Kapitel 1

Umsetzung

1.1 Ansätze

1.1.1 Iterative Closest Point

Der *Iterative Closest Point* Algorithmus (ICP) ist ein sehr bekannter und verbreiteter Algorithmus für *Rigid Registration*. Er funktioniert wie folgt:

Zuerst werden die Transformationsparameter so initialisiert, dass die beiden Punktwolken keine zu großen Unterschiede aufweisen. Danach werden die Transformationsparameter (Rotation, Translation und Skalierung) auf eine Punktwolke angewandt, sodass nun für jeden Punkt aus der transformierten Punktwolke der dichteste Punkt aus der fixen Punktwolke bestimmt werden kann. Wenn die Zuordnung bestimmt wurde, kann nun über die Summe der Abstandsquadrate zwischen den Punkten ein Fehler zwischen den Punktwolken berechnet werden. Ist dieser Fehler klein genug ist die Registrierung abgeschlossen, ist er zu groß werden die Transformationsparameter anhand der gefundenen Zuordnung neu bestimmt und der Algorithmus wird iterativ wiederholt, bis er konvergiert.
ICP existiert schon recht lange, deswegen gibt es eine Vielzahl von Erweiterungen, wie z.B. EM-ICP, welches ICP durch den EM-Algorithmus erweitert. So existieren auch Varianten die ICP beschleunigen usw.
Der ICP Algorithmus ist auf Grund seiner einfachen Funktionsweise sehr performant und leicht zu verstehen. Außerdem lässt er sich wegen seines modularen Aufbaus sehr gut erweitern und verändern, was ihn gut anpassbar für viele Situationen macht. Er hat jedoch auch einige Nachteile, diese werden im folgenden erläutert. Die Punktwolken müssen initial bereits annähernd zueinander passen, da sonst nicht die optimale Lösung bestimmt werden kann. Außerdem sorgen fehlende Punkte in einer der Punktwolken für Probleme, da diese die Zuordnung verfälschen können und somit den berechneten Fehler.
Der ICP Algorithmus wurde aus folgenden Gründen nicht in dieser Arbeit verwendet:

1. Die Rotation zwischen den beiden Punktwolken kann beliebig sein und ist

vorher nicht bestimmbar

2. Es können mehrere Punkte in den Punktwolken fehlen, da diese recht viele Punkte enthalten können

3. Die Zuordnung von Punkten ist eindeutig (0 oder 1), was das erkennen von Fehlern oder das bestimmen einer Vertrauenswürdigkeit sehr schwer macht

1.1.2 Subgraph Matching

Das Subgraph Matching ist ein weiterer Ansatz, um eine Zuordnung zwischen den Punktwolken wiederherzustellen. Hierfür würden die Punktwolken als zwei vollvermaschte Graphen betrachtet werden. Das Subgraph Matching auch Sub-Graph Isomorphismus Problem genannt basiert auf zwei gegebenen Graphen G und H, wobei gezeigt werden soll das G einen Subgraph isomorph zu H enthält. Die Eigenschaft isomorph bedeutet das eine Bijektion zwischen den Kanten von G und H existiert, sodass zwei benachbarte Kanten in G auch in H benachbarten Kanten entsprechen. Für dieses Problem existieren bereits Algorithmen, welche es für große Graphen lösen können.[Cordello 2004] [Bonnici 2013]
Die Stärke dieses Ansatzes ist die Genauigkeit, da das finden eines Subgraphen eine eindeutige Zuordnung zur Folge hätte, welche je nach Größe des Subgraphs auch als dementsprechend sicher einzustufen wäre. Zusätzlich kann sich so jede beliebige Rotation zwischen den Punktwolken finden lassen, da die Zuordnung nur auf den Abständen zwischen den Punkten basiert.
Jedoch hat dieser Ansatz eine gravierende Schwäche für die zu bearbeitende Problemstellung, welche auch für den Verwurf des Ansatzes sorgte.
Das Graph Matching kann mit Ausreißern nicht arbeiten, da es nicht fehlende Punkte erkennen kann und ein vollvermaschter Graph sich negativ auf die Laufzeit auswirkt. Die Ausreißer bei diesem Problem können in AIS- sowie ARPA-Punktwolken existieren. In der AIS-Punktwolke existieren mehr Punkte als in der ARPA-Punktwolke, weil AIS mehr Reichweite als das Radar hat und AIS geographisch nicht durch verschiedenen Höhen beschränkt ist wie das Radar. Außerdem kann die ARPA-Punktwolke auch Punkte enthalten die in der AIS-Punktwolke fehlen. Dies ist möglich falls ein Schiff kein AIS sendet oder Radarschatten fälschlich für Schiffe gehalten werden. Zusätzlich ist der Ansatz sehr Laufzeit intensiv, da die beiden Graphen zuerst erstellt werden müssen, wofür die Distanz zwischen allen Punkten berechnet werden muss.

1.1.3 Coherent Point Drift

Der in dieser Arbeit verwendete Algorithmus ist der Coherent Point Drift Algorithmus. Dieser wurde aus verschiedenen Punkten ausgewählt, welche im folgenden erläutert werden.

Der CPD kann mit verrauschten Daten arbeiten, da er eine *weiche* Zuordnung (Wert zwischen 0 und 1) berechnet anstatt einer eindeutigen Zuordnung wie der ICP Algorithmus. Durch diese *weiche* Zuordnung ist es auch möglich

eine Aussage über die Vertrauenswürdigkeit eines Zuordnungspaares zu treffen. Dadurch könnte der errechnete Wert wie genau die Zuordnung gepasst hat, als Gewichtung genutzt werden.
Eine weitere wichtige Eigenschaft von CPD ist, dass es möglich ist Ausreißer in den Punktwolken zu erkennen und diese keinem Punkt zuzuordnen. Dies war in ICP und Subgraph Matching noch ein Problem.
Zusätzlich kann der CPD auch eine Rotation der Punktwolke erkennen was essentiell für die Bestimmung des Headings des Schiffes ist (siehe Programmablauf).
Es existiert ein Problem mit diesem Ansatz. Der CPD kann keine beliebige Rotation erkennen, dieses Problem lässt sich jedoch in vielen Point Set Registration Algorithmen finden. Die Lösung dieses Problems wird in 4.2.3 genauer erklärt.
In der folgenden Tabelle 1.1???? ist die Entscheidungsfindung nochmal tabelliert. Zu erkennen sind die einzelnen Stärken der Ansätze und welche Eigenschaften fehlen, um das hier betrachtete Problem effektiv zu lösen.

	ICP	Subgraph Matching	CPD
Rauschen	1	0	1
Rotation	0	1	0.5
Ausreißer	0	0	1
weiches Matching	0	0	1

Tabelle 1.1: Vor- und Nachteile der Ansätze

1.2 Implementierung

1.2.1 Programmablauf

Das entstandene Softwaremodul umfasst verschiedene Abläufe, welche zum besseren Verständnis erläutert werden.

In Abbildung 3.1??? ist der Ablauf der Software anhand eines Aktivitätsdiagramms dargestellt, dieser wird im folgenden genauer erläutert.

C:/Users/User/Desktop/Repositorys/Thesis/Plannung/ProgrammAblauf.pn

Abbildung 1.1: Aktivitätsdiagramm des Softwaremoduls

Zuerst werden die benötigten Daten von der Brücke als NMEA-Nachrichten eingelesen. Dies geschieht durch einen NMEA-Parser von Raytheon Anschütz, welcher in das Modul eingebunden ist.

Im folgenden wird ein Algorithmus ausgewählt, welcher passend zu den verfügbaren Daten den weiteren Ablauf bestimmt. Die folgenden Daten werden hier auf ihre Existenz geprüft:

1. AIS Daten umliegender Schiffe und AtoNs
2. Die Geschwindigkeit des eigenen Schiffes
3. Empfangene AIS-Daten von mindestens drei AtoNs
   Anhand dieser Daten wird ein Algorithmus ausgewählt, welcher in der jeweiligen Situation das beste Ergebnis erzielt.

Der bestimmte Algorithmus gibt nun an, welche der drei Filter auf die Daten angewendet werden müssen, um die für den Algorithmus relevanten Daten zu extrahieren. Diese drei sind: das filtern nach den neuesten AIS und ARPA Daten der Schiffe, das filtern nach den AtoNs im AIS und das filtern nach ARPA Daten, welche keine Geschwindigkeit aufweisen.
Zusätzlich zu diesen drei Filtern werden die AIS-Daten mithilfe von Seekarten Informationen abgeglichen. Bei diesem Abgleich werden fehlende AtoNs im AIS durch AtoNs aus den Seekarten korrigiert. Dies ist möglich, da alle AtoNs in der Seekarte verzeichnet sind.

Nach der Anwendeung der Filter auf die Daten, wird der Zeitverzug der AIS-Daten korrigiert. Dieser Zeitverzug existiert, da AIS-Nachrichten in einer Frequenz zwischen 3s bis 2min gesendet werden können. Die Sendezeit ist abhängig von der Geschwindigkeit und der Kursänderung des Schiffes. Die Radar Antenne, welche für die ARPA-Daten zuständig ist benötigt jedoch für eine volle Umdrehung bzw. für das erfassen aller umliegenden Schiffe nur ca. 2s bis 4s. Dieser zeitliche Versatz zwischen den beiden Datensätzen hat einen großen Einfluss auf die Genauigkeit der Zuordnung des Algorithmus.

Im folgenden startet der spezifische Algorithmus, die einzelnen Komponenten des Algorithmus werden im Laufe der Arbeit genauer behandelt, so wird *AIS zum Koordinaten-Ursprung verschieben* in Unterabschnitt 4.2.2 behandelt und wie das initiale Heading den Algorithmus beeinflusst wird in 4.23 erklärt.
Das Ergebnis des Algorithmus ist die Zuordnung zwischen den Punkten und der Rotationsunterschied, welcher benötigt wurde um die Punktmengen ineinander zu überführen. Die Zuordnung wird für den in 4.3 erklärten Position-Fix Algorithmus benötigt und die Rotation, welche benötigt wird um beide Punktwolken einander zuzuordnen entspricht dem Heading des Eigenschiffes. Die Bestimmung des Headings auf diese Weise ist möglich, da die ARPA-Daten von der relativen Ausrichtung des Schiffes abhängen, die AIS-Daten jedoch nach Norden ausgerichtet sind. Somit entspricht eine Drehung der ARPA-Daten, bis sie mit den AIS-Daten übereinstimmen, dem Winkel zwischen der Ausrichtung des Schiffe und der Himmelsrichtung Norden. Dies ist der Winkel, in welchem das Heading eines Schiffes angegeben wird.

Mit den Ergebnissen des Coherent Point Drift Algorithmus kann nun der Position-Fix die Eigenposition bestimmen (4.3) und diese als Positionsangabe über eine NMEA-Nachricht an die Brücke senden. Zusätzlich wird noch eine Visualisierung des Ergebnisses über die von Herrn André Becker entwickelte Netzwerkschnittstelle an die ECDIS gesendet.

1.2.2 Ungenauigkeiten durch den Erdellipsoid

Der Coherent Point Drift Algorithmus basiert auf Koordinaten in kartesischer Form. In dieser Arbeit wurde dieser, auf geodätische Koordinaten angewendet. Wodurch einige Genauigkeitsprobleme entstanden, die im weiteren erläutert werden und eine Lösung präsentiert wird.

Das erste Problem beruht auf der Translation, welche der Algorithmus anwendet um die Distanz zwischen zwei Punktwolken zu minimieren.
Die Translation basiert auf dem Unterschied der Punktwolken in x und y Achse. Dies ist zwar korrekt in einem kartesischen Koordinatensystem, wo der Abstand zwischen zwei Punkten über die Euklidische Norm bestimmt werden kann, jedoch nicht wenn mit geodätische Koordinaten gearbeitet wird. Der Grund hierfür ist, dass der Abstand zwischen den Längengraden zwischen Nord- bzw. Südpol und dem Äquator sich je nach Position ändern. Als Folge hiervon ist es zwar möglich die Punktwolken zueinander zu bewegen, jedoch ist die Translation, welche der Algorithmus bestimmt, mit einem Fehler versehen. Diese Abweichungen die durch dieses Problem entstanden sind verhalten sich proportional zum Abstand der beiden Punktwolken. Der Abstand zwischen den Punktwolken kann beliebig groß sein, da die ARPA-Punktwolke ohne eine absolute Positionsangabe in Polarkoordinaten vorliegt und somit eine initial Position benötigt wird um aus dieser geodätische Koordinaten zu erstellen.

In der folgenden Tabelle wird gezeigt wie der Fehler sich mit steigender Translation verhält. Hierfür wurden die Distanzen auf dem Erdellipsoid zwischen zwei Punkten beobachtet, welche mittels einer Translation immer weiter von ihrer originalen Position verschoben wurden.

Translation (lat/lon)	(0/0)	(10/10)	(20/20)	(30/30)	(40/40)	(50/50)
Distanz (m)	24758,5	24683,9	24478,9	24164,8	23775,7	23355,8
Abweichung (m)	-	74,5581	279,609	593,68	982,744	1402,62

Tabelle 1.2: Fehler durch Translation auf dem Erdellipsoid

Es ist deutlich zu erkennen, dass desto größter die Strecke wurde über welche die Punkte mittels der Translation verschoben wurden, desto größer wurde auch der Fehler in der Distanz zwischen diesen Punkten. Somit gilt es die Translation so klein wie möglich zu halten, damit dieser Fehler keine signifikanten Auswirkungen auf das Ergebnis hat.

Das zweite Problem entstand durch die Rotation, welche Teil des Algorithmus ist. Bei einer Rotation werden alle Punkte einer Punktwolke um den Ursprung des Koordinatensystems gedreht und erhalten somit eine neue Position. Durch das Rotieren einer kartesischen Punktwolke um den Ursprung ändern sich die Positionen der Punkte, jedoch nicht die Abstände zwischen den Punkten. Wenn man geodätische Koordinaten rotiert ändern sich ebenso die Positionen der Punkte, jedoch ändert sich hier auch der Abstand zwischen den Punkten auf Grund der Unterschiedlichen Abstände zwischen Längengraden. Die Abweichungen die durch dieses Problem entstanden sind, steigen mit dem Abstand

zwischen den Punktwolken.

Die Lösung für das erste Problem (Translation), ist dass minimieren der initialen Abstände zwischen den Punktwolken. Die Translation die nun benötigt wird für das überführen der einen Punktwolke in die andere resultiert in einer kleinen Abweichung, welche in Tests maximal $7m$ entsprach.
Die Lösung für das zweite Problem (Rotation) ist, dass verschieben des Schwerpunktes der zu rotierenden Punktwolke in den Ursprung des Koordinatensystems. Eine Rotation dieser Punktwolke bewirkt nun das der Schwerpunkt seine Position nicht verändert, wodurch die Punkte der Punktwolke eine minimale Positionsänderung durch eine Rotation erfahren.

Somit ist die Lösung für diese Probleme die Translation beider Punktwolken auf den Ursprung des Koordinatensystems, um so die Ungenauigkeiten so gering wie möglich zu halten. Dies hat auch den Effekt das Fehler, welche durch das verwenden der euklidischen Norm auf dem Erdellipsoid entstehen, geringer ausfallen, weil Längen- und Breitengrad im Ursprung des Koordinatensystems fast gleich lang sind.
Für die ARPA - Punktwolke ist dies trivial, da aufgrund der Radar Daten diese Punkte in Polarkoordinaten angegeben sind.
Die AIS - Punktwolke ist jedoch an ihre absolute Position gebunden, deswegen ist eine andere Methodik notwendig, um die Punktwolke, ohne Änderung der Abstände zwischen den Punkten, verschieben zu können. Die Methodik sieht das bestimmen des Schwerpunktes vor, zu welchem die Abstände und der Richtungswinkel von jedem Punkt bestimmt wird. Nun kann dieser Punkt in verschoben werden und mithilfe der Abstände und Richtungswinkel, lässt sich die Punktwolke korrekt wiederherstellen. Der zu bestimmende Punkt ist der Schwerpunkt der AIS-Punktwolke, welcher sich über das arithmetische Mittel bestimmen lässt, solange es sich um eine nicht gewichtete Punktwolke handelt. Nun wird die Distanz und der Winkel zwischen jedem Punkt der Punktwolke und dem Schwerpunkt der Wolke bestimmt und gespeichert. Für die Berechnung der Distanz und der Winkel zwischen den Punkten wird die Vincenty Formel verwendet, damit keine Fehler durch die Erdkrümmung entstehen. Nun liegt die AIS-Punktwolke theoretisch in Polarkoordinaten vor. Das macht es möglich sie ebenso wie die ARPA - Punktwolke an den Ursprung zu platzieren.

Nun sind optimale Bedingungen geschaffen um eine Zuordnung zwischen den Punktwolken durch den Coherent Point Drift Algorithmus zu bestimmen.

1.2.3 Rotationsschwäche von Coherent Point Drift

Das erkennen von Rotation ist eine essentielle Eigenschaft, welche der Algorithmus benötigt um die Punktwolken einander zuzuordnen und somit auch das Heading des Eigenschiffes zu bestimmen. Die Rotation kann jeden beliebigem gültigen Wert entsprechen, da die Drehung der ARPA Punktwolke abhängig vom Heading ist und die Drehung der AIS Punktwolke immer nach Norden ausgerichtet ist. Somit ist jeder Rotationswinkel eine mögliche Lösung und das Problem kann nicht vorher behandelt werden.

Bei Experimenten mit dem Coherent Point Drift ist jedoch aufgefallen, das

Rotationswinkel (°)	Gefundenes Minima (°)
0 - 70	0
80 - 140	111
150 - 160	172
170 - 220	230
230 - 300	282
310 - 0	0

Tabelle 1.3: Bestimmte Rotation durch CPD an perfekten Daten

dieser Rotation nur bis ca. 60° erkennen kann. Bei Punktwolken, welche mehr als 60° Rotationsunterschied besitzen, findet der Algorithmus möglicherweise nur ein lokales Minima und gibt dies als Lösung zurück. Die Ergebnisse des Experiments sind in Abbildung ???? visualisiert. Diese Schwäche im erkennen der Rotation vom Coherent Point Drift Algorithmus wurde bereits in anderen Arbeiten festgestellt. HIER PAPER VERLINKEN.
In Tabelle 1.3??? wurden die Ergebnisse des CPD Algorithmus für verschiedene Rotationen bestimmt. Diese Messung wurde mit perfekten Daten ausgeführt, dass bedeutet das alle Punkte zugeordnet werden können, die Punkte perfekt aufeinander passen und das lediglich der Rotationswinkel der Punktwolken sie voneinander unterscheidet. Die Punkte der Punktwolke wurden zufällig anhand einer Normalverteilung erstellt und besteht aus 20 Punkten in geodätischen Koordinaten. Alle Angaben sind in Grad gegeben.

Es ist deutlich zu erkennen, dass der CPD Algorithmus ab einem bestimmten Rotationswinkel, die korrekte Rotation nicht mehr findet und in einem lokalen Minimum terminiert. In den Testdaten von Tabelle 1.2??? werden fünf verschiedene Minima gefunden, je nach Rotation der Punktwolke.

Für dieses Problem wurde eine Lösung entwickelt, um die korrekte Rotation, sowie die korrekte Zuordnung zu bestimmen.

Hierfür wird der Algorithmus mehrfach ausgeführt, wobei nach jeder Ausführung die ARPA-Punktwolke rotiert wird und die Ergebnisse gespeichert werden. Für die kontinuierliche Rotation wurden 45° gewählt. Es wurden 45° gewählt, weil 45 der nächste ganzzahlige Teiler von 360 ist und welcher kleiner als 52 ist. Die 52° ist der kleinste Raum für das Erkennen der Rotation durch den CPD unter real nahen Daten. Die Rotation der ARPA-Punktwolke ist sehr simple, da sie bereits in Polarkoordinaten vorliegt. Nachdem der Algorithmus 8 mal ausgeführt wurde ($\frac{360}{45} = 8$) werden die Ergebnisse verglichen. Hierbei wird das Ergebnis mit dem geringsten Fehler zwischen den Punkten ausgewählt, da so das globale Optimum gefunden wird.
Diese Lösung hat natürlich einen großen Nachteil, welcher die Performanz des Algorithmus ist. Der Algorithmus muss acht mal gestartet werden um ein korrektes Ergebnis zu produzieren. Die achtfache Ausführung ist jedoch nur beim initialen Ausführen des Softwaremoduls nötig, weil nach dem ersten Durchlauf des Softwaremoduls ein Heading an die Brücke gesendet wird, welches beim nächsten Aufrufen des Algorithmus mit gesendet wird und somit ein initiales

Heading existiert. Wenn ein Heading mit gesendet wird muss der Algorithmus nur einmal ausgeführt werden, weil die ARPA-Punktwolke nur um den Wert des negativen Headings gedreht werden muss und dadurch die Punktwolken eine Ähnliche Rotation aufweisen sollten.

1.3 Position-Fix

1.3.1 Bestimmung der Eigenposition

Mit dem Ergebnis des Coherent Point Drift Algorithmus können ARPA - Punkte auf ihre zugehörigen AIS - Punkte zugeordnet werden. Dadurch ist nun bekannt wo dieser Punkt absolut auf der Karte liegt (AIS-Daten) und wie er relativ zu der Schiffseigenposition liegt (ARPA-Daten). Durch diese beiden Daten ist es möglich die Schiffseigenposition für jedes Paar zugeordneter Punkte zu berechnen.
Damit diese Rückrechnung auf die Eigenposition möglich ist, muss zuerst das relative Bearing aus den ARPA - Daten in *true* Bearing umgerechnet werden. Als *True* Bearing wird das Bearing bezeichnet, welches relativ zur Nord-Richtung angegeben wird. Diese Umrechnung ist möglich, da der CPD-Algorithmus auch das Heading des Schiffes bestimmt. Das Heading des Schiffes ist relativ zur Nord-Richtung und bietet somit die Möglichkeit über folgende Formel eine relative Bearing Angabe in *True* Bearing umzurechnen:

b_{true} : *True* Bearing
b_{rel} : Relatives Bearing
h : Heading des Eigenschiffes

$$b_{true} \quad = \quad (b_{rel} - h) \quad mod\, 360$$

Nun kann von jedem Datenpaar eine individuelle Eigenposition bestimmt werden. Dies basiert auf der Lösung der 1. geodätischen Hauptaufgabe, diese ist das berechnen eines Punktes mit einem Startpunkt, einer Richtung und einer Distanz. Der Startpunkt sind die AIS Koordinaten, die Richtung ist das invertierte *true* Bearing und die Distanz kann aus den ARPA - Daten entnommen werden. Die Lösung der 1. geodätischen Hauptaufgabe lässt sich mithilfe der Vincenty Formel bestimmen.
Nachdem dies für jedes Datenpaar wiederholt wurde, existiert eine Punktwolke, welche die individuellen Eigenpositionen enthält. Zur Bestimmung der Eigenposition wird das arithmetische Mittel über diese Punktwolke gebildet. Falls die einzelnen Punkte vorher eine Gewichtung besitzen, zur Indizierung einer höheren Vertrauenswürdigkeit, kann hier auch das gewichtete arithmetische Mittel angewendet werden.
Die Auswirkung für die Verwendung eines gewichteten Arithmetischen Mittels anstatt des normalen arithmetischen Mittels ist in Abbildung 1.2??? zu erkennen. In der Abbildung haben die Punkte 6 - 10 eine höhere Gewichtung als

die Punkte 1 - 4, was dafür sorgt, dass diese beim gewichteten Arithmetischen Mittel stärker berücksichtigt werden.

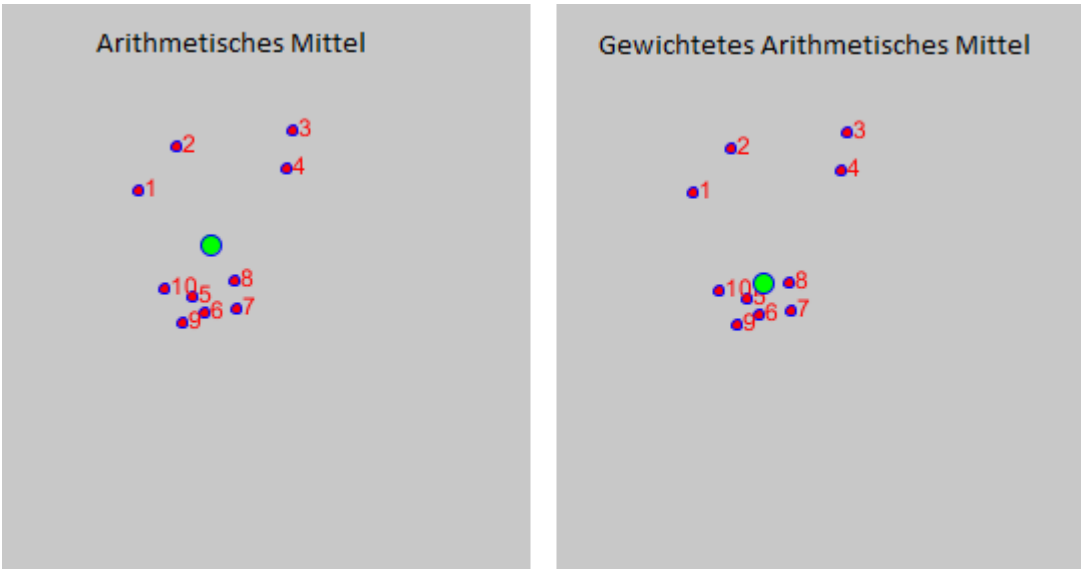

Abbildung 1.2: Arithmetisches Mittel der individuell zurückgerechneten Eigenpositionen

Die Abbildung zeigt wie stark die Genauigkeit der Positionsbestimmung erhöht werden kann, falls es möglich ist eine Aussage über die Genauigkeit oder die Vertrauenswürdigkeit der einzelnen Punkte treffen zu können. Das wäre möglich über das Unterscheiden von Schiffen und AtoNs oder einer manuellen Eingabe.

1.3.2 Bestimmung der Sicherheit der Eigenposition und Eliminierung von Ausreißern

Die Bestimmung der Eigenposition erfolgt über das arithmetische Mittel, dieses ist jedoch anfällig für Ausreißer in der Punktwolke. Zusätzlich ist es notwendig die bestimmte Eigenposition mit einer Sicherheit bzw. eines Sicherheitsbereichs angeben zu können.

Durch die Anfälligkeit des arithmetischen Mittels auf Ausreißer, ist es notwendig diese zu bestimmen und in gravierenden Fällen zu eliminieren. Dies kann jedoch nur angewendet werden, wenn genügend Punkte in der Punktwolke existieren, da sonst Ausreißer nicht eindeutig klassifiziert werden können.
Zur Erkennung der Ausreißer wird die Mahalanobis Distanz verwendet. Die Mahalanobis Distanz ist ein Abstandsmaß, welche angibt wie weit ein Punkt von der Verteilung einer Punktwolke abweicht. Das Ergebnis der Mahalanobis Distanz wird in Standardnormalabweichungen angegeben. Die Mahalanobis Distanz ist wie folgt definiert:

$\vec{X_i} = \begin{pmatrix} lat_i \\ lon_i \end{pmatrix}$: eine Zufallsvariable, welche die zurückgerechnete Position eines Datenpaars beschreibt. (*lat* ist der Breitengrad und *lon* der Längengrad des Punktes)

$\vec{\mu} = \begin{pmatrix} \mu_{lat} \\ \mu_{lon} \end{pmatrix}$: Das arithmetische Mittel der Punktwolke

Σ : Die Kovarianzmatrix der Punktwolke für welche gilt $det(\Sigma) \neq 0$

Mahalanobis-Distanz:

$$d(\vec{X}, \vec{\mu}) = \sqrt{(\vec{X} - \vec{\mu})^T \cdot \Sigma^{-1} \cdot (\vec{X} - \vec{\mu})}$$

zur Berechnung der Mahalanobis Distanz wird nun das arithmetische Mittel und die Kovarianzmatrix bestimmt, jedoch ohne einbeziehen des Punktes zu welchem die Mahalanobis Distanz bestimmt werden soll, da sonst das Ergebnis verfälscht wird. Nachdem dies für jeden individuelle Eigenposition berechnet wurde ist es möglich Punkte zu entfernen, welche außerhalb einer bestimmten Prozentzahl der Verteilung liegen. Für diese Prozentzahl wurde $99,865\%$ gewählt, das entspricht einem Z-Wert der Standardnormalverteilung von 3. Somit werden Punkte entfernt die Außerhalb von 99.865% der Verteilung liegen.
Nachdem die Ausreißer entfernt wurden ist es wichtig die Genauigkeit des Ergebnisses angeben zu können. Hierfür wurde sich entschieden einen Sicherheitsbereich zu bestimmen, durch welchen eine prozentuale Sicherheit gegeben werden kann, wie wahrscheinlich es ist sich in diesem Bereich zu befinden. Dieser Sicherheitsbereich wird mithilfe der Mahalanobis Distanz bestimmt. Man setzt die Mahalanobis Distanz auf einen festen Wert, wie z.B. $2,33$ was ca. 99% Sicherheit entspricht. Nun kann durch einsetzen des arithmetischen Mittels und der Kovarianzmatrix eine Menge an Punkten bestimmt werden, welche diese Mahalanobis Distanz haben. Die Form der Punkte entspricht einer Ellipse, welche der Sicherheitsbereich ist. Eine Visualisierung dieses Ablaufs ist in Abbildung???? zu erkennen. Die roten Punkte sind zurückgerechnete Eigenpositionen, wobei 10 ein Ausreißer ist, welcher das Resultat einer falschen Zuordnung sein könnte. Die blauen Ellipsen indizieren die Standardabweichungen vom Faktor 0,5 bis 2,0 von innen nach außen.

Abbildung 1.4: Visualisierung zur Anzeige des Ergebnisses auf der Seekarte

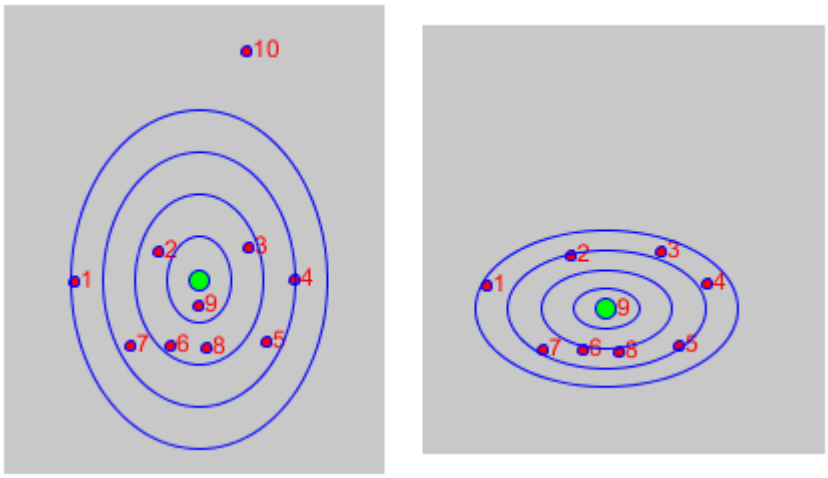

Abbildung 1.3: Eliminierung von Ausreißern und Darstellung des Sicherheitsbereichs

Es ist klar zu erkennen wie groß die Auswirkung eines Ausreißers auf die Eigenposition und den Sicherheitsbereich ist.

1.3.3 Erstellung des Ergebnisses

Nach dem Position-Fix Algorithmus wurden alle benötigten Daten berechnet um die Eigenposition, das Heading und eine geeignete Visualisierung darzustellen. Die Eigenposition und das Heading werden als NMEA-Nachricht an die Brücke gesendet, da ihre Daten so intern wie die Daten eines Sensors behandelt werden und dadurch eine Integritätsprüfung im Vergleich zu den anderen Sensoren möglich ist. Die Visualisierung und zusätzliche Daten, wie der Sicherheitsbereich, werden über die Netzwerkschnittstelle von Herrn André Becker an die ECDIS übermittelt. Die Visualisierung ist eine SVG-Grafik, welche beispielhaft in Abbildung ??? abgebildet ist.