Freie Universität Berlin
Fachbereich Mathematik und Informatik

Masterarbeit der Informatik

Probabilistisches Tracking von Bienenpfaden

von *Jakob Mischek*

Betreuer:
Prof. Dr. Tim Landgraf

9. September 2016

Abstract

Das BeesBook-Projekt am Biorobotics Lab der FU Berlin untersucht mit einem neuartigen Versuchsaufbau das Verhalten von Bienen. Dazu werden mit Binärcodes markierte Bienen im Bienenstock gefilmt. Mit Hilfe von Bilderkennungs-Software werden die Position der Codes, und somit der Bienen, auf dem Videomaterial erkannt. Da jeder Code einzigartig ist können die Bewegungen einer Biene über ihren gesamten Lebensverlauf zugeordnet und ausgewertet werden.

Durch die Rahmenbedingungen des Versuchsaufbaus und der verwendeten Technik ist die Genauigkeit mit der Codes gefunden und ausgelesen werden können beschränkt. Vor der Auswertung des Bienenverhaltens ist daher ein Tracking-Zwischenschritt nötig der die erkannten Codes zu Pfaden von Bienenbewegungen verbindet und Ungenauigkeiten herausfiltert. In dieser Arbeit wird ein Algorithmus für das Tracking-Problem auf Basis eines probabilistischen Filterings unter Verwendung von Maschinellem Lernen mit Gradient Boosting vorgestellt.

Die Arbeit beschreibt die Anforderungen an den Algorithmus die sich aus der Datengrundlage ergeben, die Entwicklung des Algorithmus aus der Lösung verschiedener Teilprobleme und die Evaluation der Tracking-Ergebnisse.

Eidesstattliche Erklärung

Ich versichere, die Masterarbeit selbstständig und lediglich unter Benutzung der angegebenen Quellen und Hilfsmittel verfasst zu haben. Ich erkläre weiterhin, dass die vorliegende Arbeit noch nicht im Rahmen eines anderen Prüfungsverfahrens eingereicht wurde.

Berlin, den 9. September 2016 _____

Jakob Mischek

Inhaltsverzeichnis

1 Einleitung

1.1 BeesBook-Projekt

Das BeesBook-Projekt am Biorobotics Lab der FU Berlin untersucht mit einem neuartigen Versuchsaufbau das Verhalten von Bienen[6]. Durch aufgezeichnetes Videomaterial aus dem Bienenstock soll eine ganzheitliche und umfassende Analyse der Organisation und Interaktion des Bienenvolkes möglich werden. Die entscheidende Innovation ist dabei ein Tracking-System, das die eindeutige Identifizierung jeder einzelnen Biene ermöglicht.

Motiviert ist dieses Vorgehen durch die Möglichkeit mit den erhobenen Daten erstmals Fragestellungen zu beantworten, für die bisher eine so umfassende Datengrundlage fehlte.

Es ist zum einen möglich jede einzelne Biene aus den Daten zu selektieren und über ihren gesamten Lebenslauf zu beobachten. Idealerweise lassen sich dabei die einzelnen Arbeitsaufgaben, die eine Biene im Laufe ihres Lebens übernimmt, erkennen und analysieren.

Zum anderen ist sämtliche Interaktion der Bienen miteinander aufgezeichnet. Insbesondere der Bienentanz, also das gegenseitige Mitteilen von Futterquellen, ist ein lohnendes Forschungsgebiet. Während die Bedeutung der Tanzbewegungen weitestgehend entschlüsselt ist, ergeben sich nun Fragestellungen die nur durch die neue Datenmenge beantwortbar sind, unter anderem:

- Wie ist die Verknüpfungsstruktur im sozialen Netzwerk des Bienenvolkes?
- Gibt es innerhalb des Bienenstockes Gruppen von Bienen die bevorzugt miteinander kommunizieren?
- Wie verändert sich das Netzwerk über die Zeit?

Der Versuchsaufbau ist auf andere Versuchsobjekte übertragbar, die Analysemethoden auf andere Daten und Netzwerke anwendbar. Durch die angestrebte und teilweise bereits erfolgte Veröffentlichung als Open-Source-Software sollen ähnliche Projekte in anderen Forschungsgebieten angeregt werden.

1.2 Versuchsaufbau

Der Bienenstock im Versuchsaufbau besteht aus einer Wabe. Diese besitzt Wabenzellen von beiden Seiten (Abbildung 1). Jede Seite wird von zwei Kameras gefilmt, wobei jede Kamera jeweils eine Hälfte der Seite abbildet. Es ergeben sich also vier unterschiedliche Kamerabilder. Die Beleuchtung erfolgt mit von Bienen nicht wahrnehmbarem Infrarotlicht. Entsprechend sind auch die Kameras modifiziert dieses Spektrum aufzuzeichnen[6, S. 4].

Das Tracking der Bienen erfolgt anhand des Videomaterials. Um die Bienen auf dem Videobild detektieren zu können, ist jede Biene auf dem Rücken mit einem Tag beklebt auf dem ein Binärcode abgebildet ist (Abbildung 2).

Abbildung 1: Die Wabe im Versuchsaufbau.

1.3 Design der Tags

Die Tags mit den Binärcodes sind einzigartig und erlauben die Identifizierung jeder einzelnen Biene. Das Design der Tags trägt dabei den Gegebenheiten des Experimentes Rechnung.

Die Tags sind rund und gewölbt, um die Bienen nicht in ihrer Bewegungsfreiheit einzuschränken und das Eintauchen in eine Wabenzelle zu ermöglichen. Die Ausrichtung des Tags wird durch einen schwarzen und weißen Halbkreis in der Mitte des Codes bestimmt. Diese sind auch dann noch auf der gewölbten Oberfläche des Tags sichtbar, wenn die Biene ihren Körper etwas neigt. Da beim Bekleben der Bienen darauf geachtet wurde, den weißen Halbkreis stets in Richtung des Bienenkopfes auszurichten, lässt sich anhand des Tags auch die Ausrichtung der Biene bestimmen. Bedingt durch die limitierte Präzision beim manuellen Bekleben der Bienen, muss aber von einer gewissen Varianz für diesen Wert ausgegangen werden.

Um den zentralen Bereich ist ein Kreis mit Flächen für 12 Bits angeordnet (Abbildung 2). 12 Bits ermöglichen eine ausreichend große Anzahl verschiedener Tags ($2^{12} = 4096$). Mehr Bitstellen wären nützlich, denn sie könnten als Paritätsbits genutzt werden, um einen gelesen Code kontrollieren und korrigieren zu können. Verschiedene Faktoren schränken die mögliche Anzahl an Bits auf dem Tag aber ein:

- Die maximale Größe eines Tags der die Biene nicht behindert.
- Die Auflösung der verwendeten Kameras.
- Die nötige Belichtungszeit einer Aufnahme und die Geschwindigkeit von Bienenbewegungen, insbesondere der tanzenden Bienen.

Bedingt durch diese Faktoren ist es schon im aktuellen Setup mit 12 Bits nichts hundertprozentig möglich jede Biene korrekt zu identifizieren. Eine Filterung und Fehlerkorrektur auf Softwareebene ist also unumgänglich.

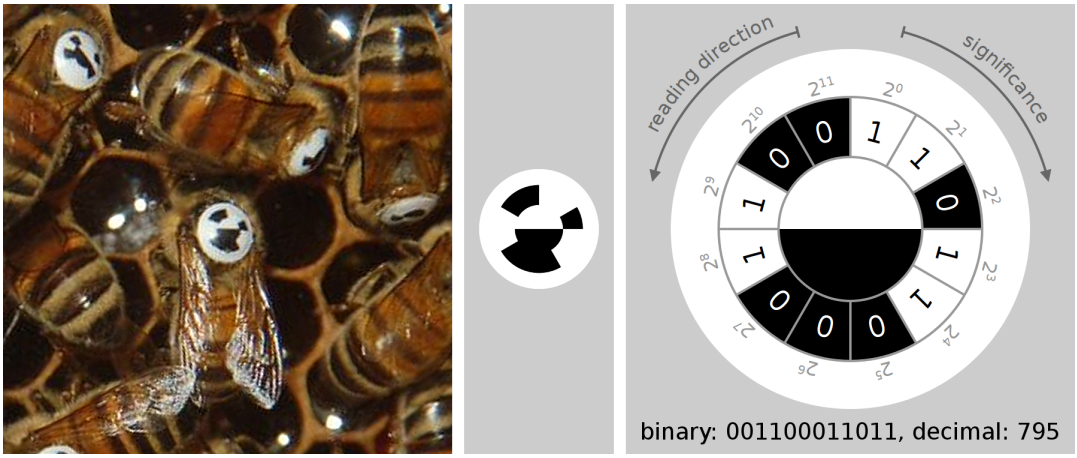

Abbildung 2: Eine Biene mit Tag, der Tag und seine Interpretation.

1.4 Datenprozessierungs-Pipeline

Die Verarbeitungs-Pipeline der BeesBook-Daten sieht im Wesentlichen folgendermaßen aus:

- Lokalisierung von Code-Tags auf einzelnen Videostandbildern (Position einer Detektion)[6, S. 4]
- Binärcode auslesen (decodieren) (ID einer Detektion)[5]
- Tracking über die Zeit, d.h. Detektionen zu Pfaden verbinden
- Pfade nach Art des Verhaltens klassifizieren
- Interaktionen zwischen Bienen erkennen
- Netzwerk der Bienen-Interaktion analysieren

Diese Arbeit befasst sich mit dem Tracking-Problem und stellt als Lösung einen Algorithmus auf Basis eines probabilistischen Filterings unter Verwendung von Maschinellem Lernen mit Gradient Boosting vor.

1.5 Struktur der Arbeit

Nachdem das übergeordnete Projekt vorgestellt und die Arbeit in das Projekt-Setup eingeordnet ist, wird im nächsten Kapitel untersucht welche Veröffentlichungen sich bereits mit ähnlichen Problemen beschäftigt haben (Stand der Forschung). Im Kapitel Implementierung wird der Algorithmus mit seinen Teilkomponenten beschrieben. Anschließend erfolgt eine Evaluierung wie gut der Algorithmus in der Lage ist das Problem zu lösen. Es folgen eine Diskussion der Ergebnisse sowie ein Ausblick auf mögliche Weiterentwicklungen. Im Anhang finden sich Informationen zur Herkunft der Vergleichsdaten, die in vielen Bereichen dieser Arbeit eine wichtige Rolle spielen.

2 Stand der Forschung

Nachdem im vorherigen Kapitel das Tracking-Problem im Rahmen des BeesBook-Projektes vorgestellt wurde, soll nun gezeigt werden, welche Lösungsansätze bereits in ähnlichen Projekten entwickelt wurden.

Ein sehr ähnliches Projekt beschrieben D. P. Mersch, A. Crespi und L. Keller in einem Artikel in Sciene[4]. In diesem Fall waren die Versuchstiere Ameisen. Sie waren ebenfalls mit Binärcodes auf dem Rücken markiert und das Tracking basierte genauso auf visueller Detektion der Tags.

Zur Generierung und Detektion der Tags wurde das ARTag-System[3] verwendet. In diesem System sind nur rechteckige Tags vorgesehen. Da, wie im Abschnitt über das Design der Tags beschrieben, für das BeesBook-Projekt runde Tags benötigt wurden, konnte diese Tool-Landschaft nicht übernommen werden.

Des Weiteren war es im Setup des Ameisen-Experimentes möglich 36 Bits auf dem Tag unterzubringen, von denen 26 zur Fehlerkorrektur verwendet wurden:

> These markers consisted of a square outline with a 36-bit digital word encoded in the interior. The digital word can generate up to 2000 unique ID numbers, which are protected from false detection by 26 bits of error correction code. [4, Suppl. S. 2]

Dadurch konnte eine enorme Genauigkeit beim Dekodieren der Codes erreicht werden:

> [...] the probability of false positives and inter-marker confusions was very low ($0.8 \cdot 10^{-7}$). [4, Suppl. S. 4]

Eine solche Zuverlässigkeit der Daten ist im BeesBook-Projekt nicht gegeben, die Nachbearbeitung der Daten also wesentlich schwieriger.

Auch was Anfang und Ende von Pfaden anging waren im Projekt mit den Ameisen wesentlich einfachere Annahmen möglich, denn die Ameisen konnten die gefilmte Arena nur über einen Tunnel an einer Seite verlassen. Im BeesBook-Projekt hingegen können die Bienen die Wabe zu jeder Seite verlassen oder durch Löcher in der Wabe auf die andere Seite wechseln. Oder sich sogar an jeder beliebigen Position auf der Wabe umdrehen und auf der Scheibe zwischen Wabe und Kamera weiterlaufen, wodurch der Tag nicht mehr zu sehen ist.

Für das BeesBook-Projekt bedeutet dies, dass ein eigenes, angepasstes Setup entwickelt werden muss, wobei insbesondere die Software zur Auswertung eine optimal zugeschnittene Eigenentwicklung erfordert.

3 Implementierung

In diesem Kapitel wird der zur Problemlösung genutzte Algorithmus mit seinen Teilkomponenten beschrieben. Wie im Abschnitt über die Datenprozessierungs-Pipeline erklärt, liegt als Eingabe für das zu behandelnde Problem eine Menge an Detektionsdaten über fortlaufende Zeitschritte vor. Eine Detektion ist in diesem Sinn die Position einer auf dem Videobild lokalisierten Biene und ihre aus dem Binärcode gelesene ID. Zusammen mit weiteren Informationen, die der Decoder ausgibt, ergibt sich folgende Datengrundlage je Detektion:

- Zeitpunkt
- Position in Bildkoordinaten
- Für jedes Bit eine Wahrscheinlichkeit dafür, dass dieses Bit weiß ist
- Daraus sich ergebend eine wahrscheinliche ID der Biene, berechnet durch Runden der Wahrscheinlichkeiten für jede Bitstelle
- Drei Winkel für die Drehung des Tags im Raum
- Wie deutlich der Localizer ein Tag an dieser Position erkennen konnte (Localizer-Saliency)

Die Aufgabe im Tracking-Schritt ist es nun die Detektionen über die Zeitrichtung zu Pfaden zu verbinden und die tatsächliche ID der Biene für jeden Pfad zu bestimmen.

Folgende Schwierigkeiten sind dabei zu beachten:

- Nicht jeder Tag kann richtig dekodiert werden. Dies ergibt sich aus den im Abschnitt Versuchsaufbau beschriebenen Rahmenbedingungen des Projektes.
- Je nachdem wie gut der Localizer arbeitet, wird er einen Teil der Bienen übersehen oder fälschlicherweise Bienen anzeigen, wo sich keine befinden. Die fehlenden Detektionen ergeben Lücken in den Pfaden; die False-Positives sollten beim Tracking möglichst als solche herausgefiltert werden.
- Unabhängig vom Localizer sind Bienen öfters nicht sichtbar. Vor allem wenn sie ihren Oberkörper in eine Wabe neigen um dort arbeiten zu verrichten. Oder wenn sie durch andere Bienen verdeckt werden, insbesondere von Bienen die an der Scheibe zwischen Kamera und Wabe laufen und somit der Kamera ihre Unterseite zeigen.
- Weiterhin ist zu beachten, dass Bienen am Rand der Wabe auftauchen und verschwinden können.

3.1 Idee des Algorithmus

Im Prinzip handelt es sich bei der Problemstellung um eine Wegsuche in einem Graphen mit gewichteten Kanten. Jede Detektion bildet einen Knoten von dem aus Kanten zu den Detektionen

des nächsten Zeitschrittes ausgehen. Mit welcher Wahrscheinlichkeit eine solche Verbindung zu einem realen Bienenpfad gehört, ist das Gewicht der Kante und muss aus den Parametern die der Decoder ausgibt bestimmt werden.

Wollte man nun einen herkömmlichen Algorithmus zur Wegsuche auf diesen Graphen anwenden, steht man vor einigen Problemen:

- Start- und Endknoten der Pfade sind nicht bekannt. Nicht mal der Zeitpunkt, an dem ein Pfad beginnen müsste, ist bekannt.
- Auch die Lücken in Pfaden, also das Fehlen einer Detektion, muss im Graph abgebildet sein.
- Des Weiteren ist der Aufnahmezeitraum des Experimentes so groß, dass nicht der komplette Graph auf einmal aufgebaut werden kann. Wird das Problem in Zeitausschnitten behandelt, ist ein zusätzliches Verfahren nötig um die Ergebnisse an der Grenze zusammenzufügen. Durch die Möglichkeit Lücken an der Grenze zu haben, würde des dabei nicht ausreichen nur Detektion aus den direkt angrenzenden Zeitschritten zu verbinden.
- Wie lässt sich sicherstellen, dass jede Detektion nur in einem Pfad Verwendung findet? Alternativ könnte auch die Mehrfachverwendung erlaubt werden. Dann ergibt sich aber das Problem, wie sich unterscheiden lässt, ob zwei Pfade unterschiedliche Bienen darstellen oder nur Varianten derselben Bienentrajektorie sind.

Aus diesen Gründen wurde ein Algorithmus entworfen der kontinuierlich entlang der Zeit den Graphen abarbeitet, und dabei alle Bienen gleichzeitig trackt. Wie später gezeigt werden soll, kann dieser Ansatz mit Wegfindungs-Algorithmen erweitert werden (Abschnitt 6.3). Dazu wird in jeder Iteration der Graph über ein begrenztes Zeitfenster aufgebaut und zur Entscheidungsfindung herangezogen.

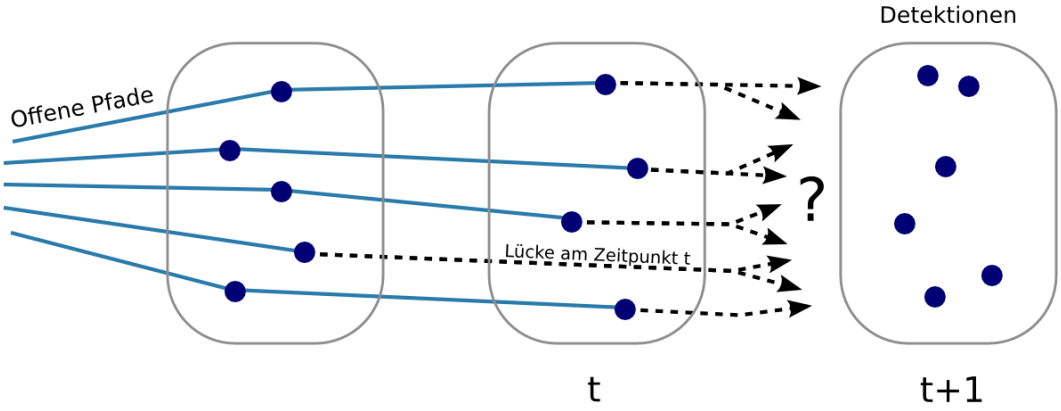

Abbildung 3: Schematische Darstellung des Tracking-Problems.

Die Grundidee des Algorithmus ist wie folgt: zu einem Zeitpunkt während der Ausführung haben wir eine Menge offener Pfade. Das sind die bis zu diesem Zeitschritt aufgebauten Bienen-Tracks. Am nachfolgenden Zeitschritt existiert eine Menge von Detektionen. Diese wollen wir

den offenen Pfaden zuteilen und diese damit weiterschreiben (Abbildung 3). Jede mögliche Verbindung wird bewertet. Anhand der Bewertungen erfolgt die letztendliche Zuteilung.

Drei Fälle sind dann noch zu behandeln:

- Offene Pfade, denen keine Detektion zugewiesen wurde, bekommen eine leere Detektion (Lücke). Es kann nämlich sein, dass die Biene an diesem Zeitschritt nicht vom Decoder erkannt wurde, eine Lücke im Pfad ist daher die richtige Lösung.
- Bienen können auch das Bild ganz verlassen haben, dann wird die Lücke immer länger. Irgendwann muss der Pfad dann aus der Menge der offenen Pfade entfernt werden, also geschlossen werden. Da nun keine Informationen mehr zu dem Track hinzukommen, kann die ID der Biene ermittelt werden.
- Übrig bleiben noch Detektionen, die keinem Pfad zugeteilt wurden. Dies sind die Anfänge neuer, offener Pfade, also insbesondere Bienen die gerade in den Kameraausschnitt gelaufen sind.

Alle Bienen gleichzeitig zu tracken hat in diesem Setup den großen Vorteil, dass das Teilproblem, welche Detektion im nächsten Zeitschritt am besten passt, auf das Problem, welche Detektion im nächsten Zeitschritt am besten passt ohne in einen anderen Pfad noch besser zu passen, reduziert wird.

Somit ergeben sich mehrere Teilprobleme: das Bewerten von Verbindungen (Scoring, Abschnitt 3.2), das Zuteilungsproblem (Abschnitt 3.3), ein Kriterium um Pfade zu schließen (Abschnitt 3.4) und die ID-Mittelung für jeden getrackten Pfad (Abschnitt 3.5).

3.2 Scoring

Aufgabe des Scorings ist es ein Maß zu finden, das aussagt, wie wahrscheinlich zwei Detektionen zur selben Biene gehören. Vorrangig betrifft dies zeitlich direkt aufeinander folgende Detektionen. Aber nicht ausschließlich, denn dort wo Detektionen fehlen und Lücken im Pfad existieren, muss die richtige Verbindung auch zwischen Detektionen über mehrere Zeitschritte hinweg korrekt erfolgen.

Da der Scoring-Wert als Grundlage für den darauf folgenden Zuteilungsschritt dient, ist es nicht nötig, dass der berechnete Score eine definitive Ja-oder-Nein-Entscheidung über die Gültigkeit einer Verbindung fällt. Es reicht aus, wenn die richtige Verbindung in ihrem lokalen Kontext den besten Score bekommt. In anderen Bereichen des Bildes können gleichzeitig falsche Verbindungen mit besserem Score existieren, solange auch dort die richtige Verbindung das lokale Optimum ist, z.B. wenn Teile des Bildes unschärfer sind und dort die Scores generell unsicherer sind.

Trotzdem ist das Maß so zu konstruieren, dass ein Schwellwert ermittelt werden kann, der komplett unmögliche Verbindungen ausschließt. Denn nach jedem Zuteilungsschritt müssen Start-Detektionen übrig bleiben denen nichts zugeteilt wurde und End-Detektionen die nicht

zugeteilt wurden. Ohne ein solches Limit könnte eine das Bild verlassende Biene mit einer das Bild an einer anderen Stelle betretenden Biene verbunden werden, da eine bessere Alternative nicht existiert (Abbildung 4). Dieses Problem taucht natürlich nicht nur an den Bildrändern auf, sondern bei jeder Lücke in den Daten.

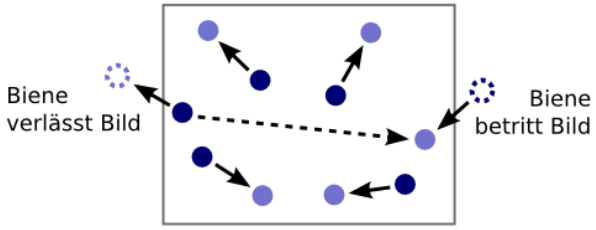

Abbildung 4: Der Algorithmus kann das Verschwinden und Auftauchen von Bienen nicht verfolgen, muss aber unwahrscheinliche Alternativen wie die gestrichelte Verbindung verhindern.

Zwei Features der vorliegenden Daten versprechen intuitiv gute Entscheidungskriterien:

Der euklidische Abstand: Da die Geschwindigkeit, mit der sich eine Biene bewegt, begrenzt ist, kann sie von einem Frame zum nächsten nicht beliebig weit gekommen sein. Sollten Lücken im Pfad sein, ist die mögliche Distanz natürlich um das Vielfache der Lückenlänge größer. Weiterhin ist zu beachten, dass es nicht automatisch besser ist je geringer der euklidische Abstand zwischen zwei Detektionen ist. Denn dass Bienen sich bewegen ist ein vollkommen valides und zu erwartendes Verhalten, und sollte vom Scoring-Maß nicht bestraft werden.

Der Hamming-Abstand bietet sich an um die Ähnlichkeit zwischen zwei Binärcodes zu quantifizieren. Er zählt die Anzahl unterschiedlicher Bit-Stellen. Da in den vorliegenden Daten für jedes Bit eine Wahrscheinlichkeit angegeben ist, kann alternativ auch die Differenz der Wahrscheinlichkeiten summiert werden. Problematisch ist allerdings, dass sich dabei ein Bias zur Wahrscheinlichkeit 0.5 ergibt, denn diese hat im Mittel den geringsten Abstand zu allen anderen Werten. Und das obwohl es von der Aussage her gerade eine unsichere Stelle ist.

Mit diesen beiden Parametern lässt sich intuitiv eine Formel konstruieren die bereits gute Ergebnisse liefert.[1]

Wie sich weitere Parameter in eine mögliche Bewertungsformel integrieren lassen, ist aber kaum noch intuitiv bestimmbar. Bei einer Analyse von Ground-Truth-Daten konnten vermutete Abhängigkeiten nicht bestätigt werden (Abschnitt 7.2). Zum Beispiel dass Bienen sich hauptsächlich vorwärts bewegen und dabei schneller als in andere Richtungen sind.

[1] Zum Beispiel: (Hamming-Abstand+1)∗Euklidischer Abstand. Hamming-Abstand+1 weil sonst eine Detektion mit Hamming-Abstand 0 immer gewinnen würde, selbst wenn sie eine unwahrscheinlich stark abweichende Position hat. Es gilt je geringer desto besser, mit dem genannten Problem, dass Bewegungen generell abgestraft werden.

Deshalb bietet es sich an für die Bewertungsfunktion maschinelles Lernen einzusetzen. Die dafür nötigen Ground-Truth-Daten sollten ohnehin für die Evaluation erstellt werden.

In den Datensatz wurden die folgenden weiteren Features aufgenommen. Sie betreffen immer die Beziehung zwischen zwei Detektionen:

Lückenlänge, zeitlicher Abstand in Frames

Anzahl benachbarter Detektionen zur Position der ersten Detektion aus dem Frame der zweiten Detektion. Gezählt jeweils im Umkreis von 50, 100, 200 und 300 Einheiten. Denn in Bereichen mit vielen Bienen sind die Bewegungen anders. Dort schieben sich die Bienen gegenseitig viel hin und her.

Maximale und mittlere Differenz der Bit-Stellen, also der Wahrscheinlichkeiten.

Konfidenz, d.h. wie sicher die Bits entschlüsselt werden konnten. Errechnet als minimaler Abstand der Bit-Stellen zu 0.5.

Localizer-Saliencies beider Detektionen und die Differenz. D.h. wie deutlich der Localizer ein Tag an dieser Position erkennen konnte.

Betrag der Winkeländerung für jeden der drei Winkel.

Für das maschinelle Lernen wurde XGBoost verwendet. Es ist ein Gradient Tree Boosting das als Open Source Projekt für viele Programmiersprachen zur Verfügung steht. Es eignet sich für eine große Bandbreite an Problemen und hat sich als Tool der Wahl vieler siegreicher Einsendungen bei Machine Learning Wettbewerben einen Namen gemacht[1].

Mit den zwei Ground-Truth-Datensätzen wurde jeweils ein Klassifikator trainiert. Werden die Klassifikatoren auf den jeweils anderen Datensatz angewendet, ergibt sich eine gute bis sehr gute Unterscheidbarkeit richtiger und falscher Verbindungen. Als Grenze wurde 0.5 angesetzt.

True Positives:	7743	False Negatives:	36
False Positives:	127	True Negatives:	418347

Konfusionsmatrix für Datensatz 1 mit XGBoost-Klassifikator 2.

True Positives:	9856	False Negatives:	29
False Positives:	19	True Negatives:	584686

Konfusionsmatrix für Datensatz 2 mit XGBoost-Klassifikator 1.

Die Analyse, wie wichtig welche Features für die trainierten Modelle sind, zeigt, dass der euklidische Abstand das aussagekräftigste Feature ist (Abbildung 5). Die Reihenfolge der restlichen Features kann je nach Datensatz und Trainingsdauer etwas variieren. Kein Zufall ist aber, dass mittlerer und maximaler Bit-Abstand den Hamming-Abstand überholen. Denn die Unterschiede zwischen den Bit-Wahrscheinlichkeiten sind aussagekräftiger als die Differenz der gerundeten Binärcodes. Von den Rotationen ist die Z-Rotation die wichtigste, sie ist die Drehung in der Ebene.

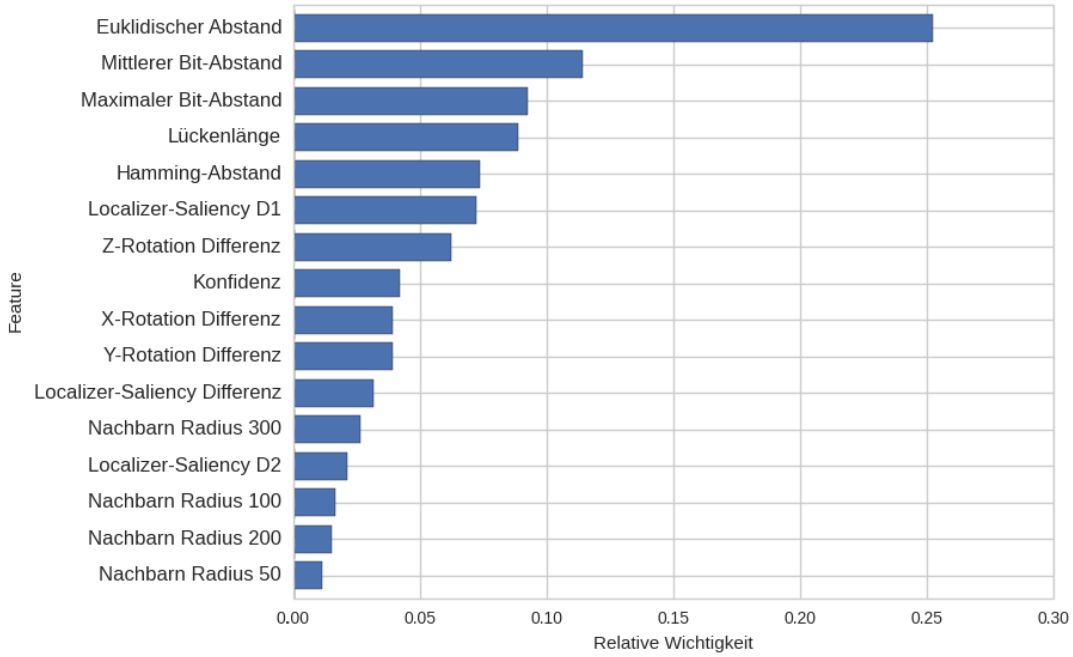

Abbildung 5: Wichtigkeit der Features beim XGBoost-Klassifikator.

3.3 Zuteilungsproblem

Im Schritt nach dem Scoring muss nun anhand der bewerten Kandidaten zugeteilt werden, welchen Verbindungen tatsächlich stattgegeben wird.

Jede Detektion soll nur einem Pfad zugeteilt werden. Ein festes Limit sortiert zu schlechte bewertete Verbindungen von vornherein aus. Da das XGBoost-Verfahren normierte Wahrscheinlichkeiten zwischen 0 und 1 ausgibt, wird dieser Schwellwert auf 0.5 festgesetzt. Und aus Effizienzgründen gehen für jeden offenen Pfad nur die besten Kandidaten ins Rennen, nach aktuellem Stand der Implementierung maximal 20.

Implementiert wurde eine Greedy-Zuteilungsverfahren. Alle Bewertungen werden sortiert und der Reihe nach zugewiesen, sofern der offene Pfad noch nicht bedient und der Kandidat noch nicht verbraucht ist.

Ein anderer Ansatz wäre ein globales Optimum der Zuteilung zu suchen, was bedeutet dass die Summe aller stattgegeben Scores maximiert werden soll. Mit der ungarischen Methode, auch Kuhn-Munkres-Algorithmus genannt, existiert dafür ein bekanntes Verfahren, das hier als zweite Variante implementiert wurde.

Es gibt aber Zweifel ob der ungarische Algorithmus geeignet ist, da in unserem speziellen Fall auch Lücken valide Lösungen sind. Es macht daher nicht Sinn die Zuteilung der Kandidaten so zu optimieren, dass ein Pfad nur seinen zweitbesten Kandidaten bekommt, damit ein anderer Pfad nicht leer ausgeht, wenn für diesen anderen Pfad die Lücke die tatsächliche Lösung wäre (Abbildung 6). Eine Lücke müsste in diesem Fall mit einem Score, der schlechte als sinnvolle Verbindungen, aber besser als falsche Verbindungen ist, bewertet werden, also tendenziell 0.5.

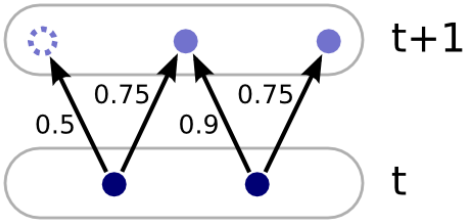

Abbildung 6: Zuteilungsproblem: bei globaler Optimierung wird hier nicht der sicheren Verbindung 0.9 stattgegeben, sondern den 0.75er.

3.4 Pfade Schließen

Pfade, denen lange keine Detektion mehr hinzugefügt werden konnte, müssen irgendwann geschlossen werden. Nach langer Unsichtbarkeit der Biene ist es sicherer im Falle des Wiederauftauchens einen neuen Pfad zu beginnen. Denn im schlimmsten Fall hat sich die Biene schon weit weg bewegt, ein sinnvoller Zusammenhang mit ihrer letzten Position ist nicht mehr möglich und stattdessen ist eine ähnliche Biene im alten Bereich aufgetaucht.

Dass Bienen auftauchen und verschwinden kann viele Ursachen haben. Nicht nur an den Rändern der Wabe betreten und verlassen Bienen das Bild. Oft neigen Bienen auch ihren Oberkörper in eine Wabenzelle, tauchen dann aber irgendwann an der gleichen Stelle wieder auf. Es passiert aber auch, dass Bienen sich umdrehen und auf der Scheibe zwischen Wabe und Kamera weiterlaufen. Dies kann überall auf der Wabe passieren und da die Biene der Kamera dann ihre Unterseite zeigt, ist ihr Tag nicht mehr trackbar.

Hartes Schließen: Bei diesem Ansatz wird der Pfad geschlossen sobald die Lücke länger als ein fester Grenzwert ist. Bedeutet aber dass zum Beispiel bei Grenzwert 20 ein Pfad mit `... - 19xLücke - Detektion - 19xLücke - Detektion - Lücke...` möglich wäre. Das ist nicht unbedingt sinnvoll.

Weiches Schließen: Ein alternatives Verfahren beachtet das Verhältnis von Detektionen und Lücken. Der Pfad soll geschlossen werden, wenn Lücken im Endabschnitt überwiegen. Unsicher ist welches Verhältnis dafür anzulegen ist. Insbesondere tanzende Bienen können oft nur punktuell getrackt werden, weil sie während des Tanzes mit hoher Frequenz wackeln und dann auf den Bildern unscharf sind. Da tanzende Bienen eines der wichtigsten Forschungsobjekte im BeesBook-Projekt sind, und zu wenig Informationen vorliegen wie zuverlässig sie vom Localizer erfasst werden können, müssen unsichere Annahmen an dieser Stelle vermieden werden und zunächst das harte Schließen bevorzugt werden.

3.5 ID-Mittelung

Für jeden Track der beim Tracking gefunden wurde, muss die ID der Biene dieses Tracks ermittelt werden.

Die simpelste Methode ist ein einfacher Mittelwert: die IDs aller Detektionen des Tracks werden bitweise aufaddiert, durch die Anzahl an Detektionen geteilt und wieder auf ganze Bits gerundet.

Der Mittelwert behandelt jede Detektion prinzipiell gleichwertig. Besser wäre es Detektionen, je nachdem wie sicher sie decodiert werden konnten, zu gewichten. Dazu wurde ein Algorithmus implementiert, bei dem jede Detektion nach ihrem Konfidenzwert gewichtet in den Mittelwert einfließt.

Ein drittes Verfahren verfolgt einen anderen Ansatz, indem es versucht wiederum Wissen aus den Truth-Daten anzuwenden. Es basiert auf der Annahme, dass Bit-Flips, also falsch ausgelesene Bits, nicht an jeder Position auf dem Tag gleich wahrscheinlich sind. Insbesondere der weiße und schwarze Halbkreis könnten die Genauigkeit beeinflussen, da sich beispielsweise ein schwarzes Bit neben dem weißen Halbkreis deutlicher abhebt als neben dem schwarzen Halbkreis. Auch auf die Bits an den Grenzen der Halbkreise ergibt sich eine Beeinflussung: bei leicht gekippten Tags kann selbst ein menschlicher Beobachter oft nicht genau unterscheiden, ob die dort liegenden Bits verdeckt oder sichtbar sind, wenn genau schwarze Bits am schwarzen Halbkreis und umgekehrt liegen.

Die Wahrscheinlichkeit eines Bit-Flips soll deshalb abhängig von der Position auf dem Tag und der benachbarten Bits betrachtet werden. Für das betrachtete Bit und seine beiden Nachbarn existieren 8 verschiedene Dreierkombinationen. Multipliziert mit der Anzahl an Bit-Positionen ergibt dies $12 \cdot 8 = 96$ verschiedene Fälle.

Für jeden der 96 Fälle wird aus den Truth-Daten die Wahrscheinlichkeit bestimmt, dass beim Auftreten dieser Kombination ein Bit-Flip vorliegt. Dieser Wert wird anstelle des gelesenen Wertes über alle Detektionen bitweise summiert, und wie bei den anderen Verfahren gemittelt.

4 Evaluierung

In diesem Kapitel soll quantifiziert werden, wie gut der Algorithmus das Problem löst. Dabei wird verglichen, welche Unterschiede die Verwendung der vorgestellten alternativen Lösungsansätze für die Teilprobleme macht.

Um automatisch und zuverlässig Änderungen an den Algorithmen bewerten zu können, wurde schon sehr früh im Projekt ein Evaluierungssystem aufgebaut. Basis aller Bewertungen sind Datensätze die das erhoffte optimale Ergebnis darstellen. Sie wurden per Hand, also durch menschliche Entscheidungen gewonnen. Wie genau diese Ground-Truth-Daten erstellt wurden, ist im Abschnitt 7.1 beschrieben.

Für die hier folgende Evaluation wurden zwei Datensätze aus dem BeesBook-Projekt der Saison 2015 verwendet. Die Vorverarbeitung erfolgte durch den Localizer und Decoder der Pipeline vom Stand Juli 2016. Die zwei Datensätze stammen von verschiedenen Kameras und unterschiedlichen Zeiträumen. Damit ist eine Kreuzvalidierung der Ergebnisse möglich. Da das Erstellen der Ground-Truth-Datensätze mit viel Arbeitsaufwand verbunden ist, konnten leider nicht weitere Datensätze zur Verifikation herangezogen werden.

Zum Start des Projektes wurde mit Datensätzen aus der Saison 2014 gearbeitet. Diese entsprechen heute nicht mehr dem Stand des BeesBook-Projektes. Sie werden im Folgenden aber teilweise noch erwähnt.

4.1 ID-Mittelung

Zuerst sollen die ID-Mittelungsverfahren (Abschnitt 3.5) evaluiert werden. Sie lassen sich einfach unabhängig vom restlichen Algorithmus testen.

Als Eingabe werden die Pfade aus den Truth-Daten genommen. Alle Detektionen die zu einem Pfad gehören bilden eine Menge, von der randomisiert die Untermenge einer bestimmten Größe entnommen wird. Die wahre Bienen-ID für den Pfad ist aus den Truth-Daten bekannt. Nun wird jedes ID-Mittelungsverfahren auf die Untermenge angewendet und geschaut, ob das Ergebnis mit der wahren ID übereinstimmt. Mit 1000 Iterationen pro Pfad wurde das Experiment für verschieden große Untermengen wiederholt.

Größe der Untermenge	1	5	10	20	50
Mittelwert	84.9%	96.2%	97.8%	98.7%	98.9%
Mittelwert nach Konfidenz	84.9%	97.4%	98.4%	98.7%	98.8%
Mittelwert nach Bit-Flip-Statistik	85.0%	94.3%	97.0%	97.9%	98.4%

Verschiedene ID-Mittelungsverfahren, angewendet auf Datensatz 1, Anteil der richtig ermittelten IDs

Größe der Untermenge	1	5	10	20	50
Mittelwert	69.2%	81.9%	83.4%	84.2%	84.2%
Mittelwert nach Konfidenz	69.2%	81.9%	85.2%	85.8%	85.9%
Mittelwert nach Bit-Flip-Statistik	69.2%	80.1%	82.3%	83.0%	84.1%

Verschiedene ID-Mittelungsverfahren, angewendet auf Datensatz 2, Anteil der richtig ermittelten IDs

Eine Untermenge der Größe 1 bildet natürlich keinen Pfad in diesem Sinne, zeigt hier aber die durchschnittliche Erkennungsrate des Decoders auf Einzeldetektionen. Interessant ist, dass sich schon bei einer Pfadlänge von 5 Detektionen die Treffsicherheit erheblich gegenüber den Einzeldetektionen verbessert hat. Mit steigender Pfadlänge erhöht sich der Wert weiter, aber nicht mehr so stark.

Die Resultate für den zweiten Datensatz sind durchgängig schlechter als für den ersten. Mit der geringeren Erkennungsrate auf Einzeldetektionen liegt allerdings auch eine schwierigere Ausgangslage vor.

Der Algorithmus, der nach der Konfidenz gewichtet, ist meist ein kleines Stück besser als die Alternativen, nie aber wesentlich schlechter. Nur der Algorithmus unter Verwendung der Bit-Flip-Statistik erzielt ausnahmslos die schlechtesten Werte. Darauf möchte ich jetzt noch etwas genauer eingehen.

4.1.1 Mittelwert nach Bit-Flip-Statistik

Das Mittelungsverfahren unter Anwendung der Bit-Flip-Statistik wie es in Abschnitt 3.5 beschrieben ist, basiert auf Untersuchungen aus der Bachelorarbeit von Ronja Deisel. Sie konnte in ihrer Analyse nachweisen, dass die Wahrscheinlichkeit von Bit-Flips signifikant von den benachbarten Bits und dem innen anliegenden Halbkreis abhängen[2, S. 15]. Das Verfahren wurde hier insofern erweitert, dass nun jede Position getrennt betrachtet wird, und nicht nur nach der Seite des Halbkreises unterschieden wird.

Es ergeben sich damit bei 12 verschieden Positionen mit je 8 verschiedenen Dreierkombinationen durch die benachbarten Bits insgesamt 96 Wahrscheinlichkeitswerte. Für drei verschiedene Datensätze aus 2014 lassen sich in einem Plot Gemeinsamkeiten finden (Abbildung 7): Ein dunkler Bereich in der oberen Mitte, ein heller Bereich daneben in der Ecke oben rechts, zwei dunkle Bereiche die eine hellere Linie flankieren unten in der Mitte. Ein Plot entsprechender Daten aus 2015 weist allerdings andere Features auf.

Unterschiede gibt es auch bei der Effektivität des Algorithmus. Auf den Daten von 2015 schneidet er wie gezeigt im Vergleich schlechter als die Alternativen ab, auf den Daten von 2014 brachte er aber eine signifikante Verbesserung der Erkennungsrate um bis zu 16%. Was könnte der Grund dafür sein?

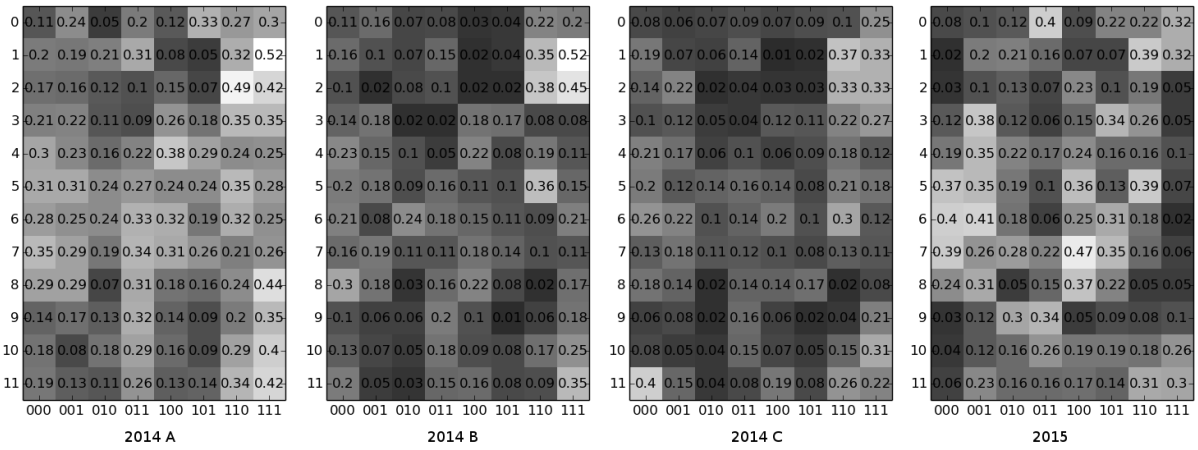

Abbildung 7: Wahrscheinlichkeit eines Bit-Flips für je 12 Positionen × 8 Bit-Tripel aus 4 Datensätzen.

Die Annahme war wie erwähnt, dass Bit-Flips anhängig von Position und Nachbarschaft sind. Es spielen aber noch andere Faktoren eine Rolle. Ob Bit-Flips überhaupt auftreten, hängt wesentlich von der Qualität der Videos ab, unter anderem wie scharf und kontrastreich die Bilder sind. Wie gut der Decoder mit nicht optimalem Ausgangsmaterial umgehen kann, beeinflusst die Statistik weiter. Unterschiedliche Implementierungen des Decoders haben mit unterschiedlichen kritischen Fällen Schwierigkeiten.

Die ursprüngliche Hoffnung, dass die Bit-Flips vorrangig von Position und Nachbarschaft abhängen, und sich somit ein allgemeingültiges Modell formulieren lässt, konnte nicht bestätigt werden.

Auch die Anzahl an Daten, die in die Statistik einfließen, spielt eine Rolle. Für die drei Datensätze von 2014 waren dies jeweils zwischen 10 000 und 20 000 Detektionen. Allerdings nur von 118, 139 und 222 verschiedenen Bienen. Bei den Datensätzen von 2015 sogar nur 79 und 82 verschiedene Bienen. Das schränkt natürlich stark ein wie häufig ein Bit-Pattern in den Daten vorkommt. Für eine allgemeinere Statistik wären also mehr Daten nötig.

4.2 Tracking

Für die weitere Evaluation wurden 4 Maße definiert. Diese werden zunächst vorgestellt.

4.2.1 Evaluationsmaße

Kongruenz: Zu wie viel Prozent die Ground-Truth-Pfade gefunden wurden. Dabei ist es erst mal egal ob die richtige Bienen-ID identifiziert wurde. Es zählt das größte als zusammengehörig erkannte Stück des Pfades, falls der Pfad nicht komplett gefunden wurde.

Überschuss: Wie viel Prozent der gefundenen Pfade nicht Teil des Ground-Truth-Pfades war. Dies misst also irrtümlich zu einem Pfad hinzugefügte Detektionen.

Sprünge: Wie oft der Pfad die verfolgte Biene verliert und eine andere weiter verfolgt. Der Wert ist prozentual zu allen Verbindungen Detektion zu Detektion angegeben.

Identifikation: Für wie viele der Detektionen am Ende die richtige Bienen-ID identifiziert werden konnte. Falls ein Pfad zwar richtig getrackt wurde aber nicht die richtige ID ermittelt werden konnte, wird jede der Detektionen hier als Fehler gezählt.

4.2.2 Ergebnisse

Zur Evaluation standen zwei Truth-Datensätze zur Verfügung. Sie wurden im Kreuzvalidierungsverfahren eingesetzt: beim Tracking eines Datensatzes wird jeweils der XGBoost-Klassifikator verwendet der auf dem anderen Datensatz trainiert wurde.

ID-Mittelung

Datensatz		Kongruenz	Überschuss	Sprünge	Identifikation
1	Mittelwert	98.12%	2.78%	0.25%	94.60%
	Mittelwert nach Konfidenz	98.20%	2.86%	0.23%	95.30%
2	Mittelwert	99.02%	0.07%	0.20%	87.85%
	Mittelwert nach Konfidenz	99.04%	0.07%	0.20%	88.32%

Das ID-Mittelungsverfahren unter Verwendung der Konfidenz erzielt eine kleine Verbesserung.

Zuteilungsproblem

Datensatz		Kongruenz	Überschuss	Sprünge	Identifikation
1	Greedy	98.12%	2.78%	0.25%	94.60%
	Ungarische Methode	98.12%	2.78%	0.25%	94.60%
2	Greedy	99.02%	0.07%	0.20%	87.85%
	Ungarische Methode	99.02%	0.07%	0.20%	87.85%

Beide Zuteilungs-Algorithmen eignen sich gleich gut.

Pfade Schließen

Datensatz		Kongruenz	Überschuss	Sprünge	Identifikation
1	Hartes Schließen	98.12%	2.78%	0.25%	94.60%
	Weiches Schließen	98.65%	1.53%	0.22%	95.99%
2	Hartes Schließen	99.02%	0.07%	0.20%	87.85%
	Weiches Schließen	98.85%	0.04%	0.20%	88.06%

Das weiche Schließen erzielt kleine Verbesserungen, bezüglich Kongruenz bei Datensatz 2 allerdings eine minimale Verschlechterung.

Allgemein lässt sich zusammenfassen, dass der Algorithmus ein gutes Ergebnis liefert. Insbesondere Kongruenz und Überschuss weisen sehr gute Werte auf. Die Werte für Identifikation beim zweiten Datensatz fallen mit 88% etwas geringer aus. Zusammen mit der trotzdem hohen Kongruenz ergibt sich als wahrscheinlichste Erklärung, dass hier für richtig verfolgte Pfade nicht die korrekte Bienen-ID ermittelt werden konnte.

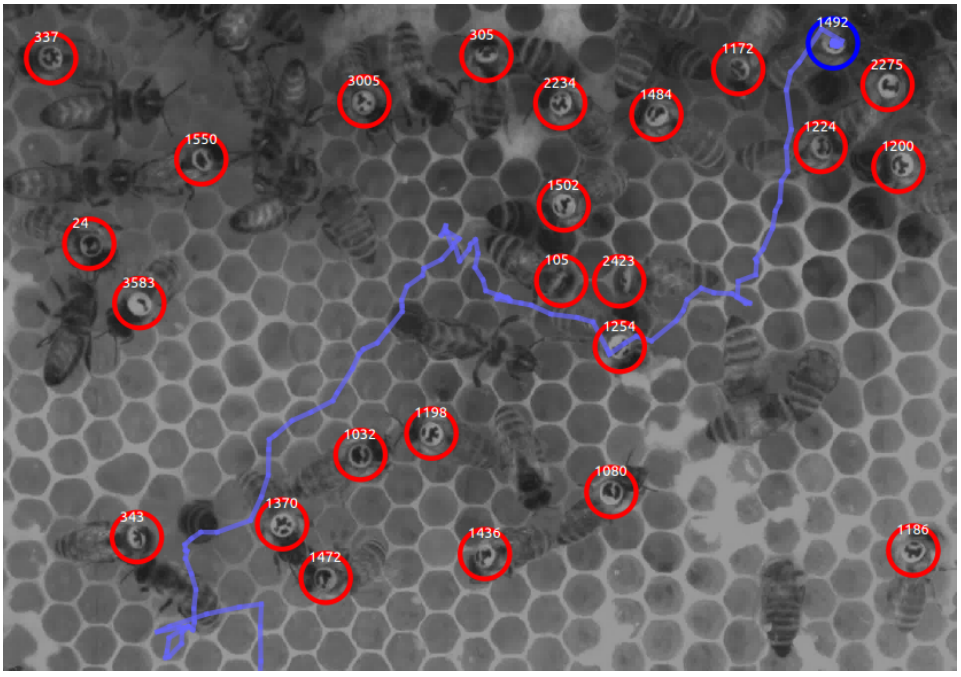

Abbildung 8: Beispiel für einen getrackten Pfad.

5 Diskussion

Der vorgestellte Algorithmus löst das Tracking-Problem zufriedenstellend. Laut Evaluation kann er das angepeilte Qualitätsmaß erreichen. Es ist keine hundertprozentige Genauigkeit, aber das konnte auch nicht erwartet werden. Es wird immer Pfade geben die nicht trackbar sind, denn manche Pfade sind zu kurz um die Biene korrekt identifizieren zu können. Auch beim Erstellen der Truth-Daten gab es diese Fälle, wo selbst der Mensch den Binär-Code zu keinem der Zeitschritte erkennen konnte.

Der Erfolg des Projektes war dabei lange Zeit nicht absehbar. Die anfangs vorhandene Datengrundlage machte enorme Schwierigkeiten. Selbst komplexere Vorgehensweisen, als die in dieser Arbeit behandelten, führten zu keinem zufriedenstellenden Ergebnis. Eine der implementierten Weiterentwicklungen mit Graphen soll in Abschnitt 6.3 noch kurz vorgestellt werden.

Der Durchbruch gelang erst mit den BeesBook-Daten von 2015 und einer verbesserten Vorverarbeitungs-Pipeline. Der Versuchsaufbau war für diese zweite Saison im BeesBook-Projekt noch einmal optimiert worden. Auf der Software-Ebene kam schließlich ein verbesserter Localizer zum Einsatz und ein neu entwickelter Decoder. Letzterer konnte dabei die Erkennungsgenauigkeit von 55% auf 96% steigern[5, S. 15], und brachte damit den wichtigsten Qualitätssprung.

Wie gut sich die mit dieser Arbeit hinzugekommene Stufe in der Verarbeitungs-Pipeline tatsächlich schlägt, wird sicherlich aber erst die jetzt folgende Analyse vollends zeigen. Möglicherweise sind dann noch Anpassungen und Verbesserungen nötig. Gleichermaßen stellte nämlich die Arbeit am Tracking-Problem die Genauigkeit des alten Decoders in Frage, denn mit neu erstellten Truth-Daten wurden deutlich geringere Erkennungsraten als in der ursprünglich geführten Evaluation gemessen (siehe Abschnitt 7.2).

Es stellt sich daher auch hier die Frage, wie gut der vorgestellte Algorithmus auf andere als die getesteten Aufnahmezeiträume anwendbar ist. Der Knackpunkt ist dabei wie allgemeingültig die hier trainierten XGBoost-Klassifikatoren sind. Denn die dafür benötigten Truth-Daten-Sätze sind in ihrer Erstellung so aufwendig, dass sie nicht für jeden zu verarbeitenden Aufnahmezeitraum neu erstellt werden können.

Es wurde deshalb bei der Auswahl der Parameter große Sorgfalt darauf gelegt möglichst unabhängig von der konkreten Aufnahme zu bleiben. Parameter wie zum Beispiel die mittlere Helligkeit des Bildbereiches um den Tag oder die konkreten Bildkoordinaten hätten womöglich den Klassifikator zunächst verbessert, wären aber nicht auf andere Kameraperspektiven oder Aufnahmezeiträume anwendbar. Mit der geführten Kreuzvalidierung konnte die Unabhängigkeit der Modelle sicherlich ein gutes Stück weit bestätigt werden.

6 Ausblick

Die hier vorliegende Arbeit ist in den Kontext des BeesBook-Projektes eingebettet und wird dort relevant bleiben. Mit der Weiterentwicklung des laufenden BeesBook-Projektes wird es auch an der hier vorgestellten Lösung Verbesserungen und Weiterentwicklungen geben. Möglicherweise muss sie sich auch dem Vergleich mit alternativen Lösungsansätzen stellen. Einige sich bereits in Vorbereitung befindende Weiterentwicklungen sollen nun kurz vorgestellt werden.

6.1 Produktiver Einsatz

Zunächst werden die Umstände für einen produktiven Einsatz des vorgestellten Algorithmus geklärt werden müssen. Zur Speicherung der Ergebnisse existiert noch keine abschließende Lösung. Ein in der Anfangsphase des Projektes genutztes Datenbank-Schema erwies sich als nicht optimal ausgelegt. Schon während des Projektes kam es deshalb zu einem Wechsel des Eingabeformates, weg von der Datenbank zu einem eigens entwickelten Binärformat hin. Für die Ausgabe existiert ebenfalls ein vorübergehendes Binärformat, hier ist die Rückkehr zu einer Datenbanklösung aber wahrscheinlich.

Eine andere bereits gestartete Arbeit beschäftigt sich mit der Parallelisierbarkeit des Verfahrens. Bevor die großen Datenmengen des BeesBook-Projektes durchgerechnet werden, ist hier eine Lösung wünschenswert.

Einige Detailfragen sind auch noch zu klären. Wie verfährt man beispielsweise mit sehr kurzen Pfadschnipseln? Für diese ist eine sichere Identifizierung der Biene meist unmöglich und die Daten sind daher in folgenden Analysen mit Vorsicht zu verwenden.

Auch doppeltes Auftreten einer Biene zur gleichen Zeit ist bisher nicht ausgeschlossen, bedeutet aber natürlich, dass für einen der beiden Pfade eine falsche Bienen-ID ermittelt wurde. Möglicherweise kann hier noch ein Verfahren zur nachträglichen Korrektur entwickelt werden.

6.2 Erweiterte Datengrundlage

Weitere Bestrebungen gehen dahin die Datengrundlage für das Tracking-Problem zu erweitern. Ein Ansatz untersucht die Verwendbarkeit zusätzlicher Features die sich aus Ähnlichkeit der Tags auf Bildebene ergeben. Wie bei allen Parametern, die direkt abhängig vom Videomaterial sind, besteht hier die Schwierigkeit unabhängig von konkreten Aufnahmezeiträumen zu bleiben.

Eine ebenfalls interessante Idee ergibt sich aus der Arbeit von Mareike Ziese. Sie entwickelte ein Verfahren, dass nicht nur die Tags auf dem Videomaterial erkennen kann, sondern auch

Bienenkörper von Waben unterscheidet[7]. Die so gewonnen Informationen sind auch für Bienen deren Tag nicht sichtbar ist vorhanden und könnten helfen Lücken in den bisherigen Daten zu schließen.

6.3 Graph Filtering

Ein großes Problem des vorgestellten Algorithmus ist, dass er immer nur die Verbindung zwischen zwei Detektionen untersucht. Er plant nicht darüber hinaus in die Zukunft oder kann einmal gefällte Entscheidungen später korrigieren.

Die Lösung ist ein Filtering mit einer Baumstruktur oder einem Graphen, um mögliche Pfade vorausschauend zu planen. Denn Informationen über die nächsten Zeitschritte sind immer schon vorhanden und können in die aktuelle Entscheidung einfließen.

Diese Erweiterung lässt sich relativ einfach in den vorgestellten Algorithmus integrieren. Es ist letztendlich nur der Scoring-Algorithmus auszutauschen. Der Unterschied ist, dass nun nicht nur einzelne Detektionen, sondern Hypothesen bewertet werden.

Eine Hypothese ist ein möglicher Pfad über zukünftige Zeitschritte mit begrenzter Länge. Die Pfade erhält man durch Traversierung eines Graphen (Abbildung 9).

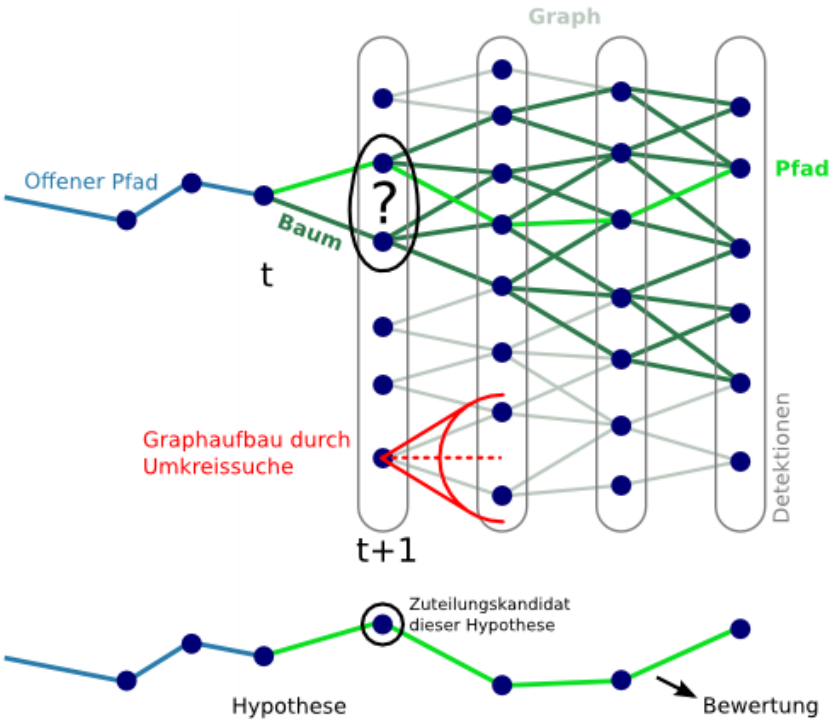

Abbildung 9: Schematische Darstellung des Graph-Trackings.

Die Bewertung erfolgt in zwei Schritten. Zunächst wird der zukünftige Pfad an sich bewertet und danach wie gut der Anschluss an einen offenen Pfad passt. Das Zuteilungsverfahren erfolgt wie im ursprünglichen Algorithmus. Es wird immer nur die erste Detektion des erfolgreichen Pfades zugewiesen, der gesamte Pfad dient also nur zur besseren Hypothesenbewertung durch vorausschauendes Planen.

Im nächsten Zeitschritt werden wieder neue Pfade im Graphen gesucht. Der Graph wächst dabei immer über einen Zeitschritt in die Zukunft und verliert die vorderste Zeitebene. Aufgebaut wird der Graph über eine Umkreissuche, bildet also lediglich die euklidischen Abstände ab. Der Graph ist somit für alle offenen Pfade gleich und muss nur einmal aufgebaut werden.

Zur Bewertung der Hypothesen kommt dabei ebenfalls der XGBoost-Klassifikator für die Detektion-zu-Detektion-Verbindungen zum Einsatz. Zusätzlich sind nun aber weitere Kriterien möglich. Es kann die ganze Geometrie des Pfades bewertet werden, seine Geradlinigkeit, Winkeländerungen etc.

Zu beachten ist aber, dass Pfade auch Lücken enthalten können. Diese entstehen nach wie vor bei der Zuteilung wenn nichts zugewiesen wird. Möchte man aber auch Lücken planen, müssen diese in den Graphen integriert werden. Dazu reicht eine leere Detektion pro Zeitschritt, sie muss nur im Unterschied zu den anderen Detektionen mehrfach zuteilbar sein. Die längstmöglich planbare Lücke ergibt sich aus der Tiefe des Graphen.

Eine der interessantesten Erkenntnisse bei der Entwicklung des Graphen-Algorithmus war, dass es für den Graphenaufbau reicht die 3 dichtesten Nachbarn im Umkreis von 150 Einheiten zu nehmen. Die Statistik aus den Truth-Daten besagt, dass man damit jede wahre Verbindung zwischen Detektionen aus aufeinanderfolgenden Zeitschritten im Graphen abgebildet hat. Die Präzision beträgt dann 43.6%, also das Verhältnis von wahren zu allen Verbindungen im Graphen.

Letztendlich brachte dieser Ansatz leider bei den 2014er Daten nicht die gewünschte Verbesserung. Außerdem stieg die Laufzeit schon bei geringer Tiefe des Graphen unpraktikabel an. Mit der Verfügbarkeit des neuen Decoders wurde ein solch komplexer Ansatz nicht mehr benötigt und schließlich nicht mehr weiter verfolgt.

7 Anhang

7.1 Ground Truth Generierung

Ground-Truth-Daten spielen an mehreren Stellen dieser Arbeit eine zentrale Rolle: beim Trainieren der Klassifikatoren, bei der Evaluierung der Ergebnisse und zur Beantwortung statistischer Analysen.

Ground-Truth-Daten sind in diesem Sinne das angestrebte bestmögliche Ergebnis für das vorliegende Problem. Jeder Algorithmus kann daran gemessen werden, ob oder zu welchem Anteil er dieses Ergebnis ebenfalls erzielt.

Es bieten sich im Wesentlichen zwei Methoden an um Ground-Truth-Daten zu erzeugen. Der erste Ansatz ist, ein Referenzsystem das Problem lösen zu lassen. Der zweite Ansatz wäre simulierte Daten, deren Lösung man also kennt, zu erzeugen. Letzterer Ansatz wurde zumindest in der Anfangsphase des Projektes genutzt: ein Generator für Dummy-Daten simulierte mögliche Eingaben (Abbildung 10). Es ist für das vorliegende Problem aber kaum möglich die richtigen Annahmen zu treffen um Daten mit den gleichen Schwierigkeiten wie bei den realen Eingabedaten zu generieren.

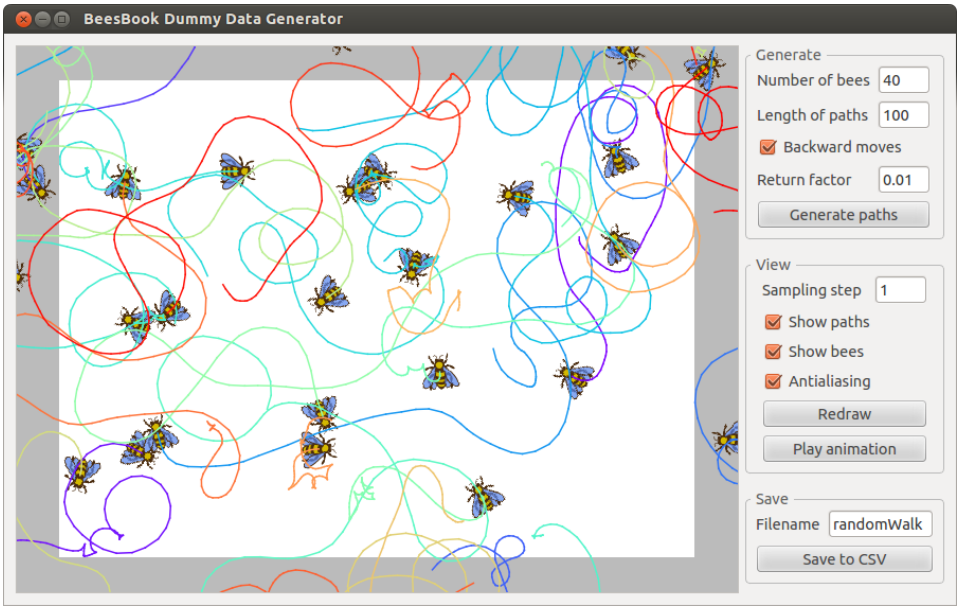

Abbildung 10: Ein Generator zur Simulation von Dummy-Daten.

Da für das Problem keine Referenzimplementierung existiert, bot sich zur Erzeugung der Ground-Truth-Daten nur der Mensch als Referenzsystem an.

Wichtig für das Problem war, dass die Truth-Daten Pfade abbilden, also über aufeinanderfolgende Zeitschritte Wahrheitsinformationen vorhalten. Dies verhinderte die Nutzung bereits

existierender Truth-Daten die zur Evaluierung des Decoders erstellt worden waren, denn für den Decoder waren Truth-Daten über möglichst viele unabhängige Zeitschritte vorteilhaft. Deshalb mussten neue Ground-Truth-Daten erstellt werden.

Zur Unterstützung bei dieser Aufgabe wurde eigens ein Editor mit graphischer Oberfläche implementiert (Abbildung 11). Der Benutzer bekommt die vom Decoder gefundenen Detektionen angezeigt und muss zusammengehörige als Pfad über die Zeit verfolgen, sowie ihre korrekte Bienen-ID entschlüsseln.

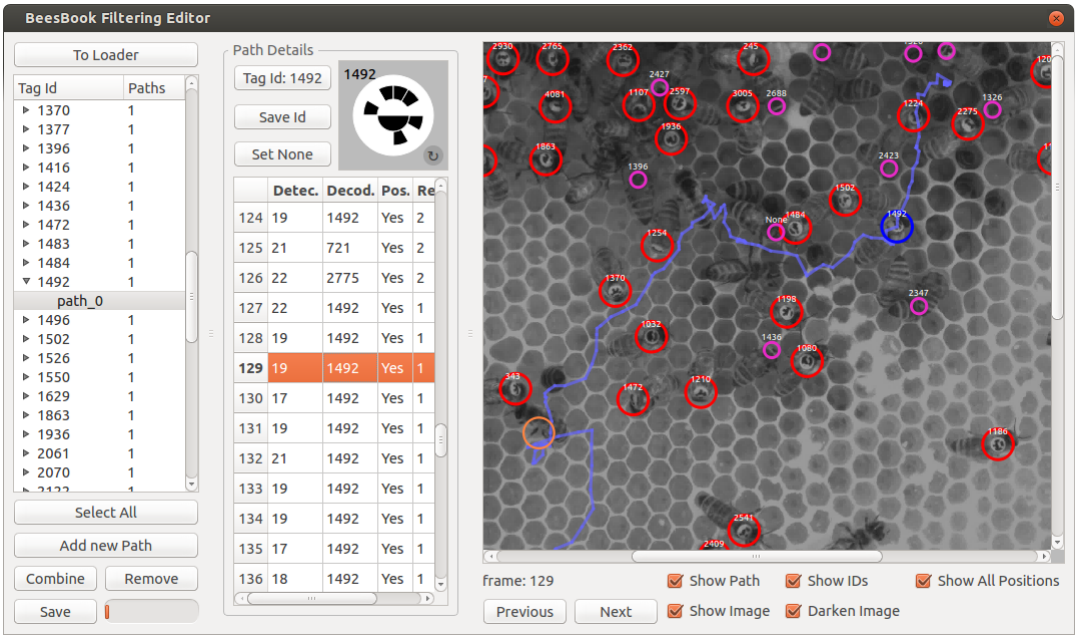

Abbildung 11: Die graphische Oberfläche zum Tracken der Ground-Truth-Daten.

Ein Problem stellt dabei dar, dass der Ground-Truth-Datensatz somit zu einer konkreten Decoder-Ausgabe gehört. Nach Änderungen am Localizer und Decoder können Detektionen detektiert worden sein, die in einem vorherigen Durchlauf nicht erkannt wurden. Damit fehlen dann auch Truth-Daten zu diesen Detektionen.

Um diesem Problem zu begegnen, wurden für die zwei Truth-Datensätze aus der Saison 2015 deshalb auch alle nicht detektierten Tags mit Truth-Daten versehen, wobei die Positionen möglichst exakt von Hand geklickt werden mussten. Der manuelle Arbeitsaufwand stieg damit nochmal erheblich, und wurde mit Hilfe von Praktikanten bewältigt.

Die Truth-Datensätze aus 2014 sind leider nicht so aufbereitet. Denn seit den verbesserten Aufnahmen der Saison 2015 sind die Aufnahmen der Saison 2014 in den Hintergrund gerückt. Der neue Decoder kann aber genauso gut auch auf die alten Daten angewendet werden. Es wäre interessant zu sehen, ob dies die Probleme des Trackings mit den 2014er Daten löst. Dass dies im Rahmen dieser Arbeit nicht untersucht werden konnte, liegt tatsächlich hauptsächlich daran, dass eine erneute Aufbereitung der Truth-Datensätze dafür nötig wäre.

7.2 Ground Truth Analyse

Die in dieser Arbeit erwähnten Ground-Truth-Datensätze weisen grundsätzliche Unterschiede auf.

Zunächst fallen Änderungen im Kamera-Setup auf. Die Bilder zu den Datensätzen von 2014 haben eine schlechtere Ausleuchtung und sind im Randbereich unscharf. Die Bilder für 2015 sind kontrastreicher, die Waben aber teilweise überbelichtet. Was beim Erstellen der Truth-Datensätze außerdem auffiel war, dass sich 2015 mehr Bienen gegenseitig verdecken. Teilweise indem sie übereinander laufen, teilweise dadurch dass sie an der Scheibe zwischen Kamera und Wabe laufen.

Trotzdem sind in den 2015er Daten die Lücken weniger häufig und kürzer, wie folgende Vergleichstabelle zeigt:

	2015 A	2015 B	2014 A	2014 B	2014 C
Datum	18.09.2015	18.09.2015	05.08.2014	05.08.2014	02.08.2014
Uhrzeit	9:36	2:51	15:17	15:17	14:40
Frames	200	200	217	158	240
Kamera-ID	0	1	0	2	0
Anzahl Bienen	79	82	118	222	139
Detektionen	7871	9973	16797	20119	10679
Mit Hamming-Abstand 0	87.0%	75.9%	37.2%	14.8%	36.9%
Häufigkeit Lücken[2]	3.5%	2.8%	15.4%	15.7%	10.2%
Mittlere Lückenlänge[3]	0.163	0.116	0.456	0.486	0.357
Decoder	Network	Network	CV[4]	CV[4]	CV[4]

Vergleich der Truth-Datensätze

Eine unerwartete Erkenntnis war, dass der alte Decoder vom Datensatz 2014 B nur 14.8% der Detektionen richtig erkennen konnte. Die ursprüngliche Evaluation des Decoders hatte ein weitaus besseres Ergebnis ergeben.

Ebenfalls unerwartet fiel eine Analyse zum Zusammenhang zwischen Geschwindigkeit und Bewegungsrichtung der Bienen aus. Die Annahme war gewesen, dass Bienen sich vorrangig vorwärts bewegen und dies mit einer größeren Geschwindigkeit als in andere Richtungen tun. Dazu wurden alle Bewegungsvektoren entsprechend der Ausrichtung der Biene normiert. Die resultierende Verteilung ist aber in alle Richtungen normalverteilt (Abbildung 12).

[2] im Verhältnis zu allen Verbindungen Detektion zu Detektion
[3] in Frames
[4] Computer Vision Pipeline

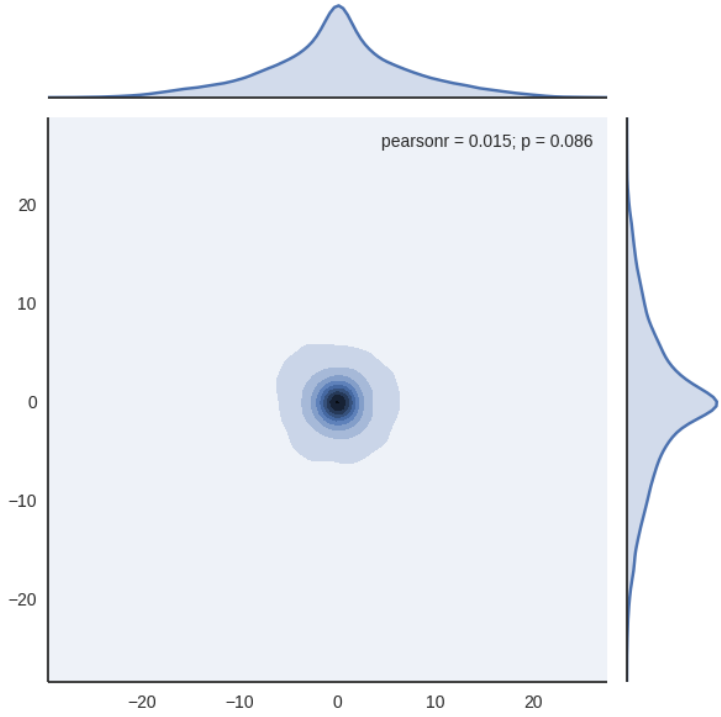

Abbildung 12: Bewegungsvektoren auf einheitliche Ausrichtung der Biene normiert.

Um die Ausrichtung der Biene zu ermitteln, wird sich auf die Orientierung des Tags verlassen. Dass diese vielleicht nicht präzise genug beim Bekleben der Biene ausgerichtet wurden, wäre eine mögliche Fehlerquelle. Die Bewegungsvektoren wurden noch einmal relativ zur vorherigen Bewegung der Biene normiert. Auch auf diesem Wege wurden keine Bewegungstendenzen deutlich. Zunächst gemachte Annahmen in diese Richtung flossen deshalb nicht mehr in den Algorithmus mit ein.

Abbildungsverzeichnis

Literatur

[1] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. *CoRR*, abs/1603.02754, 2016.

[2] Ronja Deisel. Modellierung eines probabilistischen Verfahrens zum Tracken von Bienenpfaden und Analyse der zugrunde liegenden Daten. *Freie Universität Berlin*, 2015.

[3] M. Fiala. ARTag Revision 1. A Fiducial Marker System Using Digital Techniques. *National Research Council Canada*, 1117, 2004.

[4] Danielle P. Mersch, Alessandro Crespi, and Laurent Keller. Tracking Individuals Shows Spatial Fidelity Is a Key Regulator of Ant Social Organization. *Science*, 340(6136):1090–1093, 2013.

[5] Leon Sixt. RenderGAN: Generating realistic labeled data – with an application on decoding bee tags. *Freie Universität Berlin*, 2016.

[6] Fernando Wario, Benjamin Wild, Margaret Couvillon, Raúl Rojas, and Tim Landgraf. Automatic methods for long-term tracking and the detection and decoding of communication dances in honeybees. *Frontiers in Ecology and Evolution*, 3:103, 2015.

[7] Mareike Ziese. Integral Optimization and Completion of an Automatic Decoding of Honeybees Identification Tags. *Freie Universität Berlin*, 2016.