UNIVERSITÄT
KOBLENZ · LANDAU
Fachbereich 4: Informatik

Terrainklassifikation mit Markov-Zufallsfeldern auf Basis von fusionierten Kamera- und Laserdaten

Diplomarbeit
zur Erlangung des Grades
Diplom-Informatiker
im Studiengang Computervisualistik

vorgelegt von

Marc Arends

Betreuer: Dipl.-Inform. Marcel Häselich, Institut für Computervisualistik, Fachbereich Informatik, Universität Koblenz-Landau
Erstgutachter: Prof. Dr.-Ing. Dietrich Paulus, Institut für Computervisualistik, Fachbereich Informatik, Universität Koblenz-Landau
Zweitgutachter: Dipl.-Inform. Marcel Häselich, Institut für Computervisualistik, Fachbereich Informatik, Universität Koblenz-Landau

Koblenz, im März 2011

Kurzfassung

Ein mobiles System, das sich automatisiert im Outdoor-Bereich fortbewegen soll, muss dafür über ausreichende Kenntnisse des umliegenden Terrains verfügen. Zur Analyse des Terrains werden hierbei häufig ein oder mehrere Laserentfernungsmesser, teilweise auch in Kombination mit Kameras verwendet. Probleme entstehen bei lückenhaften oder verrauschten Daten, da dies zu einer fehlerhaften Bestimmung des Geländes führen kann. Diese Arbeit hat das Ziel ein bereits vorhandenes Verfahren zu erweitern. Dieses basiert auf 3D-Daten, ermittelt durch einen 3D-Laserscanner und soll um eine kontextsensitive Komponente und Daten anderer Sensoren ergänzt werden. Die erste Erweiterung besteht aus einem Markov-Zufallsfeld, welches zum Modellieren der Nachbarschaftsbeziehungen der einzelnen Terrainabschnitte verwendet wird und somit zur Segmentierung eingesetzt werden kann. Als zweite Erweiterung werden die Laserdaten mit Kamerabildern fusioniert, um so die Verwendung zusätzlicher Terrainmerkmale zu ermöglichen.

Abstract

A mobile system, that has to navigate automatically in an outdoor environment, needs to have knowledge about its surrounding terrain. Laser-range-finders, sometimes combined with cameras, are often used to analyse terrain. Several problems, like missing or noisy data, lead to erroneous identification of environment. The target of this work is to add a context-sensitive classification component and data of other sensors to a procedure, based on 3d-data, obtained by a laser-range-finder. The first upgrade consists of a *Markov Random Field*, which is used to model the relationships between neighbouring terrainsegments and allows a segmentation of the whole terrain. The second upgrade fuses laserdata with pictures from cameras to obtain additional terrainfeatures.

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Die Vereinbarung der Arbeitsgruppe für Studien- und Abschlussarbeiten habe ich gelesen und anerkannt, insbesondere die Regelung des Nutzungsrechts.

Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden.	ja ☒	nein ☐
Der Veröffentlichung dieser Arbeit im Internet stimme ich zu.	ja ☒	nein ☐

Koblenz, den 30. März 2011

Danksagung

An dieser Stelle möchte ich mich bei allen Personen bedanken, die mich im Laufe meines Studiums unterstützt haben. Besonders möchte ich meinen Eltern Uwe Arends und Marianne Arends danken, die mir immer alles ermöglicht haben und ohne die dies alles nicht denkbar gewesen wäre. Weiterhin möchte ich Prof. Dr.-Ing. Dietrich Paulus und Marcel Häselich dafür danken, dass sie mich für ein Stipendium der *Stiftung Industrieforschung* vorgeschlagen haben, welches ich daraufhin auch erhalten habe. Dafür möchte ich mich auch bei der *Stiftung Industrieforschung* bedanken, die Studierenden mit ihren Stipendien eine bessere Fokussierung auf die Erstellung einer Qualifikationsarbeit ermöglicht. Ich möchte Marcel Häselich noch ein zweites mal danken, da er mir bei seiner Betreuertätigkeit immer mit Rat zur Seite stand, wenn ich diesen benötigt habe.

Inhaltsverzeichnis

1 Einleitung **17**
- 1.1 System Mustang Mk 1A und Kfz-Aufbau . . . 17
- 1.2 Überblick . . . 18

2 Grundlagen der Markov-Modelle **21**
- 2.1 Markov-Ketten . . . 21
- 2.2 Hidden Markov Model . . . 22
- 2.3 Markov-Zufallsfelder . . . 23
  - 2.3.1 Nachbarschaften in Markov-Zufallsfeldern . . . 24
  - 2.3.2 Definition des Markov-Zufallsfeldes . . . 25
  - 2.3.3 Äquivalenz von Markov- und Gibbs-Zufallsfeldern . . . 25
  - 2.3.4 Klassifikation mit einem Markov-Zufallsfeld . . . 26

3 Sensoren **29**
- 3.1 Sensoren des Mustang Mk 1A . . . 29
  - 3.1.1 Entfernungsmessende Sensoren . . . 29
  - 3.1.2 Bildgebende Sensoren . . . 30

4 Merkmale für die Terrainklassifikation **33**
- 4.1 Merkmale aus Laserdaten . . . 33
  - 4.1.1 Form der Punktewolke . . . 33
  - 4.1.2 Rauheit . . . 35
  - 4.1.3 Steigung zwischen 3D-Punkten . . . 37
  - 4.1.4 Dichte . . . 37
  - 4.1.5 Remission . . . 37
  - 4.1.6 Höhendifferenz . . . 37
- 4.2 Merkmale aus Bilddaten . . . 38
  - 4.2.1 Texturmerkmale . . . 38
  - 4.2.2 Farbe . . . 41
  - 4.2.3 Infrarot-Strahlung . . . 42

5 Terrainklassifikationsverfahren **43**
5.1 Markov-Modelle in der Terrainklassifikation 43
5.2 Terrainklassifikation des Mustang Mk 1A 46

6 Umsetzung einer Terrainanalyse **49**
6.1 Problemstellung 49
6.2 Ziel dieser Arbeit 50
6.3 Implementierung des Markov-Zufallsfeldes 50
6.3.1 Terrainmodell 50
6.3.2 Auswahl der Lasermerkmale 51
6.3.3 Markov-Modell des Terrains 52
6.3.4 Implementierung einer Testsoftware 54
6.3.5 Erweiterung eines Terrainklassifikationsverfahrens mit einem Markov-Zufallsfeld 57
6.3.6 Funktionsweise des erweiterten Terrainklassifikationsverfahrens 58
6.4 Umsetzung der Datenfusion 59
6.4.1 Finden von korrespondierenden Terrain- und Bildbereichen . 61
6.4.2 Auswahl der Bildmerkmale 62
6.4.3 Erweiterung der Software durch Bildmerkmalsberechnung . . 63
6.4.4 Funktionsweise der Bildmerkmalsgewinnung 63
6.5 Lernen der Terraineigenschaften 64
6.5.1 Umsetzung eines Annotationsmodus 64
6.6 Konfigurierbarkeit 65

7 Experimente und Evaluation **67**
7.1 Durchführung der Evaluation 67
7.1.1 Testszenarien 68
7.1.2 Darstellung der Ergebnisse 69
7.2 Ergebnisse 70
7.2.1 Ergebnisse der Detektion der Klasse Street 70
7.2.2 Ergebnisse der Detektion der Klasse Rough 70
7.2.3 Ergebniss der Detektion der Klasse Obstacle 72
7.2.4 Ergebnisse der Laufzeittests 72
7.2.5 Diskussion der Ergebnisse 75

8 Fazit **77**
8.1 Zusammenfassung 77
8.2 Ausblick 79

A Installation der Software **81**
A.1 Installation der Testsoftware 81
A.2 Installation des Roboter Frameworks 81

B Verwendung der Software **83**
B.1 Verwendung des Testprogramms 83
B.2 Verwendung der Framework Erweiterung 84
B.3 Verwendung des Annotationsmodus 85

C Konfigurierbare Parameter **89**

Tabellenverzeichnis

6.1 Klassen, die einer Terrainzelle zugewiesen werden können 51

7.1 Laufzeiten der verschiedenen Verfahren 75

B.1 Befehle für den Annotationsmodus 87
B.2 Merkmalsentsprechungen der Annotationsausgabe 87

C.1 Liste der konfigurierbaren Parameter 89
C.2 Liste der konfigurierbaren Mittelwerte und Standardabweichungen . 90

Abbildungsverzeichnis

1.1 Roboter und Pkw mit Aufbau 18

2.1 Beispiele für Markov Ketten und Hidden Markov Models 23
2.2 Nachbarschaften im Markov-Zufallsfeld 24
2.3 Cliquen in einer Nachbarschaft 25

3.1 *Velodyne HDL-64E S2* mit Laserstrahlen 30

4.1 Drei mögliche Ergebnisse der Principal Component Analyses 35
4.2 Die Geometrie der Local Distance Disturbance 36
4.3 Verschiedene Nachbarschaften von Pixeln 38

5.1 Markov-Modelle für die Terrainklassifikation nach Wellington et al. 45
5.2 Terrainklassifikation nach Neuhaus et al. 47

6.1 Fehlerhafte Terrainzellen 50
6.2 Klassendiagramm der Testsoftware 55
6.3 Graphische Benutzeroberfläche der Testimplementierung 56
6.4 Ablauf des Gibbs-Samplings 57
6.5 Ein- und Ausgabe der Testsoftware 58
6.6 Klassendiagramm mit integriertem Markov-Zufallsfeld 59
6.7 Ablauf des Klassifikationsverfahrens 60
6.8 Ergebnis der Terrainklassifikation mit einem Markov-Zufallsfeld . . 60
6.9 Klassendiagramm des *Workers* zum Berechnen von Texturmerkmalen 63
6.10 Annotationsmodus zum Lernen der Terrainmerkmale 65

7.1 ROC Diagramme der Erkennung der Klasse Street 71
7.2 ROC Diagramme der Erkennung der Klasse Rough 73
7.3 ROC Diagramme der Erkennung der Klasse Obstacle 74

B.1 Benutzeroberfläche der Testsoftware mit Erklärungen 84
B.2 Annotationsmodus mit Erklärungen 85

Kapitel 1

Einleitung

Das Fortbewegen von autonomen mobilen Systemen im Outdoor-Bereich erfordert die Klassifikation des Terrains. Nur wenn das Terrain bekannt ist, kann entschieden werden inwiefern es für das System passierbar ist. Falsch interpretiertes Terrain kann zu fehlerhafter Pfadplanung bis hin zu Unfällen und schweren Beschädigungen des Systems führen. Verfahren, die das Terrainklassifikationsproblem lösen, sind auf Daten angewiesen, die das autonome System mit verschiedenen Sensoren sammelt.

Die einzelnen Sensoren, die ein solches Robotik-System verwendet, sind nicht immer dazu in der Lage, das komplette Terrain zu erfassen und die gesammelten Daten werden oft zusätzlich durch Sensorrauschen gestört. Eine fehlerfreie Klassifikation ist somit nicht immer möglich. Um dennoch eine möglichst präzise Bestimmung des Terrains durchzuführen, werden Ansätze verwendet, die beispielsweise verschiedene Sensordaten fusionieren, oder das Terrain kontextsensitiv mit probabilistischen Methoden analysieren.

In dieser Arbeit wird ein Ansatz vorgestellt, der mithilfe von fusionierten Daten und der Verwendung eines Markov-Zufallsfeld-Modells eine Terrainklassifikation durchführt, mit dem Ziel, einem mobilen Roboter das Navigieren im Outdoor-Bereich zu ermöglichen. Diese Arbeit entstand im Rahmen eines Projektes, das in Zusammenarbeit mit der *Wehrtechnische Dienststelle 51*, einer Dienststelle des *Bundesamtes für Wehrtechnik und Beschaffung*, durchgeführt wird.

1.1 System Mustang Mk 1A und Kfz-Aufbau

Das mobile autonome System, für welches das in der vorliegenden Arbeit vorgestellte Terrainklassifikationsverfahren entwickelt wird, ist der Roboter Mustang Mk 1A (siehe Abbildung 1.1 (a)). Der Einsatz eines autonomen mobilen Systems ist nicht immer in allen Bereichen, besonders während dessen Entwicklungsphase, oh-

(a) **(b)**

Abbildung 1.1: Testaufbauten: Roboter Mustang Mk 1A (a) und Pkw mit auf einem Dachgepäckträger befestigten Sensoren (siehe [NDPP09]) (b)

ne Einschränkungen möglich. Speziell im Hinblick auf die Verkehrssicherheit ist es nicht denkbar, das System Mustang Mk 1A auf Straßen fahren zu lassen, auf denen andere Verkehrsteilnehmer präsent sind. Dennoch ist es erforderlich, auch in solchen Bereichen Daten zu gewinnen, um diese zur Weiterentwicklung des Systems verwenden zu können. Zu diesem Zweck wurde ein Aufbau für Pkws geschaffen (siehe Abbildung 1.1 (b)), der ebenfalls die wichtigsten, für das System relevanten Daten liefern kann. Aufgrund der Verwendung der relevanten Sensoren, macht es für die Lösung des Terrainklassifikationsproblems keinen signifikanten Unterschied, welches System zur Gewinnung der Daten letztendlich eingesetzt wird. Beide Systeme sind in der Lage, die Daten, die von den Sensoren aufgenommen werden, in einem sogenannten *Logfile* abzuspeichern. Die Software kann die Daten aus einem Logfile, in der gleichen zeitlichen Abfolge und Geschwindigkeit abspielen, in der diese aufgenommen wurden. Somit kann die Entwicklung und das Testen eines Echtzeitverfahrens auch mit aufgenommenen Datenmengen erfolgen.

1.2 Überblick

Zu Beginn dieser Arbeit werden in Kapitel 2 die mathematischen Grundlagen und Prinzipien der Markov-Zufallsfelder erläutert, da deren Kenntnisse in folgenden Kapiteln vorausgesetzt werden. Eine Auswahl an verwendbaren Sensoren wird in Kapitel 3 vorgestellt. In Kapitel 4 wird beschrieben, welche Merkmale mit den vorgestellten Sensoren extrahiert werden können. In Kapitel 5 werden Terrainanalyseansätze vorgestellt, die Markov-Modelle verwenden, um eine kontextsensitive Klassifikation durchführen zu können.

Der Inhalt von Kapitel 6 beschreibt die Entwicklung und Implementierung eines Verfahrens zur Terrainklassifikation. Es wird diskutiert, inwiefern bereits vorgestellte Ansätze und Merkmale in die Umsetzung miteinfließen und beschrieben, wie das vorhandene System um ein Markov-Zufallsfeld zur Segmentierung erweitert wird und wie durch Datenfusion Merkmale verschiedener Modalitäten verwendet werden können.

Die Ergebnisse des implementierten Terrainklassifikationsverfahrens werden in Kapitel 7 anhand von Experimenten evaluiert. Kapitel 8 fasst diese Arbeit zusammen und gibt einen Ausblick auf mögliche Weiterentwicklungen.

Kapitel 2

Grundlagen der Markov-Modelle

Ein Teil des Verfahrens, dass in dieser Arbeit vorgestellt wird, besteht aus der Verwendung eines Markov-Zufallsfeld-Modells. Markov-Zufallsfelder sind in der Bildverarbeitung bereits weit verbreitet und werden auch schon in der Robotik in diversen Verfahren zum Terrainlabelling verwendet. Dieses Kapitel behandelt die mathematischen Grundlagen, die notwendig zum Verständnis der Markov-Modelle in späteren Kapiteln sind.

Zuerst werden allgemeine Markov-Ketten und die damit verbundenen Modelle erläutert, da diese die Basis zu den danach vorgestellten Hidden Markov Models darstellen. Nach diesen eindimensionalen Fällen, werden Markov-Zufallsfelder besprochen, da diese eine Erweiterung der Markov-Modelle auf zweidimensionale Datenmengen ermöglichen. Markov-Zufallsfelder sind somit auch für die Anwendung mit einer diskreten zweidimensionalen Terraindarstellung geeignet. Es werden die Grundlagen der Markov-Zufallsfelder erläutert und die Äquivalenz zu Gibbs-Zufallsfeldern beschrieben, da diese die Modellbildung und das Lösen von Klassifikationsproblemen erleichtern, wie es Geman et al. [GG84] beschreiben.

2.1 Markov-Ketten

Markov-Ketten beschreiben eine Reihe von aufeinanderfolgenden Zufallsereignissen. Das bedeutet, dass jedes Zufallsereignis in Abhängigkeit zu anderen Zufallsereignissen steht. Im Klassifikationskontext wird das Auftreten einer bestimmten Klasse als Zufallsereignis betrachtet. Für eine Reihe von Ereignissen $\Omega = \{\Omega_1, \Omega_2, ..., \Omega_N\}$ gilt bei einer Markov-Kette erster Ordnung, dass die Wahrscheinlichkeit, dass Ω_i den Wert bzw. die Klasse ω_i annimmt, durch das Auftreten von ω_{i-1} bedingt ist. Index i beschreibt dabei den Zeitpunkt, oder auch die Position eines Zufallsereignisses in der Markov-Kette. Hierbei gilt, dass $\omega_i \in \mathcal{L}$, wobei $\mathcal{L}$

die Menge der möglichen Klassen bzw. Labels darstellt. Somit ergibt sich folgende Äquivalenz

$$P(\omega_i|\omega_{i-1}) \equiv P(\omega_i|\omega_1, \omega_2, ..., \omega_{i-1}) \tag{2.1}$$

woraus sich als Berechnung für die Wahrscheinlichkeit der kompletten Markov-Kette ergibt

$$P(\Omega) = P(w_1) \prod_{i=2}^{N} P(\omega_i|\omega_{i-1}) \tag{2.2}$$

Das Auftreten der initialen Klasse ω_1 ist das einzige Ereignis in der Markov-Kette, welches nicht von einem vorangegangenen Ereignis abhängt (siehe [TK09] S.522).

Bei einer einfachen Markov-Kette ist das Ergebnis der Zufallsereignisse direkt ersichtlich. Ein Beispiel hierfür ist der Wurf einer Münze und die daraus resultierende Kette von Zufallsereignissen (siehe [TK09] S.532). Für die in Abbildung 2.1 (a) dargestellten Übergangswahrscheinlichkeiten ergibt sich, für eine Folge $\Omega = \{Zahl, Kopf, Kopf\}$, die Wahrscheinlichkeit $P(Zahl) \cdot P(Kopf|Zahl) \cdot P(Kopf|Kopf) = 0.5 \cdot 0.5 \cdot 0.5 = 0.125$, die bei diesem Beispiel für alle Folgen von drei Würfen gleich hoch ist, da von einem fairen Münzwurf ausgegangen wird.

2.2 Hidden Markov Model

Da in vielen Fällen das eigentliche Ergebnis der Zufallsereignisse innerhalb der Kette nicht bekannt ist, sondern nur eine Beobachtung gemacht werden kann, ist eine Erweiterung der Markov-Ketten zu Hidden Markov Models notwendig. Dies bedeutet, dass das Auftreten einer Klasse ω_i nicht sichtbar ist, sondern nur eine Observation y_i an der Stelle i gemacht wird. Da in solch einem Fall die eigentliche Markov-Kette versteckt (engl.: *hidden*) ist, wird von einem Hidden Markov Model gesprochen. Theodoridis et al. ([TK09] S.522-523) machen die Annahmen, dass die Beobachtungen untereinander probabilistisch unabhänigig sind und dass die Wahrscheinlichkeitsdichtefunktionen der einzelnen Klassen ebenfalls nicht voneinander abhängen. Daraus folgt

$$p(Y|\Omega) = \prod_{i=1}^{N} p(y_i|\omega_i) \tag{2.3}$$

Hierbei steht Y für eine Reihe von Beobachtungen $y_1, y_2, ..., y_N$, die zu den Klassen $\omega_1, \omega_2, ..., \omega_N$ gemacht wurden. Die Kombination der Gleichungen 2.2 und 2.3 ergibt die Gesamtwahrscheinlichkeit des Hidden Markov Models

$$p(Y|\Omega)P(\Omega) = P(\omega_1)p(y_1|\omega_1) \prod_{i=2}^{N} P(\omega_i|\omega_{i-1})p(y_i|\omega_i) \tag{2.4}$$

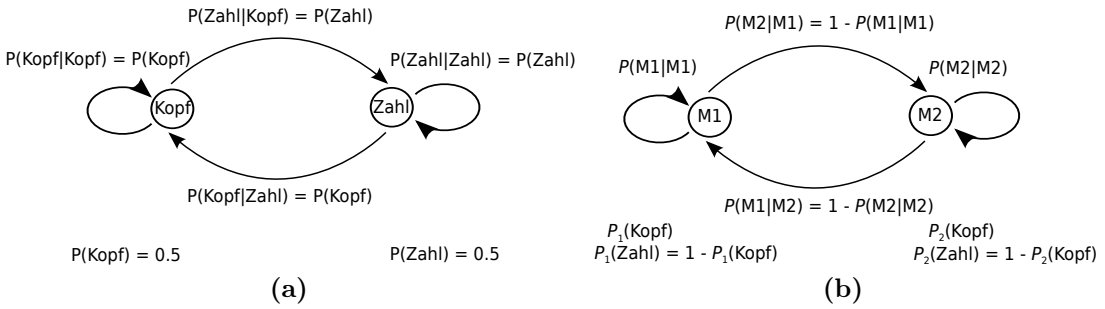


Abbildung 2.1: Markov Modell für das Werfen einer Münze (a) und ein Hidden Markov Model für das Werfen von zwei Münzen hinter einem Vorhang (b) (siehe [TK09] S.533)

Um für die unbekannte Sequenz Ω nun eine gute Klassifikation zu erhalten, sind die darin enthaltenen Klassen so zu wählen, dass das Ergebnis von Formel 2.4 maximal wird (siehe [TK09] S.522-S.523).

Ein Beispiel für ein Hidden Markov Model ist ein Wurf von zwei Münzen hinter einem Vorhang. Es wird eine Sequenz von Münzwürfen initiiert und dem Betrachter ist nicht bekannt, welche der beiden Münzen zu welchem Zeitpunkt geworfen wurde. Es kann lediglich eine Beobachtung $y_i \in \{Kopf, Zahl\}$ pro Wurf gemacht werden. Somit ist das Modell, nach welchem eine Münze zum Wurf ausgewählt wird, versteckt. Wie in Abbildung 2.1 (b) zu sehen ist, werden hierbei mehr Parameter modelliert. Zum einen die Wahrscheinlichkeit, dass eine bestimmte Münze ein bestimmtes Ergebnis liefert. $P_1(Kopf)$ ist beispielsweise die Wahrscheinlichkeit dafür, dass die erste Münze als Ergebnis *Kopf* zeigt. Zusätzlich wird die Wahrscheinlichkeit modelliert, dass eine bestimmte Münze nach einer anderen geworfen wird, zum Beispiel steht $P(2|1)$ für die Wahrscheinlichkeit des Falls, dass die zweite Münze nach einem Wurf mit der ersten Münze geworfen wird (siehe [TK09] S.533).

2.3 Markov-Zufallsfelder

Die Eigenschaften von Markov-Modellen lassen sich mit der Hilfe von Markov-Zufallsfeldern um eine zweite Dimension erweitern. Anstelle einer fortlaufenden eindimensionalen Sequenz wird hierbei ein zweidimensionales Feld von Elementen $\mathcal{S}$ mit Zufallsereignissen Ω betrachtet. Dies bedeutet im Klassifikationskontext, dass ein Zufallsereignis Ω_{ij} an der Stelle (i, j) im Markov-Zufallsfeld eine Klasse $\omega_{ij} \in \mathcal{L}$ zugewiesen bekommen kann. Die Wahrscheinlichkeit dafür, dass Ω_{ij} einen bestimmten Wert bzw. eine bestimmte Klasse annimmt, wird von den benachbarten Klassen beeinflusst, wobei $\mathcal{L}$ die Menge aller möglichen Klassen darstellt (siehe [TK09] S.554).

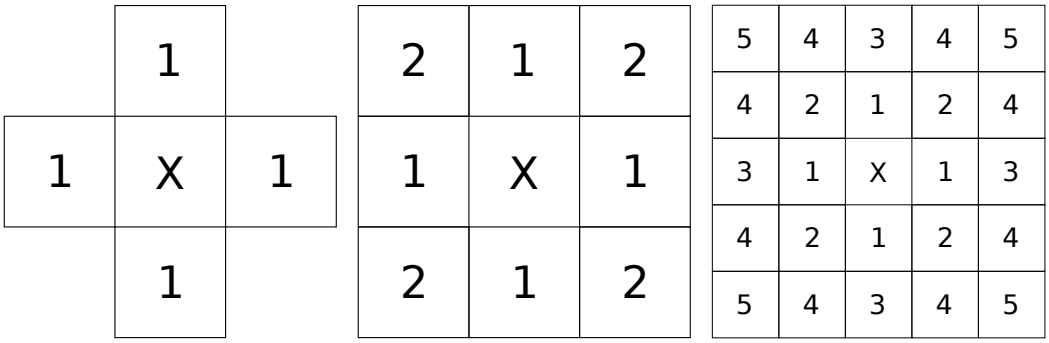


Abbildung 2.2: Nachbarschaft der 1. Ordnung (a), Nachbarschaft der 2. Ordnung (b) und Nachbarschaft der 5. Ordnung (c) (siehe [Li09] S.22)

2.3.1 Nachbarschaften in Markov-Zufallsfeldern

Die Nachbarschaft $\mathcal{N}_{ij}$ ersetzt im Markov-Zufallsfeld das vorangegangene zufällige Ereignis ω_{i-1} der Markov-Ketten. Für eine Nachbarschaft $\mathcal{N}_{ij}$ einer zugewiesenen Klasse ω_{ij} gilt

$$\omega_{ij} \notin \mathcal{N}_{ij}$$
$$\omega_{ij} \in \mathcal{N}_{kl} \Longleftrightarrow \omega_{kl} \in \mathcal{N}_{ij}$$

Dies bedeutet, dass ein Element nicht Teil seiner eigenen Nachbarschaft sein kann, aber dafür Teil der Nachbarschaften, seiner Nachbarn, ist. Hierbei ist zu beachten, dass die Indizes kl eine Position $(k, l) \neq (i, j)$ im Markov-Zufallsfeld beschreiben (siehe [TK09] S.554, [Li09] S.21).

Die Nachbarschaften eines Markov-Zufallsfeldes beschränken sich nicht auf die direkten Nachbarn eines Elements. Es gibt Nachbarschaftssysteme verschiedener Größenordnungen. Eine Auswahl an möglichen Nachbarschaften ist in Abbildung 2.2 zu sehen, wobei X jeweils das Ausgangselement repräsentiert. In jeder Nachbarschaft gibt es verschiedene Cliquen von Nachbarn, die später im Modell berücksichtigt werden. Eine Clique ist hierbei eine Teilmenge von benachbarten Elementen. Je größer ein Nachbarschaftssystem ist, desto größer wird die Anzahl der Cliquen, die berücksichtigt werden können. In Abbildung 2.3 (a) sind Cliquen der Nachbarschaft erster Ordnung zu sehen und Abbildung 2.3 (b) zeigt die zusätzlichen Cliquen der Nachbarschaft zweiter Ordnung. Es ist zu sehen, dass die Anzahl der Elemente, die Teil einer Clique sein können, mit höherer Ordnung steigt (siehe [Li09] S.21-24).

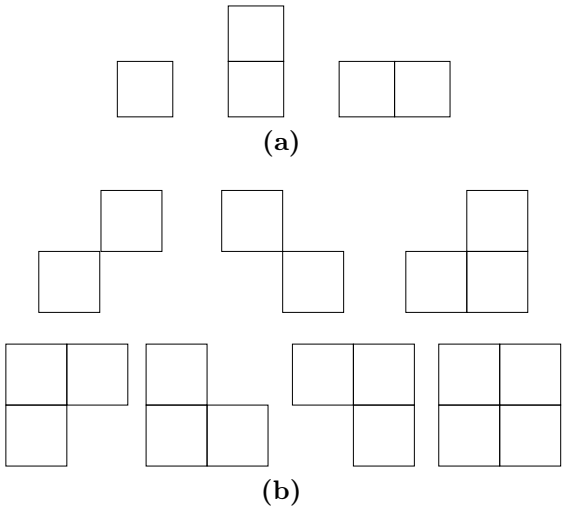


Abbildung 2.3: Cliquen der Nachbarschaft der 1. Ordnung (a), zusätzliche Cliquen Nachbarschaft der 2. Ordnung (b) (siehe [Li09] S.22)

2.3.2 Definition des Markov-Zufallsfeldes

Eine Menge von Zufallsereignissen Ω, die auf einem zweidimensionalem Feld ist, wird Zufallsfeld genannt, wenn jedes Zufallsereignis $\Omega_{ij} \in \Omega$ einen Wert $\omega_{ij} \in \mathcal{L}$ annimmt. Hat jedes Ereignis Ω_{ij} einen bestimmten Wert angenommen, was bedeutet, es gilt $\Omega = \omega$ mit $\omega = \{\omega_{11}, \omega_{12}, ..., \omega_{1m}, \omega_{21}, ...\omega_{nm}\}$, so nennt sich ω eine *Konfiguration* des Zufallsfeldes. Ω heißt Markov-Zufallsfeld, wenn gilt

$$P(\omega) > 0, \ \forall \omega \in \mathcal{L} \tag{2.5}$$

$$P(\omega_{ij}|\omega_{-\omega_{ij}}) = P(\omega_{ij}|\omega_{\mathcal{N}_{ij}}) \tag{2.6}$$

Hierbei stellt $\omega_{-\omega_{ij}}$ die Menge ω ohne das Element ω_{ij} und $\omega_{\mathcal{N}_{ij}}$ die Menge aller $\omega \in \mathcal{N}_{ij}$ dar. Formel 2.5 sagt aus, dass die Wahrscheinlichkeit nicht negativ sein darf und Formel 2.6 verlangt, dass eine Klasse ω_{ij} in einem Zufallsfeld nur von den Klassen in ihrer Nachbarschaft abhängt (siehe [Li09] S.24-26).

2.3.3 Äquivalenz von Markov- und Gibbs-Zufallsfeldern

Die Verwendung eines Gibbs-Zufallsfeldes vereinfacht auf Markov-Zufallsfeldern basierendes Modellieren und Lösen von Klassifikationsproblemen. Eine Menge von Zufallsvariablen Ω mit einem Nachbarschaftssystem $\mathcal{N}$ ist ein Gibbs-Zufallsfeld, wenn die Konfigurationen von Ω der Gibbsverteilung entsprechen, welche definiert ist als

$$P(\omega) = \frac{1}{Z} \exp\left(-\frac{1}{T} U(\omega)\right) \tag{2.7}$$

hierbei ist T der sogenannte *Temperaturparameter* und Z eine Normierungskonstante, die eine Summe über alle möglichen Konfigurationen ω von Ω berechnet mit

$$Z = \sum_{\omega} \exp\left(-\frac{1}{T}U(\omega)\right) \tag{2.8}$$

Bei der Berechnung der Energiefunktion $U(\omega)$ werden alle Elemente des Gibbs-Zufallsfeldes und deren Nachbarschaften miteinbezogen. Somit ist die Energiefunktion definiert mit

$$U(\Omega) = \sum_{ij} \sum_{k} V_k(C_k(i,j)) \tag{2.9}$$

wobei $C_k(i,j)$ die k-te Clique der Nachbarschaft an der Stelle (i,j) ist und $V_k(\cdot)$ das *Potential* einer Clique darstellt, dessen Modellierung problemspezifisch ist. Formel 2.9 zeigt, dass die Summe aller Potentiale den Wert für die Gesamtenergie eines Gibbs-Zufallsfeldes ergibt. Dies bedeutet dass

$$P(\omega_{ij}|\mathcal{N}_{ij}) = \frac{1}{Z}\exp\left(-\frac{1}{T}\sum_{k} V_k(C_k(i,j))\right) \tag{2.10}$$

gilt und somit ermöglicht, die bedingte Wahrscheinlichkeit eines Zufallsereignisses im Gibbs-Zufallsfeld zu berechnen (siehe [TK09] S.554-555 und [Li09] S.26-27).

Markov-Zufallsfelder und Gibbs-Zufallsfelder können aufgrund des *Hammersley-Clifford Theorems* als äquivalent betrachtet werden. Das Theorem besagt, dass „ein Zufallsfeld X genau dann ein Gibbs-Zufallsfeld bezüglich eines Nachbarschaftssystems $\mathcal{N} = \{\mathcal{N}_s, s \in \mathcal{S}\}$ ist, wenn X ein Markov-Zufallsfeld bezüglich eines Nachbarschaftssystems $\mathcal{N} = \{\mathcal{N}_s, s \in \mathcal{S}\}$ ist“ (übersetzt aus [DC04], siehe auch [GG84]).

2.3.4 Klassifikation mit einem Markov-Zufallsfeld

Um eine Klassifikation durchzuführen, gilt es, die Konfiguration zu finden, mit der $P(\omega)$ maximal wird. Dies bedeutet, dass Gleichung 2.7 maximiert werden muss. Bei genauerer Betrachtung fällt ebenfalls auf, dass die Maximierung der A Posteriori Wahrscheinlichkeit auch durch das Minimieren der Energie 2.9 erreicht werden kann (siehe [GG84], [DC04]).

Das Maximieren der A Posteriori Wahrscheinlichkeit im Markov-Zufallsfeld ist in der Praxis nur durch Annäherungsverfahren realisierbar, da ein Testen aller möglichen Konfigurationen nur mit hohem Zeitaufwand möglich ist. Ein Verfahren, die Maximum A Posteriori Schätzung durchzuführen, ist von Geman et al. [GG84] als *Gibbs-Sampler* vorgestellt worden. Der Gibbs-Sampler ist als Verfahren zur Bildsegmentierung entwickelt worden und kann als *Simulated Annealing*[1]

[1] engl. für Simuliertes Abkühlen

Verfahren basierend auf dem Algorithmus von Metropolis et al. [MRR$^+$53] angesehen werden (siehe [GG84]). Geman et al. machen sich hierbei die Äquivalenz von Markov-Zufallsfeldern und Gibbs-Zufallsfeldern (siehe Abschnitt 2.3.3) zu Nutze. Bei Simulated Annealing Verfahren wird der Parameter T (siehe Gleichung 2.7) bei jeder Relaxation verringert und die Konfiguration ω verändert. Dies wird durchgeführt, indem ein Zufallsereignis Ω_{ij} anhand der Verteilungsfunktion der Klassen, eine Klasse zugewiesen bekommt. Durch die Änderungen der Konfiguration wird die Energie minimiert, was wiederum zu einer Maximierung der Wahrscheinlichkeit führt. Dieser Vorgang wird solange wiederholt, bis T einen hinreichend niedrigen Wert angenommen hat, d. h. einen Schwellwert unterschreitet. Der Schwellwert wird dabei niedrig genug gewählt, um eine gute Annäherung an eine Klassifikation mit maximaler Wahrscheinlichkeit zu ermöglichen (siehe [GG84]).

Kapitel 3

Sensoren

Ein autonomes mobiles System verfügt über eine Vielzahl von Sensoren, die verschiedene Arten von Daten über dessen Umwelt sammeln. In diesem Kapitel werden Sensoren vorgestellt, die dem System Mustang Mk 1A zur Verfügung stehen und sich für die Aufgabenstellung der Terrainklassifikation eignen. Weiterhin werden Arten von Merkmalen vorgestellt, die aus den Daten gewonnen und zur Analyse der Umgebung verwendet werden können.

3.1 Sensoren des Mustang Mk 1A

Die Sensoren des Mustang Mk 1A, die sich zur Erfassung der Umgebung im Outdoor-Bereich verwenden lassen, können in zwei Kategorien aufgeteilt werden:

- Entfernungsmessende Sensoren
- Bildgebende Sensoren

Jede dieser beiden Kategorien besteht aus mehreren Sensoren verschiedener Modalitäten, die unterschiedliche Möglichkeiten zur Lösung der Aufgabenstellung bieten.

3.1.1 Entfernungsmessende Sensoren

Der wichtigste der Sensoren des Mustang Mk 1A ist der *Velodyne HDL-64E S2*, ein LIDAR[1] System, dass mit einer Vielzahl von Lasern 3D-Daten der Umgebung erfasst. Hierbei dreht sich ein Sensorkopf, der sich aus 64 einzelnen Lasern zusammensetzt, mit einer Frequenz zwischen 5 Hz und 15 Hz um 360°. Somit werden in der Umgebung um das Fahrzeug herum 1,8 Mio. Distanzmessungen pro Sekunde durchgeführt, die sich in eine 3D-Punktewolke umrechnen lassen. Die 64

[1] engl.: light detection and ranging

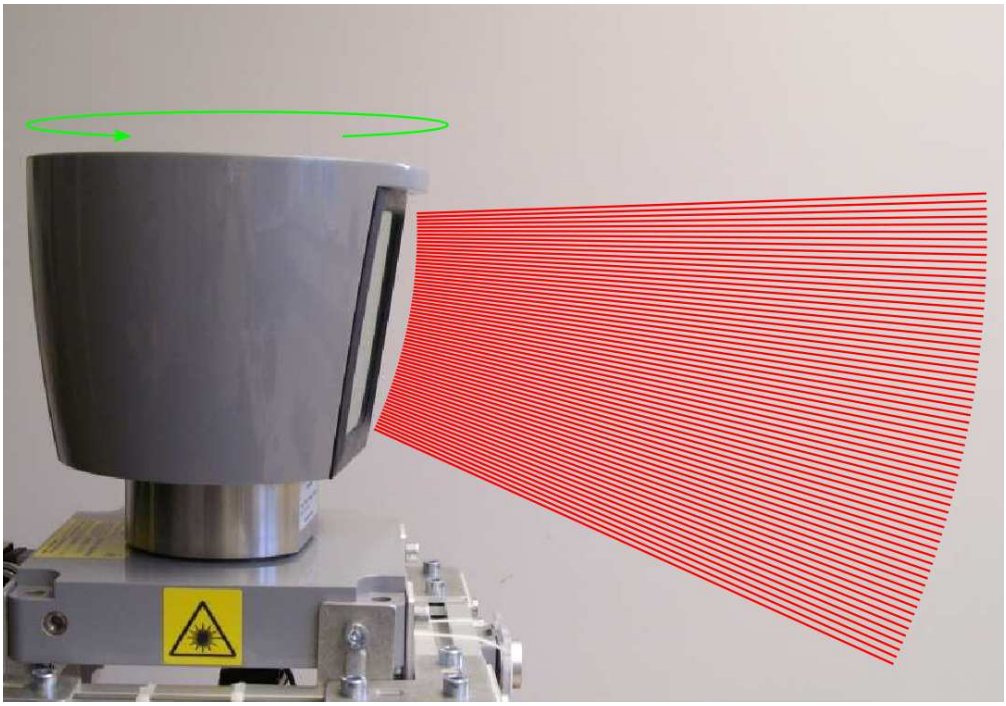


Abbildung 3.1: *Velodyne HDL-64E S2* mit visualisierten Laserstrahlen, sowie Drehrichtung (siehe [LHN+09])

Laser verteilen sich auf einen Öffnungswinkel von $26,8°$ in vertikaler Richtung, wie in Abbildung 3.1 illustriert ist, und haben eine Reichweite von ca. 120 m (siehe [LHN+09]).

Eine weitere Möglichkeit zur Distanzmessung bietet sich dem System Mustang Mk 1A durch dessen Ultraschallsensorik. Diese setzt sich aus 14 frontal und 6 am Heck angebrachten einzelnen Sensoren zusammen. Aufgrund von beschränkter Reichweite und Genauigkeit, sowie der ausschließlich waagerechten Ausrichtung eignen sich diese weniger für eine komplexe Terrainanalyse. Diese ist nämlich darauf angewiesen, genaue Daten über die zu befahrene Umgebung zur Verfügung gestellt zu bekommen.

3.1.2 Bildgebende Sensoren

Der Roboter Mustang Mk 1A ist mit diversen Kameras ausgestattet, die es ermöglichen, das Terrain auch visuell zu untersuchen. Hierzu gehören eine nach vorne gerichtete *Logitech HD Pro Webcam C910* und zwei seitlich angebrachte Kameras des Typs *Philips SPC1300NC*. Die *Logitech HD Pro Webcam C910* kann Farbbilder in einer Auflösung von bis zu 1920×1080 Bildpunkten bereitstellen. Die seitlich

ausgerichteten Kameras ermöglichen eine maximale Auflösung von 1280×1024 Bildpunkten und liefern ebenfalls Farbbilder. Beide Kameratypen können automatisch fokussieren, was das Gewinnen guter Aufnahmen erleichtert, bei einer Kalibrierung allerdings zu Schwierigkeiten führen kann.

Zusätzlich ist das System mit einer Infrarotkamera ausgerüstet, die Wärme in Form von Infrarotstrahlung wahrnimmt. So ist es möglich, Wärmebilder der Umgebung aufzunehmen, die in manchen Fällen ebenfalls bei der Terrainanalyse zur Anwendung kommen könnten.

Kapitel 4

Merkmale für die Terrainklassifikation

Um die Daten der verschiedenen Sensoren zur Terrainklassifikation einsetzen zu können, müssen diese in vielen Fällen zu Merkmalen weiterverarbeitet werden. Zu jeder Sensormodalität lassen sich mehrere solcher Merkmale berechnen, die verschiedene Aussagen über das Terrain ermöglichen. In diesem Kapitel werden Merkmale vorgestellt, die sowohl aus Laser- als auch aus Bilddaten gewonnen werden können.

4.1 Merkmale aus Laserdaten

Obwohl ein LIDAR lediglich Entfernungen bezüglich der eigenen Position und zugehörige Reflektanzwerte misst, ergibt sich daraus eine Vielzahl an Möglichkeiten, um Informationen über das Terrain zu sammeln. Mit Kenntnis der Winkel der einzelnen Laserscans und Kenntnis der Position des Sensors auf dem System, ist es möglich, Punkte in dreidimensionalen kartesischen Koordinatensystemen bezüglich des Roboters zu berechnen. Diese 3D-Punkte eignen sich für die Berechnung verschiedener Merkmale.

4.1.1 Form der Punktewolke

Aus den Entfernungsmessungen eines 3D-LIDAR lässt sich eine dreidimensionale Punktewolke berechnen, die Informationen über die Geometrie der Umgebung des Systems widerspiegelt. Neuhaus et al. [NDPP09] schlagen zur Differenzierung zwischen Hindernissen und flachem Terrain ein hierarchisches Verfahren vor, dass die Verteilung der 3D-Punkte entlang der Koordinatenachsen verwendet. Um diese Verteilung beurteilen zu können, wird die sogenannte Principal Component Analyses

(PCA) verwendet. Hierbei wird die Kovarianzmatrix der vom Laserentfernungsmesser gelieferten 3D-Punktwolke in ihre drei Eigenvektoren zerlegt. Die Eigenwerte der entsprechenden Vektoren geben Aufschluss über die Form der Wolke. Hierbei können drei Fälle auftreten, wie in Abbildung 4.1 zu sehen ist. Wenn alle drei Eigenwerte nahezu gleich groß sind, ist keine eindeutige Form der Wolke erkennbar. Sollte ein Eigenwert wesentlich größer als die beiden anderen Werte sein, kann man von einer Verteilung speziell entlang einer der Richtungsachsen ausgehen. Sollten zwei Eigenwerte viel größer sein als einer der Werte, so ist von einer Verteilung entlang von zwei Achsen auszugehen, d. h. die Punkte verteilen sich ungefähr auf einer Ebene.

Dieses Verfahren wird dabei auf die Zellen einer 2D Grid-Map angewendet, bei der eine Zelle als kleinste Terraineinheit angesehen werden kann. Bei der Wahl der Größe der Bereiche, in denen die PCA zur Anwendung kommt, müssen zwei Probleme betrachtet werden. Das erste Problem besteht darin, dass bei zu kleinen Kantenlängen Zellen entstehen, in denen keine Laserdaten vorhanden sind und somit keine Aussage über diese gemacht werden kann. Das zweite Problem ist der entgegengesetzte Fall, bei dem die Zellen so groß gewählt werden, dass zu viele verschiedene Laserdaten zusammengefasst werden, so dass eine Zelle als Hindernis klassifiziert wird, obwohl sie teilweise befahrbar wäre.

Die beiden genannten Probleme werden mit dem Hierachical PCA Algorithmus behoben. Hierbei wird zuerst das komplette Terrain betrachtet und die PCA durchgeführt. Sollte diese keine Ebene als Ergebnis liefern, so wird das Terrain in zwei Hälften geteilt und auf jeder dieser Hälften erneut eine Analyse mittels PCA durchgeführt. Abbruchkriterium ist entweder, dass ein untersuchter Bereich flach genug ist, um eine Ebene zu sein, oder der Bereich nicht mehr weiter unterteilt werden kann. Beim ersten Fall wird der entsprechende Bereich als befahrbar eingestuft, im zweiten Fall als Hindernis bzw. unbefahrbar. Als Maß für die Flachheit eines Bereichs wurde die sogenannte *Local Height Disturbance* h definiert mit:

$$h = \lambda_k \quad \textbf{mit} \quad k = \underset{i \in \{0,1,2\}}{\operatorname{argmax}} |\boldsymbol{e_i}^{\mathrm{T}} \boldsymbol{z}| \tag{4.1}$$

Hierbei ist λ_k ein Eigenwert der jeweiligen Punktewolke, $\boldsymbol{e_i}$ steht für die Eigenvektoren und $\boldsymbol{z}$ ist ein Vektor, der nach oben zeigt. Wenn h berechnet ist und unter einem Schwellwert liegt, wird von einer flachen Ebene ausgegangen (siehe [NDPP09]). Mit diesem Ansatz ist es möglich, ein Zwei-Klassen-Problem zu lösen. In diesem Fall wird das Terrain nur in Hindernisse und flache Bereiche unterteilt.

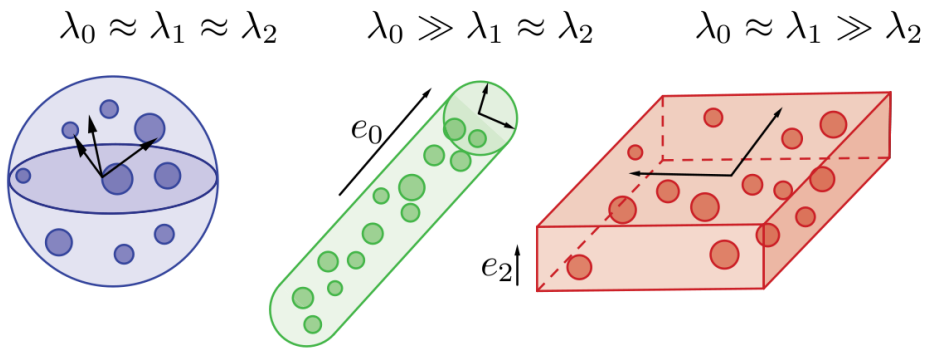


Abbildung 4.1: Die drei möglichen Ergebnisse der Principal Component Analyses mit den Eigenwerten λ_0, λ_1 und λ_2 (siehe [NDPP09])

Vandapel et al. [VHKH04] beschreiben ein weiteres Merkmal, dass auf den berechneten Eigenwerten basiert. Sie verwenden einen dreidimensionalen Vektor, der wie folgt definiert ist

$$\boldsymbol{v}_{sal} = \begin{bmatrix} \lambda_2 \\ \lambda_0 - \lambda_1 \\ \lambda_1 - \lambda_2 \end{bmatrix} \tag{4.2}$$

Jeder Wert im Vektor repräsentiert eine Eigenschaft der Punktewolke. Vandapel et al. bezeichnen den ersten Wert als *point-ness*, den zweiten als *curve-ness* und den letzten als *surface-ness*. Diese Eigenschaften variieren bei verschiedenen Terrainbeschaffenheiten und lassen sich in dieser Form für ein Mehrklassenproblem verwenden (siehe [VHKH04]).

4.1.2 Rauheit

Ein Ansatz von Neuhaus et al. [NDPP09] sieht vor, als flach klassifizierte Terrainzellen, auf einer 2D Grid-Map, mit Laserscans auf ihre *Roughness*[1] zu untersuchen. Diese Information kann beispielsweise dazu verwendet werden, Straßen von Feldern zu unterscheiden. Als Maßeinheit wird hierbei die *Local Distance Disturbance* berechnet. Diese wird verwendet, da die Local Height Disturbance aufgrund von starkem Sensorrauschen ungeeignet ist. Diese ermöglicht es nicht, die oftmals kleinen Unebenheiten wahrzunehmen, die eine Abgrenzung zwischen relativ glatten und sehr rauen Oberflächen zulassen. Bei der Local Distance Disturbance führen bereits kleine Erhöhungen zu gut unterscheidbaren Werten. Hierfür wird ein Distanzunterschied e berechnet. Dieser modelliert, wie groß der Unterschied zwischen

[1] engl.: Rauheit

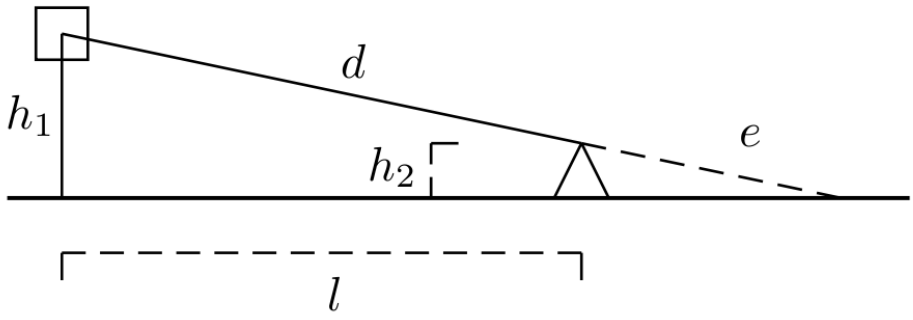


Abbildung 4.2: Die Geometrie der Local Distance Disturbance mit Sensorhöhe h_1, Höhe der Unebenheit h_2, gemessener Distanz d, Entfernung des System zur untersuchten Terrainzelle l und Distanzunterschied e (siehe [NDPP09])

der Distanzmessung des durch ein kleines Hindernis aufgehaltenen Laserscans und der Distanzmessung eines Laserscans ist, der den Boden erreicht hätte. Die Berechnung ist wie folgt definiert

$$e = d \cdot \frac{h_2}{h_1 - h_2} \tag{4.3}$$

Hierbei bezeichnet h_1 die Höhe, auf welcher der Laserentfernungsmesser montiert ist, h_2 die Höhe der Erhebung und d die Distanz, die vom Laserscan gemessen wurde, wie in Abbildung 4.2 verdeutlicht wird.

Um nun die lokale Rauheit des Terrains zu berechnen wird ein Mittelwertfilter auf eine Lasermessung angewendet und eine ungefilterte Version wird von dieser subtrahiert, was wiederum eine hochpassgefilterte Version erzeugt. Somit lässt sich die Distanzabweichung δ_i für jeden Laserscan ermitteln und die zugehörige Varianz $\hat{\sigma}_\delta^2$ für alle n Laserpunkte einer Zelle wie folgt berechnen:

$$\hat{\sigma}_\delta^2 = \frac{1}{n}\sum_{i=1}^{n}\delta_i^2 - \left(\frac{1}{n}\sum_{i=1}^{n}\delta_i\right)^2 \tag{4.4}$$

Um das enthaltene Sensorrauschen σ_L^2 zu eliminieren, wird von Neuhaus et al. ein vom Rauschen befreiter Wert σ_δ^2 bestimmt als $\max(0, \hat{\sigma}_\delta^2 - \sigma_L^2)$, der allerdings immer noch nicht unabhängig von der Distanz des Roboters zu der entsprechenden Terrainzelle ist. Durch Division durch die Distanz des Laserentfernungsmessers zu der untersuchten Zelle d_{Cell}

$$f_r = \frac{\delta_\sigma^2}{d_{\text{Cell}}^2} \tag{4.5}$$

berechnen Neuhaus et al. einen Rauheitswert, der als Terrainmerkmal verwendet werden kann (siehe [NDPP09]).

4.1.3 Steigung zwischen 3D-Punkten

Wolf et al. [WSFB05] schlagen ein Merkmal vor, welches zur Unterscheidung zwischen befahrbarem und unbefahrbarem Terrain, bzw. zwischen flachem und rauem Terrain dient und aus 3D-Laserpunkten berechnet wird. Im Gegensatz zum Ansatz von Neuhaus et al. wird keine Grid-Map verwendet, sondern die einzelnen Laserpunkte direkt mit Merkmalen versehen. Hierzu wird die Steigung zwischen einzelnen Punkte berechnet und die Differenz der Steigungen benachbarter Punktpaare gebildet. Eine flache Ebene zeichnet sich durch eine niedrige Differenz aus, eine grobe Fläche durch eine hohe Differenz (siehe [WSFB05]).

4.1.4 Dichte

Bei einem Terrainmodell, das auf einer Grid-Map basiert, verwenden Wellington et al. [WCS05] die Dichte als Merkmal. Hierbei wird das Verhältnis der Laserstrahlen, die eine Zelle durchdringen und denen, die diese nicht durchdringen, für jede Zelle berechnet. Wellington et al. schlagen vor, die einzelnen Zellen der Grid-Map um Voxel zu erweitern, die Säulen auf den Zellen bilden. Folglich wird das Dichtemerkmal für einzelne Voxel und nicht für eine komplette Terrainzelle berechnet. Das Merkmal soll zur Unterscheidung verschiedener Arten von Vegetation und Festkörpern dienen (siehe [WCS05]).

4.1.5 Remission

Manche Laserscanner, so auch der *Velodyne HDL-64E S2*, sind dazu in der Lage, neben Distanzmessungen auch zugehörige Remissionswerte des reflektierenden Materials zu messen. Dieser Wert gibt Aufschluss darüber, wie stark das getroffene Terrain den Laserstrahl reflektiert. In den Ansätzen von Wellington et al. [WCS05] und Wurm et al. [WKSB09] wird die Remission als Merkmal zur Detektion von Vegetation verwendet. Dies begründet sich darauf, dass das Chlorophyll in Pflanzen die Laserstrahlen stärker reflektiert als beispielsweise Straßenbelag und ermöglicht so die Differenzierung zwischen z. B. Wegen und Wiesen (vgl. [WKSB09]).

4.1.6 Höhendifferenz

Ein Merkmal, das beispielsweise von Happold et al. [HOJ06] verwendet wird, ist die Höhendifferenz. In der Praxis lässt sich dieses Merkmal einfach durch Bildung der Differenz des höchsten und niedrigsten 3D-Punktes einer Terrainzelle auf einer Grid-Map berechnen.

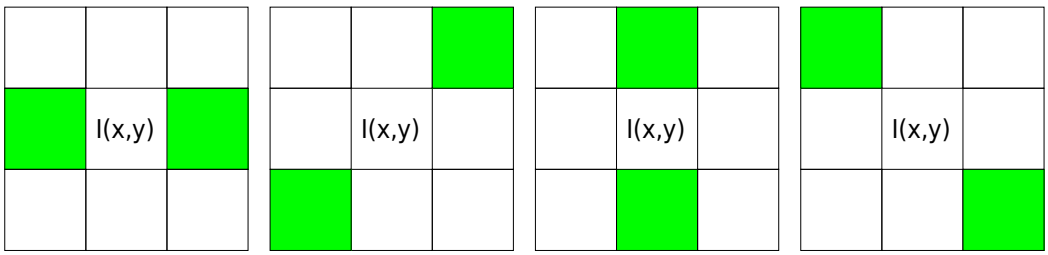


Abbildung 4.3: Richtungen zum Betrachten von Nachbarschaften: horizontal (0°) (a), diagonal (45°) (b), vertikal (90°) (c) und diagonal (135°) (d) (siehe [HDS73])

4.2 Merkmale aus Bilddaten

Neben den Laserdaten lassen sich auch Bilddaten nutzen, um Informationen über das Terrain zu sammeln. Der folgende Abschnitt beschreibt lediglich eine kleine Auswahl an möglichen Bildmerkmalen, die zur Terrainklassifikation eingesetzt werden können.

4.2.1 Texturmerkmale

Mit Kameras ist es auch möglich, visuelle Merkmale von der Umgebung zu extrahieren. Eine Möglichkeit dies zu tun ist es Eigenschaften der Textur von Terrainabschnitten zu berechnen. Einige der bekanntesten Texturmerkmale wurden von Haralick et al. [HDS73] vorgestellt, weshalb diese auch häufig als Haralickmerkmale bezeichnet werden. Diese Texturmerkmale sind nicht direkt aus dem Bild berechenbar. Nach Haralick et al. finden sich die Texturinformationen eines Bildes I in den Nachbarschaftsverhältnissen der Grauwerte der Pixel. Um diese Information abbilden zu können wird eine *Cooccurence Matrix* M_{coo} verwendet. Ein Eintrag in M_{coo} an der Stelle (i, j) gibt die relative Häufigkeit des Vorkommens der Nachbarschaft der Grauwerte i und j mit einem maximalen Abstand d, in I an. Daraus folgt, dass M_{coo} quadratisch ist und eine Seitenlänge der Anzahl der möglichen Grauwerte N_g entspricht. Bei der Erstellung der Matrix werden Nachbarschaften in verschiedenen Richtungen betrachtet, die in Abbildung 4.3 zu sehen sind. Zuerst wird die absolute Häufigkeit der Nachbarschaften in jede Richtung berechnet. Die korrespondierenden Werte der einzelnen Richtungen werden zu Gesamtwerten addiert und müssen danach normiert werden. Die Normierungskonstante ergibt sich aus der Gesamtzahl der betrachteten Nachbarschaften. Im Falle von $d = 1$ und der Betrachtung aller Richtungen, ergibt sich diese aus der Summe aller horizontalen $2N_y(N_x - 1)$, 45° diagonalen $2(N_y - 1)(N_x - 1)$, vertikalen $2N_x(N_y - 1)$ und aller 135° diagonalen $2(N_x - 1)(N_y - 1)$ Nachbarschaften, wobei N_x die Anzahl

der Bildspalten und N_y die Anzahl der Bildzeilen ist. Durch Division jeder absoluten Häufigkeit durch die Konstante erhält man den entsprechenden Wert in der Cooccurence Matrix (siehe [HDS73]).

Haralick et al. stellen 14 Merkmale vor, die sich mit der Cooccurence Matrix berechnen lassen und verschiedene Eigenschaften der Textur eines Bildes quantifizieren. Für die Berechnung gelten folgende Notationen

$$
\begin{aligned}
M_x(i) &= \sum_{j}^{N_g} M_{coo}(i,j) \\
M_y(j) &= \sum_{i}^{N_g} M_{coo}(i,j) \\
M_{x+y}(k) &= \sum_{\substack{i}}^{N_g}\sum_{\substack{j \\ i+j=k}}^{N_g} M_{coo}(i,j) \\
M_{x-y}(k) &= \sum_{\substack{i}}^{N_g}\sum_{\substack{j \\ |i-j|=k}}^{N_g} M_{coo}(i,j) \\
HX &= -\sum_{i}^{N_g} M_x(i)\log(M_x(i)) \\
HY &= -\sum_{j}^{N_g} M_y(j)\log(M_y(j)) \\
HXY1 &= -\sum_{i}^{N_g}\sum_{j}^{N_g} M_{coo}(i,j)\log(M_x(i)M_y(j)) \\
HXY2 &= -\sum_{i}^{N_g}\sum_{j}^{N_g} M_x(i)M_y(j)\log(M_x(i)M_y(j)) \\
Q(i,j) &= \sum_{k}^{N_g} \frac{M_{coo}(i,k)M_{coo}(j,k)}{M_x(i)M_y(k)}
\end{aligned}
$$

Weiterhin beschreibt μ_g den mittleren Grauwert im Bild I, μ_x den Mittelwert der Einträge von M_x, μ_y den Mittelwert der Einträge von M_y und σ_x sowie σ_y entsprechend die Standardabweichungen der Einträge von M_x bzw. M_y. Außerdem steht μ_{x-y} für den Mittelwert der Einträge von M_{x-y} und λ_{Q_1} für den zweitgrößte Ei-

genwert von Q. Unter Verwendung dieser Notationen lassen sich nun 14 Merkmale berechnen, die von Haralick et al. wie folgt definiert worden sind

$$f_{H1} = \sum_{i}^{N_g}\sum_{j}^{N_g} M_{coo}(i,j)^2 \qquad \text{Angular Second Moment} \qquad (4.6)$$

$$f_{H2} = \sum_{n=0}^{N_g-1} n^2 \left\{ \sum_{i}^{N_g}\sum_{j}^{N_g} M_{coo}(i,j) \;\middle|\; |i-j| = n \right\} \qquad \text{Contrast} \qquad (4.7)$$

$$f_{H3} = \frac{\sum_{i}^{N_g}\sum_{j}^{N_g}(ij)M_{coo}(i,j) - \mu_x\mu_y}{\sigma_x\sigma_y} \qquad \text{Correlation} \qquad (4.8)$$

$$f_{H4} = \sum_{i}^{N_g}\sum_{j}^{N_g}(i-\mu_g)^2 M_{coo}(i,j) \qquad \text{Variance} \qquad (4.9)$$

$$f_{H5} = \sum_{i}^{N_g}\sum_{j}^{N_g}\frac{1}{1+(i-j)^2}M_{coo}(i,j) \qquad \text{Inverse Difference Moment} \qquad (4.10)$$

$$f_{H6} = \sum_{i=2}^{2N_g} iM_{x+y}(i) \qquad \text{Sum Average} \qquad (4.11)$$

$$f_{H7} = \sum_{i=2}^{2N_g}(i-f_{H8})^2 M_{x+y}(i) \qquad \text{Sum Variance} \qquad (4.12)$$

$$f_{H8} = -\sum_{i=2}^{2N_g} M_{x+y}(i)\log(M_{x+y}(i)) \qquad \text{Sum Entropy} \qquad (4.13)$$

$$f_{H9} = -\sum_{i}^{N_g}\sum_{j}^{N_g} M_{coo}(i,j)\log(M_{coo}(i,j)) \qquad \text{Entropy} \qquad (4.14)$$

$$f_{H10} = \sum_{i=2}^{2N_g}(i-\mu_{x-y})^2 M_{x-y}(i) \qquad \text{Difference Variance} \qquad (4.15)$$

$$f_{H11} = -\sum_{i}^{N_g-1} M_{x-y}(i)\log(M_{x-y}(i)) \qquad \text{Difference Entropy} \qquad (4.16)$$

$$f_{H12} = \frac{f_{H9} - HXY1}{\max(HX,HY)} \qquad \text{Info. Measure of Corr. I} \qquad (4.17)$$

$$f_{H13} = (1-\exp(-2(HXY2 - f_{H9})))^{\frac{1}{2}} \qquad \text{Info. Measure of Corr. II} \qquad (4.18)$$

$$f_{H14} = (\lambda_{Q_1} \text{ von } Q)^{\frac{1}{2}} \qquad \text{Max. Corr. Coefficient} \qquad (4.19)$$

Probleme sehen Haralick et al. speziell im Bereich verschiedener Helligkeitswerte bei den zu untersuchenden Bildern. Haralick et al. schlagen vor die Merkmale f_{H1}, f_{H8}, f_{H9}, f_{H11}, f_{H12}, f_{H13} und f_{H14} zu verwenden, wenn eine Invarianz zu monotonen Grauwerttransformationen gefordert ist (siehe [HDS73]).

Um zu vermeiden, zu jedem Bildbereich, dessen Textur untersucht werden soll, eine Cooccurence Matrix berechnen zu müssen, schlagen Knauer et al. [KM10] ein mit immer gleichem Aufwand zu berechnendes Merkmal vor. Das Merkmal repräsentiert die Homogenität der Textur einer Region in einem Bild. Hierzu werden zuerst Differenzbilder aus dem Originalbild und einer um einen Pixel verschobenen Version des Bildes erstellt. Dazu wird ein horizontales Differenzbild I_{hdiff} erstellt, für dessen Pixel an der Stelle (x, y) gilt $I_{hdiff}(x, y) = |I(x, y) - I(x+1, y)|$. Analog ergibt sich für den vertikalen Fall $I_{vdiff}(x, y) = |I(x, y) - I(x, y + 1)|$. Um die Homogenität berechnen zu können, muss vorher eine Summed Area Table M_{sat} erstellt werden. Ein Eintrag in M_{sat} an der Stelle (i, j) ist rekursiv definiert

$$M_{sat}(i, j) = M_{sat}(i - 1, j) + M_{sat}(i, j - 1) - M_{sat}(i - 1, j - 1) + I(i, j) \qquad (4.20)$$

Der Wert $M_{sat}(i, j)$ ist die Summe der Grauwerte aller Pixel, des rechteckigen Bereichs mit den Eckpunkten $(0, 0)$ und (i, j) im Bild I. In diesem Anwendungsfall wird jeweils eine Summed Area Table für I_{hdiff} und I_{vdiff} erstellt und zu einer gesamten Summed Area Table addiert. Mit der Summed Area Table ist es nun möglich, schnell ein Merkmal für einen rechteckigen Bereich im Bild zu berechnen. Knauer et al. schlagen vor, den zu untersuchenden Bereich durch eine linke obere Ecke an der Stelle (x_A, y_A) und eine rechte untere Ecke an der Stelle (x_B, y_B) zu definieren und das Homogenitätsmerkmal f_{FH} wie folgt zu berechnen

$$f_{FH} = \frac{M_{sat}(x_B + 1, y_B + 1) + M_{sat}(x_A, y_A) - M_{sat}(x_A, y_B + 1) - M_{sat}(x_B + 1, y_A)}{(x_B - x_A) \cdot (y_B - y_A)} \qquad (4.21)$$

Mit Formel 4.21 ist es nun mit immer gleichem Aufwand möglich, die Homogenität der Textur eines Bildbereiches zu berechnen (siehe [KM10]).

4.2.2 Farbe

Aus Kamerabildern lässt sich die Farbe eines bestimmten Pixels sofort auslesen, ebenso sind schnelle Berechnungen des Mittelwerts oder Histogramms eines Bildbereichs möglich. Obwohl Farbe im Outdoor-Bereich aufgrund von verschiedenen Beleuchtungsituationen stark variieren kann, wird sie, wie z. B. von Wellington et al. [WCS05], auch als Merkmal zur Terrainklassifikation verwendet

4.2.3 Infrarot-Strahlung

Mit Wärmebildkameras ist es möglich, Infrarot-Bilder von der Umgebung aufzunehmen. Die Infrarotmesswerte können, ähnlich wie die Farbe eines regulären Bildes, als Merkmal in der Terrainklassifikation verwendet werden. Wellington et al. [WCS05] verwenden die Infrarotstrahlung, die ein Objekt emittiert, als Merkmal für die Klassifikation verschiedener Pflanzenarten, während Rankin et al. [RHM03] zeigen, wie mit dieser Strahlung negative Hindernisse detektiert werden können.

Kapitel 5

Terrainklassifikationsverfahren

Das Klassifizieren von Terrain ist eine wichtige Aufgabe im Gebiet der Robotik im Outdoor-Bereich, weshalb es schon verschiedene Ansätze zur Lösung dieser Problemstellung gibt. Ein Teil der Ansätze setzt auf eine Kombination von Laser- und Kameradaten zur Gewinnung der benötigten Informationen über die Umgebung. Ein besonderes Augenmerk soll dieser Abschnitt auf die Verfahren legen, die zur Klassifikation ein Markov-Zufallsfeld verwenden, um die Kontextsensitivität von Terrain zu modellieren. Markov-Zufallsfelder haben sich bereits in der Bildverarbeitung zur Lösung verschiedener Probleme etabliert, sei es bei der Segmentierung, der Kantendetektion oder wesentlich komplexeren Aufgaben, wie der Objekterkennung [Li09]. Die hier beschriebenen Verfahren bieten einen Einblick in die Übertragung der Markov-Zufallsfelder zum Lösen des Labellingproblems von Terrainzellen. Es werden Verfahren vorgestellt, die Terrainzellen als Zufallsereignisse im Markov-Zufallsfeld betrachten und so eine Klassifikation durchführen. Ebenso fasst dieses Kapitel zusammen, wie die bisherige Analyse des Terrains, im Software-Framework des Mustang Mk 1A, durchgeführt wird.

5.1 Markov-Modelle in der Terrainklassifikation

Eine Möglichkeit Terrain anhand von 3D-Laserpunkten zu klassifizieren wird von Wolf et al. [WSFB05] vorgeschlagen. Die Punktewolke wird dabei durch eine Reihe von 2D-Laserscans erzeugt, durch einen schräg auf den Boden gerichteten 2D-Laserentfernungsmesser. Jeder Scan wird hierbei als Markov-Kette betrachtet. Jedem Laserpunkt soll eine Klasse zugewiesen werden, die allerdings nicht direkt aus dem Punkt selbst ersichtlich ist. Aufgrund dessen wird jeder 2D-Scan als Hidden Markov Model (siehe Abschnitt 2.2) aufgefasst. Als Observation y_i zu einem Zufallsereignis wird das in Abschnitt 4.1.3 beschriebene Merkmal der Steigung zwischen zwei 3D-Punkten verwendet. Die Menge der möglichen Klassen beschränkt sich

hierbei auf befahrbar A und nicht-befahrbar U. Aus der Wahrscheinlichkeitsverteilung für den Zustandsübergang $P_{st} = P(\Omega_i = \omega_i|\Omega_{i-1} = \omega_{i-1})$, der Wahrscheinlichkeitsverteilung für die gemachte Beobachtung $P_o = P(Y_i = y_i|\omega_i)$ und der initialen Wahrscheinlichkeit, dafür, dass das erste Zufallsereignis eine bestimmte Klasse annimmt $P_{init} = P(\Omega_1 = \omega_1)$, ergibt sich das Hidden Markov Model. Die Klassifikation geschieht nun durch Maximieren von $P(\Omega|Y, P_{st}, P_o, P_{init})$ (siehe [WSFB05]).

Da beim Klassifizieren Fehler auftreten können und auch kein zweidimensionaler Nachbarschaftsbezug durch das Hidden Markov Model hergestellt wurde, verwenden Wolf et al. zusätzlich ein Markov-Zufallsfeld, um die Terrainkarte zu segmentieren. Hierzu werden die 3D-Punkte in eine zweidimensionale Grid-Map projiziert und den Zellen wird anhand der Klassifizierung der beinhalteten Punkte ein Label zugewiesen. Die Wahrscheinlichkeit, dass eine Zelle nun im Kontext ihrer Nachbarschaft befahrbar oder nicht befahrbar ist, wird von Wolf et al. wie folgt modelliert:

$$\frac{P(C_i = A|\ grid)}{P(C_i = U|\ grid)} = \exp\left(\alpha + \sum_{k=1}^{n}(\beta_k n_i^k)\right) \tag{5.1}$$

Die i-te Terrainzelle C_i ist hierbei eine Zufallsvariable, die den Wert A oder U annehmen kann, n_i^k ist die Anzahl der Nachbarn in Entfernung k von C_i, die der Klasse A angehören. Dabei wird β zur Gewichtung verwendet, um zu modellieren, dass eine weiter entferntere Nachbarzelle weniger Einfluss auf C_i hat als eine näher gelegene. Der Wert α ist die Vorannahme über die Anzahl der als A gelabelten Zellen. Das Markov-Zufallsfeld muss so gelabelt werden, dass sich das Ergebnis von Gleichung 5.1 dem Maximum annähert (siehe [WS08][WSFB05]).

Einen weiteren Ansatz stellen Wellington et al. [WCS05] vor. Dieser Ansatz verwendet ebenfalls Markov-Zufallsfelder für das Analysieren von Terrain, um die Befahrbarkeit zu überprüfen. Es werden zwei Markov-Zufallsfelder eingesetzt, die miteinander und mit einem zusätzlichen Hidden Semi Markov Model interagieren. Das erste Markov-Zufallsfeld modelliert die Bodenhöhe und das zweite die Kontextsensitivität von Terrain. Letzteres bedeutet, dass eine Terrainklasse meist eine größere Fläche einnimmt, und dass sich ein kleinerer Bereich einer Terrainklasse nicht inmitten eines Bereiches einer anderen Klasse befindet. Zusätzlich wird die Höhe der Vegetation mit der Hilfe eines Hidden Semi Markov Models modelliert. Ein Hidden Semi Markov Model unterscheidet sich von einem Hidden Markov Model dadurch, dass zu jedem Zufallsereignis eine eigene Sequenz von Beobachtungen gemacht werden kann, anstelle nur einer einzigen (siehe [Mur02]). Die notwendigen Daten werden von zwei 2D-Laserentfernungsmessern, zwei Farbkameras, sowie einer Infrarotkamera geliefert (siehe [WCS05]).

Das Terrain wird bei diesem Ansatz dreidimensional aus Säulen modelliert, die aus übereinander liegenden Voxeln bestehen. Die Klasse einer Voxelsäule an der Position (i, j) im Grid, wird dabei durch eine multinomialverteilte Zufallsvariable

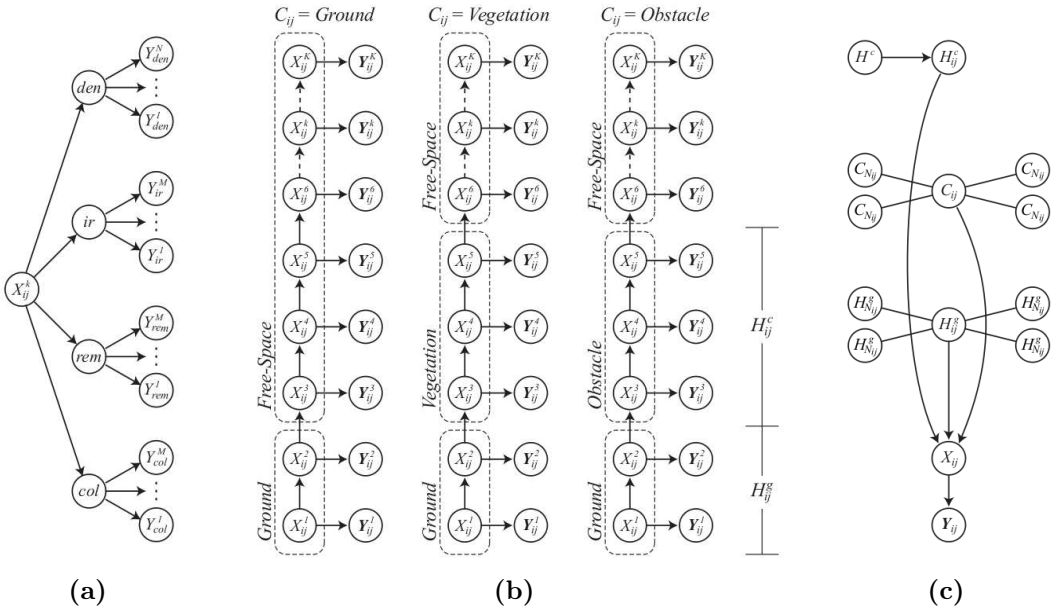


Abbildung 5.1: Modelle die Wellington et al. verwenden: Modell eines Voxels X_{ij}^k mit wahrgenommenen Merkmalen (a), Hidden Semi Markov Model einer Voxelsäule mit zugehörigen Beobachtungen Y_{ij}^k (b) und Markov-Zufallsfeld-Modell zur Beschreibung der Nachbarschaftsverhältnisse für eine Position (i, j). Hierbei ist die Klasse C_{ij} einer Zelle X_{ij} bedingt durch die Klassen der Nachbarzellen $C_{\mathcal{N}_{ij}}$. Ebenso ist die Bodenhöhe H_{ij}^g bedingt durch die Bodenhöhen $H_{\mathcal{N}_{ij}}^g$ der benachbarten Zellen. Die ermittelte Höhe H_{ij}^c der Zelle X_{ij} hängt hingegen von einer Annahme über die Höhe H^c der entsprechenden Klasse ab (siehe [WCS05])

C_{ij} angegeben, die einen Wert ω_{ij} annehmen kann, der für eine Terrainklasse steht. Als Merkmale verwenden Wellington et al. Dichte Y_{den} (siehe Abschnitt 4.1.4), Remission Y_{rem} (siehe Abschnitt 4.1.5), Infrarotstrahlung Y_{ir} (siehe Abschnitt 4.2.3) und Farbe Y_{col} (siehe Abschnitt 4.2.2). Es wird jedem Voxel X_{ij}^k ein Beobachtungsvektor $\mathbf{Y}_{ij}^k = [Y_{den}, Y_{rem}, Y_{ir}, Y_{col}]$ zugewiesen, zu sehen in Abbildung 5.1 (a), der mit Hilfe der Markov-Modelle verarbeitet wird. Hierbei steht k für die Position des Voxels in der Säule und ij für die Position (i, j) der Säule in der Grid-Map. Die Säule wird dabei als Markov-Kette gesehen und modelliert die Abhängigkeit von einem Voxel zum vorangegangenen, sowie die bedingte Wahrscheinlichkeit für den Beobachtungsvektor. Diese Art der Modellierung soll hier unter anderem verhindern, dass sich über einem Voxel, der als freier Raum gilt, kein Voxel befindet, der als Boden erkannt werden könnte. Zum Modellieren der Nachbarschaftsbezie-

hungen wird ein Markov-Zufallsfeld verwendet, für welches gilt

$$P(C_{ij} = \omega_{ij} | C_{\mathcal{N}_{ij}}) \propto \exp\left(-\beta_C \sum_{\{s,t\} \in \mathcal{N}_{ij}} (\omega_{ij} \neq \omega_{st})\right) \tag{5.2}$$

Hierbei steht $C_{\mathcal{N}_{ij}}$ für die benachbarten Terrainklassen von C_{ij} an der Position (i, j) und β_C ist ein Gewichtungsfaktor. Weiterhin beschreibt $\mathcal{N}_{ij}$ die Menge der Indizes, die auf die benachbarten Zellen der Grid-Map verweisen. Für die Annahme, dass sich die Bodenhöhe von Zelle zu Zelle nur geringfügig ändert, wird, wie beschrieben, ebenfalls ein Markov-Zufallsfeld verwendet. In diesem Fall basiert es auf der Gauß-Verteilung:

$$P(H^g_{ij} = h_{ij} \mid H_{N_{ij}}) \propto \exp\left(-\frac{1}{2\sigma_g^2}\left(h_{ij} - \frac{1}{|\mathcal{N}_{ij}|} \sum_{\{s,t\} \in \mathcal{N}_{ij}} h^g_{st}\right)^2\right) \tag{5.3}$$

Hierbei steht H^g_{ij} für eine Zufallsvariable, welche die Bodenhöhe an der Stelle (i, j) repräsentiert und einen Wert h_{ij} annehmen kann und σ_g repräsentiert die Standardabweichung der Bodenhöhe. Wellington et al. machen die Annahme, dass die Höhe H^c einer bestimmten Art von Vegetation nur leicht variiert. Diese Annahme wird wiederum mithilfe einer Gauß-Verteilung modelliert

$$P(H^c_{ij} = h_{ij} | H^c) \propto \exp\left(-\frac{1}{2\sigma^2_{H^c}}(h_{ij} - h^c)^2\right) \tag{5.4}$$

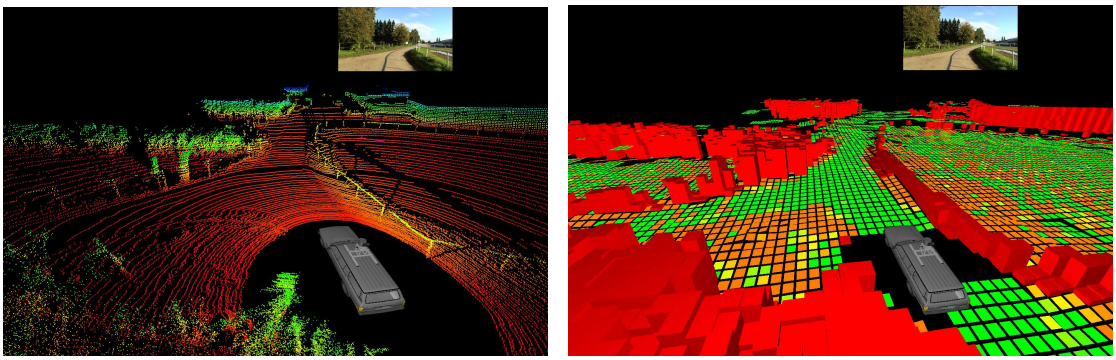
Bei dieser Gleichung bezeichnet H^c_{ij} die Zufallsvariable an der Stelle ij, die für die Höhe einer bestimmten Pflanzenart einen Wert h_{ij} annehmen kann und H^c eine Variable, welche die zu erwartende Höhe einer Vegetationsklasse beschreibt und einen Wert h^c annehmen kann. Die Standardabweichung der Höhe einer Terrainklasse ist durch σ_{H^c} beschrieben (siehe [WCS05]). Mit diesen einzelnen Modellen ist es Wellington et al. möglich, eine bedingte Wahrscheinlichkeit für das Gesamtproblem zu definieren

$$P(C_{ij}, H^g_{ij}, H^c_{ij} \mid Y_{ij}, C_{\mathcal{N}_{ij}}, H^g_{\mathcal{N}_{ij}}, H^c) \tag{5.5}$$

Aus der Maximierung dieser Wahrscheinlichkeit ergibt sich eine mögliche Lösung für die Klassifikation (siehe [WCS05]).

5.2 Terrainklassifikation des Mustang Mk 1A

Für das System Mustang Mk 1A wurde bereits ein Terrainklassifikationsverfahren von Neuhaus et al. [NDPP09] implementiert und beschrieben. Dieser Ansatz verwendet ausschließlich auf Laserscans basierende Merkmale und teilt das Terrain in



(a) **(b)**

Abbildung 5.2: Terrainklassifikation nach Neuhaus et al. [NDPP09]: Laserpunktewolke (a) und zugehörige Klassifikation mit Hindernissen (rot), sowie problemlos befahrbaren (grün) und bedingt befahrbaren rauen Zellen (gelb und orange) (b)

Zellen einer zweidimensionalen Grid-Map auf. Zu den Merkmalen zählen die Form der Punktewolke (siehe Abschnitt 4.1.1) und die Rauheit (siehe Abschnitt 4.1.2) der einzelnen Terrainzellen. Somit gelingt eine Differenzierung zwischen befahrbaren und nicht-befahrbaren Zellen, wobei die befahrbaren Zellen noch zusätzlich eine Befahrbarkeitsgüte in Form des Rauheitsmerkmals erhalten. In Abbildung 5.2 (a) ist die Punktewolke zu sehen, die dem Verfahren als Eingabe dient und in Abbildung 5.2 (b) das zugehörige Klassifikationsergebnis. Rote Säulen stehen für Hindernisse, während flache ebene Zellen als grün und unebene Zellen als gelb oder orange, je nach Rauheit, dargestellt werden (siehe [NDPP09]).

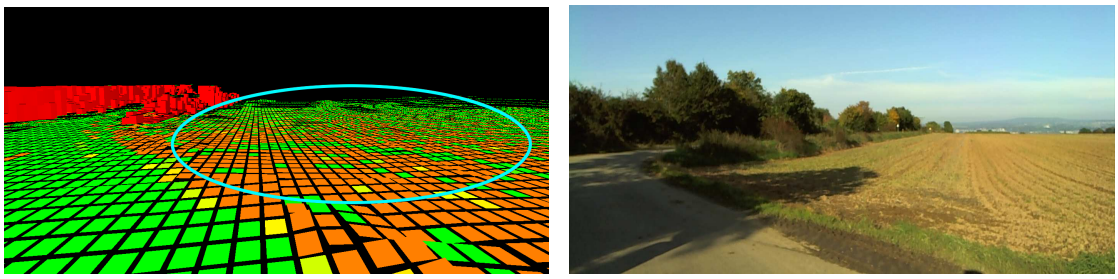
Kapitel 6

Umsetzung einer Terrainanalyse

Die vorangegangen Kapitel haben die Grundlagen der Markov-Zufallsfelder, die Sensorik der Roboter, die daraus resultierenden Terrainmerkmale, sowie bisherige Ansätze zur Terrainanalyse und Klassifikation vorgestellt. Dieses Kapitel beschreibt, wie anhand dieser Kenntnisse die Terrainklassifikation des Systems Mustang Mk 1A um weitere Merkmale und ein probabilistisches Modell erweitert wird. Es wird ein Verfahren beschrieben, dass ein Markov-Zufallsfeld in Kombination mit einem 3D-LIDAR verwenden soll und trotzdem noch in ausreichend geringer Laufzeit ausgeführt werden kann.

6.1 Problemstellung

Bei dem in Abschnitt 5.2 vorgestellten Verfahren kommt es, trotz grundsätzlich guter Ergebnisse, zu gelegentlichen Fehlern bei der Klassifikation einzelner Terrainelemente. So ist in Abbildung 6.1 zu sehen, dass inmitten eines Feldes mit sehr rauer Beschaffenheit, Zellen als sehr eben klassifiziert werden. Ein Grund dafür ist die geringe Datendichte an dieser Stelle, die dazu führt, dass keine oder nur eine geringe Anzahl von Laserpunkten in der Zelle liegen und somit eine genaue Aussage über die Rauheit erschwert wird. Ein weiteres Problem ist, dass flache asphaltierte Bereiche als grob eingestuft werden, wenn sie z. B. mit Laub bedeckt sind. Dies kann dazu führen, dass ein Pfadplanungsalgorithmus versucht, solche Bereiche zu umfahren, obwohl diese für den Roboter keinerlei Behinderung darstellen. Ein Umstand wie dieser führt nicht zwangsläufig zu einer Gefahrensituation, kann aber bei Anwendungen, bei denen das System eine Aufgabe mit einer Zeitobergrenze absolvieren muss, zum Scheitern führen.



(a) (b)

Abbildung 6.1: Fehlerhafte Terrainzellen in einem Feld, die grün dargestellt und somit nicht als rau erkannt werden (markierter Bereich) (a) und das Kamerabild des zugehörigen Terrains (b)

6.2 Ziel dieser Arbeit

Um die in Abschnitt 6.1 genannten Probleme zu beheben, soll das vorhandene Terrainklassifikationsverfahren, welches in Abschnitt 5.2 beschrieben ist, um zwei Punkte erweitert werden:

1. Einsatz eines Markov-Modells zur probabilistischen, kontextsensitiven Modellierung des Terrains
2. Fusion von Laser- und Bilddaten zur Gewinnung neuer Merkmale

Als Resultat sollen die Zellen mit einem Label gekennzeichnet sein, welches die Terrainklasse beschreibt. Zusätzlich zu einer möglichst korrekten Klassifikation soll der Anspruch der Echtzeitfähigkeit erfüllt werden. Es soll zur Realisierung der Ziele kein neues Software-Framework entstehen, sondern eine Integration in das bereits vorhandene System stattfinden, welches von Pellenz et al. in [PND+09] beschrieben wird.

6.3 Implementierung des Markov-Zufallsfeldes

Den ersten zu realisierenden Punkt dieser Arbeit stellt die Umsetzung des Markov-Zufallsfeldes dar. Hierzu muss ein mögliches Modell gefunden, die Klassen festgelegt und die verwendeten Merkmale definiert werden.

6.3.1 Terrainmodell

Die Aufteilung des Terrains unterscheidet sich geringfügig von der, die das Terrainklassifikationsverfahren verwendet, welches in Abschnitt 5.2 beschrieben ist. Dies

Label	Befahrbarkeit	Beschreibung
Unknown	k.A.	Terrainzellen ohne ausreichende Informationen
Street	uneingeschränkt	flaches Terrain (Straßen, Wege, usw.)
Rough	mäßig bis schwierig	grobes Terrain (Felder, Wiese, usw.)
Obstacle	unbefahrbar	Hindernisse (Bäume, Pfähle, usw.)

Tabelle 6.1: Klassen, die einer Terrainzelle zugewiesen werden können

bedeutet, dass die Umgebung in einzelne Zellen einer zweidimensionalen Grid-Map aufgeteilt wird. Eine Zelle ist 51.282cm × 51.282cm groß und die Größe des Grids ist auf 100m × 100m festgelegt. Da das Markov-Zufallsfeld-Modell zusätzliche Rechenzeit benötigt, wird das Grid auf 40m × 40m verkleinert, um eine gute Balance zwischen Aufwand und Informationsmenge zu erreichen. Das Terrain ist hierbei in einem lokalen Koordinatensystem definiert, mit dem Roboter als Bezugspunkt im Zentrum der Grid-Map. Jede Zelle im Terrain wird für die Verwendung eines Markov-Zufallsfeldes zusätzlich zu den Merkmalen durch eine Klasse beschrieben.

Die Klassen, die einer Zelle zugewiesen werden können, orientieren sich an den Fähigkeiten des Systems Mustang Mk 1A. Dieses ist dazu in der Lage, sich auch durch leicht unwegsames Gelände fortzubewegen, weshalb dieses nicht als Hindernis erkannt werden soll. Da ein Durchqueren von flachem Terrain allerdings zu bevorzugen ist, sollte dieses eine Klasse erhalten, die sich vom unwegsamen Gelände unterscheidet. Für Bereiche, die über keinerlei Informationen verfügen, wird eine zusätzliche Zurückweisungsklasse verwendet. Die Klassen die sich aus dieser Überlegung ergeben, sind in Tabelle 6.1 beschrieben. Daraus ergibt sich die Menge der Labels $\mathcal{L} = \{$*Unknown, Street, Rough, Obstacle*$\}$.

6.3.2 Auswahl der Lasermerkmale

Nach der Festlegung des Terrainmodells muss eine Auswahl an Merkmalen getroffen werden, die zur Klassifikation verwendet werden soll. Das erste Merkmal, welches verwendet wird, ist die Grobheit f_r (siehe Abschnitt 4.1.2), da sich dieses als gutes Merkmal in [NDPP09] erwiesen hat und eine Unterscheidung zwischen z. B. Straßen und Wiesen ermöglicht. Die Steigung zwischen zwei 3D-Punkten (siehe Abschnitt 4.1.3), wird in dem in dieser Arbeit vorgestellten Ansatz nicht verwendet. Dieser Ansatz eignet sich wegen zu vieler Punkte und Nachbarschaften nicht für Laserscans, wie sie vom *Velodyne HDL-64E S2* durchgeführt werden. Das Dichtemerkmal (siehe Abschnitt 4.1.4) ist ebenfalls schwer zu realisieren, da für jeden Laserstrahl berechnet werden muss, welche Zellen passiert werden und welche nicht. Diese Berechnung ist sehr aufwendig für die große Datenrate des 3D-LIDAR.

Die Reduktion der Datenrate sollte auch vermieden werden, da sich dies negativ auf andere Aufgaben, wie z. B. die Detektion und Verfolgung von dynamischen Hindernissen auswirken könnte. Die Verwendung der Remission (siehe Abschnitt 4.1.5) wird oft im Kontext der Erkennung von Vegetation verwendet, so z. B. in [WCS05] und [WKSB09]. Da allerdings die in Abschnitt 6.3.1 definierten Klassen *Rough* und *Obstacle*, solchen Zellen zugewiesen werden können, die unter Umständen, aber nicht zwangsweise Vegetation enthalten, wird von der Verwendung dieses Merkmals abgesehen. Dies könnte zu Komplikationen bei der Unterscheidung der beiden Klassen führen und somit auch zu einem nicht erkannten Hindernis. Die Verwendung der Höhendifferenz f_h, die in Abschnitt 4.1.6 beschrieben wird, eignet sich zur Detektion von Hindernissen, da sich diese oft dadurch definieren, dass sie zu hoch sind, um vom Roboter überfahren zu werden. Die ausgewählten Merkmale bilden einen Vektor $f_{features}$, der definiert ist als

$$f_{features} = (f_r, f_h)^T \tag{6.1}$$

und in einem Modell zur Terrainklassifikation verwendet werden kann.

6.3.3 Markov-Modell des Terrains

Bei der Findung eines passenden Markov-Modells für die Terrainklassifikation, muss das verwendete Terrainmodell betrachtet werden. Einzelne Laserscans mit einem Hidden Markov Model zu klassifizieren, wie beispielsweise in [WSFB05] beschrieben, ist bei der Verwendung des *Velodyne HDL-64E S2* weniger geeignet, da es sich hierbei um einen 3D-Laserscanner handelt, der einen großen Bereich um den Roboter mit einem Scan abdeckt. Die Repräsentation des Terrains geschieht durch Zellen auf einer zweidimensionalen Grid-Map, welche allerdings keine Voxel auf den einzelnen Zellen vorsieht, wie es z. B. in [WCS05] umgesetzt ist. Somit ist auch an dieser Stelle kein eindimensionales Markov-Modell zur Anwendung geeignet. Besser geeignet scheint hingegen die Verwendung eines Markov-Zufallsfeldes, da sich dieses gut für eine zweidimensionale Grid-Map umsetzen lässt. Das verwendete Markov-Zufallsfeld sollte nicht ausschließlich Nachbarschaftsverhältnisse berücksichtigen, wie z. B. das in [WSFB05] und [WS08] beschriebene Modell, sondern auch die extrahierten Merkmale verwenden. Von einer Verwendung von interagierenden Markov-Zufallsfeldern, wie es in [WCS05] beschrieben ist, wird abgesehen. Das zu verwendende Modell soll mit einem einzelnen Markov-Zufallsfeld die Klassifikation ermöglichen, um den Rechenaufwand zu verringern. Dies ist wichtig, da dieser Ansatz nicht mit einem aus mehreren 2D-Laserscans zusammengesetzten Scan der Umgebung arbeitet, sondern mit einem 3D-LIDAR. Diese Art des Laserentfernungsmesser erzeugt mit einem einzelnen Scan bereits hohe Mengen an Daten, die schnellstmöglich verarbeitet werden sollen.

Hierzu wird ein Modell verwendet, das in der Bildverarbeitung zur Segmentierung eingesetzt wird. Dieses Modell, wie es z. B.von Deng et al. [DC04] beschrieben wird, lässt sich auf das Terrain übertragen und ist wie folgt definiert

$$P(\Omega = \omega | F = f) = \frac{p(F = f | \Omega = \omega) P(\Omega = \omega)}{p(F = f)} \tag{6.2}$$

Hierbei ist $P(\Omega = \omega | F = f)$ die A Posteriori Wahrscheinlichkeit, dass das Terrain bestehend aus Zufallsvariablen Ω eine Konfiguration ω annimmt, in Abhängigkeit von den wahrgenommenen Merkmalen $F = f$. Durch die Äquivalenz von Markov-Zufallsfeldern und Gibbs-Zufallsfeldern (siehe Abschnitt 2.3.3) lässt sich, wie in [DC04] beschrieben, um die Wahrscheinlichkeit zu maximieren, die Energie minimieren, es gilt

$$\begin{aligned} \hat{\omega} &= \operatorname{argmax} P(\Omega = \omega | F = f) \\ &= \operatorname{argmax} \frac{1}{Z} \exp\left(-\frac{1}{T} E\right) \\ &= \operatorname{argmin} E \end{aligned} \tag{6.3}$$

Hierbei beschreibt E die berechnete Energie, T ist der Temperaturparameter der Gibbsverteilung und Z dient der Normierung (siehe auch 2.3.3). Unter dieser Annahme werden die Energiependants zu den Komponenten der A Posteriori Wahrscheinlichkeit modelliert (siehe [DC04]).

Die Komponente $p(F = f | \Omega = \omega)$ beschreibt dabei die Wahrscheinlichkeitsverteilung des Auftretens von Merkmalen zu einer bestimmten Klasse. Bei einem mehrdimensionalen Merkmalsvektor ergibt sich die Wahrscheinlichkeitsverteilung aus dem Produkt der einzelnen Verteilungen der k Merkmale

$$p(F = f | \Omega = \omega) = \prod_{k=1}^{K} \left(p(f_k | \Omega = \omega)\right) \tag{6.4}$$

In diesem Modell werden die Merkmale als gaußverteilt angenommen. Damit ergibt sich für die Verteilung der Merkmale an der Stelle (i, j) in der Grid-Map

$$p(f_{ij_k} | \Omega_{ij} = \omega_{ij}) = \frac{1}{\sqrt{2\pi\sigma_{ij_k}^2}} \exp\left(-\frac{(f_{ij_k} - \mu_{ij_k})^2}{2\sigma_{ij_k}^2}\right) \tag{6.5}$$

Hierbei sind μ_{ij_k} und σ_{ij_k} der Mittelwert und die Standardabweichung des k-ten Merkmals der Klasse ω_{ij}, die Ω_{ij} zugewiesen wurde und f_{ij_k} steht für das k-te Merkmal an der Stelle (i, j). Daraus ergibt sich die Möglichkeit zur Berechnung der Energie bezüglich der Merkmale aller Felder im Markov-Zufallsfeld

$$E_f = \sum_{ij} \left(\sum_{k=1}^{K} \left(\frac{(f_{ij_k} - \mu_{ij_k})^2}{2(\sigma_{ij_k})^2} + \log\left(\sqrt{2\pi}\sigma_{ij_k}\right) \right) \right) \tag{6.6}$$

Da das Markov-Zufallsfeld auch Nachbarschaftsbeziehungen modellieren soll, stellt Gleichung 6.6 nur einen Teil der zu berechnenden Energie dar (siehe [DC04]).

Die Wahrscheinlichkeit $P(\Omega = \omega)$ aus Gleichung 6.2 beschreibt die Kontextsensitivität der Terrainzellen. Um die Energie der Nachbarschaft zu berechnen, muss die Klasse einer Terrainzelle mit den Klassen der Zellen in der Nachbarschaft $\mathcal{N}_{ij}$ verglichen werden. Für einen paarweisen Vergleich der Zellen ergibt sich

$$E_n = \sum_{ij} \left(\beta \sum_{kl} \delta(\omega_{ij}, \omega_{kl}) \right) \text{ mit } \omega_{kl} \in \mathcal{N}_{ij} \tag{6.7}$$

Hierbei ist β ein Gewichtungsfaktor und $\delta(\cdot)$ eine Funktion, die als Ergebnis -1 liefert, wenn $\omega_{ij} = \omega_{kl}$ gilt und $+1$ liefert wenn $\omega_{ij} \neq \omega_{kl}$ gilt (siehe [DC04]).

Nur $p(F = f)$ wurde noch nicht als Energiekomponente modelliert. Dies ist auch nicht notwendig, da sich dieser Wert für unterschiedliche Ausprägungen von $\Omega = \omega$ nicht ändert und somit nicht relevant für die Berechnung der Energie ist. Für die gesamte Energie im Markov-Zufallsfeld ergibt sich

$$E = E_n + \alpha E_f \tag{6.8}$$

In Gleichung 6.8 ist α ein Gewichtungsfaktor, der verwendet werden kann, um zu steuern, wie groß der Einfluss der Merkmalskomponente E_f auf die Gesamtenergie ist (siehe [DC04]).

Das vorgestellte Modell wurde ausgewählt, da es sich schon in der Bildverarbeitung bewährt hat und sich das Segmentieren des Bildes auf das Labellingproblem im Bereich der Terrainklassifikation übertragen lässt. Das Modell bietet den Vorteil, dass es gleichzeitig die wahrgenommenen Merkmale berücksichtigt und auch die Klassen der Nachbarzellen in die Berechnung miteinbezieht. Somit ist das Modell in der Lage die Bedingung zu erfüllen, das Terrain probabilistisch und kontextsensitiv zu analysieren.

6.3.4 Implementierung einer Testsoftware

Um die Funktionsweise und Durchführbarkeit der Terrainklassifikation mit einem Markov-Zufallsfeld besser abschätzen zu können, wird zuerst eine Testsoftware implementiert. Hierbei wird das Markov-Zufallsfeld auf einem 5×5 Zellen großen Grid angewendet. Die verwendeten Merkmale, so wie die benötigten Erwartungswerte und Standardabweichungen sind fiktiv und werden lediglich im Rahmen der Testimplementierung verwendet.

Das Programm ist unter Verwendung des *Qt-Frameworks*[1] mit der Programmiersprache *C++* entstanden. Das dazugehörige Klassendiagramm ist in Abbildung 6.2 zu sehen. Die Klasse *Terrain* enthält eine zweidimensionale Datenstruk-

[1] Weitere Informationen unter http://qt.nokia.com/

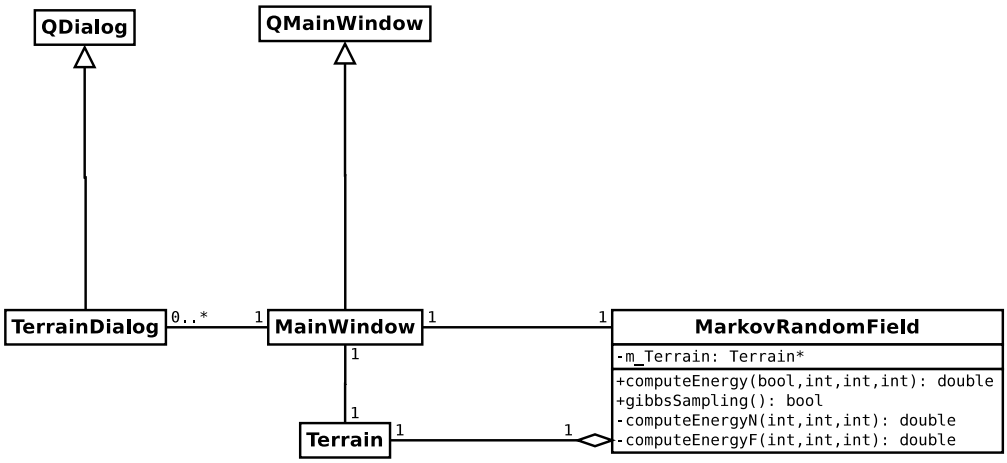


Abbildung 6.2: Klassendiagramm des Testprogramms mit erweiterter Darstellung der Klasse MarkovRandomField. Die Methoden *computeEnergyN* und *computeEnergyF* berechnen die einzelnen Energiekomponenten und werden von *computeEnergy* aufgerufen, welche einen Gesamtenergiewert zurückgibt. Die Methode *gibbsSampling* führt die Klassifikation des Terrains durch

tur, deren einzelne Elemente mit einer Terrainklasse und Merkmalswerten versehen werden können. Als Merkmale werden ein fiktiver Rauheitswert und ein Farbwert im *RGB*-Farbraum verwendet. Diese Eingaben können über eine *GUI*[2], die in Abbildung 6.3 zu sehen ist, getätigt werden. Die GUI ist als Klasse *MainWindow* implementiert, die sich von der Klasse *QMainWindow* aus dem *Qt-Framework* ableitet. Ein Dialog dient zur Visualisierung der Klasse *Terrain* und ist als Klasse *TerrainDialog* ableitend von *QDialog* umgesetzt. Die Klasse *MarkovRandomField* enthält die zu testende Funktionalität. Diese Klasse bietet Methoden zum Setzen der Parameter des Markov-Zufallsfeldes, auf welche die Klasse *MainWindow* Zugriff hat, um eine Konfiguration anhand der GUI zu ermöglichen. Der Benutzer kann die Mittelwerte für die einzelnen Merkmale für jede Klasse, ebenso wie die entsprechenden Standardabweichungen festlegen. Weiterhin kann zwischen einer Nachbarschaft 1. und 2. Ordnung (siehe Abschnitt 2.3.1) gewählt werden. In der Klasse *MarkovRandomField* ist zur Klassifikation ein Gibbs-Sampler, wie in Abschnitt 2.3.4 beschrieben, implementiert. Der beim Lösungsverfahren verwendete Temperaturparameter ist ebenfalls in der Benutzeroberfläche beeinflussbar. Der Nutzer kann den Startwert, den Wert, der zum Terminieren des Verfahrens führt, sowie den Faktor, der die Temperatur verringert, festlegen.

Teile der Berechnung der Energie und des Samplingverfahrens orientieren sich an der Beispielimplementierung[3] von Csaba Gradwohl und Zoltan Kato, deren

[2]engl.: Graphical User Interface

[3]Download unter http://www.inf.u-szeged.hu/ kato/software/mrfdemo.html

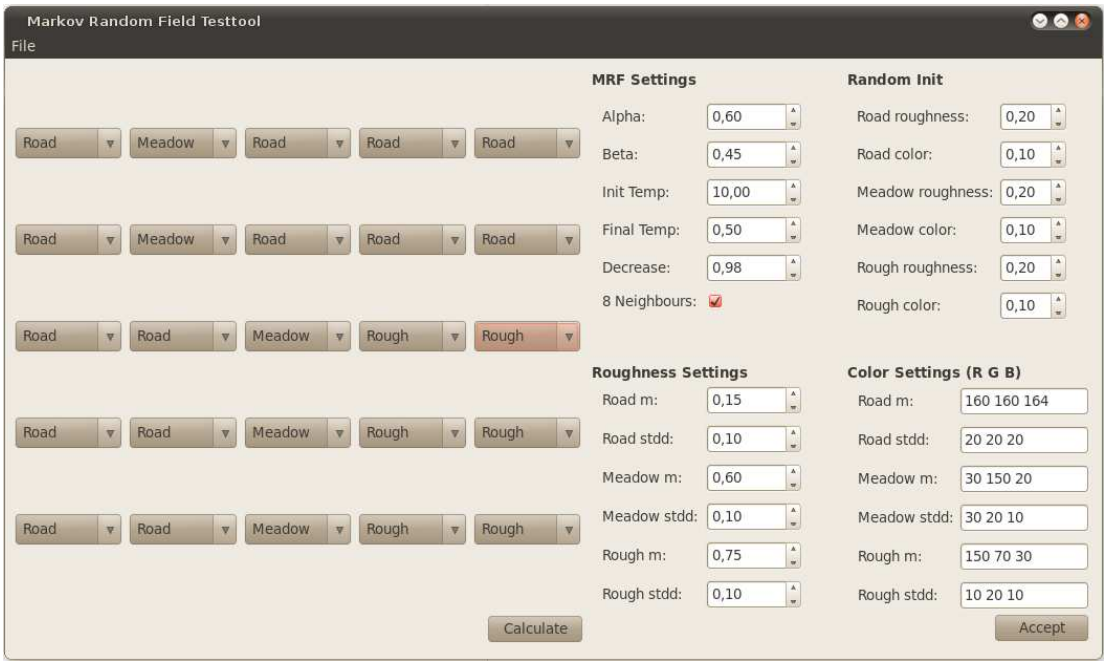


Abbildung 6.3: Graphische Benutzeroberfläche der Testimplementierung

theoretische Grundlagen in [BKYZ96], [KZB92] und [Kat94] beschrieben sind. In der Beispielimplementierung terminiert das Samplingverfahren, wenn die Änderung der Energie einen gewissen Schwellwert unterschreitet. Dies erfordert zusätzliche Rechenzeit, da hierfür nach jeder Relaxation die Energie des gesamten Terrains berechnet werden muss. Deshalb wird in der implementierten Testsoftware der Sampler verändert. Es wird ein initialer Temperaturparameter festgelegt, der nach jeder Relaxation durch Multiplikation mit einem Faktor verringert wird. Das Verfahren terminiert, wenn der Temperaturwert einen finalen Temperaturwert erreicht, bzw. diesen unterschreitet. So ist es auch möglich die Laufzeit des Verfahrens besser abzuschätzen, da somit eine vorher definierte Anzahl an Relaxationen durchgeführt wird. In Abbildung 6.4 ist der Ablauf des Samplingverfahrens dargestellt.

Um die Funktionsweise des Markov-Zufallsfeldes zu testen, wird in dem Programm ein Terrain eingestellt. Dabei kann zwischen drei verschiedenen Klassen gewählt werden. Jeder Zelle werden beim Auswählen einer Terrainklasse Merkmale zugewiesen. Diese Merkmale weichen dabei zufällig bis zu einem Maximum, welches der Anwender festlegen kann, vom Mittelwert des Merkmals der ausgewählten Klasse ab. Dies wird gemacht, um die Auswirkungen von verschieden stark abweichenden Merkmalswerten auf das Klassifikationsverfahren testen zu können. Das

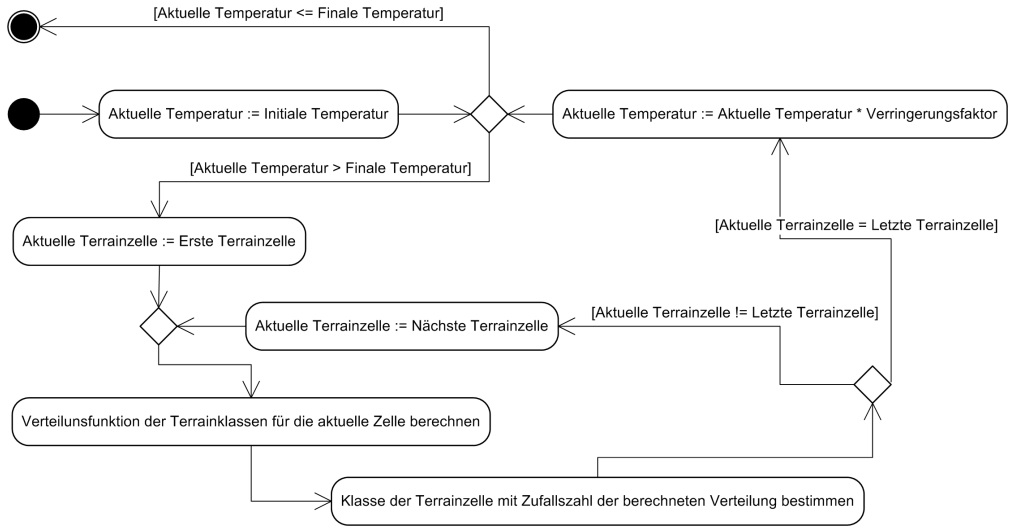


Abbildung 6.4: Darstellung des Ablaufs beim Gibbs-Sampling: Den einzelnen Terrainzellen werden durch Sampling neue Klassen zugeordnet und dieser Vorgang wird wiederholt, bis der Temperaturparameter unter einen Schwellwert sinkt

initiale Labelling des Terrains ist dabei zufällig. Nachdem alle Parameter ausgewählt sind, kann das Terrain klassifiziert werden.

Eine mit dem Testprogramm durchgeführte Klassifikation ist in Abbildung 6.5 zu sehen. Im Ergebnisbild wird eine festgelegte Farbe, zur besseren Visualisierung, abhängig von der erkannten Klasse zugeteilt. Eine Bedienungsanleitung des Programms befindet sich in Anhang B.1.

6.3.5 Erweiterung eines Terrainklassifikationsverfahrens mit einem Markov-Zufallsfeld

Das bereits vorhandene Software-Framework (siehe [PND+09]) und das darin realisierte Terrainklassifikationsverfahren, sollen durch ein Markov-Zufallsfeld-Modell erweitert werden. Die in Abschnitt 6.3.4 vorgestellte Klasse *MarkovRandomField* muss hierfür angepasst werden, sodass als Eingabe die Datenstruktur dient, die im vorhandenen Framework zur Repräsentation des Terrains verwendet wird. Die einzelnen Terrainzellen, die in dieser Datenstruktur abgelegt sind, werden um die Möglichkeit erweitert, eine Terrainklasse zugewiesen zu bekommen.

Das vorhandene Framework setzt sich aus verschiedenen *Modulen* zusammen, die sich auf verschiedene Aufgabengebiete aufteilen. Diese sind über einen Verteiler, hier *Dispatcher* genannt, in der Lage, miteinander über *Messages* zu kommunizieren. Ein *Modul* dient dabei nur zur Kommunikation, d. h. sollen tiefer gehende Aufgaben erfüllt werden, so werden *Worker* in die *Module* integriert. Abbildung 6.6

(a) **(b)**

Abbildung 6.5: Terrain, das als Eingabe dient (a) und das ausgegebene Klassifikationsergebnis (b) des Testprogramms

zeigt ein Klassendiagramm, dass den Zusammenhang der einzelnen Bestandteile und den Anschluss des Markov-Zufallsfeldes erläutern soll. Für den Aufgabenbereich der Terrainklassifikation ist bereits ein *Modul* und ein *Worker* vorhanden, weshalb lediglich eine Erweiterung des *Workers* notwendig ist. Hierbei wird der *Worker TerrainClassifier* um einen zusätzlichen *Worker* erweitert, der aus der Klasse *MarkovRandomField* entstanden ist.

6.3.6 Funktionsweise des erweiterten Terrainklassifikationsverfahrens

Eine Klassifikation beginnt, sobald das *TerrainClassificationModule* eine *Message* vom Typ *Velodyne3DPointsM* erhält. Diese Nachricht wird mit dem *Worker TerrainClassifier* weiterverarbeitet. Dieser führt das schon vorhandene Terrainklassifikationsverfahren (siehe Abschnitt 5.2) durch, welches die Umgebung in Zellen aufteilt. Danach werden für jede Zelle die zu verwendenden Merkmale Rauheit (siehe 4.1.2) und Höhendifferenz (siehe 4.1.6) berechnet. Dies geschieht, um eine genauere Aussage machen zu können, nicht für Zellen, die zu wenige 3D-Punkte enthalten. Anhand des Rauheitsmerkmals wird eine initiale Klassifikation der einzelnen Terrainzellen durchgeführt. Die vorabklassifizierte Grid-Map des Terrains wird

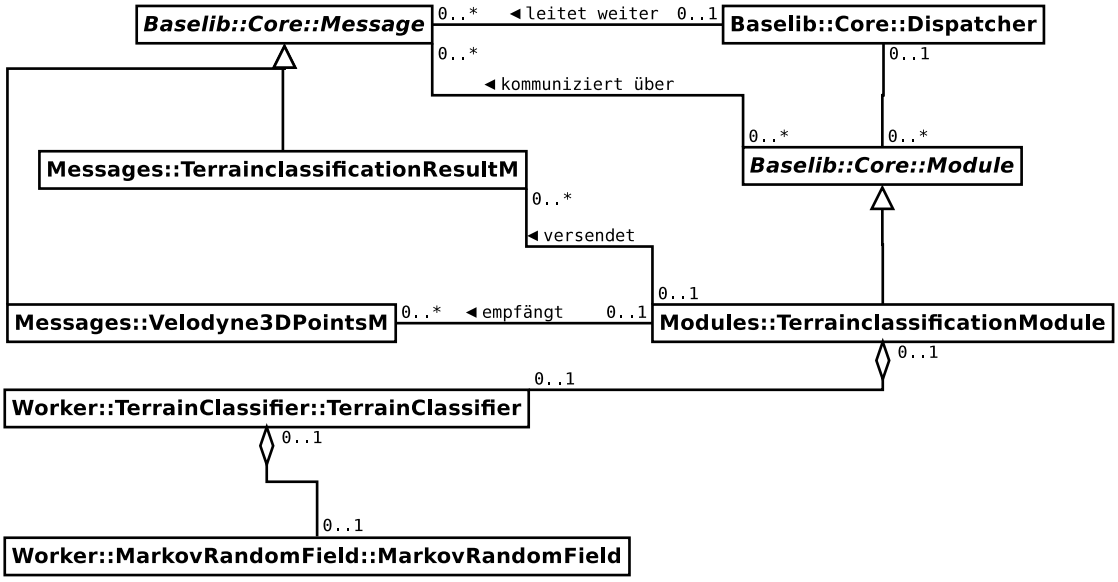


Abbildung 6.6: Klassendiagramm zur Integration des Markov-Zufallsfeldes, in die vorhandene Architektur

daraufhin von dem *Worker MarkovRandomField* weiterverarbeitet. Dieser bietet Methoden zur Berechnung der Energie im Markov-Zufallsfeld und eine Implementation des Gibbs-Samplers, mit dem eine Klassifikation des Terrains durchgeführt wird. Die Methode zur Energieberechnung wurde gegenüber dem Testprogramm um die Möglichkeit erweitert, auch mit Terrainzellen umgehen zu können, die keine Merkmale besitzen. Hier wird ein konstanter Energiewert zurückgegeben und so ist es möglich, diese Zellen alleine aufgrund ihrer Nachbarschaftsinformationen zu klassifizieren. Nach dem Samplingverfahren erstellt der *Worker TerrainClassifier* eine *TerrainClassificationResultM Message*, welche die Grid-Map enthält und übergibt diese als Rückgabewert an das *TerrainClassificationModule*. Das *Modul* versendet die Nachricht daraufhin, so dass diese von anderen *Modulen* in der Software verwendet werden kann. Der Ablauf mit den verschiedenen Zwischenschritten des Verfahrens ist in Abbildung 6.7 dargestellt. Ein Ergebnis des Verfahrens ist in Abbildung 6.8 zu sehen. In dieser ist Terrain der Klasse *Obstacle* rot, der Klasse *Rough* braun und der Klasse *Street* grau dargestellt. Die Zellen sind als Quader dargestellt, um die Höhendifferenz zu visualisieren. Eine Beschreibung, wie das implementierte Verfahren getestet werden kann findet sich in Anhang B.2.

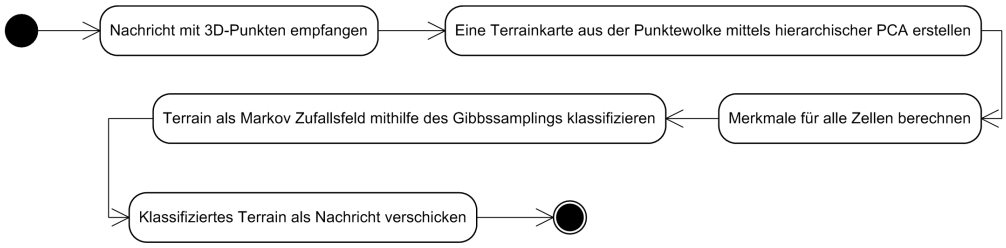
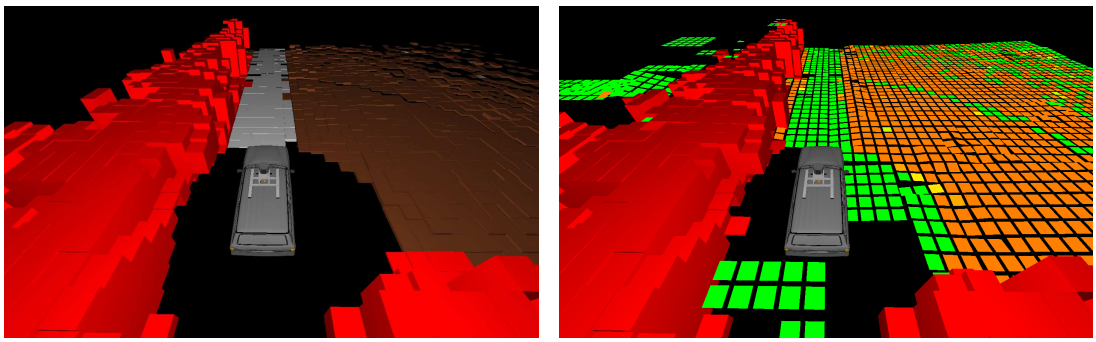


Abbildung 6.7: Ablauf des Terrainklassifikationsverfahrens mit einzelnen Schritten, von einer 3D-Punktewolke bis hin zur klassifizierten Terrainrepräsentation



(a) **(b)**

Abbildung 6.8: Ergebnis der Terrainklassifikation mit einem Markov-Zufallsfeld, mit rot dargestellten Zellen der Klasse *Obstacle*, braunen Zellen der Klasse *Rough* und grauen Terrainzellen der Klasse *Street* (a) und Ergebnis des alten Terrainklassifikationsverfahrens (siehe Abschnitt 5.2) (b)

6.4 Umsetzung der Datenfusion

Um weitere Merkmale für die Terrainklassifikation verwenden zu können, sollen zusätzlich zu den Laserdaten noch Bilddaten ausgewertet werden. Damit einer Terrainzelle aus Kamerabildern gewonnene Merkmale zugewiesen werden können, werden in dieser Arbeit Laser- und Kameradaten fusioniert. Mit dem Wissen welcher Teil eines Bildes welche Terrainzelle zeigt, ist es möglich, für diese Zelle Merkmale zu extrahieren, um diese besser beschreiben zu können.

6.4.1 Finden von korrespondierenden Terrain- und Bildbereichen

In dem verwendeten Software-Framework existieren bereits notwendige Funktionalitäten zur Datenfusion. Diese werden von dem in dieser Arbeit vorgestellten Verfahren verwendet und werden im folgenden Text erläutert. Es wird der *Worker Fusion* verwendet, der es ermöglicht, die durch den Laser ermittelten 3D-Punkte mit einem Bild von einer der drei Kameras zu fusionieren. Um die Laserdaten nun in Kamerakoordinaten umzurechnen, wird von einem *Scenegraph Worker*, der die Transformationen zwischen den Sensoren zur Verfügung stellt, die notwendige Umrechnungsmatrix $m_{LaserToCam}$ angefordert. Zusätzlich wird noch eine Matrix m_{inCam} verwendet, welche die intrinsischen Kameraparameter berücksichtigt. Die genauen Matrizen werden durch eine Kalibrierung ermittelt, die parallel zu dieser Arbeit in einer weiteren Qualifikationsarbeit entsteht. Somit sind die verwendeten Matrizen nocht nicht final. Ein Punkt $\boldsymbol{p}_{loc}$, der im lokalen Koordinatensystems des Roboters definiert ist, wird um eine homogene Koordinate erweitert und kann wie folgt zu einem Punkt $\boldsymbol{p}_{img}$ im Bildkoordinatensystem transformiert werden

$$\boldsymbol{p}_{img} = m_{inCam} \cdot m_{LaserToCam} \cdot \boldsymbol{p}_{loc} \tag{6.9}$$

Danach muss $\boldsymbol{p}_{img}$ noch durch seine homogene Koordinate dividiert werden. Die Bildkoordinaten werden daraufhin in Pixelkoordinaten umgerechnet mit

$$\boldsymbol{p}_{Pixel_x} = \lfloor \left(\left(\boldsymbol{p}_{img_x} + 1 \right) \cdot 0.5 \cdot res_x \right) + 0.5 \rfloor \tag{6.10}$$

$$\boldsymbol{p}_{Pixel_y} = \lfloor \left(\left(\boldsymbol{p}_{img_y} + 1 \right) \cdot 0.5 \cdot res_y \right) + 0.5 \rfloor \tag{6.11}$$

Hierbei stehen $\boldsymbol{p}_{Pixel_x}$ und $\boldsymbol{p}_{Pixel_y}$ für den x bzw. y-Anteil der Koordinaten des berechneten Pixelwertes. Analog stehen $\boldsymbol{p}_{img_x}$ und $\boldsymbol{p}_{img_y}$ für x und y-Koordinaten des Punktes im Bildkoordinatensystem. Die Werte res_x und res_y geben die Auflösung des Bildes in x und y-Richtung an. Nach der Berechnung wird getestet, ob die berechneten Pixelkoordinaten innerhalb der Bildgrenzen liegen. Ist dies der Fall, so werden die Korrespondenzen zwischen Laserpunkt und Pixel in einer *Lookup-Tabelle* eingetragen.

Um nun den Bildbereich zu finden, der für eine bestimmte Terrainzelle relevant ist, werden die Pixelkoordinaten der in der Zelle befindlichen Laserpunkte aus der Lookup-Table des *Fusion Workers* ausgelesen. Es wird die *Bounding-Box* um diese Pixelwerte berechnet, indem nach den maximalen, sowie minimalen x- und y-Koordinaten $\{x_{max}, y_{max}, x_{min}, y_{min}\}$ dieser Werte gesucht wird und daraus die zwei Eckpunkte $\boldsymbol{p_{min}} = \boldsymbol{(x_{min}, y_{min})^T}$ und $\boldsymbol{p_{max}} = \boldsymbol{(x_{max}, y_{max})^T}$ gebildet werden. Diese beiden Punkte reichen aus, um das Rechteck zu definieren, welches den Bildausschnitt darstellt, der für die Merkmalsgewinnung für eine Zelle betrachtet wird. Der betrachtete Ausschnitt gibt hierbei nicht exakt den ausschließlichen Inhalt einer Zelle wieder, sondern kann als Approximation verstanden werden.

6.4.2 Auswahl der Bildmerkmale

Wenn bekannt ist, welcher Bildausschnitt welcher Terrainzelle zugeordnet werden kann, ist es möglich Bildmerkmale, für die entsprechende Zelle, zu berechnen. Die Texturmerkmale, die in Abschnitt 4.2.1 beschrieben sind, werden zur Verwendung in Betracht gezogen. Die Merkmale, die von Haralick et al. [HDS73] vorgestellt werden, werden verwendet, weil diese verschiedene Aussagen über eine Textur ermöglichen. Dennoch ist hier eine Auswahl sinnvoll, da die Berechnung aller Merkmale viel Rechenzeit aufwendet, wie erste Tests gezeigt haben, und der Echtzeitanspruch der Software gewahrt bleiben soll. Ausgewählt werden die Merkmale *Angular Second Moment* f_{H1}, *Variance* f_{H4} und *Inverse Difference Moment* f_{H5}. Diese lassen sich zur Optimierung der Rechenzeit zusammen beim Iterieren über die Cooccurence Matrix berechnen, wie es in [MM11] vorgeschlagen wird. Das in [KM10] vorgestellte Homogenitätsmerkmal f_{FH} eignet sich ebenfalls zur Verwendung, da dieses sehr schnell zu berechnen ist, da für ein Bild nur einmal eine neue Datenstruktur, in Form einer Summed Area Table, erstellt werden muss. Dies steht im Gegensatz zu den Haralickmerkmalen, bei denen für jeden Bildbereich eine eigene Cooccurence Matrix berechnet werden muss. Das Merkmal Farbe f_{col} (siehe Abschnitt 4.2.2) wird in Form des Farbwinkels des HSV Farbraums verwendet, da hierbei die Farben einfach anhand eines Wertes unterscheidbar sind. Um den Farbwert einer Zelle zu berechnen wird der Mittelwert der Farbe aller zur Zelle gehörigen Pixel gebildet. Auf die Verwendung von Infrarotstrahlung, wie sie in Abschnitt 4.2.3 beschrieben ist, wird verzichtet, da eine Infrarotkamera noch nicht Teil des Pkw-Aufbaus (siehe 1.1) ist und eine Verwendung zur Erkennung von negativen Hindernissen beispielsweise von Tageszeit und Witterung abhängt, wie es in [RHM03] erklärt wird. Die ausgewählten Merkmale werden zu einem Vektor

$$f_{imgFeatures} = (f_{H1}, f_{H4}, f_{H5}, f_{FH}, f_{col})^T \tag{6.12}$$

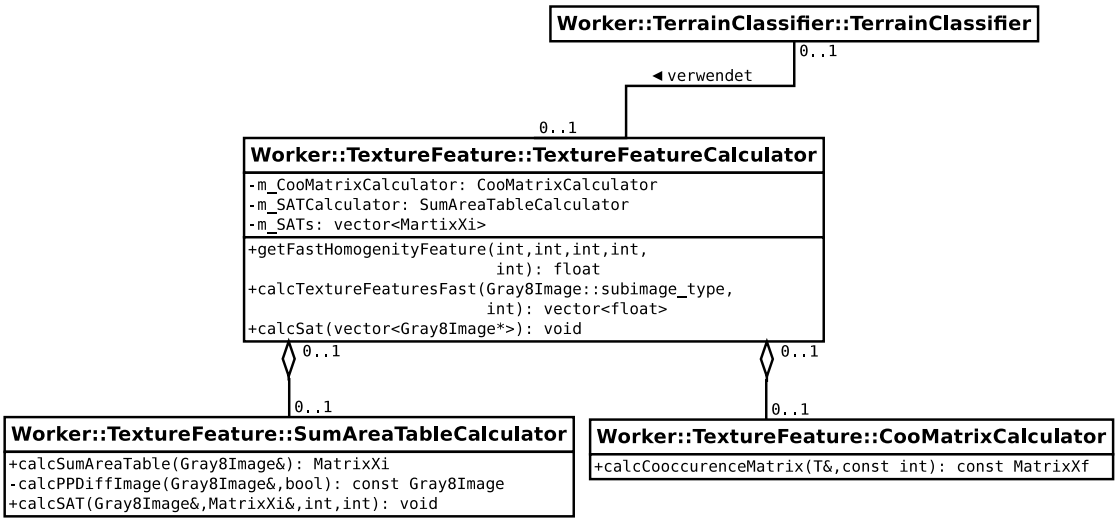


Abbildung 6.9: Klassendiagramm des *Workers TextureFeature* zum Berechnen von Texturmerkmalen, mit zugehörigen Klassen *TextureFeatureCalculator*, *CooMatrixCalculator* und *SumAreaTableCalculator*

zusammengefasst, der somit als Erweiterung des vorhandenen Terrainklassifikationsverfahrens dient und den schon vorhandenen Merkmalsvektor, siehe Formel 6.1, vergrößert.

6.4.3 Erweiterung der Software durch Bildmerkmalsberechnung

Das Verfahren, welches in Abschnitt 6.3.6 vorgestellt wurde, soll nun zusätzlich zu den Lasermerkmalen die Auswahl an Bildmerkmalen, die in Abschnitt 6.4.2 beschrieben ist, verwenden. Die Berechnung der Merkmale und die vorhergehende Fusion der Laser- und Bilddaten muss deshalb vor dem Sampling des Markov-Zufallsfeldes stattfinden. Hierzu wird ein *Worker TextureFeature* implementiert. Dieser setzt sich aus drei Klassen zusammen, die unterschiedliche Aufgaben übernehmen. Die Klasse *TextureFeatureCalculator* dient hierbei zur Berechnung der Texturmerkmale. Da diese Merkmale nicht direkt aus einem Bild berechnet werden können, sondern entweder eine Cooccurence Matrix oder eine Summed Area Table benötigen, werden Klassen verwendet, die ein Bild in die richtige Form konvertieren. Diese sind *CooMatrixCalculator* und *SumAreaTableCalculator*. Der Zusammenhang dieser Klassen zum *Worker TerrainClassifier* ist in Abbildung 6.9 in einem Klassendiagramm veranschaulicht.

6.4.4 Funktionsweise der Bildmerkmalsgewinnung

Die Berechnung der Bildmerkmale wird von dem *Worker TerrainClassifier* eingeleitet. Das *TerrainClassificationModule* empfängt die hierfür notwendigen Kamerabilder als Nachricht und übergibt diese an die Klasse *TerrainClassifier*. Diese verarbeitet die Bilder nicht augenblicklich weiter, die Weiterverarbeitung geschieht erst innerhalb der Methode, welche die *Message Velodyne3DPointsM* auswertet. So wird gewährleistet, dass die Bilder und der Laserscan zeitlich möglichst nah beieinander liegen, da Laserscans mit einer geringeren Frequenz als die Kamerabilder eintreffen. In der verarbeitenden Methode wird zuerst jedes Farbbild in ein Grauwertbild konvertiert und aus diesen jeweils eine Summed Area Table berechnet. Hierfür werden die Bilder an die Klasse *TextureFeatureCalculator* übergeben, welche diese Bilder wiederum an die Klasse *SumAreaTableCalculator* übergibt. Diese bildet die benötigten Differenzbilder und berechnet die Summed Area Tables, wie es in Abschnitt 4.2.1 beschrieben ist und gibt diese wieder zurück. Nach der Datenfusion, wie sie in Abschnitt 6.4.1 beschrieben ist, werden die Merkmale für die einzelnen Terrainzellen berechnet. Das Homogenitätsmerkmal f_{FH} lässt sich dabei schnell aufgrund der vorhandenen Summed Area Tables ermitteln, im Gegensatz dazu benötigen die weiteren Texturmerkmale eine vorherige Berechnung der Cooccurence Matrix eines Bildausschnittes. Hierfür werden von *TextureFeatureCalculator* Methoden der Klasse *CooMatrixCalculator* verwendet, um daraufhin die ausgewählten Merkmale aus der Cooccurence Matrix zu berechnen. Das Farbmerkmal f_{col} wird in der Klasse *TerrainClassifier* direkt aus den Ausschnitten der vorhandenen Farbbilder ermittelt.

Nachdem die Bildmerkmale berechnet wurden, lässt sich das Terrain, wie in Abschnitt 6.3.6 beschrieben, klassifizieren. Dabei fließen für die Zellen, die in den Kamerabildern zu sehen sind, die zusätzlichen Merkmale mit in das Verfahren ein.

6.5 Lernen der Terraineigenschaften

In Abschnitt 6.3.3 ist zu sehen, dass das verwendete Markov-Zufallsfeld-Modell für die Terrainklassifikation, Mittelwerte und Standardabweichungen der Merkmale für alle Terrainklassen benötigt. Diese müssen vor der Verwendung geschätzt werden. Um Werte schätzen zu können, wird die Software um die Möglichkeit erweitert, Terrain mit den verschiedenen Klassen zu annotieren und die entsprechenden Mittelwerte und Standardabweichungen zu berechnen.

6.5.1 Umsetzung eines Annotationsmodus

Der Benutzer soll die Möglichkeit erhalten, Terrainzellen von Hand zu annotieren. Es ist notwendig, dass für das zu annotierende Terrain die Merkmale bereits berech-

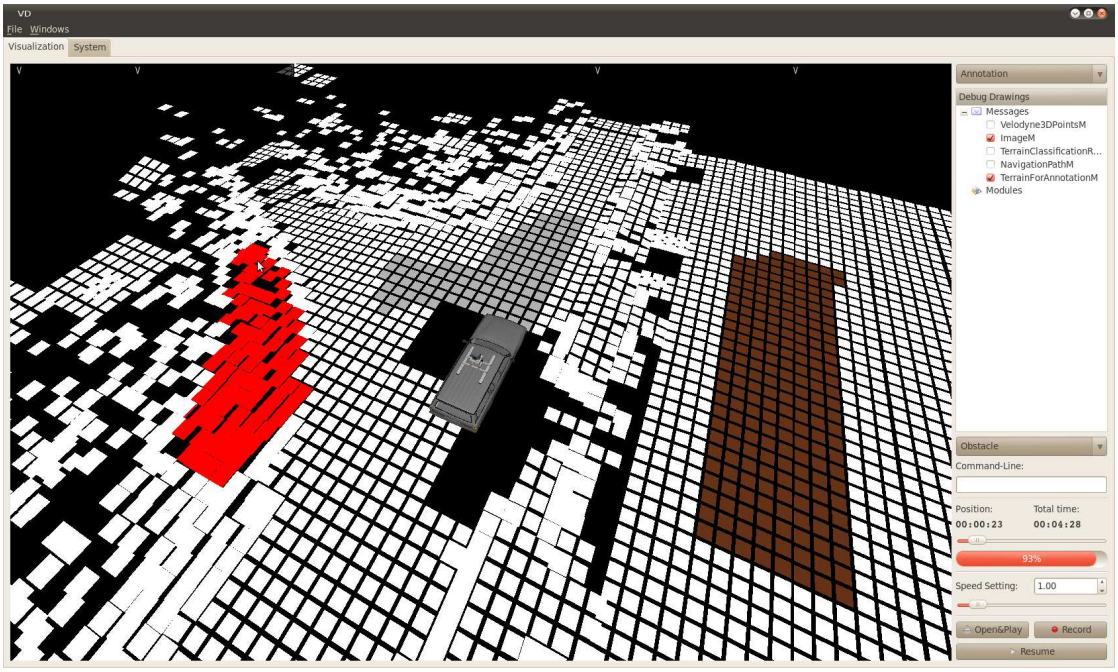


Abbildung 6.10: Annotationsmodus zum Lernen der Terrainmerkmale

net wurden. Deshalb wird eine neue *Message TerrainForAnnotationM* eingeführt, die vom *Worker TerrainClassifier* erstellt und dann von dem *Modul TerrainClassificationModule* versendet wird. Jede Zelle enthält dabei die für sie berechneten Merkmale und ist initial als *Unknown* klassifiziert. Das als *Message* versendete Terrain wird in der GUI, die in dem vorhandenen Framework bereits implementiert ist, visualisiert und kann dort vom Benutzer annotiert werden, wie es in Abbildung 6.10 zu sehen ist. Der Anwender kann eine Klasse auswählen und diese per Mausklick einer Zelle zuweisen. Das Auswählen einer Zelle ist dabei mit der Methode *gluProject* aus der *OpenGL*[4] Graphikbibliothek realisiert. Mit dieser werden die Koordinaten der Zellenmittelpunkte in der 3D-Ansicht in Bildschirmkoordinaten umgerechnet und bei einem Mausklick die Zelle ausgewählt, deren Mittelpunkt in Bildschirmkoordinaten den geringsten Abstand zum Mauszeiger hat.

Aus den markierten Zellen ist es möglich, eine Schätzung für Mittelwerte und Standardabweichungen für die verschiedenen Merkmale für jede Klasse zu berechnen. Die Verwendung des Annotationsmodus ist in Anhang B.3 genauer beschrieben.

[4]Für nähere Informationen siehe http://www.opengl.org

6.6 Konfigurierbarkeit

Das eingesetzte Software-Framework bietet die Möglichkeit, in einer externen *XML*[5]-Datei Konfigurationsparameter festzulegen. Diese Möglichkeit wird von dem in dieser Arbeit vorgestellten Verfahren ebenfalls genutzt. So ist es möglich die Verwendung des Markov-Zufallsfeldes aus- und anzuschalten, ebenso wie die Verwendung der drei Arten von Bildmerkmalen. Bei ausgeschaltetem Markov-Zufallsfeld kommt das schon vorher vorhandene Terrainklassifikationsverfahren (siehe Abschnitt 5.2) zur Anwendung.

Neben dem Ein- und Ausschalten von Funktionalitäten dient die Konfigurationsdatei allerdings auch dazu, die benötigten Parameterwerte zur Verfügung zu stellen. So lassen sich die initiale Temperatur, die Endtemperatur, sowie der Faktor zum Verringern der Temperatur eingeben, welche für das in Abschnitt 2.3.4 beschriebene Annäherungsverfahren benötigt werden. Ebenso ist es möglich die Mittelwerte und Standardabweichungen der einzelnen Merkmale für die verschiedenen Terrainklassen festzulegen.

Mit den verwendeten Konfigurationsmöglichkeiten lässt sich das Markov-Zufallsfeld für verschiedene Situationen anpassen und eine schnelle Änderung der Merkmalsparameter ist möglich. Somit ist bei Experimenten keine Änderung des implementierten Programmcodes notwendig. Eine Übersicht aller konfigurierbaren Parameter ist in Anhang C zu sehen.

[5]Extensible Markup Language

Kapitel 7

Experimente und Evaluation

In diesem Kapitel werden Experimente und deren Evaluation beschrieben, die zur Auswertung der Verwendbarkeit des vorgestellten Verfahrens durchgeführt werden. Die Ergebnisse sollen dabei Aufschluss über die Eignung eines Markov-Zufallsfeldes im Bereich der Terrainanalyse, unter Verwendung eines 3D-LIDAR, liefern. Ebenso sollen Erkenntnisse über die Eignung von fusionierten Texturdaten gewonnen werden und ob diese einen Vorteil gegenüber einer, auf ausschließlich Lasermerkmalen basierenden Klassifikation bieten. Zusätzlich wird die benötigte Rechenzeit betrachtet, um eine Aussage über die Echtzeitfähigkeit des Verfahrens zu machen.

7.1 Durchführung der Evaluation

Um Ergebnisse über die Güte des Terrainklassifikationsverfahrens mit Markov-Zufallsfeldern zu erhalten, wird zuerst eine Möglichkeit im Programm geschaffen, die es ermöglicht, ein Klassifikationsergebnis mit der korrekten Klassifikation zu vergleichen. Die korrekte Klassifikation muss dabei von einem Benutzer erstellt werden, der anhand der in der GUI visualisierten Sensor- und Kameradaten eine Entscheidung bezüglich der Klassen der Terrainzellen trifft. Hierzu wird der Annotationsmodus erweitert, der in Abschnitt 6.5.1 beschrieben ist. Um einen Vergleich durchzuführen, kann der Benutzer die Terrainzellen, welche die berechneten Merkmale zu einem bestimmten Zeitpunkt enthalten, annotieren und das annotierte Terrain danach als Datei abspeichern. So kann eine Annotation wiederverwendet werden, um die Auswirkungen verschiedener Konfigurationen des Verfahrens auszuwerten. Hierbei werden nur Zellen mit einer der drei Klassen versehen, bei denen der menschliche Betrachter eine klare Aussage machen kann. Schwer zu bestimmenden Zellen wird die Zurückweisungsklasse Unknown zugeordnet und diese werden in der Auswertung des Verfahrens nicht berücksichtigt.

Für die Evaluation werden Logfiles verwendet, da diese die Daten in der gleichen zeitlichen Abfolge wiedergeben, wie sie von den Sensoren ursprünglich aufgezeichnet wurden. So ist gewährleistet, dass die Funktionsweise des Verfahrens bei realer Verwendung mit einem Roboter vergleichbar ist.

Das Schätzen von Mittelwerten und Standardabweichungen (siehe Abschnitt 6.5.1) geschieht aus den Daten verschiedener Logfiles, verschiedener Szenarien. Allerdings werden nur die für die Lasermerkmale berechneten Werte für alle Logfiles verwendet. Die Bildmerkmale sind stark von der Umgebung und der Witterung abhängig, da sich diese mit variierender Umgebung und unterschiedlichen Lichtverhältnissen ändern können. Diese Merkmale werden für jedes zu evaluierende Szenario separat berechnet und verwendet.

Die Auswertung erfolgt für verschiedene Konfigurationen der Verwendung von Merkmalen für die einzelnen Terrainklassen. Es werden vier Kombinationen, aus denen in den Abschnitten 6.3.2 und 6.4.2 ausgewählten Merkmalen, gebildet. Zuerst werden die Lasermerkmale ohne zusätzliche Bildmerkmale verwendet, danach Lasermerkmale in Kombination mit den drei ausgewählten Haralickmerkmalen, daraufhin die Lasermerkmale in Kombination mit dem Homogenitätsmerkmal und abschließend Lasermerkmale in Verbindung mit der Farbe. Bei der Evaluation werden für die verschiedenen Konfigurationen die gleichen Szenen betrachtet, um eine korrekte Vergleichbarkeit zu gewährleisten. Betrachtet werden vier verschiedene Szenarien. Diese teilen sich auf in Feldwege, Waldgebiet, Wohngebiet und breite Straßen.

Da eine Kalibrierung, die in einer parallel zu dieser Arbeit entstehenden Qualifikationsarbeit realisiert wird, noch nicht für die frontale Kamera des Systems Mustang Mk 1A vorliegt, werden in den Experimenten ausschließlich die Seitenkameras verwendet. Die Kalibrierung dieser ist aufgrund der automatischen Fokussierung der Kameras nur eine Annäherung, was zu einer Einschränkung der Genauigkeit der Fusion führt.

Um Ergebnisse über die Laufzeit des Verfahrens zu erhalten, werden 100 aufeinanderfolgende Klassifikationen durchgeführt und für diese die Zeit gemessen. Dies ermöglicht den Vergleich der verschiedenen Konfigurationen.

Die Auswertungen in diesem Kapitel wurden mit einem initialen Temperaturparameter $T_{init} = 10.0$, finalen Temperaturparameter $T_{final} = 0.5$ und einem Verringerungsfaktor $F_{TempD} = 0.8$ durchgeführt, da sich diese Werte als gut funktionierend herausgestellt haben, um ein Gleichgewicht zwischen Güte und Geschwindigkeit zu schaffen.

7.1.1 Testszenarien

Bei Feldwegen ist die Terrainklasse Rough dominierend, da große Flächen in diesem Terrain landwirtschaftlich genutzt werden und eine sehr raue Oberfläche besitzen.

Die befahrenen Wege sind von den Feldern klar anhand ihrer Oberflächenbeschaffenheit differenzierbar und werden als Klasse Street annotiert. Als Hindernisse kommen in diesem Szenario hauptsächlich kleinere Bäume und Sträucher vor und werden als Klasse Obstacle annotiert.

Waldgebiete zeichnen sich durch ein hohes Aufkommen der Klasse Obstacle aus, da die dichten Wälder für das System Mustang Mk 1A nicht passierbar sind. Die Klasse Rough ist nur in kleinen Bereichen vertreten und dies meist als Randstreifen zwischen Straße und Wald. Die befahrene Straße ist wiederum klar von der restlichen Umgebung differenzierbar und häufig vertreten.

In Wohngebieten muss der Roboter dazu in der Lage sein, durch enge Straßen mit parkenden Autos zu navigieren. Neben abgestellten Pkws kommen häufig Häuserwände als Hindernis vor. In diesen Bereichen finden sich hauptsächlich die Klassen Obstacle und Street. Die Klasse Rough ist wenig vertreten und kommt lediglich in Form von begrünten Flächen in Gärten oder kleineren Parks vor.

Breite Straßen bilden das letzte untersuchte Szenario. Hier dominieren große asphaltierte Flächen, denen die Klasse Street zuzuordnen ist. Die Straßen werden in den meisten Fällen klar durch Hindernisse begrenzt. Raue Flächen, wie etwa Wiesen oder Felder bilden in diesem Szenario eine Ausnahmeerscheinung.

7.1.2 Darstellung der Ergebnisse

Für jedes Szenario werden vier einzelnen Szenen von Hand annotiert und mit dem von der Software berechneten Ergebnis verglichen. Die vier Testszenen werden hierbei zu einem Ergebnis zusammengefasst, als wäre eine größere Szene klassifiziert worden. Zur Auswertung wird für jede der verschiedenen Terrainklassen, für alle Konfiguration des Verfahrens, die Richtig-Positiv-Rate $Rate_{TP}$, auch Sensitivität genannt und die Falsch-Positiv-Rate $Rate_{FP}$ berechnet. Die Berechnung geschieht wie folgt

$$Rate_{TP} = \frac{Cells_{TP}}{Cells_{TP} + Cells_{FN}} \tag{7.1}$$

$$Rate_{FP} = \frac{Cells_{FP}}{Cells_{FP} + Cells_{TN}} \tag{7.2}$$

Hierbei ist $Cells_{TP}$ die Anzahl der richtig-positiv, $Cells_{FP}$ die Anzahl der falsch-positiv, $Cells_{FN}$ die Anzahl der falsch-negativ und $Cells_{TN}$ die Anzahl der richtig-negativ klassifizierten Zellen.

Um diese Werte zu visualisieren werden ROC[1] Diagramme verwendet. In diesen Diagrammen stellt die x-Achse die False-Positiv-Rate und die y-Achse die Sensitivität dar. Je weiter ein Punkt sich in der linken oberen Ecke des Diagramms

[1] engl.: Receiver Operating Characteristic

befindet, desto besser ist das Ergebnis. Daraus ergibt sich, dass ein Ergebnis umso schlechter ist, je mehr sich der Eintrag der rechten unteren Ecken annähert.

Die Auswertung der Laufzeittests erfolgt über die Berechnung des Mittelwertes, der Standardabweichung, des Maximal- und des Minimalwertes der Zeit, die eine Softwarekonfiguration für Klassifikationsvorgänge benötigt.

7.2 Ergebnisse

Die Ergebnisse der Evaluation werden für jede der drei möglichen Terrainklassen einzeln dargestellt, da deren Vorkommen für die verschiedenen Szenarien unterschiedlich ist. In jedem Diagramm werden die Ergebnisse der vier verschiedenen Konfigurationen visualisiert. Zu beachten ist, dass die Diagramme unterschiedliche Wertebereiche visualisieren, um den besseren Vergleich der verschiedenen Konfigurationen zu ermöglichen.

Die Ergebnisse der Laufzeittests werden in einer Tabelle dargestellt, um einen Vergleich der einzelnen Konfigurationen zu ermöglichen.

7.2.1 Ergebnisse der Detektion der Klasse Street

In Abbildung 7.1 sind die Ergebnisse für die Klasse Street in den verschiedenen Umgebungen dargestellt. Es ist deutlich zu sehen, dass die zusätzliche Verwendung von Bildmerkmalen nur zu geringen Unterschieden in der Qualität der Klassifikationsergebnisse führt. Teilweise ist dies darauf zurückzuführen, dass meist nur geringe Teile der Straße in den verwendeten Seitenkameras zu sehen sind, dennoch zeigt Abbildung 7.1 (d), dass der Unterschied auch bei großen sichtbaren Straßenbereichen nur gering wahrnehmbar ist. Allgemein ist jedoch zu sehen, dass die Klasse Street gut von dem Klassifikationsverfahren erkannt wird, bei einer durchschnittlichen Sensitivität von über 90% und einer Falsch-Positiv-Rate von unter 10%.

7.2.2 Ergebnisse der Detektion der Klasse Rough

Die Ergebnisse der Erkennung der Klasse Rough sind in den ROC Diagrammen in Abbildung 7.2 dargestellt. Im Vergleich zu den Ergebnissen der Klasse Street (siehe Abschnitt 7.2.1) ist zu sehen, dass das Klassifikationsverfahren bei der Erkennung von rauem Terrain wesentlich schlechtere Ergebnisse liefert. Hier ist allerdings ebenfalls festzustellen, dass bis auf im Stadt Szenario (siehe Abbildung 7.2 (c)), keine großen Unterschiede bei der zusätzlichen Verwendung von Bildmerkmalen zu erkennen ist. Die besonders schlechten Werte, die in Abbildung 7.2 (b) zu sehen sind, lassen sich durch das geringe Aufkommen der Klasse Rough in Waldgebie-

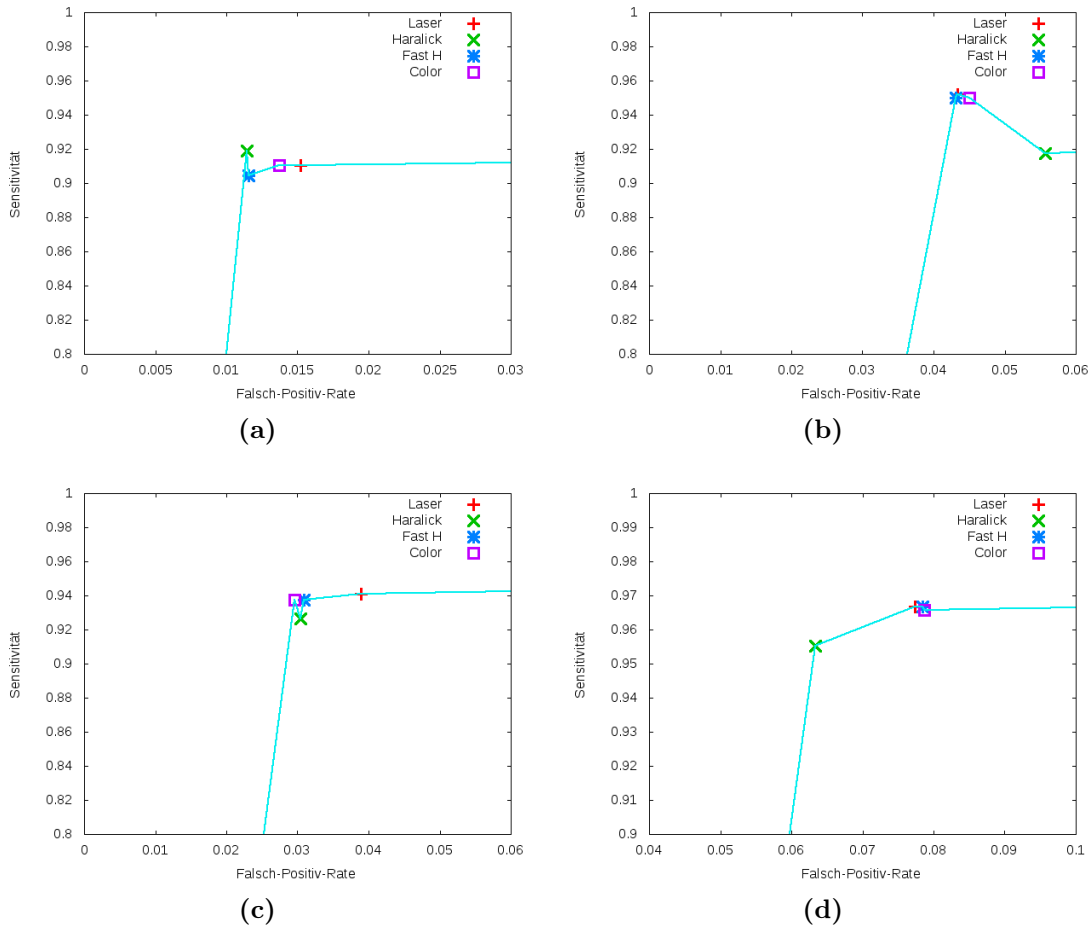


Abbildung 7.1: ROC Diagramm für die Erkennung der Klasse Street in einem Feldwegszenario (a), in einem Wald Szenario (b), in einem Stadt Szenario (c) und in einem Straßen Szenario (d). Laser steht für die Konfiguration, die ausschließlich Lasermerkmale verwendet, Haralick steht für die Konfiguration, die Laser- und Haralickmerkmale verwendet, FastH steht für die Konfiguration, die Lasermerkmale und das Homogenitätsmerkmal verwendet und Color steht für die Konfiguratio, die Farbwinkel und Lasermerkmale verwendet

ten erklären. Da das verwendete Markov-Zufallsfeld die Annahme modelliert, dass sich gleiche Terrainklassen gruppieren, werden die einzelnen auftretenden rauen Terrainzellen einer anderen Klasse zugeordnet. Diese Zellen kommen, wie bereits erwähnt, meist nur in Form eines kleinen Streifens zwischen Wald und Straße vor und werden somit häufig als Klasse Obstacle oder Street interpretiert. Weitere Probleme können bei vom System weit entfernten Terrainzellen auftreten. Diese werden häufig nur von wenigen Laserstrahlen getroffen, für welche die Wahrscheinlichkeit auf einer Ebene zu liegen höher ist, als für viele Laserstrahlen. Dies führt häufig zu einer Erkennung als Zellen der Klasse Street. Die Bildmerkmale liefern hierfür ebenfalls ungenaue Daten, da weit entfernte Zellen nur einen sehr kleinen Bereich im Kamerabild einnehmen, dessen Textur nicht mehr aussagekräftig ist.

7.2.3 Ergebniss der Detektion der Klasse Obstacle

Die in Abbildung 7.3 dargestellten Ergebnisse zur Erkennung der Klasse Obstacle zeigen, dass diese ebenso gute Ergebnisse liefert, wie die Erkennung der Klasse Street (siehe Abschnitt 7.2.1). Es ist ebenfalls zu sehen, dass auch hier eine zusätzliche Verwendung von Bildmerkmalen nicht zu deutlichen Änderungen der Ergebnisse führt. Lediglich in Abbildung 7.3 (d) ist zu sehen, dass der Einsatz von Haralickmerkmalen im Feld Szenario die Erkennungsergebnisse verschlechtert. Dies lässt sich auf eventuelle Ähnlichkeiten der Texturen der Klasse Rough und Obstacle zurückführen. Dies rührt daher, dass beide Klassen diverse Formen von Vegetation beinhalten können und eine Differenzierung durch ausschließlich geometrische Eigenschaften effektiver wäre.

7.2.4 Ergebnisse der Laufzeittests

Tabelle 7.1 zeigt die verschiedenen Mittelwerte und Standardabweichung, sowie Maxima und Minima der Laufzeiten für verschiedene Konfigurationen des Terrainklassifikationsverfahrens. Dargestellt ist neben der Zeit, welche die verschiedenen beschriebenen Kombinationen von Merkmalen benötigen, auch die Laufzeit des Klassifikationsverfahrens, das in Abschnitt 5.2 beschrieben ist. Die Auswertung erfolgte jeweils auf einem Computer mit dem 4-Kern Prozessor *Intel Core i7 Q820M* mit 1.73 GHz Taktfrequenz. Da zwischen zwei Laserscans des *Velodyne HDL-64E S2* ca. 66.6ms liegen, ist es aufgrund von Abweichungen der Rechenzeit nicht möglich, die Echtzeitfähigkeit bei Verwendung des Markov-Zufallsfeldes zu garantieren.

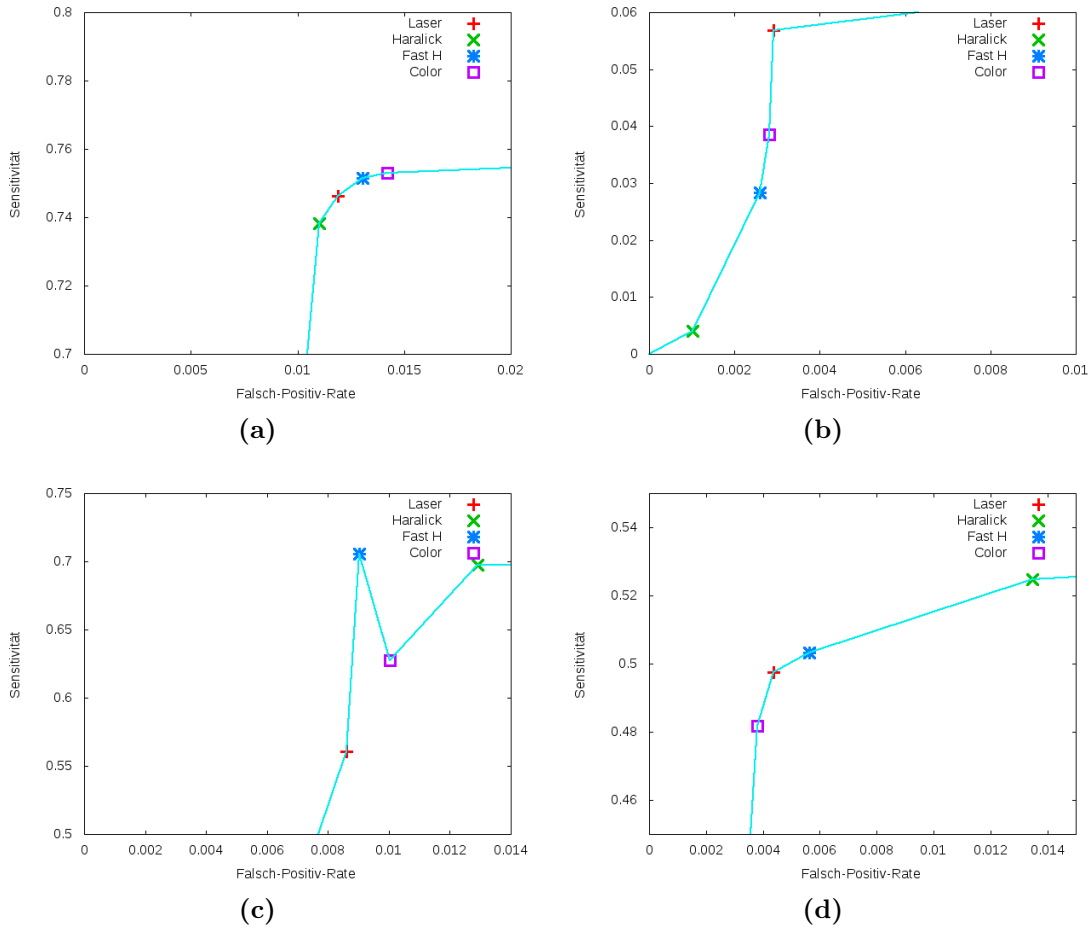


Abbildung 7.2: ROC Diagramm für die Erkennung der Klasse Rough in einem Feldwegszenario (a), in einem Wald Szenario (b), in einem Stadt Szenario (c) und in einem Straßen Szenario (d). Laser steht für die Konfiguration, die ausschließlich Lasermerkmale verwendet, Haralick steht für die Konfiguration, die Laser- und Haralickmerkmale verwendet, FastH steht für die Konfiguration, die Lasermerkmale und das Homogenitätsmerkmal verwendet und Color steht für die Konfiguratio, die Farbwinkel und Lasermerkmale verwendet

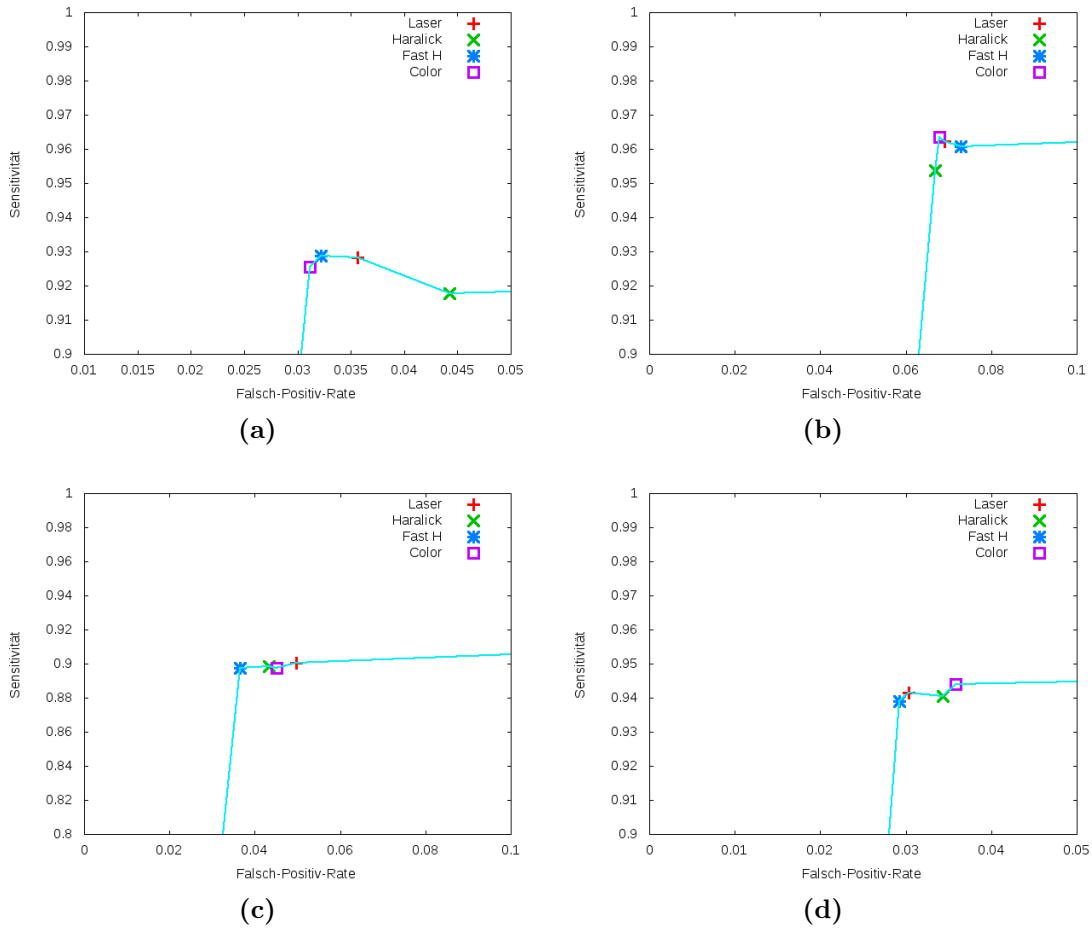


Abbildung 7.3: ROC Diagramm für die Erkennung der Klasse Obstacle in einem Feldwegszenario (a), in einem Wald Szenario (b), in einem Stadt Szenario (c) und in einem Straßen Szenario (d). Laser steht für die Konfiguration, die ausschließlich Lasermerkmale verwendet, Haralick steht für die Konfiguration, die Laser- und Haralickmerkmale verwendet, FastH steht für die Konfiguration, die Lasermerkmale und das Homogenitätsmerkmal verwendet und Color steht für die Konfiguratio, die Farbwinkel und Lasermerkmale verwendet

Verfahren	Mittelwert	Standardabw.	Maximum	Minimum
PCA	9.83169ms	1.75729ms	16.145ms	5.871ms
L (MRF)	62.6674ms	7.35659ms	82.24ms	48.787ms
L+H (MRF)	744.917ms	131.507ms	993.987ms	421.465ms
L+FH (MRF)	141.379ms	11.2331ms	188.42ms	121.733ms
L+C (MRF)	144.439ms	12.5663ms	187.013ms	117.326ms

Tabelle 7.1: Laufzeiten der verschiedenen Verfahren: PCA beschreibt das von Neuhaus et al. [NDPP09] implementierte Verfahren, (MRF) bedeutet, dass bei dem jeweiligen Verfahren das Markov-Zufallsfeld verwendete wurde. L, H, FH und C stehen für die Verwendung der Lasermerkmale, der drei Haralickmerkmale, des Homogenitätsmerkmals sowie des Farbwinkels

7.2.5 Diskussion der Ergebnisse

Die Ergebnisse zeigen, dass das vorgestellte Klassifikationsverfahren, das ausschließlich Lasermerkmale verwendet, durchaus für den Einsatz auf einen mobilen autonomen Systems geeignet ist. Auch wenn die Echtzeitfähigkeit nicht gegeben ist, da es vorkommen kann, dass die Laserscans schneller zur Verfügung gestellt werden, als das Verfahren diese analysieren kann, ist es doch dazu in der Lage, ausreichend schnell ein Ergebnis zur Verfügung zu stellen. Dies begründet sich darauf, dass sich ein autonomes mobiles System wie der Mustang Mk 1A vergleichsweise langsam fortbewegt und da nur der aktuellste Scan ausgewertet wird, vergrößert sich die Verzögerung auch nicht mit der Zeit.

Die Verwendung von Bildmerkmalen hat, in der vorgestellten Umsetzung, zu keinen signifikanten Änderungen der Klassifikationsergebnisse geführt. In Anbetracht der erhöhten Laufzeiten, speziell bei den Haralickmerkmalen, scheint eine Verwendung nicht sinnvoll. Der geringe Einfluss der Bildmerkmale lässt sich teilweise dadurch erklären, dass sich die verschiedenen Formen des Terrains auf vielfache Weise, selbst in eingeschränkten Szenarien, manifestieren können. So können Straßen je nach Asphaltart und Verschmutzungsgrad eine homogen oder eine inhomogen Textur haben. In der Stadt kann ein Hindernis eine weiße Wand, aber auch ein rotes Fahrzeug sein. Durch diese Vielfalt entstehen hohe Standardabweichungen der Merkmale, was wiederum dazu führt, dass sich der berechnete Energiewert für verschiedene Klassen kaum unterscheidet und somit die Lasermerkmale ausschlaggebend für die Klassifikation sind.

Kapitel 8

Fazit

Die Klassifikation von Terrain ist im Outdoor-Bereich für ein autonom agierendes mobiles System, wie es der Mustang Mk 1A darstellt, unerlässlich. Zur Lösung des Terrainklassifikationsverfahrens existieren verschiedene Herangehensweisen und Methoden. Die vorliegende Arbeit hat solche Methoden besprochen und ein Verfahren vorgestellt und evaluiert. Dieses Kapitel fasst das beschriebene zusammen und gibt einen Ausblick über mögliche Erweiterungen und Verbesserungen.

8.1 Zusammenfassung

In dieser Arbeit wurde das Problem der Terrainklassifikation im Outdoor-Bereich für autonome mobile Systeme betrachtet. Ziel war es, ein bereits implementiertes Verfahren zu erweitern, um eine kontextsensitive Betrachtung von Terrain zu ermöglichen. Die Kontextsensitivität sollte durch die Verwendung von Markov-Modellen geschehen.

Um Terrain analysieren zu können, ist es vorerst notwendig, Informationen über die Umgebung zu sammeln. Das Verfahren sollte für den Roboter Mustang Mk 1A umgesetzt werden, weshalb nur Sensoren zur Auswahl standen, die zum Zeitpunkt der Erstellung dieser Arbeit bereits auf dem System montiert und einsetzbar waren. Diese Arbeit beschreibt zunächst die verfügbare Sensorik und die Art der Daten, die diese ermitteln.

Anhand der verfügbaren Sensoren wurde die Möglichkeit untersucht, Merkmale aus den Daten der Sensorik zu berechnen, welche eine Aussage über die Beschaffenheit der Umgebung machen. Hauptaugenmerk wurde dabei auf Merkmale gelegt, die durch Laserentfernungsmessungen berechnet werden können. Laserentfernungsmessungen wurden in dieser Arbeit als besonders geeignet betrachtet, da diese sich im Bereich der Robotik etabliert haben und es somit bereits eine große Auswahl

an möglichen Merkmalen gibt. Weiterhin basiert das zu erweiternde Terrainklassifikationsverfahren auf Lasermerkmalen, die somit weiterverwendet werden können. Die Verwendung des 3D-Laserscanners des Mustang Mk 1A ermöglicht darüber hinaus eine schnelle flächendeckende Erfassung der Umgebung. Zusätzlich zu Lasermerkmalen, wird in dieser Arbeit auch ein Augenmerk auf die Möglichkeit der Verwendung von Bilddaten gelegt. Motiviert wird dies durch das oftmals unterschiedliche Erscheinungsbild der verschiedenen Terrainkategorien.

Nach der Erörterung verschiedener Merkmale werden verschiedene Klassifikationsverfahren beschrieben, die bereits erfolgreich Markov-Modelle eingesetzt haben. Diese Veröffentlichungen dienten als Motivation dafür, Markov-Modelle auch in dem in dieser Arbeit entwickelten Verfahren einzusetzen.

Der zweite Teil der Arbeit beschreibt, wie das vorhandene Klassifikationsverfahren um ein Markov-Zufallsfeld-Modell erweitert wird. Dabei wird eine Auswahl der zu verwendenden Merkmale getroffen und analysiert, inwiefern Markov-Modelle aus anderen Verfahren einsetzbar sind.

Vor der Umsetzung eines Verfahrens innerhalb eines bestehenden Frameworks wurde eine Testsoftware implementiert. Diese hatte zur Aufgabe, die grundsätzliche Funktionalität des Markov-Zufallsfeldes zu überprüfen. Hierfür wurde ein Modell, das zur Segmentierung in der Bildverarbeitung verwendet wird, an das Terrainklassifikationsproblem angepasst. Das Markov-Zufallsfeld verwendet dabei eine zweidimensionalen Repräsentation des Terrains, welches in einzelne zu klassifizierende Zellen aufgeteilt wird. Das verwendete Modell berücksichtigt dabei sowohl die berechneten Merkmale einer Terrainzelle, als auch die Klassen, welche den Nachbarzellen zugewiesen wurden. Nach der Beschreibung des Testprogramms wird erläutert, wie die Integration des implementierten Markov-Zufallsfeldes in ein vorhandenes Software-Framework erfolgt. Hierbei wurden zwei Lasermerkmale zur Verwendung ausgewählt, die bereits effektiv von dem vorhandenen Software-Framework berechnet werden. Um ein verbessertes Klassifikationsergebnis zu erreichen, wurde zusätzlich noch die Verwendung von Bildmerkmalen umgesetzt. In der Arbeit wurde erläutert, wie es in dem vorhandenen Framework möglich ist, Laser- und Bilddaten zu fusionieren und somit Bildausschnitte für einzelne Terrainzellen zu berechnen. Nach der Fusion war es möglich, Klassen in die Software zu integrieren, die für einzelne Zellen Bildmerkmale berechnen können, die vom Markov-Zufallsfeld bei der Klassifikation berücksichtigt werden können. Da es notwendig ist diverse Parameter für die einzelnen Merkmale zu ermitteln, wurde ein Annotationmodus umgesetzt, der es dem Benutzer ermöglicht, einzelne Terrainzellen von Hand zu klassifizieren und sich die entsprechenden Parameter berechnen zu lassen.

Der letzte Teil der Arbeit beschäftigte sich mit der Auswertung der Ergebnisse des Klassifikationsverfahrens. Hierzu wurden verschiedene Terrainszenen von

Hand annotiert und mit dem Ergebnis des Klassifikationsverfahrens verglichen. Die Ergebnisse zeigten, dass die Klassifikation mit einem Markov-Zufallsfeld grundsätzlich gute Ergebnisse liefert. Schwächen des Verfahrens liegen prinzipiell im Bereich der Berechnungsgeschwindigkeit. Die Geschwindigkeitstests zeigten, dass Laserscans teilweise verworfen werden müssen, da diese schneller zur Verfügung gestellt werden, als das Verfahren ein Klassifikationsergebnis berechnen kann. Dennoch ist das Verfahren schnell genug, um von einem langsam fahrenden Roboter eingesetzt werden zu können. Die zusätzliche Verwendung von Bildmerkmalen zeigte in den Tests nur geringe Unterschiede in der Qualität der Ergebnisse, führte aber zu erhöhten Berechnungszeiten. Somit ist es nicht ratsam, die Bildmerkmale in ihrer aktuellen Form miteinzubeziehen, sondern weiterhin auf die Lasermerkmale, die eine bessere Beschreibung der Geländegeometrie bieten, zurückzugreifen.

8.2 Ausblick

Das in dieser Arbeit vorgestellte Verfahren stellt einen ersten Entwurf dar, wie die Verwendung eines Markov-Zufallsfeldes das im Software-Framework des Roboters Mustang Mk 1A vorhandene Terrainklassifikationsverfahren, erweitern kann. Folglich bieten sich noch an diversen Stellen, Möglichkeiten zur Optimierung.

Bei weiterführenden Arbeiten bezüglich des Klassifikationsverfahrens, sollte das Augenmerk auf die Verbesserung der Rechenzeit gelegt werden. Eine Möglichkeit das Verfahren zu beschleunigen, besteht darin, das eingesetzte Samplingverfahren zu verbessern. Durch die heutzutage weite Verbreitung von Mehrkernprozessoren ist es möglich, den Samplingprozess zu parallelisieren, wie es bereits von Geman et al. [GG84] für Mehrprozessorsysteme vorgeschlagen wurde.

Um die Klassifikationsergebnisse zu verbessern, besteht die Möglichkeit, die notwendigen Parameter, der einzelnen Merkmale automatisch in der Software zu berechnen. Dies kann ermöglichen, dass sich die Parameter automatisch an sich langsam änderndes Terrain anpassen, während der Roboter dieses durchfährt.

Eine weitere Möglichkeit zur Verbesserung der Klassifikation könnte durch die Anpassung des Verfahrens zur Bildmerkmalsgewinnung erreicht werden. Durch die große Entfernung mancher Terrainzellen zur Kamera sind diese nur auf kleinen Bereichen im Bild zu sehen. Dies führt zu Texturmerkmalen, die keine genaue Aussage über die Klasse der Zelle ermöglichen. Sinnvoller wäre eine Vorabsegmentierung des Bildes anhand der Homogenität der Textur in verschiedene Regionen, wie es von Knauer et al. [KM10] beschrieben ist. Berechnet man nun Merkmale für die Textur einer ganzen Region und weist diese allen zu der Region gehörigen Zellen zu, umgeht man das Problem der weit entfernten Zellen.

Das Markov-Zufallsfeld-Modell, welches in der vorliegenden Arbeit verwendet wird, hat sich als funktional erwiesen. Dennoch sollte untersucht werden, ob es möglich ist, ein besser zu den Terraineigenschaften passendes Modell zu finden.

Anhang A

Installation der Software

Dieser Abschnitt dient zur Erklärung, wie die in dieser Arbeit beschriebene Software installiert werden kann.

A.1 Installation der Testsoftware

Die Testsoftware wurde unter Linux entwickelt und wurde nicht für andere Betriebssysteme getestet. Diese Installationsanleitung bezieht sich deshalb ausschließlich auf eine Verwendung unter Linux. Für die Installation muss das *Qt-Framework* installiert sein, welches über das Paketverwaltungsprogramm der Distribution oder unter `http://qt.nokia.com/downloads` (Stand 03/2011) erhältlich ist. Weiterhin müssen ebenfalls die Programme `make`, sowie `qmake` installiert sein, welche ebenfalls über die Paketverwaltungssoftware der Linux-Distribution erhältlich sein sollten. Die Installation erfolgt über die Kommandozeile, in dem Ordner, indem sich der Programmordner `MarkovRandomField` befindet. Folgende Befehle sind nacheinander einzugeben:

```
cd MarkovRandomField
qmake
make
```

Danach lässt sich die Software ausführen.

A.2 Installation des Roboter Frameworks

Das Framework, in welches das in dieser Arbeit beschriebene Verfahren integriert wurde, verfügt über ein Installationsskript für Ubuntu Linux. Das Skript installiert bei Ausführung alle benötigten Bibliotheken. Folgende Befehle sind vom Benutzer

auf der Kommandozeile im Ordner `trunk` auszuführen, um das Installationsskript zu starten

```
sudo chmod +x install.sh
./install.sh
```

Alternativ ist in `trunk/install.txt` beschrieben, welche Bibliotheken installiert werden müssen. Danach muss das Framework kompiliert werden, hierzu sind folgende Befehle auszuführen

```
cd mainapp
qmake
make
```

Danach lässt sich das Framework ausführen.

Anhang B

Verwendung der Software

Dieser Abschnitt beschreibt, wie die in dieser Arbeit entwickelten Verfahren in der Software verwendet werden können.

B.1 Verwendung des Testprogramms

Das Testprogramm lässt sich nach dessen Installation, im Installationsordner durch den Befehl

```
./MarkovRandomField
```

starten. Daraufhin erscheint die in Abbildung B.1 zu sehende Benutzeroberfläche. Um nun die Klassifikation mit einem Markov-Zufallsfeld zu testen, kann der Benutzer mehrere Parameter einstellen. Dafür muss dieser wie folgt vorgehen

1. Eintragen der Mittelwerte und Standardabweichungen der verschiedenen Merkmale für alle Terrainklassen (siehe Abbildung B.1 blauer Bereich)

2. Einstellen, um welchen Faktor die Merkmale beim Initialisieren des Terrains von ihrem Mittelwert abweichen können (siehe Abbildung B.1 gelber Bereich)

3. Einstellen der Parameter des Markov-Zufallsfeldes (siehe Abbildung B.1 roter Bereich)

4. Die Einstellungen mit dem *Accept*-Button bestätigen

5. Die Klasse der einzelnen Terrainzellen einstellen (siehe Abbildung B.1 grüner Bereich)

6. Mit drücken des *Calculate*-Buttons die Klassifikation starten

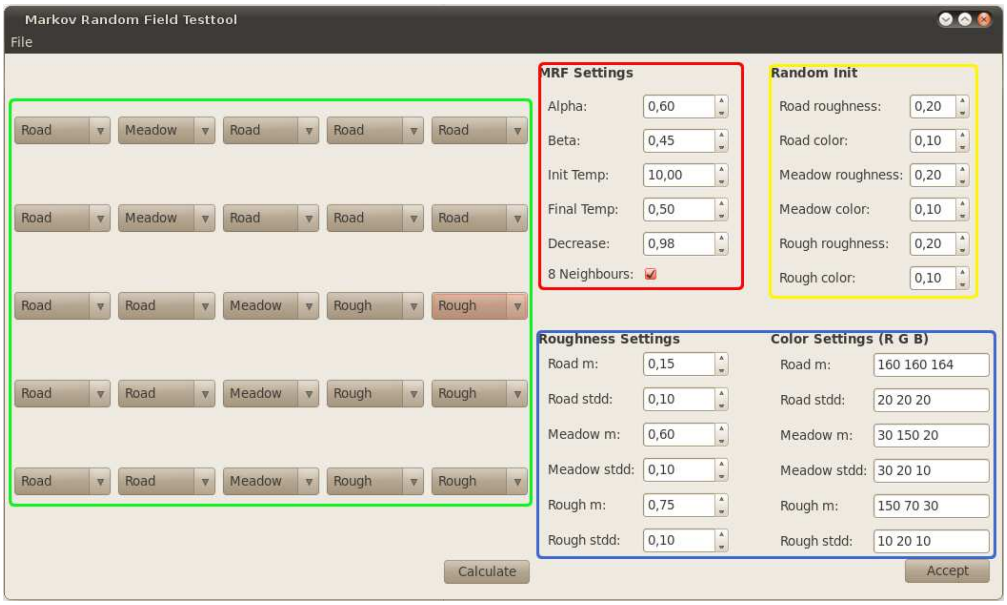


Abbildung B.1: Benutzeroberfläche der Testsoftware mit einstellbarem Terrain (grüner Bereich), Einstellungen des Markov-Zufallsfeldes (roter Bereich), Initialisierungsparametern (gelber Bereich) und einstellbaren Merkmalsparametern (blauer Bereich)

Als Ergebnis werden dem Benutzer zwei Fenster angezeigt. Ein Fenster enthält das initialisierte und das andere Fenster das klassifizierte Terrain. Wenn das klassifizierte Terrain dem vom Benutzer eingestelltem Terrain entspricht, war die Klassifikation erfolgreich.

B.2 Verwendung der Framework Erweiterung

Das Klassifikationsverfahren mit Markov-Zufallsfeld kann in dem Roboter-Framework mithilfe eines dafür vorgesehenen Profils getestet werden. Vor der Ausführung kann der Benutzer die verschiedenen Konfigurationsparameter in der Datei `Config.xml` bestimmen. Diese befindet sich im Ordner `mainapp/Config` und eine Liste aller für das Markov-Zufallsfeld relevanten, einstellbaren Werte ist in Anhang C aufgeführt. Das Framework kann aus dem Ordner `mainapp` mit dem Befehl

```
./mainapp mrf
```

gestartet werden. Der Benutzer kann daraufhin ein Logfile starten, indem der *Open&Play*-Button gedrückt wird und im folgenden Dialog eine entsprechende Datei ausgewählt wird. Danach beginnt die Software, aufgenommene Sensordaten abzuspielen. Die Klassifikation anhand des Markov-Zufallsfeldes kann betrachtet werden, indem im *Debug Drawings*-Bereich (siehe Abbildung B.2 roter Bereich)

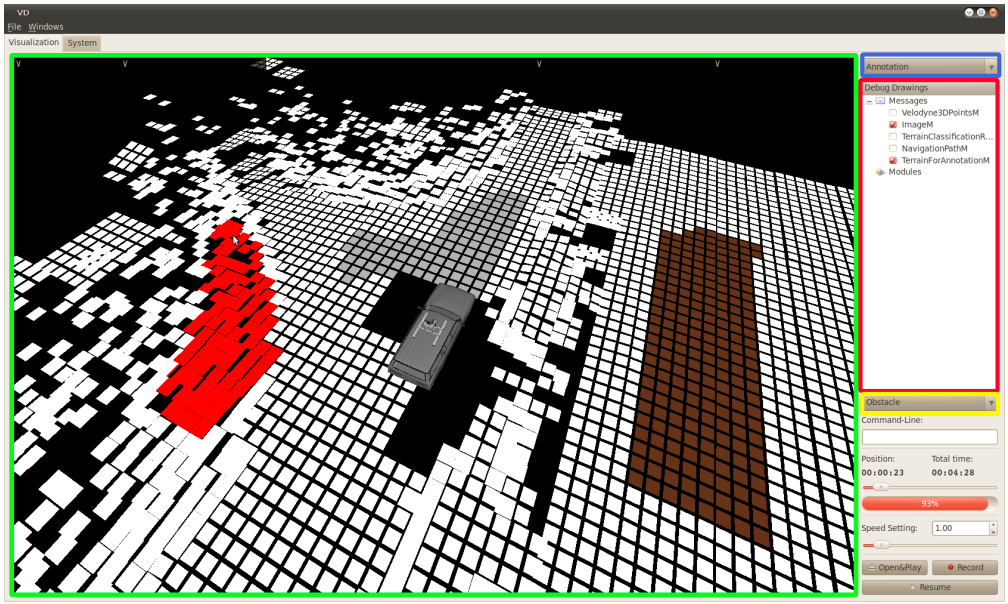


Abbildung B.2: Benutzeroberfläche des Frameworks im Annotationsmodus mit auswählbarer Darstellung (blauer Bereich), Debug-Drawings Bereich (roter Bereich), Auswahlmöglichkeit der Terrainklasse (gelber Bereich) und Visualisierungsbereich (grüner Bereich)

der Benutzeroberfläche, *TerrainClassificationM* aktiviert wird. Die Terrainklassifikation wird nun visualisiert. Durch einen Klick auf diese Visualisierung (siehe Abbildung B.2 grüner Bereich) und drücken der `m`-Taste kann zwischen altem und neuem Klassifikationsverfahren umgeschaltet werden. Hierbei ist zu beachten, dass nicht zwischen den Verfahren, sondern lediglich zwischen der Visualisierung umgeschaltet wird, da beide Verfahren aktiv sind. Dies begründet sich damit, dass das alte Verfahren die Eingabedaten für das Markov-Zufallsfeld-Verfahren liefert.

B.3 Verwendung des Annotationsmodus

Der Benutzer hat die Möglichkeit, Terrain von Hand zu annotieren. Hierfür müssen folgende Schritte durchgeführt werden

1. Durch einen Klick auf den *Pause*-Button, das Abspielen der Sensordaten anhalten

2. Die Darstellung des Terrains auf Annotation (siehe Abbildung B.2 blauer Bereich) einstellen

3. Ein weiteres Menü erscheint und kann zur Auswahl einer Terrainklasse (siehe Abbildung B.2 gelber Bereich) verwendet werden

4. Im *Debug Drawings*-Bereich (siehe Abbildung B.2 roter Bereich) muss nun *TerrainForAnnotationM* ausgewählt und *TerrainClassificationM* abgewählt werden

5. Einer visualisierten Terrainzelle (siehe Abbildung B.2 grüner Bereich) kann durch Halten der Taste `Strg` und einen Linksklick die ausgewählte Klasse zugewiesen werden

Das annotierte Terrain bietet verschiedene Verwendungsmöglichkeiten. In Tabelle B.1 sind die Befehle beschrieben, die der Anwender im Kommandozeilen-Interface der Benutzeroberfläche eingeben kann.

Es muss darauf geachtet werden, dass beim Laden von annotiertem Terrain, das Abspielen der Sensordaten an der richtigen Stelle pausiert wird. Ansonsten spiegelt das annotierte Terrain nicht die gleichen Sensordaten wider, die zu diesem Zeitpunkt dargestellt werden.

Der `annotation extract`-Befehl ermöglicht es dem Anwender, Terrainmerkmale von mehreren annotierten Stellen eines oder mehrerer Logfiles zur Gewinnung der Mittelwerte und Standardabweichungen zu verwenden. So ist es möglich, erst eine breite Auswahl an Merkmalen zu sammeln, bevor diese zur Parameterberechnung verwendet werden.

Da bei der Ausgabe der berechneten Werte keine explizite Benennung der zugehörigen Merkmale erfolgt, ist eine Zuordnung in Tabelle B.2 dokumentiert. Hierbei ist darauf zu achten, dass die Zuordnung der Bildmerkmale nur gültig ist, wenn der Benutzer diese alle in der Konfigurationsdatei aktiviert hat.

Befehl	Beschreibung
`annotation save`	Öffnet einen Dialog zum Speichern der aktuellen Annotation
`annotation load`	Öffnet einen Dialog zum Laden einer gespeicherten Annotation
`annotation extract`	Extrahiert die Merkmale des aktuell annotierten Terrains
`annotation means`	Berechnet die Mittelwerte der extrahierten Merkmalen
`annotation devs`	Berechnet die Standardabweichungen der extrahierten Merkmale
`annotation print`	Gibt die berechneten Mittelwerte und Standardabweichungen im Terminal aus
`evaluate terrain`	Vergleicht annotiertes und klassifiziertes Terrain und gibt die Ergebnisse aus

Tabelle B.1: Befehle für den Annotationsmodus

Befehl	Beschreibung
`Feature 0`	Rauheit
`Feature 1`	Höhendifferenz
`TextureFeature 0`	Angular Second Moment
`TextureFeature 1`	Variance
`TextureFeature 2`	Inverse Difference Moment
`TextureFeature 3`	Homogenität
`TextureFeature 4`	Farbwinkel

Tabelle B.2: Merkmalsentsprechungen der Annotationsausgabe

Anhang C

Konfigurierbare Parameter

Die Einstellungsmöglichkeiten des Roboter-Frameworks sind über eine Konfigurationsdatei möglich. Die Parameter, die für das in der vorliegenden Arbeit vorgestellte Verfahren relevant sind, sind in den Tabellen C.1 und C.2 beschrieben. Tabelle C.2 zeigt Parameter, die jeweils als Mittelwert und Standardabweichung vorhanden sind. Dies wird in der Konfigurationsdatei durch die Tags `<Means>...</Means>`, für Mittelwerte und `<StdDeviations>...</StdDeviations>`, für Standardabweichungen realisiert.

Parametername	**Beschreibung**
`bUseMRF`	(De)aktiviert das Verfahren
`bUseFastHomogenity`	(De)aktiviert die Verwendung des Homogenitätsmerkmals
`bUseTextureFeatures`	(De)aktiviert die Verwendung der Haralickmerkmale
`bUseColorAngle`	(De)aktiviert die Verwendung des Farbwinkels
`fAlpha`	Gewichtungsfaktor α
`fBeta`	Gewichtungsfaktor β
`fInitialTemperature`	Initialer Temperaturparameter
`fFinalTemperatur`	Abbruch-Temperaturparameter
`fTemperatureDecrease`	Faktor zur Verringerung der Temperatur
`bEightNeighbours`	Schaltet zwischen 8er und 4er Nachbarschaft um

Tabelle C.1: Liste der konfigurierbaren Parameter

Parametername	**Beschreibung**
f[Class]Roughness	Mittelwert bzw. Standardabweichung der Rauheit der Terrainklasse [Class]
f[Class]Height	Mittelwert bzw. Standardabweichung der Höhendifferenz der Terrainklasse [Class]
f[Class]AngularSecondMoment	Mittelwert bzw. Standardabweichung des Second Angular Moments der Terrainklasse [Class]
f[Class]SumOfSquaredVariance	Mittelwert bzw. Standardabweichung der Variance der Terrainklasse [Class]
f[Class]InverseDifferenceMoment	Mittelwert bzw. Standardabweichung des Inverse Difference Moments der Terrainklasse [Class]
f[Class]Homogenity	Mittelwert bzw. Standardabweichung der Homogenität der Terrainklasse [Class]
f[Class]ColorAngle	Mittelwert bzw. Standardabweichung des Farbwinkels der Terrainklasse [Class]

Tabelle C.2: Liste der konfigurierbaren Mittelwerte und Standardabweichungen: [Class] muss durch *Street*,*RoughCombined* oder *Obstacle* ersetzt werden und bezieht sich entsprechend auf die Terrainklassen Street, Rough oder Obstacle

Literaturverzeichnis

[BKYZ96] Berthod, Mark ; Kato, Zoltan ; Yu, Shan ; Zerubia, Josiane: Bayesian Image Classification Using Markov Random Fields. In: *Image and Vision Computing* Bd. 14, 1996, S. 285–295

[DC04] Deng, Huawu ; Clausi, David A.: *Unsupervised image segmentation using a simple MRF model with a new implementation scheme.* 2004

[GG84] Geman, S. ; Geman, D.: Stochastic relaxation, gibbs distribution, and bayesian restoration of images. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6 (1984), Nr. 6, S. 721–741

[HDS73] Haralick, R. M. ; Dinstein, I. ; Shanmugam, K.: Textural Features for Image Classification. In: *IEEE Transactions on Systems, Man, and Cybernetics* Bd. 3, 1973, S. 610–621

[HOJ06] Happold, M. ; Ollis, M. ; Johnson, N.: Enhancing Supervised Terrain Classification with Predictive Unsupervised Learning. In: *Proceedings of Robotics: Science and Systems.* Philadelphia, USA, August 2006

[Kat94] Kato, Zoltan: *Phd-Thesis: Multi-scale Markovian Modelisation in Computer Vision with Applications to SPOT Image Segmentation.* INRIA Sophia Antipolis, France, 1994

[KM10] Knauer, Uwe ; Meffert, Beate: Fast Computation of Region Homogenity with Application in a Surveillance Task. In: *ISPRS Commission V Mid-Term Symposium Close Range Image Measurement Techniques* Bd. XXXVIII Part 5, 2010

[KZB92] Kato, Zoltan ; Zerubia, Josiane ; Berthod, Mark: Satellite Image Classification Using a Modified Metropolis Dynamics. In: *IEEE Proceedings of International Conference on Acoustics, Speech and Signal Processing*, 1992

[LHN+09] Lang, Dagmar ; Häselich, Marcel ; Neuhaus, Frank ; Dillenberger, Denis ; Volk, André ; Paulus, Dietrich: Abschlussbericht Autonomiefähigkeit unbemannter Landfahrzeuge / Universistät Koblenz-Landau, www.uni-koblenz.de. 2009. – Forschungsbericht

[Li09] Li, S. Z.: *Markov Random Field Modeling in Computer Vision*. London : Springer, 2009

[MM11] Miyamoto, Eizan ; Merryman, Thomas: *Fast Calculation of Haralick Texture Features*. Human Computer Interaction Institute, Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, 2011. – Zuletzt aufgerufen 02/2011, Datum der Veröffentlichung unbekannt

[MRR+53] Metropolis, Nicholas ; Rosenbluth, Arianna ; Rosenbluth, Marshall ; Teller, Augusta ; Teller, Edward: Equation of State Calculations by Fast Computing Machines. In: *The Journal of Chemical Physics* 21 Number 6 (1953), S. 1087–1092

[Mur02] Murphy, Kevin P.: Hidden semi-markov models. 2002. – Forschungsbericht

[NDPP09] Neuhaus, Frank ; Dillenberger, Denis ; Pellenz, Johannes ; Paulus, Dietrich: Terrain Drivability Analysis in 3D Laser Range Data for Autonomous Robot Navigation in Unstructured Environments. In: *14th IEEE International Conference on Emerging Technologies and Factory Automation*, 2009

[PND+09] Pellenz, Johannes ; Neuhaus, Frank ; Dillenberger, Denis ; Droege, Detlev ; Paulus, Dietrich: Abschlussbericht Untersuchung und Integration des Velodyne HDL-64E S2 / Universität Koblenz-Landau. Koblenz, 2009. – Forschungsbericht

[RHM03] Rankin, Arturo ; Huertas, Andres ; Matthies, Larry: Negative obstacle detection by thermal signature. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003, S. 906– 913

[TK09] Theodoridis, Sergios ; Koutroumbas, Konstantinos: *Pattern Recognition*. 4. Academic Press, 2009

[VHKH04] Vandapel, Nicolas ; Huber, Daniel ; Kapuria, Anuj ; Hebert, Martial: Natural Terrain Classification using 3-D Ladar Data. In: *IEEE International Conference on Robotics and Automation* Bd. 5, 2004, S. 5117 – 5122

[WCS05] Wellington, Carl ; Courville, Aaron ; Stentz, Anthony: Interacting Markov Random Fields for Simultaneous Terrain Modeling and Obstacle Detection. In: *Robotics: Science and Systems*, 2005, S. 1–8

[WKSB09] Wurm, Kai M. ; Kümmerle, Rainer ; Stachniss, Cyrill ; Burgard, Wolfram: Improving robot navigation in structured outdoor environments by identifying vegetation from laser data. In: *Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems*. Piscataway, NJ, USA : IEEE Press, 2009, S. 1217–1222

[WS08] Wolf, Denis F. ; Sukhatme, Gaurav S.: Semantic Mapping Using Mobile Robots. In: *IEEE Transactions on Robotics* 24 (2008), Nr. 2

[WSFB05] Wolf, D.F. ; Sukhatme, G. ; Fox, D. ; Burgard, W.: Autonomous Terrain Mapping and Classification Using Hidden Markov Models. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005. ICRA 2005.* Los Angeles, CA, USA, 2005, S. 2026–2031