, also ob `kalman_x_pos` und `kalman_y_pos` ungleich Null sind, da dies dann der Fall ist, wenn der jeweilige Mittelpunkt schon zum Verfolgen eines bestehenden Tracks verwendet wurde. Ist dies nicht der Fall, wird der neue Kalman Filter angelegt und weiterhin bis zur Deaktivierung des Filters in jedem Schritt ein Update erhalten.

3.4.3 Aktualisieren des Kalman Filters

Für die Aktualisierung der Kalman Filter dienen alle Mittelpunkte im gezeigten Binärbild. In einer Schleife über alle `actualKalmanFilter` wird für den jeweiligen Filter `predicted`

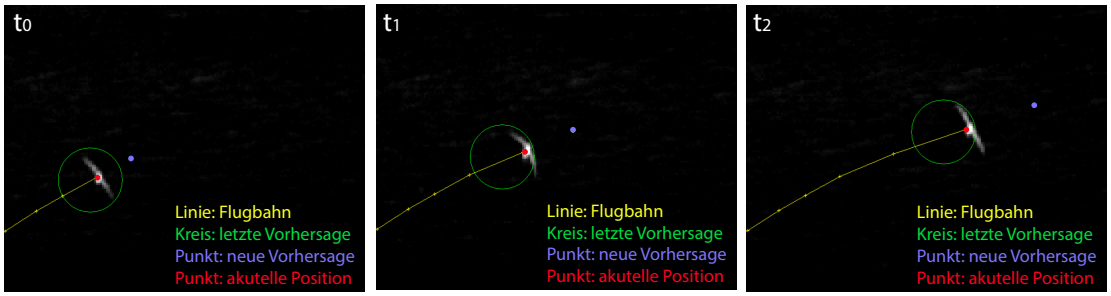


Abbildung 6: Beispiel für die Anwendung des Kalman Filter anhand von drei Zeitschritten. Der Vogel fliegt von links nach rechts durch das Bild, wobei er beschleunigt, da er in den letzten beiden Zeitschritten immer am rechten Rand der letzten Vorhersage und damit also etwas schneller als erwartet ist. Die Messung ist direkt im Schwerpunkt des Vogels und kann im Bild nicht dargestellt werden, da die aktuelle Position des Kalman Filter so nah an der Messung gewählt wird (d.h. sehr hohe Gewichtung der Messung), dass die Messung nicht erkennbar wäre.

`Point = kf(actualKalmanFilter).getPrediction';` der nächste Punkt ohne Update des Tracks vorhergesagt, um dann nach einem Mittelpunkt in der Nähe zu suchen, welcher als Messpunkt für den Vogel geeignet ist. Dieser wird nur durch die Nähe zum vorhergesagten Punkt ermittelt. Der Abstand muss so gewählt werden, dass nicht ständig Wellen in der Nähe verfolgt werden, wenn man den Vogel nicht erkennt, allerdings auch mindestens so groß, dass der Vogel jederzeit im Auswahlbereich liegt, wenn eine unerwartete Beschleunigung oder Abbremsung auftritt (was bei der projizierten Information auch bei vielen Richtungsänderungen bereits der Fall ist). Befindet sich keine helle Stelle im Binärbild im Umkreis, so erfolgt ein Update ohne neuen Messwert. Erfolgt ein Update mit Messwert wird der verwendete Mittelpunkt anschließend in x- und y-Koordianten auf 0 gesetzt um für weitere Messungen nicht mehr berücksichtigt zu werden. Ein Beispiel für die Aktualisierung eines Kalman Filter findet sich in Abb. 6.

3.4.4 Deaktivieren eines Kalman Filters

Schon während dem Tracking sollte eine Vorauswahl getroffen werden, welche Pfade weiter verfolgt werden sollen oder nicht, um Prozessorleistung zu sparen und das Programm nicht unnötig zu verlangsamen. Die verwendeten Grenzwerte sind dabei so gewählt, dass das Programm bei den untersuchten Datensätzen die bestmöglichen Ergebnisse liefert.

So wird ein Track, der öfter als 8 mal mit `kf(actualKalmanFilter).predict(NaN,NaN);` geupdatet wird, deaktiviert und damit ab diesem Zeitpunkt nicht weiter aktualisiert. Das bedeutet, dass ein Track, der wiederholt ohne Messung Updates erhält, deaktiviert wird. Anschließend werden die letzten 8 Schritte abgeschnitten, da diese nicht auf neuen Messwerten basieren.
Ist ein Pfad länger als 8 Schritte wird ab dieser Länge die zurückgelegte Pfadlänge mit der Anzahl an Zeitschritten verglichen. Ist dabei die Distanz `kf(actualKalmanFilter).travelDistance` zu klein gegenüber der Zahl der Zeitschritte, wird der Pfad deaktiviert. Dies trifft beispielsweise einen Track, der einfach auf verschiedenen hellen Wellenpunkten in der Nähe hin und her wandert.
Als weitere Maßnahme während dem Tracking wird die Anzahl der Aktualisierungen mit Messupdate mit der Anzahl an Aktualisierungen ohne Messupdate verglichen: Wird in mehr als 80% der Fälle ohne Messpunkt aktualisiert, wird der Track deaktiviert. Dies liegt darin begründet, dass ein Vogel, der sinnvoll verfolgt werden soll, in wenigstens 80% der Zeitschritte irgendwie im Binärbild erkennbar sein soll.
Um zu verhindern, dass ein Track, der sich auf einer Reflexion gebildet hat, anschließend auf einen Vogel überspringt und somit im Endbild ein Track aus beispielsweise halb Vogelflug und halb Reflexionswanderung entsteht, wird folgendermaßen vorgegangen: Ändert ein Track seine Helligkeit gemittelt über die letzten Bilder signifikant gegenüber dem Gesamtmittelwert, wird dieser Track an der Stelle deaktiviert. Der Grund dafür ist, dass Vögel und Wellen grundsätzlich im gleichen Datensatz unterschiedliche Helligkeitswerte haben sollten und somit an der Stelle der Track getrennt wird.
Diese vier Maßnahmen sind dabei bewusst sehr grob gehalten. Sie sollen allerdings auch nur die große Menge an mitverfolgten Reflexionen reduzieren ohne dabei zum Verlust von wirklichen Vogeltracks zu führen. Genau diese Aufgabe erfüllen die Kriterien bei bisher allen überprüften Datensätzen. Alle auf diese Weise deaktivierten Pfade werden später noch daraufhin überprüft, ob es sich um einen Vogel handeln könnte. Die Deaktivierung dient also nur der Einsparung von Rechenleistung bei Tracks, die sowieso nicht sinnvoll weiter verfolgt werden können. Durch die Deaktivierung der nicht benötigten Tracks läuft das Tracking je nach Prozessorleistung beinahe in (i5-3317U) oder teilweise sogar ganz in Echtzeit (i7-2600K). Dabei ist allerdings noch anzumerken, dass der Code nicht auf Geschwindigkeit optimiert ist und dass durch gezieltere Vektorielle Programmierung bessere Geschwindigkeiten erreicht werden können. Auffällig ist jedoch, dass das Tracking zum Ende eines Datensatzes hin langsamer wird, da mit jedem Zeitschritt trotz Deaktivierung immer mehr Kalman Filter betrachtet werden müssen (nach den

verwendeten Parametern übersteigt die Anzahl der neuen also die Zahl der deaktivierten Filter). Bedingt dadurch, kann es bei größeren Datensätzen von deutlich über einer Minute wieder zu Verlangsamungen kommen, was allerdings behoben werden kann, wenn alle paar Minuten die Tracks deaktiviert und ggf. neu gestartet werden. Speichert man neben dem reinen Tracking zu Demonstrationszwecken oder zur späteren Analyse auch noch einzelne Bildausschnitte der jeweiligen Tracks ist das Programm natürlich erheblich verlangsamt und braucht für eine Minute rund 45 Minuten Laufzeit.

3.5 Klassifizierung der Tracks mittels einer Support Vector Machine

Nachdem alle Tracks in einem Datensatz aufgenommen sind, werden diese mit Hilfe einer Support Vector Machine (SVM) klassifiert, um Vogeltracks von anderen zu unterscheiden, da pro Minute mehrere hundert Tracks aufgenommen werden. Ein Vorteil hierbei ist, dass Matlab bereits eine SVM in der Statistics Toolbox anbietet, welche gut dokumentiert [16] und somit einfach zu verwenden ist. Die grundsätzliche Aufgabe einer SVM ist das Trennen von Daten anhand der Eigenschaften des dazu gehörigen Feature Vector, der die verschiedenen analysierten Eigenschaften des jeweiligen Tracks enthält. Dabei wird versucht eine Hyperfläche zwischen die Datenpunkte im *Feature Space* zu legen und deren Position wird durch die Abstandsbestimmung mittels Innerer Produkte (im Prinzip Skalarprodukte) der Feature Vektoren festgelegt . Die Feature Vektoren werden dabei in einem hochdimensionalem Raum dargestellt, in dem dann die Punkte mit unterschiedlichen Eigenschaften bestmöglich getrennt werden. Für das Innere Produkt wird eine sogenannte Kernel Funktion $K(x_1, x_2) =< \Phi(x_1), \Phi(x_2) >$ verwendet, wobei die beiden Φ die Features darstellen. Die Kernel Funktion kann dabei beispielsweise linear, oder auch, wie im später verwendeten Fall, eine radiale Basisfunktion (*Gaussian Radial Basis Function*, kurz RBF) $K(x_1, x_2) = \exp(-(x_1 - x_2)^2/2\sigma)$ sein. Im einfachsten Fall eines linearen Kernels wird dabei also versucht eine ungekrümmte Hyperebene zwischen die Daten zu legen. Für die Verwendung der SVM muss diese zuerst mit einem gelabelten Datensatz, also Daten, die in Vögel und *Anderes* bereits unterteilt sind, trainiert werden. Durch das Training wird dann die Hyperebene festgelegt, welche die Daten im *Feature Space* möglichst gut trennt. Diese Hyperebene wird von den namensgebenden *Support Vectors* aufgespannt. Es handelt sich also bei den *Support Vectors* um diejenigen Samples, welche am nächsten an der trennenden Hyperebene liegen. Durch die Verwendung des RBF Kernels anstelle eines linearen Kernels besteht diese Hyperebene statt aus einer ungekrümmten (multidimensionalen) Fläche auch aus radialen Basisfunktionen. An dieser Stelle kommt der später verwendete Parameter σ ins Spiel, der der

skalierende Faktor im Exponenten der RBF ist. Der zweite später verwendete Parameter ist der Soft Margin bzw. Boxconstraint. Dieser kommt ins Spiel, wenn die Daten durch die Hyperebene nicht ohne Fehler getrennt werden können: Der Gesamtfehler durch alle Datenpunkte, die die Trennung durch eine Hyperebene verhindern, muss minimiert werden, wobei der Boxconstraint hierbei eine Gewichtung für den Fehler angibt. Wird also der Boxconstraint besonders hoch gewählt, wird die Ebene sehr gut an die gegebenen Trainingsdaten angepasst, um die Fehler wirklich zu minimieren, kann dann allerdings zu sehr auf den Datensatz zurecht geschnitten sein und bei den Testdaten gegebenenfalls schlecht funktionieren (*Overfitting*). Setzt man den Boxconstraint zu niedrig, wird eine Falschklassifizierung bei den Trainingsdaten weniger streng bewertet und die Hyperebene damit allgemeiner gehalten. Abb. 7 stellt ein zweidimensionales Beispiel für die

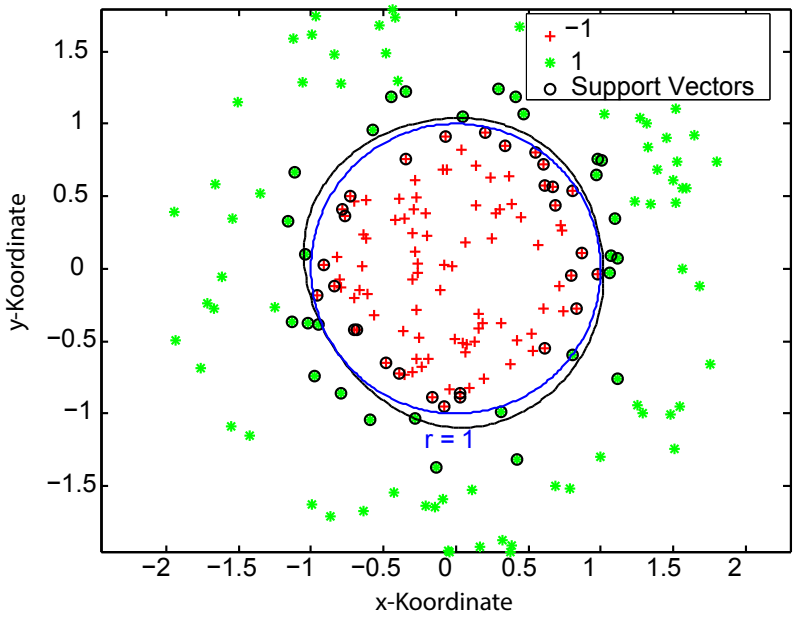


Abbildung 7: Einfaches Beispiel einer SVM mit RBF als Kernel: Punkte innerhalb eines Kreises mit Radius 1 (Label: -1) und Punkte außerhalb des Kreises (Label: 1) werden mittels SVM getrennt. Der blaue Kreis zeigt den Kreis mit Radius 1, die schwarze Linie stellt die Hyperebene der SVM zur Trennung der Punkte dar. Durch Umrandung markiert sind die Punkte, welche als *Support Vectors* die Trennung stützen.

Verwendung einer SVM mit RBF als Kernel dar [16]. Die durch die *Support Vectors* (mit

schwarzer Umrandung markierte Punkte) aufgespannte schwarze Linie, die Hyperebene, welche die Punkte voneinander trennen soll, stellt hier mit den Standardwerten für den Boxconstraint nahezu einen Kreis dar. Erhöht man den Boxconstraint erhält die Linie immer mehr Beulen und fittet den gegebenen Datensatz dadurch besser. Jedoch entfernt sie sich dadurch auch von der eigentlich kreisförmigen Trennung durch die blaue Linie. Aus diesem Grund muss für eine SVM, um die bestmögliche Lösung zu erhalten, ein Trainingsdatensatz mit einem getrennten Testdatensatz über verschiedene Parameter iteriert werden, bis die bestmögliche Aufteilung gefunden wird (*Cross-validation*).
Für die Beurteilung der Qualität der Aufteilung durch die SVM können mehrere Größen verwendet werden. Grundsätzlich gibt es zwei Arten von richtiger Einteilung und zwei Arten von Fehler: True positive (tp) heißt ein Punkt, der richtigerweise als positiv markiert wurde, true negativ (tn) heißt, dass ein Punkt der richtigerweise als negativ markiert wurde, false positive (fp) heißt, einen Punkt fälschlicherweise als positiv und false negativ (fn) einen Punkt fälschlicherweise als negativ markiert zu haben. Daraus lassen sich verschiedene handlichere Größen ableiten. Die *Precision*, definiert durch $\frac{tp}{tp+fp}$, gibt dabei den Anteil an richtig klassifizierten Daten unter den relevanten positiven Ergebnissen an. Bei der *Precision* werden allerdings die fn- und tn-Werte garnicht berücksichtigt, welche für die umfassende Beurteilung der SVM auch eine Rolle spielen. Dagegen gibt der *Recall* $\frac{tp}{tp+fn}$ das Verhältnis der relevanten Punkte im Ergebnis zu den insgesamt relevanten Punkten an. Der eigentlich natürlichste Wert ist die *Accuracy*:

$$\frac{tp + tn}{tp + tn + fp + fn} \tag{21}$$

Diese gibt einen Wert für den Anteil an richtiger Klassifizierung in der Gesamtmenge an. Ist der Wert für die *Accuracy* also 100%, sind alle Punkte richtig klassifiziert. Da in den später zur Verfügung stehenden Daten der Anteil an Vogeltracks im Gegensatz zu anderen Tracks allerdings sehr gering (rund 10%) ist, stellt die *Accuracy* hier kein gutes Maß für die Quantifizierung dar. Beispielsweise könnte die SVM alles als falsch deklarieren und würde damit eine *Accuracy* von ca. 90% erreichen, obwohl diese letztendlich nichts gemacht hätte. Aus diesem Grund wird bei Datensätzen mit deutlich verschiedenen Klassengrößen der *f1-Score*

$$\frac{2tp}{2tp + fp + fn} \tag{22}$$

verwendet, der das harmonische Mittel aus *Precision* und *Recall* darstellt. Damit gibt der *f1-Score* ein Maß für *Precision* und *Recall* zusammen an, welches gut für die Beurteilung der Effektivität der SVM geeignet ist. Wie die anderen Größen reicht der Wert

von 0% bis 100%, berücksichtigt aber insbesondere ungleichmäßig verteilte Klassen. Eine weitere im Lauf der Arbeit verwendete SVM ist die LIBSVM [17], welche ebenfalls direkt mit Matlab Interface verfügbar ist und Funktionen über die Matlab interne SVM hinaus bietet, wie beispielsweise die unterschiedliche Gewichtung von Eigenschaften.
Die für die SVM verwendeten Eigenschaften der Tracks setzen sich zusammen aus den Ableitungen und den zur Analyse von Random Walks verwendeten Methoden und Eigenschaften der Tracks aus den Bildern: Mittelwert der ersten Ableitung, Mittelwert der zweiten Ableitung, Mittelwert der `noUpdateMatrix`, Straightness Index, MSD und Eigenschaften der Turning Angle Distribution (Varianz, Mittelwert, Kurtosis). Erste und zweite Ableitung sind hierbei ein Maß für Geschwindigkeit und Beschleunigung des Vogels, die natürlich entfernungsabhängig sind, jedoch trotzdem dazu beitragen können, Vogelflugbahnen von anderen zu unterscheiden. Der Mittelwert der `noUpdateMatrix` dient auch hier als Maß des prozentualen Anteil an Messupdates, der bei Vögeln gegenüber beliebigen Reflexionen deutlich höher liegen sollte, da die Vögel im Gegensatz zu den Reflexionen nicht plötzlich verschwinden können und sich lediglich in manchen Fällen nicht genug vom Hintergrund abheben. Der Straightness Index, wie oben erklärt, dient hier als Maß für die Persistenz der Fortbewegung und sollte bei Vögeln, die die meiste Zeit aktiv Strecke im Bild zurücklegen deutlich höher ausfallen. Allerdings ist hier anzumerken, dass durch Interpolation und Auftauchen von Reflexionen an anderen weiter entfernten Stellen auch diese manchmal in sehr kurzer Zeit weite Strecken zurücklegen können. Bei der MSD wird für Vögel durch die gezielte Flugbahn eine superdiffusive oder beinahe ballistische Flugbahn erwartet, weshalb für die MSD hier ein Wert im Bereich 1.5 bis 2 erwartet wird. Das wohl wichtigste Kriterium für die Analyse der Tracks ist die Turning Angle Distribution. Entscheidend trägt dazu bei, dass die Turning Angle Distribution im Gegensatz zu den anderen Kriterien unabhängig von der Entfernung des Vogels zur Kamera ist. Um den maximalen Nutzen aus der Turning Angle Distribution zu ziehen wird diese für $\Delta t = 1$, $\Delta t = 2$ und $\Delta t = 3$ Zeitschritte berechnet. Größere Zeitspannen machen aufgrund der durschnittlichen Tracklänge von knapp 20 Zeitschritten keinen Sinn und würden zum Auschluss von zu vielen Tracks führen. Die Parameter werden dann normalisiert und zusammen mit einem von Hand bestimmten Anteil an Tracks der SVM übergeben, welche damit für die Unterscheidung zwischen Vögeln und anderen Tracks trainiert wird.

3.6 Bestimmung der Entfernung des Vogels für 3D Tracks

Ein großer Nachteil der Daten von einer einzigen Kamera ist, dass jegliche Tiefeninformation fehlt und damit wichtige Größen wie Entfernung, Geschwindigkeit und die umfassende Flugbahn nicht genauer analysiert werden können. Im Bild ist nicht unterscheidbar, ob ein Vogel gerade steht, ob er sich auf direkter Sichtlinie von der Kamera weg oder auf die Kamera zu bewegt. Auch wäre es mit bekannter Entfernung möglich, die Größe des Vogels anhand der Anzahl an hellen Pixel oder der Länge der Halbachsen einer einschließenden Ellipse anzugeben. Mit der Größe wäre dann schon eine gewisse Einschränkung auf einige Arten möglich. Aus diesen Gründen macht es Sinn, sich mit der Entfernungsbestimmung vom Schiff aus zu befassen. Für die Entfernungsbestimmung von Meeressäugern gibt es bereits eine sehr gute Näherung basierend auf [18], der allerdings zugrunde liegt, dass sich die Meeressäuger bei Sichtung immer direkt auf der Oberfläche befinden. Auf diese Weise lässt sich mittels Entfernung zum Horizont der Abstand vom Schiff zuverlässig bestimmen. Da Vögel ab einer gewissen Entfernung vom Schiff meist auf einer ziemlich konstanten Höhe über dem Meeresspiegel fliegen, kann man beispielsweise eine Flughöhe von fünf Metern annehmen und dafür die Entfernung abschätzen. Abb. 8 skizziert das Prinzip dieser Abschätzung für Vögel in einer Flughöhe b über dem Horizont. Mit trigonometrischen Beziehungen lassen sich die Winkel α und β durch

$$\alpha = \arctan(\sqrt{2hR + h^2}/R) \tag{23}$$

und

$$\beta = \pi/2 - \alpha - \sigma \tag{24}$$

bestimmen. Dabei ist σ durch den Abstand vom Vogel zum Horizont über eine Beziehung mit dem Öffnungswinkel der Kamera gegeben[2]. Mittels Kosinussatz ergibt sich für den Abstand auf Sichtlinie D_0 der Zusammenhang:

$$D_0 = (R + h)\cos(\beta) - \sqrt{(R + h)^2 \cos^2(\beta) - (2R(h - b) + h^2 - b^2)} \tag{25}$$

Aus diesem Abstand lässt sich über $\delta = \arcsin(\sin(\beta)\frac{D_0}{R})$ der Abstand vom Schiff bestimmen:

$$D = \delta R \tag{26}$$

[2]Bei einem vertikalen Abstand n des Vogel zum Horizont im Bild: σ ist gleich n multipliziert mit dem Öffnungswinkel geteilt durch die vertikale Auflösung der Kamera. Der Öffnungswinkel der Kamera beträgt 18° bei einer vertikalen Auflösung von 576 Pixel.

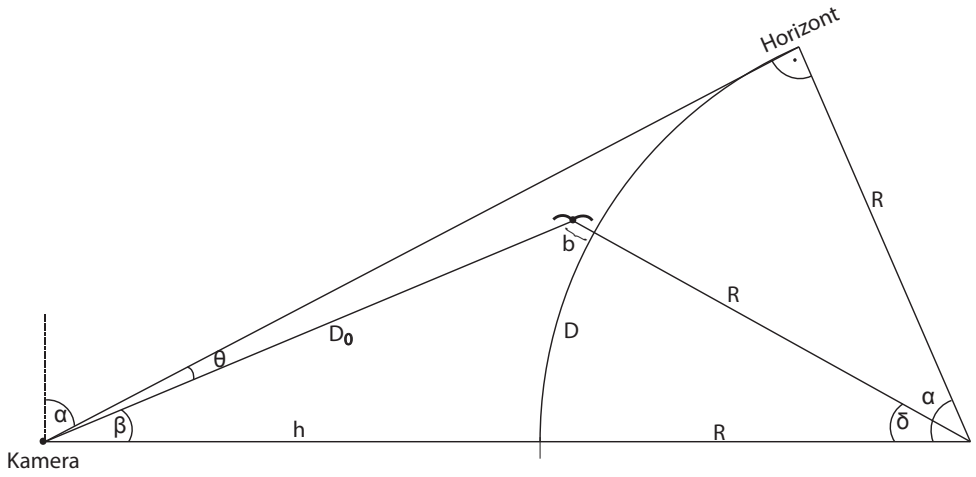


Abbildung 8: Entfernungsbestimmung zum Vogel anhand des Horizonts: Die Kamera (im Bild links unten) mit Aufnahme in Richtung Horizont, wobei sich in Entfernung D vom Schiff ein Vogel befindet. Die Kamera befindet sich auf Höhe h über der Meeresoberfläche und diese ist in der Skizze zur Verdeutlichung aller Größen sehr groß gewählt. R stellt den Erdradius dar.

Schätzt man die Entfernung für einen Vogel in angenommenen fünf Metern Flughöhe ab und lässt einen Fehler von einem Meter nach oben und unten zu, ergibt sich die Abschätzung in Abb. 10 zu den in Abb. 9 gezeigtem Track: Abb. 10 zeigt, dass der Fehler an sich akzeptabel wäre, jedoch erkennt man leider bei genauerer Betrachtung auch, dass der Vogel sich nach dieser Abschätzung innerhalb von weniger als vier Sekunden von einer Entfernung von ca. 200 m zum Horizont in 20 000 m Entfernung bewegt. Damit muss dann die Abschätzung einer Flugbahn in der Höhe von rund 5 m aufgegeben werden. Was an diesem Beispiel recht schnell gezeigt werden kann trifft leider auf die meisten der Flugbahnen zu. Damit würde eine Abschätzung der Entfernung zu viele Sonderfälle und Ausnahmen enthalten, bei denen sie nicht gilt, als dass sie als sinnvolle Größe für die Unterscheidung der verschiedenen Flugbahnen geeignet wäre.

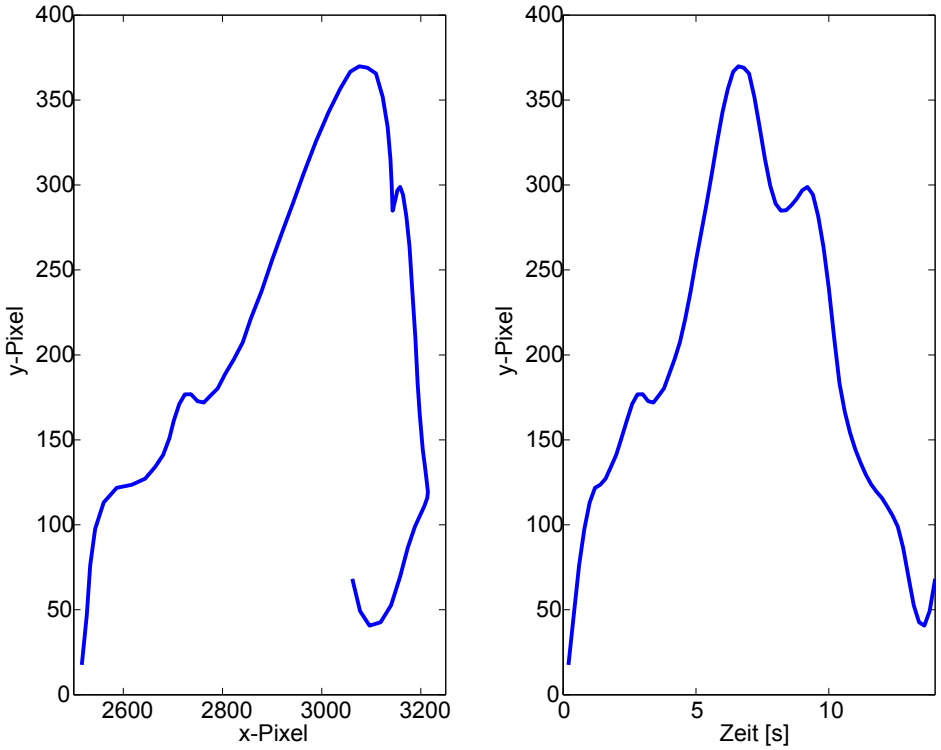


Abbildung 9: Links: Verlauf der Flugbahn im Ausgangsbild. Rechts: Track mit Auftragung des y-Pixel über der Zeit (Da für die Abschätzung nur die y-Komponente relevant ist).

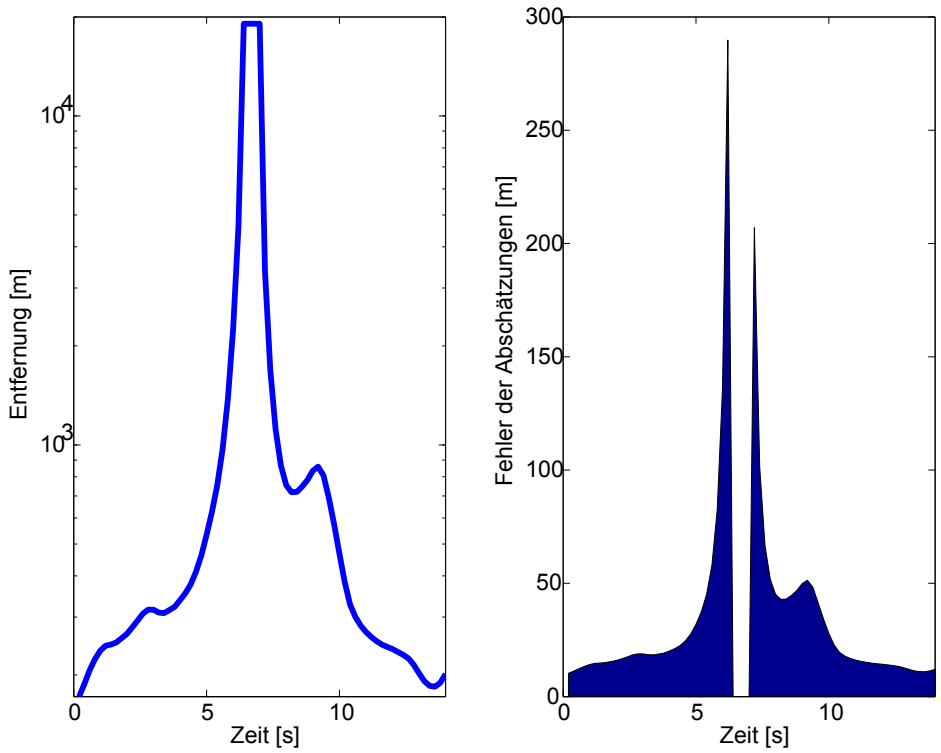


Abbildung 10: Links: Abschätzung der Entfernung bei einer Flughöhe von fünf Metern. Die Konstante Entfernung im Bereich zwischen sechs bis sieben Sekunden ergibt sich dadurch, dass der Vogel in diesem Zeitraum über dem Horizont ist und somit als am Horizont befindlich angenommen wird. Rechts: Fehler in der Abschätzung der Entfernung bei der Annahme einer Flugbahn mit zwischen vier und sechs Metern Flughöhe über dem Meeresspiegel. Auch hier ist der Fehler am Horizont Null, da sich da kein Unterschied abhängig von der Flughöhe ergibt. Abgesehen vom Horizont liegt der Fehler dabei immer bei ca. 6% des Wertes für die Entfernung in beide Richtungen.

3.7 Aufbau eines Correlated Random Walk als Modell

Um die Analyse Methoden umfassender testen zu können ist ein Modell für den Vogelflug notwendig. Bei der Analyse wird davon ausgegangen, dass die Flugpfade als Random Walks analysiert werden können, daher liegt es auch nahe, hierfür das gezeigte Modell eines Correlated Random Walk zu verwenden. Die Implementierung hält sich dann auch hier direkt an die oben gezeigte mathematische Formulierung:

```
x(counter,:)= q * x(counter-1,:) + sigma*randn(1,3);
```

Um für Vögel passende Ergebnisse zu liefern ist für `q` ein Wert zwischen 0.85 und 1 zu wählen, während `sigma` = 2 ist. Erstaunlich ist hierbei, wie gut dieses einfache Modell rein qualitativ mit dem des Vogelflugs zusammenpasst. Andere Implementierungen hingegen, die vielfach und mit verschiedensten Verteilungen erprobt wurden, zeigen nicht im entferntesten eine solche Ähnlichkeit. Diese Ähnlichkeit macht Abb. 11 an einem Beispiel deutlich. Begründet liegt diese wohl auch darin, dass der Kalman Filter, als lineare Interpolation für Punkte mit Gaußschem Messfehler, gut für das Tracking geeignet ist, der damit sehr ähnlich zum Aufbau des verwendeten Random Walk ist. Um die Random Walks, welche ursprünglich in 3D implementiert sind, wie die Vogeltracks auf eine Ebene zu projizieren wird folgende Projektionsmatrix in die x-y-Ebene verwendet:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{27}$$

Zur Ausführung dieser Projektion müssen die Punkte des Random Walks zunächst in homogenen Koordinaten dargestellt werden und dazu zu einem Vektor mit vier Zeilen erweitert werden, wobei für die vierte Zeile eine 1 gesetzt wird. Die Projektion wird dann durch eine Multiplikation mit der Projektionsmatrix ausgeführt, wobei die gezeigte Projektionsmatrix in die x-y-Ebene projiziert. Der Vektor nach der Multiplikation enthält (bedingt durch die Eigenschaften der Matrix) die gleichen Werte für die dritte und vierte Komponente. Durch eine Normierung dieser Komponenten auf 1 (Division des Vektors durch den Wert der Komponente) ergeben sich für die erste und zweite Komponente die x- und y-Werte des neuen Vektors. Damit die Projektion in diese Ebene immer ohne Probleme ausgeführt werden kann, werden die Tracks vor der Projektion um +500 Pixel in z-Richtung verschoben, wodurch die Entfernung zur Ebene sichergestellt ist. 500 ist hierbei grundsätzlich beliebig gewählt, steht aber in der gleichen Größenordnung zur

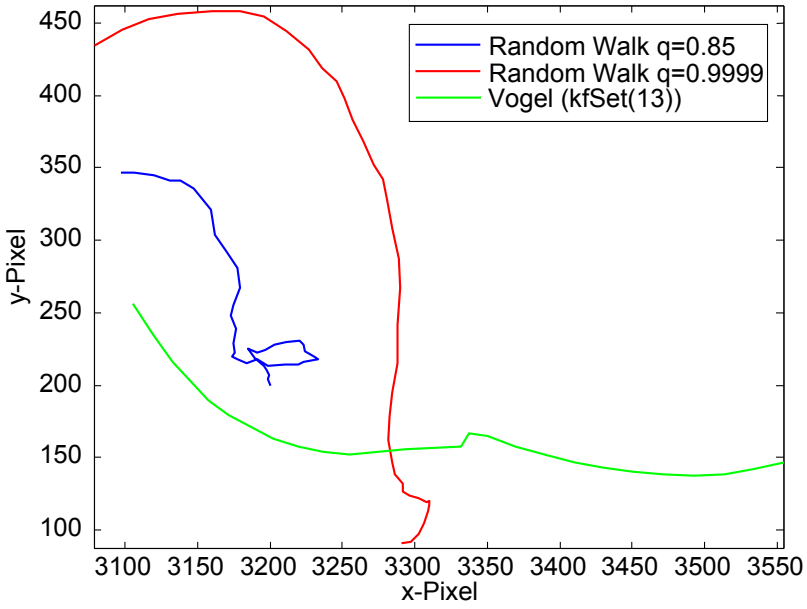


Abbildung 11: Beispiel für zwei Random Walks mit unterschiedlicher Persistenz q und einen Vogeltrack. Die Ähnlichkeit des Random Walks mit hoher Persistenz und des Vogeltracks ist qualitativ gut.

Tracklänge bei den Random Walks wie die ungefähre Tracklänge zur Entfernung zur Kamera bei den Vögeln. Auf eine spezialisierte Projektionsmatrix mit Anpassung an die verwendete Kamera wird bewusst verzichtet, da dies neben einem schwierigen Aufwand für die Berechnung (bedingt durch das spezielle Format der Kamera mit Rotation bei den Aufnahmen) kaum Änderungen ergeben würde und daher der Unterschied vernachlässigt werden kann. Insbesondere sind die Änderungen wesentlich geringer als die ohnehin gegebene Abweichung der Random Walks von den natürlichen Vogelflugbahnen. Die Arbeit mit den Random Walk hat für das Testen der Analyse-Methoden jetzt deutliche Vorteile: Ohne großen Aufwand können beliebig viele verschiedene Tracks erzeugt werden, durch die Parameter können einfach verschieden stark kurvige Tracks erzeugt werden und die Länge der Tracks kann nach Belieben festgelegt werden. Damit ergibt sich insbesondere die Möglichkeit, Tracks in verschiedenen Längen und verschieden starker Unterschiede zu generieren und zu testen wie gut man diese voneinander unterscheiden kann. Für einen solchen Versuch zur Unterscheidbarkeit von Random Walks mit unterschiedlichen Parametern für q werden jeweils 50 Pfade erstellt (die Größenordnung ist

hier bewusst klein gehalten, da auch das Ansammeln und Labeln von Vogeltracks einen zeitlich hohen Aufwand darstellt) und diese dann mittels `kmeans`, welches bereits direkt in Matlab verfügbar ist, und der SVM analysiert. `kmeans` ist eine Methode zur Clusteranalyse, die auf der Vektorquantisierung basiert, das heißt dass die Vektoren immer dem ähnlichsten Merkmalsvektoren zugeordnet werden. Wichtig ist hierbei, dass `kmeans` eine Form des unüberwachten Lernens (*Unsupervised Learning*) ist und damit ohne Trainingsdatensatz auf den Daten arbeiten kann. Dabei hat jeder Track einen Punkt in einem Multidimensionalen Raum und `kmeans` versucht eine Aufteilung in Cluster zu finden, bei der die Objekte zum Clustermittelpunkt jeweils eine möglichst geringe Entfernung haben. Gleichzeitig soll die Entfernung zu Objekten in anderen Clustern möglichst groß sein. Die Vorgehensweise ist dabei iterativ, es werden also die Clustermittelpunkte nach und nach bewegt, bis keine bessere Positionierung mehr gefunden werden kann. Eine detailliertere Erklärung zu `kmeans` ist in der Mathworks Dokumentation [19] zu finden. Für die Aufteilung mit `kmeans` werden die gleichen Features wie im Kapitel Selektion der Tracks verwendet, abgesehen von der `noUpdateMatrix`, welche für die Random Walks natürlich nicht vorhanden ist. Besonders interessant ist dabei die Aufteilung für verschiedene Tracklängen zu betrachten, um überprüfen zu können, wie gut das Ergebnis mit der Tracklänge skaliert.

f

4 Ergebnisse und Diskussion

4.1 Vogeltracks

Die im Rahmen des Tracking entwickelten Methoden wurden an mehreren Datensätzen getestet. Diese umfassen Datensätze der Expedition ARK-XXIV mit einigen Auschnitten à ein oder zwei Minuten Länge am 27.06.2009, 30.06.2009 und 03.07.2009 auf dem Weg von Bremerhafen in die Grönlandsee. Ein weiterer Datensatz ist von der Expedition ANT-XXVIII mit der Fahrtroute von Kapstadt zur Neumayer-III-Station auf dem Meridian von Greenwich vom 28.12.2011. Besonders herausfordernd ist letzterer Datensatz, weil dieser sehr viele Eisschollen enthält. Abb. 12 zeigt einige Beispiele für von

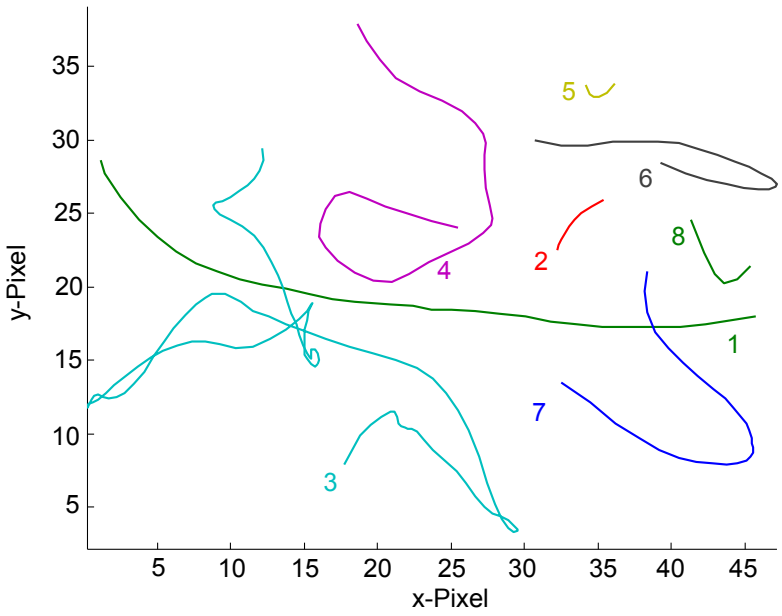


Abbildung 12: Darstellung einiger ausgewählter Vogeltracks aus den Daten. (Die Daten sind zur Darstellung skaliert und verschoben)

Hand selektierte Vogeltracks in den Datensätzen. Hierbei fällt bereits auf, dass es ein paar sehr lange Tracks (Beispiel 1 und 3), jedoch auf der anderen Seite auch sehr viele kurze Vogeltracks (Beispiel 2 und 5) gibt. Betrachtet man in Abb. 13 weitere Tracks in den Datensätzen, bei denen kein Vogel dabei ist, fällt eine ähnliche Aufteilung für die

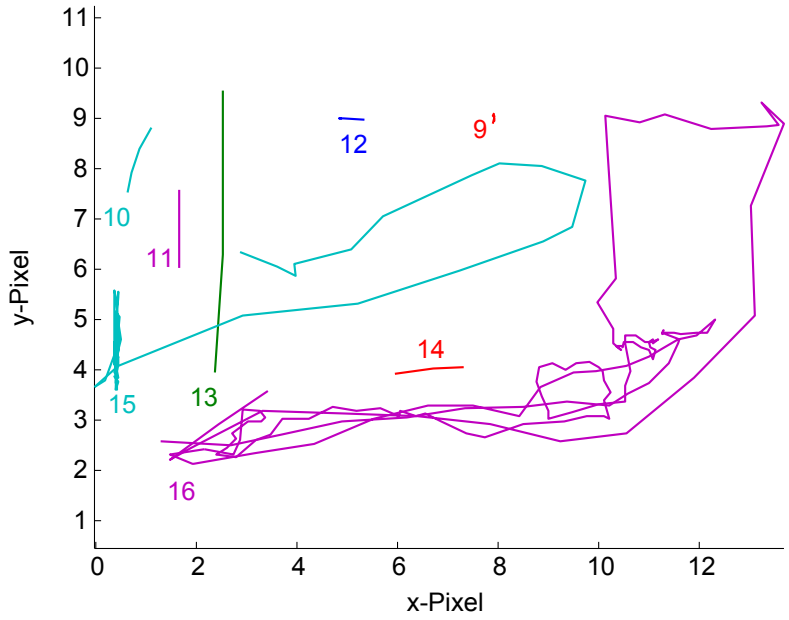


Abbildung 13: Darstellung einiger ausgewählter Nicht-Vogeltracks aus den Daten, wie beispielsweise Reflexionen, Eisschollen, etc. (Die Daten sind zur Darstellung skaliert und verschoben)

Tracklänge auf. Der größte Teil dieser Tracks sind Reflexionen des Sonnenlichts auf der Wasseroberfläche und dabei größtenteils auf Wellenbergen. Eigentlich wäre zu erwarten, dass diese Reflexionen dann passend zur Schiffsbewegung eine bestimmte mittlere Geschwindigkeit aufweisen, allerdings ist dies in den Datensätzen nicht der Fall. Grund dafür ist wohl, dass die Reflexionen in sehr vielen einzelnen Bildern nicht vorhanden sind und der Pfad dadurch auf die nächste Reflexion in der Nähe überspringt, woraus folgt, dass es keine Vorzugsrichtung für die Bewegung gibt. Insgesamt gleicht die Bewegung dieser Reflexionen damit stark einem völlig unkorrelierten Random Walk. Wird in der Nähe keine weitere Reflexion gefunden, werden diese Pfade oft durch den Kalman Filter interpoliert, wodurch teilweise lange geradlinige Strecken zustande kommen. Eine weitere Quelle für diese Tracks sind Reflexionen auf Eisschollen, die durch ihre weiße Färbung besonders viel Sonnenlicht reflektieren und durch ihre Größe auch meistens mehrere Reflexionen in der Nähe anbieten, wodurch hier selten interpoliert wird und der Track über eine lange Strecke auf der Eisscholle verläuft. Beispiel 16 in Abb. 13 zeigt diesen Effekt. Die Tracks können dadurch sehr lang werden und reißen in einem Datensatz teilweise

nicht einmal ab. Durch die beiden Phänomene der Wellen und Eisschollen sind die wohl kritischsten und häufigsten Quellen von unbrauchbaren Tracks abgedeckt. Ein paar weitere Tracks kommen hinzu durch Reflexionen auf Teilen des Schiffes, wie beispielsweise der Reling oder durch Fehlstellen im Bild, welche bei der Zusammensetzung aus der 360° Drehung der Kamera entstehen. Die letzten beiden Fehlerquellen können durch eine Deaktivierung dieser Bereiche für das Tracking vermieden werden, stellen jedoch nur einen sehr kleinen Teil der zusätzlichen Tracks dar, wodurch sich der Aufwand dafür nur bedingt lohnt. Eine ähnliche Deaktivierung für das Tracking wäre auch auf Eisschollen denkbar, allerdings würden dabei dann natürlich alle Vogeltracks verloren gehen, die über Eisschollen führen.

4.2 Selektion der Vogeltracks aus der Gesamtheit der Tracks

In den Datensätzen finden sich dann insgesamt 127 Vogel- und 1570 andere Tracks. Aus diesem Grund ist es für eine völlig automatisierte Umsetzung sehr wichtig, die Vogeltracks automatisch von den anderen zu trennen. Ermöglicht wird dies durch die oben beschriebene Methode der SVM. Analysiert man die Daten dabei mit verschiedenen

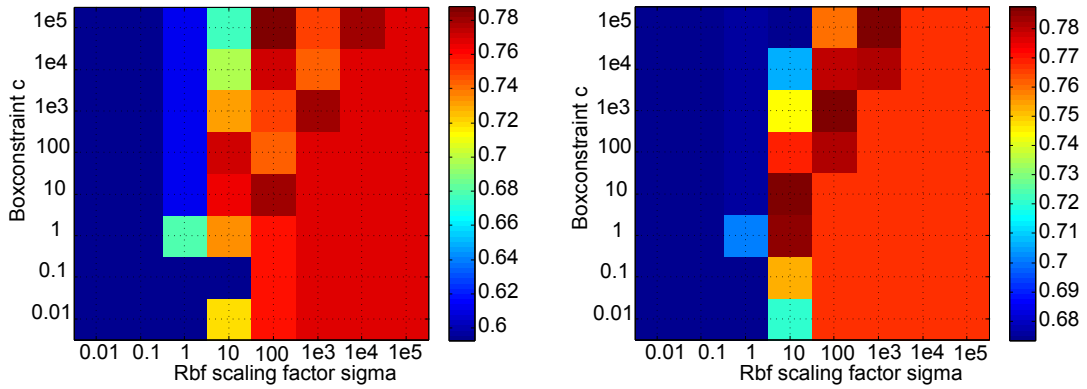


Abbildung 14: Ermittelung ob es sich um einen Vogel oder um eine Reflexion handelt mittels SVM. Darstellung des *f1-Score* auf der Farbskala abhängig von Sigma für die RBF und dem Boxconstraint c. (Weitere Parameter: Verhältnis der Trainingsdaten zu Testdaten 2:1, wobei über alle drei Gruppen trainiert, getestet und anschließend gemittelt wird, Tracklängen sind ab 21 Zeitschritten berücksichtigt)

Parametern Sigma für die Gaussian Radial Basis Function (RBF) und verschiedenen Boxcontraints c ergibt sich Abb. 14. Die Darstellung zeigt den oben erklärten *f1-Score*

(22), da die *Accuracy* bei einem Datenverhältnis von 1570 anderen Tracks zu 127 Vogeltracks keine sinnvolle Größe darstellt. Die Darstellungen in Abb. 14 sind jeweils drei zufällig ausgewählte Gruppen (mittels `randi(3,length,1)`), die wiederum jeweils im Verhältnis 3:1 Trainingsdaten zu Testdaten untersucht werden. Jede der drei Gruppen wird dabei jeweils einmal als Testgruppe verwendet, während die anderen beiden Gruppen als Trainingsdaten herangezogen werden. Anschließend wird über alle Durchläufe gemittelt und der gemittelte *f1-Score* dargestellt. An den zwei Darstellungen über die Variablen Boxconstraint und Sigma mit unterschiedlichen Werten für den besten *f1-Score* zu den jeweiligen Werten wird deutlich, dass sich auch bei gleichen Boxconstraint und Sigma in beiden Abbildungen unterschiedliche Werte für den *f1-Score* ergeben. So ist im linken Bild das Maximum bei ca. 79% für einen Boxconstraint von 10^5 und einem Sigma von 10^2, während im rechten Bild das Maximum bei 79% bei 10^5 und 10^3 liegt. Eine Untersuchung in Richtung höheren Boxconstraint ist durch die deutlich erhöhte Zeit bis zur Konvergenz in einen festen Zustand nicht möglich. Durch die Unterschiede in den beiden Plots wird deutlich, dass die Testgruppe zu klein ist und damit Schwankungen durch die unterschiedliche Gruppenwahl zu hohe Einflüsse haben. Für eine genauere Ermittlung der Parameter ist eine größere Menge an gelabelten Daten notwendig. Allerdings deutet das Ergebnis mit einem *f1-Score* von maximal rund 80% schon deutlich an, dass mit den verwendeten Parametern eine Trennung zwischen Vogeltracks und anderen Tracks möglich ist. Eine größere Menge an Trainings- und Testdaten würde dieses Ergebnis noch weiter verbessern, ist allerdings auf der anderen Seite auch aufwendig zu erzeugen. Abb. 15 zeigt im linken Bild ein weiteres Beispiel für eine Ermittlung der Werte analog zu Darstellung 14, während im rechten Bild die gleiche Ermittlung mit einer minimalen Tracklänge von 13 Zeitschritten statt 21 Zeitschritten gezeigt wird. Dabei wird sofort deutlich, dass das Maximum mit rund 67% *f1-Score* unter dem Wert für längere Tracks liegt, da die Klassifizierung für kurze Tracks wesentlich schwieriger ist. Tracks mit unter 13 Zeitschritten werden vor der Klassifizierung in jedem Fall entfernt, da diese auf keinen Fall eine Aussagekraft über die Vogelart enthalten. Außerdem kommt es bei dieser Tracklänge bereits zu Problemen bei der Berechnung der MSD und der Turning Angle Distribution mit $\Delta t = 3$. Um die Relevanz der Ergebnisse in Abb. 15 genauer verstehen zu können ist es notwendig, die Verteilung der aufgenommenen Tracklänge in den Datensätzen zu betrachten. Das linke Bild in Darstellung 16 zeigt die jeweilige Anzahl an Vogeltracks zu einer gegebenen Länge und macht deutlich, dass der Großteil der Vogeltracks sehr kurz ist. Dies liegt darin begründet, dass zum einen ein großer Teil der Vögel nur sehr kurz durch das Bild fliegt, beispielsweise von unten in das Bild und

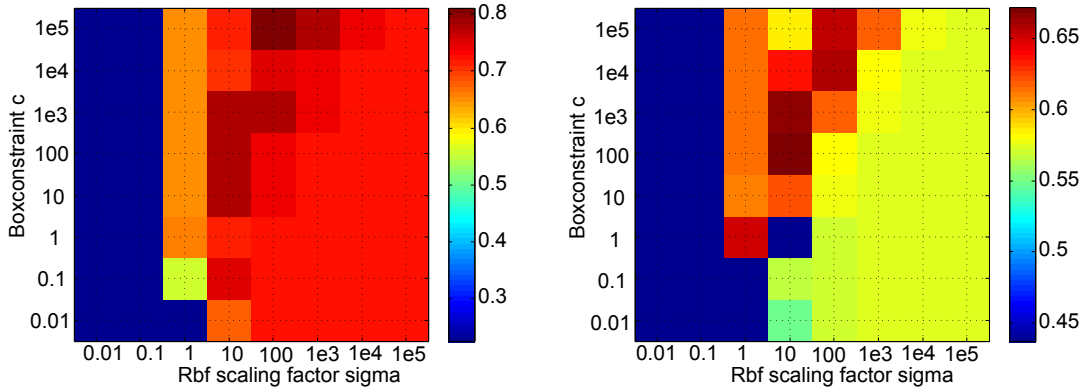


Abbildung 15: Ermittelung ob es sich um einen Vogel oder um eine Reflexion handelt mittels SVM. Darstellung des *f1-Score* auf der Farbskala abhängig von Sigma für die RBF und dem Boxconstraint c. (Weitere Parameter: Verhältnis der Trainingsdaten zu Testdaten 2:1, wobei über alle drei Gruppen trainiert, getestet und anschließend gemittelt wird) Links: Tracklängen ab 21 Zeitschritten (analog zu Abb. 14), Rechts: Tracklängen ab 13 Zeitschritten.

dann mit einer 180° Kurve direkt wieder aus dem Bild. Außerdem reißen bereits beim Tracking einige der Pfade wieder ab, wenn der Vogel in mehreren Bildern weniger deutlich sichtbar ist, zum Beispiel durch seine Orientierung zur Kamera oder durch den Flug über einer Eisscholle. Bei diesen häufigen kurzen Längen von rund 15 Zeitschritten (ein großer Teil liegt unter der Durchschnittslänge von 19.4 Zeitschritten) ist die Berechnung der MSD und vor allem der Turning Angle Distribution gerade so möglich, allerdings dann durch die wenigen Datenpunkte weniger aussagekräftig. Insgesamt ist nach Abb. 15 also abzuwägen, ob es sinnvoll ist, die kürzesten Tracks zu entfernen und nur mit Tracks ab einer Länge von ca. 20 Zeitschritten zu arbeiten.

4.3 Untersuchung verschiedener Random Walks mittels SVM

Da ein Großteil der Ungenauigkeit bei der Einteilung mittels SVM auf den Daten damit einher geht, dass die Tracks zu kurz und die Anzahl an gelabelten Daten im Moment zu gering ist, macht es Sinn, hierbei wieder das Modell des Correlated Random Walks für das Testen der Analyse Methoden zu verwenden. Vorteil hierbei ist, dass Daten beliebiger Tacklänge und durch den geringen Rechenaufwand in nahezu unbegrenzter Menge mittels der Random Walks erzeugt werden können. Mit oben gezeigten Random

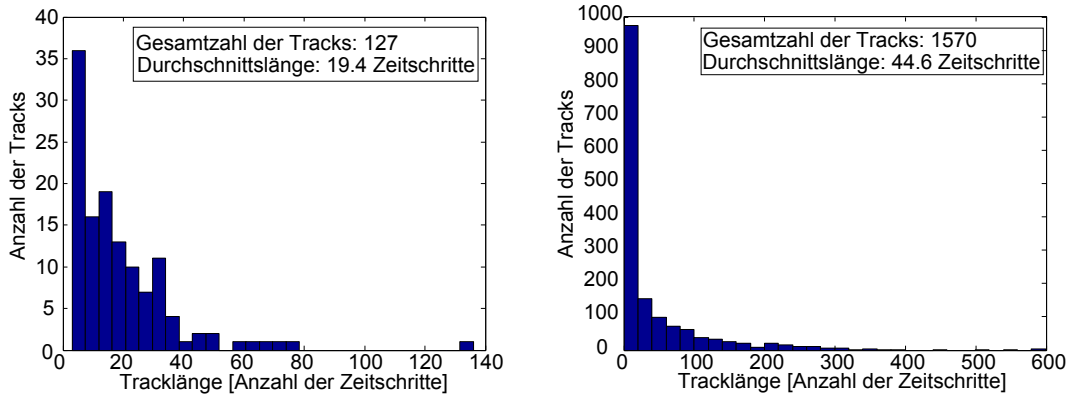


Abbildung 16: Auftragung der Anzahl an Tracks mit der jeweiligen Länge über der Tracklänge. Links: Darstellung für Vogeltracks, Rechts: Darstellung für die anderen Tracks.

Walk Modell ergeben sich im Bereich einer Persitenz (q) von 0.85 bis 0.9999 sinnvolle Tracks, die denen von Vögeln ähneln. Erzeugt man damit zwei Gruppen Random Walks mit unterschiedlicher Persistenz und analysiert diese analog zu den Vogeltracks mit einer SVM, um die Tracks mit unterschiedlichen Werten für die Persistenz zu trennen, ergibt sich, wie in Abb. 17 gezigt, eine Möglichkeit, die Qualität der Aufteilung über der Tracklänge aufzutragen. In der Darstellung ist für die Qualität der Aufteilung die *Precision* $tp/(tp + fp)$ aufgetragen, welche hier durch die gleich großen Gruppen einen guten Wert für die Einschätzung liefert. Vor der Untersuchung der Plots wurde analog zu Abb. 14 der beste Parameterwert für Boxconstraint und Sigma zu je 10^2 ermittelt. Dabei wird deutlich, dass die Trennung der beiden Gruppen q=0.95 und q=0.9999 (in blau) wesentlich schlechter erfolgt als die Trennung der Gruppen q=0.85 und q=0.9999 (in rot), da die Ähnlichkeit durch den geringeren Unterschied bei der Persistenz natürlich höher ausfällt. Besonders interessant ist jedoch der Verlauf der einzelnen Kurven: Die Trennung von q=0.85 und q=0.9999 steigt für kleine Tracklängen mit zunehmender Länge sehr stark an, sodass sich ein logarithmischer Verlauf zeigt, der dann ab einer Tracklänge von ca. 100 Zeitschritten und einer *Precision* von rund 95% stagniert. Damit ist in diesem Fall also eine sinnvolle Größe für die im besten Fall notwendige Tracklänge bei ca. 100 Zeitschritten erreicht. Eine weitere Verbesserung des Trackings darüber hinaus würde dann kaum mehr Vorteile bringen. Anders hingegen sieht dies beim Verlauf der Kurve für q=0.95 und q=0.9999 aus, bei dem die Tracks schon eine sehr hohe Ähnlichkeit aufweisen. Hier steigt die *Precision* der Aufteilung über den gesamten Verlauf von einer

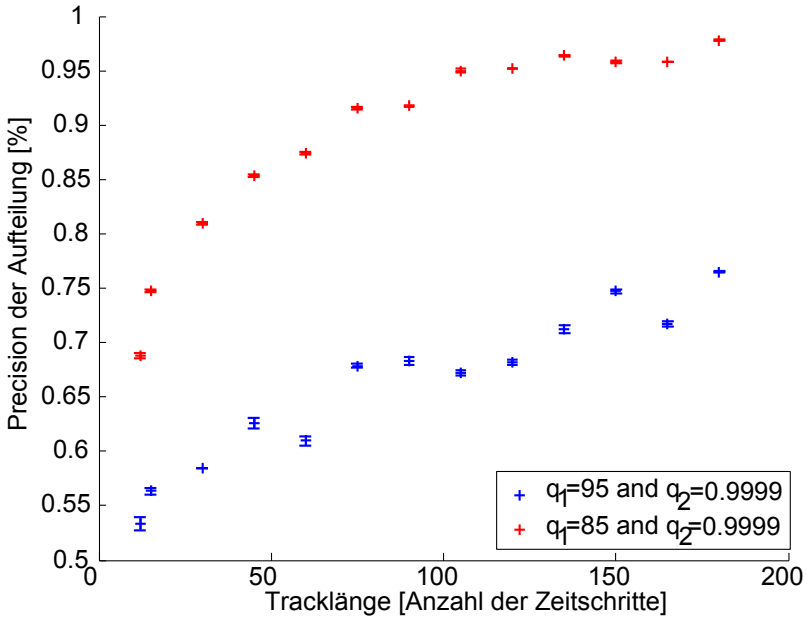


Abbildung 17: Trennung von zwei Gruppen à 40 Random Walks mit unterschiedlicher Persistenz q mittels SVM. Darstellung einer Mittelung über mehrere Trainings- und Testdaten mit Fehlerbalken. (Genaue Parameter: Verhältnis der Trainingsdaten zu Testdaten 3:1, wobei über alle vier Gruppen jeweils trainiert, getestet und anschließend gemittelt wird, Boxconstraint und Sigma je 10^2)

Tracklänge von 12 bis 200 nahezu linear an und es ist hier zu erwarten, dass es auch darüber hinaus noch Verbesserungen gibt. Vergleicht man die Größenordnungen mit denen für die oben gezeigten Vogeltracks, dann scheinen die Vogeltracks gegenüber den anderen Tracks aber eher im Bereich der Unterscheidbarkeit von q=0.85 und q=0.9999 zu liegen, wodurch jede weitere Verbesserung am Tracking für längere Tracks bereits einen großen Sprung in Richtung einer besseren Aufteilung liefern sollte. Sind mittels SVM dann die Vogeltracks von den anderen extrahiert steht der Weg frei, den eigentlich spannendsten Aspekt der Arbeit zu betrachten: Ist es möglich, die Vögel in verschiedene Klassen, im besten Fall analog zur Art der Vögel, zu unterteilen? Da die bisher gesammelten Vogeltracks in ausreichender Länge für diesen Versuch eindeutig zu wenig sind, wird auch dies mit dem Correlated Random Walk Modell betrachtet. Da die Unterteilung der Vogeltracks automatisch erfolgen soll, muss hier eine Methode des *Unsupervised*

Learning verwendet werden. Dafür bietet sich das in den Methoden beschriebene `kmeans` aus Matlab an. Betrachtet man mit `kmeans` Random Walks mit unterschiedlicher Persistenz (q) ergibt sich Abb. 18. Analog zur SVM Analyse bei den Random Walks ist hier

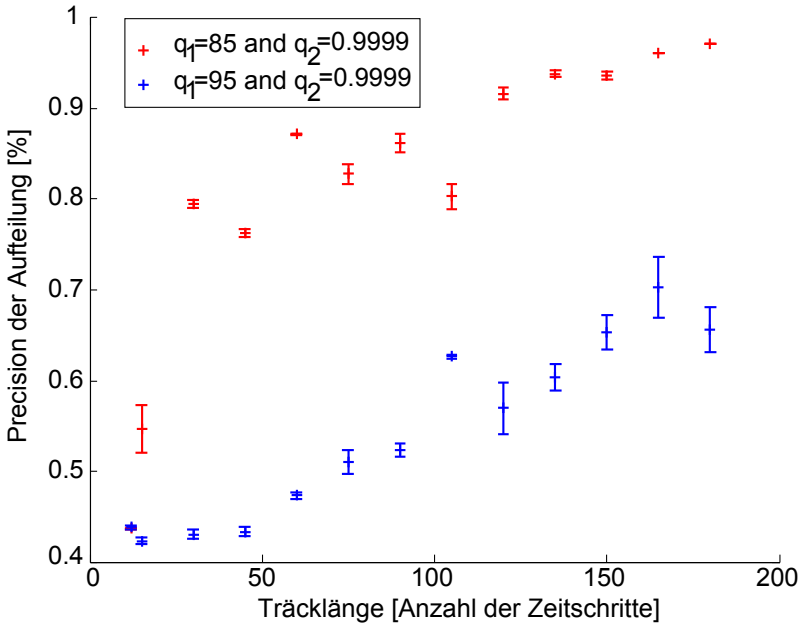


Abbildung 18: Trennung von zwei Gruppen à 50 Random Walks mit unterschiedlicher Persistenz q mittels kmeans. Darstellung einer Mittelung über 6 Testdatensätze je Tracklänge mit Fehlerbalken.

die *Precision* über der Tracklänge dargestellt. Die Werte liegen hier in einer ähnlichen Größenordnung wie bei der SVM. Dabei ist lediglich die *Precision* im Schnitt ein wenig niedriger, was bei einer Aufteilung ohne vorherigem Training zu erwarten ist. Bedingt durch eine Analyse ohne vorherigem Training gibt es hier auch größere Schwankungen zwischen den einzelnen Durchläufen, was man an den deutlich größeren Fehlerbalken erkennt. Insgesamt ist aber besonders vielversprechend, dass die Aufteilung ohne vorherigem Training ähnlich gut funktioniert, wie die Aufteilung mit Training. Dadurch kann für die Möglichkeit, die Daten nach einem Training auf die Unterscheidung zwischen Vogeltracks und anderen automatisch zu analysieren, eine hohe Erwartung ausgesprochen werden.

5 Zusammenfassung und Ausblick

Zusammenfassend ist es durch das Programm für das Tracking möglich, Vögel in den Datensätzen zu detektieren und anschließend zu verfolgen. Die dabei entstehenden Tracks für Vögel und andere Dinge, wie Reflexionen auf Wellen oder Eisschollen, können anschließend mittels einer ausreichend trainierten SVM getrennt werden. Für die automatisierte Trennung der Vögel in unterschiedliche Klassen deutet die Analyse an den Correlated Random Walks mittels `kmeans` eine vielversprechende Methode an.

Weiteres Verbesserungspotential sehe ich besonders in einer weiteren Optimierung des

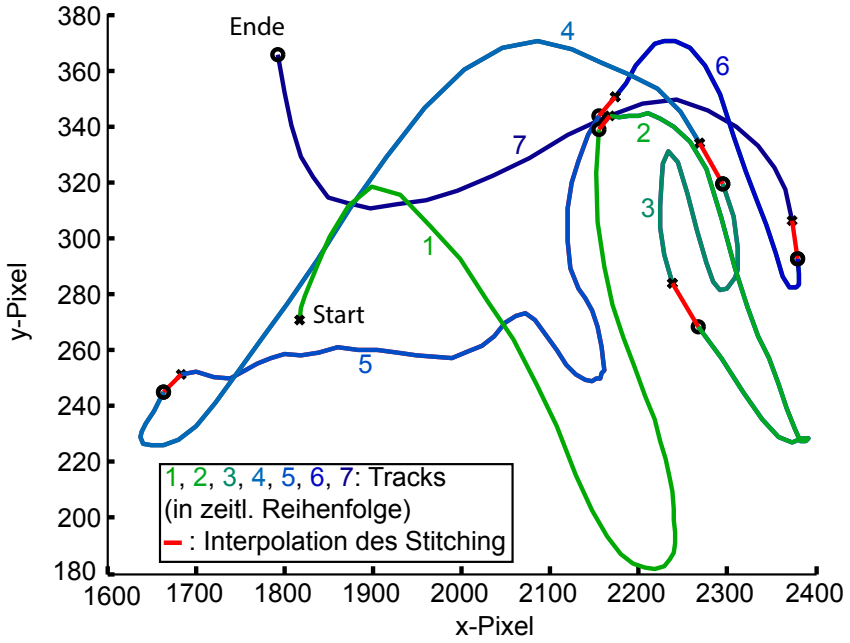


Abbildung 19: Beispiel für die Verwendung des *Stitching* nach [20]. Die Tracks sind hierbei in zeitlicher Reihenfolge von 1 bis 7 nummeriert und der Anfang jeweils durch ein schwarzes x, das Ende durch einen schwarzen Kreis o markiert. Durch eine Analyse der Dynamik der Bewegung wird der jeweils folgende Pfad zugeordnet und bei ausreichender Ähnlichkeit linear verbunden. Die sieben Flugbahnen sind dabei 21 bis 47 Zeitschritte (im Durchschnitt 30 Zeitschritte) lang, während der gesamte Track 207 Zeitschritte (ohne den interpolierten Zeitschritten) umfasst. Die Lücken zwischen den Tracks betragen je rund drei Zeitschritte.

Tracking und damit längeren Tracks, da die Einteilung mit SVM und `kmeans` stark von einer erhöhten Tracklänge profitieren würde. Möglichkeiten der Verbesserung sind an dieser Stelle in einem weiteren Verfeinern des Algorithmus für das Tracking, beispielsweise durch eine weitgehendere Analyse der Kandidaten für den nächsten Zeitschritt anhand von Form oder Größe. Ein weiterer Ansatz besteht darin, die Tracks nicht so schnell abzuschneiden, sondern im Zweifelsfall über mehrere Pfade fortzusetzen und anschließend automatisiert den geeignetsten auszuwählen. Vielversprechend scheint insbesondere auch das Paper *The Way They Move: Tracking Multiple Targets with Similar Appearance* von Caglyan Dicle, Mario Sznaier und Octavia Camps der Northeastern University [20], das mit bereits vorhandenem Programmcode eine Möglichkeit bietet, einzelne Trackabschnitte anhand von jeweiliger Geschwindigkeit und Bewegungsrichtung wieder miteinander zu verbinden (*Stitching*). Abb. 19 zeigt ein Beispiel, bei dem mittels dieser Methode sieben zusammengehörige Tracks miteinander verbunden werden. Der Einfachheit halber wurde hierfür ein Datensatz verwendet in dem die Lücken zwischen den Tracks nur recht klein sind. Mit abgestimmten Parametern sehe ich in dieser Methode jedoch noch deutlich mehr Potential.

Eine völlig andere Herangehensweise für die Analyse ist die reine Bildanalyse der Vögel. Die Grafiken in Abb. 20 zeigen die Möglichkeiten dafür. In der oberen Reihe sieht man Vögel deren Umriss recht deutlich im Bild zu erkennen ist. Dafür ist es dann möglich, eine Abschätzung der Form oder auch des Verhältnisses von Flügelspannweite zu Körpergröße anzugeben. Für die meisten Tracks trifft es auch zu, dass der Vogel zumindest einmal während seiner Flugbahn so orientiert ist, dass man die Form gut erkennen kann. Auf der anderen Seite gibt es auch Vögel, wie in der unteren Reihe in Abb. 20, die entweder zu klein oder zu weit weg sind, um wirklich einen sinnvollen Umriss zu erkennen. Kombiniert mit einer groben Entfernungsabschätzung, wie in Kapitel 3.6 erklärt, lässt sich damit auch eine Abschätzung für die Größe der Vögel geben.

Funktioniert die Einteilung der Vögel in verschiedene Klassen mit ausreichender statistischer Signifikanz, muss die Einteilung mindestens für einen Datensatz mit einer genauen Sichtung durch Ornithologen am Schiff abgeglichen werden, um auszuschließen, dass die Trennung der Vögel in Klassen nicht beispielsweise zwischen nahen und fernen Vögeln, sondern zwischen verschiedenen Spezies erfolgt. Eine Basis für die Erstellung einer *Ground Truth* ist also notwendig. Im Gegensatz zu den bisherigen Berichten des Alfred-Wegener-Instituts [21] ist für diese Verifikation allerdings eine genaue Dokumentation der Sichtung (im besten Fall mit Richtung und geschätzter Entfernung) und nicht nur eine Aufzählung der Gesamtzahl an Sichtungen notwendig. Dies ist natürlich ein

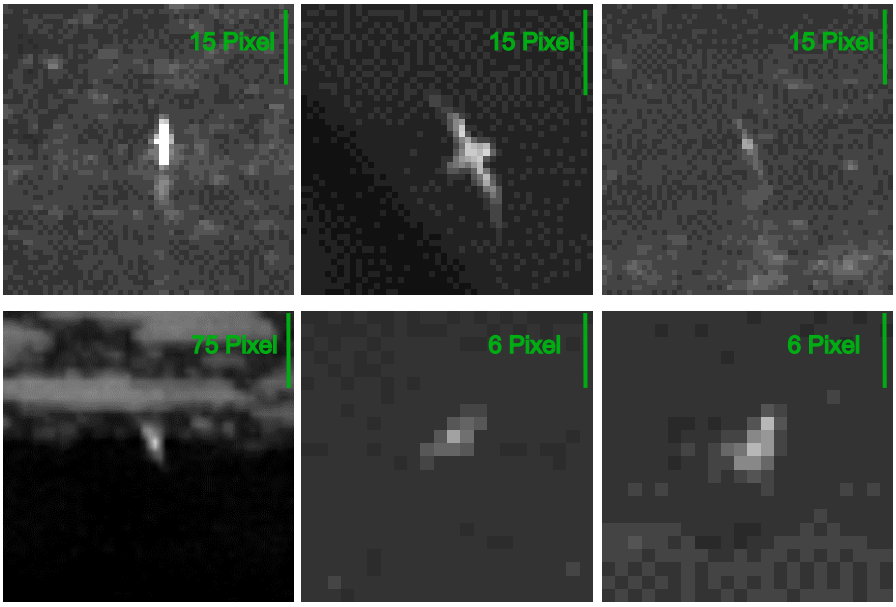


Abbildung 20: Beispiele für verschiedene Formen der Vögel in den Datensätzen. Die Daten sind unterschiedlich stark skaliert, um die Vögel immer in erkennbarer Größe darzustellen. Die erste Reihe zeigt Vögel, deren Umriss deutlich erkennbar ist, während die Vögel in der zweiten Reihe aus sehr wenigen Pixeln bestehen oder unscharf erscheinen.

deutlicher Mehraufwand, der zumindest für einen kleinen Zeitraum zur Überprüfung notwendig ist, welcher in meinen Augen jedoch sehr sinnvoll investiert wäre, da diese Methode damit vielversprechende Ergebnisse liefern könnte.

Literatur

[1] Seabird populations. `http://www.heardisland.aq/research/seabird-populations`. Stand 07.03.2014.

[2] Mallon K. Wormworth J. Bird Species and Climate Change. Technical report, Climate Risk Pty Limited (Australia), 2014.

[3] Kate Madin. Seabirds Face Risks from Climate Change. 50(2):14–15, 2013.

[4] Susan Joy Hassol. Impacts of a warming Arctic. Technical report, Cambridge University, 2004.

[5] Thomas G. Hallam, Aruna Raghavan, Haritha Kolli, Dobromir T. Dimitrov, Paula Federico, Hairong Qi, Gary F. McCracken, Margrit Betke, John K. Westbrook, Kimberly Kennard, and Thomas H. Kunz. Dense and sparse aggregations in complex motion: Video coupled with simulation modeling. *Ecological Complexity*, 7(1):69–75, March 2010.

[6] Nickolay I Hristov, Margrit Betke, and Thomas H Kunz. Applications of thermal infrared imaging for research in aeroecology. *Integrative and comparative biology*, 48(1):50–9, July 2008.

[7] Mark (Department of Coastal Zone Ecology) Desholm. Thermal Animal Detection System. Technical Report 440, Ministry of the Environment Denmark, 2003.

[8] Makoto Yomosa, Tsuyoshi Mizuguchi, and Yoshinori Hayakawa. Spatio-temporal structure of hooded gull flocks. *PloS one*, 8(12):e81754, January 2013.

[9] Daniel P Zitterbart, Lars Kindermann, Elke Burkhardt, and Olaf Boebel. Automatic round-the-clock detection of whales for mitigation from underwater noise impacts. *PloS one*, 8(8):e71217, January 2013.

[10] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series)*. The MIT Press, 2005.

[11] Carina Raupach, Daniel Zitterbart, Claudia Mierke, Claus Metzner, Frank Müller, and Ben Fabry. Stress fluctuations and motion of cytoskeletal-bound markers. *Physical Review E*, 76(1):011918, July 2007.

[12] JA Byers. Correlated random walk equations of animal dispersal resolved by simulation. *Ecology*, 82(6):1680–1690, 2001.

[13] Tin-yu Jonathan Hui. Testing for random walk hypothesis with or without measurement error. 2012.

[14] Edward a Codling, Michael J Plank, and Simon Benhamou. Random walk models in biology. *Journal of the Royal Society, Interface / the Royal Society*, 5(25):813–34, August 2008.

[15] Tashtego. `http://tashtego.org`. Stand 07.03.2014.

[16] Support Vector Machines (SVM). `http://www.mathworks.de/de/help/stats/support-vector-machines-svm.html`. Stand 07.03.2014.

[17] LIBSVM – A Library for Support Vector Machines. `http://www.csie.ntu.edu.tw/\~{}cjlin/libsvm/`. Stand 07.03.2014.

[18] W Cronin, I Fasick, and C Howland. Calculating sighting distances from angular readings during shipboard, aerial, and shore-based marine mammal surveys. 14(3), 1998.

[19] k-Means Clustering. `http://www.mathworks.de/de/help/stats/k-means-clustering.html`. Stand 07.03.2014.

[20] Caglayan Dicle and Mario Sznaier. The Way They Move : Tracking Multiple Targets with Similar Appearance. 2013.

[21] Olaf Boebel. Berichte zur Polar- und Meeresforschung. Technical report, Alfred-Wegener-Institut, 2013.

Eidesstattliche Erklärung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Ort, Datum	Unterschrift