



Bachelorarbeit am Institut für Informatik der Freien Universität Berlin,
Arbeitsgruppe Intelligente Systeme und Robotik

Objekterkennung anhand von farbklassifizierten Integralbildern und der Kullback-Leibler Divergenz

Michael Thio Pluhatsch

Matrikelnummer: 4127640

michael.pluhatsch@fu-berlin.de

Gutachter:

Prof. Dr. Raúl Rojas

Prof. Dr. Daniel Göhring

Berlin, 24. Juli 2017

Zusammenfassung

Die umfangreichen Regeländerungen der RoboCup Kid-Size League erfordern neue Methoden zur Erkennung der Spielobjekte. In dieser Arbeit wird eine Objekterkennung beschrieben, die in Echtzeit das Spielfeld, die Tore und einen überwiegend weißen Ball erkennt. Es wird ein Verfahren vorgestellt, dass effizient Bilder segmentiert und Farban-sammlungen findet. Am Ball wird erläutert, wie eine Klassifizierung anhand weniger Trainingsdaten funktionieren kann. Die Objekterken-nung zeigt eine gute Erkennungsrate über mehrere Meter bei geringer Fehlerquote.

Eidesstattliche Erklärung

Ich versichere hiermit an Eides Statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

Berlin, 24. Juli 2017

Michael Thio Pluhatsch

Inhaltsverzeichnis

1	Einleitung	1
1.1	Motivation	1
1.2	Ziel der Arbeit	1
1.3	Aufbau der Arbeit	2
2	Grundlagen	4
2.1	YCbCr-Farbmodell	4
2.2	Farbklassifizierung	5
2.3	Integralbilder	6
2.4	Kullback-Leibler Divergenz	9
3	Verwandte Arbeiten	11
4	Objekterkennung	13
4.1	Farbklassifizierte Integralbilder	14
4.2	Shrink-Methode	15
4.3	Erkennung von einfarbigen Objekten	18
4.3.1	Einfarbiger Ball	18
4.3.2	Spielfeld	20
4.3.3	Tore	21
4.4	Erkennung von mehrfarbigen Objekten	22
4.4.1	Kandidatenfindung	23
4.4.2	Klassifizierung mit der Kullback-Leibler Divergenz	26
5	Auswertung	29
5.1	Laufzeitanalyse	29
5.2	Erkennungsrate	30
5.3	Vor- und Nachteile	34
6	Fazit und Ausblick	36

Abbildungsverzeichnis

1	CbCr-Fläche des YCbCr-Farbmodells im RGB-Farbraum auf einer Helligkeitsebene von $Y = 0,5$	4
2	Artefakte bei der JPEG Kompression	5
3	Überführung vom Kamerabild zum farbklassifizierten Bild	6
4	Aufgespanntes Rechteck	7
5	Überführung vom Eingabebild zum Integralbild	8
6	Berechnung von Pixelsummen in Bildausschnitten	9
7	Farbklassifizierte Bilder	14
8	Überführung vom Binärbild zum farbklassifizierten Integralbild	15
9	Suche einer Farbansammlung	16
10	Suchen und Finden eines einfarbigen Balls	20
11	Feldkontur ermitteln	21
12	Suchen und Finden der Tore	22
13	Unterschiedliche Bälle	23
14	Schwierige Aufgabe: Erkennen eines überwiegend weißen Balls	24
15	Blätter und Ballkandidaten	25
16	Helligkeitsunterschied eines Balls	26
17	Kleine Rastergrößen	27
18	Größere Raster erhöhen die Genauigkeit	27
19	Gefundener überwiegend weißer Balls	28
20	Laufzeiten der relevanten Module	31
21	Helligkeitsunterschiede zweier baugleicher Kameras mit gleichen Einstellungen	34

Tabellenverzeichnis

1	Durchschnittliche Laufzeit der Objekterkennung	29
2	Erkennung eines ruhenden Balls	32
3	Erkennung eines rollenden Balls	33
4	Erkennung eines Balls, während der Roboter zum Ball läuft .	33

1 Einleitung

1.1 Motivation

Von Automobilherstellern über Technologiekonzerne bis zu Social Media Anbietern arbeitet eine Vielzahl von Unternehmen und Wissenschaftlern an dem Ziel, Maschinen noch intelligenter zu machen. Das Fachgebiet der künstlichen Intelligenz und Robotik erhält zunehmend Einfluss in unserer Gesellschaft. Schon heute sind viele Systeme mit künstlicher Intelligenz im Einsatz. Beispiele sind Flugzeugautopiloten, unterstützende Operationsroboter, Assistenten rund um das sehr populäre Thema autonomes Fahren oder Gesichtserkennungen zum Freischalten von Smartphones.

Das Verhalten künstlicher Intelligenz bestimmt sich wie beim Menschen, vor allem aus den Daten und Informationen, die zur Verfügung stehen. Die visuelle Wahrnehmung ist ein Mittel, um Informationen aus der Umgebung aufzunehmen, zu verarbeiten und zu interpretieren.

Roboter nehmen ihre Umgebung durch Sensoren wahr. Für die visuelle Aufnahme der Umgebung kommen z. B. Kameras zum Einsatz. Durch Detektieren und Klassifizieren von Objekten in den aufgenommenen Kamerabildern können Entscheidungen getroffen werden.

Das Detektieren und Klassifizieren von Objekten in einem überschaubaren Umfeld wird unter anderem im Rahmen der Initiative RoboCup [1][2] erforscht. In unterschiedlichen Ligen spielen Teams aus der ganzen Welt mit Robotern gegeneinander Fußball. Ziel ist es, 2050 den amtierenden FIFA-Weltmeister zu besiegen [3].

1.2 Ziel der Arbeit

Diese Arbeit befasst sich mit einer Objekterkennung für die Roboter der FHumanoids [4], ein studentisches Projekt der Freien Universität Berlin. Das Team der FHumanoids spielt in der Humanoid Kid-Size League des RoboCups [5].

Die sich ständig ändernden Spielsituationen im (Roboter-)Fußball erfordern eine präzise Verarbeitung von Bildern in Echtzeit. Das Ziel dieser Arbeit ist eine Lösung bereitzustellen, welche unterschiedliche Objekte auf dem Spielfeld erkennt. Hierfür sollen möglichst wenige rechenintensive Schritte zum Einsatz kommen. Diese dienen wiederum als Basis für schnelle und effiziente Methoden zur Analyse von Bildern und Bildsegmenten. Das Ergebnis ist eine Objekterkennung, die in Echtzeit berechnet. Eine Echtzeitberechnung ist

nach Tomas Akenine-Möller et al. [6] bei 15 Bildern pro Sekunde gegeben. Die hier beschriebene Objekterkennung soll 30 Bilder pro Sekunde (Kapitel 2.2) verarbeiten. Dabei muss die Erkennungsrate der gesuchten Objekte möglichst maximal sein. Falsch erkannte Objekte sollen minimiert werden. Neben dem Finden des Spielfeldes und der Tore liegt das Hauptaugenmerk auf dem Erkennen von überwiegend weißen Bällen.

Die Objekterkennung erfordert darüber hinaus ein hohes Maß an Effizienz, da weiterhin genügend Rechen- und Speicherressourcen für andere Aufgaben wie Laufen, Kommunikation und Entscheidungsfindung zur Verfügung stehen müssen.

1.3 Aufbau der Arbeit

Kapitel 2 - Grundlagen Dieses Kapitel gibt einen Überblick über das hier verwendete YCbCr-Farbmodell und die Farbklassifizierung der FUManoids. Des Weiteren wird das Integralbild und die sich daraus ergebenen Vorteile für die Bildanalyse erklärt. Abschließend folgt eine Beschreibung der Kullback-Leibler Divergenz, welche in dieser Arbeit zur Klassifizierung von Objekten benötigt wird.

Kapitel 3 - Verwandte Arbeiten Zunächst werden Arbeiten der FUManoids und eines anderen Teams beschrieben, die sich mit dem Thema der Objekterkennung befasst haben. Darüber hinaus werden zwei weitere Objekterkennungen vorgestellt. Sie sind auf Grund ihrer Verfahren für diese Arbeit wichtig.

Kapitel 4 - Objekterkennung Die Objekterkennung stellt den Hauptteil dieser Arbeit dar.

Hier wird das farbklassifizierte Integralbild vorgestellt. Es dient als Basis für eine Methode, über die ein Bild in wenigen Schritten auf Bildausschnitte von Farbansammlungen geschrumpft werden kann. Diese Methode ist zentraler Bestandteil der folgenden Objekterkennung.

Infolgedessen wird die Erkennung von einfarbigen Objekten wie Spielfeld, Tore und einfarbiger Ball beschrieben. Am Ende des Kapitels wird beispielhaft eine Erkennung von mehrfarbigen Objekten anhand eines überwiegend weißen Balls vorgestellt. Hierzu zählen die Kandidatenfindung, d. h. das Finden von interessanten Bildregionen, und die Klassifizierung.

Kapitel 5 - Auswertung Zunächst wird eine Laufzeitanalyse der vorgestellten Algorithmen vorgenommen und die Erkennungsrate von überwiegend weißen Bällen untersucht. Anschließend folgt eine Beschreibung von Vor- und Nachteilen der vorgestellten Objekterkennung.

Kapitel 6 - Fazit und Ausblick Schließlich werden die wichtigsten Erkenntnisse dieser Arbeit zusammengefasst. Darüber hinaus werden Verbesserungsansätze vorgeschlagen und über mögliche zukünftige Einsatzszenarien der vorgestellten Objekterkennung nachgedacht.

2 Grundlagen

Die in diesem Kapitel vorgestellten Grundlagen sind notwendige Voraussetzung für das Verständnis und die Umsetzung der beschriebenen Objekterkennung.

2.1 YCbCr-Farbmodell

Mit der Einführung von Farbfernsehern wurde das YUV-Farbmodell geschaffen. Damalige Schwarzweißfernseher erhielten lediglich ein Helligkeitssignal. Um das Farbfernsehen zu ermöglichen und gleichzeitig Abwärtskompatibilität bei bestehenden Schwarzweißfernsehern zu erhalten, wurde das Signal um zwei Farbkomponenten (Chrominanz) erweitert [7].

Diese beiden Farbkomponenten beschreiben die Differenz zwischen der Luminanz Y (Helligkeit) und dem blauen Farbanteil U beziehungsweise dem roten Farbanteil V. Der grüne Farbanteil lässt sich über U, V und Y berechnen und muss daher nicht als Signal übertragen werden.

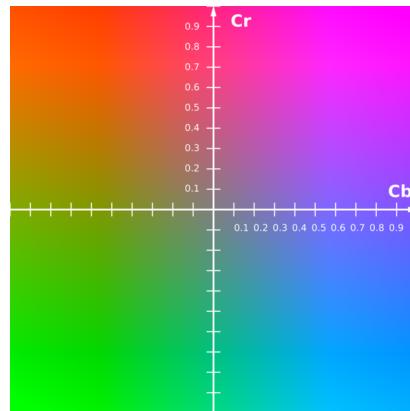


Abbildung 1: CbCr-Fläche des YCbCr-Farbmodells im RGB-Farbraum auf einer Helligkeitsebene von $Y = 0,5$

Das YUV-Farbmodell entstammt noch dem analogen Bereich. Für diese Arbeit wird das digitale YCbCr-Farbmodell benutzt. Es wird in ITU-R Recommendation BT.601 [8] definiert. Berechnet wird YCbCr durch eine Helligkeitskorrektur des analogen RGB-Farbraums [9]:

$$Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \quad (1)$$

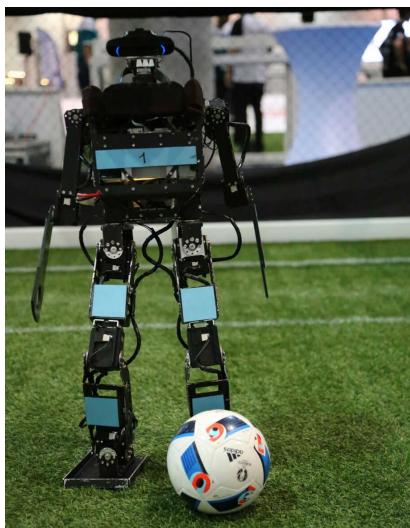
$$C_B = -0.169 \cdot R - 0.331 \cdot G + 0.500 \cdot B + 128 \quad (2)$$

$$C_R = 0.500 \cdot R - 0.419 \cdot G - 0.081 \cdot B + 128 \quad (3)$$

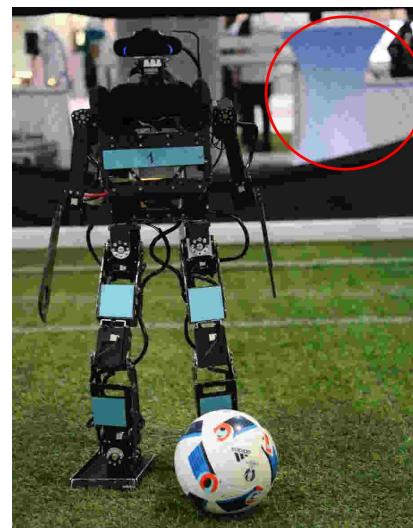
Der RGB-Farbraum ist den drei Zapfentypen auf der Netzhaut des menschlichen Auges nachempfunden. Die Helligkeitskorrektur erfolgt, da der Mensch bei gleicher Intensität der Farben Rot, Grün und Blau die grüne Farbe am hellsten wahrnimmt. Es folgen die Farben Rot und Blau [10].

2.2 Farbklassifizierung

Die Roboter der FHumanoids sind mit einer Logitech HD Pro Webcam C910 ausgestattet. Die Kamera nimmt Bilder mit einer Auflösung von 960x720 Pixeln bei einer Geschwindigkeit von 30 Bildern pro Sekunde auf. Die Bilder werden als Motion JPEG¹ übertragen und in YCbCr Bilder umgewandelt. Die Übermittlung der Rohdaten wäre zu langsam, um hohe Auflösungen nutzen zu können. Durch die Kompression in JPEG kommt es allerdings zu Artefakten im Bild, da Informationen verloren gehen (Abbildung 2).



(a) Originalbild



(b) Auffällig: Die Artefakte im Hintergrund (rot markiert)

Abbildung 2: Artefakte bei der JPEG Kompression

Eine geeignete Objekterkennung muss dementsprechend robust gegenüber Artefakten sein.

Das Framework der FHumanoids verfügt über eine Möglichkeit, jedem Pixel des Kamerabildes eine logische Farbe zuzuordnen. Logische Farben im RoboCup sind traditionell Grün, Weiß, Gelb, Orange, Cyan, Magenta und

¹Jedes Bild wird separat als JPEG komprimiert

Schwarz. Alle weiteren Farben werden als logische Farbe „Unbestimmt“ zusammengefasst. Sie ergeben sich aus den Regeln des RoboCups [2] und sind Spielementen zugeordnet, wie zum Beispiel das Grün dem Spielfeld (Abbildung 3).

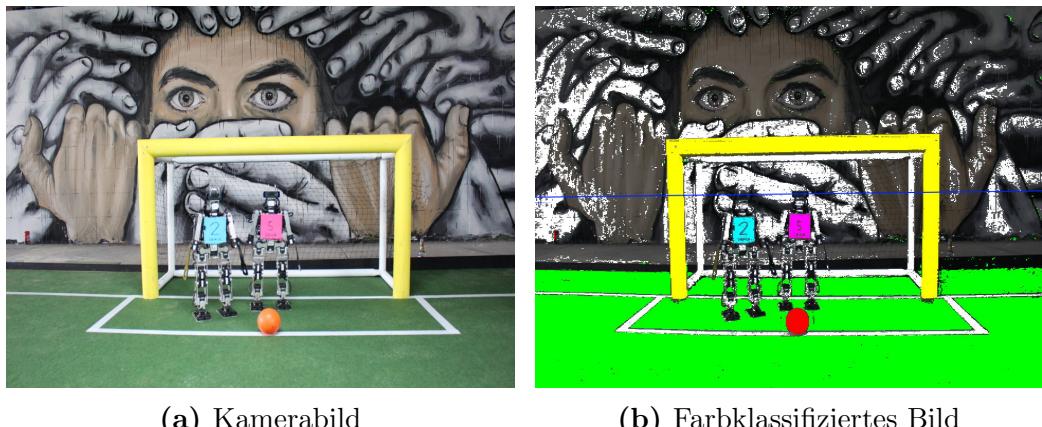


Abbildung 3: Überführung vom Kamerabild zum farbklassifizierten Bild

Zur Klassifizierung der einzelnen Pixel zu einer logischen Farbe werden Cluster genutzt, welche die logischen Farben repräsentieren. Über manuell ausgewählte Punkte werden die Cluster gebildet. Durch die Berechnung des Mittelwerts und der Kovarianz der gewählten Punkte wird eine Gaußkurve beschrieben, die eine logische Farbe darstellt. Zur Klassifizierung der Pixel zu einer logischen Farbe wird der Expectation-Maximization-Algorithmus verwendet [11].

Zur Beschleunigung der Klassifizierung wird bei der Farbkalibrierung eine Lookup-Tabelle aufgebaut. Die Kalibrierung der Farben findet vor jedem Spiel manuell statt.

2.3 Integralbilder

Integralbilder wurden ursprünglich im Jahr 1984 unter dem Namen *Summed-Area Table* von Franklin C. Crow für Texture Mapping [12] vorgestellt. Erstmals in der Bildverarbeitung nutzten Paul Viola und Michael Jones Integralbilder zur Gesichtserkennung[13][14]. Ein Integralbild dient zur Berechnung von Pixelsummen einer beliebigen rechteckigen Bildregion [15].

Die Summe einer reellwertigen Funktion $f(x)$ im Intervall $[a; b]$ kann durch ein Integral bestimmt werden.

$$\int_a^b f(x) dx. \quad (4)$$

Die Stammfunktion von $f(x)$ ist von Vorteil, wenn oft Berechnungen mit anderen Integrationsgrenzen erfolgen.

$$F(t) = \int_{-\infty}^t f(x) dx. \quad (5)$$

Eine einfache Subtraktion ersetzt somit den Aufwand wiederkehrender Integrationen.

$$\int_a^b f(x) dx = F(b) - F(a) \quad (6)$$

Dieses Prinzip ist auf Bilder übertragbar. Jeder Punkt $I(x, y)$ im Integralbild enthält die Summe aller Punkte $i(x, y)$ im Eingabebild in dem aufgespannten Rechteck zwischen dem Ursprung und dem Punkt (x, y) (Abbildung 4).

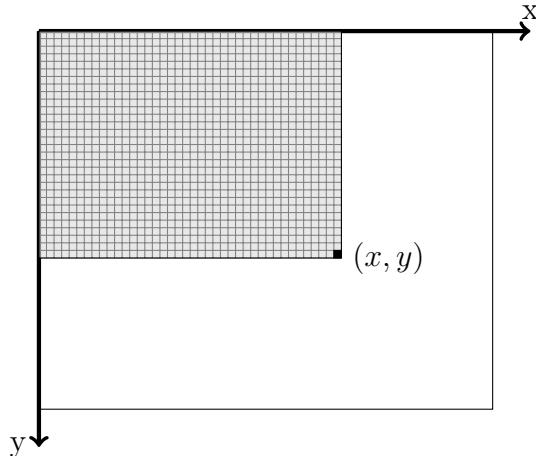


Abbildung 4: Aufgespanntes Rechteck

$I(x, y)$ berechnet sich folgendermaßen:

$$I(x, y) = \sum_{j=0}^{j \leq x} \sum_{k=0}^{k \leq y} i(j, k) \quad (7)$$

Für die Berechnung des Integralbildes muss auf einzelne Punkte aus dem Eingabebild mehrfach zugegriffen werden. Diese Formel ist daher in der Praxis ungünstig. Für eine Berechnung in einem einzigen Durchlauf über das Eingabebild eignen sich folgende Rekursionsgleichungen:

$$\begin{aligned} R(x, y) &= R(x, y - 1) + i(x, y) \\ I(x, y) &= I(x - 1, y) + R(x, y) \end{aligned} \tag{8}$$

$R(x, y)$ ist die kumulative Zeilensumme. Es gilt:

$$R(x, -1) = 0 \text{ und } I(-1, y) = 0.$$

Gegenüber dem Eingabebild ist das Integralbild um eine Spalte und eine Zeile größer. Die zusätzliche Spalte und Zeile werden links und oberhalb des Integralbildes eingefügt und enthalten nur Nullen (Abbildung 5). So können auch Rechtecke am oberen und linken Rand eines Bildes berechnet werden.

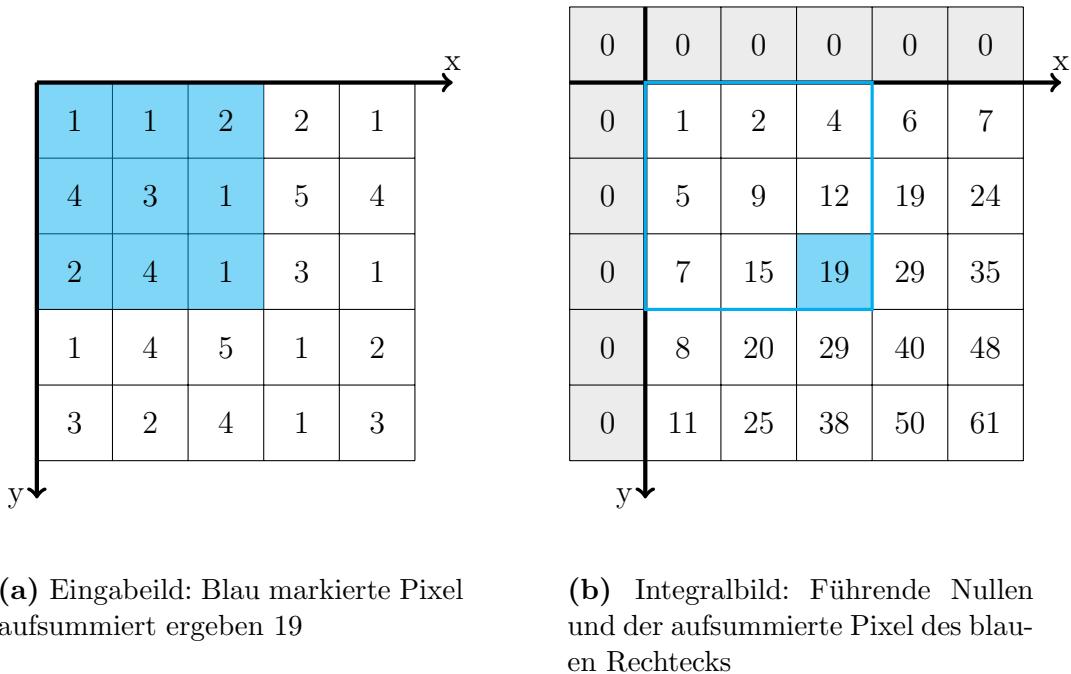
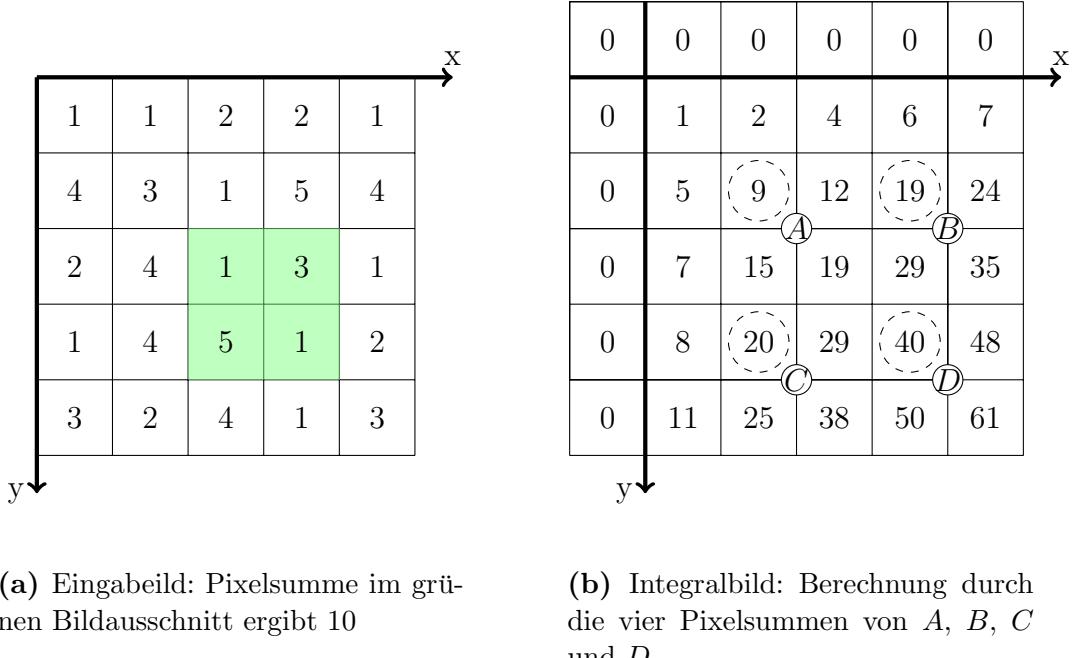


Abbildung 5: Überführung vom Eingabebild zum Integralbild

Ein Vorteil von Integralbildern ist, dass sich die Summe S der Werte (Formel 9) des Eingabebildes innerhalb eines beliebigen Rechteckes an der Position (x, y) mit einer Breite w und einer Höhe h durch vier Speicherzugriffe berechnen lässt. Die Laufzeit ist demnach konstant. Die Berechnung erfolgt durch die entsprechenden vier Pixelsummen $A = I(x, y)$, $B = I(x + w, y)$, $C = I(x, y + h)$ und $D = I(x + w, y + h)$ aus dem Integralbild und ist unabhängig von der Position und Größe des Rechtecks:

$$S = D + A - (B + C) \tag{9}$$

**Abbildung 6:** Berechnung von Pixelsummen in Bildausschnitten

Im Beispiel aus Abbildung 6 ergibt sich die Pixelsumme des grünen Rechtecks in Abbildung 6a aus den Werten 9, 19, 20 und 40 des Integralbildes (Abbildung 6b) wie folgt: $10 = 40 + 9 - (19 + 20)$.

Weitere Vorteile sind die Möglichkeit des schnellen Downsamplings von Bildern und der Durchschnittsberechnung von Pixelwerten in Bildausschnitten. Die in Formel 9 gezeigte Berechnung von Pixelsummen in beliebigen rechteckigen Bildausschnitten kann hierfür verwendet werden. Hierzu muss eine simple Teilung der Pixelsumme durch die Anzahl der Pixel im Ausschnitt vorgenommen werden. Mit dieser Berechnung können ganze Bildregionen zusammengefasst und das Eingabebild „down gesamplet“ werden.

2.4 Kullback-Leibler Divergenz

Die Kullback-Leibler Divergenz (KL Divergenz) wird genutzt, um die Unterschiedlichkeit zweier Wahrscheinlichkeitsverteilungen über die selbe Variable x zu messen. Benannt ist die Methode nach Solomon Kullback und Richard Leibler [16][17], die die Divergenz erstmals 1951 im Rahmen der Wahrscheinlichkeits- und Informationstheorie vorstellten.

Sind P und Q zwei Wahrscheinlichkeitsverteilungen einer diskreten Variablen

x , so ist die KL Divergenz $D_{KL}(P, Q)$ ein Maß für den Informationsverlust, wenn Q verwendet wird, um P zu approximieren.

Die KL Divergenz wird wie folgt definiert:

$$D_{KL}(P, Q) = \sum_x P(x) \cdot \log \frac{P(x)}{Q(x)} \quad (10)$$

Typischerweise repräsentiert P Beobachtungen oder eine präzise Wahrscheinlichkeitsverteilung, während Q ein Modell oder eine Approximation darstellt.

Die KL Divergenz ist allerdings nicht symmetrisch. Das bedeutet, dass die KL Divergenz von P nach Q nicht gleich der KL Divergenz von Q nach P ist.

Des Weiteren gilt:

$$D_{KL}(P, Q) > 0, \text{ für } P \neq Q \quad (11)$$

$$D_{KL}(P, Q) = 0, \text{ für } P = Q \quad (12)$$

3 Verwandte Arbeiten

Zu Beginn wird ein Überblick über frühere und aktuelle Objekterkennungen im RoboCup sowie aus anderen Bereichen gegeben. Einige dieser Arbeiten haben teilweise die in Kapitel 2 beschriebenen Grundlagen verwendet und dienten als Anregung für die in Kapitel 4 vorgestellte Objekterkennung.

Mitglieder der FUmanoids haben sich in früheren Arbeiten mit der Erkennung von Objekten auseinander gesetzt.

Naja v. Schmude [18] beschreibt die eingesetzte Objekterkennung der FUmanoids aus dem Jahr 2010. Auf Basis von farbklassifizierten Bildern, Scanlines und dem sogenannten Gradient Vector Gridding (GVG) Verfahren [19] werden Objekte erkannt. Dabei wird die Komplexität des Bildes über ein Raster von gleich großen Zellen reduziert. Zu jeder Zelle werden sogenannte Edge Representer bestimmt, welche die Hauptkantenrichtungen der Zelle widerspiegeln. Die Kanten im Bild werden zuvor mit einer vereinfachten Kantenerkennung einmal berechnet. Zusätzlich werden die Farbklassen der Pixel links und rechts des Edge Representer überprüft. Die am häufigsten vorkommende Farbe für jede Seite wird gespeichert. Die Bestimmung von Objekten anhand ihrer Farbe kann so über den Kantenverlauf bestimmt werden.

Hamid Moballegh [20] versucht, die runde Eigenschaft von Bällen zur Erkennung zu verwenden. Mit GVG, Edge orientation histograms und einer rekursiven Suche auf Teilbildern von Histogram Integral Images nutzt der beschriebene Algorithmus die Tatsache, dass die Orientierung der Kanten eines runden Objekts idealerweise gleich verteilt ist. Motiviert ist die Ballerkennung unter anderem durch das von Dalal und Triggs [21] vorgestellte Histogram of oriented Gradients.

Auch Max Losch [22] versucht, den Ball anhand der Form zu erkennen. Zum Einsatz kommt die Hough-Transformation für Kreise. Die Berechnung der Hough-Transformation ist rechenintensiv, wenn keine geeignete Vorverarbeitung der Bilder stattfindet. Um geeignete Bildausschnitte zu finden, wird eine korrigierte Version der rekursiven Suche auf Teilbildern genutzt. Auch in dieser Arbeit kommen Integralbilder zum Einsatz.

Die letzte Objekterkennung der FUmanoids stammt von Robert Kriehs [23]. Er verwendet Scanlines in Kombination mit einem Region-Growing Algorithmus. Der orangefarbene Ball wird bis zu einer Reichweite von 3,5 Metern erkannt.

Daniel Speck et al. [24] stellte auf dem Symposium des RoboCups 2016 eine Ballerkennung mit einem Convolutional Neural Network vor. Zur Lokalisierung wird das neuronale Netz mit Wahrscheinlichkeitsverteilungen über die

Breite und Höhe des gesamten Bildes trainiert. Da die zu trainierenden Daten zu groß für die eingesetzten Roboter sind, müssen die Eingabebilder auf eine Auflösung von 200x150 reduziert werden. Es kann eine Erkennungsrate von ca. 70% bei Entferungen von bis zu einem Meter erzielt werden. Die Ballerkennungsrate verringert sich jedoch teilweise auf unter 30% bei Entfernungen von über 2 Metern.

Die Gesichtserkennung von Viola und Jones (Kapitel 2.3) ist eines der bekanntesten Verfahren in der Objekterkennung. Sie kann in vier Schritten beschrieben werden. Zunächst werden sogenannte Haar-like Features ausgewählt. Jedes Feature besteht aus mehreren Rechtecken, wobei die Rechtecke dunkle oder helle Regionen repräsentieren. Die Auswahl erfolgt über Gegebenheiten, welche die meisten Gesichter teilen. Zum Beispiel ist die Augenregion dunkler als der Bereich um das Jochbein. Im zweiten Schritt wird ein Integralbild berechnet. Mit Hilfe des Integralbildes wird die Helligkeitsdifferenz der einzelnen Haar-like Features berechnet. Aufgrund der Vielzahl an Möglichkeiten die Features an unterschiedlichen Positionen und in unterschiedlichen Größen zu platzieren wird der AdaBoost-Algorithmus [25] verwendet. Durch die Verwendung von AdaBoost werden Klassifikatoren gewichtet und aus der Verknüpfung von schwachen Klassifikatoren werden starke erstellt. Zum Schluss werden die einzelnen Klassifikatoren nach deren Effizienz kaskadiert.

Im Bereich des autonomen Fahrens haben Kalinke und von Seelen [26] mit ihrer Arbeit „Kullback-Leibler Distanz als Maß zur Erkennung nicht rigider Objekte“ ein Verfahren zur Erkennung von Fußgängern und Zweirädern beschrieben. In einem Bildausschnitt wird die Grauwertverteilung der Bildpunkte bestimmt, d.h. wie häufig kommt ein Grauwert im Bildausschnitt vor. Diese Verteilung wird mit einem gegebenen Modell verglichen. Die Ähnlichkeit wird mit der Kullback-Leibler Divergenz bestimmt.

Die von Paul und Viola sowie Seelen und Kalinke vorgestellten Objekterkennungen dienen als Inspiration für die im folgenden Kapitel gezeigte Objekterkennung.

4 Objekterkennung

Die erfolgreiche Teilnahme am RoboCup erfordert das Schießen und Verhindern von Toren. Um diese Ziele zu erreichen, ist es unabdingbar, dass die Roboter ihre Umgebung erkennen und die aufgenommenen Informationen sinnvoll verarbeiten. Dazu gehört das Finden von elementaren Spielobjekten wie das Spielfeld, die Tore, die Gegner und den Ball.

Durch Dribblings und Schüsse ist der Ball das schnellste Objekt auf dem Spielfeld. Die Ballerkennung muss deshalb so oft wie möglich stattfinden. Bei der Erkennung ist es außerdem erforderlich, möglichst keine scheinbaren Bälle (falsch positiv) zu erkennen. Falsch erkannte Bälle führen zu einem kontraproduktiven Verhalten. Die Roboter laufen in die falsche Richtung und entfernen sich womöglich vom richtigen Ball, teilen ihren Mitspielern eine falsche Position vom Ball mit, wodurch der Gegner einen taktischen Vorteil erhält.

Die Analyse des Bildes und die abschließende Klassifizierung der Objekte ist zeitkritisch. Daher müssen Methoden zur Analyse des Bildes bereit gestellt werden, welche Berechnungen mit hoher Laufzeit ausschließen bzw. auf ein Minimum reduzieren.

Die in Kapitel 2.3 vorgestellten Integralbilder dienen hier als Grundlage. Hinzu kommt ein Verfahren zur Segmentierung von Bildern. Die Segmentierung wird für das Finden von interessanten Bildregionen verwendet.

Auf diese Regionen wird eine Klassifizierung anhand der in Kapitel 2.4 beschriebenen Kullback-Leibler Divergenz angewandt.

Nachfolgend wird ein Verfahren zur Objekterkennung beschrieben, das den Gegebenheiten des RoboCups entspricht. Die genannten Grundlagen werden miteinander verknüpft und neue Methoden zur Spielfeld-, Tor- und Ballerkennung vorgestellt.

4.1 Farbklassifizierte Integralbilder

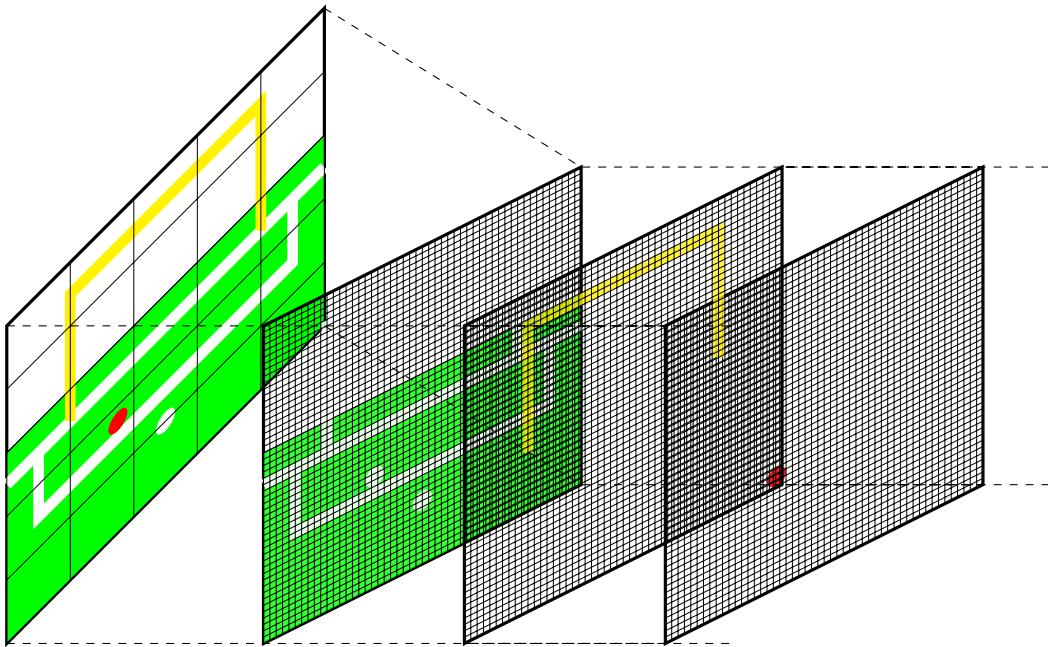


Abbildung 7: Farbklassifizierte Bilder

Aufbauend auf der Farbklassifizierung aus Kapitel 2.2 und den Integralbildern aus Kapitel 2.3 werden farbklassifizierte Integralbilder erstellt.

Die eindeutige Zuweisung eines Pixels zu einer logischen Farbe (Abbildung 7) kann anhand einer binären Schreibweise wiedergegeben werden. Zu jeder Farbe wird ein Binärbild erstellt. Jedem Pixel mit entsprechender Farbe wird eine Eins und jedem anderen Pixel eine Null zugeordnet (Abbildung 8a).

Durch Aufsummieren der Pixel eines Binärbildes entsteht zu jeder logischen Farbe ein sehr simples farbklassifiziertes Integralbild. In diesem farbklassifizierten Integralbild steht im untersten rechten Pixel (x, y) die Summe aller im Bild befindlichen Pixel gleicher Farbe (Abbildung 8b). Mit vier Zugriffen auf das Integralbild wird die Information bereitgestellt, ob und wie viele farbige Pixel sich in einem Bild(-ausschnitt) befinden. Beim gesamten Eingabebild ist sogar nur ein Zugriff notwendig. Die Anzahl der farbigen Pixel gibt Auskunft darüber, ob eine tiefer gehende Analyse vorgenommen werden muss. Unterschreitet der Wert einen vorher definierten Schwellenwert (Hyperparameter) oder ist die Pixelsumme gleich Null, so kann darauf geschlossen werden, dass kein relevantes Objekt im Bild(-ausschnitt) vorhanden ist.

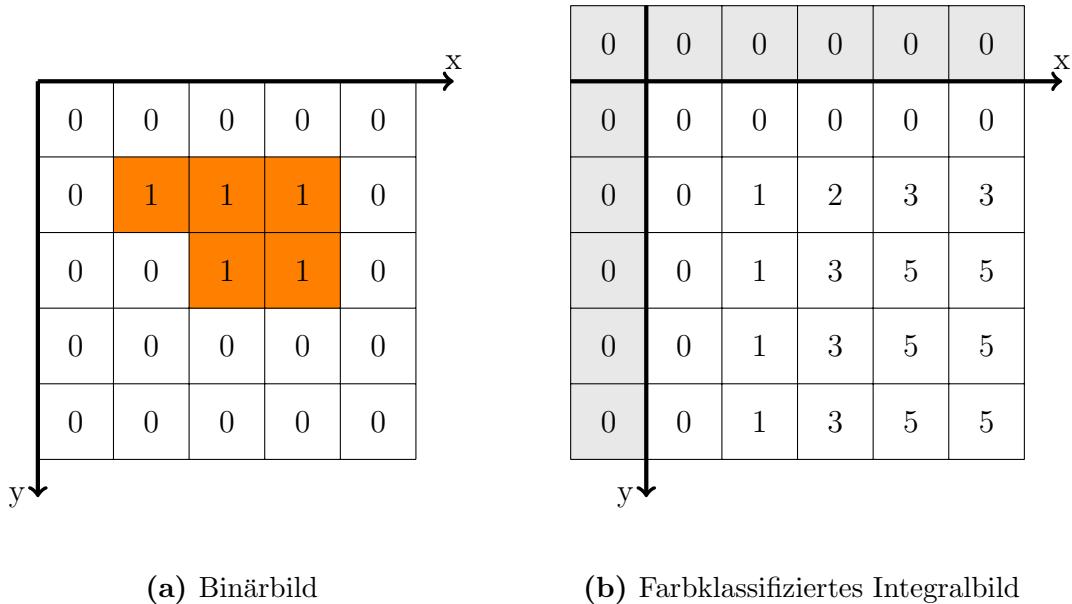


Abbildung 8: Überführung vom Binärbild zum farbklassifizierten Integralbild

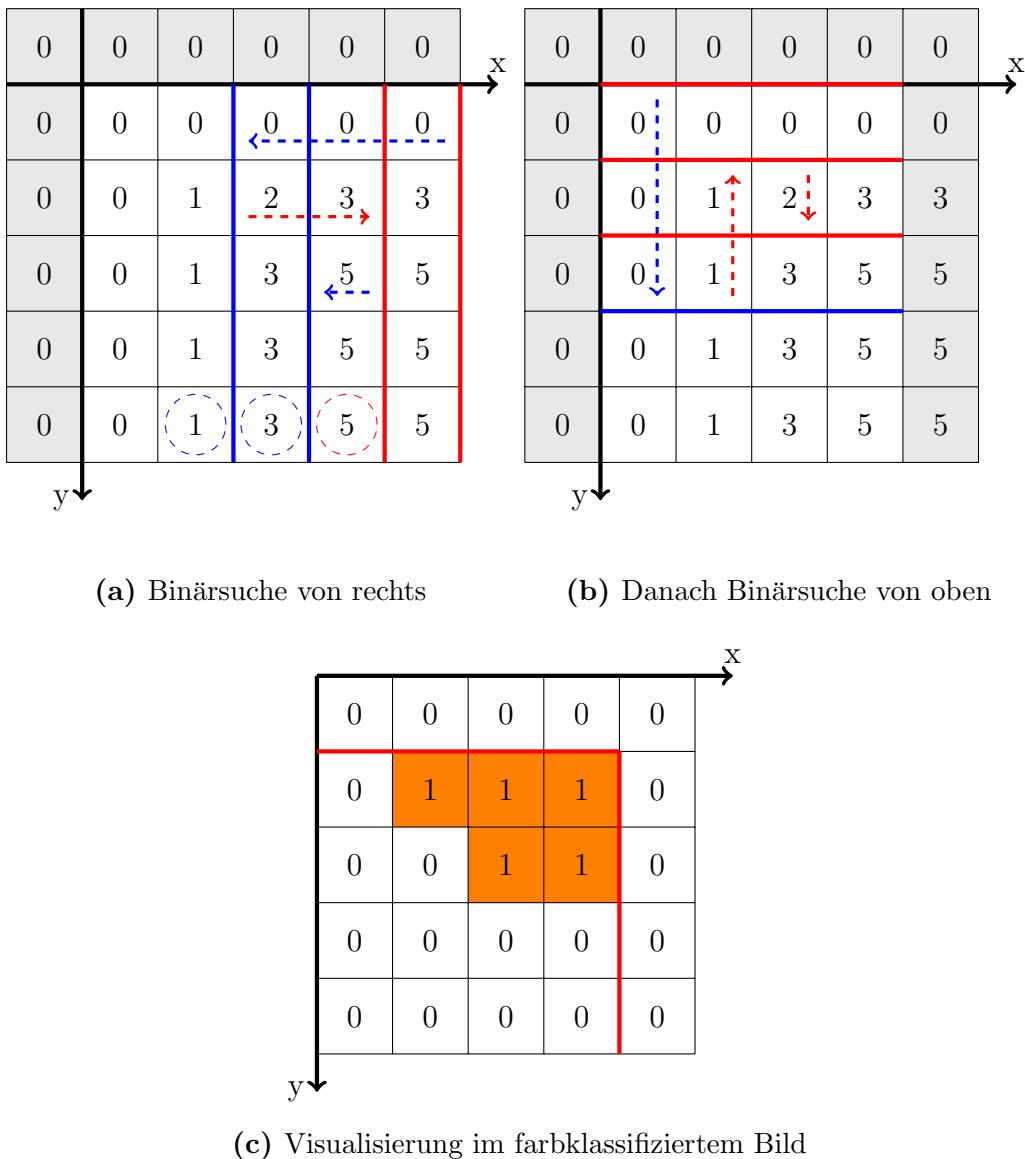
Darüber hinaus ist der relative Anteil von farbigen Pixeln zu allen Pixeln in einem Bildausschnitt schnell bestimmt.

4.2 Shrink-Methode

Für eine genauere Betrachtung von farbklassifizierten Integralbildern wird eine Methode genutzt, welche die mehrfache Binärsuche auf das Integralbild anwendet. Diese Methode ermöglicht das effiziente Suchen und Finden von Farbansammlungen, d.h. von Anhäufungen gleichfarbiger Pixel im Bild. Genauer werden die vier Seiten eines Rechtecks gesucht, die alle farbigen Pixel minimal umschließen. Da man das Bild auf einen kleineren Bildausschnitt schrumpft (engl. to shrink), wird das Verfahren in dieser Arbeit Shrink-Methode genannt.

Zunächst wird für die gesuchte Farbe die Anzahl der Pixel mit der Farbe, nach der gesucht wird, bestimmt. Die Berechnung erfolgt, wie in Kapitel 4.1 beschrieben, über vier Zugriffe auf das Integralbild. Anschließend werden Binärsuchen durchgeführt. Hier exemplarisch von rechts (Abbildung 9a). Dabei wird die Stelle gesucht, an der sich der Pixelsummenwert das erste mal von der Gesamtpixelsumme des Integralbildes unterscheidet. Die Kante zwischen diesen Pixeln ist die rechte Grenze des gesuchten Rechtecks.

Vor der ersten Suche wird das gesamte Bild von einem Rechteck (Bounding-

**Abbildung 9:** Suche einer Farbansammlung

Box²) umschlossen. Die Länge der Bounding-Box wird halbiert (Schrittweite). Der neu entstandene Bildausschnitt wird wiederum auf das Vorkommen der farbigen Pixel untersucht. Ist der Pixelsummenwert identisch mit dem Wert des gesamten Bildes, so wird die Schrittweite nochmals halbiert. Dies geschieht solange bis die Schrittweite nicht weiter halbiert werden kann oder sich die Werte unterscheiden.

²Die Bounding-Box ist ein Rechteck, das einen Bildausschnitt beschreibt.

Algorithmus 1 : Shrink-Algorithmus (von rechts)

Input :

Bildausschnitt: bb;
maximale x -Koordinate: dimension;
Breite: length;
zu suchende Farbe: c;
Anzahl farbiger Pixel: colorCount

```

1 bound = dimension - length;
2 while length > 1 do
3     /* Schrittweite */                      */
4     length = length / 2 ;
5     if dimension - length >= bound then
6         originalDimension = dimension;
7         dimension -= length;
8         ccSub = getSumForRect(bb, c);
9         if ccSub != colorCount then
10            dimension = originalDimension;
11        end
12    end
13 end
```

Unterscheiden sich die Werte, so wird die rechte Kante der Bounding-Box auf die Hälfte zwischen aktueller Kante und der Kante der vorherigen Bounding-Box gesetzt. Die Schritte werden solange wiederholt, bis die gesuchte Stelle gefunden wird (Algorithmus 1). Die Komplexität liegt wie bei der klassischen Binärsuche bei $\mathcal{O}(\log n)$. Die Suche der anderen Seiten des minimal umschließenden Rechtecks der Farbansammlung erfolgt analog, wobei sich nur die Suchrichtung ändert. Das Shrink-Verfahren von allen Seiten ist in Algorithmus 2 abgebildet.

Beispielhaft wird die Suche der rechten und oberen Kante in Abbildung 9 gezeigt.

Wird das Gesamtbild betrachtet, so reduzieren sich die Zugriffe für die rechte und untere Kante nur auf jeweils einen Zugriff. Das Vorkommen aller farbigen Pixel kann im untersten rechten Pixel abgelesen werden (Abbildung 9a).

Algorithmus 2 : Vollständige Shrink-Methode für eine beliebige Bounding-Box

```

1 Function shrinkBB(bb, color):
2     colorCount = getSumForRect(bb, color);
3     shrinkRight(bb, bb maxX, bb.getWidth(), c, colorCnt);
4     shrinkTop(bb, bb minY, bb.getHeight(), c, colorCnt);
5     shrinkBot(bb, bb maxY, bb.getHeight(), c, colorCnt);
6     shrinkLeft(bb, bb minX, bb.getWidth(), c, colorCnt);
7     return bb;

```

4.3 Erkennung von einfarbigen Objekten

Die Erkennung von einfarbigen Objekten, speziell im RoboCup, ist relativ einfach. Oft reicht eine Suche nach Farbansammlungen einer Farbe aus. Dies ergibt sich aus dem Grund, dass Spielemente meist einfarbig sind und jedem Spielement eine eindeutige Farbe zugewiesen wurde.

4.3.1 Einfarbiger Ball

Anhand eines einfarbigen Balls wird das Zusammenspiel von farbklassifizierten Integralbildern und der Shrink-Methode gezeigt. Im Idealfall, welcher in der Regel im RoboCup vorliegt, wird nur ein Ball mit einer eindeutigen Farbe im Bild gesucht.

In Abbildung 3b in Kapitel 2.2 wird die Sicht eines Roboters dargestellt. Die blaue Linie kennzeichnet den Horizont. Bei einem aufrecht stehenden Roboter mit Blick parallel zum Spielfeld ist der Horizont die Linie, die das Bild in der Mitte teilt. Durch Neigung des Kopfes rotiert und verschiebt sich der Horizont. Die FHumanoids stellen eine Methode zur Verfügung, mit der die Rotation und Verschiebung berechnet wird.

Der Bereich über dem Horizont ist niemals relevant. Die Spielfeldausdehnung und die Größe der Roboter haben den Effekt, dass sich das Spielfeld immer unterhalb des Horizonts befindet. Die folgende Ballerkennung bedient sich dieser Eigenschaft und betrachtet nur Bereiche unterhalb des Horizonts.

Darüber hinaus wird mit der Perspective-n-Point Methode [27] die Bestimmung des Ortes eines korrespondierenden Punktes zu einer Pixelkoordinate genutzt. Die Lösung ermöglicht eine Abschätzung, wie weit Objekte auf dem Spielfeld entfernt sind. Ebenso kann ermittelt werden, wie groß der Bereich ist, den ein Pixel abdeckt.

Zunächst wird aus dem farbklassifizierten Bild, das die logische Farbe des Balls widerspiegelt, ein farbklassifiziertes Integralbild erstellt. Danach wird mit vier Speicherzugriffen (Formel 9) ermittelt, ob und wie viele Pixel der Farbe im Bild vorhanden sind.

Sind entsprechend farbige Pixel vorhanden, so ist die Pixelsumme größer als Null. Eine Suche per Shrink-Algorithmus wird von rechts, unten, links und oben durchgeführt.

Da nur der Ball die gesuchte Farbe³ enthält, reicht die Suche nach den Grenzen der Farbansammlung aus.

Durch Reflektion des einfarbigen Balls oder unsaubere Farbkalibrierung können an mehreren Stellen im Bild vereinzelt farbige Pixel auftauchen. Ebenso ist die Methode bei einem weiteren Ball nicht mehr einsetzbar. Eine Suche nach den Grenzen würde zu dem unerwünschten Verhalten führen, dass die Bounding-Box nicht nur den Ball, sondern auch andere Bereiche einschließt.

Die Probleme werden rekursiv gelöst. Das geschrumpfte Bild wird in der Mitte der längeren Seite in zwei Bilder geteilt. Auf die zwei Teilbilder wird die Shrink-Methode angewandt. Dies geschieht solange, bis vorher definierte Kriterien erfüllt sind. Für den einfarbigen Ball ist dies der relative Anteil an entsprechend farbigen Pixeln im Teilbild. Wie schon erwähnt (Kapitel 4.1), sind die farbklassifizierten Integralbilder auch hierfür geeignet. Zudem ist die Größe des Balls bekannt, weshalb die Größe eines Teilbildes Auskunft darüber gibt, wie groß der abgedeckte Bereich ist. Zu große Bildausschnitte müssen daher geteilt werden, um den Ball minimal rechteckig zu umschließen. Ein weiteres Kriterium ist das Seitenverhältnis der Bounding-Box. Das den Ball umschließende Rechteck in einem 2D-Bild, ist im Idealfall ein Quadrat. Weicht das Seitenverhältnis von diesem Ideal zu stark ab, so kann ausgeschlossen werden, dass ein Ball gefunden wurde.

Diese Kriterien bringen das Problem mit sich, dass gegebenenfalls zu kleine Bildausschnitte gefunden werden. Daher müssen nach Durchlauf der Shrink-Methode benachbarte Bildausschnitte zusammengefügt werden.

³Ergibt sich aus den Regeln des RoboCups bis 2015



Abbildung 10: Suchen und Finden eines einfarbigen Balls

Die Erkennung eines einfarbigen Balls mit der beschriebenen rekursiven Lösung wird in Abbildung 10 dargestellt.

4.3.2 Spielfeld

Das Spielfeld ist das dominante Objekt, da es in der Regel in jedem Bild vorkommt. Die große grüne Fläche wird nur stellenweise von Robotern und dem Ball überdeckt. Die Schwierigkeit der Erkennung besteht darin, die äußeren Ränder des Spielfelds zu finden. Bildregionen außerhalb des Spielfelds sind nicht von Bedeutung. Sie führen womöglich sogar zu längeren Berechnungen oder beeinflussen die Entscheidungen der Roboter negativ.

Die farbklassifizierten Integralbilder und die Shrink-Methode sind auch bei diesem Problem nützlich. Das Bild mit der Breite w und der Höhe h wird in n vertikale äquidistante Streifen geteilt (Abbildung 11a). Jeder Bildausschnitt wird unter Zuhilfenahme der Shrink-Methode von oben solange geschrumpft, bis der relative Anteil an grünen Pixeln im Bildausschnitt einem vorher festgelegten Schwellenwert entspricht (Abbildung 11b).

Der gemeinsame Punkt von jeweils benachbarten Bildausschnitten mit kleinstem y -Koordinate wird als Feldkonturpunkt extrahiert. In Abbildung 11c sind dies die Punkte A und B . Dazu kommen immer die vier blau markierten Feldkonturpunkte ($0, y$ -Koordinate von A), (w, y -Koordinate von B), (w, h) und $(0, h)$.

Über diese Feldkonturpunkte wird nun die konvexe Hülle bestimmt. Sie stellt die Kontur des Spielfeldes dar (Abbildung 11d).

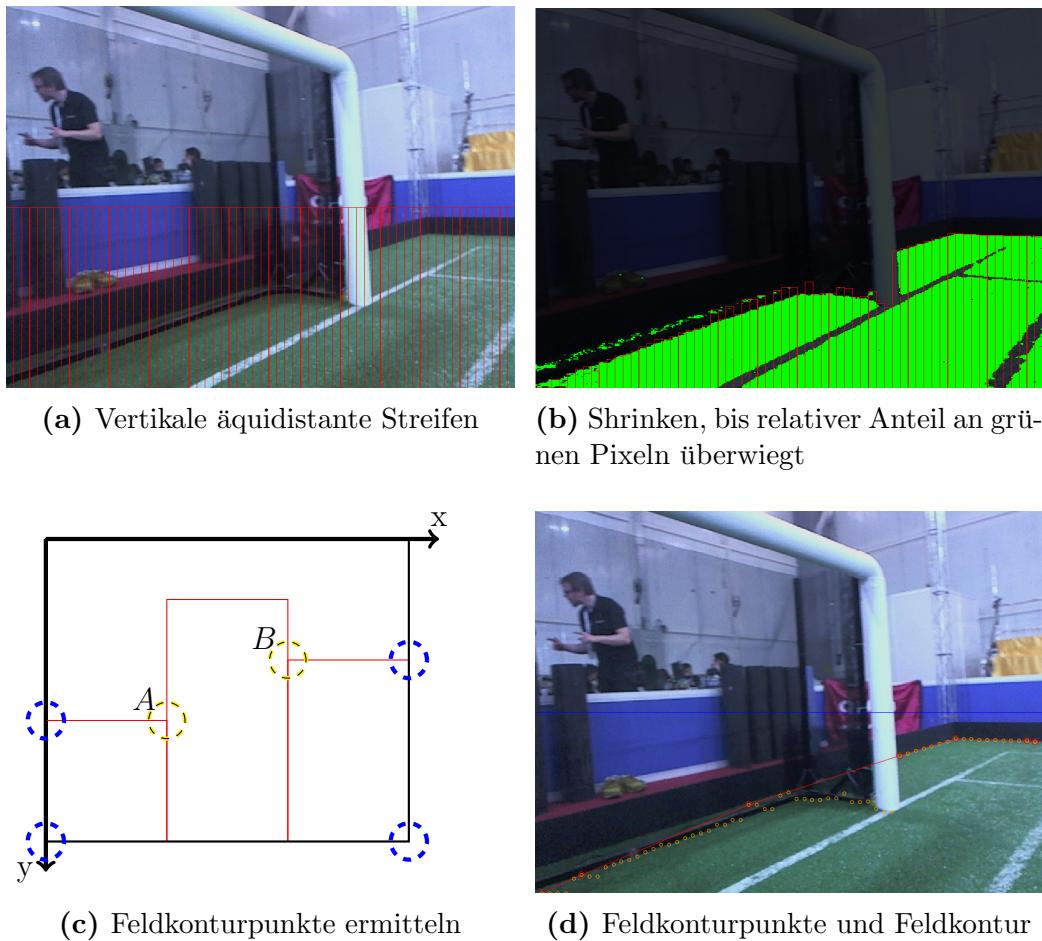


Abbildung 11: Feldkontur ermitteln

4.3.3 Tore

Für die Torerkennung (Abbildung 12) wird entlang der Feldkontur nach weißen Farben gesucht. Dies geschieht mit vielen, sich überlappenden Rechtecken. Die farbklassifizierten Integralbilder erlauben einen schnellen Zugriff auf die Farbinformationen der Rechtecke. Nicht relevante Rechtecke, d.h. Rechtecke die keinen oder zu wenig weißen Farbanteil enthalten, werden ignoriert. Wird das Kriterium für genügend weiße Pixel im Rechteck erfüllt, so findet eine tiefere Analyse statt. Da die Torpfosten immer senkrecht zum Spielfeld im Raum stehen, wird die Region über den relevanten Rechtecken geprüft. Diese Prüfung erfolgt über den relativen Farbanteil von weißen Pixeln in einem Rechteck und zwar solange, bis der Horizont erreicht ist. Die Tore sind höher als die Spielfeldkontur und ragen über den Horizont hinaus. Deshalb müssen überwiegend weiße Regionen bis zum Horizont existieren. Mit einer



Abbildung 12: Suchen und Finden der Tore

Analyse der Bildregion unter der Feldkontur wird der Basispunkt eines Pfostens bestimmt.

Die Ermittlung vom linken und rechten Torpfosten findet über die Position der zwei gefundenen Torpfosten statt. Sind diese ermittelt, wird ein Tor modelliert.

4.4 Erkennung von mehrfarbigen Objekten

Die Erkennung von mehrfarbigen Objekten ist nicht so trivial wie die Suche nach einfärbigen Objekten. Eine einfache Suche nach Farbansammlungen reicht nicht aus. Objekte können sich aus mehreren gleichfarbigen und unterschiedlichen Farbansammlungen zusammensetzen.

Durch die geänderten Regeln der Kid-Size League des RoboCups im Jahr 2015, wird die Schwierigkeit der Erkennung von mehrfarbigen Objekten anschaulich. Anstatt auf einem flachen, grünen Teppich wird auf einem FIFA zertifizierten Kunstrasen gespielt. Dieser fängt das Licht je nach Wuchsrichtung unterschiedlich auf, wodurch der Kunstrasen teils große Helligkeitsunterschiede aufweist. Darüber hinaus sind die ehemals gelben Tore nun weiß.

Die schwierigste Aufgabe im Bereich der Objekterkennung ist jedoch die Regeländerung für den Ball. Der Ball ist nicht mehr einfärbig orange, sondern muss zu mindestens 50 % weiß sein. Die restlichen Farben sind unbestimmt, wobei der Großteil der restlichen Fläche des Balls weder Cyan, Magenta noch Grün sein darf. Der eingesetzte Fußball im RoboCup orientiert sich oft an den jeweils aktuell offiziellen Fußbällen der internationalen Turniere FIFA-Weltmeisterschaft und UEFA-Europameisterschaft [13].



(a) RoboCup 2015 in China



(b) German Open 2015 in Magdeburg



(c) RoboCup 2016 in Leipzig

Abbildung 13: Unterschiedliche Bälle

Die Abgrenzung zwischen Feldlinien, Toren und Ball ist nicht mehr allein über die Suche nach Farbansammlungen möglich. Eine Methode zur Feststellung geeigneter Ballkandidaten und einer genauen Klassifizierung ist erforderlich.

4.4.1 Kandidatenfindung

Ähnlich wie in Kapitel 4.2 werden über farbklassifizierte Integralbilder Farbansammlungen gesucht. Aufgrund der Tatsache, dass sich die auftretenden Farben im RoboCup im Grunde auf Weiß und Grün beschränken, ist die Ballerkennung nicht ausschließlich über die in Kapitel 4.3.1 vorgestellte Methode einsetzbar. Viele weiße Farbansammlungen wie z.B. Feldlinien, Mittel- und Elfmeterpunkt sowie Tore und Ball befinden sich auf dem Spielfeld (Abbildung 14a). Um diese Ansammlungen zu trennen reicht eine Suche nach den Grenzen der Farbansammlungen nicht aus, da diese vielfach im Bild auftreten können.

Zunächst wird das Bild per Shrink-Methode geschrumpft, so dass im äußeren Bereich keine weißen Pixel vorhanden sind. Dieser Bildausschnitt deckt aller Voraussicht nach einen großen Bereich des ursprünglichen Bildes ab, da über-

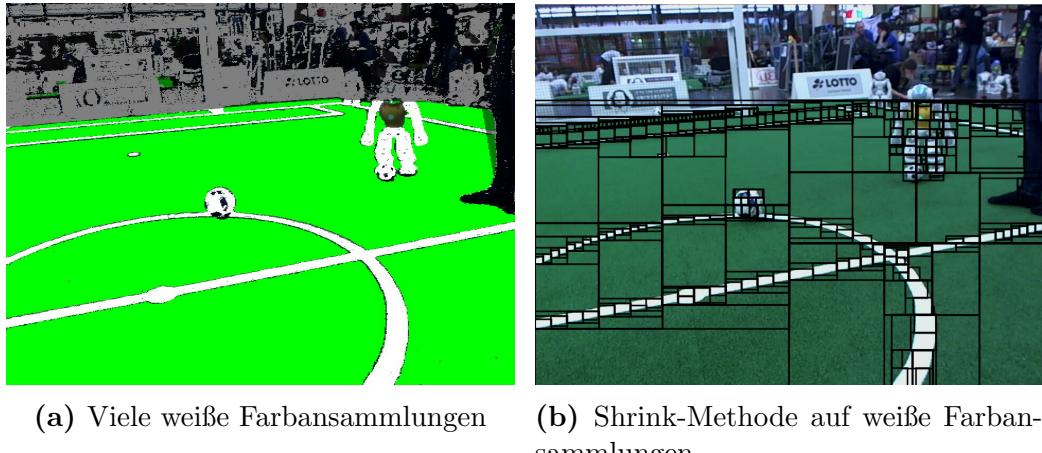


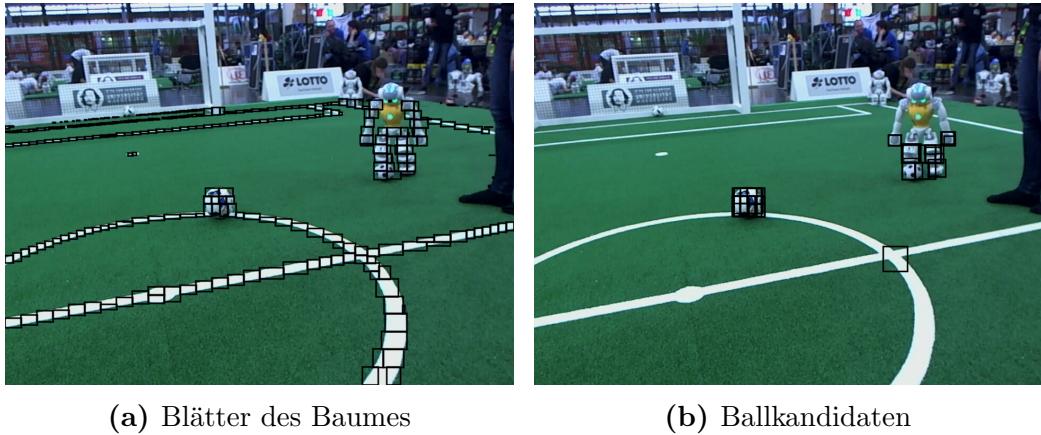
Abbildung 14: Schwierige Aufgabe: Erkennen eines überwiegend weißen Balls

all im Bild weiße Pixel vorzufinden sind. Er wird als Wurzelknoten bezeichnet und in einer Baumstruktur abgespeichert. Diese Bounding-Box wird in der Mitte der längeren Seiten geteilt. Die so entstandenen zwei neuen Bounding-Boxen werden per Shrink-Methode verkleinert. Wird entlang der Trennung des Wurzelknotens in keinem der Bounding-Boxen die Kante verschoben, so sind sie adjazent und bilden eine Farbansammlung. Wird auch nur eine Bounding-Box entlang der Trennung geschrumpft, so sind diese Bounding-Boxen nicht adjazent und es sind mindestens zwei Farbansammlungen im Gesamtbild vorhanden. Die entstandenen Bounding-Boxen sind Kindknoten des Wurzelknotens. Die Schritte des Teilens und Schrumpfens werden solange auf die Kindknoten ausgeführt, bis die Bounding-Boxen vorher definierte Kriterien erfüllen.

Diese Kriterien sind:

- Mindestanzahl an weißen Pixeln
- Relativer Mindestanteil an weißen Pixeln
- Relativer Maximalanteil an grünen Pixeln
- Minimale Fläche, die ein Pixel abdeckt

Je nach Strenge der Kriterien entsteht ein ungewichteter binärer Baum (Abbildung 14b) mit vielen oder wenigen Blättern (Abbildung 15a). Sind Blätter adjazent, so werden diese zusammengefügt. Dabei wird ein Rechteck über die adjazenten Blätter, von minimaler zu maximaler x - und y -Koordinate der Blätter, gespannt. Diese Rechtecke sind vorläufige Kandidaten.

**Abbildung 15:** Blätter und Ballkandidaten

Die vorläufigen Kandidaten werden auf mehrere Merkmale untersucht. Nur wenn sie einer Kette von unterschiedlichen Überprüfungen standhalten, werden sie als mögliche Ballkandidaten (Abbildung 15b) in Betracht gezogen. Mit jeder Überprüfung werden die Kriterien strenger, um frühst möglich Kandidaten aussortieren zu können.

Zunächst werden die Kandidaten anhand ihrer minimalen Seitenlänge untersucht. Sind Seiten zu kurz, kommen Kandidaten nicht weiter als möglicher Ball in Frage. Dies ergibt sich daraus, dass die Größe des Balls bekannt ist.

Danach werden die Kandidaten auf ihr Seitenverhältnis überprüft (Kapitel 4.3.1).

Die Verschiebung der Helligkeit von unten nach oben ist eine weitere Eigenschaft eines Balls, der den Rasen berührt. Der Ball ist in Bodennähe dunkler und wird nach oben immer heller (Abbildung 16). Zur Untersuchung dieser Eigenschaft bedarf es einer effizienten Methode der Überprüfung. Auch hier eignen sich Integralbilder (siehe Viola und Jones). Zusätzlich zu den farbklassifizierten Integralbildern werden YCbCr-Integralbilder erstellt. Es werden die einzelnen Y-, Cb- und Cr- Werte aufaddiert.

Die Helligkeitsverschiebung wird anhand des Y-Wertes im Y-Integralbild ermittelt. Hierzu wird der Kandidat in eine obere und eine untere Hälfte geteilt. Die Eigenschaft der Integralbilder, einen beliebigen Bildausschnitt auf seinen Durchschnittswerte zu überprüfen, ist hier von Vorteil. Mit jeweils vier Speicherzugriffen pro Kandidatenhälfte wird ermittelt, wie sich die Helligkeit unterscheidet. Ist die obere Hälfte nicht heller als die untere, so ist davon auszugehen, dass der Kandidat keinen Ball enthält. Ein Spezialfall ist ein Ball in unmittelbarer Nähe der Kamera. Ist die Kamera von oben auf



(a) Y-Channel



(b) Durchschnittliche Y-Werte der oberen und unteren Hälften

Abbildung 16: Helligkeitsunterschied eines Balls

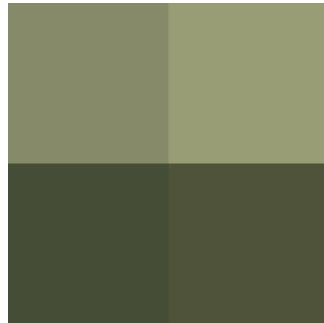
den Ball gerichtet, so ändert sich die Eigenschaft der Helligkeitsverteilung des Balls. In der Mitte des Balls ist ein heller Kreis, welcher nach außen immer dunkler wird. Die Entfernung des Balls ist ausschlaggebend, auf welche Helligkeitsverteilung des Balls geprüft wird. Die Entfernung wird über das Perspective-n-Point Verfahren berechnet.

4.4.2 Klassifizierung mit der Kullback-Leibler Divergenz

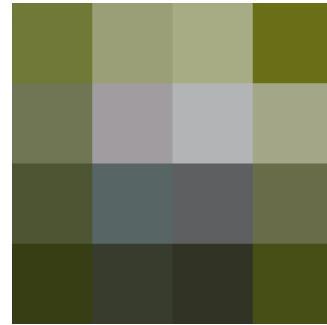
Die endgültige Klassifizierung eines Balls findet erst statt, wenn alle Kriterien zum Kandidaten erfüllt sind. Zur Klassifizierung werden im Vorfeld Beispieldbilder vom Spielball aufgenommen. Diese exemplarischen Bilder dienen als Referenz und bilden die Stichprobe P des Balls in der KL Divergenz. Es erfolgt ein Abgleich der Kandidaten mit dieser Stichprobe.

Die Referenzbilder werden mit einem $n \times m$ Raster belegt, d. h. das Referenzbild wird in n Spalten und m Zeilen unterteilt. Anhand der YCbCr-Integralbilder wird aus jedem Feld des Rasters der durchschnittliche Y-, Cb- und Cr-Wert der Pixel berechnet. Ein Tripel aus Y, Cb und Cr stellt eine Stichprobe dar.

Die gewählte Rastergröße ist nicht unwesentlich. Eine kleine Rastergröße von 2x2 (Abbildung 17a) reduziert die Farbinformationen der Pixel eines Kandidaten auf lediglich vier ($k = 4$) Farbtripel. Dies führt zu einem hohen Verlust an Informationen.



(a) 2x2 Raster



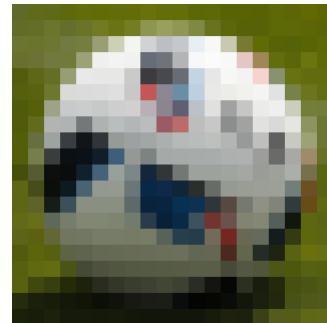
(b) 4x4 Raster

Abbildung 17: Kleine Rastergrößen

Bei einem 4x4 Raster (Abbildung 17b) werden dem Referenzbild schon 16 ($k = 16$) Stichproben entnommen. Entsprechend größere Raster erhöhen die Genauigkeit und stellen einen geringeren Informationsverlust dar (Abbildung 18).



(a) 10x10 Raster



(b) 20x20 Raster

Abbildung 18: Größere Raster erhöhen die Genauigkeit

Diese Stichproben dienen zur Berechnung der allgemeinen multivariaten Normalverteilung

$$f_X(x) = \frac{1}{\sqrt{(2\pi)^p |\Sigma|}} \exp \left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\}. \quad (13)$$

Y, Cb und Cr sind die drei Dimensionen der Verteilung, deshalb ist $p = 3$. Weiterhin sind

$$X = \begin{bmatrix} y_1 & y_2 & \dots & y_k \\ cb_1 & cb_2 & \dots & cb_k \\ cr_1 & cr_2 & \dots & cr_k \end{bmatrix}, \Sigma = \text{Cov}(X), |\Sigma| = \det(\text{Cov}(X)) \text{ und } \mu = \begin{bmatrix} \bar{y} \\ \bar{cb} \\ \bar{cr} \end{bmatrix}.$$

Die Kandidaten werden mit einem Raster gleicher Größe belegt. Für die KL Divergenz (Formel 10) ergibt sich für P die multivariate Normalverteilung der aufgenommenen Stichprobe des Balls und für Q die Stichprobe des Kandidaten.

Der Wert der KL Divergenz ist ein Maß für die Ähnlichkeit der beiden Stichproben. Ein Wert von Null würde bedeuten, dass der Kandidat vollkommen (Formel 12) mit dem Referenzbild übereinstimmt. Da jedoch der Wert praktisch immer größer Null ist (Formel 11), muss ein geeigneter Schwellenwert definiert werden, ab dem der Kandidat zu großen Abweichungen von der Stichprobe aufweist.

Die eingesetzte Ballklassifizierung findet in drei Schritten statt. Es kommen drei Rastergrößen (4x4, 8x8, 10x10) zum Einsatz. Aufgrund der steigenden Genauigkeit erhöht sich die Berechnungszeit. Ziel ist es mit jeder Rastergröße Kandidaten auszusortieren, um die Berechnungszeit gering zu halten. Nur wenn alle Kandidaten die drei Schritte positiv durchlaufen, wird der Kandidat zum Ball (Abbildung 19).



Abbildung 19: Gefundener überwiegend weißer Balls

Befinden sich mehrere Bälle auf dem Spielfeld, so dient der Divergenzwert auch als eine qualitative Bewertung. Der Ball mit dem kleinsten Divergenzwert ist der bevorzugte Spielball.

5 Auswertung

Das Ziel der Auswertung ist, ein repräsentatives Ergebnis über die Laufzeiten der eingesetzten Methoden und die Erkennungsrate des überwiegend weißen Balls zu erhalten. Hierzu wurden die beschriebenen Methoden (Kapitel 4) auf den Robotern der FUMANOIDS getestet und bewertet.

Ferner werden die Vor- und Nachteile der beschriebenen Objekterkennung dargelegt.

5.1 Laufzeitanalyse

Für die Analyse der Laufzeit wurde versucht, eine typische Spielsituation herzustellen. Die Laufzeitmessung fand auf einem 2016er Roboter statt. Auf den RoboCup German Open 2017 in Magdeburg wurde der Roboter an mehrere unterschiedliche Stellen des Spielfelds gestellt. Dabei befand sich der Ball (Abbildung 13c) zunächst immer auf dem Mittelpunkt. Sobald der Roboter den Ball fand, ist dieser autonom auf den Ball zugelaufen und dribbelte ihn zum Tor.

Betrachtet wurden nur die in dieser Arbeit vorgestellten Algorithmen. Dazu zählen die Berechnungen der farbklassifizierten Integralbilder sowie der YCbCr-Integralbilder, die Suche von Feldkontur, Toren und Ballkandidaten auf Grundlage der Shrink-Methode und die Klassifizierung des Balls per KL Divergenz. Der Durchschnitt der gemessenen Laufzeiten über die zehn Versuche ist in Tabelle 1 abgebildet.

	$\bar{\Omega}$ Laufzeit in ms
farbkl. Integralbilder	7,550
YCbCr-Integralbilder	2,640
Feldkontur	0,126
Tore	0,562
Ballkandidaten	0,745
Ballklassifizierung	1,890
Summe	13,513

Tabelle 1: Durchschnittliche Laufzeit der Objekterkennung

Die durchschnittliche Laufzeit von 13,513 ms erlaubt theoretisch eine Berechnung von ca. 74 Bildern pro Sekunde. In der Praxis wurde dieser Wert nicht erreicht. Zum einen lieferte die Kamera nur 30 Bilder pro Sekunde und zum anderen könnten Schwankungen in der Laufzeit zu Ausreißern führen.

Daher wurde untersucht, wie stark die Laufzeiten der einzelnen Algorithmen schwanken. Hierzu wurde exemplarisch eine Sequenz von 300 Bildern gewählt. Die Sequenz stammte aus einer der oben beschriebenen Untersuchungen. Auf den ersten Bildern waren weder ein Ball, noch Tore zu sehen. Grafisch dargestellt sind die Laufzeitschwankungen in Abbildung 20.

Die Laufzeit der Feldkontur (Abbildung 20e) war relativ konstant. Dies lag daran, dass bei der Berechnung keine unerwarteten Fälle auftraten. Zur Ermittlung der Feldkontur musste kein rechenintensiver Algorithmus mit mehreren zu überprüfenden Kriterien durchlaufen werden.

Die Berechnungen der Tore (Abbildung 20f), der Ballkandidaten (Abbildung 20c) und der Ballklassifizierung (Abbildung 20d) schwankten recht stark.

Bei den Toren lag dies daran, dass nicht in jedem Bild ein Tor vorhanden war. Der Algorithmus suchte entlang der Feldkontur nach auffälligen Regionen. Waren keine vorhanden, so brach der Algorithmus ab. Bei auffälligen Regionen musste jedoch eine tiefere Analyse stattfinden und weitere Berechnungen durchgeführt werden. Ähnlich verhielt sich die Suche nach den Ballkandidaten. So konnte es vorkommen, dass sowohl Tore, Feldlinien, Mittel- und Elfmeterpunkt sowie der Ball im Bild gemeinsam zu sehen waren. Hierdurch wurden viele Shrink-Operationen zur Suche nach weißen Farbansammlungen durchgeführt. Jede dieser Farbansammlungen musste auf die beschriebenen Kriterien (Kapitel 4.4.1) untersucht werden.

Schließlich konnte es vorkommen, dass eine Vielzahl an Kandidaten mit der KL Divergenz überprüft werden musste.

Die Schwankungen und Ausreißer der Laufzeiten für die Tore, die Kandidaten und die abschließende Klassifizierung sind somit abhängig vom Kamerabild und lassen sich erklären.

Auffällig war die teils große Diskrepanz der Laufzeiten bei der Berechnung der Integralbilder. Während die Berechnung der YCbCr-Integralbilder relativ konstant blieb (Abbildung 20b), waren die Unterschiede der Laufzeiten bei den farbklassifizierten Integralbildern (Abbildung 20a) recht häufig. Eine Erklärung wäre, dass acht farbklassifizierte Integralbilder gegenüber drei Integralbildern bei YCbCr berechnet werden mussten. Es könnte häufiger zu „Cache Misses“ kommen, da die Abarbeitung der Module auch von anderen Prozessen und dem Linux-Scheduler abhängig sind.

5.2 Erkennungsrate

Für die Erkennungsrate wurden verschiedene Szenarien untersucht. Die Objekterkennung bezieht sich hier jedoch nur auf den überwiegend weißen Ball.

5.2 Erkennungsrate

Michael Thio Pluhatsch

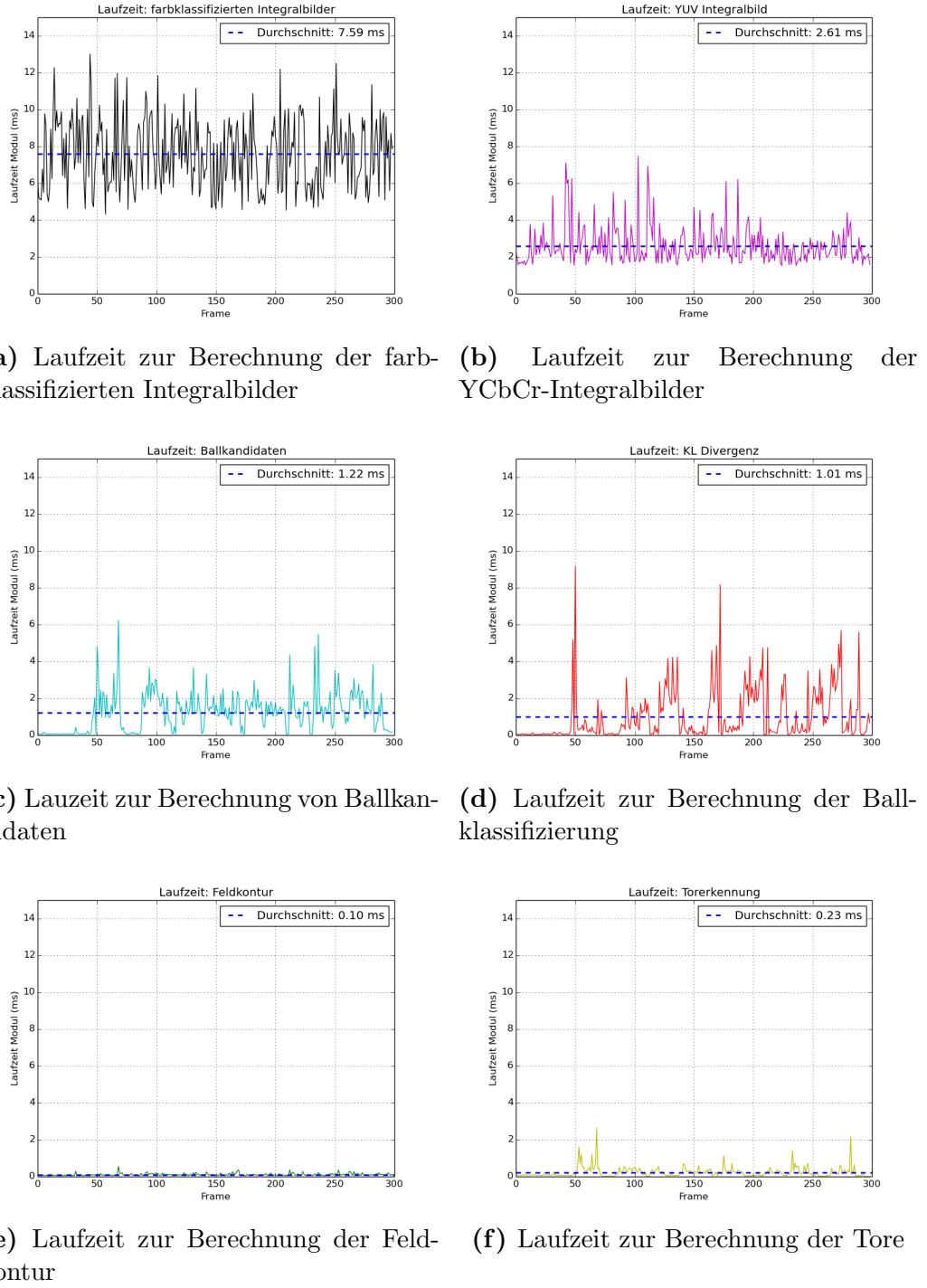


Abbildung 20: Laufzeiten der relevanten Module

Jedes Bild mit einem Ball wurde annotiert. Ein Ball galt als erkannt, wenn er überwiegend mit einem Rechteck umrahmt war.

Die Messungen konzentrierten sich auf die Klassifizierung und die Auswirkungen von unterschiedlichen Schwellenwerten der KL Divergenz auf die Erkennungsrate⁴. Für die Erkennung des Balls wurden unterschiedliche Rastergrößen benutzt. Diese hatten die Größe 4x4, 8x8 und 10x10. Die Schwellenwerte der KL Divergenz für alle Rastergrößen wurde zur besseren Vergleichbarkeit gleich gesetzt. Der Ball galt als klassifiziert, wenn jede einzelne Überprüfung positiv war. Für die Messung waren folgende Aspekte von Bedeutung:

- Richtig erkannte Bälle (richtig positiv)
- Scheinbar richtig erkannte Bälle (falsch positiv)
- Bälle, die nicht erkannt wurden (falsch negativ)

Bei der ersten Messung wurde ein ruhender Ball untersucht. Die Kamera wurde auf den zwei Meter entfernten Ball gerichtet und zeigte nichts weiter als den Ball auf dem Rasen.

Ruhender Ball: insg. 300 Bilder, 300 Bilder mit Ball						
Schwellenwert	0,001		0,0005		0,0001	
Erkannt	#	%	#	%	#	%
richtig positiv	300	100	299	99	244	71
falsch negativ	0	0	1	1	86	19
falsch positiv	0	0	0	0	0	0

Tabelle 2: Erkennung eines ruhenden Balls

Die Erkennung von ruhenden Bällen in dieser idealen Umgebung war bei den untersuchten Schwellenwerten konstant hoch.

⁴Gerundet wurde immer zu Ungunsten der Erkennung.

Bei der zweiten Messung wurde der Roboter auf eine grüne Wiese gestellt. Der Ball ist bis auf eine Entfernung von circa fünf Metern gerollt worden.

Rollender Ball: insg. 40 Bilder, 40 Bilder mit Ball						
Schwellenwert	0,001		0,0005		0,0001	
Erkannt	#	%	#	%	#	%
richtig positiv	31	77	25	62	9	22
falsch negativ	9	23	15	38	31	78
falsch positiv	0	0	0	0	0	0

Tabelle 3: Erkennung eines rollenden Balls

Die unterschiedlichen Schwellenwerte hatten einen erheblichen Einfluss auf die Erkennung. Ein niedriger Wert führte zu weniger richtig erkannten Bällen. Dennoch konnte ein sich bewegender Ball, auch außerhalb des RoboCup-Umfelds, erkannt werden.

Für die letzte Messung spielte der Roboter autonom auf einem Spielfeld des RoboCups. Auf dem Rasen befanden sich ein weißer Roboter⁵ und ein Mensch. In der ausgewählten Sequenz schaute der Roboter abwechselnd zum Ball, zum Tor, zum Menschen und zum weißen Roboter.

Zum Ball laufender Roboter: insg. 300 Bilder, 125 Bilder mit Ball						
Schwellenwert	0,001		0,0005		0,0001	
Erkannt	#	%	#	%	#	%
richtig positiv	112	89	100	80	21	16
falsch negativ	13	11	25	20	104	84
falsch positiv	8	3	0	0	0	0

Tabelle 4: Erkennung eines Balls, während der Roboter zum Ball läuft

Die Auswertung für den autonom spielenden Roboter zeigte, dass ein hoher Schwellenwert sich positiv auf die Erkennung von Bällen auswirkt, jedoch die Zahl der falsch positiven Erkennungen ebenfalls zunimmt. Der gegenteilige Effekt trat bei einem niedrigen Schwellenwert ein. Für ein optimales Ergebnis müssen die Schwellenwerte bestmöglich eingestellt werden.

⁵Ein NAO Roboter.

5.3 Vor- und Nachteile

Die vorgestellte Lösung zur Erkennung von Objekten ist schnell, robust und liefert eine theoretische und praktisch einsetzbare Objekterkennung in Echtzeit. Trotzdem birgt sie auch Nachteile.

Der größte Nachteil ist die Abhängigkeit vom Referenzbild und den gefundenen Kandidaten. Eine hinreichend gute Farbklassifizierung ist notwendig, um Kandidaten zu finden. Darüber hinaus führen Helligkeitsunterschiede zu abweichenden Erkennungsraten. Ist eine Spielfeldhälfte heller als die andere, so können aufgenommene Referenzbilder von einer Spielfeldseite nicht zwangsläufig für die andere verwendet werden. Der Einfluss der Helligkeit führt auch dazu, dass Referenzbilder möglichst zeitnah vor dem Einsatz der Objekterkennung aufgenommen werden sollten.

Ein weiterer Nachteil sind die eingesetzten Kameras. Trotz gleicher Modelle und den gleichen Einstellungen unterscheiden sich die aufgenommenen Bilder der Kameras teils stark voneinander (Abbildung 21). Eine Angleichung der Kamerabilder durch verschiedene Kameraparameter oder eine Farbklassifizierung für jede Kamera ist notwendig. Dementsprechend sind gegebenenfalls auch separate Referenzbilder aufzunehmen.



Abbildung 21: Helligkeitsunterschiede zweier baugleicher Kameras mit gleichen Einstellungen

Auch die Hyperparameter für die Klassifizierung per KL Divergenz sind problematisch. Ein zu klein eingestellter Schwellenwert verringert die positive Erkennungsraten. Ein zu großer Wert führt zu einer falsch positiven Erkennung.

Die Ermittlung des Schwellenwertes erfolgt anhand von Erfahrungswerten. Falsch positive Bälle dürfen nicht vorkommen. Eine geringere Erkennungsraten

te wird bewusst akzeptiert. Für akzeptable Ergebnisse sollte sie jedoch bei mindestens 50% liegen.

Trotz der angesprochenen Nachteile überwiegen die Vorteile der vorgestellten Objekterkennung. Sie liefert gute Ergebnisse bei gleichzeitig geringer Laufzeit. Die Farbklassifizierung und die Aufnahme von Referenzbildern kann schnell an die Umgebung angepasst werden. Notwendig sind in der Regel nicht mehr als zehn Referenzbilder. Die Hyperparameter sind zwar problematisch, die Anzahl ist jedoch überschaubar und leicht verständlich.

Darüber hinaus ist die Erkennung von weit entfernten Objekten möglich. Bei optimalen Bedingungen wurde der Ball zuverlässig auf zehn Meter erkannt. Die Reichweite der Erkennung wird durch die Auflösung der Kamera limitiert. Bei weit entfernten Objekten bilden zu wenige Pixel das Objekt ab. Die wenigen Pixel geben nicht genügend Farbinformationen wieder. Beachtet werden muss, dass eine höhere Auflösung zu längeren Berechnungen der Integralbilder führen. Dies führt zu einem Kompromiss zwischen Laufzeit und Reichweite der Objekterkennung. Für den Einsatz im RoboCup ist die eingesetzte Auflösung von 960x720 Pixeln ausreichend.

Die Algorithmen skalieren jedoch annähernd linear, weshalb die beschriebene Objekterkennung für höhere Auflösungen auf entsprechender Hardware einsetzbar wäre.

6 Fazit und Ausblick

In der vorliegenden Arbeit wird eine Objekterkennung vorgestellt, welche im überschaubaren Umfeld des RoboCups zum Einsatz kam. Dazu wurden die Grundlagen der Farbklassifizierung, der Integralbilder und der Kullback-Leibler Divergenz vorgestellt. Durch Verknüpfung dieser Grundlagen und der Einführung der Shrink-Methode wird gezeigt, wie unterschiedlich aussehende Objekte anhand ihrer markanten Merkmale wie Farbe, Größe und Lage bestimmbar sind.

Integralbilder werden vielfach in der Bildverarbeitung eingesetzt. Die Möglichkeit in konstanter Zeit beliebige Bildregionen zu analysieren ist für Echtzeitanwendungen bedeutsam. Der relative Farbanteil in Bildausschnitten und die Anzahl von Farbpixeln ist ein gutes Kriterium zum Bestimmen von markanten Merkmalen. Durch farbklassifizierte Integralbilder können diese Informationen in konstanter Zeit abgefragt werden. Die Shrink-Methode ist ein schnelles Verfahren, um markante Merkmale zu finden. Die Kullback-Leibler Divergenz ist eine zuverlässige Möglichkeit, die gefundenen Regionen mit vorher aufgenommenen Bildern zu vergleichen. Die Genauigkeit kann, mit höherem Rechenaufwand, bis auf einzelne Pixel gesteigert werden.

Die Objekterkennung wurde auf den Robotern der FUManoids eingesetzt und erzielte im Vergleich zur vorherigen Methoden (siehe Kriehs, Kapitel 3) bessere Ergebnisse. Die Spielfeld- und Torerkennung ist effizient und zuverlässig. Der einfarbige Ball wurde früher bis zu einer Entfernung von 3,5 Metern erkannt. Nun ist eine Erkennung über das gesamte Spielfeld⁶ möglich. Auch ein überwiegend weißer Ball wird zuverlässig über weitere Entfernung erkannt. Die Erkennungsrate ist ausreichend, um den derzeitigen Anforderungen des RoboCups zu genügen.

Die Laufzeitanalyse zeigt, dass die in dieser Arbeit geschaffene Objekterkennung in Echtzeit arbeitet und die angestrebten 30 Bilder pro Sekunde verarbeiten kann. Dabei ist die Auslastung der eingesetzten CPU nicht ausgeschöpft. Andere Module, die für das Laufen und das Verhalten zuständig sind können so weiterhin ausgeführt werden.

Die zeitkritische Berechnung der Integralbilder könnte in einem nächsten Schritt auf eine GPU ausgelagert werden. Des Weiteren könnte ein echtzeitfähiger Linux Kernel eingesetzt werden, um herauszufinden, ob dieser die Schwankungen in der Laufzeit stabilisiert und Ausreißer eliminiert.

Um den Aufwand der Kalibrierung der Objekterkennung zu minimieren, wären normalisierte Bilder interessant. Sie hätten die Vorteile, dass eine Farb-

⁶Spielfeldgröße: 10m x 6m

klassifizierung nur einmal vorgenommen werden muss und Helligkeitsschwankungen keinen Einfluss auf die Klassifizierung haben. Der Einsatz unter freiem Himmel wäre möglich. Referenzbilder von Objekten könnten ortsunabhängig aufgenommen werden.

Der Einsatz der Objekterkennung unter freiem Himmel könnte wiederum Szenarien außerhalb des eingeschränkten Umfelds des RoboCups interessant machen. So wäre möglicherweise ein Einsatz im Bereich des autonomen Fahrens vorstellbar. Verkehrsschilder heben sich oft durch ihre Farbe und Form vom Hintergrund ab. Können geeignete Bildausschnitte gefunden werden, so ist die Klassifizierung mit der Kullback-Leibler Divergenz vorstellbar.

Für die Suche von geeigneten Bildausschnitten könnte die Arbeit von Jan Draegert [28] über sogenannte PLANTs (partition-locating, axis-aligned, and node-queuing tree) interessant sein. Das von ihm beschriebene Verfahren beschleunigt die Suche nach Farbansammlungen und ist nicht auf farbklassifizierte Integralbilder angewiesen.

Literatur

- [1] Hiroaki Kitano, editor. *RoboCup-97: Robot Soccer World Cup I*, volume 1395 of *Lecture Notes in Computer Science*. Springer, 1998.
- [2] RoboCup. Offizieller Webauftitt des RoboCups. <http://www.robocup.org>, 2017. [Online; besucht 16-Juli-2017].
- [3] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. Robocup: The robot world cup initiative. In *Proceedings of the First International Conference on Autonomous Agents*, AGENTS '97, pages 340–347, New York, NY, USA, 1997. ACM.
- [4] Jan Draegert, Simon Gene Gottlieb, Michael Pluhatsch, Arne Schmidt, Till-Julius Krüger, Anahid Roshandel, Christopher Mühl, and Raúl Rojas. Berlin United - FUMANOIDs Team Description Paper for RoboCup 2017.
- [5] Reinhard Gerndt, Daniel Seifert, Jacky Hansjoerg Baltes, Soroush Sadeghnejad, and Sven Behnke. Humanoid robots in soccer: Robots versus humans in robocup 2050. *IEEE Robot. Automat. Mag.*, 22(3):147–154, 2015.
- [6] Tomas Akenine-Moller, Eric Haines, and Naty Hoffman. *Real-Time Rendering*. A. K. Peters, Ltd., Natick, MA, USA, 3rd edition, 2008.
- [7] Meiqing Wang and Choi-Hong Lai. *A Concise Introduction to Image Processing using C++*. Chapman and Hall/CRC, 2008.
- [8] ITU. Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios., October 1995.
- [9] Wen Gao and Siwei Ma. *Video Coding Basic Principle*, pages 17–34. Springer International Publishing, Cham, 2014.
- [10] Daniel Seifert. Automatische Farbkalibrierung fußballspielender Roboter. Diplomarbeit, Humboldt-Universität zu Berlin, September 2010.
- [11] Niklas Rughoeft. Maximum-Likelihood-Methode zur Farbklassifikation der „FUMANOID“-Roboter. Bachelorarbeit, Freie Universität Berlin, Januar 2015.

- [12] Franklin C. Crow. Summed-area tables for texture mapping. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, pages 207–212, New York, NY, USA, 1984. ACM.
- [13] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features, 2001.
- [14] Paul Viola and Michael Jones. Robust real-time object detection. In *International Journal of Computer Vision*, 2001.
- [15] Derek Bradley and Gerhard Roth. Adaptive thresholding using the integral image. *J. Graphics Tools*, 12(2):13–21, 2007.
- [16] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 1951.
- [17] Solomon Kullback. *Information Theory and Statistics*. Wiley, New York, 1959.
- [18] Naja von Schmude. Farb- und kantenbasierte Objekterkennung humanoider Roboter im RoboCup-Szenario. Diplomarbeit, Freie Universität Berlin, November 2011.
- [19] Hamid Moballegh, Naja von Schmude, and Raúl Rojas. Gradient vector gridding: An approach to shape-based object detection in robocup scenarios. In Thomas Röfer, N. Michael Mayer, Jesus Savage, and Uluç Saranli, editors, *RoboCup 2011: Robot Soccer World Cup XV*, volume 7416 of *Lecture Notes in Computer Science*, pages 162–173. Springer Berlin Heidelberg, 2012.
- [20] Hamid Mobalegh. *Development of an Autonomous Humanoid Robot Team*. PhD thesis, Freie Universität Berlin, December 2011.
- [21] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*, pages 886–893, 2005.
- [22] Max Losch. Formbasierte Ballerkennung für humanoide Fußball-Roboter. Bachelorarbeit, Freie Universität Berlin, September 2012.
- [23] Robert Kriehs. Entwurf und Implementierung einer kalibrierungsfreien Computer Vision für die FUManoids. Masterarbeit, Freie Universität Berlin, September 2013.

- [24] Daniel Speck, Pablo Barros, Cornelius Weber, and Stefan Wermter. Ball localization for robocup soccer using convolutional neural networks. In *RoboCup 2016: Robot World Cup XX*, Lecture Notes in Computer Science. Springer International Publishing, 2016.
- [25] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, EuroCOLT '95, pages 23–37, London, UK, UK, 1995. Springer-Verlag.
- [26] Thomas Kalinke and Werner von Seelen. *Kullback-Leibler Distanz als Maß zur Erkennung nicht rigider Objekte*, pages 501–508. Springer Berlin Heidelberg, Berlin, Heidelberg, 1997.
- [27] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.
- [28] Jan Draegert. Efficient superpixel creation in high-resolution images by applying a plant. Masterarbeit, Freie Universität Berlin, Juli 2017.