Mapmanagement durch Verwendung von Lebenspunktemodell

B. Eng Oliver Stein
Fakultät Informationstechnik
Hochschule Esslingen
Esslingen, Deutschland
oliver-stein@hotmail.de

Prof. Dr.-Ing. Reiner Marchthaler
Fakultät Informationstechnik
Hochschule Esslingen
Esslingen, Deutschland
Reiner.Marchthaler@hs-esslingen.de

***Zusammenfassung*—Diese Arbeit stellt einen Lösungsansatz für das Erstellen einer Karte für autonome Systeme auf Grundlage von Objekten mit Lebenspunkten vor. Dafür werden von den Sensoren erfasste Punkte mit einem DBSCAN verarbeitet, um zusammenhängende Cluster zu finden. Diese Cluster werden mit Lebenspunkten versehen, welche die Existenzplausibilität widerspiegeln.**

***Index Terms*—Autonomes Fahren, Kartieren, dynamisch, Map Management, Feature Selektion**

I. Motivation

Autonome Fahrzeuge haben das Potenzial die Automobilindustrie zu revolutionieren. Die Grundidee von autonomen Fahrzeugen ist bereits seit fast einem Jahrhundert in den Gedanken der Menschen. In den frühen 30er Jahren wurden erstmals Autos beschrieben, die komplett selbständig und ohne das Eingreifen der Menschen fuhren [2] [3]. Das Entwickeln eines autonomen Fahrzeugs bedeutet, ein System zu entwickeln, welches alle komplexen Verkehrssituationen erfassen und verarbeiten kann [4]. Eine sinnvolle Interpretation der Umgebung, welche dynamische und statische Objekte enthalten kann, ist eine wichtige Aufgabe eines autonomen Systems. Dafür wird mithilfe verschiedener Sensoren eine Karte erstellt, um die Umgebung für das Fahrzeug verständlich zu machen. Bei Sensoren ist es keine Seltenheit eine falsche Detektion zu erfassen. Damit dennoch eine sinnvolle Karte der Umgebung erstellt werden kann, wird ein Konzept einer Karte mithilfe von Lebenspunkten vorgestellt, welches zu jedem Objekt eine Existenzplausibilität mithilfe von Lebenspunkten berechnet.

II. Features

Um eine Karte mit Informationen zu füllen werden sogenannte Features von verschiedenen Sensoren erkannt und in eine Karte aufgenommen. Dabei wird, wenn möglich, aus einer Vielzahl an Sensordaten nur ein kleiner Teil davon als Feature angesehen. Dafür werden zum Beispiel Reichweitensensoren verwendet, um Kanten, Ecken und Lokale Minima zu detektieren. In der echten Welt können diese Detektionen in den Sensordaten Wände, Ecken oder Objekte wie Baumstämme abbilden [1].

A. Statische Features

Features, die sich während einer gesamten Aufzeichnung nicht bewegen und immer an derselben Stelle bleiben, werden als statische Features bezeichnet. Ihre Position ist dauerhaft gleich und werden häufig als Landmarks bezeichnet, da sich solche Objekte besonders zur Orientierung eignen [1].

B. Dynamische Features

Dynamische Features bewegen sich unabhängig vom Fahrzeug. Deren Position oder Stellung verändert sich über die Zeit. Ist das Fahrzeug von mindestens einem dynamischen Feature umgeben, befindet es sich in einer dynamischen Umgebung. Eine dynamische Umgebung kann außerdem durch wechselnden Randbedingungen wie Tageslicht, oder eine nasse Fahrbahn hervorgerufen werden [1].

III. Stand der Technik

A. Provisional Landmark List

Eine gängige Methode mit Outlier umzugehen ist es sie in einer Provisional Landmark List zu sammeln. Die Objekte in dieser Liste werden nicht für die Navigation verwendet und werden erst nach mehrmaligen erkennen aus dieser Liste entfernt und die Karte aufgenommen [1].

B. Landmark Existence Probability

In vielen State-of-the-Art Implementierungen wird eine Landmark Existence Probability aufgestellt. Es wird für jedes Landmark eine Wahrscheinlichkeit angelegt, welche bei mehrmaliger Erkennung des Landmarks um einen fixen Wert inkrementiert wird. Wird ein Landmark nicht erkannt, wird ein fixer Wert abgezogen. Erst ab einer bestimmten Schwelle werden Landmarks in die Map aufgenommen [1].

IV. Kartierung

Während der Fahrt werden Detektionen in eine temporäre Liste L_{detect} aufgenommen. Dabei werden die detektierten Punkte gesammelt und mit ihrer globalen Position in die Liste abgelegt.

$$L_{detect} = \left[\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \quad \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}, \quad \ldots, \quad \begin{bmatrix} x_n \\ y_n \end{bmatrix} \right] \tag{1}$$

mit $x :=$ Position des Fahrzeugs in X Richtung

$y :=$ Position des Fahrzeugs in Y Richtung

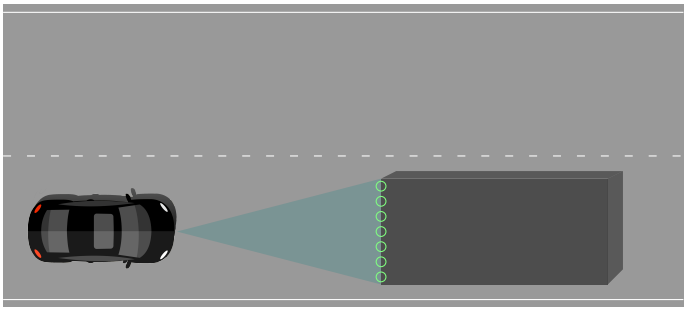

Abbildung 1. Detektion von Objekten mit Distanzsensor.

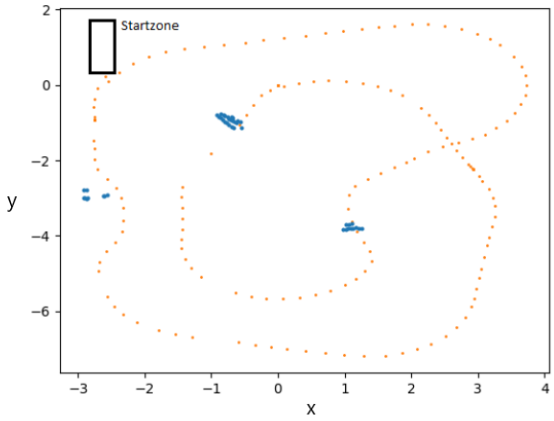

Abbildung 2. Exemplarische Sensordaten L_{detect} (blau) mit Fahrzeugposition (orange).

Abbildung 1 zeigt die Detektion von Objekten mithilfe von einem Distanzsensor. Die grün markierten Punkte stellen die x/y Koordinaten des Objekts relativ zum Fahrzeug dar. Nach der Umrechnung in ein globales Koordinatensystem werden die Punkte in die Liste L_{detect} aufgenommen.

Ein exemplarischer Datensatz zeigt die vereinzelten Punktwolken, die bei einer Fahrt in die Liste L_{detect} aufgenommen werden. Wird diese Liste zusammen mit der Fahrzeugposition in einem X/Y Schaubild dargestellt entsteht eine Karte der Fahrt, wie in Schaubild 2 zu sehen.Damit für die einzelnen unabhängigen Punkte eine sinnvolle Zusammengehörigkeit bestimmt wird, wird ein DBSCAN Algorithmus auf den Daten aus L_{detect} ausgeführt.

V. DBSCAN

Der DBSCAN Algorithmus sieht Cluster als Bereiche mit hoher Punktdichte, die durch Bereiche mit geringerer Dichte getrennt sind. Dafür wird der gesamte Datensatz untersucht und jeder Punkt auf eine Mindestanzahl an Punkten im räumlich naheliegenden Bereich überprüft. Der Algorithmus sucht von jedem Punkt in einem vorgegebenen Suchradius ϵ nach einer Anzahl an mindestens $MinPts$ weiteren Punkten. Werden genug Punkte gefunden, werden alle Punkte im Radius zu einem „*Cluster*“ zusammengefasst. Punkte die nach Beenden des Algorithmus noch nicht zu einem Cluster zugeordnet wurden, zählen als Fehldetektionen bzw. Outlier [5].

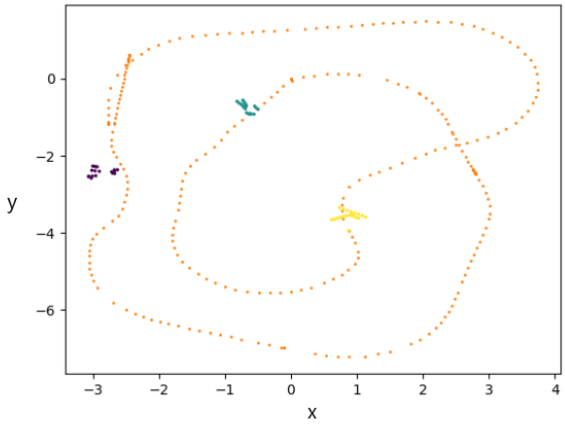

Abbildung 3. Gefundene Cluster aus exemplarischen Datensatz L_{detect} (farbig) mit Fahrzeugposition (orange).

Aus dem vorgegebenen Datensatz L_{detect} ergeben sich dadurch Cluster $C_{1,2,\ldots,n}$ die eine echte Teilmenge von L_{detect} abbilden und zu einer Menge aus Clustern C zusammengefasst werden.

$$C = \begin{bmatrix} C_1 & C_2 & \ldots & C_n \end{bmatrix} \tag{2}$$

$$\text{mit } C_{1,2,\ldots,n} \subsetneq L_{detect}$$
$$\text{und } C_i \neq C_n$$

VI. Lebenspunktemodell

Das Konzept der Lebenspunkte für temporäre Objekte ist schon mehrere Jahre in Videospielen und Brettspielen vertreten. Eines der klassischen Lebenspunktemodelle sieht 100 Lebenspunkte als maximale Anzahl von Lebenspunkte vor. Im Gegensatz dazu sind 0 Lebenspunkte die minimale Anzahl an Lebenspunkten und bedeutet das Ziel ist „*verstorben*“. In Spielen ist es möglich die Lebenspunkte durch Aktionen zu reduzieren bzw. zu erhöhen.

Angelehnt an das klassische Modell werden Lebenspunkte für erkannte Objekte vergeben. Dabei ist der Ansatz für Objekte umgekehrt aufgebaut. Abbildung 4 zeigt das Model für die Vergabe von Lebenspunkten. Die Lebenspunkte stehen für die Existenzplausibilität der einzelnen Objekte. Jedes erkannte Objekt startet mit 0 Lebenspunkten. Die Lebenspunkte können nur durch mehrfache Detektion erhöht werden. Bei erstmaliger Erkennung werden einem Objekt 15 Lebenspunkte hinzugefügt. Bei jeder neuen Erkennung werden weitere 15 Punkte auf die Lebenspunkte aufaddiert. Kann das Fahrzeug ein Objekt nicht erneut an der zuletzt gespeicherten Position erkennen werden 50 Lebenspunkte von dem entsprechenden Objekt abgezogen. Zusätzlich werden im Laufe der Zeit die Lebenspunkte aller Objekte zyklisch mit einem Verfallsfaktor verrechnet. Damit werden die Objekte nicht dauerhaft gespeichert und mit fortschreitender Zeit exponentiell abgebaut bis sie unter einen Schwellwert fallen und gänzlich gelöscht

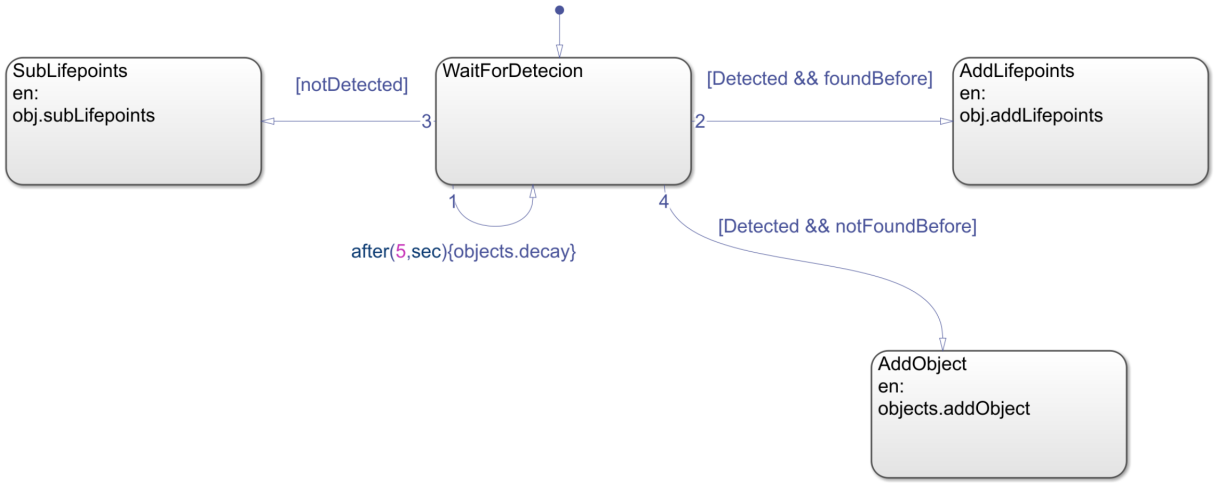

Abbildung 4. Lebenspunktemodell.

werden. Mit diesem Model wird die Menge C angepasst und um die Lebenspunkte erweitert und ergibt die Liste L_{health}.

$$L_{health} = \left[\begin{bmatrix} C_1 \\ h_1 \end{bmatrix} \quad \begin{bmatrix} C_2 \\ h_2 \end{bmatrix} \quad \dots \quad \begin{bmatrix} C_n \\ h_n \end{bmatrix} \right] \tag{3}$$

$$\text{mit } h := \text{Lebenspunkte} \in [0, 100]$$

Dadurch wird eine dynamisch Karte M_{Map} erzeugt und mit Objekten gefüllt ist, die einen Schwellenwert $h_{threhold} = 50$ überschreiten. Dadurch, dass die Objekte einen mit einem Verfall verrechnet werden fallen sie nach einer Zeit aus der dynamischen Karte raus.

$$M_{Map} \subsetneq L_{health} \tag{4}$$

VII. Versuchsaufbau

Hierfür wurde das Fahrzeug auf der Laborstrecke zusammen mit mehreren Objekten positioniert. Dabei fährt das Fahrzeug und scannt mithilfe des verbauten Lidars die erscheinenden Objekte. Der Versuchsaufbau ist in Abbildung 6 abgebildet.

Insgesamt sind vier Runden gefahren worden und dabei wurden Objekte versetzt und hinzugesellt um die Zuverlässigkeit und die Funktionalität des Mapmanagements mit Lebenspunkten zu testen. Die erste Runde beginnt in der Startzone und endet immer, wenn das Fahrzeug wieder auf selber Position wie beim Start der Strecke ist. Im Abbildung 6 ist der Startpunkt unten links mit dem Fahrzeug gekennzeichnet. In der letzten Runde wurde noch ein Objekt hinzugefügt nach der scharfen Kurve oben links im Bild hinzugefügt. Am Ende der Fahrt wurde das Fahrzeug angehalten und für eine bestimmte Zeit stehen gelassen, um den Verfall zu demonstrieren.

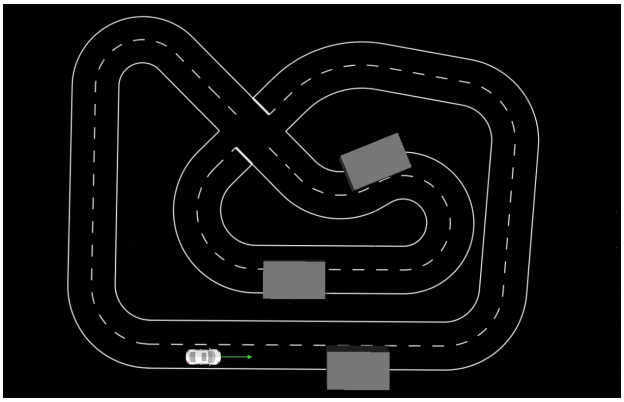

Abbildung 5. Versuchsaufbau.

VIII. Ergebnis

In der Abbildung 7 ist der zeitliche Verlauf der Lebenspunkten der erkannten Objekte zu sehen. Über die Zeit hinweg werden die Lebenspunkte der einzelnen Objekte bei mehrmaliger Erkennung um 15 addiert. Zusätzlich nehmen die Lebenspunkte jedes Objekts über die Zeit hinweg exponentiell ab.

Bei dem Versuchsaufbau wurde in der letzten Runde ein Objekt zu dem Versuch hinzugefügt. Dieses hinzugefügte Objekt wurde in die Liste der erkannten Objekte L_{health} aufgenommen und mit einer Lebenspunktezahl gekennzeichnet. Allerdings wurden zu wenig Punkte erkannt, um Lebenspunkte zu vergeben. Am Ende der Aufzeichnung wird der Verfall der einzelnen Objekte sehr deutlich sichtbar, da die Objekte in L_{health} nicht mehr erkannt wurden.

IX. Fazit

Für das Mapmanagements mit Lebenspunkten im realen Umfeld wurde ein Versuch aufgebaut, der die verschiedenen

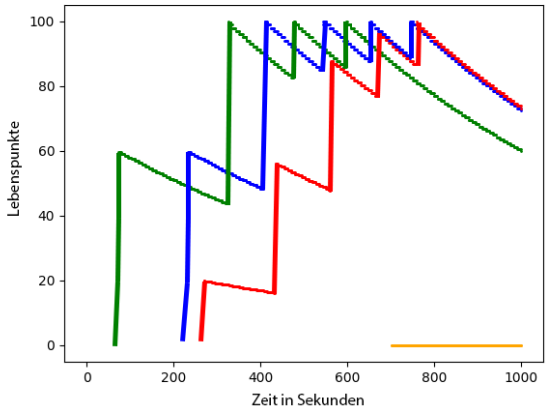

Abbildung 6. Lebenspunkte der erkannten zusammenhängenden Cluster.

Funktionen prüft und das Erstellen der Karte für den exemplarischen Versuch zeigt. Hierfür wurden das Fahrzeug zusammen mit 3 weiteren Objekten auf der Fahrbahn platziert. Beim Fahren auf der Strecke wurden die verschiedenen Objekte vom Lidar wahrgenommen und in eine Liste mit Objekten L_{detect} aufgenommen und ab einer bestimmten Lebenspunktezahl in die Karte M_{Map} aufgenommen. Durch das Clustering mithilfe des DBSCAN kann jeder erkannte Punkt des Lidars zu einem zusammengehörigen Objekt zugeordnet werden und damit auch Lebenspunkte vergeben werden. Dies führte zu einer Karte gefüllt mit Punktwolken mit zugehörigen Lebenspunkten. Für die eigentliche Position des Objekts wird der zentralste Punkt eines Clusters angenommen. Dieser Punkt wird auch Centroid genannt und ist der Punkt, welcher zu allen weiteren Punkten in derselben Punktwolke den geringsten Abstand hat. Bei Betrachten der Punktwolken fällt auf, dass sie sich stark in ihrer Form unterscheiden. Ein idealer Cluster ist ein immer wieder an derselben Stelle erkannter Punkt und damit eine Menge an immer wieder dem gleichen Punkt. Das Gegenteil wären innerhalb der gesetzten Parametern weit verteilten Punkten. Die Form der Punktwolken kann gewichtet in die Verrechnung der Lebenspunkte verwendet werden. Für die Verwendung im realen Fahrzeug ist das Mapmanagement mit Lebenspunkten geeignet, allerdings ist es ratsam die Karte in Form einer Liste mit Positionen abzuspeichern.

LITERATUR

[1] Thrun, Sebastian ; Burgard, Wolfram ; Fox, Dieter: Probabilistic robotics. MIT Press, 2010

[2] Herrmann, Andreas ; Brenner, Walter ; Stadler, Rupert: Autonomous Driving: How the Driverless Revolution Will Change the World. Emerald Publishing Limited. http://dx.doi.org/10.1108/9781787148338. – ISBN 978–1–78714–834–5 978–1–78714–833–8

[3] Maurer, Markus (Hrsg.) ; Gerdes, J. C. (Hrsg.) ; Lenz, Barbara (Hrsg.) ; Winner, Hermann (Hrsg.): Autonomous Driving. http://dx.doi.org/10.1007/ 978-3-662-48847-8

[4] Self-driving car technology — Between man and machine. https://www.bosch.com/stories/autonomous-driving-interview-with-moritz-dechant/

[5] Clustering. scikit-learn. scikit-learn 0.22.2 documentation. https://scikit-learn.org/stable/modules/clustering.html