DIPLOMARBEIT

Echtzeit Objekterkennung auf low power Embedded Systems

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Diplom-Ingenieurs unter der Leitung von

O. Univ. Prof. Dipl.-Ing. Dr.techn. Dietmar Dietrich
und
Univ.Ass. Dipl.-Ing. Dr.techn. Stefan Mahlknecht
als verantwortlich mitwirkendem Universiätsassistenten am
Institutsnummer: 384
Institut für Computertechnik

eingereicht an der Technischen Universität Wien
Fakultät für Elektrotechnik und Informationstechnik

von

Roland Oberhammer
Mtr.Nr. 9526443
Johann Strauss Gasse 39/19, 1040 Wien

04.02.2007 _____

Abstract

Object detection is a broad area that comprises a variety of methods for measuring and categorising objects, through the use of various sensor systems. These sensor systems differ according to their environments, allowed lighting conditions and the type of objects that they can detect; they have advantages and disadvantages accordingly. This diploma thesis is about object detecion in the area of robotics, especially about its use in Robot-soccer - which is increasing in popularity due to the associated research possibilities. The designed robot is required to follow the rules of the Mirosot Category, specified by the international organisation for advancement of Robot-soccer called FIRA. A digital camera is the sensor used for object detection: Starting from constraints in the measurements (based on the Mirosot Category and Rules), this thesis presents an appropriate hardware-platform and the reasons for choosing particular components. Based on standard methods of image processing and approximations for their algorithmic complexity, a software concept was sought-after, that meets the requirements. The main aim is the detection of the ball (an orange golfball) and the coloured goalposts. Since standard methods are not optimized for the specific hardware, and the efficiency is a major concern, a concept is presented, which allows basic geometric shapes (e.g. circles, rectangles) to be detected and measured. This new concept partly uses well-known methods from the standard literature, that were optimized for the required tasks; but also includes a new approach that makes it possible to handle the detection-complexity of the dynamic motion of the target objects. The result is a image processing system with a detection-rate of 60 images per second. The precision of the ball-detection lies at +/- 1 mm at a mean distance of 50 cm. The average power consumption is 1,5 Watt and the hardware-dimensions are such, that the hardware could be housed in a cube with sides of 7.5cm length.

Kurzfassung

Objekterkennung ist ein weit gefächertes Gebiet und umfasst eine Vielzahl von Verfahren zur Vermessung und Kategorisierung von Objekten mittels unterschiedlichster Sensorsysteme, die je nach Umgebung, Beleuchtung und Art der Objekte die detektiert oder vermessen werden sollen, unterschiedlich geeignet sind und ensprechend ihre Vor- und Nachteile besitzen. Diese Diplomarbeit beschäftigt sich mit der Objekterkennung im Rahmen der Robotik, speziell im Roboterfußball, welcher sich in den letzten Jahren, aufgrund der sich daraus ergebenden Forschungsmöglichkeiten, immer größerer Beliebtheit erfreut. Der Roboter soll den Regeln der Mirosot Kategorie der internationalen Organisation zur Förderung des Roboterfußballs FIRA entsprechen. Als Sensor für die Objekterkennung dient eine Digitalkamera. So wird in dieser Arbeit ausgehend von den Einschränkungen in den Abmessungen, bestimmt durch die Regeln der Kategorie Mirosot, eine geeignete Hardware-Plattform erarbeitet und Gründe für den Entscheid zur Wahl bestimmter Komponenten dargelegt. Ausgehend von gängigen Methoden der Bildverarbeitung und der Aufwandsabschätzung für diese Algorithmen, wird nach einem Softwarekonzept gesucht, das auf der gewählten Plattform den gestellten Anforderungen entspricht. Ziel ist in erster Linie die Erkennung des Spielballes, ein oranger Golfball und der farbig markierten Tore. Da Standardverfahren aber nicht auf die spezielle Hardware optimiert sind und auch ansonsten hohe Anforderungen an die Rechenleistung stellen, wird ein Konzept vorgestellt, das es erlaubt einfache geometrische Formen wie Kreise oder Rechtecke zu detektieren und zu vermessen. Dieses neue Konzept greift teilweise auf in der Fachliteratur bekannte Methoden zurück die hierfür entsprechend optimiert werden, enthält aber auch einen neuen Ansatz, der es erlaubt den Anforderungen durch die hohe Dynamik der Bewegung der zu detektierenden Objekte Rechung zu tragen. So entstand ein Bilderkennungssystem mit einer Erkennungsrate von 60 Bildern pro Sekunde. Für den Spielball liegt die Genauigkeit bei +/- 1 mm in einer mittleren Entfernung von 50 cm. Dies bei einem durchschnittlichen Leistungsverbrauch von 1,5 Watt und Dimensionen der Hardware die in einem Würfel von 7,5 cm Kantenlänge Platz finden.

Danksagung

Ich möchte meinem Betreuer und meinen Freunden für ihre Unterstützung während des gesamten Studiums und der Diplomarbeit danken, besonders Herrn Stephan Tratter meinem „Studienorganisator“ ohne dem ich wohl manche Anmeldung versäumt hätte und dessen Unterlagen mir manche Prüfung erleichtert haben.

Ein besonderer Dank gilt meinen Eltern, natürlich für ihre Unterstützung sei es in finanzieller wie in jeglicher anderer Hinsicht, aber nicht zuletzt wegen ihrer schier unendlichen Geduld.

Abkürzungen

AD	Analog Devices (Hersteller von Halbleitern)
ADC	Analog to Digital Converter (Analog-Digital-Wandler)
AEC	Automatic Exposure Control (Automatische Einstellung der Belichtungszeit)
ALU	Arithmetic Logic Unit (Einheit in Mikrocomputern)
APS	Aktiv Pixel Sensor
AWB	Automatic White Balance (Automatischer Weißabgleich)
BGA	Ball Grid Array (Ein Gehäusetyp für Halbleiterbausteine)
BPS	Bilder pro Sekunde (Bildwiederholrate)
CIE	Commision Internationale de l'Eclairage
CCITT	International Telegraph and Telephone Consultative Committee
CCLK	Core-Clock
CPU	Central Processing Unit (Hauptprozessor)
DFT	Diskrete Fourier-Transformation
DMA	Direct Memory Access (Prozessorunabhängiger Datentransfer)
DSP	Digitaler Signalprozessor
EAV	End of Active Video
EBIU	External Bus Interface Unit (Controller für peripheren Adress- Datenbus im Blackfin)
FFT	Fast Fourier Transformation
FHS	Farbton Helligkeit Sättigung
FIRA	Federation of International Robosoccer Association
FPGA	Fiel Programmable Gate Array
FPS	Frames per Second (Bildwiederholrate)

FPU	Floating Point Unit (Fließkommaeinheit)
GPIO	General Purpose Input Output
HSB	Hue Saturation Brightness
HSI	Hue Saturation Intensity
HSV	Hue Saturation Value
IDE	Integrated Development Environment
IEEE	Institute of Electronics and Electrical Engineers (Weltweite Vereinigung von Ingenieuren aus dem Bereich Elektrotechnik)
IFP	Image Flow Processor
ISR	Interrupt Service Routine (Interruptbehandlungsfunktion)
ITU	International Telecommunication Union
JTAG	Jount Test Action Group (Eine Schnittstelle für das Hardware-Debugging)
MAC	Multiply Accumulate (Multiplikation und Addition)
MMR	Memory Mapped Register (Register die als Speicherzugriff zugänglich sind.)
MOS	Metal Oxide Semiconductor (Metalloxyd-Halbleiter)
NTSC	National Television System Committee
PAL	Phase Alternating Line
PCI	Peripheral Component Interconnect (Bus zur Verbindung von Geräten in PC-Systemen)
PDA	Personal Digital Assistant (Pocketcomputer)
PF	Purpose Flag (allgemeiner digitaler Ein- oder Ausgang)
PLL	Phase Locked Loop (Regelkreis zur Taktsynchronisation)
PPI	Parallel Peripheral Interface (Synchrone parallele Schnittstelle)
PWM	Puls Width Modulation (Puls-Pause-Modulation)
RAM	Random Access Memory
RISC	Reduced Instruction Set Computer (Eine Mikroprozessor Architektur)
RTC	Real Time Clock (Echtzeituhr)
SAV	Start of Active Video
SCCB	Serial Camera Control Bus (Steuerbus für Omnivision Kamerachips)
SCLK	System-Clock

SDRAM	Synchronous Dynamic RAM
SLD	Shortline Detection
SNR	Signal to Noise Ratio (Signal- zu Rauschverhältnis)
SPI	Serial Peripheral Interface (Synchrone serielle Schnittstelle)
SRAM	Static RAM (Speicherzellen realisiert mittels Flip-Flops)
UART	Universal Asynchronous Receiver Transmitter (Asynchrone serielle Schnittstelle)
USB	Universal Serial Bus
WDT	Watchdog Timer

Inhaltsverzeichnis

Kapitel 1 Aufgabenstellung

Objekterkennung ist ein weit gefächertes Gebiet und es gibt mittlerweile zahllose Verfahren mittels unterschiedlichster Sensoren zur Detektion und Vermessung von Objekten. Im vorliegenden Fall geht es um die Entwicklung eines passiven bildgebenden Verfahrens zur Entfernungsbestimmung eines autonomen mobilen Roboters von anderen Objekten. Andere Objekte heißt in diesem Fall, da der Roboter Fußball spielen soll, um den Ball die Tore und evt auch die Gegner. Die Herausforderung liegt in den kleinen Abmessungen des Roboters und der damit verbundenen eingeschränkten Verfügbarkeit von Energie und Rechenleistung. Ziel ist somit die Entwicklung eines „intelligenten Sensors“ der die Abstände des Roboters zu Objekten definierter Größe an einer Kommunikationsschnittstelle zur Verfügung stellt. Die Bildaufnahme soll mittels einer Videokamera erfolgen. Getestet wird dieser Sensor im Rahmen von Roboter Fußballspielen.

1.1 Roboterfußball

Fußball spielende Roboter erfreuen sich immer größerer Beliebtheit, da dieses Spiel ein breites Forschungsfeld bietet und unterschiedlichste Disziplinen und Forschungsschwerpunkte vereint. Nicht nur die Testmöglichkeiten unter „realen“ Bedingungen, sondern auch die hohe Dynamik des Spiels, das höchste Anforderungen an Sensorik und Mechanik stellt, bieten einen Anreiz sich diesem Thema zu widmen.

Es gibt derzeit zwei große konkurrierende Verbände die internationale Meisterschaften im Roboterfußball in den unterschiedlichsten Kategorien veranstalten. Es sind dies die FIRA und der Robocup.

1.1.1 FIRA

FIRA steht für „Federation of International Robosoccer Association“. Sie wurde 1995 von Prof. Jong-Hwan Kim gegründet und organisiert seither jährlich internationale Meisterschaften im Roboterfußball in unterschiedlichen Kategorien [FIRA06]. Die Ligen

reichen von einer reinen Simulationsliga bis hin zu humanoiden Robotern die in einem Elfmeterschiessen gegeneinander antreten.

Im folgenden wird die Liga in der der Roboter antreten soll, vorgestellt und die Regeln für die physikalischen Grenzen des Roboters erläutert, um zu zeigen, mit welchen Bedingungen die Objekterkennung zurecht kommen muss.

Mirosot:

Die Roboter der Mirosot-Liga besitzen in der derzeitigen Konfiguration kaum „Intelligenz“. Sie werden von einem externen Host-Rechner gesteuert der das gesamte Spielgeschehen mittels einer über dem Spielfeld montierten Kamera aufnimmt wie es in Abbildung 1.1 dargestellt ist. Jeder Roboter ist obenauf mit einer Farbkodierung gekennzeichnet um im von der globalen Kamera aufgenommenen Bild Gegner und die Spieler der eigenen Mannschaft unterscheiden zu können. Die gesamte Software zur Bildauswertung und Objekterkennung und auch jene Algorithmen zur Steuerung jedes einzelnen Roboters bzw. die „Strategie“ laufen auf einem handelsüblichen Desktop-Rechner. Dieser sendet nur mehr die Steuersignale an jeden einzelnen Roboter.

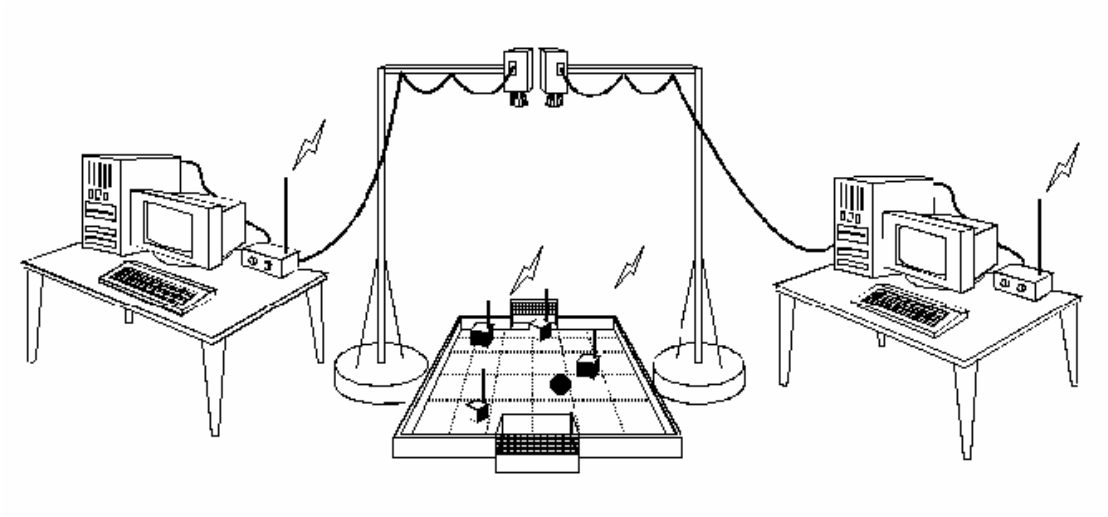

Abbildung 1.1: Aufbau des MIROSOT Systems

Die Roboter selbst sind würfelförmig mit einer Kantenlänge von maximal 75 mm. Sie werden von zwei unabhängigen Rädern an den Seiten angetrieben.

Gespielt wird mit einem orangen Golfball mit einem Durchmesser von ca. 43 mm [DGV04] auf grünem Untergrund. Die Abmessungen des Spielfelds hängen von der Unterkategorie ab. Derzeit werden Spiele mit 3, 5 oder 11 Spielern pro Mannschaft veranstaltet [FIRA06].

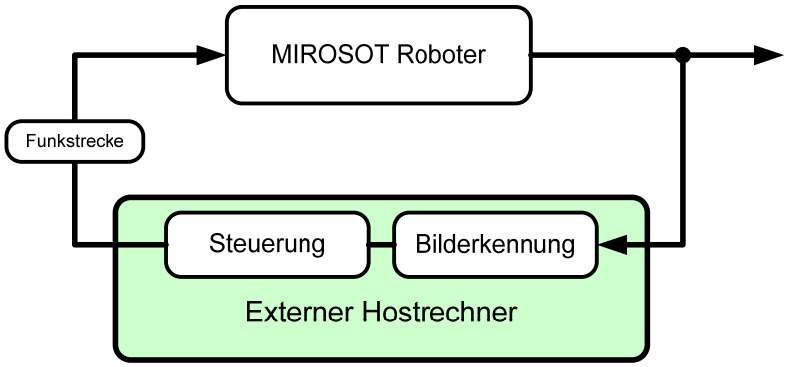

Abbildung 1.2: Regelschleife eines MIROSOT Roboters

Abbildung 1.2 zeigt ein prinzipielles Problem der Mirosot Roboter. Durch die Funkstrecke zwischen dem externen Host-Rechner und den Robotern entsteht eine Verzögerung die den Roboter entsprechend verspätet auf die im momentan aufgenommenen Kamerabild sich bietende Situation reagieren lässt. Dieser Umstand ließe sich umgehen, würde ein entsprechender Sensor direkt am Roboter montiert und dort eine Bildauswertung und Steuerung vorgenommen, beisbielsweise in besonders zeitkritischen Situation wie etwa den Abfangen sehr schnell geschossener Bälle, die Geschwindigkeiten von 5 m/s [NVK02] erreichen können.

Die Roboter selbst bewegen sich mit Geschwindigkeiten an die 3 m/s [NVK02].

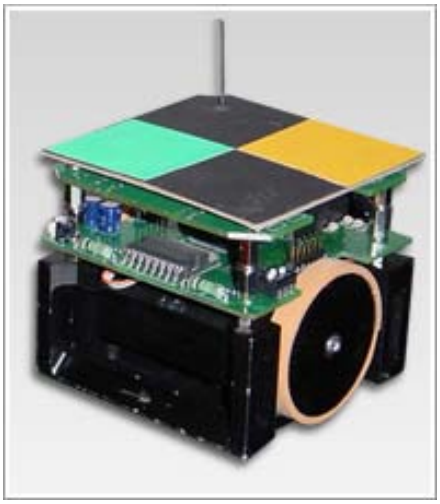

Abbildung 1.3: Typischer Mirosot Roboter

Die Verbesserung des Mirosot Systems soll jedoch nur ein Zwischenschritt sein auf dem Weg zum vollkommen autonomen Mirosot Roboter um aufzeigen zu können wie weit

fortgeschritten die Technologien im Bereich der „Embedded Systems“ schon sind und welche Entwicklungsstufen in den nächsten Jahren noch zu erwarten sind.

1.1.2 Robocup

Robocup ist der zweite große Veranstalter internationaler Turniere im Roboterfußball. Die Organisation hat ihre Anfänge im Jahr 1993 in Japan und wurde dort von einem japanischen Komitee angestoßen. Die ersten offiziellen internationalen Meisterschaften fanden 1997 statt [RBC06]. Die Kategorien des Robocup sind jenen der FIRA Meisterschaften sehr ähnlich.

Abbildung 1.4: Typischer Robocup Roboter

1.2 Anforderungen and das Vision System

In Titel dieses Abschnitts spreche ich zum ersten Mal vom „Vision System“. Dieser Begriff wird im Verlauf der Arbeit noch einige Male gebraucht werden. Gemeint ist damit die gesamte Einheit, Software und Hardware, die notwendig ist, um vom aufgenommenen Bild zu Positionskoordinaten von Objekten zu kommen, relativ zum Ursprung des Roboterkoordinatensystems. Der Begriff beinhaltet selbstverständlich auch den Aufnahmesensor selbst.

Die im Abschnitt 1.1 dargelegten Daten vermitteln bereits einen Eindruck, welch hohe Anforderungen an eine Objekterkennung basierend auf einem bildgebenden Verfahren gestellt werden.

sehr geringe Abmessungen:

Die Außenabmessungen dürfen eine Kantenlänge von **7.5 cm** nicht überschreiten

Leistungsverbrauch:

Ein Spiel dauert zwei mal fünf Minuten, die ein Roboter auf jeden Fall ohne Wiederaufladen der Akkus durchhalten muss. Wünschenswert wäre eine halbe Stunde, vielleicht mehr. Es steht ein NiMh Akku-Paket, bestehend aus 9 Zellen zu je 1.2 Volt mit 500 mAh zur Verfügung. In Serie geschaltet ergibt dies eine Nennspannung von 7.2 Volt [NVK02]. Seine Leistung ist durch die zur Verfügung stehenden Abmessungen begrenzt. Der Großteil davon (ca. 60 % bis 70 %) entfällt dabei auf die Motoren. Die Hardwareplattform für die Strategieeinheit und der Mikrokontroller für die Steuerung der Motoren benötigen ebenfalls einen Anteil an der zur Verfügung stehenden Leistung. Unter diesen Gesichtspunkten sollte der Leistungsverbrauch des Vision Systems **2 Watt** nicht übersteigen.

Dynamik des Spiels:

Wie im Abschnitt 1.1.1 erwähnt bewegt sich der Ball mit einer Geschwindikeit von an die 3 m/s. Nehmen wir als Beispiel eine Bildwiderholrate von 25 Bildern pro Sekunde wie sie etwa der europäischen Fernsehnorm entspricht, so zeigt eine einfache Rechnung, dass sich der Ball innerhalb zweier aufeinander folgender Bilder bei einer Ball-Geschwindigkeit von 5 m/s um 20 cm weiterbewegt hat. Dies reicht aus um ihn zwischen der Aufnahme der Bilder gänzlich „aus den Augen“ zu verlieren. Gewünscht ist also eine Bildwiederholrate von mindestens **50 Bildern pro Sekunde** (engl. *„frames per second“*), auch im Hinblick darauf einen Geschwindigkeitsvektor der Objektbewegung berechnen zu können, für die das Objekt für mindestens 2 im Zuge einer erhöhten Genauigkeit für 3 oder mehr Bilder im Blickfeld des Sensors sein muss. Nicht berücksichtigt ist dabei die Eigenbewegung des Roboters, sodass sich oben angenommene Ballgeschwindigkeit im schlimmsten Fall zur Geschwindigkeit des Roboters addiert.

Die hohe Dynamik birgt noch einen Nachteil der die Objekterkennung erschwert. In zwei aufeinander folgenden Bildern kann sich die Position des gesuchten Objekts im Bild vollständig ändern, sodass ein vermeintlicher Vorteil bei der Bilderkennung, nämlich dass ein einmal gefundenes Objekt sich im darauf folgenden Bild in der Umgebung der letzten bekannten Position befinden muss, hinfällig. Ein etwaiger daraus resultierender Geschwindigkeitsvorteil ist in vielen Spielsituationen nicht mehr vorhanden. Ob es trotzdem Vorteile bringt die Suche des Objekts bei einem Folgebild, auf die umliegenden Regionen der letzten detektierten Position zu konzentrieren, gilt es zu ermitteln.

Trägheit des Roboters:

Die hohe Dynamik des Spiels bringt einen weiteren Nachteil mit sich. Die Kamera kann immer nur die Szene, die sie gerade im Blickfeld hat, beobachten. Somit sind Vorgänge die hinter dem Roboter passieren nicht erfassbar, bzw. braucht der Roboter bei einer schnellen Ballbewegung die den Ball auf die von der Kamera abgewandte Seite spielt, eine Zeit bis er

sich umgedreht hat. Wünschenswert wären somit zwei Kameras eine in jede Fahrtrichtung des Roboters, sodass vielleicht in einem Suchmodus der Bereich vor und hinter dem Roboter abgedeckt werden kann, bzw. sollte der Ball in den „Rücken“ des Roboters gespielt werden, sich dieser nicht umdrehen muss, sondern durch ein einfaches und wesentlich weniger zeitaufwendiges Umschalten der Kamera, der Ball weiter verfolgt werden kann.

Entfernungen:

Die Abmessungen des Spielfeldes ändern sich je nach Anzahl der Roboter pro Mannschaft. Der Golfball ist das kleinste Objekt im Spiel, dieses sollte auf einer Entfernung von **einem Meter** noch erkennbar sein. Welche Auflösung für dieses Kriterium notwendig sein wird, gilt es zu bestimmen.

Lichtverhältnisse:

Die Regeln der FIRA schreiben die Stärke der Beleuchtung über dem Spielfeld nicht exakt vor. Vorgeschrieben ist eine Beleuchtungsstärke von mindestens 500 Lux. Empfohlen wird eine flimmerfreie Lichtquelle [FIRR06]. Trotzdem kann es aus eigener Erfahrung, von Spielfeld zu Spielfeld zu erheblichen Schwankungen in der Beleuchtungsstärke kommen. Die Bilderkennung soll zu Demonstrationszwecken aber auch unter Lichtverhältnissen wie sie etwa in Büros herrschen, funktionieren. Dies erfordert eine gewisse Robustheit gegenüber Helligkeitsschwankungen. Dadurch, dass die Beleuchtung nicht genau spezifiziert ist, und zwar nicht nur in Art und Stärke sondern vor allem auch in der Position, entfallen von vornehinein alle Verfahren zur Objektidentifikation die mit Reflexionen arbeiten.

Verdeckung:

Da der Blickwinkel der Kamera im Gegensatz zum System mit globaler Kamera parallel zum Spielfeld liegt, kann es bei den Objekten zu Verdeckungen kommen. Ein gewisser Grad an Verdeckung sollte zulässig sein. Zu bedenken gilt es an dieser Stelle auch, dass das gesucht Objekt sich erst in das Blickfeld bewegen bzw. sich daraus entfernen kann, wodurch es möglich ist, dass sich in manchen Bildern das gesuchte Objekt noch nicht vollständig im Blickfeld des Roboters liegt, was einer partiellen Verdeckung gleichkommt. Inwieweit es möglich ist, dies zu berücksichtigen, gilt es zu Untersuchen. Auch inwieweit die Genauigkeit der Erkennung mit dem Grad der Verdeckung verknüpft ist und ob sich diesbezüglich Aussagen treffen lassen, soll ermittelt werden.

Variable Objektgröße:

Ein wichtiger Faktor der bei einer Bilderkennung für bewegliche Roboter im Allgemeinen zu berücksichtigen ist, ist die unterschiedliche Größe der Objekte im Bild je nach Entfernung des Roboters zum Objekt von Interesse. Die ist insofern wichtig da dadurch keine statischen Modelle für die Objekterkennung verwendet werden können, da die Größe des Objekts im Bild nicht im vorhinein feststeht wie sich zeigen wird, scheiden manche Verfahren zur Objektidentifikation bereits aus.

Genauigkeit:

Bei einer Roboterbreite von 7.5 cm und einem Balldurchmesser von 43 mm bedeutet dies, dass die Fläche des Roboters mit der der Ball gespielt wird, nicht einmal doppelt so breit ist wie der Ball selbst. Um dem Ball bei einem Schuss auch noch eine Richtung geben zu können, muss er also mit einer sehr hohen Genauigkeit getroffen werden. Sehr hoch bedeutet in diesem Fall die Abweichung sollte nicht mehr als 10 bis 20 mm betragen um sicher zu stellen, dass dem Ball gezielt eine Richtung gegeben werden kann. Dies gilt für den Nahbereich, wenn es zur Kollision mit dem Ball kommt. Ist der Ball weiter vom Roboter entfernt, reicht auch eine ungenaure Ballposition für die Entscheidungsfindung der Strategie, da im Zuge der Bewegung des Roboters die Positionen der einzelnen Objekte laufend aktualisiert werden.

Die Anforderungen lassen sich somit wie folgt zusammenfassen:

Tabelle 1.1: Anforderungen an die Objekterkennung

Bildwiederholrate	mind. 50 bps / fps
Detektionsbereich	Ball bis einem Meter
Genauigkeit im Nahbereich	+/- 10 mm
Lichtverhältnisse	variabel
Verdeckung	möglich
Leistungsaufnahme	kleiner 2 Watt

Kapitel 2 Begriffsbildung

In der Fachliteratur gibt es im Rahmen der Bildverarbeitung eine Reihe von Begriffen, die oft auch fälschlich verwendet werden. Da uns die Bildverarbeitung auch im täglichen Leben immer wieder begegnet und sich viele Begriffe im täglichen Sprachgebrauch wieder finden möchte ich dem Leser vermitteln woher die Begriffe rund um die Bildverarbeitung stammen, und wie sie in dieser Arbeit verstanden und verwendet werden.

Zuallererst aber ein Hinweis über die Schreibweise von Zahlen in dieser Arbeit. Wenn nicht anders angegeben werden Zahlen immer in Dezimalschreibweise verstanden. Da in der Computertechnik jedoch das hexadezimale Zahlensystem sehr häufig verwendet wird, werden auch in dieser Arbeit Zahlen in diesem Zahlensystem gebraucht. Um Verwechslungen vorzubeugen, werden Hexadezimalzahlen immer mit einem vorangestellten „0x“ gekennzeichnet. Entlehnt wird diese Schreibweise aus der sehr weit verbreiteten Programmiersprache „C/C++“.

Da letztendlich immer der Mensch derjenige ist, der ein bearbeitetes Bild oder die Ergebnisse einer Bildverarbeitung interpretiert, sind viele Modelle wie etwa jene der Farbräume dem menschlichen Sehen entnommen oder zumindest an dasselbige angelehnt. Die Bild- oder Objekterkennung versucht letztendlich das menschliche Auge oder allgemeiner das menschliche Sehen als ganzes zu modellieren oder imitieren, wenn das derzeit auch nur bis zu einem gewissen Grad gelingen kann, da hinter dem menschlichen Sehen die Leistungsfähigkeit des menschlichen Gerhirnes sitzt, mit all seinem Wissen, seiner Lernfähigkeit und der Möglichkeit zur Abstraktion. Dinge, die nach heutigem Stand der Forschung auf dem Gebiet der künstlichen Intelligenz nur in sehr stark vereinfachten Modellen verstanden und umgesetzt werden können. Letztendlich ist die Modellierung des menschlichen Sehvermögens in seiner Gesamtheit das Fernziel der Entwicklungen auf dem Gebiet der künstlichen Intelligenz und der Bilderkennung. Beide Gebiete können nicht voneinander getrennt betrachtet werden.

Ich möchte darauf hinweisen, dass viele Begriffe oder Abkürzungen der Bildverarbeitung im allgemeinen Sprachgebrauch aber auch in der deutschen Fachliteratur auf englisch verwendet werden, weshalb ich in dieser Arbeit auch die englischen Begriffe aufführen werde und wo es

mir sinnvoll erscheint auch jene anstelle der deutschen im weiteren Verlauf der Arbeit verwenden werde.

2.1 Das menschliche Sehen

Das menschliche Sehen ist in seiner Gesamtheit sehr komplex und noch nicht gänzlich verstanden, da es eng verbunden ist mit der menschlichen „Intelligenz“. Die Interpretation der vom Auge stammenden Bildinformationen hängt sehr mit unseren Erfahrungen unserer Lernfähigkeit zusammen, umfasst also alle Bereiche des menschlichen Denkens. Es soll hier auch nur auf das Auge selbst eingegangen werden und anhand von dessen Aufbau die Aufnahme der Farb- und Helligkeitsinformationen erläutert werden, dessen Verständnis für mehr Anschaulichkeit bei vielen Begriffen der Bildverarbeitung beitragen kann.

2.1.1 Der Aufbau des Auges

Die Bildaufnahme und die Bildverarbeitung sind beim menschlichen Auge untrennbar verbunden, da bereits im Auge eine Vorverarbeitung stattfindet [STN93].

Das menschliche Auge hat einen Durchmesser von ca. 24 mm und liegt in einem Fettpolster eingebettet in der Augenhöhle geschützt durch die es umgebenden Schädelknochen.

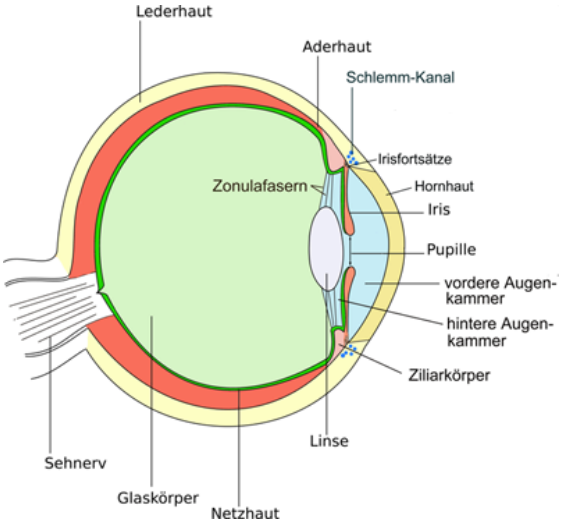

Abbildung 2.1: Querschnitt durch den Augapfel

Abbildung 2.1 zeigt einen Querschnitt des menschlichen Auges und beschreibt die einzelnen Teile. Die Hülle des Augapfels besteht aus 3 Schichten. Die äußerste ist die Lederhaut die vorne in die durchsichtige Hornhaut *(lateinische Bezeichnung Cornea)* übergeht. Hinter der Hornhaut liegt die Pupille, eine Lichtdurchlässige Öffnung, die durch eine Veränderung ihres

Durchmessers, eine Regulierung der Lichteintrittsmenge erlaubt und somit die Blende des menschlichen Auges darstellt. Direkt an die Pupille grenzt die Linse. Der rund ums sie anliegende Ziliarmuskel erlaubt eine Änderung der Linsenform, sodass damit die Lichtbrechung und letztendlich die Fokussierung verändert werden kann. Innen an die Lederhaut legt sich die Aderhaut, die für eine gute Durchblutung des Auges sorgt. Die innerste Schicht des Auges bildet die Netzhaut *(lateinische Bezeichnung Retina)*, die hinten in den Sehnerv übergeht, der die Reize der Netzhaut an das Gehirn weiterleitet. Licht fällt durch Hornhaut und Pupille und erzeugt im hinteren Teil der Netzhaut ein auf dem Kopf stehendes reelles Bild [BVN96, STN93].

Die Netzhaut:

Spezielle Zellen an der Netzhaut wandeln das eintreffende sichtbare Licht in elektrische Reize um, die über den Sehnerv ins Gehirn geleitet werden. Unterschiedliche Zellen reagieren dabei unterschiedlich stark auf Wellenlänge und Intensität des einfallenden Lichts. Man unterscheidet grundsätzlich zwei unterschiedliche Zellentypen an der Netzhaut: die Stäbchen und die Zäpfchen. Sie liegen in der äußersten Schicht der Netzhaut, darunter liegen die Ganglienzellen angedockt an den Nervenfasern die letztendlich in den Sehnerv münden [BVN96, STN93]. Abbildung 2.2 zeigt einen Querschnitt durch die Netzhaut mit der Lage der Stäbchen und Zäpfchen.

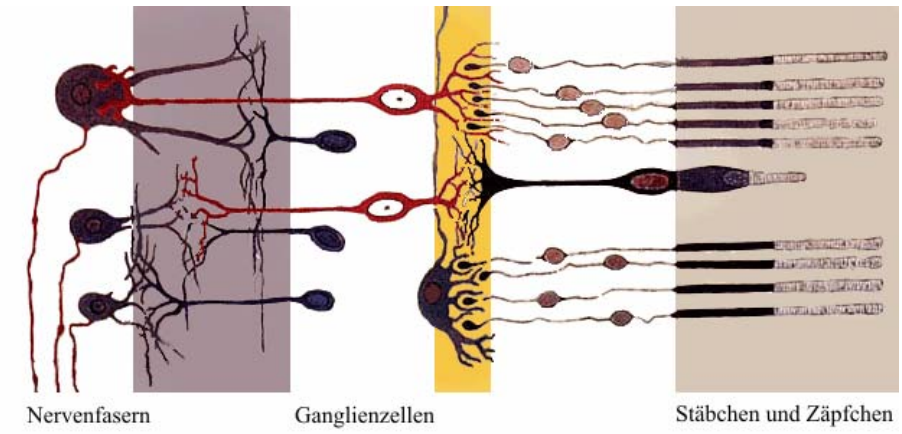

Abbildung 2.2: Querschnitt durch die Netzhaut

Die Zäpfchen:

Sie sind die Farbsensoren des Menschen. Sie reagieren also vorwiegend auf Unterschiede in der Wellenlänge des Lichts. Der Mensch besitzt drei verschiedene Arten von Zäpfchen, die sich in ihrer spektralen Empfindlichkeit unterscheiden, das heißt, jede Zapfenart hat ihre maximale Empfindlichkeit in einem anderen Wellenlängebereich. Somit unterteilt man die drei Arten nach dem Maximum der Empfindlichkeit in blaue, rote und grüne Zäpfchen. In der Abbildung 2.3 ist die spektrale Empfindlichkeit bzw. der Absorptionsgrad der drei Arten dargestellt. Die Ordinate stellt die normierte Empfindlichkeit in % dar. Die Kurven der drei

Zäpfchen *bz* für die blauen, *rz* für die rotempfindlichen und *gz* für die grünen, sind in dieser Grafik gleich stark gewichtet, was nicht ganz der Realität entspricht. Die blauen Zäpfchen weisen in Wahrheit eine etwas höhere Maximalempfindlichkeit auf [RDG93]. Für das Verständnis ist dies aber von geringerer Bedeutung. Die Zahlen am Scheitelpunkt der Kurven geben jeweils die Wellenlänge an, bei der der entsprechende Zäpfchentypus seine maximale Empfindlichkeit respektive den maximalen Absorptionsgrad hat. Deutlich erkennbar die starke Überlappung der rot- und grünempfindlichen Zäpfchen. Zum Gesamteindruck trägt aber nicht nur die Empfindlichkeit der drei Rezeptoren bei, sondern natürlich auch deren Verteilung auf der Netzhaut. So ist der Anteil der grünempfindlichen Zellen höher als der der blauen oder roten. Zusätzlich ist auch deren Verteilung auf der Netzhaut nicht konstant. Die größte Konzentration findet man im so genannten „Gelben Fleck“ [BVN96]. Die Summe dieser Faktoren bestimmt schlussendlich unser Farbsehen. In der Abbildung 2.4 ist die aufsummierte Gesamtempfindlichkeit für das Tag- und Nachtsehen in Abhängigkeit der Wellenlänge dargestellt. Tagsehen bedeutet in diesem Fall, die Beleuchtungsstärke reicht aus um die Zäpfchen anzuregen. Liegt sie unter einem bestimmten Schwellwert so spricht man von Nachtsehen, da nur mehr die Reize der Stäbchen (siehe unten) im Gehirn verarbeitet werden [RDG93]. In der Literatur wird die Hellempfindlichkeitskurve für Tagsehen auch als *fotoptische spektrale Empfindlichkeit* bezeichnet, jene für Nachtsehen als *skotopische spektrale Empfindlichkeit* [RDG93].

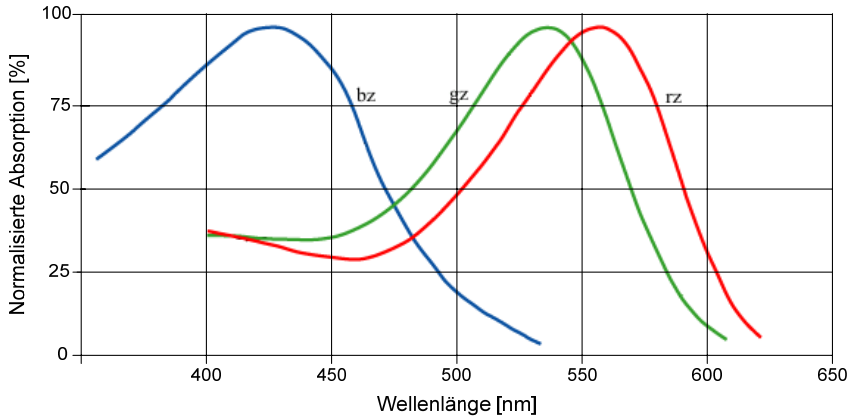

Abbildung 2.3: Spektrale Verteilung der Absorption der einzelnen Zäpfchentypen

Die maximale Empfindlichkeit für Tagsehen liegt bei etwa 555 nm. Das menschliche Auge besitzt somit für die Farbe „gelbgrün“ die größte Empfindlichkeit. Dieser wichtige Umstand findet in der Bildverarbeitung allgemein und auch im weiteren Verlauf dieser Arbeit Berücksichtigung.

Die Farbe „gelbgrün“ war in der frühen Periode der Menschheit die dominante Farbe bei der Suche nach pflanzlicher Nahrung. Eine Tatsache, die die Empfindlichkeit genau für diese Farbe evolutionstechnisch erklären könnte [RDG93].

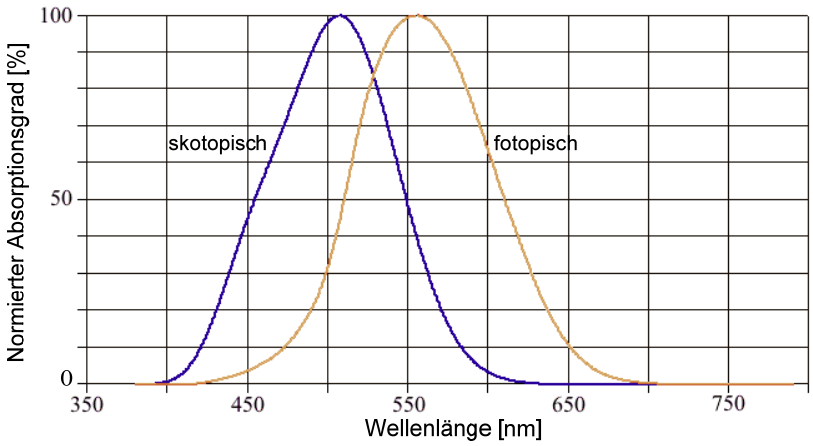

Abbildung 2.4: Normierte spektrale Empfindlichkeit für das Tag- und Nachtsehen

Die Stäbchen:

Liegt die Beleuchtungsstärke unter einem bestimmten Schwellwert, so reagieren die Zäpfchen nicht mehr in ausreichendem Maße auf eintreffendes Licht. An ihrer Stelle treten dann die Stäbchen die einerseits in größerer Zahl auftreten, ca. 75 bis 150 Millionen im Gegensatz zu den 6 bis 7 Millionen Zäpfchen [STN93], andererseits sind auch immer mehrere Stäbchen mit einem Nerv verbunden, im Durchschnitt an die 130 [STN93]. Diesen zwei Faktoren verdanken wir es, dass die Empfindlichkeit der Stäbchen um einige Größenordnungen höher als jene der Zäpfchen liegt, wodurch sehen auch unterhalb des Schwellwertes bei dem Zäpfchen noch angeregt werden, möglich ist. Da es von den Stäbchen aber nur einen Typus gibt ist in der Dunkelheit eine Farbunterscheidung nicht mehr möglich. Die unterschiedliche Anzahl und Verteilung bedingt aber auch, dass das menschliche Auge eine wesentlich bessere räumliche Auflösung für Helligkeits- als für Änderungen im Farbton hat. Ein weiterer Umstand der in der Bildverarbeitung Verwendung findet.

2.2 Farbton, Helligkeit und Sättigung

Die von den drei im Abschnitt 2.1.1 beschriebenen Farbdetektoren des menschlichen Auges ausgesendeten Signale werden unabhängig voneinander in das Gehirn gesendet und dort zu einem Gesamteindruck kombiniert und ergeben somit unser Farbempfinden [HBL89].

Der Mensch charakterisiert eine Farbe im Allgemeinen nach den drei Größen *Farbton*, *Helligkeit* und *Sättigung*. Diese Begriffe finden sich auch in der Bildverarbeitung wieder. Im Folgenden sollen die Begriffe nun erläutert werden. Sie werden im Zuge dieser Arbeit auch noch unter mathematischen Gesichtspunkten betrachtet werden. Da diese Begriffe immer wieder auch in der deutschen Fachliteratur in ihrer englischen Fassung verwendet werden, wird diese ebenfalls angeführt.

Farbton (engl. hue):

Der Farbton definiert die Art der Farbe als solches ohne Berücksichtigung der Helligkeit oder der Sättigung, z.B. Rotton oder Gelbton. Werden als Beispiel nur die grünen und roten Farbrezeptoren stimuliert, empfinden wird den Farbton als Gelb. Werden grüne und blaue Zäpfchen stimuliert sehen wir Cyan. Bei einem gleichmäßigen Reiz aller drei Typen empfindet man den Farbton weiß und je nach Intensität der Strahlung unterscheiden wir die unterschiedlichen Grautöne [RDG93].

Im allgemeinen Sprachgebrauch wird mit dem Begriff Farbe, meistens der Farbton gemeint.

Helligkeit (engl. brightness):

Bleibt das Verhältnis der Erregung der drei unterschiedlichen Zäpfchen konstant, aber die Stärke des auf die Netzhaut treffenden Lichts ändert sich, empfinden wir den durch das spektrale Verhältnis bestimmten Farbton als heller oder dunkler je nachdem ob die Lichtstärke zu- oder abnimmt. Bei den unbunten Farben, wie die Graustufen auch oft genannt werden, verschiebt sich die Empfindung von schwarz über unterschiedlich helles grau bis hin zu weiß [RDG93].

Sättigung (engl. saturation):

Die Sättigung einer Farbe beschreibt salopp ausgedrückt den Weißanteil der Farbe. Reine Spektralfarben, das heißt, Licht das aus nur einer Wellenlänge besteht auch monochromatisches Licht genannt, ist vollständig gesättigt. Es besitzt keinen Weißanteil. Weiß ist ungesättigt [RDG93]. Anders formuliert ist die Sättigung ein Maß für die spektrale Reinheit einer Farbe. Je breiter der Wellenlängebereich der zur farbempfindung beiträgt, desto geringer die Reinheit. Umgekehrt bedeutet ein schmaler Wellenlängebereich eine größere Reinheit und damit Sättigung [KSC96].

Die Sättigung wird in Prozent bzw. normiert mit einem Maximalwert von 1 angegeben. Liegt der Sättigungsgrad zwischen 10 und 20% so spricht man von *schwacher* Sättigung. Liegt sie zwischen 40 und 60% bezeichnet man sie als *stumpf* und eine als *kräftig* bezeichnete Farbe hat einen Sättigungsgrad von 80 bis 100% [RDG93]. Diese Angaben sind in Tabelle 1.1 noch einmal zusammengefasst.

Tabelle 2.1: Interpretation von Sättigungsgraden

Bezeichnung	Sättigungsgrad
schwach	10 bis 20%
stumpf	40 bis 60%
kräftig	80 bis 100%

Im weiteren Verlauf wird auch ein mathematischer Zugang zu den Begriffen im Sinne der Bildverarbeitung erarbeitet.

2.3 Farbräume und ihre Entstehung

Aus der Betrachtung der Funktionsweise des menschlichen Auges erkennt man, dass der Farbeindruck eine Kombination aus dem Grad der Anregung der drei unterschiedlichen Farbrezeptoren ist. Das heißt, fällt Licht auf die drei Zäpfchentypen wird von jedem ein seinem Spektralbereich entsprechender Reiz ins Gehirn übertragen, welches diese drei getrennten Reize zu einem Farbton kombiniert oder „mischt“.

2.3.1 Drei-Komponenten-Theorie

Diese Erkenntnis legt die Vorgehensweise nahe, den Farbraum als eine Kombination dreier linear unabhängiger Basisvektoren zu interpretieren, wie es im Folgenden dargestellt ist.

Als Strahlungsquelle dient ein monochromatischer Sender, der mit einer bestimmten Wellenlänge λ sendet. Seine *spektrale Farbvalenz* $\vec{F}(\lambda)$ kann dann wie folgt als ein Vektor im Farbraum dargestellt werden:

$$\vec{F}(\lambda) = A(\lambda)\vec{A} + B(\lambda)\vec{B} + C(\lambda)\vec{C} \tag{2.1}$$

Die drei Basisvektoren $\vec{A}$, $\vec{B}$ und $\vec{C}$ können dabei jeweils eine bestimmte Wellenlänge im Farbraum repräsentieren oder selbst wieder eine Vektorsumme darstellen. Im Allgemeinen wird ersteres verwendet. Die Komponenten A, B und C heißen *spektrale Farbwerte* und hängen von λ ab. Die Farbvalenz stellt sich somit als gewichtete Summe dreier linear unabhängiger Basisvektoren dar [RDG93].

2.3.2 Der RGB-Farbraum

Wie im vorigen Abschnitt beschrieben, ist die Wahl der Basisvektoren für den Farbraum nicht eindeutig. Nahe liegend wäre eine Verwendung der mittleren Wellenlängen der

Spektralbereiche Blau, Grün und Rot. Sie liegen für Rot bei 700 nm, 546 nm für Grün und für Blau bei 436 nm. Tut man dies, so spricht man vom RGB-Farbraum [RDG93]. Die Basisvektoren $\vec{A}$, $\vec{B}$ und $\vec{C}$ nennen wir dann $\vec{R}$ für Rot bei λ = 700 nm, $\vec{G}$ für Grün bei λ = 546 nm und $\vec{B}$ für Blau bei λ = 436 nm. Der Farbraum der damit aufgespannt wird nennt sich RGB-Raum und die Farbvalenz für diesen Raum berechnet sich wie folgt:

$$\vec{F}(\lambda) = R(\lambda)\vec{R} + G(\lambda)\vec{G} + B(\lambda)\vec{B}. \tag{2.2}$$

Für die *spektralen Farbwerte* gilt dabei:

$$R(\lambda) = L^R(\lambda)\bar{r}(\lambda),\quad G(\lambda) = L^R(\lambda)\bar{g}(\lambda),\quad B(\lambda) = L^R(\lambda)\bar{b}(\lambda).$$

$L^R(\lambda)$, die spektrale Strahlungsdichte beschreibt im Wesentlichen die abgestrahlte Leistung eines monochromatischen Senders bei der Wellenlänge λ pro Flächeneinheit. Sie wird mit der *spektralen Empfindlichkeit* der Quelle gewichtet. Abbildung 2.5 zeigt die drei *spektralen Empfindlichkeiten* $\bar{r}(\lambda)$, $\bar{g}(\lambda)$ und $\bar{b}(\lambda)$ der Quelle, wie sie für einen Standardbeobachter von der CIE (Commision Internationale de l'Eclairage) 1931 bzw. nach DIN5033 und noch einmal 1964, nach empirischer Ermittlung definiert worden sind.

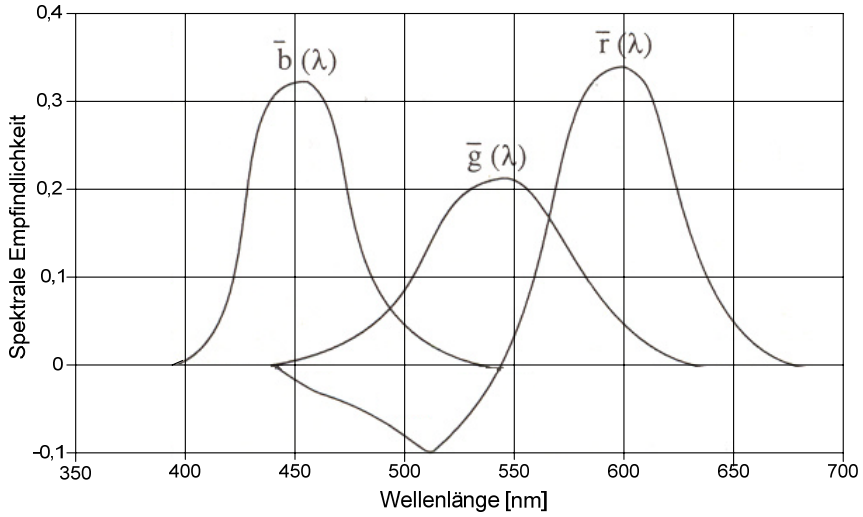

Abbildung 2.5: Normierte spektrale Empfindlichkeiten des RGB-Farbraums

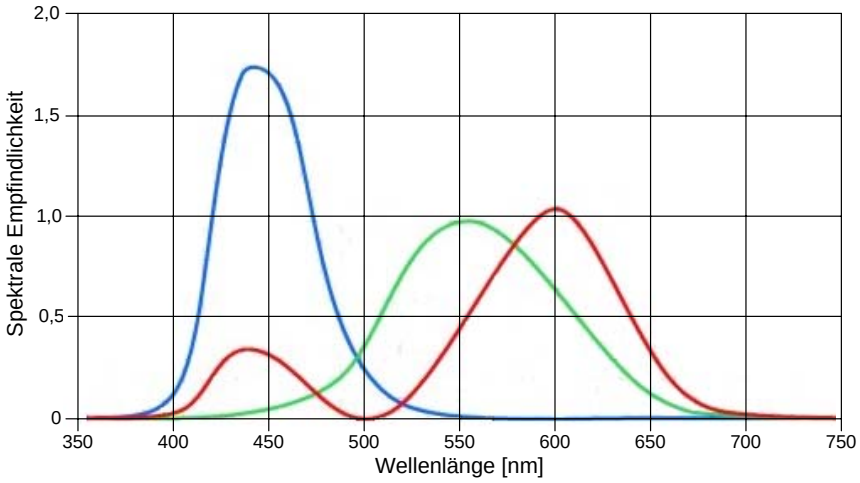

Abbildung 2.6: Normierte spektrale Empfindlichkeiten des XYZ-Farbraums

Deutlich erkennbar ist, dass $\bar{r}(\lambda)$ negative Werte annehmen kann, diese Farbwerte zwischen ca. 450 und 550 nm wären somit physikalisch nicht realisierbar. Durch lineare Transformation gelangt man aber zu drei neue Basisvektoren die keine negativen Werte mehr aufweisen, dieser neue Farbraum wird in der Literatur verallgemeinert als XYZ-Farbraum bezeichnet [RDG93].

Mit Hilfe dieses Farbmodells lassen sich nun alle in der Natur vorkommenden Farben darstellen. Dies stellt nur einen kleinen Auszug aus der Theorie der Farbenlehre dar und sollte nur als Hintergrundinformation dienen um zu einem besseren Verständnis zu kommen woher sich, die in heutigen Bildverarbeitungssystemen gebräuchlichen Farbdarstellungen, ableiten. Weiterführende Informationen finden sich in [FBS86].

2.3.3 Von der analogen in die digitale Welt

Die kontinuierlichen analogen Signale des Spektrums des sichtbaren Lichts müssen zur Bildverarbeitung in diskreter Form in der digitalen Welt repräsentiert werden. Dabei wird die von einem Aufnahmegerät aufgenommene Szene in einem Raster digitaler Bildpunkte umgesetzt, wobei jedem Bildpunkt eine Leuchtdichte zugeordnet wird. In der digitalen Bildverarbeitung bezeichnet man diesen Wert jedoch nicht als Leuchtdichte, da er nicht der physikalischen Interpretation des Begriffs, wie im obigen Abschnitt verwendet, entspricht, sondern als *Grauwert* (siehe folgende Abschnitte). Die digitale Repräsentation der aufgenommenen Szene nennen wir *Grauwertbild* oder *Graubild*. Der Grund für diese Bezeichnung wird aus den Erläuterungen der folgenden Abschnitte ersichtlich. Das Grauwertbild stellt somit eine Funktion $f(x,y)$ von Grauwerten dar. Bei farbigen Bildern wird jeder Bildpunkt als ein n-Tupel von Funktionen repräsentiert $(f_1(x,y), f_2(x,y),\ldots,f_n(x,y))$. Die Anzahl und Art der Funktionen sind dabei Abhängig vom verwendeten Farbmodell, welche in den folgenden Abschnitten beschrieben werden.

Der digitale RGB-Farbraum

In der praktischen Anwendung in digitalen Bildverarbeitungssystemen ist der verallgemeinerte XYZ-Farbraum ohne Einschränkung nicht unmittelbar verwendbar. In der digitalen Welt muss auf Quantisierung und darstellbarem Wertebereich geachtet werden. In der Praxis wird für die drei Basisvektoren R, G, und B jeweils ein Wertebereich von 0 bis 255 festgelegt. Dieser Bereich ist mittels 8 Bit Datenbreite darstellbar. Die Farbe eines jeden Bildpunktes $C(x, y)$ in einem dreikanaligen Farbbild C ist somit eine Funktion der drei Variablen Rot, Grün und Blau auch *Primärfarben* oder *Primärvalenzen* genannt [KSC96], die jeweils Werte zwischen 0 und 255 einnehmen. Die Variablen x und y beschreiben hierbei die Position des Punktes im Bild.

$$C(x, y) = (R, G, B) .$$

Es lassen sich durch oben genannte Einschränkungen nicht mehr alle in der Natur vorkommenden Farben darstellen. Bei drei Byte pro Bildpunkt sind aber immerhin noch 16 772 216 Farben unterscheidbar.

Stellt man den Farbraum als dreidimensionales Diagramm dar, wobei jede der Primärfarben eine Achse einnimmt, so gelangt man zum RGB-Farbwürfel wie er in Abbildung 2.7 dargestellt ist. Schwarz liegt dabei im Koordinatenursprung $C(x, y) = (0,0,0)$, und Weiß an der Hauptdiagonale am Würfelrand $C(x, y) = (255,255,255)$. Die Hauptdiagonale wird auch *Unbuntgerade* genannt, da ihr entlang die oft auch als unbunte Farben bezeichneten Grauwerte liegen [RDG93].

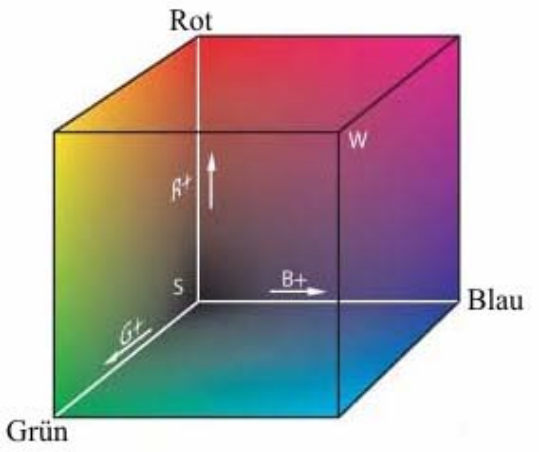

Abbildung 2.7: Der RGB-Farbwürfel
Die Farbdarstellung soll hier nur als Vorstellungshilfe dienen und nicht als reale Darstellung der Farben, da die Farbwiedergabe von Medium zu Medium unterschiedlich sein kann.

Die Farbechtheit ist in dieser digitalen Form des RGB-Farbraums nicht mehr gegeben, da die Primärfarben von jedem Darstellungsgerät etwas anders wiedergegeben werden.

Abbildung 2.8: RGB-Farbbild

Im Folgenden werden andere Farbräume und die Umrechnung aus dem RGB-Raum erläutert. Dabei wird aber nur mehr auf die in der digitalen Bildverarbeitung verwendeten eingegangen, ohne ausführlich auf das theoretische Farbmodell aus Abschnitt 2.3.2 zurückzugreifen.

2.3.4 Der FHS-Farbraum

FHS steht in diesem Fall für die Begriffe Farbton, Helligkeit und Sättigung. Sie dienen als Basis dieses Farbraumes, der näher am menschlichen Farbempfinden angelehnt ist. Im theoretischen Farbmodell werden diese neuen Basisvektoren durch Linearkombinationen aus den verallgemeinerten XYZ Vektoren gebildet (siehe Abschnitt 2.3.1) [RDG93].

Im Folgenden soll nun im Einzelnen erläutert werden, wie man von den im vorigen Abschnitt beschriebenen RGB Werten zu den Werten für Farbton, Helligkeit und Sättigung gelangt.

Farbton:

Der Farbton, englische Bezeichnung *hue*, charakterisiert die in einem Bildpunkt eines Farbbildes dominant enthaltene Farbe. Er wird in Grad angegeben und besitzt somit einen Wertebereich von 0° bis 360°, wobei beide Werte denselben Farbton beschreiben und dieser Punkt als Rot festgelegt wird. Formal ist H gegeben durch $F=\delta$ für alle $B \leq G$ und $F=360°-\delta$ für alle $B>G$, wobei gilt:

$$\delta = arccos\left(\frac{\frac{(R-G)+(R-B)}{2}}{\sqrt{(R-G)^2+(R-B)\cdot(G-B)}} \right). \qquad (2.3)$$

Für Rottöne gekennzeichnet durch $B=G=0$ und $R \neq 0$ gilt beispielsweise $\delta = \arccos(1)$, das heißt, $F=\delta=0°$ [KSC96]. Aus der Gleichung ist ersichtlich, dass es einen undefinierten Wert für $R=G=B$ gibt. Diese RGB-Konstellation definiert einen Grauwert, womit der Farbton obsolet ist. Er wird daher in den meisten Fällen gleich null gesetzt.

Helligkeit:

Die hier beschriebene Helligkeit ist nicht gleich zu setzten mit der im Abschnitt 2.3.1 beschriebenen, da der Begriff im Zusammenhang mit dem FHS-Farbraum keine Aussage über die Leuchtstärke der Quelle erlaubt. Es handelt sich hier eigentlich um den der Farbe entsprechenden Grauton, sprich der Projektion der Farbe auf die Unbuntgerade des RGB-Farbwürfels (siehe Abschnitt 2.3.2). In der Literatur hat sich aber der Begriff Helligkeit durchgesetzt, weshalb er auch in dieser Arbeit für diesen Farbraum verwendet wird. Oft ist auch der Begriff *Intensität* im Gebrauch, der aber ebenfalls mit der physikalischen Intensität der Strahlung eines Senders verwechselt werden kann und deshalb im Sinne einer konsistenten Terminologie hier genauso wenig verwendet werden sollte. Ebenso verhält es sich auch für die englischen Begriffe für Helligkeit *(Brightness)* und Intensität *(Intensity)* [RDG93].

Formal ist der Parameter der Helligkeit gegeben durch

$$H = \frac{R + G + B}{3} \tag{2.4}$$

und entspricht dem arithmetischen Mittel der drei Farbkanäle R, G und B. Der Wertebereich für die Helligkeit ist derselbe wie jener im RGB-Farbraum.

Sättigung:

Die Interpretation des Begriffs Sättigung ist analog zu der im Abschnitt 2.2 beschriebenen. Der Wert der Sättigung ist abhängig von der Anzahl der Wellenlängen die zum Gesamteindruck der Farbe beitragen. Er lässt sich berechnen aus

$$S = 1 - 3 \cdot \frac{min(R,G,B)}{R + G + B}. \tag{2.5}$$

Für $R=G=B\neq 0$, dies entspricht einem Grauwert ist die Sättigung gleich 0. Ist entweder R, G oder B null, die Summe jedoch ungleich null, so ist S stets maximal. Der Wertebereich für die Sättigung entspricht dem des RGB-Farbraums [KSC96].

Der FHS-Farbraum lässt sich in einer Doppelkegelform darstellen. Abbildung 2.9 zeigt eine solche Darstellung.

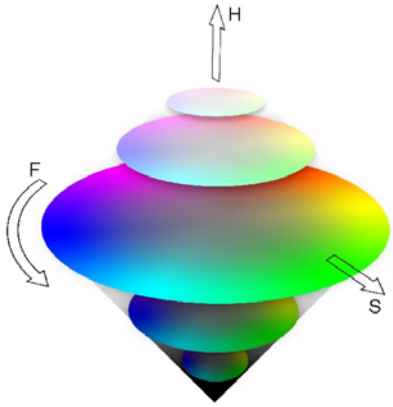

Abbildung 2.9: Darstellung des FHS-Farbraums

Die Farben des RGB-Farbwürfels sind somit durch die drei Parameter Farbton *(F)*, Sättigung *(S)* und Helligkeit *(H)* vollständig beschrieben. Die oben stehenden Gleichungen bieten das Werkzeug um den RGB- vollständig in den FHS-Farbraum überzuführen, mit Ausnahme einiger Singularitäten und etwaiger Rundungsfehler. Die Transformationsgleichungen sind bis auf die erwähnten Singularitäten umkehrbar.

Aufgrund der englischen Begriffe *hue*, *saturation* und *intensity*, wird dieser Farbraum auch als HSI-Farbmodell bezeichnet. Wird anstelle von „intensity“ der englische Begriff *„lightness“* für Helligkeit verwendet, spricht man vom HSL-Modell.

2.3.5 FHS ähnliche Modelle

Es sind in der Literatur und auch in diversen Grafikprogrammen noch einige andere dem FHS sehr ähnliche Farbmodelle in Verwendung. Sie unterscheiden sich meistens nur in der Berechnung der Sättigung und des Helligkeitskanals, nicht aber in der Berechnung des Farbtons. Genannt seien an dieser Stelle das HSB (*„Hue, Saturation, Brightness“*) Modell, oftmals auch HSV (*„Hue, Saturation, Value“*) genannt. Das V bezeichnet die Grauwerte. B steht für die absolute Helligkeit (engl. *brightness*), wobei dieser Begriff hier nicht unbedingt in seiner physikalischen Bedeutung zu verstehen ist. Dieser Farbraum wird im Gegensatz zum FHS in der Form eines einfachen Kegels dargestellt. Die Transformation aus dem RGB in den HSV-Raum erfolgt ähnlich wie beim FHS-Farbmodell. Die Berechnung des S und V-Kanals unterscheiden sich aber.

2.3.6 Das YUV-Farbmodell

Auch das YUV-Modell ist ebenfalls an das menschliche Sehvermögen angelehnt. Der Kanal *Y* beinhaltet die Helligkeitsinformationen im Sinne des FHS-Farbmodells, wobei in diesem Fall, bei dessen Bildung aus dem RGB-Raum, die Empfindlichkeiten des menschlichen

Auges berücksichtigt werden. Der Y-Kanal wird oft auch als Luminanzkanal bezeichnet. *U* und *V* beinhalten die Farbinformationen und werden als Chrominanz- oder Farbdifferenzkanäle bezeichnet, was bereits auf die Art und Weise ihrer Bildung aus dem RGB-Raum hindeutet. Der Y-Kanal errechnet sich Formal aus dem RGB-Raum wie folgt:

$$Y = 0,299 \cdot R + 0,587 \cdot G + 0.114 \cdot B \,. \tag{2.6}$$

Aus der Gleichung ersichtlich, errechnet sich der Gesamthelligkeitseindruck zu ca. 60% aus dem Grün-, zu ca. 30% aus dem Rot- und nur zu ca. 10% aus dem Blauanteil der Farbe. Damit berücksichtigt der Y-Kanal im YUV-Farbmodell die spektrale Empfindlichkeitskurve aus Abschnitt 2.1.1 (vergleiche Abbildung 2.4), die ihre maximale Empfindlichkeit in der Nähe des grünen Bereichs aufweist.

Die Chrominanzkanäle entstehen aus der Differenz zwischen Blauanteil und Lumakanal, bzw. aus dem Rotanteil und dem Lumakanal. Formal sieht dies folgendermaßen aus:

$$U = B - Y \,,\; V = R - Y \,.$$

Wenn man den Wertebereich der RGB-darstellung zugrunde legt liegt der Wertebereich von Y zwischen 0 und 255. U und V können jedoch auch negative werte annehmen.

Auch die Funktionsgleichungen des YUV-Modells sind umkehrbar, womit sich aus der YUV-Darstellung wieder eindeutig das RGB-Modell gewinnen lässt.

Abbildung 2.10 zeigt die drei Kanäle Y, U und V gewonnen aus dem Farbbild aus Abschnitt 2.3.3. Das linke Bild zeigt den Y-Kanal, das mittlere die Farbkomponente U und das rechte den Kanal V.

Abbildung 2.10: Das Farbbild aus Abbildung 2.8 in der YUV-Darstellung

2.3.7 Das YCbCr-Farbmodell

Das YCbCr-Modell entspricht im Wesentlichen dem YUV-Modell. Sie unterscheiden sich nur in den geringfügig unterschiedlichen Transformationsgleichungen die in der Norm CCIR601 [ITU94], bzw. in der Norm IEC601 spezifiziert wurden. Sie lauten wie folgt:

$$Y = round(0,256788 \cdot R + 0,504129 \cdot G + 0,097906 \cdot B) + 16 \,, \tag{2.7}$$

$$U = round(-0,148223 \cdot R - 0,290993 \cdot G + 0,439216 \cdot B) + 128 \,, \tag{2.8}$$

$$V = round(0,439216 \cdot R - 0,367788 \cdot G - 0,07147 \cdot B) + 128 \,. \tag{2.9}$$

Durch die Additionen des Offsets von 128 wird vermieden, dass die Kanäle *U* und *V* negative Werte annehmen, was gewisse Vorteile bei der Datenspeicherung oder –übertragung bietet. Die Addition des Wertes 16 zu Y ist notwendig, da die Werte für Y auf den Bereich von 16 bis 235 begrenzt werden. Schwarz liegt somit bei einem Wert von 16 und Weiß entspricht 235. Die Y-Werte aus dem YUV Modell werden damit durch

$$Y_{CbCr} = \frac{235-16}{255} Y_{YUV} + 16 \tag{2.10}$$

auf die Y-Werte für YCbCr abgebildet. Die Einschränkung erlaubt es in den Datenstrom Steuercodes einzufügen die außerhalb dieses Bereichs liegen und somit eindeutig von den Bilddaten unterschieden werden können. Durch diese Vorkehrungen ist das Modell gut geeignet für die digitale Videoübertragung, wo es auch am häufigsten eingesetzt wird, sowie auch für digitale Videoaufzeichnungen.

2.3.8 Abtastfrequenzen

Das menschliche Auge reagiert auf Veränderungen der Helligkeit wesentlich stärker als auf Veränderungen im Farbverlauf. Dieser Umstand kann ausgenutzt werden um die Datenmenge eines Farbbildes zu reduzieren. Gibt es im verwendeten Farbmodell getrennte Kanäle für die Helligkeit und die Farbinformationen, wie es zum Beispiel im YUV- bzw. YCbCr Farbraum der Fall ist, ist eine Reduzierung der Datenmenge möglich, indem man beispielsweise nur die Farbwerte jedes zweiten Bildpunktes überträgt bzw. speichert, ohne dass die Qualität merklich darunter leidet. In der Signalverarbeitung spricht man hierbei von einer *Unterabtastung* oder im englischen Sprachgebrauch von *subsampling* der Chromakanäle. Es gibt dabei mehrere Möglichkeiten eine Unterabtastung vorzunehmen, die in der Fachliteratur wie folgt bezeichnet werden:

4:4:4

In diesem Fall findet keine Unterabtastung statt. Die Auflösung der Chrominanzkanäle ist identisch mit jener der Luminanz.

4:2:2

Hierbei handelt es sich um eine horizontale Unterabtastung der Farbkanäle, indem nur die Farbinformation für jeden zweiten Bildpunkt in horizontaler Richtung übertragen wird. Somit wird das Datenaufkommen für die Chrominanzkanäle halbiert. Diese Kodierung entspricht der *„Studioqualität“* und wird in professioneller Videoverarbeitungshardware verwendet.

4:2:0

Bei einer 4:2:0-Kodierung wird die Farbauflösung horizontal und vertikal halbiert. Die Datenmenge der Chromakanäle dadurch geviertelt.

4:1:1

Dies bedeutet nur jede vierte Farbinformation wird verwendet. In vertikaler Bildrichtung bleibt die Auflösung somit unverändert.

Es werden in der Fachliteratur noch andere Farbmodelle beschrieben, die meist den speziellen Anforderungen des Bildverarbeitungssystems angepasst wurden. Sie weisen meist nur geringe Abweichungen zu den oben genannten auf und werden somit an dieser Stelle nicht angeführt.

2.4 Auflösung von digitalen Bildern

Die *Auflösung* eines digitalen Bildes beschreibt die Anzahl der Bildpunkte (engl. „*pixel*“) in horizontaler und vertikaler Richtung. Sie kann im Allgemeinen natürlich beliebig groß sein und wird eigentlich nur vom zur Verfügung stehenden Speicherplatz in Verbindung mit dem verwendeten Farbmodell begrenzt. Es haben sich aber im Laufe der Entwicklung, teils aus historischen Gründen, teils durch Normung verschiedene „Standardauflösungen“ durchgesetzt, die eigene Bezeichnungen erhalten haben. In Tabelle 2.2 sind einige aufgelistet.

Tabelle 2.2: Gängige Auflösungen von digitalen Bildern

Name	**Auflösung in Pixel**
QVGA	320 x 240
VGA	640 x 480
XGA	1024 x 768

Meist werden digitale Bilder in Zeilen und Spalten eingeteilt, wobei die Spalten die horizontale Unterteilung oder Unterteilung in x-Richtung, die Spalten die vertikale oder Unterteilung in y-Richtung kennzeichnen. Somit ist auch ein zweidimensionales karthesisches Koordinatensystem gegeben, mit dem Ursprung in der linken oberen Bildecke und positiven x in horizontaler und positiven y in vertikaler Richtung. Diese Definition ist nicht verbindlich, wird aber am häufigsten verwendet und soll auch in dieser Arbeit, wenn nicht anders angegeben, das zugrunde liegende Koordinatensystem sein.

2.5 Videoübertragung

Es wurden im Laufe der Zeit eine ganze Menge an Formaten und Technologien für die Übertragung von Bildern entwickelt. Es soll an dieser Stelle kein Überblick über die

zahllosen Standards in der analogen und digitalen Übertragungstechnik gegeben werden, denn das würde den Rahmen der Arbeit bei Weitem sprengen. Die analoge Übertragungstechnik wird hierbei sogar vollkommen außer Acht gelassen, da sie für die vorliegende Aufgabe keine Relevanz besitzt. Es soll nur auf zwei standardisierte digitale Verfahren zur Übertragung von Farbbildern über eine Kabelverbindung eingegangen werden, da sie für die diese Arbeit und die Bildbearbeitung auf „embedded systems“ im Allgemeinen wichtig sind: ITU656 und ITU601.

2.5.1 Bildübertragung allgemein

Bei der Übertragung einer Bildfolge bedarf es einer Synchronisation zwischen Sender und Empfänger, um letzteren die Möglichkeit zu geben, zu erkennen, wann im Datenstrom ein neues Bild beginnt. Zu diesem Zweck wurden zwei Steuersignale eingeführt: HSYNC und VSYNC.

HSYNC:

Das Wort HSYNC leitet sich ab aus „*horizontal synchronisation*“, der englischen Bezeichnung für horizontale Synchronisation. Dieses Signal kennzeichnet den Beginn und das Ende einer Bildzeile.

VSYNC:

VSYNC steht für vertikale Synchronisation und kennzeichnet den Beginn und das Ende eines Bildes, englisch *frame*.

Progressive und Interleaved:

Es gibt bei manchen Übertragungen auch noch ein drittes Signal, das so genannte „*frame sync*“. Dies wird manchmal verwendet, wenn es, wie es beispielsweise bei der Fernsehübertragung der Fall ist, das Bild nicht linear von der ersten bis zur letzten Bildzeile übertragen wird (engl. „*progressive*“), sondern zuerst alle ungeraden Zeilen und dann die geraden (english „*interleaved*“) gesendet werden. Das Bild wird sozusagen aus zwei Halbbildern aufgebaut. Dies lässt sich aus der historischen Entwicklung der Fernsehübertragung erklären, soll hier aber nicht weiter erläutert werden.

2.5.2 ITU-R BT.601

Dieser 1982 erstmals festgeschriebene Standard, damals noch CCIR601 [ITUCCU04] heute ITU-R BT.601, wurde ursprünglich zur einfachen digitalen Übertragung der beiden Fernsehnormen PAL und NTSC für den europäischen und nordamerikanischen Raum eingeführt. Die Spezifikationen sind auf Grund dessen auf die Parameter dieser beiden Fernsehstandards zugeschnitten. Als Farbmodell für die Übertragung kann YCbCr im 4:4:4

Modus YCbCr 4:2:2 oder RGB verwendet werden (siehe obige Abschnitte dieses Kapitels). In der Praxis wird fast ausschließlich YCbCr im Modus 4:2:2 verwendet [ITU94].

Die Signale für Chrominanz und Luminanz sind mit 8 Bit kodiert, der Wertebereich entspricht den im Abschnitt 2.3.3 beschriebenen.

Im Standard sind zwar weiters die horizontale Auflösung und die Anzahl der Zeilen spezifiziert, genauso wie die Breite der horizontalen Austastlücke und die Anzahl an Zeilen in der vertikalen Austastlücke sowie noch einige andere Parameter, da diese jedoch spezifisch für die beiden Normen PAL und NTSC sind, soll an dieser Stelle nicht weiter darauf eingegangen werden. Wird dies notwendig sein, so werden an entsprechender Stelle im Verlauf der Arbeit nähere Erläuterungen angeführt.

Da ITU601 keine Norm sondern lediglich eine Empfehlung der ITU (International Telecommunication Union) darstellt, wird dieses Übertragungsverfahren heute ohnehin für verschiedenste Auflösungen und Bildwiederholraten verwendet und ist nicht mehr nur an die Übertragung der beiden Fernsehsignale gebunden. Im Einzelnen müssen diese Informationen also den Datenblättern des Bildverarbeitungssystems entnommen werden.

Die Empfehlung der ITU definiert im Standard BT.601 die physischen Eigenschaften der Übertragung nicht. Das heißt, es gibt keine Standardisierung über die Anzahl der Leitungen oder der Synchronisationssignale. Findet man in der Literatur oder in Datenblättern jedoch die Angabe, dass es sich um eine ITU601 kompatible Übertragung handelt, so ist damit meist ein bitparalleles Interface gemeint mit zwei zusätzlichen Signalleitungen für die Synchronisationssignale HSYNC und VSYNC und einem Clocksignal mit dem die Daten übernommen werden.

2.5.3 ITU-R BT.656

Dieser Standard baut auf dem im obigen Abschnitt beschriebenen auf. Somit sind für ITU-R BT.656, meist einfach ITU656 oder CCIR656 genannt, die Spezifikationen von ITU-R BT.601 gültig. Zusätzlich wird aber noch das digitale Interface und die Interfacesignale beschrieben.

Interfacesignale:

Es ist sowohl ein bitparalleles als auch bitserielles Interface spezifiziert. Das bitparallele besteht aus 8 Bit und einer Clockleitung. Die zur Syncronisation erforderlichen Steuersignale sind in den Datenstrom als spezielle Steuercodes eingebettet. Die Werte 0 und 255 sind dafür reserviert, womit sie in den Signalwerten für Helligkeit und Chrominanz nicht vorkommen dürfen. Der Wertebereich für diese Signale ist somit beschränkt auf 254 Werten von 1 bis 254. Für die Werte des Helligkeitskanals gilt ohnehin die Beschränkung aus Abschnitt 2.3.7. Die Signale HSYNC, VSYNC und „frame sync“ sind in den beiden Steuercodesequenzen SAV und EAV codiert [ITU656].

EAV und SAV:

EAV steht für *„End of Active Video“* und zeigt das Ende einer sichtbaren Bildzeile, SAV bedeutet *„Start of active video“* und zeigt den Beginn einer solchen.

Beide Sequenzen bestehen aus 4 Byte: FF 00 00 XY. Die Werte sind in hexadezimaler Schreibweise notiert. Die ersten drei Byte sind eine fixe Preambel, das vierte Byte enthält die Kodierung der Steuersignale wie sie in Tabelle 2.3 dargestellt ist [ITU656].

Tabelle 2.3: Kodierung der EAV und SAV Steuersequenzen [ITU656]

Byte Nr.	**MSB**			**Bit Nr.**				**LSB**
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	1	1	1	1	1	1	1	1
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	1	F	V	H	P3	P2	P1	P0

Wobei gilt:

F=0 für das erste Halbbild, *F=1* für das zweite. Im „progressive“ Modus ist F immer 0 (siehe Abschnitt 2.5.1).

V=0 während der Übertragung sichtbarer Bildzeilen, *V=1* während der vertikalen Austastlücke.

H=0 in SAV, *H=1* in EAV

Die Bits P3 bis P0 dienen zur Überprüfung und Fehlerkorrektur. Ihre korrekte Kodierung ist in Tabelle 2.4 aufgelistet.

Tabelle 2.4: Kodierung der Prüfbits in EAV und SAV [ITU656]

Bit Nr.	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Funktion	**1**	**F**	**V**	**H**	**P3**	**P2**	**P1**	**P0**
	1	0	0	0	0	0	0	0
	1	0	0	1	1	1	0	1
	1	0	1	0	1	0	1	1
	1	0	1	1	0	1	1	0
	1	1	0	0	0	1	1	1

	1	1	0	1	1	0	1	0
	1	1	1	0	1	1	0	0
	1	1	1	1	0	0	0	1

Clocksignal

Da auch der ITU-R BT.656 Standard mittlerweile nicht mehr nur für die Übertragung genormter Fernsehsignale verwendet wird gilt auch hier eine Einschränkung der Auflösung und der Frequenz des Clocksignales nicht mehr. Sie wird an die jeweilige Auflösung und Framerate angepasst. Die Daten werden mit der steigenden Flanke des Clocksignals übernommen [ITU656].

Die Abbildung 2.11 zeigt einen ITU656 kompatiblen Datenstrom für eine Bildzeile. Die Angaben über die Anzahl der Byte beziehen sich auf ein PAL-Fernsehsignal mit einer Auflösung von 720 Pixel in horizontaler Richtung und einer Austastlücke (engl. *blanking*) von 268 Byte. Deutlich erkennbar ist die Datensequenz für zwei Pixel: Cb, Y, Cr, Y. Manche Videoverarbeitungs-ICs erlauben die Umstellung auf andere Sequenzen.

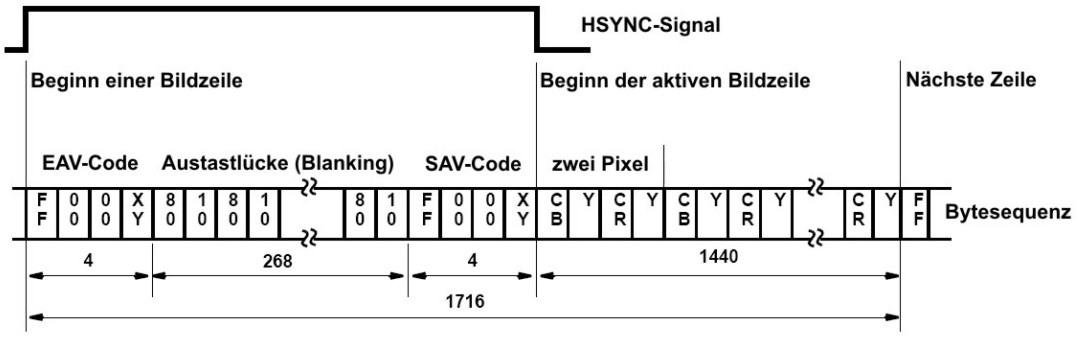

Abbildung 2.11: Datenstrom einer PAL-Bildzeile im ITU656 Format

Der Standard beschreibt ebenfalls die Anzahl der Zeilen, Pixel pro Zeile und horizontale und vertikale Austastlücken. Aber wie bereits erwähnt, gelten diese Spezifikationen nur im Zusammenhang mit der Übertragung von einem PAL- oder NTSC-Signal, können also bei der Übertragung eines digitalen Videosignals anderer Parametrisierung, unbeachtet bleiben, bzw. den Anforderungen angepasst werden.

Kapitel 3 Methoden der Bilderkennung

Aufgrund steigender Bedeutung der Bildverarbeitung in Industrie und für Überwachungsaufgaben, wurden eine Reihe von Methoden für die Objekterkennung und –Klassifizierung entwickelt. Die Vorverarbeitungsschritte unterscheiden sich dabei nur wenig und es hat sich im Laufe der Zeit eine Vorgehensweise durchgesetzt, die heute in fast allen Anwendungen der Bilderkennung zu finden sind. Es sollen nun ausgehend von einer Definition der Begriffe Objekt und Bildstruktur der grundsätzliche Ablauf und die gängigsten Möglichkeiten für die einzelnen Verarbeitungsschritte dargestellt werden. Dabei wird natürlich kein Anspruch auf Vollständigkeit gelegt. Es wird weiters versucht, für die dargestellten Bildbearbeitungsoperationen Aufwandsabschätzungen zu machen, um im Zuge einer späteren Betrachtung, einschätzen zu können, ob sich mit diesen Methoden im Rahmen der zur Verfügung stehenden Rechenleistung, die Anforderungen an das Objekterkennungssystems erfüllen lassen oder nicht.

3.1 Grundlegende Begriffe

Wie erwähnt, werden Begriffe von unterschiedlichen Autoren oft unterschiedlich verwendet. Im Sinne einer konsistenten Terminologie sollen an dieser Stelle, für einige Begriffe die die Methodik der Bildverarbeitung betreffen, definiert werden, wie sie in dieser Arbeit verstanden und verwendet werden. Die Definitionen wurden dabei aus [RDG93] entnommen oder an die dort verwendete Definition angelehnt.

Objekt:

Als ein Objekt wird ein realer, beschreibbarer Gegenstand beliebiger Komplexität verstanden. Im Gegensatz zur objektorientierten Programmierung, wo damit eine beliebige Struktur gemeint ist.

Bildelement:

Der Begriff Bildelement fungiert als Oberbegriff für alle Bildkomponenten und findet sich in allen Schichten des in Abbildung 3.1 gezeigten Modells wieder. Ein Bildelement kann dabei

sowohl ein einzelnes Pixel im Bild, als auch ein Kreissegment oder Farbverlauf nach der Segmentierung sein.

Bildprimitiven:

Unter dem Begriff Bildprimitiven verstehen wir die nach einer Segmentierung erhaltenen Bildelemente.

Bildstruktur:

Sie beschreibt eine komplexere Struktur, die aus Bildelementen oder Bildprimitiven zusammengesetzt wird und selbst wieder Teil komplexerer Bildelemente sein kann.

Merkmal:

Merkmale sind charakteristische Eigenschaften einer Menge von Bildpunkten nach der Vorverarbeitung, von Bildprimitiven nach der Segmentierung, von Komponenten oder von Bildstrukturen. Sie haben den notwendigen Informationsgehalt um mit ihrer Hilfe Bildelemente klassifizieren oder identifizieren zu können.

Modell:

Unter einem Modell verstehen wir prototypische Eigenschaften für ein Objekt, wie sie sich in den einzelnen Schichten des Modells der Bilderkennung wiederspiegeln. Die formale Beschreibung des Modells kann dabei auf unterschiedliche Art und Weise erfolgen und ist abhängig von der Schicht in der wir uns befinden, sowie vom zu modellierenden Objekt selbst. Die Modellierung von Objekten muss somit dem Bilderkennungssystem angepasst werden.

Bilderkennung und Bildverstehen:

Es kommt vor, dass in der Fachliteratur in Zusammenhang mit Objekterkennung von Bildverstehen die Rede ist. In dieser Arbeit soll jedoch bewusst nur der Begriff Bilderkennung in diesem Zusammenhang verwendet werden. Es geht um die automatisierte Erkennung und Vermessung von Objekten. Die Bildauswertung liefert damit als Ergebnis Messwerte. Die Summe der Verarbeitungsschritte um ausgehend von der Aufnahme einer realen Szene zu, für die Reaktion auf die Szene, relevanten Messwerten zu kommen, wird als Bilderkennung bezeichnet. Wie diese Messdaten weiterverarbeitet werden können, wird nicht weiter untersucht. Der Begriff Verstehen geht aber weit darüber hinaus und bedarf bereits einer Modellierung kognitiver Fähigkeiten und selbst dann muss ausdrücklich darauf hingewiesen werden, dass es nicht oder nur sehr eingeschränkt mit dem menschlichen Verständnis für Bilder und Szenarien in Verbindung gebracht werden kann. Sogar beim Menschen selbst ist dieser Begriff nicht eindeutig anwendbar.

3.2 Das Schichtenmodell

Wie bereits erwähnt hat sich im Laufe der Zeit eine Vorgehensweise bei der Bilderkennung etabliert wie sie in Abbildung 3.1 dargestellt ist. Der Prozess der Bilderkennung lässt sich demnach in einem Schichtenmodell repräsentieren, das von der Aufnahme und damit verbundenen Digitalisierung einer Szene bis hin zur Ausgabe eines Ergebnisses reicht. Wobei das Ergebnis natürlich abhängig ist von der vorher definierten Aufgabe des Bilderkennungssystems.

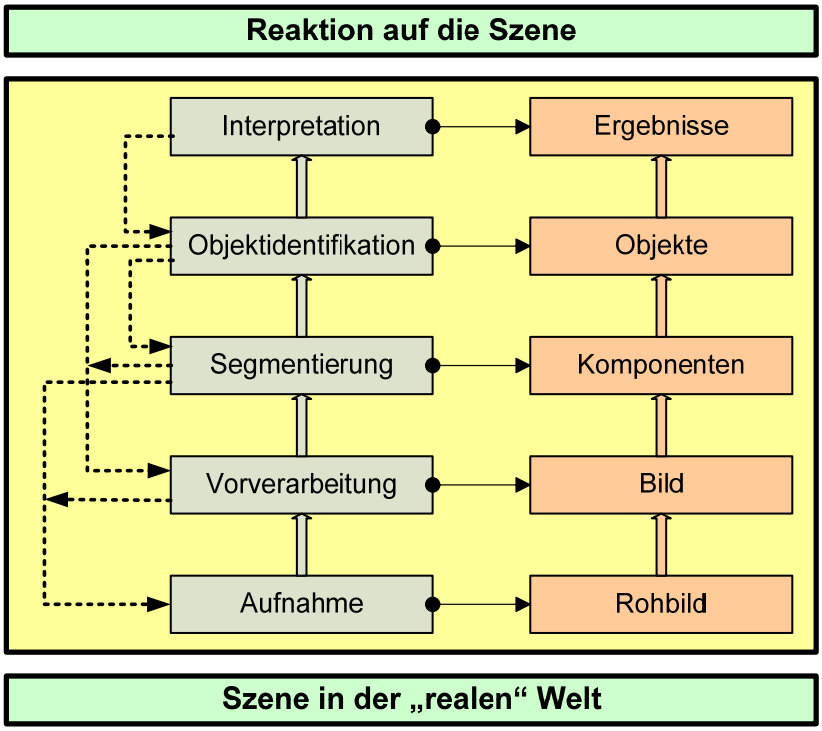

Abbildung 3.1: Schichtenmodell der Bilderkennung

Das Schichtenmodell beschreibt die einzelnen Arbeitsschritte ausgehend von einem „realen“ beobachtbaren Szenario bis hin zu einer Reaktion des Systems auf die Szene, wobei dies nicht mehr Teil der Bilderkennung, so wie sie in dieser Arbeit verstanden wird, darstellt, da zur Entscheidungsfindung über die Art der Reaktion auch andere Sensoren bzw. andere Informationsquellen beitragen können. Das eigentliche Bildverarbeitungssystem wird vom Bereich der vom mittleren Rechteck in der Abbildung 3.1 eingeschlossen wird, gebildet. Die rechts stehenden Begriffe sind die jeweils aus dem links daneben stehenden Arbeitsschritt gewonnenen Komponenten. Sie, und die dazu gehörenden Operationen, werden im Folgenden näher erläutert. Die strichlierten Linien sollen gewisse Rückkopplungen zwischen

den einzelnen Arbeitsschritten darstellen. So können beispielsweise Erkenntnisse aus einem Verarbeitungsschritt beim nächsten Frame die Parameter des vorhergehenden Schritts beeinflussen. Auch dies wird im Laufe dieser Arbeit untersucht werden [RDG93, STN93].

Im Grunde ist die Bilderkennung ein Messvorgang. Nach der Bildvorverarbeitung werden durch entsprechende Operationen Merkmale aus dem Bild extrahiert was im Sinne der Signalverarbeitung einer Messung entspricht. Die anschließende Erkennung von Komponenten auf Basis dieser Messung bzw. Merkmale ist analog zu einer Messdatenauswertung [RDG93].

In dieser Arbeit ist der Ausgangspunkt für die Bildverarbeitung immer ein Farbbild. Die in den nächsten Abschnitten gezeigten Verfahren arbeiten jedoch immer nur auf einem Kanal, sodass für das Farbbild der Operator auf alle Kanäle des Farbbildes angewendet werden muss. Es kann dabei aber vorkommen, dass es für die Weiterverarbeitung ausreicht den Vorgang nur auf einen Kanal anzuwenden, da er aussagekräftig genug für die untersuchten Bildelemente ist. Wenn bei einem bestimmten Verfahren eine besondere Vorgehensweise bei der Behandlung von Farbbildern notwendig ist, so wird dies angeführt. Ansonsten wird die Funktionsweise des Operators anhand eines Kanals gezeigt.

Im Folgenden werden nun gängige Bearbeitungsschritte in den einzelnen Schichten vorgestellt. Dabei finden jedoch die einzelnen gezeigten Methoden nicht zwingend nur in der Schicht in der sie erwähnt werden ihre Anwendung. Es kann durchaus vorkommen, dass ein Verfahren auch noch in einer anderen Schicht für die Lösung bestimmter Aufgaben interessant sein kann.

3.3 Aufnahme

Die Aufnahme liefert als Ausgang ein digitales Rohbild. Da die Digitalisierung auf unterschiedlichem Weg erfolgen kann, schließt dieser Punkt die gesamte Aufnahmeeinheit mit ein, also von der Linse bis zum digitalen Rohbild. Die Aufnahme kann dabei auch mittels einer „Analogkamera“ gemacht werden und anschließend mittels eines Analog-Digital-Wandlers digitalisiert werden. Damit zählt die gesamte Aufnahmeeinheit inklusive des AD-Wandlers zur Aufnahmeschicht des in Abbildung 3.1 gezeigten Schichtenmodells.

Im Abschnitt 4.3 werden einige Aufnahmeeinheiten und –Technologien vorgestellt. Dabei bieten moderne Kameras die Möglichkeit, einige der im nächsten Abschnitt vorgestellten Vorverarbeitungsschritte, bereits direkt in der Kamera auszuführen. Trotzdem wird in diesem Fall die gesamte Kamera der ersten Schicht zugeordnet. Auch aufgrund dessen, da in vielen Fällen trotzdem noch einige den Anforderungen angepasste Vorverarbeitungsschritte durchgeführt werden müssen.

3.4 Vorverarbeitung

Zu den Aufgaben in dieser Schicht, zählen zum Beispiel die Entzerrung, die manchmal notwendig ist um Fehler im Linsensystem oder allgemeiner Aufnahmefehler zu kompensieren. Weiters die Verbesserung des Kontrastes oder die Eliminierung von Rauschen oder anderer dem Bild überlagerter Störungen.

3.4.1 Charakterisierung digitaler Bilder

Für die Weiterverarbeitung von Bildern und die Auswahl von geeigneten Verfahren zur Bildverarbeitung kann es unter Umständen wichtig sein, a priori Informationen über das Bild zu gewinnen. So gibt es einige charakteristische Werte für digitale Bilder, die es erlauben noch vor einer Verarbeitung verschiedene Aussagen über das Bild zu machen. Die im Folgenden gezeigten Eigenschaften beziehen sich immer auf das Gesamtbild. Es kann aber für eine Weiterverarbeitung unter Umständen sinnvoll sein, die Charakterisierung auf Teile des Bildes zu beschränken.

Mittelwert:

Ausgehend von einem einkanaligen Grauwertbild $C=g(x,y)$ mit P Zeilen und Q Spalten, berechnet sich der mittlere Grauwert auch Mittelwert m genannt wie folgt:

$$m = \frac{1}{M} \cdot \sum_{x=0}^{P-1} \sum_{y=0}^{Q-1} g(x,y), \tag{3.1}$$

wobei $M = L \cdot R$, die Anzahl der Bildpunkte von C, ist. Aus dem Mittelwert lässt sich in erster Linie ablesen, ob ein Bild eher dunkler oder heller ist [HBR91].

mittlere quadratische Abweichung:

Während der Mittelwert nur eine Aussage über die durchschnittliche Helligkeit eines Bildes erlaubt, kann mit Hilfe der mittleren quadratischen Abweichung q eine Aussage über den Kontrast des Bildes getroffen werden [HBR91]. Sie errechnet sich aus:

$$q = \frac{1}{M} \cdot \sum_{x=0}^{P-1} \sum_{y=0}^{Q-1} [g(x,y) - m]^2 . \tag{3.2}$$

Mit Hilfe der binomischen Formel lässt sich obige Gleichung so umformen, dass der Mittelwert und die mittlere quadratische Abweichung in einem Durchgang berechnet werden können. Darauf soll an dieser Stelle jedoch nicht näher eingegangen werden. Für weter führende Informationen sei an dieser Stelle auf [HBR91] hingewiesen.

Histogramm:

Das Histogramm zeigt die Verteilung der relativen Häufigkeiten der Grauwerte. Aus einem Bild $C=g(x,y)$ mit dem Wertebereich [0, 255] ergibt sich das Histogramm aus:

$$p(s)=\frac{a_s}{M},\ s=0,1...,255.$$

Dabei beschreibt a_s die Anzahl der Bildpunkte mit dem Wert s. Dieser wird mit der Gesamtanzahl der Bildpunkte in C normiert. Dadurch gilt:

$$\sum_{s=0}^{255} p(s)=1,\ \text{bzw.}\ 0\le p(s)\le 1.$$

Die nach obiger Gleichung berechneten Werte werden in einem zweidimensionalen karthesischen Koordinatensystem eingetragen. Auf der Abszisse stehen die Grauwerte s von 0 bis 255. Auf der Ordinate wird der zu dem Grauwert gehörende Wert $p(s)$ als Balken eingezeichnet. Die Abbildung 3.2 zeigt diese Darstellung des Histogramms, gebildet aus dem links daneben stehenden Graustufenbild.

Abbildung 3.2: Graustufenbild und zugehöriges Histogramm

Der oben erwähnte Mittelwert und die mittlere quadratische Abweichung können auch aus dem Histogramm ermittelt werden, mittels:

$$m=\frac{1}{M}\sum_{s=0}^{255} s\cdot M\cdot p(s)=\sum_{s=0}^{255} s\cdot p(s) \tag{3.3}$$

und

$$q=\frac{1}{M}\sum_{s=0}^{255}(g-m)^2\cdot M\cdot p(s)=\sum_{s=0}^{255}(g-m)^2\cdot p(s). \tag{3.4}$$

.

Bei einem Bild mit viel Kontrast sind die relativen Häufigkeiten mehr oder weniger gleich verteilt. Ist das Bild kontrastarm gibt es ein lokales Maximum bei den höheren oder niedrigeren Grauwerten, je nachdem ob das Bild heller oder dunkler ist. Gibt es einen hellen

und einen dunklen Bereich so zeigt das Histogramm zwei lokale Maxima. Histogramme dieser Art werden *bimodal* genannt [HBR91].

Farbbilder bestehen aus drei Kanälen. Für jeden Kanal kann man dabei ein Histogramm erstellen. Es ist aber auch möglich das einkanalige Histogramm auf N Kanäle zu verallgemeinern. Das N-dimensionale Histogramm eines *N*-kanaligen Bildes $C=g(x,y,n)$, mit $n=0,1,\ldots N\text{-}1$ ist definiert als:

$$p(s_0,s_1,\ldots,s_{N-1}) = \frac{a_{s_0 s_1 \ldots s_{N-1}}}{M}, \tag{3.5}$$

wobei $a_{s_0 s_1 \ldots s_{N-1}}$ die Häufigkeit der Grauwertkombination mit dem Grauwert s_n im Kanal n ist und M die Anzahl der Pixel [HBR91].

3.4.2 Punktoperatoren

Ein einkanaliges Bild, das kann ein Kanal eines Farbraumes für ein Farbbild oder ein Graustufenbild sein, kann als eine Funktion zweier Variablen $f(x,y)$ gesehen werden (vergleiche dazu 2.3.3). Durch die Anwendung eines Punktoperators wird diese Funktion auf eine neue Funktion $f'(x,y)$ abgebildet. Er stellt somit eine Berechnungsvorschrift dar, die den Wert eines jeden Bildpunkts ändert, ohne den Wert der benachbarten Bildpunkte zu berücksichtigen. Daher der Name „Punktoperator“. Im Folgenden werden die Bildpunkte der Einfachheit halber des Öfteren als Grauwerte bezeichnet. Es soll daran erinnert werden, dass es sich auch um die Werte eines Farbkanals handeln könnte.

Eine Möglichkeit einer solchen Grauwertmanipulation ist die Skalierung. Man unterscheidet dabei lineare und nichtlineare Skalierungen.

lineare Skalierung:

Die lineare Skalierung beschreibt eine Abbildungsvorschrift mittels einer linearen Funktion. Das Ergebnisbild $g'(x,y)$ wird gebildet aus:

$$g'(x,y) = a \cdot g(x,y) + b \tag{3.6}$$

Durch die Addition des Faktors b wird dabei eine konstante Verschiebung der Grauwerte erreicht. Je nach dem ob c positiv oder negativ ist erreicht man eine Verschiebung zu helleren oder dunkleren Werten. Der Mittelwert des Ergebnisbildes wird somit zu helleren oder dunkleren Werten verschoben, womit das Bild optisch heller oder dunkler erscheint. Damit kann man also ein dunkles Bild aufhellen. Im Histogramm (siehe dazu Abschnitt 3.4.1) äußert sch diese Operation durch eine Verschiebung nach links oder nach rechts. Es entsteht dadurch aber keine Kontrastverbesserung. Subjektiv kann dies zwar der Fall sein. Das liegt aber nur daran, dass das menschliche Sehsystem eine logarithmische Kennlinie bezüglich der Helligkeitsempfindung aufweist und daher die Grauwertunterscheidung in dunkleren Bereichen geringer ist (vergleiche dazu Abschnitt 2.1).

Eine echte Veränderung des Kontrastes erreicht man durch eine Multiplikation mit der Konstanten *a*. ist *a<1* wirken die Bilder kontrastärmer. Ist *a>1*, kontrastreicher. Dabei kann man den Multiplikationsfaktor *a* so wählen, dass der gesamte Wertebereich ausgenutzt wird. Dies erreicht man formal durch

$$g'(x,y) = (g(x,y) - g_{min}) \frac{G_{max} - G_{min}}{g_{max} - g_{min}}, \tag{3.7}$$

wobei g_{min} und g_{max} den maximalen und minimalen im Bild vorkommenden Grauwert bezeichnen und G_{min} und G_{max} die Ober- und Untergrenze des Wertebereichs darstellen [STN93]. Die Abbildung 3.3 zeigt den Effekt obiger Skalierung. Das linke Bild wurde dabei mit dem Faktor *a=3,5* multipliziert und um *b*=60 verschoben. Das rechte Bild zeigt das Resultat.

Abbildung 3.3: Kontrastverbesserung und Aufhellung durch lineare Skalierung

Bisher wurde die Abbildungsvorschrift immer auf das gesamte Bild angewendet. Es kann in manchen Fällen sinnvoll sein, nur Teile des Bildes zu verändern. Beispielsweise wenn Teile im Schatten liegen andere aber in der Sonne, kann es notwendig sein nur den im Schatten liegenden Teil aufzuhellen. Das heißt, die Faktoren *a* und *b* sind nicht über das ganze Bild konstant, sondern beispielsweise abhängig von der Helligkeit. Diese könnte stückweise über den Mittelwert (siehe Abschnitt 3.4.1) der Grauwerte bestimmt werden.

Eine Skalierungsfunktion muss nicht zwingend linear sein. Wenn wir obige Gleichung betrachten, so lässt sie sich leicht verallgemeinern auf:

$$g'(x,y) = (f(g(x,y)) - g_{min}) \frac{G_{max} - G_{min}}{g_{max} - g_{min}}, \tag{3.8}$$

wobei $f(g(x,y))$ eine beliebige Funktion ist. Denkbar wäre beispielsweise eine logarithmische Abbildungsvorschrift der Art:

$$g'(x,y) = log(g(x,y) + 1) \frac{G_{max} - G_{min}}{g_{max} - g_{min}}. \tag{3.9}$$

Die Logarithmierung der Grauwerte des Ausgangsbilds führt dazu, dass die dunklen Bereiche des Bildes stärker im Kontrast angereichert werden, als die hellen, für die das menschliche

Auge ohnehin eine größere Empfindlichkeit besitzt [HBR89]. Die Addition von 1 zu den Grauwerten des Ausgangsbilds dient lediglich zur Verhinderung der Singularität bei log(*0*).

Es sei darauf hingewiesen, dass darauf geachtet werden muss, dass durch die Anwendung der Skalierungsfunktion der Wertebereich nicht überschritten wird. Anderenfalls kommt es zu Verfälschungen. Dies kann durch das so genannte *„clipping"* verhindert werden. Dabei wird überprüft, ob das Ergebnis den Wertebereich über- oder unterschreitet. Ist dies der Fall, so wird der Wert bei Überschreitung des Maximums auf das Maximum bzw. bei unterschreiten des Minimums auf das Minimum des Wertebereichs gesetzt. Dies kann unter Umständen von Nutzen sein, sofern „uninteressante" Teile des Bildes dadurch ausgeblendet werden.

In der Praxis werden die Skalierungsfunktionen meist mittels einer so genannten *„lookup table"* realisiert. Nachdem sich der Wertebereich der Ausgangsbilder für gängige digitale Farbräume meist auf 256 Werte von 0 bis 255 beschränkt, genügt eine Tabelle mit 256 Einträgen die jedem der 256 Ausgangswerte einen Ergebniswert entsprechend der Skalierungsfunktion zuordnet, um die Berechnung durch zu führen. Es muss zur Bildung des Ergebnisbildes nur mehr für jeden Bildpunkt das Ergebnis aus der Tabelle gelesen werden. Damit wird die Berechnungszeit erheblich gesenkt und auch eine getrennte Überprüfung des Ergebnisses auf Gültigkeit im Sinne des Wertebereichs entfällt. Damit können auch stückweise stetige Funktionen realisiert werden [STN93].

Histogrammeinebnung:

Ein kontrastreiches Bild zeigt sich im Histogramm durch eine möglichst gute Verteilung der Grauwerte über den zur Verfügung stehenden Wertebereich. Ist dies nicht von vorneherein der Fall, so kann man versuchen dies durch eine Transformation der Form:

$$G'(i) = \frac{G_{max}}{T} H_x(x_i) \tag{3.10}$$

zu erreichen. Wobei

$$H_x(x_i) = \left(\sum_{x=x_L}^{x_i} h(x) \right) - 0{,}5 \cdot \left[h(x_i) + h(x_L) \right] \tag{3.11}$$

die kumulative Verteilungsfunktion, T die Anzahl der Pixel und $h(x_L)$ bzw. $h(x_U)$ die Häufigkeit des kleinsten bzw. größten Grauwertes ist. $G_{\max}$ bezeichnet den größten möglichen Grauwert [STN93].

Damit ist es also möglich, die Grauwertverteilung einem vorgegebenen Histogramm anzupassen. Ein Sonderfall davon, wäre der Versuch eine Gleichverteilung zu erreichen. Dann würde jeder Gauwert mit der gleichen Häufigkeit h_{const}=1/256 auftreten [HBR91].

Die Bilder werden durch die Histogrammeinebnung im Allgemeinen kontrastreicher. Kommt allerdings ein Grauwert oder ein begrenzter Grauwertbereich sehr oft vor so führt die

Histogrammeinebnung zu unbefriedigenden Ergebnissen, da der Kontrast in Bereichen der weniger häufig vorkommenden Grauwerte schlechter wird.

Auch im Falle der Histogrammeinebnung gilt, dass sie nicht zwingend auf das Gesamtbild angewendet werden muss. Hierbei erzielt man besonders gute Ergebnisse wenn man das gesamte Bild in rechteckige Teilbilder unterteilt, die in etwa der Größe der interessanten Objekte entsprechen. Man spricht in diesem Fall von *adaptiver Histogrammeinebnung* Auf jedes dieser Teilbilder wendet man nun die Histogrammeinebnung an. Dies führt an den Rändern der Teilbilder natürlich zu sichtbaren Übergängen. Diese kann man eliminieren, indem man die Grauwerte jedes Feldes umgekehrt proportional zu ihren euklidischen Abständen zu den Mittelpunkten aller Nachbarfelder gewichtet [STN93].

In der Fachliteratur findet man auch oft den Begriff *Histogrammausgleich* für die Histogrammeinebnung, bzw. Den Begriff *lokaler Histogrammausgleich* für die adaptive Histogrammeinebnung.

Histogrammhyperbolisierung:

In manchen Fällen werden auch gute Ergebnisse erzielt, wenn man keine Gleichverteilung der Grauwerte anstrebt, sondern eine hyperbolische. Angelehnt an den Verlauf des Helligkeitsempfindens des menschlichen Auges aus Abbildung 2.4 kann dann entweder ein logarithmischer oder ein Verlauf mit der dritten Wurzel für die Grauwertverteilung angesetzt werden [STB88].

Abbildung 3.4: Ausgangsbild mit 400 x 250 Pixel und dazugehörendes Histogramm

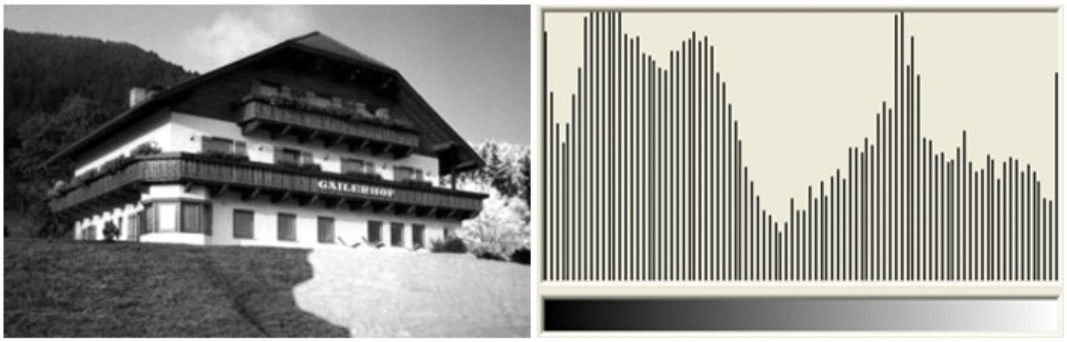

Abbildung 3.5: Ergebnisbild einer Histogrammeinebnung

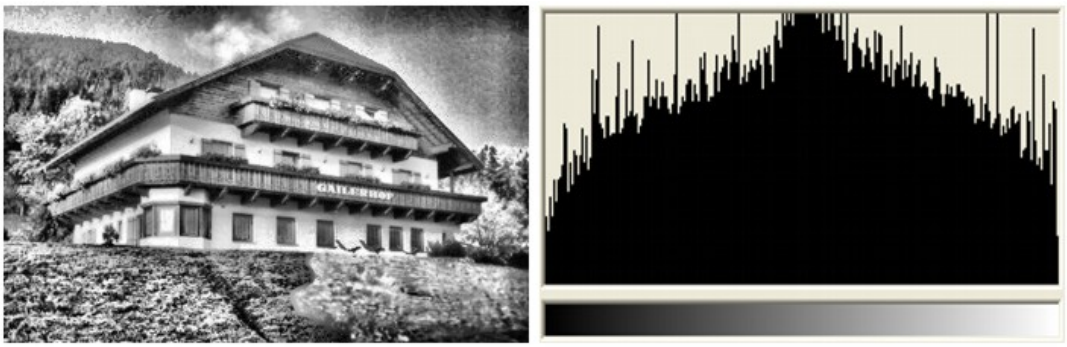

Abbildung 3.6: Ergebnisbild einer adaptiven Histogrammeinebnung

Abbildung 3.5 zeigt das Ergebnis einer Histogrammeinebnung angewendet auf das Grauwertbild aus Abbildung 3.4. Wendet man auf dasselbe Bild eine adaptive Histogrammeinebnung mit einer Teilbildgröße von 50x50 Pixel an, so erhält man als Resultat die in der Grafik Abbildung 3.6 gezeigte Kontrastverbesserung. Das Ausgangsbild hat eine Auflösung von 400 x 250 Pixel. Das Diagramm neben den Bildern zeigt das zum jeweiligen Bild gehörende Histogramm. Im Ausgangsbild konzentrieren sich die Grauwerte auf den Bereich von 64 bis 192, das Bild hat sehr wenig Kontrast. Man sieht in den Histogrammen der Ergebnisbilder nach der Histogrammeinebnung sehr deutlich die bessere Verteilung der Grauwerte auf den gesamten Wertebereich, eine Tatsache die sich in einer deutlichen Kontrastverbesserung zeigt.

3.4.3 Globale Operatoren

Der Name *„globaler Operator“* leitet sich aus der Arbeitsweise der im Folgenden dargestellten Operationen ab. Es werden im Gegensatz zu die unter Punkt 3.4.2 vorgestellten Punktoperatoren, wo jeder Pixel für sich isoliert behandelt wird, auch die den Punkt unmittelbar umgebenden Nachbarpixel miteinander verknüpft um zu einem Ergebnispunkt zu kommen. Mehrere kann dabei im Extremfall auch das ganze Bild bedeuten. Man nennt die globalen Operatoren auch *„Operatoren im Ortsbereich“*, da sie nicht im Frequenzbereich des Bildes arbeiten sondern im Ortsbereich. Das Bild wird dabei wiederum als Funktion zweier reeller Variablen betrachtet $C=f(x,y)$.

Lineare Filterung im Ortsbereich, die Faltung:

Die wichtigste Operation im Ortsbereich ist die *Faltung*. Sie ist definiert sowohl in einer Dimension, wo sie der Faltung im Zeitbereich von reellwertigen Funktionen entspricht, als auch in zwei Dimensionen, wobei letztere wichtiger ist für die Bildverarbeitung. Zur Veranschaulichung sei zuerst die Faltung im eindimensionalen Fall erklärt. Sie ist für diskrete Signale definiert als:

$$g_m = f_m * h_m = \sum_j^{M-1} f_j h_{m-j} = \sum_j^{M-1} f_{m-j} h_j \tag{3.12}$$

M ist dabei die Länge des Ortsvektors, f_j ein Element der eindimensionalen Bildmatrix und h_j ein Element der ebenfalls eindimensionalen Faltungsmatrix der Länge M. Abbildung 3.7 zeigt beides.

f_0	f_{M-1}	f_{M-2}	...	f_2	f_1
h_0	h_1	h_2	...	h_{M-2}	h_{M-2}

Abbildung 3.7: Eindimensionale Bildmatrix und Faltungsmatrix

Zur Berechnung der Faltung für einen Index m muss das Produkt $f_{m-j}h_j$ über alle Indizes j von 0 bis M-1 aufsummiert werden.

Die Wirkungsweise der Faltungsoperation soll am folgenden Beispiel Schritt für Schritt veranschaulicht werden.

Schritt 1: Die Funktion aus Abbildung 3.7 wird am Ursprung gespiegelt, das heißt, am Punkt m=0:

f_0	f_{M-1}	f_{M-2}	...	f_2	f_1

Abbildung 3.8: Spiegelung der Funktion aus Abbildung 3.7 am Ursprung

Schritt 2: Addition von M zu den negativen Indizes der Funktion:

f_0	f_1	f_2	...	f_{M-2}	f_{M-1}
h_0	h_1	h_2	...	h_{M-1}	h_{M-2}

Abbildung 3.9: Addition von M zu den negativen Indizes

Schritt 3: Multiplikation der übereinander stehenden Faktoren und Summation der Ergebniszeile.

Mit der auf eins normierten Faltungsmaske aus Abbildung 3.10, bei der nur die Elemente h0, h1 und hM-1 von null verschieden sind,

$\frac{1}{3}$	1	1	0	...	1	0

Abbildung 3.10: Auf eins normierte Faltungsmaske

ist somit die in Abbildung 3.11 gezeigte Berechnung durchzuführen.

f_0	f_{M-1}	f_{M-2}	...	f_2	f_1
1	1	0	...	1	0

Abbildung 3.11: Beispiel für die Berechnung einer Faltung

Als Ergebnis erhält man:

$$g_0 = \frac{1}{3}(f_0 + f_M + f_1)\,. \tag{3.13}$$

Dies ist der Mittelwert aus dem Punkt f_0 und den Nachbarpunkten f_{-1} und f_1. Die Faltung ist damit jedoch nur am Punkt m=0 berechnet. Für den nächsten Punkt m=1 muss entweder die Funktion oder die Faltungsmaske um eins nach rechts verschoben und die Berechnung erneut durchgeführt werden [JHN91].

Bisher wurde die Faltung zur Veranschaulichung anhand eindimensionaler Signale gezeigt. Sie ist jedoch auch für zweidimensionale Signale definiert, wobei die Faltung in zwei Dimensionen eine wesentlich größere Bedeutung in der Bildverarbeitung hat. Analog zur eindimensionalen ist sie definiert als:

$$G_{m,n} = F_{m,n} * H_{m,n} = \sum_{k}^{M-1}\sum_{l}^{N-1} F_{k,l} H_{m-k,n-l} = \sum_{k}^{M-1}\sum_{l}^{N-1} F_{m-k,n-l} H_{k,l} \tag{3.14}$$

Der Bildvektor sowie die Faltungsmatrix hat hier die Dimension M x N, wobei in der Praxis fast ausschließlich Masken der Größe M x M verwendet werden. Gebräuchliche Größen sind hierbei 3 x 3, 5 x 5 oder 7 x 7.

Zur Veranschaulichung sei die Faltung einer 3 x 3 Bildmatrix mit einer 3 x 3 Rechteckmaske $H_{u,v}$ gezeigt. Die Indizes u und v bewegen sich von -1 bis 1. Abbildung 3.12 zeigt die zugehörige Matrix.

$$\frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Abbildung 3.12: Zweidimensionale Rechteckmaske

Die Bildung der zweidimensionalen Faltung verläuft nun analog zur eindimensionalen. Die Maske wird am Punkt $H_{0,0}$ gespiegelt. Für die Bestimmung des Wertes Gm,n der Faltung legen wird die Rechteckmaske über die Bildmatrix, multiplizieren die übereinander liegenden

Werte und summieren sie auf. Analog zur eindimensionalen Faltung werden für die Berechnung des nächsten Punktes der gleichen Zeile die Rechteckmatrix um eins nach links geschoben und wieder alle Werte multipliziert und aufsummiert [JHN91].

Um Übersteuerungen zu vermeiden, das heißt, keine Grauwerte außerhalb des Wertebereichs zu erhalten, wird die Maske immer auf eins normiert. Im Falle der Maske aus Abbildung 3.12 würde das Ergebnis für jeden Punkt der Faltung mit $1/M^2$ multipliziert.

Besondere Beachtung muss bei der zweidimensionalen Faltung auf die Randbereiche gelegt werden. Prinzipiell gibt es zwei Möglichkeiten. Entweder es werden Zeilen und Spalten hinzugefügt deren Werte gleich den Randpunkten gesetzt oder linear extrapoliert werden, oder es werden die Randbereiche in der Größe von einer halben Maskenbreite ausgelassen, was zu einer Verkleinerung des Bildes führt und nur bei kleinen Masken sinnvoll ist [JHN91].

Eine überschlagsmäßige Rechnung zeigt, dass für die Bildung der Faltung neun Multiplikationen und neun Additionen pro Bildpunkt erforderlich sind, was einen erheblichen Rechenaufwand bedeutet. Bei einer Auflösung von 320 x 240 Pixel sind dies 691 200 Multiplikationen und Additionen pro Bild.

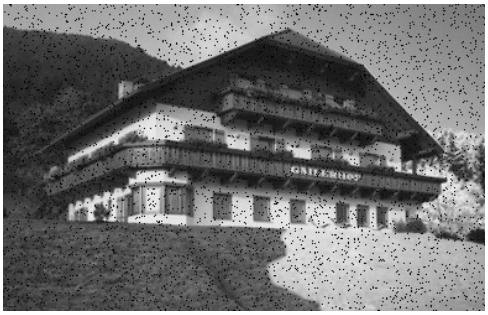

Abbildung 3.13: Testbild mit zufällig verteilten Bildpunktstörungen

Mit der Hilfe der Faltung lassen sich sehr einfach Signalfilter im Ortsbereich realisieren. Es seien an dieser Stelle einige Tiefpassfilter erwähnt, da eine Anwendung im Zusammenhang mit der Bildvorverarbeitung sinnvoll sein kann, um ein dem Bild überlagertes Rauschen zu eliminieren. Rauschen ist eine hochfrequente Störung, weshalb sie mit einem Tiefpassfilter gut eliminiert werden kann. Abbildung 3.13 zeigt ein solch verrauschtes Bild.

Mittelwertfilter:

Die Maske für das Mittelwertfilter ist jene aus Abbildung 3.12. Damit wird für jeden Bildpunkt der Mittelwert aus ihm und seinen acht Nachbarn gebildet, daher der Name. Durch die Anwendung des Mittelwertfilters wirkt das Bild etwas unschärfer, da wie erwähnt hochfrequente Anteile im Bild unterdrückt werden und damit scharfe Kanten weicher werden, feine Bildstrukturen verschwinden dadurch unter Umständen.

Abbildung 3.14: Anwendung des Mittelwertfilters auf Abbildung 3.13

Abbildung 3.13 zeigt ein Bild, in das zufällige Bildpunktstörungen integriert wurden. Diese Störungen simulieren ein zufälliges Bildrauschen. Wird auf dieses verrauschte Bild nun das Mittelwertfilter angewendet, so ergibt sich das in Abbildung 3.14 gezeigt Ergebnis. Die Bildstörungen sind zwar weniger ausgeprägt, jedoch immer noch sichtbar. Insgesamt wurde das Bild aber deutlich unschärfer.

Gauß- und Binomialfilter:

Das Gaußfilter liefert besonders gute Ergebnisse beim Glätten von Bildern und der Rauschunterdrückung. Die Filterfunktion entspricht der *„gaußschen Glockenkurve“*, die in der Praxis durch die *„Binomialverteilung“* angenähert wird [JHN91]. Daraus ergibt sich die in Abbildung 3.15 gezeigte Filtermaske.

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Abbildung 3.15: Filtermaske eines Binomialfilters

Nicht lineare Filterung im Ortsbereich:

Die Faltung ist eine lineare ortsinvariante Operation. Das heißt, ihre Wirkung ist ortsunabhängig und sie wirkt auf ein überlagertes Rauschen genauso wie auf den eigentlichen Bildinhalt wie man in Abbildung 3.14 sehr schön sieht. Deshalb bietet das oben gezeigte Mittelwertfilter nur einen Kompromiss zwischen Rauschunterdrückung und der Eliminierung feiner Bildstrukturen. Diesen Nachteil der Faltung umgeht man bei Anwendung von nichtlinearen Operatoren, wenn man die zugrunde gelegte Funktion abhängig macht vom Wert der Bildpunkte (Ortsabhängigkeit). Ein Beispiel für ein solches nichtlineares Filter ist das Medianfilter. Ihm liegt somit keine Faltung mehr zugrunde, sondern eine nichtlineare Transformation, bei der aber die Nachbarpunkte zur Bestimmung des Ergebnisses herangezogen werden, weshalb es sich immer noch um einen globalen Operator handelt.

Medianfilter:

Es wurde erstmals Anfang der 70er Jahre vorgeschlagen [GLW81]. Es wird wieder eine symmetrische Maske angewendet. Diesmal ist es aber, wie bereits erwähnt, keine Faltungsoperation. Ein Bildpunkt im Ergebnisbild ergibt sich in dem man den mittleren Grauwert (*Median*) des von der Maske abgedeckten Bildbereichs als Ergebnis verwendet. Da dies eine nichtlineare Operation ist, zählt das Medianfilter zu den nichtlinearen Filtern. Für den nächsten Bildpunkt wird die Maske analog zur Faltung um eins nach rechts verschoben [STN93]. Die Vorteile des Medianfilters vor allem gegenüber dem Mittelwertfilter sind:

- der Erhalt scharfer Kanten
- die gute Unterdrückung von Störimpulsen.

Ein großer Nachteil ist allerdings die zeitaufwendige Berechnung, da die Folge der von der Maske überdeckten Grauwerte sortiert werden muss, um den Median zu finden. Dazu kann ein Standardsortieralgorithmus wie *„Bubblesort“* oder *„Quicksort“* verwendet werden.

Abbildung 3.16: Anwendung des Medianfilters auf Abbildung 3.13

Wendet man auf das Bild aus Abbildung 3.13 ein Medianfilter mit einer Maskengröße von 3 x 3 Pixel an so ergibt sich das in Abbildung 3.16 gezeigte Ergebnis. Die Störungen wurden großteils eliminiert und das Bild ist schärfer als das Mittelwert gefilterte.

3.4.4 Operationen im Frequenzbereich

Bisher wurden alle Operationen im Ortsraum durchgeführt. Dazu wurden die Bilder als Funktion zweier Variablen, bzw. als eine M x N Matrix der Bildpunkte betrachtet. In der Mathematik sind auch andere Darstellungsformen von Funktionen bekannt. Eine Möglichkeit ist die Repräsentation als endliche oder unendliche Reihe. In manchen Fällen bietet eine solche Darstellungsform erweiterte Möglichkeiten zur Untersuchung von Funktionen bzw. zu deren Transformation insbesondere der Filterung. Es gibt eine Reihe von Transformationen vom Ortsraum in den so genannten Frequenzraum, wobei auch Transformationen in andere Räume denkbar sind. Erwähnt sei an dieser Stelle die Kosinus-Transformation, die große

Bedeutung bei der Bildkomprimierung erlangt hat oder die Wavelet-Tansformation. Eine Transformation, nämlich die Fourier-Transformartion soll nun genauer untersucht werden.

Fourier-Transformation:

Wie bereits erwähnt. Lassen sich Funktionen reeller Variablen oft als endliche oder unendliche Reihen darstellen. Besondere Bedeutung haben dabei die nach Jean Babtiste Joseph Fourier (1768-1830) [WJF01] benannten Reihen. Er erkannte, dass sich solche Funktionen als Überlagerung harmonischer Schwingungen unterschiedlicher Frequenz und Amplitude beschreiben lassen. Eine eindimensionale periodische Funktion *f(x)* mit der Periode $T = 2\pi$ lässt sich somit folgendermaßen darstellen:

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} (a_k \cos(kx) + b_k \sin(kx)) . \tag{3.15}$$

Die Amplituden a_k und b_k der einzelnen harmonischen Schwingungen ergeben sich aus:

$$\begin{aligned} a_k &= \frac{1}{\pi} \int_{-\pi}^{\pi} f(x)\cos(kx)dx, \qquad k = 1,2,..., \qquad T = 2\pi \\ b_k &= \frac{1}{\pi} \int_{-\pi}^{\pi} f(x)\sin(kx)dx, \qquad k = 1,2,..., \qquad T = 2\pi \end{aligned} \tag{3.16}$$

Für die Handhabung der Fourierreihen ist die trigonometrische Darstellung aus (3.15) nicht besonders gut geeignet, deshalb werden sie meist in einer Darstellung mit komplexwertigen Exponentialfunktionen verwendet. Der Zusammenhang ergibt sich aus der Euler-Identität:

$$e^{ikx} = \cos(kx) + i\sin(x) . \tag{3.17}$$

Dadurch erhält man folgende Darstellung:

$$f(x) = \sum_{k=-\infty}^{\infty} c_k e^{ikx}, \quad mit \quad c_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) e^{-ikx} dx = \begin{cases} \frac{a_0}{2} & k = 0 \\ \frac{1}{2}(a_k - ib_k) & k > 0 . \\ \frac{1}{2}(a_{-k} + ib_{-k}) & k < 0 \end{cases} \tag{3.18}$$

Die Koeffizienten c_k sind komplexwertig und stellen das diskrete Frequenzspektrum der Funktion *f(x)* dar [STN93].

Eine ausführlichere Darstellung der Fourier-Transformation würde den Rahmen dieser Arbeit sprengen. Für weiter führende Informationen sei an dieser Stelle auf [DRS92] verwiesen.

Obige Behandlung der Fourier-Transformation gilt für kontinuierliche Funktionen einer Veränderlichen. In der Bildverarbeitung haben wir es aber mit diskreten Funktionen zweier Variablen zu tun, somit sind obige Gleichungen ungeeignet. Für eine Transformation derartiger Funktionen wird die Transformation in der Form der *„Diskreten zweidimensionalen Fourier-Transformation“* (DFT) verwendet [STN93].

Diskrete Fourier-Transformation:

Zunächst wieder die Annahme es sei eine eindimensionale Funktion *f(x)* zu transformieren. Die diskrete Fourier transformierte *F(u)* der Funktion ist in diesem Fall definiert als:

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x) e^{\left(\frac{-i2\pi ux}{N}\right)}, \qquad u = 0,1,...,N-1. \tag{3.19}$$

Die Rücktransformtaion vom Frequenz- in den Ortsbereich ergibt sich zu:

$$f(x) = \frac{1}{N}\sum_{x=0}^{N-1} F(u) e^{\left(\frac{-i2\pi ux}{N}\right)}, \qquad x = 0,1,...,N-1. \tag{3.20}$$

Die Werte *f(x)* entsprechen in der diskreten Fourier-Transformation den Abtastwerten der Funktion an den Stellen 0, Δx, $2\Delta x$, … Je dichter man dabei die so genannten Stützstellen legt, desto genauer wird die kontinuierliche Funktion approximiert. Sind viele Stützstellen *N* bekannt, so können die auftretenden Frequenzen gut bestimmt werden. Bei digitalen Bildern entspricht *N* der Auflösung.

Der Term

$$k_u = e^{\frac{-i2\pi ux}{N}} \tag{3.21}$$

Wird als Kern der diskreten Fourier-Transformation bezeichnet. Er stellt die Basisfunktionen dar, welche nach der Euler-Identität (3.17) Sinus- bzw. Kosinusfunktionen unterschiedlicher Frequenz sind [STN93].

Analog dazu gilt im zweidimensionalen Fall für *M* x *N* Stützstellen bzw. einer Bildauflösung von *M* x *N*:

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) e^{-i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}, \qquad u = 0,1,...,M-1,\ \ v = 0,1,...,N-1. \tag{3.22}$$

Analog ist die Rücktransformation gegeben durch:

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v) e^{-i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}, \qquad x = 0,1,...,M-1,\ \ y = 0,1,...,N-1. \tag{3.23}$$

Fast-Fourier-Transformation:

Bei der Diskreten Fourier-Transformation bedarf es einer großen Anzahl von Multiplikationen und Additionen. Um die Berechnungszeit und den Speicherplatzbedarf von Algorithmen zur Berechnung der Transformation zu verringern wurde und wird ständig versucht dieses Verfahren zu optimieren. 195 wurde von Cooley und Tukey der *„Fast-Fourier-Transformationsalghorithmus“* (FFT) vorgeschlagen. Die Geschwindigkeits-steigerung wird dabei vor allem durch die Nutzung der Symmetrieeigenschaften einer DFT erreicht [STN93].

Es gibt heute eine Vielzahl von Optimierungen für den FFT-Algorithmus, bzw. eine Reihe von Implementierungen die für die jeweilige Hardware-Plattform optimiert wurden. Eine ausführliche Beschreibung des FFT-Algorithmus, sowohl eindimensional wie auch in mehreren Dimensionen, gibt es in [JHN91].

3.4.5 Transformation der Ortskoordinaten

In den bisher gezeigten Verfahren wurden Transformationsvorschriften immer auf die jeweiligen Grauwerte angewendet, wobei die Position der Bildpunkte unverändert blieb. Es kann aber in manchen Fällen sinnvoll sein, eine Transformation der Ortskoordinaten vorzunehmen. Vorstellbar wären dabei Beispielsweise eine Vergrößerung oder Verkleinerung des Bildes, um es an vorgegebene Abmessungen anpassen zu können, oder zwei Bilder die mit unterschiedlicher Auflösung aufgenommen wurden, miteinander vergleichen zu können. Es sei $C=s(x,y)$ das Ausgangsbild das in $C' = s'(x', y')$ übergeführt werden soll:

$$s'(x', y') = s(x, y) \tag{3.24}$$

wobei gilt,

$$x' = f_1(x, y) \text{ und } y' = f_2(x, y). \tag{3.25}$$

Somit wird erscheint der Grauwert $g=s(x,y)$ des Originalbildes im transformierten Bild an der Stelle x' und y'. Die Art der Transformation ist dabei abhängig von den beiden Funktionen f_1 und f_2 [HBR91].

Eine Möglichkeit ist die erwähne Vergrößerung oder Verkleinerung wobei die beiden Transformationsfunktionen f_1 und f_2 in diesem Fall die folgende Form haben:

$$f_1(x, y) = ax \text{ und } f_2(x, y) = ay. \tag{3.26}$$

Entzerrung:

Eine sehr wichtige Anwendung obiger Transformation ist die Entzerrung von Bildern. Durch das Linsensystem der Kamera erfährt das digitale Bild eine Verzerrung in den Ortskoordinaten. Das heißt, die Positionen von Objekten in einer aufgenommenen Szene werden am Bild nicht korrekt wiedergegeben. Um diese Linsenverzerrungen aus dem Bild zu

eliminieren wird eine Ortskoordinatentransformation nach dem obigen Verfahren durchgeführt. Die beiden Funktionen f_1 und f_2 sind dabei a priori nicht bekannt und müssen erst für das vorliegende Aufnahmesystem ermittelt werden. Eine gängige Methode hierfür ist die *„Passpunktmethode“*. Angenommen wird dabei die Transformationsfunktionen ließen sich durch lineare Polynome in x und y beschreiben:

$$x' = f_1(x,y) = a_0 + a_1 x + a_2 y\,, \tag{3.27}$$

$$y' = f_2(x,y) = b_0 + b_1 x + b_2 y\,. \tag{3.28}$$

Es gilt nun die sechs unbekannten Koeffizienten a_0 bis a_2 und b_0 bis b_2 zu bestimmen. Dazu wird die Position dreier Punkte (*Passpunkte*) in einem Referenzbild (Sollposition) und in dem daraus aufgenommenen Bild bestimmt. Man erhält daraus drei einander entsprechende Koordinatenpaare (u_i,v_i) «» (x_i,y_i) mit i=1,2,3. In die Funktionen (3.27) und (3.28) eingesetzt, erhält man daraus zwei lineare Gleichungssysteme der Form:

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} \tag{3.29}$$

und

$$\begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{pmatrix} \cdot \begin{pmatrix} b_0 \\ b_1 \\ b_2 \end{pmatrix}, \tag{3.30}$$

deren Lösung die sechs gesuchten Koeffizienten ergibt [HBR91].

Es gibt noch eine Reihe anderer Transformationsmethoden die an dieser Stelle unerwähnt bleiben. Für tiefer gehende Informationen sei an [HBR91] verwiesen.

3.5 Segmentierung

Mit der Segmentierung beginnt der Übergang von einer rein numerischen Darstellung des Bildes auf der Basis einer zweiwertigen Funktion der Grauwerte zu einer symbolischen. Es werden einfache Objekte wie Linien Flächen oder Punkte ermittelt, die aber noch keinem realen Objekt zugeordnet werden. Ziel der Bildsegmentierung ist es, das Bild in Teilbereiche mit gleichartigen, homogenen Eigenschaften zu unterteilen, die für eine Weiterverarbeitung von Interesse sein könnten. Diese Regionen werden in der englischen Fachliteratur of als *„regions of interest“* bezeichnet. Damit erfolgt eine erste Bedeutungszuweisung zu bestimmten Bereichen im Bild. Wichtig ist in der Praxis abzuklären, welche und wie viele Eigenschaften zu unterscheiden sind. Die beiden wichtigsten davon sind in diesem

Zusammenhang Kanten und bei Farbbildern Flächen homogener Farbe. Im Folgenden sollen nun einige Verfahren zur Segmentierung aufgezeigt werden [RDG93, STN93].

3.5.1 Schwellwertverfahren

Im einfachsten Fall kann ein Bild durch die Verwendung eines Schwellwertes (engl. *threshold*) segmentiert werden. Das Ergebnis ist dann ein Binärbild (*Binarisierung*). Dies ist besonders sinnvoll wenn man im vorneherein weiß, das sich die beobachtete Szene in „Objekt von Interesse“ und Hintergrund einteilen läßt. Auch die Anwendung mehrerer voneinander verschiedener Schwellwerte ist möglich. Man spricht in diesem Fall von Quantisierung des Bildes oder Aquidensitenbildung [HBR91]. Man erhält daraus als Ergebnis wiederum ein Graustufenbild mit einer reduzierten Anzahl der Grauwerte, abhängig von der Zahl der verwendeten Schwellwerte. Bei Farbbildern kann die Anwendung von Schwellwerten zur Farbidentifikation verwendet werden. Die daraus entstehenden Regionen homogener Farbe werden oft als *„Blobs“* bezeichnet. Entscheidet dabei ist der Farbraum (siehe Abschnitt 2.3) in dem die Segmentierung vorgenommen wird. Mathematisch genügt die Binarisierung folgendem Zusammenhang. Es sei $C=s(x,y)$ das Ausgangsbild, dann gilt:

$$s'(x,y)=\begin{cases}1, & falls \quad s(x,y)>c\\ 0, & falls \quad s(x,y)\le c\end{cases}. \tag{3.31}$$

Die Art und Weise nach der man den Schwellwert c wählt, ist entscheidet für eine sinnvolle Bildsegmentierung. Bei Farbbildern sind zur Identifikation einer Farbe zwei Schwellwerte notwendig:

$$s'(x,y)=\begin{cases}1, & falls \quad s(x,y)>c_1 \quad und \quad s(x,y)<c_2\\ 0, & sonst\end{cases}. \tag{3.32}$$

Konstanter Schwellwert:

Für Grauwertbilder ist es kaum möglich einen konstanten Schwellwert über alle Bilder einer Bildfolge zu verwenden. Somit ist, wenn wir hier von einem konstanten Schwellwert sprechen, nur konstant über das gesamte, gerade untersuchte Bild gemeint. Er muss aber für jedes Bild neu ermittelt werden, anderenfalls sind kaum gute Ergebnisse zu erzielen. Eine Möglichkeit einen günstigen Schwellwert zu finden, bietet das Histogramm. Besonders einfach ist es, wenn das Histogramm einen bimodalen Charakter aufweist (siehe Abschnitt 3.4.1). Der Schwellwert kann dann zwischen den beiden lokalen Maxima gewählt werden. Dies ist aber nur ein Indiz und weder ein notwendiges noch ein hinreichendes Kriterium dafür, dass das Schwellwertverfahren brauchbare Ergebnisse liefert [RDG93, STN93].

Multischwellwert:

Weist das Histogramm mehrere lokale Maxima auf, so kann man mehrere Schwellwerte im Sinne der oben erwähnten Quantiserung verwenden, um mehrere Bildprimitiven

unterscheiden zu können. Wichtig dabei ist, dass die Maxima weit genug auseinander liegen um eine deutliche Trennung zu ermöglichen. Für die Anwendbarkeit gilt dasselbe wie beim Verfahren mit konstantem Schwellwert.

Variabler Schwellwert:

Oben genannte Verfahren führen nur dann zum Erfolg, wenn bestimmte Grauwerte nur in einem Objekt vorkommen. Dies ist in der Praxis aber oft nicht der Fall. Dann empfiehlt sich ein Verfahren mit variablen oder dynamischen Schwellwert. Das Bild wird dabei in Regionen zerlegt, für die jeweils ein eigener Schwellwert berechnet wird. Im extremen Fall kann dabei auch für jeden Pixel ein neuer Schwellwert in Abhängigkeit seiner Nachbarn bestimmt werden. Die Art der Unterteilung bzw. die Größe der Regionen ist im Grunde beliebig. Es empfehlen sich aber im Allgemeinen Rechtecke. Auf jedes dieser Rechtecke kann nun entweder die Methode mit konstanten oder Multischwellwert angewendet werden [RDG93]. Werden die Rechtecke nicht allzu groß gewählt, so kann auch der Mittelwert über die Grauwerte des jeweiligen Bildausschnittes als Schwellwert verwendet werden. Besonders bei einer sehr feinmaschigen Einteilung wo jedes Rechteck ungefähr gleich viele Punkte vom Objekt wie vom Hintergrund enthält, eignet sich der Mittelwert sehr gut als Schwellwert [HBR91].

3.5.2 Kantendetektion

Ein in der Bildverarbeitung sehr wichtiges Bildprimitiv sind Objektkanten. Sie sind meist sehr einfach zu detektieren und geben oft sehr genau Aufschluss über Lage und Art des Objekts, weshalb die Kantendetektion zu den wichtigsten und eine der am besten untersuchten Methoden der Segmentierung gehört. Es gibt unzählige Vorschläge für Kantenoperatoren, sowohl im Frequenzbereich als auch im Ortsbereich, lineare und nichtlineare, ortsvariante und ortsinvariante, rekursive und nicht rekursive. Viele davon sind empirisch entstanden. Der Grund dafür ist, dass es keine allgemein gültige Zielsetzung gibt die sich mathematisch formulieren ließe, sondern die Bewertung des Ergebnisses menschlichen Kriterien unterliegt, die nicht objektiv und abstrakt spezifizierbar sind.

Eine umfassende Auflistung der möglichen Kantenoperationen ist nicht Sinn und Zweck und würde auch den Rahmen dieser Arbeit bei weitem sprengen. Deshalb beschränkt sich folgender Überblick auf die Anwendung differenzieller Operatoren realisiert mittels der diskreten Faltung im Ortsraum. Eine Übertragung in den Frequenzraum lässt sich daraus leicht bewerkstelligen. Es wird in diesem Zusammenhang auch auf die zugrunde liegende mathematische Beschreibung der einzelnen Operatoren in ihrer kontinuierlichen Form verzichtet. Anschließend wird eine Variante eines nichtlinearen Kantenoperators vorgestellt.

Gradient:

Ausgehend von der bereits bekannten Darstellung des Grauwertbildes als $C=f(x,y)$, bedeutet eine Kante eine starke ortsbegrenzte Änderung von f innerhalb Δx oder Δy. Sichtbar wird

dies im Gradienten oder der Ableitung von *f(x,y)* als lokales Maximum, oder als Nullstellen in der zweiten Ableitung. Für beide Fälle gibt es entsprechende Operatoren. Der Gradient bzw. die erste Ableitung von *f(x,y)* ist gegeben durch:

$$\nabla f(x,y) = \left(\frac{\partial f(x,y)}{\partial x}, \frac{\partial f(x,y)}{\partial y} \right), \tag{3.33}$$

der Betrag durch

$$|\nabla f(x,y)| = \sqrt{\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2} \; . \tag{3.34}$$

Er zeigt in Richtung der maximalen Grauwertänderung.

Die meisten in der Praxis verwendeten Kantenoperatoren verwenden das Prinzip der Gradientenbildung, da die Kantenrichtung in den meisten Fällen gebraucht wird und diese Operatoren zudem wenig rauschempfindlich sind. Es gibt aber auch rotationssymetrische Kantenoperatoren die keinerlei Informationen über die Richtung der Kante liefern.

Aufgrund der bereits erwähnten Eigenschaft des Auges empfindlicher auf Helligkeitsänderungen als auf Änderungen der Farbe zu reagieren, werden Kantenoperatoren meist ausschließlich auf den Helligkeitskanal angewendet. Dieser entspricht je nach Farbraum einer relativen Helligkeit oder der totalen Lichtstärke (siehe Abschnitt 2.3). Es muss im Einzelnen entschieden werden welche Grauwertdarstellung sich am besten eignet. In manchen Fällen kann es aber auch sinnvoll sein Kanten in den Farbkanälen zu untersuchen um Grenzen an Oberflächen finden zu können, die sich nicht so sehr in ihrer Helligkeit wohl aber in der Farbe unterscheiden.

Laplace-Filter:

Die Filtermaske für das diskrete Laplace-Filter auf Basis der Faltung zeigt Abbildung 3.17.

0	1	0
1	-4	1
0	1	0

Abbildung 3.17: 3 x 3 Faltungsmaske für das Laplace-Filter

Es wirkt wie ein Hochpassfilter. Nur die hohen Frequenzen werden berücksichtigt, was dazu führt, dass er sehr empfindlich gegenüber Bildrauschen ist und auch einzelne Punkte als Kanten detektiert. Das Laplace-Filter ist zwar nicht rotationsymmetrisch da es auf diagonale Kanten stärker reagiert als auf horizontale oder vertikale, trotzdem erhält man keine Information über die Richtung des Gradienten [RDG93].

Sobel-Filter:

Wird die Information über die Richtung der Kante benötigt, so empfiehlt sich das Sobel_Filter. Es definiert einen Satz von Faltungsmasken für die horizontalen und vertikalen Kanten. Der Vorfaktor von 1/8 in Abbildung 3.18, die einen Satz von Sobel-Filtern zeigt, dient nur zur Normierung. Die linke Bildhälfte zeigt die Maske für die vertikalen, die rechte für die horizontalen Kanten [JHN91].

$$\frac{1}{8}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \text{ für } \frac{\partial}{\partial x} \qquad \frac{1}{8}\begin{bmatrix} 1 & 2 & -1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \text{ für } \frac{\partial}{\partial y}$$

Abbildung 3.18: Sobel-Filter

3.5.3 Kantennachbearbeitung

Kantenverkettung:

Unter der Kantenverkettung versteht man die Verbindung einzelner Kantenstücke um so einen durchgehenden Kantenverlauf zu Erzeugen. Dadurch werden kleine Lücken die durch eine unsaubere Kantendetektion entstehen können geschlossen. Das in 3.5.4 beschriebene Closing angewendet auf das Kantenbild kann solche kleinen Lücken schließen.

Kantenverjüngung:

Manche Operatoren zur Gradientenbildung erzeugen Kantenlinien von mehreren Pixeln Breite. Für eine nachfolgende Merkmalsextraktion sind aber oft ausgedünnte Kanten mit nur einem Pixel Breite von Vorteil. Zu diesem Zweck werden die Kanten verjüngt. Ein gängiges Verfahren zur Kantenverjüngung findet sich in [STN93].

3.5.4 Morphologische Operatoren

Morphologische Operatoren verändern die Gestalt von Bildprimitiven [BBS91]. Das heißt, es werden die Flächen oder die Ränder der Bildelemente verändert, daher der Name.

Erosion - Dilatation:

Bei der Anwendung der Erosion und der Dilatation wird meist eine 3 x 3 Matrix zugrunde gelegt. Das heißt, ein Ergebnispunkt wird bestimmt von den ihn umgebenden 8 Nachbarn ähnlich der globalen Operatoren aus 3.4.3 [STN93].

Bei der *Erosion* werden Punkte vom Rand oder Inneren eines Bildelementes entfernt. Es wird ein Bildpunkt ins Ergebnis übertragen wenn mindestens T Punkte aus der Maske mit den

Bildpunkten übereinstimmen. Bei einer 3 x 3 Matrix bzw. Maske wird meist *T=9* gesetzt. Das heißt, ein Punkt wird im Ergebnisbild gesetzt, sofern beim darüber legen der Maske alle 9 Punkte in der Maske Punkte des Bildelements im Originalbild sind [STN93]. Abbildung 3.19 zeigt das Ergebnis der Anwendung der Erosion auf ein 7 x 10 Pixel großes Bild. Die schwarzen Bildpunkte entsprechen dabei dem zu untersuchenden Merkmal nach der Segmentierung, weiß ist der Hintergrund.

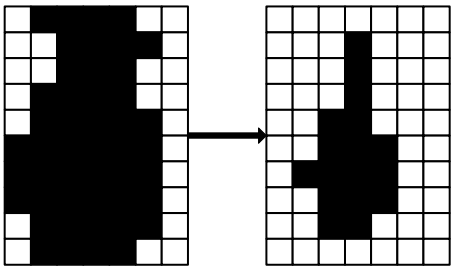

Abbildung 3.19: Originalbild und Ergebnis einer Erosion mit einer 3x3 Maske

Formal betrachtet ergibt sich ein Punkt $f'(x,y)$ im Ergebnisbild aus dem Ausgangsbild *f(x,y)* zu:

$$f'(x,y) = \begin{cases} 1 falls \sum_{i=-n}^{n} \sum_{j=-n}^{n} f(x+i, y+j) \geq T \\ 0 sonst \end{cases} \tag{3.35}$$

Die *Dilatation* beschreibt dem Namen nach einen Wachstums- oder Ausdehnungsvorgang. Auch hier wird abhängig von den Nachbarn eines Bildpunktes ein Ergebnispunkt gesetzt oder nicht. Der unterschied zur oben beschriebenen Erosion ist lediglich die Wahl von *T* die in diesem Fall gleich eins gesetzt wird. Somit wird bei einer Dilatation ein Punkt im Ergebnisbild gesetzt sofern sich mindestens ein Punkt des untersuchten Merkmals innerhalb der Maske befindet [STN93]. Abbildung 3.20 zeigt dies wiederum mit einem 7 x 10 Pixel großem Bild.

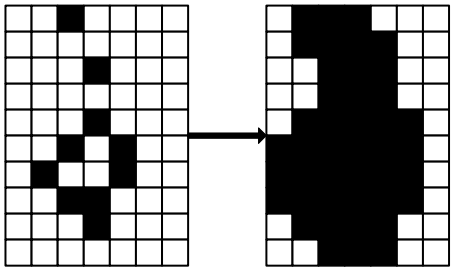

Abbildung 3.20: Originalbild und Ergebnis einer Dilatation mit einer 3x3 Maske

Damit ist die formale Beschreibung der Dilatation ebenfalls durch obige Gleichung, für den Fall *T=1*, gegeben.

Das Ergebnis der Erosion stellt somit eine Schnittmenge zwischen der über das Originalbild bewegten Maske und dem Originalbild selbst dar, die Dilatation hingegen die

Vereinigungsmenge. Die Wirkung der beiden Operatoren ist zwar gegenläufig aber nicht invers, da die beiden Operatoren nichtlinear sind [STN93].

Opening - Closing:

Wie bereits erwähnt lässt sich die Wirkung einer Erosion zwar durch eine Dilatation kompensieren nicht aber umkehren. Die Wirkung der beiden Operatoren heben sich nicht auf. Im Gegenteil. Die Reihenfolge der Anwendung der beiden Operatoren führt zu völlig unterschiedlichen Ergebnissen. Eine aufeinander folgende Anwendung der Erosion und der Dilatation wird als *Opening* bzw. *Closing* bezeichnet.

Unter einem *Opening* versteht man dabei die Anwendung einer Erosion gefolgt von einer Dilatation. Diese Kombination dient vorwiegend zur Beseitigung von Störungen in Form vereinzelter Pixel, kleiner Strukturen oder Auswüchsen. Die Störungen müssen dabei kleiner sein als die Erosionsmaske [STN93].

Beim *Closing* wird eine Erosion nach einer Dilatation angewendet. Sie dient zur Schließung kleinerer Lücken.

Die Anwendung der morphologischen Operatoren kann ein Vorverarbeitungsschritt für die auf die Segmentierung folgende Schicht im Schichtenmodell sein. Das Opening oder Closing kann aber auch dazu benutzt werden, um bestimmte Strukturen im Bild zu detektieren, wobei die Verfahren somit selbst zur Objektidentifikation dienen, weshalb sie im Abschnitt 3.6 noch einmal Erwähnung finden.

3.6 Merkmalsextraktion und Objektidentifikation

Bei der Objektidentifikation geht es nun darum die ermittelten Segmente den Objekten von Interesse zuzuordnen. Dazu werden oft Merkmale von Segmenten wie etwa „kreisförmig“ oder „gerade“ bei Kanten, ermittelt, da diese Merkmale in manchen Fällen bereits eindeutig Objekten zugeordnet werden können. Zum Beispiel könnte eine kreisförmige Kante einem Ball zugeordnet werden. Aus diesem Grund sind die Begriffe Merkmalsextraktion und Objektidentifikation eng miteinander verbunden. Die Merkmalsextraktion wird oft auch als Mustererkennung bezeichnet.

3.6.1 Opening und Closing

Die beiden morphologischen Operatoren wurden bereits in Abschnitt 3.5.4 im Hinblick auf die Segmentierung erklärt. Die Opening und Closing Operationen können aber auch gezielt zur Objektdetektion eingesetzt werden. Für das Opening muss dazu die Maskengröße und Maskenform so gewählt werden, dass die Objekte von Interesse garantiert gelöscht werden. Das Ergebnisbild enthält dann alle größeren Strukturen sowie den Hintergrund. Das so erhaltene Bild wird mit dem Ausgangsbild mittels einer XOR-Funktion verknüpft. In dem

daraus gewonnenen Bild sind die Objekte als Ränder der größeren Strukturen identifizierbar, wobei das Bild aber noch Störungen und kleinere Objekte enthält. Durch eine nachfolgende Erosion mit einer speziellen Maske, die die Form der gesuchten Objekte berücksichtigt, werden alle nicht erwünschten Strukturen gelöscht. Eine nachfolgende Dilatation reproduziert dann die gesuchten Objekte in etwa ihrer Originalgröße [STN93]. Für die Aufgabenstellung dieser Arbeit ist aus dem bereits genannten Grund der variablen Objektgröße im Bild, abhängig von der Entfernung des Roboters vom Objekt, diese Methode jedoch unbrauchbar.

3.6.2 Vektorisierung

Kettencodes:

Kettencodes erzeugen nicht direkt Vektoren sondern Listen von Daten die Objektkanten beschreiben. Dabei werden ausgehend von einem Kantenpunkt seine acht möglichen Nachbarn auf weitere Kantenpixel hin untersucht. Wurde ein weiteres gefunden, so wird ausgehend von diesem wiederum in seiner Umgebung nach weiteren gesucht. So entsteht eine Kette von Kantenpunkten die zu einer Kante gehören. In der Folge kann dann diese Kette auf spezielle Charakteristika untersucht werden um ihr eventuell ein Merkmal zuordnen zu können. Interessant sind dabei Fragen wie: Liegen die Punkte auf einer Gerade oder ist die Kette geschlossen? Damit lassen sich Merkmale wie „kreisförmig“ oder „linienförmig“ bestimmen [STN93].

Hough-Transformation:

Das Grundprinzip der Hough-Transformation wurde erstmals in einer Patentschrift von P. V. C. Hough 1962 veröffentlicht [HOU62]. Sie ist ein heutzutage oft verwendetes Verfahren zur Merkmalsextraktion. Die Hough-Transformation eignet sich zum Auffinden von Geraden und Kreisen bzw. Ellipsen im Kantenbild. Sie beschreibt eine Abbildung des zweidimensionalen Bildraums auf den ebenfalls zweidimensionalen Parameterraum. Jedem Kantenpunkt im Bildraum wird dabei eine Gerade im Parameterraum zugeordnet. Punkte werden im Parameterraum durch die ihre Koordinaten x und y repräsentiert, im Parameterraum durch die Steigung a und den Achsabschnitt b. Schneiden sich mehrere Geraden im Parameterraum wird dadurch eine Gerade im Bildraum spezifiziert (siehe Abbildung 3.21). Verfahren um diese Schnittpunkte zu finden, zeigt [STN93].

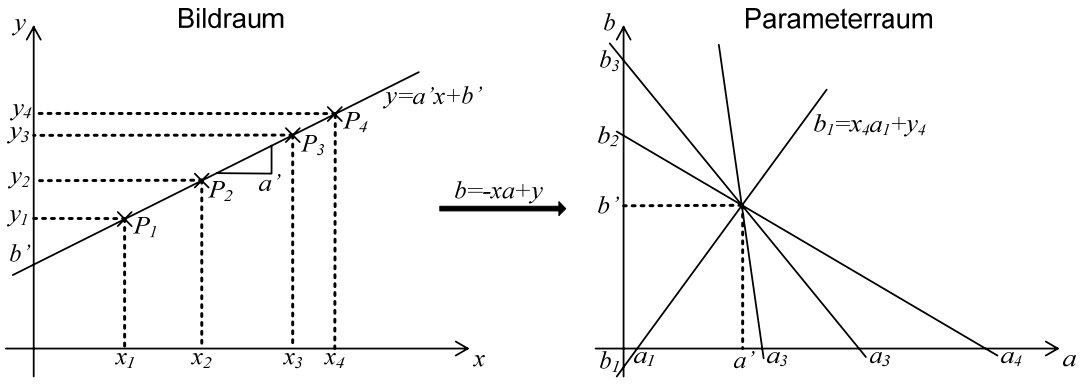

Abbildung 3.21: Die Hough-Transformation für Geraden

Wird der Parameterraum auf mehrere Dimensionen erweitert, so ist damit auch die Erkennung von Kreissegmenten oder Ellipsen möglich [STN93]. Durch die komplexeren Parameter und die Dimensionserweiterung wird der Rechenaufwand aber stark erhöht. Diese Art der Merkmalsextraktion ist deshalb für die in dieser Arbeit gefordert hohen Frameraten eher ungeeignet.

3.6.3 Modellierung

In den meisten Fällen basiert die Objektidentifikation nach erfolgter Segmentierung auf den Vergleich mit einem Modell. Dieser Vergleich kann auf unterschiedlichste Art und Weise erfolgen, ebenso wie die Erstellung des Modells. Das Modell besteht dabei aus einfachen geometrischen Primitiven die mit der Bildstruktur verglichen werden. Jene Struktur im Bild, die den höchsten Übereinstimmungsgrad hat, wird dem gesuchten Objekt zugeordnet. Es sei noch einmal darauf hingewiesen, dass für das vorliegende Problem aufgrund der variablen Objektgrößen keine statischen Modelle verwendet werden können (siehe Abschnitt 1.2). Eine Art der Modellrepräsentation und die dazugehörenden mathematischen Methoden zum Vergleichen der Modell- mit der Bildstruktur finden sich in [RDG93].

Analytische Modellierung:

Entsprechen die gesuchten Objekte einfachen geometrischen Formen, wie etwa Kreisen oder Rechtecken tun dies auch die Modelle. Damit ist oft eine analytische Modellierung brauchbar. Gemeint ist damit, dass die mathematischen Eigenschaften des Modells zur Objektidentifikation eingesetzt werden können. Kreise und Rechtecke lassen sich mathematisch eindeutig beschreiben und haben bestimmte mathematische Charakteristika. Nach diesen kann in der Bildstruktur gesucht und somit das Objekt von Interesse identifiziert werden. Diese Art der Modellierung ist auch bei variabler Objektgröße brauchbar. Deshalb wird das Augenmerk darauf gelegt.

3.7 Interpretation

Die Interpretationsschicht ist die oberste im Modell aus Abbildung 3.1. Nach der Identifikation der Objekte werden die daraus gewonnen Informationen nun hinsichtlich der gestellten Aufgabe bewertet. Es können an dieser Stelle keine allgemeinen Verfahren gezeigt werden, da die Interpretation der Ergebnisse aus einer Objektidentifikation vollständig von der gestellten Aufgabe abhängt. Es kann dies beispielsweise die Bestimmung der Entfernung zu den detektierten Objekten sein, wie es im Rahmen dieser Arbeit gefordert ist, aber auch der Versuch, die beobachtete Szene einer Aktion zuzuordnen wie etwa: „Hand greift nach einer CD“. In der Qualitätskontrolle unterliegt die Bewertung der Ergebnisse einer Objektdetektion wiederum völlig anderen Kriterien (siehe dazu [DSW02]).

3.8 Rückkopplungen

Im Schichtenmodell aus Abbildung 3.1 sind mögliche Beeinflussungen einer Schicht auf eine der vorhergehenden eingezeichnet. Dies ist prinzipiell möglich und kann in manchen Fällen sehr sinnvoll sein. Es gibt aber keine allgemeinen Verfahren oder Vorschriften die eine solche Rückkoppelung beschreiben, weshalb an dieser Stelle nur kurz darauf hingewiesen werden soll. Es geht dabei primär um mögliche Eigenschaften eines Bildes die in einem Verarbeitungsschritt ermittelt werden. Diese Eigenschaften könnten dann dazu verwendet werden um die Parameter eines Verfahrens in Hinblick auf die Weiterverarbeitung zu optimieren.

Denkbar wäre zum Beispiel, dass bei einer Kantendetektion eine unzureichende Anzahl an Kanten detektiert wird, obwohl aufgrund der aufgenommenen Szene eine größere Anzahl erwartet werden muss. Möglicherweise liegt dass an einem zu geringen Kontrast des Ausgangsbildes, der dann im nächsten Bild oder auch noch für das gerade untersuchte, mit Hilfe eines der in Abschnitt 3.4.2 beschriebenen Verfahren, verbessert werden könnte. Eine andere vielfach angewendete Methode ist die Beeinflussung der Vorverarbeitung durch die Interpretationsschicht. Es kommt in der Praxis oft vor, dass ein Objekt über eine Reihe von Bildern hinweg verfolgt werden muss. Hat man das Objekt gefunden, und soll es in den anschließend aufgenommenen Bildern verfolgt (engl. *tracking*) werden, so ist es in Hinblick auf eine Geschwindigkeitssteigerung der Erkennung sinnvoll, die Vorverarbeitung für die nächsten Bilder auf die Umgebung der detektierten Position hin zu optimieren.

Dieses Kapitel erhebt keinerlei Anspruch auf Vollständigkeit. Es sollte hier nur ein Überblick gegeben werden über die gängigsten Methoden der Bildverarbeitung auf die die meisten, auch derzeit entwickelten Verfahren, basieren. Immer wieder kommen neuere Methoden hinzu, oder werden bestehende weiterentwickelt bzw. neue Varianten derselben vorgestellt.

Besonders durch die fortschreitende Entwicklung der Rechenleistung werden vor allem immer aufwendigere Verfahren ersonnen die die Probleme der Bildverarbeitung durch höheren Rechenaufwand in den Griff zu bekommen versuchen. Diese Algorithmen benötigen aber eine Rechenleistung wie sie heute verfügbare Systeme die vom Leistungsverbrauch und ihrem Platzbedarf innerhalb der Vorgaben für das Mirosot Roboter-Fußballspiel bleiben, nicht bieten können. Des Weiteren bedarf es bei Problemen der Bildverarbeitung oft individueller, den Bedingungen angepasste Lösungen, die oft nicht verallgemeinert werden können. Aus den in Abschnitt 1.2 genannten Gründen, wurden Verfahren die ein a priori Wissen über die Art der Beleuchtung voraussetzen, ganz außer Acht gelassen.

Kapitel 4 Systemarchitektur und Wahl der Komponenten

Im Abschnitt 2.3.3 wurde etwas vorgegriffen und ein direkter Sprung von der analogen in die digitale Welt gemacht. In diesem Kapitel soll nun unter anderem gezeigt werden, wie dieser Schritt technisch realisiert wird, in dem einige grundlegende für die Arbeit relevante Bildaufnahmeverfahren erläutert werden und auch konkrete diesbezügliche am Markt erhältliche Hardware vorgestellt wird. Neben der Bildaufnahme wird aber auch das Gesamtkonzept der Objekterkennung im Rahmen der im Punkt 1.2 dargelegten Spezifikationen erläutert.

4.1 Rahmenbedingungen

In Abschnitt 1.2 wurden bereits einige Anforderungen an das System zusammengefasst. Es sollen an dieser Stelle zwei Punkte davon, die für die Entwicklung der Objekterkennung besonders bedeutsam sind, noch einmal kurz beleuchtet und zusätzliche, für die Systemarchitektur relevante Eigenschaften, aufgeführt werden.

4.1.1 Abmessungen

Die Abmessungen des Roboters sind nach dem FIRA-Reglement auf 7,5 cm Kantenlänge begrenzt (siehe Abschnitt 1.1.1). Abbildung 4.1 zeigt den mechanischen Aufbau des Roboters Tinyphoon. Die Platine für das Vision System wird oben an dieser Konstruktion befestigt, womit die Außenabmessungen auf 7,5 cm beschränkt sind. Auf diesem Print soll aber auch die Motorsteuerung untergebracht werden, weshalb man grob mit einer zur Verfügung stehenden Fläche von 7,5 x 3,5 cm rechnen kann.

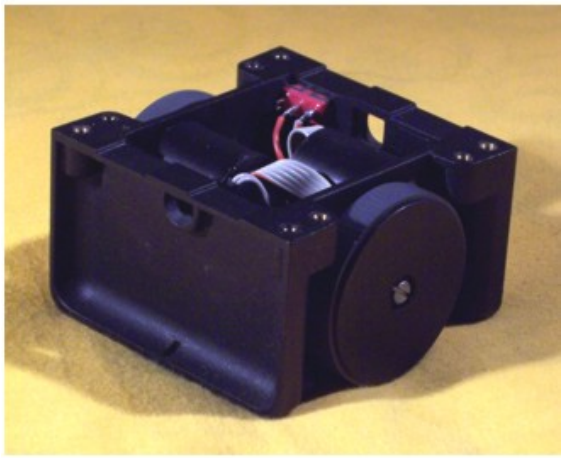

Abbildung 4.1: Mechanische Konstruktion des Roboters

Noch nicht berücksichtigt in dieser Kalkulation sind die Abmessungen des Aufnahmesensors und dessen Montage. Der Spielraum für die Abmessungen der Kamera ist allerdings gering, da der Raum im inneren der Mechanik großteils von den Motoren und dem Akku ausgefüllt wird. Nach oben steht ungefähr 1 cm zur Verfügung aufgrund der beschränkten Höhe von 7,5 cm, innerhalb der auch noch die Strategieeinheit untergebracht werden und für eventuelle Erweiterungen noch Platz sein soll (siehe [STM06]).

4.1.2 Leistungsverbrauch

Die überschlagsmäßige Rechnung aus Abschnitt 1.2 kalkuliert mit einer maximal zur Verfügung stehenden Leistung von zwei Watt für die gesamte Einheit zur Objekterkennung. Dies beinhaltet die Aufnahmeeinheit, den Prozessor mit der benötigten Peripherie und den Arbeitsspeicher.

4.2 Architektur des Objekterkennungssystems

In Abbildung 4.2 findet sich eine Gesamtübersicht über die Architektur des Objekterkennungssystems. Der mittlere, grün hinterlegte Block beschreibt das System. Der rechte Block zeigt die entsprechenden Schichten des Modells aus Abbildung 3.1. Die Systemarchitektur wurde dabei stark an das Schichtenmodell zur Bildverarbeitung angelehnt. Der linke Block ordnet die einzelnen Schichten bzw. Teile des Systems den Komponenten zu, für die in den folgenden Abschnitten dieses Kapitels eine Auswahl getroffen wird.

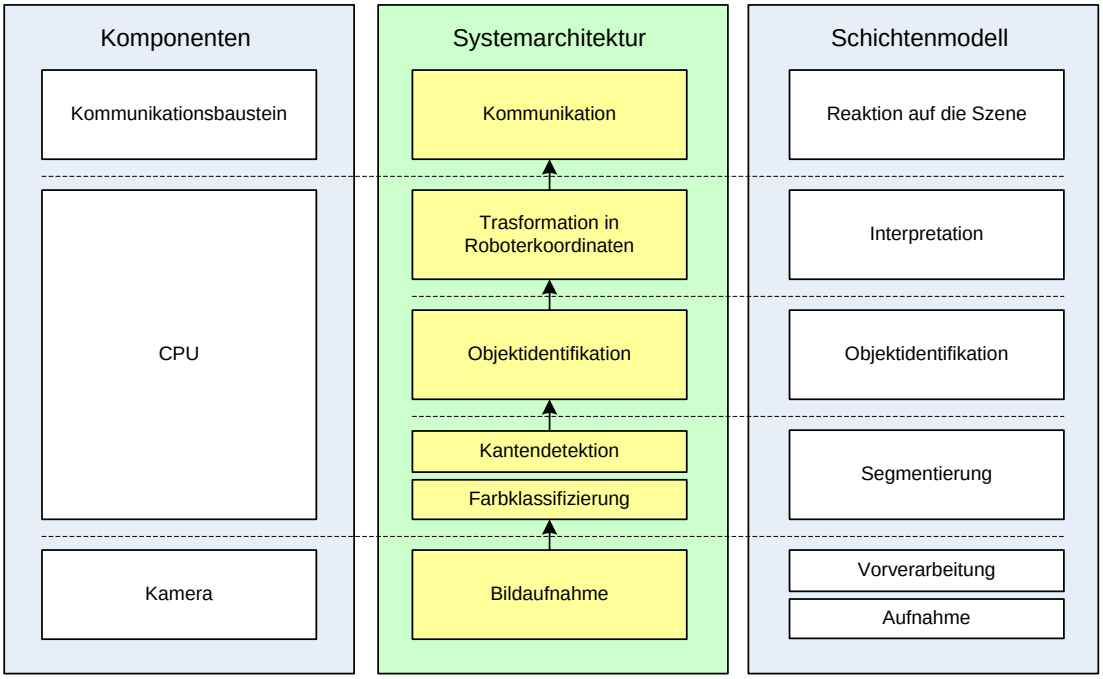

Abbildung 4.2: Architektur des Objekterkennungssystems

4.3 Bildaufnahme

Die Verarbeitungsschritte von einer natürlichen Szene bis hin zu deren digitalen Repräsentation zeigt Abbildung 4.3. Das von den Objekten ausgesandte Licht wird über ein Linsensystem dem Objektiv zum eigentlichen Fotosensor geleitet und dort in ein elektrisches Signal umgewandelt. Anschließend wird es mittels eines Analog-Digital-Wandlers (*ADC*) quantisiert und steht zur Weiterverarbeitung zur Verfügung.

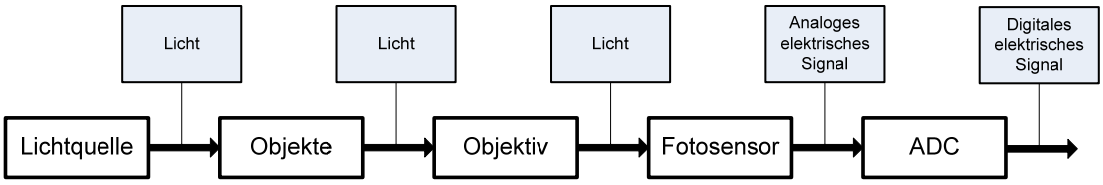

Abbildung 4.3: Kamerasystem

4.3.1 Objektiv

Das Objektiv ist entscheident für die Qualität des aufgenommenen Bildes und muss auf die Entfernung des Kamerasystems vom betrachteten Objekt, die Beleuchtungsstärke und die

Größe der Chipfläche abgestimmt werden. Ist die Entfernung des Objekts vom Objektiv nicht konstant, so bedarf es für die Scharfstellung der Möglichkeit den Focus zu variieren. Dies wird erreicht in dem die Linsen im Objektiv gegeneinander verschoben werden. Bei automatischen Bilderkennungssystemen wie es im Roboter der Fall ist, kann keine dauernde manuelle Nachstellung des Fokus durchgeführt werden, weswegen ein automatischer Fokus (*Autofokus*) wünschenswert wäre. Dieser bedarf aber eines eigenen Elektromotors und einer aufwendigen Mechanik, weshalb die Abmessungen der am Markt erhältlichen Objektive mit Autofokus den Anforderungen an unseren Roboter nicht genügen. Es gibt zwei grundsätzliche Verfahren zur Einstellung der Schärfe: aktiv und passiv. Bei ersterem wird eine Abstandsmessung zum Objekt mittels Infrarot oder Ultraschall durchgeführt, letzteres verwendet das aufgenommene Bild um den Kontrast zu berechnen und variiert die Linseneinstellungen bis er maximal wird. Der große Nachteil ist, dass beide Verfahren Zeit brauchen, bis sie die Schärfeneinstellung vorgenommen haben, nicht zuletzt aufgrund der mechanischen Trägheit des Systems und letzteres zudem sehr rechenintensiv ist. Wenn wir uns die Ballgeschwindigkeit und die dadurch benötigte Framerate, die in Abschnitt 1.2 abgeschätzt wurde, in Erinnerung rufen, so reicht die Zeit nicht aus um eine Schärfeneinstellung mittels Autofokus durchzuführen, da sich das Objekt zwischen zwei Frames bereits erheblich weiterbewegt haben kann, auch durch die Eigenbewegung des Roboters selbst. Wir beschränken uns deshalb auf ein Objektiv mit fixem Fokus.

Für die Auswahl des Objektivs sind Grundkenntnisse in der Optik von Vorteil. Der Umfang der Arbeit erlaubt an dieser Stelle keine Einführung in dieses sehr umfassende Gebiet. Es gibt eine Vielzahl von Fachbüchern, die sehr gut in diese Thematik einführen. Stellvertretend sei an dieser Stelle auf [HFR03] und [KHL04] hingewiesen.

Objektivhersteller bieten auf ihren Internetseiten Onlinekalkulatoren an, um aus ihrem Sortiment nach Eingabe der Größe des Objekts und dessen Distanz zur Kamera, bzw. der Chipfläche des Fotosensors und dessen Abstand zum Objektiv, das passende auswählen zu können [DMN06, SCL06].

4.3.2 Fotosensor

Es gibt verschiedene Arten von Fotosensoren zur Umwandlung des Lichts in elektrische Signale. Die zwei gebräuchlichsten Technologien hierfür sind CCD und CMOS.

CCD-Sensoren

Diese Technologie wurde in den 70er Jahren in den Bell-Laboratorien entwickelt. Ursprüngliches Ziel war es einen neuen Halbleiterspeicher zu entwerfen.

Grundlage für den CCD-Sensor ist der MOS-Kondensator (*MOS-Capacitor*). MOS steht für *„Metal-Oxide-Semiconductor“*. Es handelt sich dabei um einen Feldeffekttransistor dessen Gate-Kapazität von einfallenden Photonen gesteuert wird. Abbildung 4.4 zeigt den Querschnitt durch einen MOS-Capacitor.

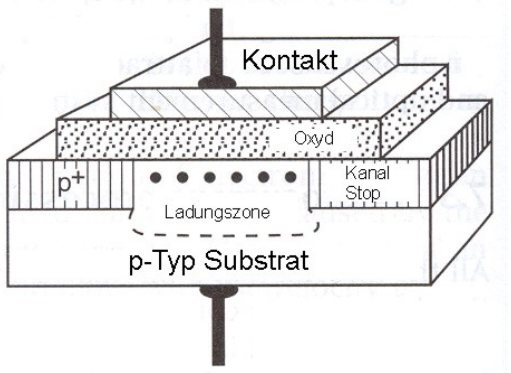

Abbildung 4.4: Querschnitt durch einen MOS-Capacitor

Auf die Metalloxydschicht auftreffende Photonen werden je nach Wellenlänge reflektiert oder absorbiert. Im Bereich des sichtbaren Lichtes durchdringen sie die Schicht und lösen Ladungen aus dem Valenzband die durch die Potentialbarriere im Topf des Substrats gefangen bleiben. Je länger die Beleuchtungsdauer (engl. *„exposure time"*) und je höher die Intensität des eintreffenden Lichts desto mehr Ladungen befinden sich im Topf. Nach einer definierten Beleuchtungsdauer ist die Menge der Ladungen im Topf proportional zur Beleuchtungsstärke. Jeder Potentialtopf kann nur eine bestimmte Menge an Ladungen aufnehmen. Wird diese Menge überschritten fallen Ladungen in den Topf einer benachbarten Kapazität. Dies führt zum so genannten „Blooming-Effekt" bei Kameras.

In einem CCD-Sensor sind solche MOS-Capacitors in einer Matrix angeordnet. Jede Kapazität entspricht dabei einem Pixel. Das Auslesen der Ladungen der einzelnen Kapazitäten gab dem Sensor seinen Namen. CCD steht für *„Charge-coupled devices"*. Frei übersetzt bedeutet dies ladungs-gekoppeltes „Device" und deutet damit auf die Art des Auslesens der Ladungen in den Potentialtöpfen hin. Die Ladungen der untersten Zeile werden seriell durch zyklische unterschiedliche Polungen, das heißt, einen entsprechenden Spannungsverlauf an den Elektroden, bis an den Rand transportiert, wo sie verstärkt und detektiert werden. Anschließend wird die nächste Zeile ausgelesen. Abbildung 4.5 zeigt den Ladungstransport über eine Zeile und über den gesamten Sensor. Durch dieses Verfahren kann immer nur der gesamte Sensor ausgelesen werden und nie ein einzelnes Pixel.

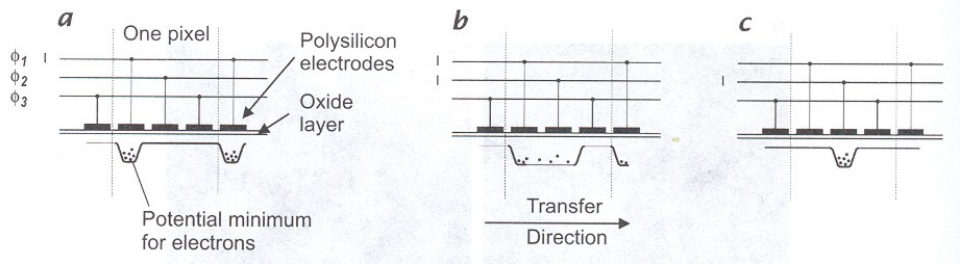

Abbildung 4.5: Ladungstransport in einem CCD-Sensor

Es bedarf einer aufwendigen Takt-Logik für das Auslesen der Ladungen, die meist über externe Bausteine realisiert wird und nicht am Sensorchip selbst integriert ist. Übliche

Taktraten liegen bei rund 40 Mhz [JHG99]. Der Ladungsverlust während des Transports von einem Potentialtopf zu einem anderen ist heutzutage vernachlässigbar.

CMOS-Sensoren

Erst Anfang der 90er Jahre gelang es, diese Technologie für bildgebende Sensoren in den Griff zu bekommen, um Kameras mit entsprechender Qualität produzieren zu können. Durch die konstante Verbesserung und Weiterentwicklung, verbreitet sich ihr Einsatzgebiet stetig.

Die lichtempfindlichen Elemente der CMOS-Sensoren sind Fotodioden nach dem Schema aus Abbildung 4.6. Ein in Sperrichtung gepolter pn-Übergang dient hier als Potentialtopf. Das am pn-Übergang anliegende Feld trennt Ladungen die durch Einwirkung von Photonen im Feld frei werden. Die Ansammlung der Ladungen an der Kapazität des pn-Übergangs verringert dessen Sperrspannung, die gemessen werden kann. Die Differenz zwischen der Spannung nach der Belichtungszeit und der ursprünglichen Sperrspannung ist proportional zur Beleuchtungsstärke.

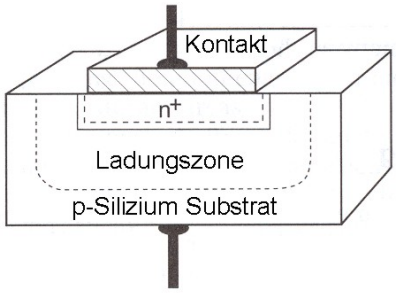

Abbildung 4.6: Querschnitt durch eine Fotodiode

Das besondere an den CMOS-Sensoren ist dabei, dass die Transistoren für das Auswerten der Ladung direkt an jedem Pixel untergebracht werden. Diese Architektur wird „Aktive Pixel Sensor“ (APS) Technologie genannt. Abbildung 4.7 zeigt den Schaltplan eines Pixels in dieser Architektur. Der Reset-Transistor dient zum Entladen des pn-Übergangs und dem Wiederherstellen der ursprünglichen Sperrspannung von typisch 3 bis 5 Volt [JHG99]. Der als Sourcefolger geschaltete Transistor dient als Verstärker. Somit besitzt in CMOS-Sensoren im Unterschied zu den CCDs jeder Pixel seinen eigenen Verstärker. Der dritte Transistor dient als Schalter um es der Auswerteelektronik zu ermöglichen das Auslesen der Pixel zu steuern. Dies erlaubt es jeden Pixel einzeln, gezielt auszulesen.

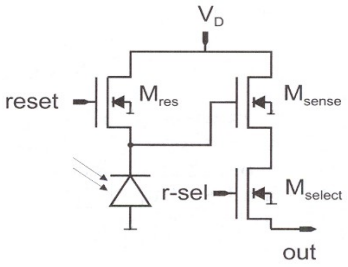

Abbildung 4.7: Schaltplan eines Pixel eines CMOS-Sensors in APS-Technologie

Durch das Integrieren der drei zusätzlichen Transistoren verringert sich aber die dem Licht ausgesetzte Fläche pro Bildpunkt wodurch sich die Lichtempfindlichkeit des Sensors insgesamt reduziert. Das Verhältnis zwischen Pixelfläche und lichtempfindlichen Teil des Pixels nennt man *„Apertur“* (engl. *„optical fill factor“*). Er liegt bei den CMOS-Sensoren üblicherweise zwischen 30 und 80 %. Da CMOS-Sensoren im Gegensatz zu den CCDs einen Verstärkungstransistor pro Pixel besitzen, erhöht sich das konstante Offsetrauschen. Dieses *„fixed pattern noise“* genannte Rauschen entsteht durch vom Herstellungsprozess bedingten Unterschieden in den einzelnen Verstärkungstransistoren. Da es sich aber um ein konstantes Rauschen handelt, kann es durch eine Nachbearbeitung des Bildes großteils eliminiert werden, was heutzutage bereits am Sensor selbst durch einen integrierten digitalen Signalprozessor bewerkstelligt wird. Darin liegt auch ein großer Vorteil der CMOS-Sensoren. Die gesamte Auswerteelektronik, die ADCs und oft auch ein DSP, der bereits Funktionen wie Bildstabilisierung, Weißabgleich (engl. *„white balance“*), Beleichtungszeit (engl. *„exposure control“*) und weitere Funktionalität, können am Sensor mit integriert werden.

Um die Effizienz der Sensoren beider Technologien zu erhöhen wird heutzutage vielfach auf jedem Pixel der Sensoren eine Mikrolinse (engl. *„microlense“*) aufgesetzt. Diese bündelt das Licht und konzentriert es auf den lichtempfindlichen Teil der Pixelfläche wodurch sich die Apertur und damit insgesamt die Lichtempfindlichkeit erhöht. Durch Mikrolinsen kann eine Apertur von 30 % auf über 80 % erhöht werden. Bei der Verkleinerung der Pixelfläche wird es allerdings immer schwieriger geeignete Linsen herzustellen.

Farbaufnahmen:

Bei obiger Beschreibung der beiden Technologien für Bildsensoren wurde die Aufnahme von Farbbildern zunächst vernachlässigt. Ohne Zusatzmaßnahmen sind diese Sensoren nur in der Lage Grauwertbilder zu liefern (*„monochrom“*). Um Farbbilder aufnehmen zu können ist es notwendig, dass es eine getrennte Empfindlichkeit der Pixel für den Rot-, den Grün- und den Blauanteil des Lichtes nach dem RGB-Farbmodell aus den Abschnitten 2.3.2 und 2.3.3 gibt. Das kann auf verschiedene Art und Weise erreicht werden. Heute bis auf Spezialanwendungen in der Medizin eher ungebräuchlich ist die Variante drei Bilder hintereinander aufzunehmen, wobei je eine Aufnahme mit einem Farbfilter für Rot, eine mit einem für Grün und die dritte mit einem Filter für Blau gemacht wird. Die drei Farbfilter sind dabei auf einer drehbaren Scheibe montiert die mechanisch mittels Elektromotor weiterbewegt wird. Dies funktioniert nur bei unbewegten Objekten, da das Weiterdrehen der Filter, entsprechend Zeit in Anspruch nimmt. In heutigen Digitalkameras des oberen Preissegmentes ist für jeden Farbkanal ein eigener Chip vorhanden. Ein Prisma verteilt das Licht auf die drei Chips, wobei ein Chip mit einem Filter für Rot ein zweiter für Grün und der dritte für Blau ausgestattet ist. Dieser hohe Aufwand schlägt sich nicht nur im Platzbedarf sondern auch in den Kosten nieder. Aus diesem Grund erhält bei den Sensoren der niedrigen bis mittleren Preisklasse jeder Pixel ein eigenes Farbfilter und es werden immer Pixel mit roten, grünen und blauen Farbfiltern nebeneinander positioniert. Die Pixel werden dabei meist im so genannten Bayerpattern angeordnet.

Bayerpattern:

Abbildung 4.8 zeigt die Verteilung der rot-, grün- und blauempfindlichen Pixel auf einem Sensor mit Bayerpattern. Auffällig dabei ist, dass es doppelt so viele grün- wie rot- und blauempfindliche Pixel gibt. Damit wird einer Eigenschaft des menschlichen Auges Rechnung getragen, das im Wellenlängebereich des grünen Lichtes die größte Empfindlichkeit besitzt (siehe Abbildung 2.4).

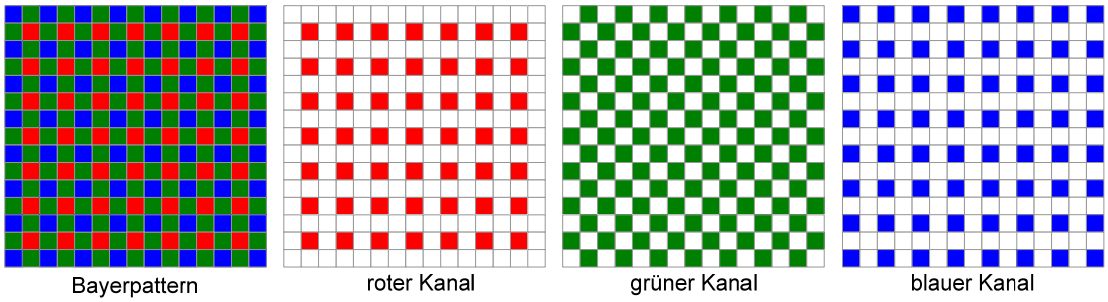

Abbildung 4.8: Bayerpattern

Diese Aufteilung der Farbpixel bringt den Nachteil mit sich, dass sich die effektive Auflösung des Sensors verringert. Ein weiterer Nachteil ist das häufigere Auftreten des so genannte *„Moire Effekts“* der bei sehr feinen Strukturen im Bild gehäuft vorkommen kann. Die Bildaufnahme mittels eines digitalen Bildsensors entspricht einer Abtastung des analogen Signals „Licht“. Es gilt das *„Shannen-Abtast-Theorem“* wonach ein Signal mit dem doppelten der höchsten im Signal vorkommenden Frequenz abgetastet werden muss, um das ursprüngliche Signal exakt rekonstruieren zu können. Da nun die Pixel für die einzelnen Farbkanäle weiter auseinander liegen, ist eine Unterabtastung leichter möglich. Sie kann aber auch bei Sensoren die nicht das Bayerpattern verwenden, besonders bei CMOS-Sensoren mit kleinen Aperturen, auftreten, da dort die lichtempfindlichen Flächen der Pixel ebenfalls weiter auseinander liegen (siehe oben).

Um aus dem Bayerpattern ein RGB-Farbbild zu gewinnen sind jeweils mindestens drei Pixel notwendig, um einen Bildpunkt im RGB-Bild zu berechnen, üblicherweise verwendet werden vier. Diesen Berechnungsvorgang nennt man *„Demosaicking“*. Es gibt eine Vielzahl von Demosaicking-Algorithmen, die sich vor allem in der Anzahl und der Verteilung der Pixel aus dem Bayerpattern unterscheiden, die zur Berechnung eines Pixels im RGB-Bild herangezogen werden.

Vergleich zwischen CMOS und CCD:

Durch die unterschiedlichen Technologien haben beide Sensortechniken ihre Stärken und Schwächen. Der große Nachteil der schlechteren Bildqualität bei CMOS-Sensoren hat sich in der letzten Zeit durch die stete Weiterentwicklung relativiert.

Vorteile von CMOS-Sensoren:

- geringe Baugröße
- geringer Leistungsverbrauch
- Digitalisierung und Zusatzfunktionalität am Chip. Dadurch ist keine zusätzliche Hardware nötig.
- Direkte Adressierung der Pixel. Dadurch können auch nur Teile des Bildes ausgelesen werden (*„Windowing“*).
- keine Bloomingeffekte

Nachteile von CMOS-Sensoren:

- kleine Apertur (siehe oben)
- größeres Offsetrauschen (*„fixed pattern noise“*)

Vorteile von CCD-Sensoren:

- Hohe Bildqualität da die Anzahl der Verstärker geringer ist und sich somit Bauteilschwankungen geringer auswirken.
- große Apertur, dadurch hohe Lichtempfindlichkeit
- Geringeres Bildrauschen: Da nur wenige Verstärker benötigt werden, können rauschärmere Schaltungen mit höherem Aufwand integriert werden.

Nachteile von CCD-Sensoren:

- größerer Leistungsverbrauch
- aufwendige Zusatzhardware erforderlich, dadurch auch mehr Platzbedarf
- Bloomingeffekte
- kein Windowing

Die obige Auflistung der Stärken und Schwächen beider Technologien soll im Folgenden ergänzt werden durch den Vergleich einiger für Bildsensoren wichtiger Parameter.

Reaktivität:

Unter der Reaktivität versteht man das Verhältnis der erzeugten Signalstärke am Pixel zur Menge an einfallenden Photonen. Durch die komplementären MOSFET-Verstärker, die

eine höhere Verstärkung bei geringerem Leistungsverbrauch erlauben, sind CMOS-Sensoren hier im Vorteil.

Dynamik:

Die Dynamik ist definiert als das Verhältnis der Sättigung eines Pixels zu dessen Lichtempfindlichkeit, angegeben in dB. Hier liegen die Vorteile eindeutig bei den CCD-Sensoren die teilweise bei gleichen Parametern doppelte Dynamik aufweisen.

Uniformität:

Werden zwei entfernte Pixel mit der gleichen Beleuchtungsstärke bestrahlt, so sollten sie denselben Grauwert liefern. Das ist in der Praxis selten der Fall. Ein Maß dafür ist die Uniformität. Durch Unterschiede in den Verstärkertransistoren, die bei CMOS-Sensoren für jeden Pixel vorhanden sind, liegt der CMOS-Sensor hier etwas im Nachteil.

Geschwindigkeit:

Durch die einfachere Steuerlogik, die Möglichkeit der direkten Adressierung der Pixel und die Onchip-Funktionalität, haben die CMOS-Sensoren in Punkto Geschwindigkeit eindeutige Vorteile.

Windowing:

Diese Fähigkeit der CMOS-Sensoren wurde an anderer Stelle schon erwähnt, soll hier aber noch einmal aufgelistet werden. Durch die Fähigkeit jeden Pixel direkt zu adressieren, ist es möglich nur Teile des Bildes auszulesen. Dieses Windowing genannte Verfahren kann für die Bildverarbeitung viele Vorteile mit sich bringen.

Versorgungsspannung:

CCD-Sensoren benötigen im Allgemeinen höhere verschiedene Versorgungsspannungen. CMOS kommen dagegen meist mit zwei Versorgungsspannungen aus die sich im heutigen üblichen Bereich für Digitalelektronik von 1.2 bis 3.3 Volt bewegen.

Tabelle 4.1: Vergleich zwischen CMOS- und CCD-Sensoren

Merkmal	CCD	CMOS
Füllfaktor / Apertur	hoch	mittel
Verstärkungsstörungen	keine	mittel
Systemrauschen	niedrig	mittel
Systemkomplexität	hoch	niedrig
Sensorkomplexität	hoch	niedrig

Kamerakomponenten	PCB + Zusatz-Chips + Linse	Chip + Linse
Reaktivität	mittel	etwas besser
Dynamik	hoch	mittel
Uniformität	hoch	niedrig bis mittel
Geschwindigkeit	mittel bis hoch	höher
Windowing	begrenzt möglich	immer möglich
Spannungsversorgung	hohe Spannungen, komplex	niedrige Spannung, einfach

Tabelle 4.1 fasst die Vor- und Nachteile der Sensoren noch einmal zusammen und bietet einen direkten Vergleich der beiden Technologien [GHR02]. Wenn man sich die wichtigsten Anforderungen an das Vision System aus Abschnitt 1.2 in Erinnerung ruft, so sind unter anderem vor allem geringer Leistungsverbrauch, geringe Abmessungen und hohe Geschwindigkeiten gefordert. Die Vorteile liegen hier eindeutig bei der CMOS-Technologie, weshalb die Wahl des bildgebenden Sensors für das Vision System auf einen CMOS-Sensor fällt.

Analog-Digital-Wandler:

Die Analog-Digital-Wandlung ist ein breit gefächertes Gebiet und soll hier nur der Vollständigkeit halber angeführt werden. Wie oben erwähnt, haben CMOS-Sensoren den ADC zur Analog-Digital-Wandlung bereits am Chip integriert, weshalb kein externer ADC mehr erforderlich ist. Bei den CCD-Sensoren wird die Analog-Digital-Wandlung auf einem externen Chip durchgeführt. Die Hersteller von CCD-Sensoren empfehlen im Allgemeinen Chips die mit ihren Sensoren kompatibel sind, oder stellen sie sogar selbst her. Meist bieten sie ein Referenzdesign für ein komplettes Kameramodul, das alle erforderlichen externen Bauteile enthält. Externe Analog-Digital-Wandlung ist eigentlich nur mehr erforderlich bei Verwendung von Kameramodulen die ein analoges Ausgangssignal in einem gängigen Videoformat wie PAL oder NTSC (siehe Abschnitt 2.5) liefern. Entsprechende Analog-Digital-Wandler, auch Videodecoder genannt, sind von verschiedenen Herstellern erhältlich, zum Beispiel der ADV7183 von Analog Devices [ADVD02].

4.3.3 Image-Sensoren und Kameramodule am Markt

Aufgrund der Eigenschaften von CMOS- und CCD-Sensoren fiel die Entscheidung für das Vision System auf einen CMOS-Sensor. Derzeit sind in der mittleren Preisklasse CMOS-Sensoren mit Auflösungen bis 5 Megapixel im Bayerpatternformat (siehe oben) erhältlich, wobei die Sensoren bei dieser Auflösung noch eine Framerate von ca. 7.5 fps liefern. Ein

Blick in Tabelle 1.1 zeigt eine erforderliche Framerate von 50 fps. Diese Sensoren liefern bei der geforderten Framerate immerhin noch eine Auflösung von QVGA (320 x 240 Pixel). Sensoren mit VGA-Auflösung (640 x 480 Pixel) bei 60fps sind in Entwicklung und werden demnächst verfügbar sein. Ein Nachteil der hoch auflösenden Chips ist die größere Chipfläche. Dadurch muss auch der Objektivdurchmesser größer gewählt werden, was sich wiederum negativ auf den Platzbedarf auswirkt. Da die erforderliche Framerate ohnehin eine maximale Auflösung von QVGA zulässt, ist eine maximale Sensor-Auflösung von VGA ausreichend, wodurch aufgrund der kleineren Chipfläche Sensorchips in sehr kleinen Bauformen erhältlich sind, ideal für die kleinen Abmessungen des Roboters. Besonders interessant für das Projekt sind Kameramodule wie sie in Handys oder PDAs (Pocketcomputer) Verwendung finden.

Derzeit gibt es unter anderen zwei Hersteller die für das vorliegende Projekt besonders interessante Sensoren anbieten, Omnivision und Micron. Je ein in Frage kommendes Produkt der beiden Anbieter soll nun vorgestellt werden.

Micron MT9V131

Der MT9V131 von Micron ist ein automatischer Farb-Kamerachip, der nur mehr eine Versorgungsspannung und ein Objektiv benötigt und bereits ein digitales Ausgangssignal liefert. Das Format des Ausgangssignals ist konfigurierbar. Unter anderem wird der in Abschnitt 2.5.3 beschriebene ITU-R BT.656 Standard unterstützt, sowie verschiedene RGB Modi mit 16 Bit pro Pixel (RGB565, 5 Bit für Rot, 6 Bit für Grün und 5 Bit für Blau) und mit 12 Bit pro Pixel (RGB 444), mit den für diese Modi benötigten Synchronisationssignalen HSYNC (LINE_VALID) und VSYNC (FRAME_VALID) zur horizontalen und vertikalen Synchronisation. Der Kamerachip ist ausgestattet mit einem digitalen Signalprozessor (DSP), Micron nennt ihn *„image flow processor“* (IFP), um Operationen wie Farb- oder Gammakorrektur und die Formatierung bzw. Aufbereitung des Ausgangssignals vornehmen zu können. Mittels eines Input Pins (STANDBY) kann der Chip in einen Low-Power Sleepmodus versetzt werden, in dem die Stromaufnahme bei ca. 5 µA liegt. Ein Pin bietet die Möglichkeit, die Ausgangssignale in einen hochohmigen Zustand zu versetzten (engl. *„high impedance“*). Der Chip bietet zusätzlich ein Ausgangspin, mit dem ein Blitzlicht gesteuert werden kann. Alle Einstellungen können durch das Beschreiben chip-interner Register über einen I^2C- kompatiblen (siehe [WIC02]) Steuerbus vorgenommen werden.

Die maximale Auflösung des Chips liegt bei 640 x 480 (VGA) bei 30 fps. Bei QVGA-Auflösung bietet er eine Framerate von 90 fps.

Die Daten wurden entnommen aus [MCR05]. Eine genauere Auflistung der Kenndaten des MT9V131 finden sich in der Tabelle 4.2.

Omnivision OV7660FSG

Omnivision bietet mit dem OV7660FSG nicht nur einen Kamerachip sondern ein komplettes Kameramodul basierend auf den Chip OV7660 inklusive Fixfokus-Linse an. Damit entfällt

die Suche nach einem geeigneten Objektiv, und die nicht immer ganz unkritische Montage desselben. Es ist dies ein Modul wie es auch in Handys oder PDAs verwendet wird. Das Kameramodul OV7660 bietet ähnliche Funktionen wie der Kamerachip von Micron. Ein Pin für einen Sleep bzw. Powerdown Modus ist vorhanden (PWDN). Dem OV7660 fehlt aber der Ausgang zur Blitzlichtsteuerung sowie der Enable-Pin für die Ausgangsleitungen. Das Ausgangsformat ist auch beim Omnivision Chip konfigurierbar, wobei dieser es auch erlaubt die Daten im Rohformat (engl. *„raw“*), das heißt direkt das Bayerpattern, auslesen zu können. Die Konfiguration erfolgt hier wieder über interne Register wobei als Steuerbus ein proprietäres Interface von Omnivision genannt SCCB (*„serial camera control bus“*) verwendet wird, das dem I^2C Standard aber sehr ähnlich ist. Ein kleiner Nachteil ist die Notwendigkeit von drei Versorgungsspannungen. Wird die analoge und die Versorgungsspannung für die I/O Pins identisch gewählt, was natürlich nur bei entsprechender Verträglichkeit mit der angeschlossenen Elektronik möglich ist, so reduzieren sie sich auf zwei [OVT04].

Die unterstützten Auflösungen sind identisch, wobei Omnivision bei QVGA-Auflösung nur 60 fps liefert. Genauso wie der Micron-Chip unterstützt auch das OV7660FSG Modul das für CMOS-Chips typische Windowing (siehe oben). Ein Flicker-Filter, gemeint ist damit das Herausfiltern von störendem 50 oder 60 Hz Flimmern von Beleuchtungen, ist ebenfalls bei beiden vorhanden.

Tabelle 4.2: Kenndaten des MT9V131 und des OV7660FSG im Vergleich

Parameter	**MT9V131**	**OV7660FSG**	**Einheit**
Versorgungspannungen	2,8	1,8 2,5 und 3,3	V
maximale Stromaufnahme	45[1)]	25[1)]	mA
Leistungsaufnahme	80	40	mW
Stromaufnahme im Sleepmodus	10	20	µA
Pixelgröße	5,6 x 5.6	4,2 x 4,2	µm
Diagonale des Sichtfensters	4,48	3,44	mm
Optisches Format	1/4	1/5	inch
Anzahl aktiver Pixel	640 x 480	664 x 492	Pixel
maximale Framerate	30 bei VGA, 90 bei QVGA	30 bei VGA, 60 bei QVGA	fps

[1)] Bei darüber stehenden Versorgungsspannungen.

Datenformate	ITU-R BT.656, RGB565, RGB555, RGB444	ITU-R BT.656, RGB565, RGB555, RAW	
Dynamik	60	72	dB
Signal- zu Rauschverhältnis (SNR)	45	48	dB
Steuerbus	I^2C	SCCB	
Abmessungen (L x B x H)	7 x 7 x 1,12 [1)]	6,5 x 6,5 x 4,94 [2)]	mm
Objektiv	extra	eingebaut	
Farbe / monochrom	Farbe	Farbe	
Flicker-Filter	50 / 60	50 / 60	Hz
Windowing	unterstützt	unterstützt	
Betriebstemperatur	-20 bis +60	-20 bis +80	°C

Die Tabelle 4.2 fasst die wichtigsten Kenndaten beider Chips zusammen und stellt sie einander gegenüber. Wenn nicht anders angegeben, wurden für die Tabelle immer die Durchschnittswerte aus den Datenblättern verwendet [MCR05, OVT04].

Die Auflistung zeigt, dass die Kenndaten beider Kamerachips ziemlich ausgeglichen sind. Der Micron Chip hat Vorteile bei der Framerate im QVGA-Modus aber auch das OV7660FSG-Modul erfüllt mit 60 fps die Anforderungen. Leichte Vorteile hat der OV7660 bei der Dynamik, deutlich besticht er aber durch seinen Leistungsverbrauch der deutlich unter dem des Micron-Chips liegt. Der große Vorteil liegt aber in der Tatsache, dass im OV7660FSG das Objektiv bereits eingebaut ist und in Verbindung mit dem geringeren Leistungsverbrauch und den Abmessungen des kompletten Moduls inklusive Objektiv, ist es für die vorliegende Aufgabenstellung die bessere Alternative. Abbildung 4.9 zeigt eine schematische Zeichnung des Moduls inklusive dem Verbindungskabel.

[1)] Abmessungen des Chips im iSCP-Package [MCR05], ohne Objektiv und Objektivhalterung.

[2)] Gesamtes Modul inklusive Objektiv, jedoch ohne Verbindungskabel

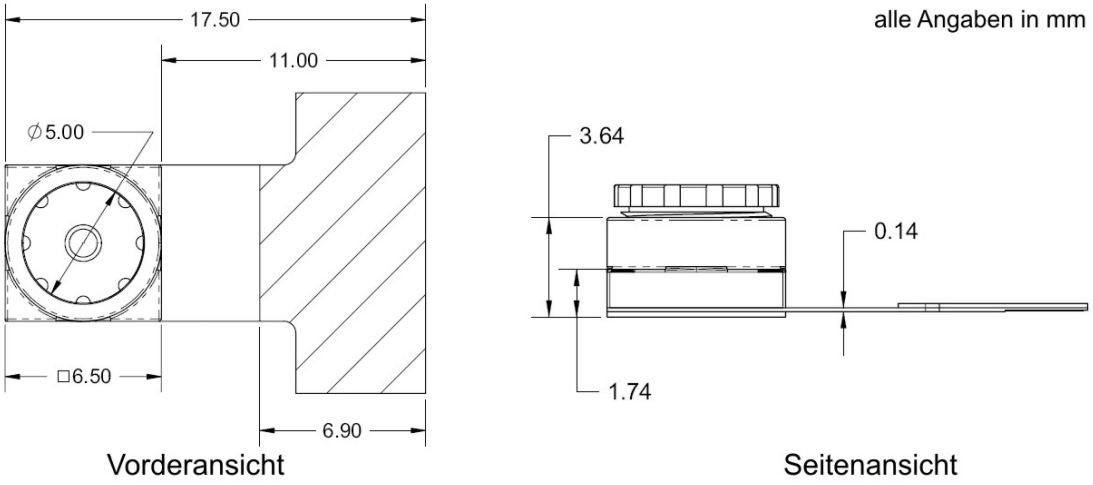

Abbildung 4.9: Schematische Zeichnung des OV7660FSG

4.3.4 Monokamera gegen Stereokamera

Immer wenn man über die Entwicklung eines Bildverarbeitungssystems diskutiert, ist die Frage ob eine, zwei oder gar mehrere Kameras benötigt werden, zu überlegen. Das menschliche „Bildverarbeitungssystem“ verwendet zwei „Kameras“ um anhand der Parallaxe Tiefeninformationen gewinnen zu können. Damit ist die Entfernung zu jedem beliebigen Objekt abschätzbar. Mit nur einem Auge ist nur eine zweidimensionale Sicht der Umwelt möglich und Entfernungen können nur aufgrund unserer Erfahrung abgeschätzt werden. Auch im Bereich der digitalen Bildverarbeitung verhält es sich ähnlich. Durch die Verwendung zweier Kameras in einem gewissen Abstand voneinander die dieselbe Szene beobachten, können Informationen über die Entfernung zu Objekten aus dem Unterschied der beiden Bilder ermittelt werden. Dies erfordert aber aufwendige Berechnungen, die sich in der geforderten Framerate sicher nicht durchführen lassen. Sind die Form und die Abmessungen der Objekte von Interesse jedoch bekannt, so können Entfernungen und Positionen der Objekte auch nur mit einer Kamera bestimmt werden. Im vorliegenden Fall, bei dem es sich bei den Objekten von Interesse um einen Golfball und die eingefärbten Tore, also einfärbige Rechtecke definierter Größe handelt, ist ein Monokamerasystem vollkommen ausreichend um die Position dieser Objekte relativ zum Roboter berechnen zu können.

4.3.5 Kameramontage

Ein wichtiger Faktor für die Bilderkennung sind natürlich die Parameter der verwendeten Kamera. Ebenso wichtig ist aber die Montageposition und damit der Blickwinkel auf die beobachtete Szene und der damit verbundene Einfall des Lichtes. Die Mirosot Regeln der Fira (siehe Abschnitt 1.1.1) beschränken die Höhe des Roboters auf 7.5 cm. Damit sind die Möglichkeiten zur Montage bereits eingeschränkt. Der Durchmesser des Balles beträgt ca. 43

mm (siehe Abschnitt 1.1.1). Da die Hauptaufgabe dieses Projekts in einer möglichst genauen Entfernungsbestimmung des Balles liegt, ist es sinnvoll die Kamera in Höhe der Balloberkante zu montieren. Dies bietet mehrere Vorteile:

- Die Oberkante des Balles liegt immer, unabhängig von dessen Entfernung, in einem schmalen horizontalen Band in der Bildmitte. Kanten darüber und darunter könnten im Sinne einer Optimierung vernachlässigt werden, wie Abbildung 4.10 in einer schematischen Zeichnung zeigt. Je größer die Verzerrungen der Linse, die Bodenunebenheiten oder je ungenauer die Ausrichtung der Kamera, desto größer muss dieser Detektionsbereich gewählt werden.
- Auch bei direktem Kontakt des Roboters mit dem Ball „sieht“ die Kamera immer noch die Oberkante des Balles und macht so eine Berechnung möglich.
- Bei einer üblichen Beleuchtung von oben wirken sich Reflexionen vom Boden, da die Kameraachse parallel dazu verläuft, nicht so störend aus.
- Alle relevanten Objekte liegen immer im unteren Bereich des Bildes. Das obere Drittel braucht daher nicht berücksichtigt zu werden, was die Ausführungszeit der Bilderkennung erheblich reduziert.

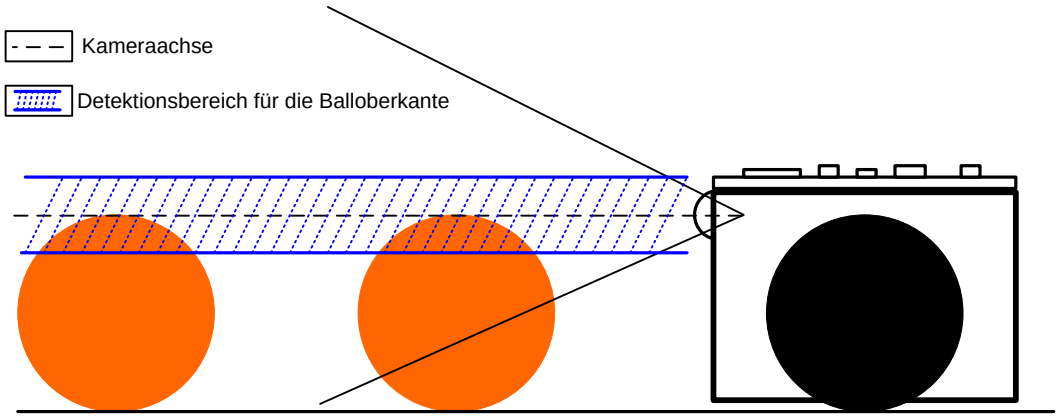

Abbildung 4.10: Position und Blickwinkel der Kamera

Die Art der Montage wie sie Abbildung 4.10 zeigt, birgt noch einen weiteren Vorteil. Die Kamera „liegt“ sehr nahe an der Platine mit der restlichen Elektronik. Dadurch ergeben sich kurze Leitungslängen die negative Auswirkungen auf die Signalqualität vermeiden helfen. Aufgrund der doch einigermaßen hohen Frequenzen bei der digitalen Videoübertragung mit höheren Bildwiederholraten ist dies durchaus ein Thema.

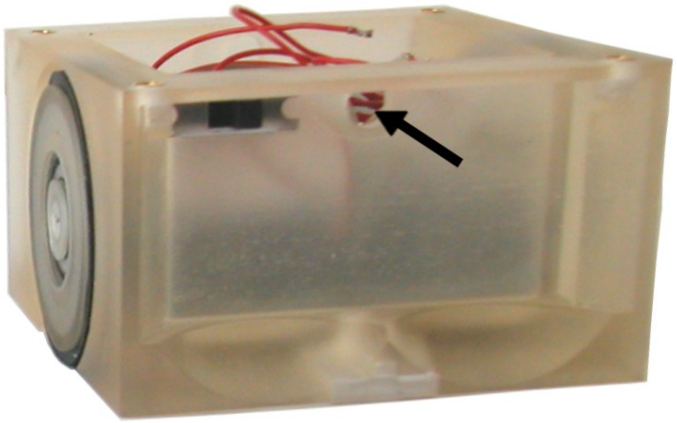

Abbildung 4.11: Montage des Kameramoduls

Abbildung 4.11 zeigt die Position der Kamera in der Robotermechanik. Das Loch in das die Kamera gesteckt wird, erlaubt noch ein Verdrehen der Kamera um sie auszurichten. Der Durchmesser ist aber klein genug um das Modul darin fest zu halten.

Die Montage beeinflusst auch das Bildformat. Die Kamera steht bedingt durch die Einbauvariante am Kopf. Dies ist aber die vom Hersteller vorgezogene Einbaurichtung. Das Bild erscheint nach dem Auslesen im Speicher dann mit der richtigen Ausrichtung, das heißt, das linke, obere Eckpixel liegt an der niederwertigsten Adresse.

4.4 Prozessor

In diesem Abschnitt soll nun die Auswahl des Prozessors, auf dem die Bildverarbeitungs-software letztendlich ausgeführt wird, diskutiert werden. Das Angebot an Prozessoren ist ähnlich breit gefächert, wie es Algorithmen zur Bildverarbeitung gibt. Ausschlaggebend bei der Auswahl ist wiederum der Platzbedarf und der Leistungsverbrauch in Verbindung mit der Rechenleistung. Hinzu kommt um den Hardwareaufwand möglichst gering zu halten, da Platz und Leistung begrenzt sind, sollte der Prozessor ein geeignetes Interface haben, das einen direkten Anschluss des Kamermoduls ohne zusätzliche Logik erlaubt. Im Bereich der digitalen Signalprozessoren (DSP) haben die meisten Hersteller mindestens ein Modell mit entsprechender Schnittstelle im Programm. Da es sich bei der Bildverarbeitung um eine Aufgabe im Bereich der Signalverarbeitung handelt, statten viele Hersteller ihre DSPs mit entsprechenden Befehlssatzerweiterungen für die Videoverarbeitung aus. Von ihrer Rechenleistung sind einige für die Videokomprimierung ausgelegt. Ein sehr rechenintensiver Prozess dessen Algorithmen durchaus Ähnlichkeit haben mit den in Kapitel 3 beschriebenen. Überhaupt ist die Architektur von DSPs wie der Name schon sagt, für die Signalverarbeitung optimiert. Um im Bereich der batteriebetriebenen Geräte wie Handys, PDAs oder Digitalkameras Einsatz zu finden, werden sie außerdem im Hinblick auf den

Leistungsverbrauch optimiert. Aus diesem Grund scheinen DSPs bestens geeignet für einen Einsatz im Vision System.

Kurz diskutiert werden muss im Zusammenhang mit Bildverarbeitung die Verwendung eines FPGAs (*„field programmable gate array“*), da sie in vielen Systemen Einsatz finden und große Vorteile in der Ausführungszeit der Bildverarbeitungs-Algorithmen bieten nicht zuletzt aufgrund der Möglichkeit massiver Parallelisierung. Für die vorliegende Aufgabenstellung sprechen aber einige Punkte gegen den Einsatz eines FPGAs:

- Hinsichtlich des Leistungsverbrauchs gibt es kaum Vorteile eher Nachteile. Moderne DSPs bieten bereits ausgeklügelte Power Management Funktionen um den Leistungsverbrauch zusätzlich zu senken.
- FPGAs ausreichender Größe sind meist nur in größeren Gehäusen erhältlich.
- Für das Projekt sind Kommunikationsschnittstellen zu den anderen Systemen des Roboters erforderlich die in einem FPGA erst realisiert werden müssten.
- Aufwendigere Programmierung
- Weniger Flexibilität in der Software

4.4.1 Fixkomma gegen Fließkomma

Bei der Auswahl eines geeigneten DSPs spielt die Frage ob eine FPU (*„floating point unit“*) für eine Hardwareunterstützung zur Fließkommaberechnung von Nöten ist oder nicht eine wichtige Rolle. Bei einer Software-Emulation von Fließkommaoperationen liegen die Ausführungszeiten um Größenordnungen bis zu 100 über jenen mit Hardwareunterstützung durch eine FPU. Die Frage ist also, sind Fließkommaoperationen in größerem Ausmaß notwendig. Wenn man sich die Algorithmen aus Kapitel 3 in Erinnerung ruft, so stellt man fest, dass der Wertebereich sowohl für die Eingangswerte als auch für die Ergebnisse meist im Bereich zwischen 0 und 255 liegt. Es gibt natürlich Operationen die sich innerhalb der Menge der reellen Zahlen bewegen, wie etwa einige Gleichungen zur Farbraumkonvertierung (siehe Abschnitt 2.3) doch lassen sich diese ohne großen Aufwand auch in der Menge der ganzen Zahlen berechnen oder mittels Fixkommaoperationen, da die erforderliche Genauigkeit und der Wertebereich im vorhinein bekannt sind. Für die Bildverarbeitung ist eine FPU also nicht notwendig. Das Vorhandensein einer FPU schlägt sich vor allem im Preis und in einem höheren Leistungsverbrauch nieder.

4.4.2 DSPs am Markt

Es gibt einige Hersteller die DSPs mit den erwähnten Anforderungen im Programm haben. Die Kenndaten je eines Prozessors von Texas Instruments (TSM320DM642) und von Analog Devices (Blackfin ADSP-BF533) die den Vorgaben entsprechen, werden in der Tabelle 4.3

einander gegenübergestellt. Die Werte wurden aus dem Datenblatt des jeweiligen DSPs entnommen [TSM06, ADS06].

Tabelle 4.3: Eckdaten zweier Signalprozessoren im Vergleich

Parameter	**TSM320DM642**	**ADSP-BF533**	**Einheit**
I/O Versorgungsspannung	3,3	2,5 oder 3,3	V
Coreversorgungsspannung	1,2 – 1,4	0,7 – 1,27	V
maximale Taktfrequenz[1)]	600	600	Mhz
Leistungsaufnahme[2)]	1940	350	mW
Videoschnittstellen	3	1	
interner Speicher	288	148	kByte
Chipgröße (B x H)	27 x 27	9 x 9	mm
Pins / Gehäuse	160 / Mini-BGA	548 / BGA	
SDRAM Interface	vorhanden	vorhanden	

Der Blackfin ADSP-BF533 hat zwei ALUs (*„arithmetic logic unit“*) mit je 40 Bit Breite. Das Register- und Instruction-Model entspricht einer RISC (*„reduced instruction set computer“*) Architektur. Zwei 16 bit MAC-Operationen (*„multiply accumulate“*) können parallel ausgeführt werden. Selbiges gilt auch für den DSP von Texas Instruments, wobei der Befehlssatz aber keiner RISC-Architektur entspricht. Auch in Punkto Kommunikations-Schnittstellen sind sich die beiden Prozessoren ebenbürtig. Beide verfügen außerdem über eine leistungsstarke DMA-Einheit (*„direct memory access“*) mit mehreren Kanälen. Alles in allem spricht aber die Einsparung beim Platzbedarf und der Leistungsaufnahme für den Blackfin (siehe Tabelle 4.3).

Blackfin ADSP-BF533

Die Architektur des Blackfin BF533 wurde im obigen Abschnitt schon kurz angesprochen. An dieser Stelle noch ein paar Anmerkungen hinsichtlich seiner DMA-Einheit und der Kommunikations-Schnittstellen (Peripherie).

[1)] Abhängig vom Typ. Typen bis 750Mhz sind von beiden erhältlich.

[2)] Bei durchschnittlicher Prozessorauslastung, maximaler Versorgungsspannung und Taktfrequenz.

DMA:

DMA steht für *„direct memory access“* und bezeichnet eine Einheit die es erlaubt Daten ohne Eingriff der CPU (*„central processing unit“*) zu transferieren. Ein eigener Kontroller steuert dabei die Datentransfers. Der Blackfin BF533 verfügt über zwei unabhängige DMA-Kontroller die jeweils zwei Kanäle verwalten können für Transfers von Speicher zu Speicher. Zwölf Kanäle stehen für den Transfer zwischen Peripherie und Speicher zur Verfügung.

Adressraum:

Die Prozessoren der Blackfin-Familie verfügen über einen linearen Adressraum von 4 GByte, die aufgeteilt werden zwischen SDRAM, vier asynchronen Speicherbänken, internem statische RAM (SRAM) und die „memory mapped“ Register (MMR). Abbildung 4.12 zeigt die Adressraumaufteilung des BF533.

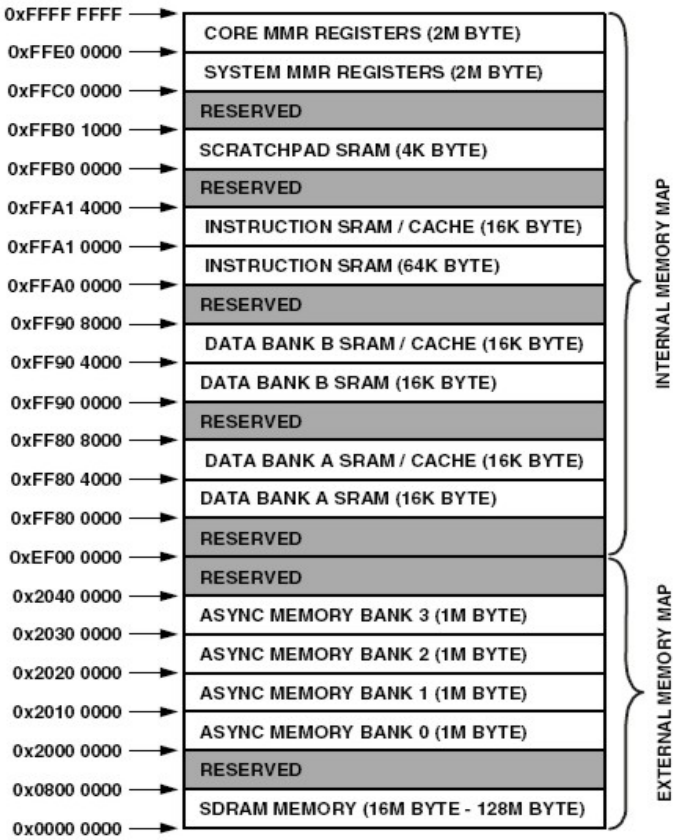

Abbildung 4.12: Adressraum des Blackfin BF533

Peripherie:

Es soll an dieser Stelle nur ein Überblick über die Peripherie des BF533 gegeben werden. Falls verwendet, wird die jeweilige Schnittstelle im Abschnitt 5.5 näher beschrieben.

- 1 x PPI: *„Parallel peripheral interface“,* 8 bis 16 Bit breite parallele Datenschnittstelle mit Synchronisationstakt und bis zu drei Steuerleitungen. Geignet für Videointerfaces, Kameras, Displays und ähnliches.
- 1 x SPI: *„Serial peripheral interface“*, synchrone serielle Schnittstelle.
- 2 x SPORT: Synchrone serielle Schnittstelle, full duplex, getrennter Takt für Senden und Empfangen, geeignet für digitale Audioübertragungen.
- 1 x UART: *„Universal asynchronous receiver transmitter“*, asynchrone serielle Schnittstelle.
- JTAG: Für Programmierung und Debug-Operationen
- SDRAM: Interface für den Anschluss von bis zu 128 MByte an SDRAM, maximale Taktrate 133 Mhz.
- Asynchrones Speicherinterface: Für den Anschluss von Flash, statischem Speicher oder memory mapped devices. Vier Bänke zu je einem MByte sind verfügbar.

Weitere Funktionen:

- RTC: *„Real time clock“*, Echtzeituhr
- WDT: *„Watchdog timer“*, Bei Überlauf erzwingt dieser Timer einen Neustart des Prozessors.
- Timer: Drei Timer mit Unterstützung für PWM-Signale.
- Powermanagment: Drei Prozessormodi mit unterschiedlicher Leistungsaufnahme, interne PLL (*„phase locked loop“*), flexible Einstellmöglichkeiten des Prozessortaktes und der Coreversorgungsspannung über den internen Spannungsregler.

Noch ein Hinweis: Aufgrund der Architektur, RISC-Prozessor in Verbindung mit DSP-Funktionalität spricht Analog Devices bei der Blackfin-Serie nicht nur von DSPs, sondern von *„Embedded Processors“*.

4.5 Speicher

Der gesamte Speicher des Blackfin ist über den linearen 4 GByte großen Adressraum ansprechbar. Im unteren Bereich, von Adresse 0 an, befindet sich das SDRAM. Ab Adresse 0x2000 0000 beginnen die asynchronen Bänken. (siehe Abbildung 4.12). Dieser Adresse kommt dabei eine besondere Bedeutung zu, da es die Einsprungadresse für den Bootvorgang ist. An dieser Stelle muss somit der nichtflüchtige Speicher, der das Programm enthält, beginnen.

4.5.1 RAM

Intern verfügt der BF533 nur über 148kByte an statischem RAM. Die Zugriffe darauf erfolgen aber mit dem Prozessortakt, womit es der Speicher mit den geringsten Latenzzeiten ist. Für die Videoverarbeitung ist es aber notwendig dass einige Frames im Speicher abgelegt werden können. Zu diesem Zweck kann der Blackfin mit bis zu 128 MByte externem SDRAM bestückt werden. Da es sich um einen dynamischen Speicher handelt, sind regelmäßige *„Refreshzyklen“* notwendig. Diese können vom SDRAM-Controller automatisiert durchgeführt werden. Der benötigte Speicher hängt natürlich von der verwendeten Auflösung ab. Die Kamera liefert eine maximale Auflösung von 640 x 480 Pixel. Im RGB-Farbraum mit einer Kodierung von einem Byte pro Kanal ergibt sich damit ein Speicherbedarf von 921600 Byte also knapp einem MByte. Sollten drei oder vier Frames abgespeichert werden bedarf es bereits an die 4 MByte. Der Blackfin muss mindestens mit 16 MByte bestückt werden was schon über den Mindestanforderungen liegt. Da SDRAM-Bausteine in kleinem Gehäuse (BGA) mit 32 MByte erhältlich sind, werden diese verwendet.

4.5.2 Flash

Als nichtflüchtiger Speicher wird ein Flash verwendet. Die ersten zwei asynchronen Bänken werden davon belegt, womit der Flashspeicher wie gefordert bei 0x2000 0000 beginnt und eine Größe von 2 MByte hat. Die gesamte Bildverarbeitungssoftware darf somit nicht größer als 2 MByte werden.

4.6 Tinyphoon

Das Vision System ist nur ein Teil des gesamten Roboters der den Namen Tinyphoon erhalten hat. Die Abbildung 4.13 zeigt ein Foto mit montierter Vision/Motion Platine. Das Aussehen ändert sich dabei ständig aufgrund der steten Weiterentwicklung und des Einbaus neuer Funktionen. In [STM06] findet sich ein guter Überblick über die Entwicklung und alle Funktionen und Einheiten des Tinyphoon.

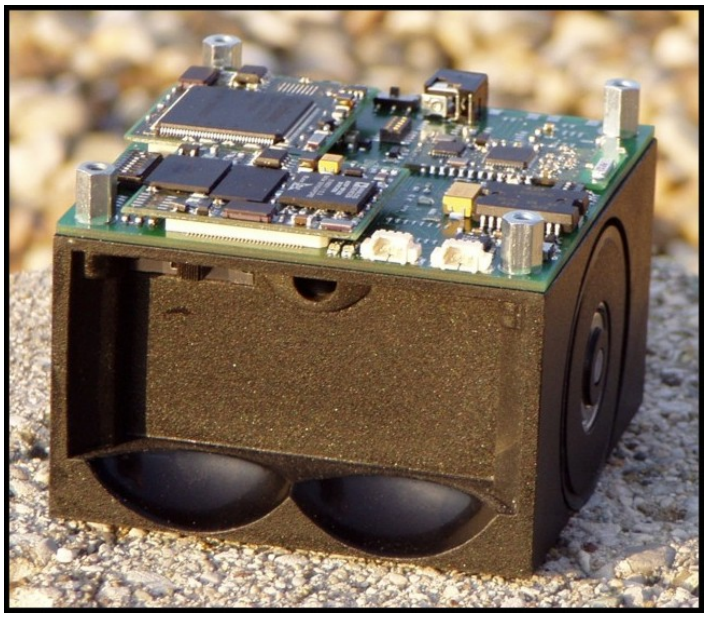

Abbildung 4.13: Tinyphoon

Kapitel 5 Realisierung und Ergebnisse

Auf Basis der in Kapitel 4 vorgestellten Architektur wird nun die konkrete Realsierung des Systems zur Objekterkennung gezeigt. Auch die notwendigen Vorarbeiten sollen dabei kurz angesprochen werden, speziell Treiberfunktionalitäten und die Werkzeuge die zur Entwicklung notwendig waren. Problembereiche werden ebenfalls aufgezeigt um zukünftigen Entwicklungen in diesem Bereich zu helfen bereits vorab Fehler zu vermeiden.

5.1 Entwicklungstools

Für die Softwareentwicklung und die Programmierung wurden ausschließlich Tools von Analog Devices (AD) verwendet. Ich möchte Analog Devices an dieser Stelle danken für die unentgeltliche Zurverfügungstellung der Entwicklungstools. Für die Programmierung des Blackfin ist ein JTAG-Device in Verbindung mit der Entwicklungssoftware erforderlich. Mittlerweile gibt es zwei Anbieter dafür.

5.1.1 Altium Designer

Der „Altium Designer“ ist eine CAD-Software für die Schaltplan und Platinenlayout Entwicklung der Firma „Altium“. Die Software ging hervor aus dem bereits sehr beliebten Tool Protel. Für das vorliegende Projekt wurde der „Altium Designer“ für die Erstellung des Schaltplanes und des Layoutes der Platine (engl. *„printed circiut board“*, PCB) verwendet.

5.1.2 VDSP++

Die von Analog Devices zur Verfügung gestellte Software für die Softwareentwicklung am Blackfin, aber auch anderer DSPs von AD nennt sich VDSP++, eine moderne integrierte Entwicklungsumgebung (engl *„integrated development environment“*, IDE). Neben der Programmcodeentwicklung in C, C++ und Assembler erlaubt die IDE auch das Laden, Starten, Stoppen sowie Debuggen von Programmen über einen JTAG (siehe Abschnitt 5.1.3). Interessant und sehr hilfreich sind Zusatzfunktionen wie der *„image viewer“* der es erlaubt

einen beliebigen Speicherbereich des Blackfin als Bild anzuzeigen. Mehrere Farbräume werden dabei unterstützt. Der Speicher kann aber auch in einer Datei abgespeichert (engl. *„dump“*) werden, oder als zwei- oder dreiwertige Funktion interpretiert werden. Selbstverständlich ist auch das Visualisieren und Auswerten von Variablen und Ausdrücken möglich. Eine schrittweise Ausführung von C- oder Assembleranweisungen (engl. *„step“*) ebenfalls. Für Entwicklungsumgebungen im embedded Bereich typisch, ist die Möglichkeit zur Anzeige der Inhalte von Prozessorregistern. VDSP++ gibt es derzeit nur für das Betriebssystem Microsoft Windows. Eine genaue Beschreibung der VDSP++ IDE findet sich in [VDU06] oder auf [WAD06].

5.1.3 JTAG

Für eine komfortable Softwareentwicklung unerlässlich ist der JTAG. Die Abkürzung steht für *„joint test action group“* und ist ein IEEE-Standard für das Debugging von Hardware. Durch die Möglichkeit direkt in die Hardware einzugreifen hat sich der JTAG in der Praxis über den ursprünglichen Standard hinausentwickelt und erlaubt nun auch das Debugging von Software. Dabei ging die Uniformität verloren und die JTAG-Interfaces der Prozessoren unterschiedlicher Hersteller sind nicht immer kompatibel, wohl auch aus wirtschaftlichen Gründen.

Alle Prozessoren der Blackfin Reihe besitzen eine JTAG-Schnittstelle. Analog Devices bietet seine JTAG-Devices für den Blackfin in zwei Varianten an: als USB-Version oder als PCI Einschubkarte, wobei sich gezeigt hat, dass die PCI-Version weniger zu Abstürzen neigt.

Derzeit arbeiten die JTAG-Devices von AD nur mit der Entwicklungsumgebung VDSP++ zusammen.

Icebear:

Die zum Zeitpunkt des entstehens dieser Arbeit einzig verfügbare Alternative zu den JTAG-Geräten von Analog Devices war der „Icebear“. Dieser wird von einer kleinen schweizer Firma produziert und arbeitet nur unter dem Betriebssystem Linux. Somit ist er mit dem VDSP++ nicht verwendbar. Weitere Informationen zum Icebear finden sich auf [WSC03].

Die gesamte Entwicklung der Software im Rahmen dieses Projekts, erfolgte in der Entwicklungsumbgebung VDSP++ Version 3.1 bis 4.0 in Verbindung mit dem HPPCI JTAG von Analog Devices.

5.2 Schaltplan und Platinenlayout

Im Folgenden soll nun kurz der Schaltplan und das PCB-Layout diskutiert werden. Es wird dabei nur auf den Teil des Vision Systems eingegangen, ohne auf die restliche Schaltung auf derselben Platine, die zur Motorsteuerung dient, einzugehen. Diese ist in [STM06]

dokumentiert. Wichtige Vorarbeiten für die Entwicklung des Schaltplans wurden dabei von Dietmar Bruckner im Rahmen seiner Diplomarbeit geleistet [DBD04]. Im Gegensatz zu der darin beschriebenen Plattform sollen der Blackfin, das SDRAM und das Flash gemeinsam auf einer Platine zu einem eigenen Prozessormodul (engl. „*coremodule*“) zusammengefasst werden. Die restlichen Baugruppen und der Stecker für den Anschluss des Kameramoduls werden auf einer getrennten Platine untergebracht, auf die dieses Prozessormodul mittels Platinenverbindungsstecker aufgesteckt werden kann. Dies hat den Vorteil, dass mit verschiedenen Prozessoren getestet werden kann, ohne jedes Mal die gesamte Platine neu entwerfen und herstellen zu müssen. So können auch zukünftige neue Blackfinderivate mit einer höheren Prozessorleistung auf dieser Basisplattform getestet werden.

5.2.1 Schaltplan

Abbildung 5.1 zeigt ein Blockschaltbild der gesamten Schaltung inklusive der Kamera und den Schnittstellen zu den anderen Einheiten des Roboters. Jener Teil der auf dem Prozessormodul, CM-BF533 genannt, untergebracht wurde, ist fett umrandet. Die restlichen Teile befinden sich am Basisboard.

Der Anschluss von SDRAM und Flashspeicher entsprechen großteils dem Referenzdesign von Analog Devices. Beim Anschluss der Kamera wurden auch die Signaleitungen für horizontale und vertikale Synchronisation verbunden um ein Höchstmass an Flexibilität zu garantieren, auch wenn bei Verwendung des ITU-R BT.656 Modus diese nicht erforderlich wären.

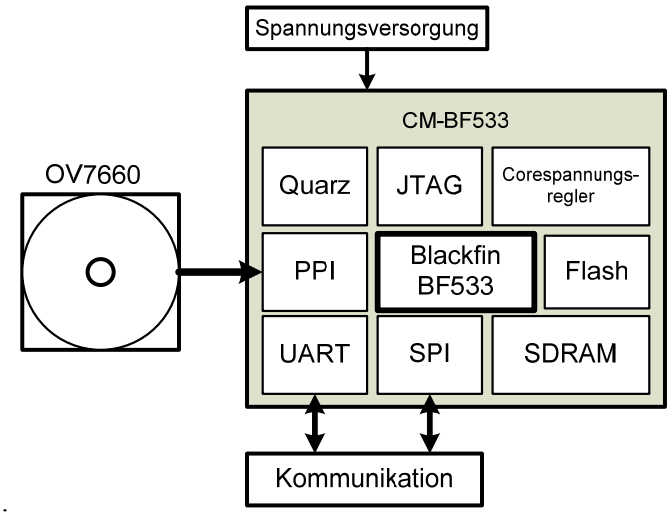

Abbildung 5.1: Blockschaltbild

5.2.2 PCB-Layout

Eine Abbildung des Layouts für das Prozessormodul und das Basisboard finden sich im Anhang. Nachdem das Board mit der Kamera zur Ballerkennung direkt auf die Mechanik aufgesetzt wird, kann die Kamera mit dieser Platine verbunden werden. Besonders zu achten ist dabei auf die Position des Steckers. Er muss so platziert werden, dass die Kamera in die dafür vorgesehene Öffnung in der Front der Mechanik (siehe Abbildung 4.11) eingeführt werden kann, ohne das Anschlusskabel zu knicken.

5.3 Betriebssystem

Immer wenn man im Bereich der Softwareentwicklung für „Embedded Systems“ arbeitet, kommt man an einem Punkt wo sich die Frage: Betriebsystem ja oder nein und falls ja welches, stellt. Zuerst gilt es zu definieren, was in diesem Zusammenhang mit Betriebsystem gemeint ist. Es gibt keine allgemein gültige Definition und dies soll auch nicht der Rahmen sein um eine vorzuschlagen. Die deutsche Ausgabe der online Enzyklopädie Wikipedia meint dazu:

> „Ein Betriebssystem ist die Software, die die Verwendung (den Betrieb) eines Computers ermöglicht. Es verwaltet Betriebsmittel wie Speicher, Ein- Ausgabegeräte und steuert die Ausführung von Programmen.“ [WWP04]

Es ist natürlich ein Unsinn, dass ein Betriebssystem zwingend notwendig ist für den Betrieb eines Computers oder Prozessors. Ebenfalls ist die Aussage ein Betriebssystem ist erforderlich für die Ausführung von Programmen in dieser pauschalen Form nicht korrekt. Dies gilt natürlich nur für Programme die für das entsprechende Betriebssystem kompiliert wurden. Obiges Zitat soll eigentlich nur zeigen, wie vielfältig die Meinungen über dieses Thema sind, und wie weit sie auseinander liegen. Uneinig ist man sich in der Fachliteratur auch über das was zu einem Betriebssystem gehört oder nicht. Einigkeit herrscht jedoch darüber, dass die Verwaltung von Speicher und Ein- Ausgabegeräten eine zentrale Aufgabe von Betriebssystemen ist. In dieser Arbeit wird unter dem Begriff Betriebssystem eine Software verstanden, die den Zugriff auf den Speicher, die Ein- und Ausgabegeräte sowie weiterer Kommunikationsschnittstellen verwaltet. Das Betriebssystem erlaubt das Starten und Ausführen von spezieller Anwendungssoftware und stellt ihr Funktionen zur Prozesssynchronisation zur Verfügung. Mit Ein- und Ausgabegeräte sind hier vor allem die Geräte für die Mensch-Maschine-Kommunikation wie Tastatur und Bildschirm gemeint sowie das Dateisystem. Da dies für das Vision System nicht benötigt wird und um die hohen Anforderungen zu erfüllen eine sehr hardwarespezifische Programmierung erforderlich ist, die sich in Betriebssystemen nur mit größerem Aufwand realisieren lässt, bringt die Verwendung eines Betriebssystems einen Mehraufwand mit sich, wobei ein Vorteil nicht gegeben ist. Auf die Verwendung eines Betriebssystems wird daher verzichtet.

5.4 Software-Bibliotheken zur Bilderkennung

Im Rahmen dieser Arbeit wurden auch Software-Bibliotheken zur Bildverarbeitung auf ihre Verwendbarkeit für das vorliegende Projekt untersucht. Dies ist zum einen „CMVision“, ein Open Source Project [WCM01] und die „Open CV Library“ von Intel [WIN05], die aber ebenfalls als Open Source Projekt zur Verfügung gestellt wird. Teilweise werden bei diesen Bibliotheken allerdings Fließkommaoperationen verwendet, womit sie ungeeignet für einen Einsatz am Blackfin sind. Aber auch jene Teile die keine FPU benötigen, sind natürlich nicht auf die DSP-Architektur optimiert. Da die Vorteile des DSP im Bezug auf den Blackfin-Assembler Befehlssatz ausgenutzt werden sollen, sind diese Software-Bibliotheken nicht geeignet für den Einsatz in diesem Projekt, zumal Analog Devices eigene Bibliotheken mit Bildverarbeitungsfunktionen im Blackfin-Assembler anbietet. Es sind dies vor allem globale Operatoren im Ortsbereich wie etwa die Dilatation und die Erosion oder der Sobel-Operator. Von der Dilatation und der Erosion (siehe Abschnitt 3.5.4) wird für ein Opening oder Closing (siehe Abschnitt 5.9.4) Gebrauch gemacht.

5.5 Treibersoftware

Sowohl mit, als auch ohne Betriebssystem sind Softwarepakete erforderlich, die die angeschlossenen Geräte konfigurieren und Funktionen für die Kommunikation zur Verfügung stellen. Als Beispiel sei hier das Kameramodul erwähnt. Diese Pakete sind notwendige Vorarbeiten für das Vision System und sind in diesem Abschnitt unter dem Stichwort „Treiber“ (engl. *„driver“*) zusammengefasst.

5.5.1 PLL

Die PLL regelt ausgehend vom montierten Quarz den Prozessortakt, sowie den Systemtakt. Mit dem Prozessortakt (engl. *„coreclock, CCLK“*) wird der Core und der interne Speicher getaktet. Der Systemtakt (engl. *„systemclock, SCLK“*) versorgt die Peripherie. Der Quarzoszillatortakt wird mittels eines internen Multiplikators auf die so genannte VCO-Frequenz erhöht. Davon wird je nach Registereinstellungen der System- bzw. Prozessortakt abgeleitet. Der maximale Prozessortakt liegt für den BF533 (Consumer Temperaturbereich) bei 600Mhz, der maximale Systemtakt bei 133Mhz. Die Einstellung der Taktfrequenzen erfolgt durch die Software mittels einer von Analog Devices definierten Programsequenz (siehe [HRM06]).

Für die Vision Unit wird aufgrund der hohen Anforderungen an die Rechenzeit, die höchstmögliche Prozessortaktfrequenz bei einem verwendeten Quarz von 25 Mhz eingestellt. Aufgrund des erforderlichen geraden Teilerfaktors ergeben sich die vollen 600 Mhz. Da der

Systemtakt vom Prozessortakt abgeleitet wird muss auch die Division dieser beiden einen geraden Teiler ergeben. Dies führt zu einem Systemtakt von 120 Mhz.

5.5.2 SDRAM-Controller

Für das SDRAM-Interface verfügt der Blackfin über einen eigenen Controller. Dieser muss je nach Größe und Organisation des angeschlossenen SDRAMs initialisiert werden um den Zugriff auf den Speicher zu ermöglichen. Die SDRAM Kontrollregister sind Teile der EBIU (*„external bus interface unit"*). Das entwickelte Prozessormodul (Coremodul) ist mit 32Mbyte bestückt (siehe Abschnitt 4.5.1). Der SDRAM-Controller unterstützt PC100 und PC133 kompatible SDRAM-Bausteine. Die vier assoziierten Register sind:

- SDRAM Memory Global Control Register (EBIU_SDGCTL)
- SDRAM Memory Bank Control Register (EBIU_DBCTL)
- SDRAM Memory Refresh Rate Control Register (EBIU_SDRRC)
- SDRAM Control Status Register (EBIU_SDSTAT)

Im „Memory Bank Control Register" werden Einstellungen über die Organisation des SDRAM vorgenommen, wie viele Banken bestückt sind, somit die Größe des SDRAMs und wieviele Adressleitungen verbunden sind.

Das „Global Control Register" beschreibt vor allem das Timing für den Zugriff auf das SDRAM. Zusätzlich lässt sich das SDRAM-Interface über dieses Register ein- und ausschalten und die Art der „Refreshzyklen" einstellen.

Da das SDRAM seine Informationen durch eine Selbstentladung der Speicherkondensatoren verliert, ist ein periodischer Refreshzyklus notwendig. Dieser wird durch einen Befehl vom SDRAM-Controller an das SDRAM initiiert. In welchen Abständen dieser Befehl gesendet wird, hängt vom Wert im „Memory Refresh Rate Control Register" ab. Dieser wird nach folgender Formel berechnet

$$RDIV = \frac{f_{SCLK} \cdot t_{REF}}{NRA} - \left(t_{RAS} + t_{RP}\right), \tag{5.1}$$

Wobei *RDIV* der in das Register EBIU_SDRRC geschriebene Wert ist, f_{SCLK} die Systemclockfrequenz in Hertz (siehe Abschnitt 5.5.1), t_{REF} die Refresh-Periode in Sekunden und *NRA* die Anzahl an Zeilenadressen. Die Werte t_{RAS} und t_{RP} sind jene aus dem „Global Control Register".

Mit folgenden Werten:

- t_{RAS} = 6,
- t_{RP} = 3,

- NRA = 8192, (13 Adressleitungen sind verbunden, 2^{13} = 8192)
- f_{SCLK} = 120 Mhz,
- t_{REF} = 64 ms,

ergibt sich *RDIV* zu 928 oder 0x3A0. Damit sind die Registereinstellungen für das Coremodul folgende:

EBIU_SDRRC = 0x03A0,

EBIU_SDBCTL = 0x0013,

EBIU_SDGCTL = 0x0091998d.

Wird der Wert für den Refreshzyklus kleiner gewählt, so ist dies unproblematisch für die Funktionalität, geht aber auf Kosten der Zugriffszeit, da eine sich im Refreshzyklus befindende interne SDRAM-Bank nicht gelesen werden kann. Die SDRAM interne Bank ist hier nicht zu verwechseln mit den SDRAM-Bänken des BF533.

Die beschriebene Initialisierung wird während des Bootvorganges (siehe Abschnitt 5.6) ausgeführt. Zusätzliche Informationen zum SDRAM-Controller des Blackfin finden sich in [HRM06].

5.5.3 GPIO

GPIO steht für *„general purpose input output“* und kennzeichnet allgemeine Ein-Ausgabeleitungen die mittels Software als Eingang oder Ausgang konfiguriert und beliebig gesetzt bzw. rückgesetzt werden können oder falls als Eingang konfiguriert kann der anliegende Spannungspegel gelesen werden.

Der GPIO –Treiber abstrahiert diese Hardwareebene und stellt der Software ein definiertes Interface zur Verfügung um GPIOs als Ein- oder Ausgang zu konfigurieren, zu setzen und rückzusetzen, sowie zu lesen.

Da jedes GPIO, beim Blackfin auch PF-Flag (*„purpose flag“*) genannt auch einen Interrupt (Unterbrechung des ablaufenden Programms um auf ein Erreignis reagieren zu können.) auslösen kann, wurde auch diese Funktionalität in den Treiber integriert.

5.5.4 SCCB

SCCB steht für *„serial camera control bus“* und ist ein dem I^2C-Standard ähnliches Businterface zur Steuerung von Kamerachips der Marke Omnivision. Da der BF533 kein solches Interface in Hardware hat, musste eine Softwareemulation über GPIO-Leitungen (*„general purpose input output“*) geschrieben werden. Aufgesetzt wurde der SCCB-Treiber auf den in 5.5.3 beschriebenen GPIO-Treiber, der die Hardware dahingehend abstrahiert und ein hardwareunabhängiges Interface dafür zur Verfügung stellt. Das SCCB Interface besteht

aus zwei Leitungen SIO_D, die bidirektionale Datenleitung und SIO_C die vom Master, in diesem Fall dem Blackfin, getrieben wird. Jeder Typ von Kamerachip hat eine Geräteadresse (engl. *„device address“*) und einen Satz von Registern die mittels einer Registeradresse (engl. *„subaddress“*) adressiert werden. Das Timing des Schreib- und Lesezyklus findet sich in [SCC02].

Prinzipiell wird bei jedem Zugriff zuerst die Geräteadresse inklusive des R/nW-Bits um anzuzeigen ob es sich um einen Schreib- oder Lesezugriff handelt, und dann die Registeradresse gesendet. Anschließend erfolgt das Lesen bzw. Schreiben des acht Bit breiten Datums.

Das Setzten bzw. Rücksetzten der GPIOs dem momentan bearbeiteten Bit des Schreib- oder Lesezyklus entsprechend, wird wie erwähnt über den GPIO-Treiber durchgeführt. Die erforderliche Wartezeit wird über eine Warteschleife mit Hilfe des Cycle Count „Registers“ [HRM06] realisiert.

Vier Funktionen wurden implementiert:

- sccbOpen: initialisiert die GPIOs und liefert einen Handle zurück.
- sccbWriteByte: Schreibt ein Byte an eine angegebene Registeradresse.
- sccbReadByte: Liest ein Byte von der angegebenen Registeradresse.
- sccbClose: Schließt die SCCB-Unterstützung.

Interrupts oder DMA werden nicht unterstützt.

5.5.5 Kamerainitialisierung

Das Modul OV7660 hat an die hundert Register für die Konfiguration. Die genaue Zahl wird vom Hersteller Omnivision nicht offen gelegt. Es soll an dieser Stelle auch keine Beschreibung der Register geben. Aufgrund des mit Omnivision unterzeichneten Übereinkommens zur Nichtveröffentlichung (engl. NDA, *„non disclosure agreement“*) ist dies rechtlich auch nicht möglich. Es soll aber trotzdem kurz auf die Konfiguration der Kamera eingegangen werden.

Der in der Kamera integrierte DSP verfügt neben den Funktionen für das Demosaiking und der Datenaufbereitung für die unterschiedlichen Datenformate (siehe Abschnitt 4.3.3) auch über Methoden für den automatischen Weißabgleich (engl *„automatic white balance, AWB“*). Ein halbautomatischer Weißabgleich ist nicht verfügbar. Ebenfalls implementiert ist eine automatische Einstellung der Belichtungszeit (engl *„automatic exposure control, AEC“*). Im Roboterfußballspiel kann aufgrund der definierten Umgebung und Lichtverhältnisse unter Umständen eine manuelle Einstellung der Belichtungszeit und des Weißabgleichs von Vorteil sein. Diese müsste vor Ort vorgenommen werden. Da wie im Abschnitt 1.2 erwähnt, die Erkennung jedoch auch unter „Bürobedingungen“ mit nicht vorher

definiertem Licht funktionieren sollte ist eine automatsiche Einstellung von Nöten. Beide Möglichkeiten werden im Treiber vorgesehen.

Notwendig ist natürlich auch eine Einstellung der Auflösung und der Framerate.

Das Flickerfilter, das typische Störungen durch das 50 Hz Flimmern der Beleuchtung eliminieren soll, wird ebenfalls eingeschaltet.

Datenformat:

Die Kamera liefert eine Vielzahl von Datenformaten (siehe Abschnitt 4.3.3). Für das RAW-Datenformat müsste das Demosaiking am Blackfin durchgeführt werden, was ein zusätzlicher Rechenaufwand wäre. Als Alternative bleiben noch RGB oder YUV. Die Kamera bietet die Möglichkeit eine ITU-R BT.656 (siehe Abschnitt 2.5.3) kompatiblen Datenstrom zu liefern. Auch das PPI-Interface des Blackfin unterstützt diesen Standard. Es ergeben sich weitere Vorteile durch dessen Verwendung:

- Es werden Steuerleitungen eingespart, die für eine andere Verwendung frei sind.
- Durch die Verwendung einer standardisierten Schnittstelle sollte das Ersetzen der Kamera durch ein anderes Modell unproblematisch sein.

Laut Spezifikation des ITU-Standards ist der bei der Übertragung verwendete Farbraum YUV4:2:2 (siehe Abschnitt 2.5.3). Trotzdem erlaubt es die Kamera auch unter Verwendung des ITU656 Formats den RGB565 Farbraum zu verwenden. Um aber standardkonform zu bleiben und da YUV ein geeigneter Farbraum ist für eine eventuell notwendige Konvertierung in einen anderen, oder möglicherweise sogar direkt für die Weiterverarbeitung geeignet ist, wird er verwendet. Mehr dazu in Abschnitt 5.9.

Pixeltakt:

Für die Kamera wird ein Oszillator von 25 Mhz verwendet. Für eine Auflösung von QVGA mit 60 fps und Verwendung des ITU-R BT.656 Modus ergibt sich für den von der Kamera gelieferten Pixeltakt 12,5 Mhz. Da das PPI-Interface des Blackfin maximal mit der Hälfte des Systemtaktes getaktet werden kann, muss dieser mindestens 25 Mhz betragen. Da wie in 5.5.1 festgelegt 120 Mhz verwendet werden, gibt es diesbezüglich keine Probleme. Dies reicht auch für eine VGA Auflösung bei 30 fps bei der sich der Pixeltakt auf 25 Mhz erhöht.

5.5.6 Kamerkalibrierung

Die Kamerakalibrierung dient im Allgemeinen dazu Linsenfehler und nichtlinearitäten des Chips zu kompensieren. Das Kameramodul OV7660 enthält bereits eine Linsenkorrektur „on Chip“. Ansonsten wird keine Kalibrierung der Kamera vorgenommen, da diese bei der Transformation der Bild- in die Weltkoordinaten implizit enthalten ist (siehe Abschnitt 5.10).

Wichtig ist der Einbau der Kamera. Hierbei muss darauf geachtet werden, dass sie möglichst horizontal ausgerichtet wird. Dies kann durch entsprechende Testbilder verifiziert werden.

5.5.7 PPI und DMA

Die Initialisierung des PPI-Interface und des dazugehörenden DMA-Kanals erfolgt ebenfalls über einen im Rahmen der Diplomarbeit entwickelten PPI-DMA-Treiber der diese Hardwareeinheit für die Anwendungssoftware abstrahiert. Der Treiber erlaubt es das PPI-Interface für jeden Modus zu initialisieren. Eine eigene Methode für den ITU-R BT656 Modus nimmt alle notwendigen Einstellungen dafür vor. Als Parameter dieser Funktion wird unter anderem die Auflösung der empfangenen Frames übergeben, um die korrekte Initialisierung des mit der PPI assoziierten DMA-Kanals vorzunehmen. Dieser schreibt die über die PPI empfangenen Daten ohne Belastung der CPU in das SDRAM. Eine genauere Beschreibung der DMA-Operationen finden sich in Abschnitt 5.8.

Für die Synchronisation werden entsprechende Steuerzeichen in den ITU656-Datenstrom eingebunden (siehe Abschnitt 2.5.3). Das PPI-Interface bietet die Möglichkeit diese Steuerzeichen aus dem Datenstrom herauszufiltern, so dass über DMA nur mehr die reinen Bilddaten in den Speicher geschrieben werden, wovon intensiv Gebrauch gemacht wird.

Genauere Informationen über das PPI-Interface des BF533 bietet [HRM06].

5.5.8 Flashspeicher

Der Flashspeicher dient zur Abspeicherung von Parametern und dem Programm selbst, das nach Anlegen der Versorgungsspannung vom internen Bootloader gestartet wird (siehe Abschnitt 5.6). Nach der Kompilierung des Programms mittels der Entwicklungsumgebung wird es über das integrierte „Flash Programming Tool“ ins Flash geschrieben. Zu diesem Zweck muss am Blackfin der Flashtreiber geladen und ausgeführt werden. Da die Hardware des Prozessormoduls eine Eigenentwicklung ist, musste auch der Flashtreiber selbst geschrieben werden.

Am Prozessormodul befindet sich ein intel-strata kompatibler Flash. Die Bezeichnung intel-strata gibt dabei die Art der Programmierung, sprich den Ablauf und die Befehlssequenzen, vor. Eine Befehlsreferenz sowie Ablaufdiagramme zu den einzelnen Befehlssequenzen finden sich in [NTL03].

Wie bei den anderen Treibern, abstrahiert auch der Flashtreiber die darunter liegende Hardware und bietet Methoden zum Lesen, Schreiben und Löschen des Flashspeichers.

5.6 Bootvorgang

Die Prozessoren der Blackfin Reihe unterstützen mehrere Bootmodi. Der hier verwendete soll an dieser Stelle kurz erläutert werden und einige für die Programmierung relevante Punkte herausgestrichen werden.

Verwendet wird der Bootmodus *„Boot from external flash“*. Dieser wird mittels zweier dedizierter Pins des BF533 eingestellt. Das mittels des VDSP++ für diesen Modus assemblierte Programm muss an der ersten Adresse des Flashspeichers beginnen. Die Struktur des Programms im Flash enthält Blöcke die dem internen Bootloader, der nach einem Reset in diesem Bootmodus gestartet wird, anzeigen, an welche Speicherstelle er den Programmblock kopieren soll. Anschließend setzt er den Programcounter an die Startadresse des internen Codespeichers und die Ausführung des Anwenderprogramms beginnt.

Damit falls erforderlich Blöcke auch ins SDRAM kopiert werden können muss vor dem Kopiervorgang der SDRAM-Controller initialisiert werden. Dies wird durch einen kleinen Programmblock erreicht der vor dem eigentlichen Anwenderprogramm im Flashspeicher steht und vom prozessorinternen Bootloader vor dem Kopiervorgang ausgeführt wird. Dieser Programmblock enthält den Code für die Initialisierung des SDRAMs die natürlich an die im Rahmen dieses Projekts entwickelte Hardware angepasst werden musste. In [HRM06] finden sich detailierte Informationen zum Bootprozess des Blackfin BF533.

5.7 Aufnahmetests

Die Aufnahmen aus Abbildung 5.2 zeigen bereits eine wichtige Problematik in der Auswertung der Bilder. Aufgenommen wurde in beiden Fällen ein oranger Golfball doch der Farbverlauf ändert sich markant bei Änderung der Lichtverhältnisse. Das bedeutet, dass die Grenzwerte für die Farbklassifizierung entsprechend tolerant gewählt werden müssen, was wiederum eine Fehl-Erkennung (engl. *„false positive“*) begünstigt. Abhilfe schaffen kann hier eine zusätzliche Berücksichtigung des Kantenbildes. Die Annahme, dass durch die Verwendung eines Farbraumes der Helligkeits- und Farbkänäle trennt, Helligkeitsschwankungen des Umgebungslichtes die Farberkennung nicht beeinflussen gilt nur, wenn die Beleuchtungsstärke nicht allzu sehr schwankt. Wie man an den Bildern deutlich erkennen kann, kommt beim Golfball noch seine spiegelnde Oberfläche erschwerend hinzu. Unter Beschränkung auf das Mirosot Spiel kann man aber davon ausgehen, dass die Beleuchtung ausschließlich senkrecht von oben erfolgt.

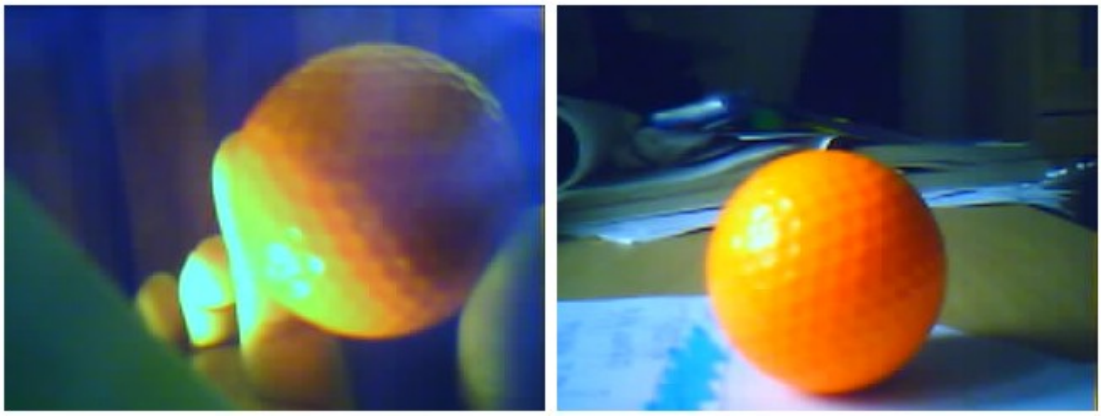

Abbildung 5.2: Aufnahmetests bei unterschiedlichen Lichtverhältnissen

5.7.1 AEC und AWB

Wie im vorigen Abschnitt erwähnt, wirken sich Unterschiede in der Beleuchtung auch in Unterschieden im Farbverlauf und damit in den Farbkanälen aus. Entscheidend dafür sind die Belichtungsdauer und deren Steuerung (engl. *„exposure control“*). Da unterschiedliche Lichtquellen mit unterschiedlicher spektraler Verteilung ebenfalls unterschiedliche Farbverläufe erzeugen, ist weiters ein Weißabgleich (engl. *„white balance“*) erforderlich, um weitgehende Farbechtheit zu ermöglichen. Die Steuerung der Belichtungsdauer und der Weißabgleich können automatisch im Kameramodul selbst erfolgen (*„automatic exposure control“* AEC und *„automatic white balance“* AWB), sodass in den meisten Fällen keine nachträgliche Korrektur mehr erforderlich ist. Da die Farben der zu erkennenden Objekte a priori bekannt sind, wäre allerdings eine speziell auf diese Farben abgestimmte, nachträgliche Korrektur der Helligkeit oder ein nachträglicher Weißabgleich denkbar. Versuche haben aber gezeigt, dass eine Korrektur allein durch Berücksichtigung bestimmter Bildmerkmale wie etwa die hellste und dunkelste Stelle sehr schwierig und aufwendig sind und oft nicht zu den gewünschten Resultaten führen, weshalb sie nicht weiter verfolgt wurden. In den meisten Fällen verlassen wir uns für das vorliegende Projekt deshalb auf die in das Kameramodul integrierte AEC und AWB. Nur in Ausnahmefällen zur Optimierung auf bestimmte konstante Lichtverhältnisse kann die Belichtungsdauer manuell eingestellt werden. Diese ist dann aber durch geeignete Testaufnahmen vor dem Betrieb durch Beschreiben der entsprechenden Kameraregister einzustellen.

Erwähnt sei an dieser Stelle noch, dass AWB und AEC zwei iterative Prozesse sind, die den gezeigten Rückkopplungen im Schichtenmodell entsprechen.

5.7.2 Auflösung und Objektgrößen

Eine wichtige Frage die es zu klären gilt, ist jene nach der erforderlichen Auflösung um die Anforderungen an den Algorithmus erfüllen zu können. Je kleiner die benötigte Auflösung desto schneller die Erkennung da weniger Daten bearbeitet werden müssen. Andererseits bedeutet eine geringere Auflösung, dass bei entsprechender Entfernung weniger Pixel pro Objekt zur Verfügung stehen. Wenn man sich die Anforderungen aus Abschnitt 1.2 in Erinnerung ruft, so ist eine Erkennung des Balles bis zu einer Entfernung von einem Meter erforderlich. Die Abbildung 5.3 zeigt den Ball in diesem Abstand bei einer Auflösung von QVGA (320 x 240 Pixel). Nur bei dieser Auflösung liefert das Kameramodul auch die geforderte Framerate von 60 fps. Der Ball ist noch erkennbar, wenngleich wenige Pixel zur Erkennung zur Verfügung stehen. Die Genauigkeit der Abstandsbestimmung wird in diesem Fall deshalb stark abnehmen. Wichtiger ist in einer solchen Situation aber der Winkel den der Roboter zum Ball einnimmt, wobei keine sehr hohe Genauigkeit erforderlich ist, da die Trajektorie zum Ball ohnehin mit jedem Frame nachgeregelt wird. Bei einem Abstand zum Roboter von 50 cm (Abbildung 5.3 links) ist der Ball schon sehr gut zu erkennen. Aus genannten Gründen ist eine Auflösung von QVGA ausreichend, womit auch die erforderliche Framerate geliefert werden kann.

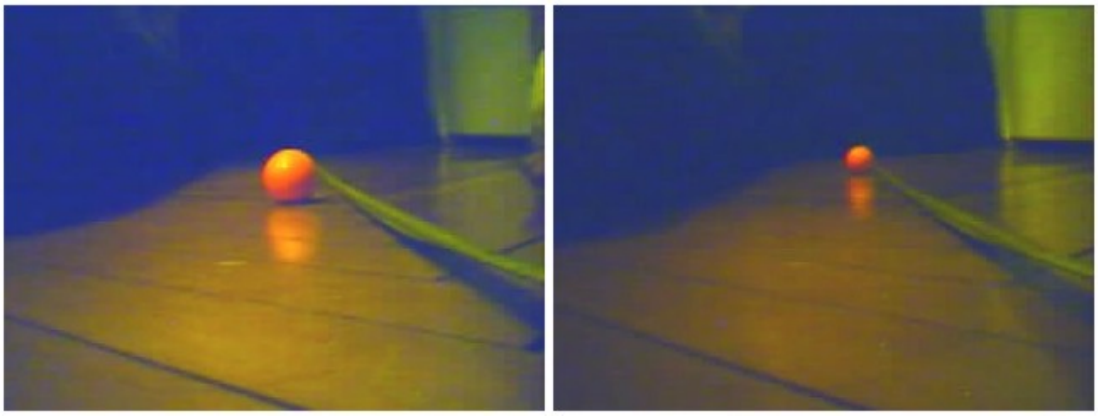

Abbildung 5.3: Ball in 50 cm und 100 cm Entfernung

5.8 DMA und Speicherverwaltung

Um auf die notwendige Performance sprich die erforderlichen Ausführungszeiten zu kommen, ist eine intensive Nutzung von DMA unerlässlich. Dabei müssen in erster Linie die Anzahl der Speicherzugriffe vom Programm aus, sprich unter Nutzung des Prozessors, minimiert werden und zweitens versucht werden, diese auf den mit Prozessortakt arbeitenden internen L1-Speicher zu beschränken. Bei maximalen Taktfrequenzen arbeitet der interne

Speicher mit 600 Mhz das SDRAM mit 133 Mhz. Dies bedeutet eine theoretische Verfünffachung der Ausführungszeit bei einem ausschließlichen Zugriff auf den internen Speicher. In der Praxis bedeutet es sogar eine Versiebenfachung, wie aus Tabelle 5.1 ersichtlich wird. Sie zeigt einen Vergleich der Ausführungszeiten von in der Programmiersprache „C“ programmierten Schleifen mit 100 000 Lese- bzw. Schreibzugriffen auf die unterschiedlichen Speicher des BF533. Dabei wurde der Programmcode einmal im internen SRAM und einmal im SDRAM abgelegt. Gelesen bzw. geschrieben wurden jeweils eine 32 Bit große Zahl („unsigned long“ in der Programmiersprache „C“) an dieselbe Speicherstelle. Vergleichend dazu die Ausführungszeiten von DMA-Transfers mit derselben Anzahl Bytes, einmal vom SDRAM ins interne SRAM und umgekehrt. Da der interne Speicher für die Datenmenge nicht groß genug ist, wurden Blöcke zu 3200 Byte kopiert, wobei der interne Speicher immer wieder überschrieben wurde. Der Prozessortakt betrug für alle Versuche 600 Mhz, der Systemtakt 120 Mhz.

Tabelle 5.1: Ausführungszeiten von Schleifen mit 100 000 Speicherzugriffen

	Schreibzugriffe	**Lesezugriffe**
Code im L1 SRAM	Daten im L1 SRAM: 2,3 ms Daten im SDRAM: 16,5/3,5[1)] ms	Daten im L1 SRAM: 2,3 ms Daten im SDRAM: 16,5 ms
Code im SDRAM	Daten im L1 SRAM: 32 ms Daten im SDRAM: 54,6/43[1)] ms	Daten im L1 SRAM: 32 ms Daten im SDRAM: 54,6 ms

	SDRAM in L1 SRAM	**L1 SRAM in SDRAM**
DMA – Transfers	1,3 ms	0,18 ms [1)]

Ein besonders deutlicher Unterschied in der Ausführungszeit zeigt sich zwischen der Verwendung von DMA und Speicherzugriffen mittels des Prozessors um Daten vom SDRAM in den internen L1 Cache zu kopieren. Hier haben wir einen Faktor von 12.

Die Verwendung von DMA bietet einen weiteren Vorteil, nämlich die Möglichkeit der Parallelisierung. Während Speicherbereiche mit neuen Daten per DMA gefüllt werden, können gleichzeitig bereits kopierten Daten bearbeitet werden.

[1)] Die geringere Zeit ergibt sich, wenn nicht auf den Abschluss des Buszugriffes gewartet wird (*“ssync”*). Die Daten werden vor dem eigentlichen SDRAM-Zugriff in einen Buffer der EBIU geschrieben.

In den folgenden Abschnitten wird die Konfiguration der DMA-Einheit des BF533 für die Objekterkennungssoftware beschrieben.

5.8.1 Framebuffer

Der erforderliche Speicherplatz S für ein Frame errechnet sich zu:

$$S = X_{res} \cdot Y_{res} \cdot BPP\,, \tag{5.2}$$

Wobei X_{res} und Y_{res} die Anzahl der Pixel in X und Y und BPP (*"bits per pixel"*) die Anzahl der Bits pro Pixel angibt. Bei einer QVGA Auflösung (320 x 240 Pixel) und der Verwendung des YUV4:2:2 (16 Bit pro Pixel) Formates ergibt sich der erforderliche Speicherbedarf pro Frame zu 1228800 Bits, gleich 153600 Bytes. Laut Tabelle 4.3 beträgt der interne Speicher des BF533 148 kByte. Davon stehen aber nur 64 kByte für die Datenspeicherung zur Verfügung. Der Rest des internen Speichers ist für die Speicherung des Programms (Instruction RAM). Das interne SRAM ist somit zu klein um ein gesamtes Bild von der Kamera speichern zu können. Aus diesem Grund wird es im SDRAM abgelegt. Dies erfolgt ohne zutun des Prozessors über die DMA-Funktionalität des Blackfin. Es werden dabei zwei Speicherbereiche im SDRAM (Framebuffer) reserviert, die von der DMA-Einheit abwechselnd beschrieben werden. So kann ein Bild in einem Framebuffer bearbeitet werden, während gleichzeitig der zweite Puffer vom DMA-Controller beschrieben wird. Der DMA-Kanal arbeitet dabei im so genannten „Deskriptormodus" bei der ohne Eingriff der Software jedes Frame vom PPI-Interface abwechselnd in die beiden Buffer geschrieben wird. Dazu sind zwei Deskriptoren notwendig die vom DMA-Controller nach komplettiertem Zyklus automatisch abwechselnd geladen werden. Ein Deskriptor enthält dabei die Startadresse des ersten Buffers der zweite Deskriptor jene des zweiten. Die DMA-Einheit des Blackfin bietet die Möglichkeit zu zweidimensionalen DMA-Transfers. Gemeint ist damit die Möglichkeit einen Speicherbereich blockweise zu Lesen bzw. Schreiben. Angegeben werden müssen nur die Blockgröße und deren Anzahl. Dabei kann nach jedem übertragenen Block oder nach Ablauf des gesamten Zyklus ein Interrupt ausgelöst werden. Eine blockweise Übertragung wäre im vorliegenden Fall nicht notwendig. Da die Register für die Anzahl der zu übertragenden Elemente aber nur 16 Bit breit sind, ist eine maximale Übertragung von 65535 Elementen pro DMA-Zyklus möglich. Ein Element besteht dabei aus einem, zwei oder vier Byte. Um die Anzahl an Buszugriffen zu verringern, können jeweils vier über das PPI-Interface empfangene Bytes zu einem so genannten Burst zusammengefasst werden und mit einem einmaligen Bustransfer übertragen werden. Da die Übertragung zum SDRAM aber wiederum nur mit 16 Bit Datenbreite pro Zugriff erfolgen kann und auch der interne Bus nur 16 Bit breit ist, ist kein weiterer Geschwindigkeitsvorteil ersichtlich. Um Performanceverluste zu vermeiden ist aber ein Zusammenfassen von jeweils zwei Byte zu einem Burst unbedingt erforderlich.

Nach jedem übertragenem Frame wird ein Interrupt ausgelöst und eine globale Variable aktualisiert die den momentan vom DMA gefüllten Framebuffer anzeigt.

5.8.2 Cache der Bilddaten

Tabelle 5.1 zeigt, dass eine effiziente Programmausführung nur dann möglich ist, wenn sich Daten und Programmcode im internen L1-Speicher befinden. Da die Framebuffer, wie in obigem Abschnitt beschrieben, gezwungenermaßen im SDRAM liegen müssen, ist eine stückweise Pufferung der Bilddaten zur Bearbeitung im internen Speicher unumgänglich. Dazu wird das Bild in Blöcke unterteilt, die der Puffergröße im internen L1 RAM entsprechen. Versuche haben gezeigt, dass ab einer Puffergröße von ein oder zwei Bildzeilen, kein nennenswerter Geschwindigkeitsvorteil durch eine weitere Vergrößerung des Puffers erzielbar ist, da die Ausführungszeit des Algorithmus der auf die Bilddaten arbeitet, in den meisten Fällen höher ist, als jene des DMA-Transfers. Die Anzahl der Zeilen die jeweils gepuffert wird, muss natürlich durch die vertikale Auflösung des Bildes teilbar sein. Manche Algorithmen, vor allem jene zur Kantenerkennung, müssen auch Pixel in ihrer vertikalen Nachbarschaft, sprich über oder unter dem aktuell bearbeiteten Pixel, berücksichtigen. Aus diesem Grund ist es notwendig eine größere Anzahl von Zeilen im Puffer zu halten. Die Größe sollte aber im Rahmen bleiben, da auch andere Datenstrukturen aus Gründen der Zugriffszeit im internen Speicher abgelegt werden müssen. Wird die Anzahl der Zeilen jedoch zu klein gewählt erhöht sich der Overhead zur Synchronisation und Initialisierung der DMA-Transfers. Ein guter Kompromiss der die eben genannten Faktoren vereint, ist eine Puffergröße von vier Zeilen. Die benötigte Speichergröße für einen Puffer bei einer horizontalen Auflösung des Bildes von 320 Pixeln und einem Speicherbedarf von zwei Byte pro Pixel beträgt somit 2560 Byte. Die Anzahl der DMA-Transfers um ein gesamtes Frame bei einer vertikalen Auflösung von 240 Pixeln zu bearbeiten beträgt 60.

Ping-Pong-Schema:

Um das Kopieren der Daten und die Ausführung des Algorithmus parallelisieren zu können, sind zwei Eingangs- und zwei Ausgangspuffer im internen L1 SRAM notwendig, die in einem so genannten Ping-Pong-Schema abwechselnd beschrieben, bzw. ausgelesen werden. Je ein Eingangspuffer wird aus dem SDRAM befüllt und gleichzeitig ein Ausgangspuffer ins SDRAM zurück geschrieben, während der Algorithmus auf dem jeweils anderen Aus- bzw. Eingangspuffer arbeitet. Eine graphische Aufbereitung dieser Ping-Pong-Pufferung zeigt Abbildung 5.4.

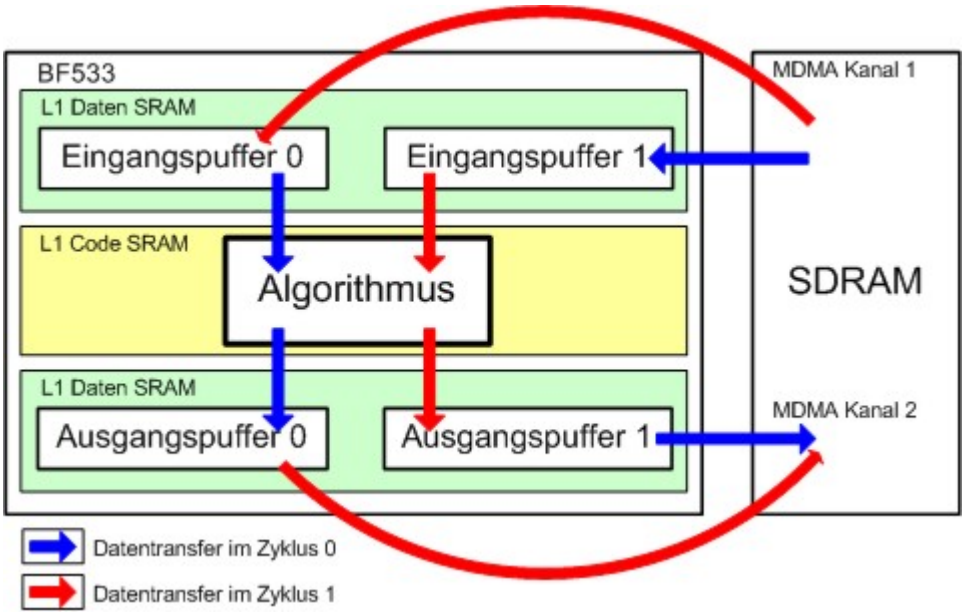

Abbildung 5.4: Ping-Pong-Schema der internen Pufferung

5.8.3 Synchronisation der DMA-Transfers

Die Abbildung 5.5 gibt einen Überblick über die im Blackfin für die Objekterkennung ablaufenden DMA-Transfers. Obige Abschnitte haben die einzelnen Transfers näher erläutert. Hier noch einmal in einer Zusammenfassung:

- Kopieren der Daten der Kamera in das SDRAM, abwechselnd in zwei verschiedene Puffer.
- Blockweiser Transfer der Bilddaten vom SDRAM in das interne L1 SRAM, wiederum abwechselnd in zwei verschiedene Puffer.
- Abwechselndes, blockweises Zurückschreiben der Ergebnisdaten in das SDRAM aus den Zwei Ergebnispuffern (Ausgangspuffer).

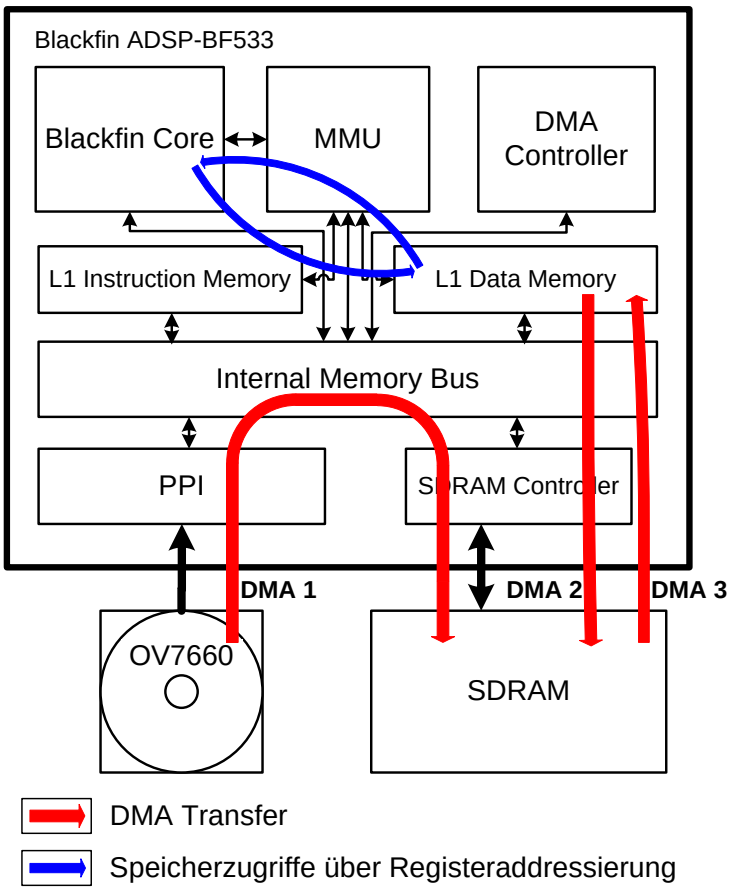

Abbildung 5.5: DMA-Transfers für den Bildverarbeitungsalgorithmus

Die einzelnen DMA-Transfers sind natürlich nicht unabhängig voneinander, sondern müssen synchronisiert werden um einen Datenverlust zu vermeiden. Das Timing wird dabei im Wesentlichen vom Kopieren der Bilddaten bestimmt. Diese werden automatisch ohne Eingriff der CPU mittels zweier Deskriptoren übertragen (siehe Abschnitt 5.8.1). Um das Füllen des internen Buffers zu starten, wird der DMA-Transfer in Blöcke unterteilt, die der Größe der internen Eingangspuffer entsprechen, in diesem Fall vier Zeilen (siehe Abschnitt 5.8.2). Der Algorithmus wartet bis genug Bildzeilen von der Kamera ins SDRAM kopiert wurden, im vorliegenden Fall sind das vier Zeilen. Dies ist über ein Auslesen der zum entsprechenden DMA-Kanal gehörenden Register problemlos zu bewerkstelligen. Anschließend wird der DMA-Transfer gestartet, der diese Bildzeilen in den internen Eingangspuffer 0 kopiert. Sind weitere vier Zeilen im SDRAM bereit, wird auch der Transfer zum Füllen des zweiten Eingangspuffers im internen L1 SRAM gestartet. Diese und die nachfolgenden Schritte sind im Flussdiagramm aus Abbildung 5.6 dargestellt.

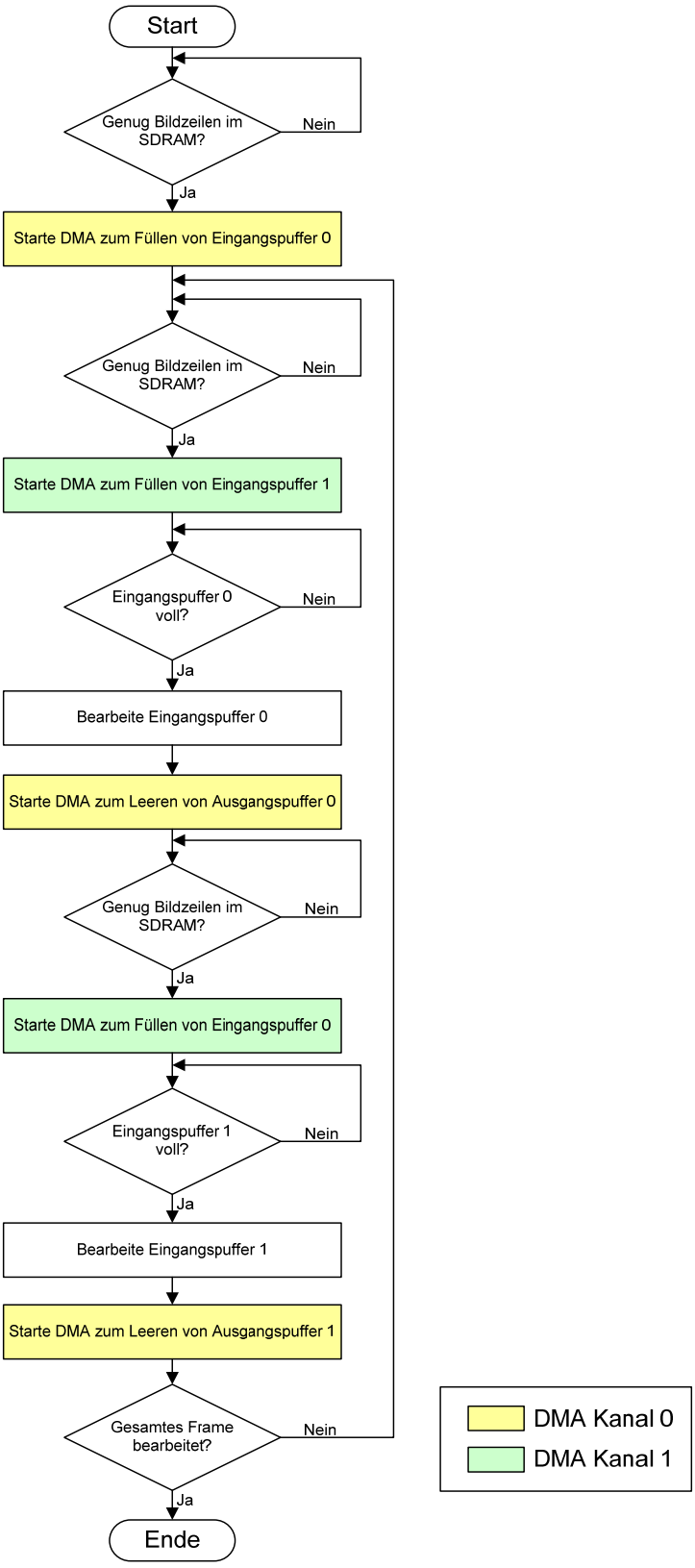

Abbildung 5.6: Flussdiagramm zur DMA Synchronisation

Ein kompletter Durchlauf dieses Flussdiagramms bewerkstelligt die Bearbeitung eines kompletten Frames, sprich aller Bildzeilen. Die Art der Bearbeitung ist dabei beliebig. Das entsprechende Prozessfenster im Flussdiagramm steht stellvertretend für die in den nächsten Abschnitten dargelegten Verarbeitungsschritte. Die grün und gelb hinterlegten Prozessschritte bezeichnen die beiden an den Transfers beteiligten DMA-Kanäle, jeweils das Füllen eines Eingangspuffers und das Leeren des entsprechenden Ausgangspuffers laufen parallel und werden von zwei verscheidenen Kanälen ausgeführt (siehe Abbildung 5.5). Da es, wie in Abschnitt 5.8.1 beschrieben, zwei Speicherbereiche im SDRAM gibt, die abwechselnd vom DMA-Kanal des PPI-Interfaces mit Bilddaten beschrieben werden, müssen die DMA-Kanäle die die internen Eingangspuffer füllen alternierend aus diesen beiden Bildpuffern lesen. Diese Überprüfung ist im obigen Flussdiagramm nicht eingezeichnet. Um zu wissen welcher der beiden Speicherbereiche im SDRAM gerade vom PPI-DMA beschrieben wird wird eine globale Variable *g_cCurrentFrameBuffer* deklariert, die von der Interruptfunktion (*„interrupt service routine“*, ISR) des PPI-DMA-Kanals aktualisiert wird. Dieser Interrupt wird nach jedem transferierten Frame aufgerufen und führt die im Flussdiagramm aus Abbildung 5.7 dargestellten Operationen aus. Die beiden globalen Variablen *g_nFrameCounter* und *g_nFrameLostCounter* dienen zu Kontrollzwecken.

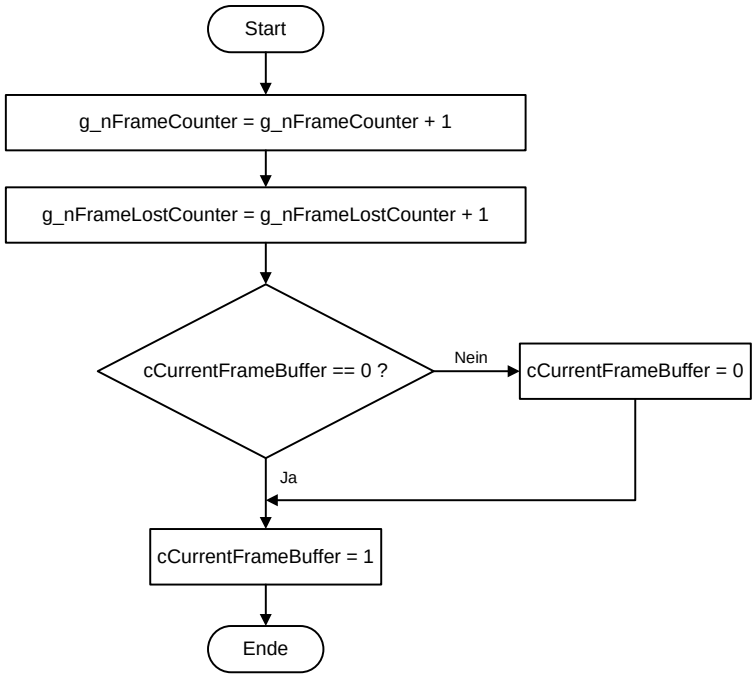

Abbildung 5.7: Flussdiagramm der ISR des PPI-DMA-Kanals

5.9 Der Objekterkennungsalgorithmus

In obigen Abschnitten wurden bereits einige Abläufe und wichtige Treiberfunktionalitäten bzw. das DMA-Modell gezeigt. Der eigentliche Algorithmus zur Objekterkennung gliedert sich im Sinne des Schichtenmodells aus Abbildung 3.1 in mehrere Teile. Die Aufnahme der Szene mit Hilfe der Farbkamera wurde dabei bereits ausführlich behandelt. Die nächsten Abschnitte sollen die einzelnen Schritte genauer beleuchten.

5.9.1 Vorverarbeitung des Bildes

Zur Vorverarbeitung gehören im Wesentlichen Algorithmen zur Verbesserung der Bildqualität. Aufgrund des in dem Kameramodul integrierten DSPs (siehe Abschnitt 4.3.3) der schon Funktionen zur Bildverbesserung enthält, bis hin zu einer Methode für die Kantenschärfung, entfällt im Allgemeinen eine algorithmische Vorverarbeitung zur Steigerung der Bildqualität, solange sich die Umgebungsbeleuchtung im Rahmen hält. Im Rahmen, bedeutet in diesem Fall Tageslicht oder ausreichend künstliche Beleuchtung zum Einen und keine Überbelichtung durch zu starkes Scheinwerferlicht zum Anderen. Quantativ bedeutet dies für die verwendete Kamera eine Beleuchtung zwischen 300 und 3000 Lux. Darüber und darunter stoßen die AEC und AWB (siehe Abschnitt 5.7.1) der Kamera an ihre Grenzen und das Bild erscheint Über, bzw. Unterbelichtet. Abhilfe kann dann eine manuelle Einstellung der Belichtungszeit schaffen, die über Kameraregister einstellbar ist oder es wird mittels der in Abschnitt 3.4.2 gezeigten linearen Skalierung das Bild aufgehellt oder verdunkelt. Eine Aufhellung funktioniert im Allgemeinen sehr gut, eine Verminderung der Helligkeit aufgrund Übersteuerungen im Bild eher weniger, in diesem Fall ist eine Reduzierung der Belichtungszeit empfehlenswert. Um feststellen zu können ob das Bild- über oder Unterbelichtet ist, wird der Mittelwert über den Luminanzkanal gebildet. Liegt dieser unterhalb eines Schwellwertes so ist das Bild unter- über einem bestimmten Grenzwert überbelichtet. Der Mittelwert wird laufend mitberechnet. Dies geschieht im Prozessfenster „Bearbeite Eingangspuffer x“ aus dem Flussdiagramm in Abbildung 5.6. Wird in einem Frame eine Über- oder Unterschreitung der Schwellwerte festgestellt, so wird im folgenden Frame, ebenfalls im selben Prozessfenster die entsprechende Skalierung vorgenommen, oder es wird die Belichtungszeit der Kamera umgestellt, wobei dies eine Verzögerung mit sich bringt, da das Beschreiben der Kameraregister eine gewisse Zeit benötigt.

5.9.2 Wahl eines geeigneten Farbraums

Die Kamera liefert die Bilder im YUV-Farbraum, wie im ITU-R BT.656 Standard beschrieben ist (siehe Abschnitt 2.5.3). Bei diesem Farbraum werden die Farbinformationen (Crominanz) und die Helligkeitsinformation (Luminanz) bereits getrennt übertragen. Dies ist vorteilhaft für eine Farbsegmentierung und Kantenerkennung. Die Farbsegmentierung wird

dabei auf den Crominanzkanälen, die Kantenerkennung auf dem Luminanzkanal ausgeführt. Sieht man sich die drei Kanäle des YUV-Farbraums jedoch getrennt als Graubilder an (siehe Abbildung 5.8) erkennt man deutlich, dass der Kontrast in den Farbkanälen eher gering ist.

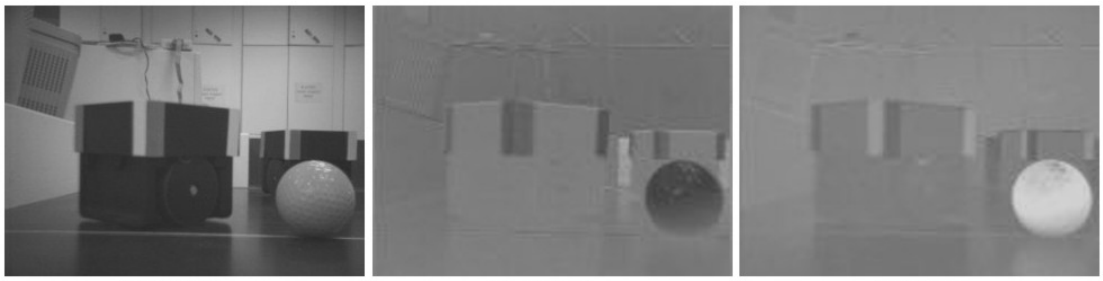

Abbildung 5.8: Kanaltrennung eines Farbbildes im YUV-Farbraum

Der Y-Kanal entspricht dem nach Farbempfindlichkeit gewichteten Graustufenbild und würde sich für eine Weiterverarbeitung eignen. Um aber die Segmentierung nach Farbklassen robuster zu machen, wird nach einem Farbraum gesucht, der in den Farbkanälen größere Kontrastwerte aufweist. Ausgehend vom HSI-Farbmodell (siehe Abschnitt 2.3.4) wird in [JSM02] ein Farbraum vorgeschlagen, der anstelle des Farbtons *hue* einen Wert H' basierend auf den CIELab-Farbraum verwendet, der wie folgt berechnet wird:

$$H' = tan^{-1}\frac{b^*}{a^*}, \tag{5.3}$$

wobei b^* und a^* aus dem CIELab-Farbraum stammen und definiert sind als:

$$a^* = Rot - Grün, \qquad b^* = Gelb - Blau\ . \tag{5.4}$$

Die Berechnung erfordert aber einen arcustangens, der ohne FPU in akzeptabler Ausführungszeit nur mit einer Lookup Tabel realisierbar ist. Der Wertebereich von a^* und b* liegt zwischen -128 bis +127. Dies bedeutet es ist eine Tabelle mit 256² - 256 = 65280 Einträgen notwendig. Zudem bedeutet es einen zusätzlichen Rechenaufwand. Das HSI-Modell allein bietet aber in den beiden Farbkanälen H und S bereits einen weitaus größeren Kontrast als dies im YUV-Modell der Fall ist, wie Abbildung 5.9 zeigt.

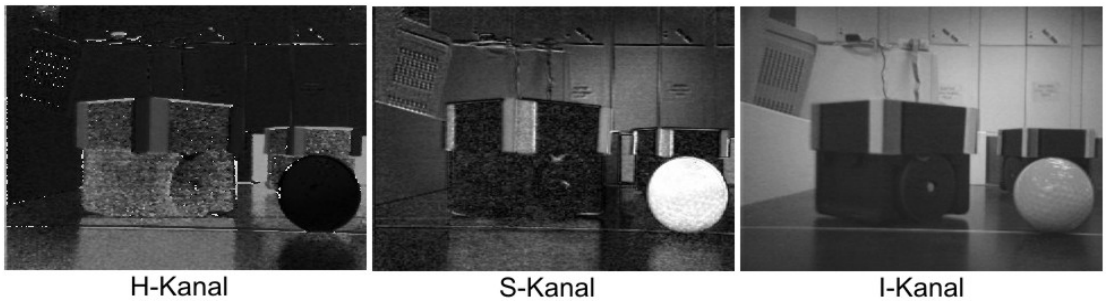

Abbildung 5.9: Kanaltrennung eines Farbbildes im HSI-Farbraum

In den beiden Helligkeitskanälen Y und I besteht kaum ein sichtbarer Unterschied, wenn die beiden auch etwas unterschiedlich gebildet werden (siehe Abschnitt 2.3). Aus diesem Grund kann auf eine Berechnung des I-Kanals verzichtet werden, und es werden nur die Kanäle H und S aus dem YUV-Farbraum gebildet. Die Berechnung erfolgt mittels Umweg über den RGB-Raum. Aus diesem ergeben sich dann mit Hilfe einer nichtlinearen Transformation die beiden Kanäle H und S. Die Vorschriften für die Bildung des RGB-Tripels aus dem YUV Farbraum lauten wie folgt:

$$R = V - 127 + Y\,, \tag{5.5}$$

$$B = U - 127 + Y\,, \tag{5.6}$$

$$G = \frac{433 \cdot Y - 133 \cdot R - 49 \cdot B}{256}\,. \tag{5.7}$$

Der Abzug von 127 ist notwendig, da der Wertebereich von U und V -127 bis +128 beträgt. Dieses Tripel wird dann unter Verwendung der Berechnungsvorschriften aus Gleichung (5.9) und (5.10) in ein HS Tupel umgeformt.

$$I = \frac{R + G + B}{3}\,, \tag{5.8}$$

$$H = \begin{cases} 0, & \textit{für} \quad max = min \\ 60° \cdot \dfrac{G - B}{max(R,G,B) - min(R,G,B)}, & \textit{für} \quad max = R \quad und \quad G \geq B \\ 60° \cdot \dfrac{G - B}{max(R,G,B) - min(R,G,B)} + 360°, & \textit{für} \quad max = R \quad und \quad G < B \\ 60° \cdot \dfrac{B - R}{max(R,G,B) - min(R,G,B)} + 120°, & \textit{für} \quad max = G \\ 60° \cdot \dfrac{R - G}{max(R,G,B) - min(R,G,B)} + 240°, & \textit{für} \quad max = B \end{cases} \tag{5.9}$$

$$S = 255 - 3 \cdot \frac{min(R,G,B) \cdot 261120}{(R + G + B) \cdot 1024} \tag{5.10}$$

Der Vollständigkeit halber ist auch die Gleichung für die Berechnung von I angegeben. Da dieser Kanal für die Farbklassifizierung aber nicht verwendet wird, und zur Kantendetektion der Y-Kanal aus dem YUV-Modell herangezogen wird, muss er nicht berechnet werden.

Diese Formeln sind nicht identisch mit jenen aus dem in Abschnitt 2.3.4 vorgestellten FHS Farbraum. Dies liegt zum einen daran, dass die Gleichungen auf den verwendeten Wertebereich von 0 bis 255 skaliert werden müssen und zum anderen daran, dass eine weitere Skalierung vorgenommen werden muss um ganzzahlige Divisionen ohne die Notwendigkeit von Fließkommaoperationen durchführen zu können. Da der Blackfin keine Hardware-Unterstützung für Fließkommaoperationen bietet. Ein weiterer Grund für die geänderten

Gleichungen liegt in einer Adaption ohne die Verwendung trigonometrischer Funktionen. Es handelt sich also um Näherungen zu Gunsten geringerer Ausführungszeit.

Ausgangspunkt der Farbraumkonvertierung ist der im ITU-R BT.601 festgeschriebene Farbraum YUV4:2:2 wobei 4:2:2 eine Unterabtastung der Cromakanäle bedeutet (siehe Abschnitt 2.5.2). Dies ist das Farbmodell das die Kamera liefert. Die Unterabtastung bedeutet, dass zur Berechnung zweier aufeinander folgender RGB-Trippel zweimal dieselben U und V Werte herangezogen werden müssen. Dies wird im Algorithmus natürlich berücksichtigt.

5.9.3 Kantendetektion und Farbklassifizierung

Beide Operationen, sowohl die Kantendetektion als auch die Farbklassifizierung lassen sich im Schichtenmodell aus Abbildung 3.1 bei der Segmentierung einordnen. Als Segmente oder Komponenten erhält man Linien und Flächen definierter Farbe.

Kantendetektion:

Im Kapitel Kapitel 3 wurden einige Methoden zur Kantendetektion vorgestellt. Sie basieren auf einer linearen Filteroperation im Ortsbereich. Die Aufwandsabschätzungen zeigen aber, dass sie sehr rechenintensiv sind, da pro Pixel mehrere Multiplikationen und Additionen erforderlich sind. Da die Verarbeitungsschritte der Segmentierung aber in einem Schleifendurchgang durchgeführt werden sollen, wird ein neues Verfahren zur Kantendetektion vorgestellt. Wie in Kapitel Kapitel 3 bereits dargestellt, handelt es sich bei einer Kante um eine lokale, sprunghafte Änderung der Helligkeit oder der Grauwerte. Dieser Umstand wird direkt ausgenutzt. Es wird für jeden Pixel die Differenz zum übernächsten linken und übernächsten unteren Nachbarn bestimmt. Übersteigt diese Differenz einen Schwellwert, so wird dieser Pixel als Kantenpunkt markiert. Durch die Berücksichtigung des oberen und unteren Nachbarn, erhält man die Kanten in horizontaler und vertikaler Richtung. Die Kanten werden etwas dicker als bei der Verwendung beispielsweise eines Sobeloperators (siehe Abschnitt 3.5.2), sind aber sehr akkurat wie Abbildung 5.10 zeigt. Der Kantenalgorithmus wurde dabei auf den Y-Kanal des links stehenden Bildes durchgeführt.

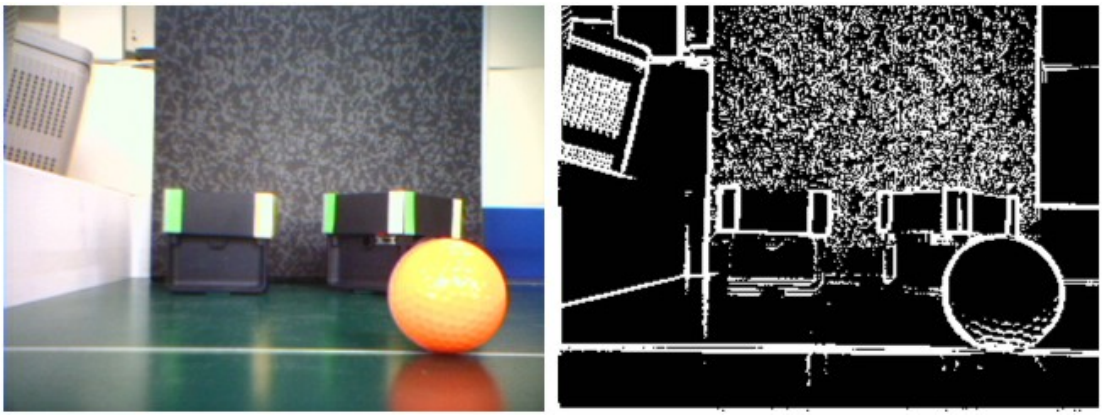

Abbildung 5.10: Kantenextraktion

Durch die Wahl des Schwellwertes, kann man den Algorithmus sehr gut beeinflussen. Für Abbildung 5.10 wurde ein Schwellwert von 30 verwendet, bei einem Wertebereich der Grauwerte von 0 bis 255. Wählt man den Schwellwert geringer, erhält man weniger und teilweise nicht mehr durchgehende Kanten. Wählt man ihn größer so werden auch größeres Bildrauschen und sonstige Unsauberkeiten im Bild als Kanten erkannt. Punktuelles Bildrauschen ist für diese Art der Kantendetektion kein Problem. Einzelne fehlerhafte Punkte werden nicht als Kante erkannt.

In Abschnitt 5.8.2 wurde die Größe der internen Puffer für die Zwischenspeicherung der Bilddaten diskutiert. Dabei wurde eine Puffergröße für vier Zeilen festgelegt. Auf eine Besonderheit aufgrund des Kantenalgorithmus sei an dieser Stelle noch hingewiesen. Wie beschrieben, werden horizontale Kanten dadurch gefunden, dass die Differenz in den Grauwerten übereinander liegender Pixel mit einer Zeile Abstand gebildet wird. Dies lässt pro Pufferzyklus nur die Bildung der Kanten für die ersten beiden Zeilen zu, da für die anderen beiden Zeilen auch die daran anschließenden zwei Zeilen gebraucht werden, diese aber erst im nächsten Zyklus geladen werden. Aus diesem Grund wird der Kantendetektionsalgorithmus nur auf zwei der vier sich im internen Puffer befindenden Zeilen ausgeführt, die anderen beiden bleiben unbearbeitet, verbleiben aber im Puffer. Beim nächsten Zyklus werden nicht die nächsten vier Zeilen nachgeladen, sondern nur mehr zwei, wobei die bereits bearbeiteten Zeilen überschrieben werden. Abbildung 5.11 zeigt den Ablauf der Kantendetektion und des Nachladens der Bildzeilen. Die roten Pfeile zeigen an auf welche beiden Zeilen der Algorithmus zur Kantendetektion momentan arbeitet. Gelb hinterlegt sind jene Zeilen die aus dem vorhergehenden Zyklus im Puffer verblieben sind, grün hinterlegt jene die mittels DMA nachgeladen wurden. Stellvertretend werden die ersten vier Zyklen dargestellt.

Abbildung 5.11: Schema des Nachladens der Bildzeilen aus dem SDRAM

Dieses Verfahren des Nachladens neuer Bildzeilen bringt den Vorteil, dass keine Zeilen zwischengespeichert werden müssen. Eine Alternative wäre ein Zirkularpuffer. Der Nachteil eines solchen Puffers ist allerdings die aufwendigere Synchronisation der DMA-Transfers und der Speicherzugriffe der Software. Die Anzahl der insgesamt notwendigen DMA-Zyklen zur Bearbeitung eines Frames verdoppelt sich dadurch.

Farbklassifizierung:

In derselben Schleife in dem nach Abbildung 5.11 die Kanten zweier Bildzeilen ermittelt werden, und auch die Farbraumkonvertierung erfolgt, wird das entsprechende Pixel hinsichtlich seiner Farbe klassifiziert. Es werden dazu die beiden Kanäle H und S aus dem HSI Farbraum herangezogen (siehe Abschnitt 5.9.2). Durch empirische Versuche müssen die Schwellwerte für beide Kanäle pro zu detektierender Farbe ermittelt werden. Für jeden Kanal wird ein unterer und oberer Schwellwert festgelegt. Liegen die Werte für H und S innerhalb dieser Schwellwerte so wird dieser Pixel als zu der Farbe gehörend klassifiziert. Durch die ausschließliche Verwendung der Cromakanäle zur Farbklassifizierung ist die Empfindlichkeit gegenüber Helligkeitsschwankungen geringer. Trotzdem schlägt sich eine zu starke oder zu schwache Beleuchtung auch in den Farbkanälen nieder, besonders bei stark reflektierenden Oberflächen. Abbildung 5.12 zeigt das Ergebnis einer Segmentierung mittels einer Klassifizierung nach Farben.

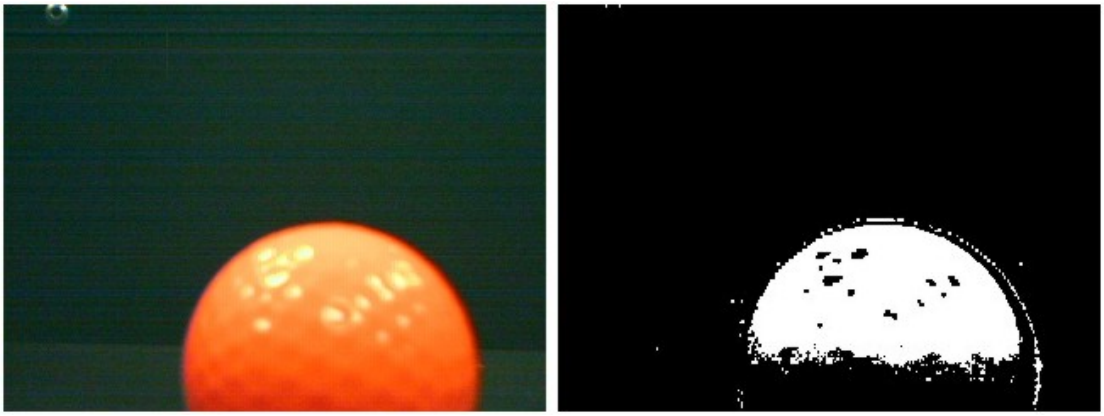

Abbildung 5.12: Ergebnis einer Segmentierung durch Farbklassifizierung

Kodierung der Ergebnisse im Speicher:

Als Ergebnisse der beiden oben stehenden Operationen erhalten wir einerseits die Information ob dieses Pixel Teil einer Kante ist, und andererseits ob seine Farbe zu einer bekannten Farbklasse gehört, und wenn ja zu welcher. Im Fall der Kanteninformation würde ein Bit zur Kodierung der Information reichen, Kantenpixel ja oder nein. Im Fall der Farbklassifizierung, da maximal 16 Farben unterschieden werden können, bedarf es 4 Bit für die Kodierung der Farbklassenzugehörigkeit. Beide Informationen lassen sich zu einem Byte kombinieren. Somit beträgt der erforderliche Speicherplatz für das Abspeichern des Kanten- und Farbbildes ein Byte pro Pixel. In Abbildung 5.13 ist die Kodierung dargestellt.

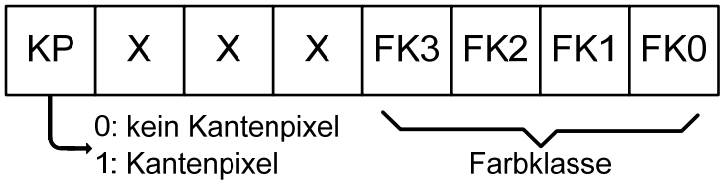

Abbildung 5.13: Kodierung der Ergebnisse der Segmentierung

5.9.4 Nachbearbeitung der Segmente

Die gefundenen Segmente nach der Farbklassifizierung weisen oft vereinzelte verstreute Punkte auf bzw. Lücken in eigentlich durchgehenden Flächen. Diese würden durch ein Opening bzw. Closing (siehe Abschnitt 3.4.3), angewendet nach der Segmentierung, vermindert werden. Der Aufwand ist aber in Hinblik auf die zur Verfügung stehende Rechenzeit zu groß und trägt nur marginal zu einer Verbesserung der Erkennung bei.

5.9.5 Das „Short Line Detection“ Konzept

Ein wichtiger Schritt für die Erkennung des Balles und die Bestimmung seiner Position ist die Erkennung kreisförmiger Elemente im Bild, da der Ball aus jeder Perspektive im Bild als Kreis erscheint, sofern sich Verzerrungen aufgrund des Objektivs im Rahmen halten. Wie im folgenden Abschnitt 5.9.6 erläutert wird, sind kurze horizontale Geradenstücke kennzeichnend für die Oberkante von kreisförmigen Gebilden. Somit kann man die Suche des Balles auf die Umgebung solcher kurzer Geradenstücke in der Farbe des Balles beschränken. Zu diesem Zweck werden mit der Farbklassifizierung auch gleich Geradenstücke in der Ballfarbe detektiert, deren Länge im Rahmen zweier empirisch ermittelter Schwellwerte bleibt. Die Grenzwerte für die minimale und maximale Länge der relevanten Geradenstücke lassen sich leicht durch Probeaufnahmen ermitteln, wobei der Ball einmal auf die maximale noch zu detektierende Entfernung gelegt und im anderen Fall direkt am Roboter positioniert wird. Die ermittelten Geradenstücke werden in einer eigenen Tabelle

im internen Speicher abgelegt. Bei der anschließenden Bestimmung, ob es sich bei dem Segment tatsächlich um einen Kreis handelt und wenn ja die Ermittlung von Radius und Mittelpunkt, spielen die kurzen Geradenstücke eine wichtige Rolle, da sie die potentielle Oberkante des Balles darstellen, wie Abbildung 5.14 zeigt (siehe Abschnitt 5.9.6).

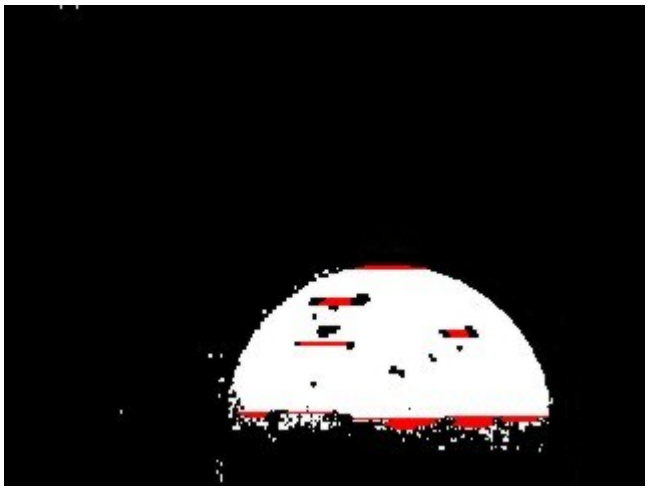

Abbildung 5.14: Balloberkante mit eingefärbten "kurzen Geradenstücken"

Aufgrund der englischen Bezeichnung „short line" für „kurze Gerade" nennen wir den Vorgang „Short Line Detection" (SLD) [MNO04].

Die SLD wird sowohl für die Cromakanäle für die Farbklasse des Balles bzw. des Kreises, als auch im Kantenbild durchgeführt. Das heißt, man erhält als Ergebnis sowohl kurze Kantenstücke als auch kurze Geradenstücke in der interessanten Farbe. In den Farbsegmenten sind somit nur jene Geradenstücke von Bedeutung, die sich in der Umgebung eines kurzen Kantenstücks befinden. Diese müssen aber nicht immer direkt übereinander liegen, da sich der Farbverlauf zum Rand hin oft sehr stark ändert, und die im Kantenbild erscheinenden Ränder von Objekten deshalb manchmal ein paar Pixel vom Rand des dazugehörenden Farbsegmentes entfernt sind. Dieser Effekt hängt sehr stark von der Technologie des Sensors ab und ist bei auf Bayerpattern basierenden Sensoren, aufgrund der geringeren räumlichen Auflösung der Farbinformationen und bei CMOS-Sensoren, aufgrund der geringeren Apertur der Pixel, größer (siehe Abschnitt 4.3.2).

5.9.6 Die Modellierung und Objektbestimmung

Nach der Segmentierung haben wir als Ergebnisse folgende Komponenten im SDRAM:

- Kantenbild des gesamten Frames.
- Bild der Farbklassen des gesamten Frames.
- Tabelle mit kurzen Geradenstücken, sowohl in den interessanten Farbklassen, als auch im Kantenbild.

Zur Bestimmung und Detektion von Objekten nach der Schicht „Objektidentifikation“ aus dem Schichtenmodell in Abbildung 3.1, sind Modelle erforderlich anhand derer die gesuchten Objekte mit Hilfe der oben aufgelisteten Elemente identifiziert werden können.

Modellierung des Balles:

Die Vorteile bei der Modellierung des Balles liegen auf der Hand. Ein Ball ist in einer zweidimensionalen Abbildung stets ein Kreis. Die Modellierung des Balles kann also auf die Modellierung eines Kreises reduziert werden. Ein Kreis ist mathematisch durch Radius und Mittelpunkt vollständig beschrieben. Für das Modell des Kreises ist es zunächst einmal wichtig, dessen Eigenschaften zu erfassen:

- Konstante Krümmung über den gesamten Rand.
- Die Oberkante ist begrenzt von kurzen Geradenstücken.
- Farb- und Helligkeitsverlauf sind aufgrund von Abschattungen nicht über die gesamte Oberfläche konstant.

Besonderes Augenmerk bei der Modellierung wird dabei auf die Eigenschaft gelegt, dass die Oberkante von kurzen Geradenstücken begrenzt wird. Relevant für die weitere Untersuchung sind dabei nur Geradenstücke die zur Farbklasse des Balles gehören und sich in unmittelbarer Umgebung von kurzen Kantenstücken befinden, da die Balloberkante auch im Kantenbild durch kurze Geradenstücke repräsentiert wird. Der Mittelpunkt dieses Geradenstücks der Oberkante liefert bereits die horizontale Position des Kreismittelpunktes. Somit ist ein Parameter, nämlich die x-Position des Ballmittelpunktes bereits gefunden, sofern es sich bei dem untersuchten Segment tatsächlich um einen Kreis handelt. Ob dies der Fall ist, wird überprüft in dem ausgehend von diesem Geradenstück, das als Oberkante des Balles angenommen wird, der Radius vergrößert wird, bis sich das ganze Farbsegment im inneren des Kreises befindet. Handelt es sich bei dem Segment tatsächlich um einen Kreis, ergibt dies eine charakteristische Verteilung der Randpixel des Kreises, die sich mit Pixel des Farbsegmentes decken. Abbildung 5.15 soll diesen Vorgang verdeutlichen.

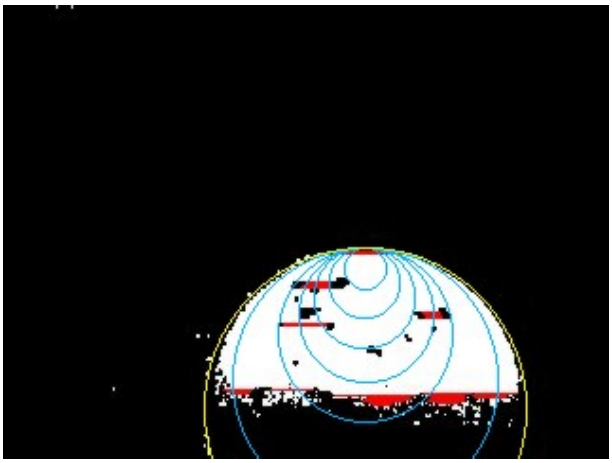

Abbildung 5.15: Modellierung von Kreisen

Wird durch dieses Testverfahren bestätigt, dass es sich um einen Kreis handelt, ergibt der erste Testpunkt, indem sich das gesamte Farbsegment innerhalb des Kreises befindet, die y-Koordinate des Mittelpunktes und den Radius, in Abbildung 5.15 symbolisiert durch den gelben Kreis.

Die eben beschriebene Modellierung des Kreises birgt auch gleich eine wichtige Einschränkung des Verfahrens. Um sinnvolle Ergebnisse zu erhalten, dürfen maximal 50 % des Balles durch andere Objekte verdeckt sein, anderenfalls wird die Genauigkeit der Erkennung schlagartig geringer.

Der große Vorteil dieses Verfahrens liegt in der geringen Ausführungszeit. Dadurch dass nur wenige Testpunkte notwendig sind, und bereits sofort nach der Vorverarbeitung des Bildes bereits ein möglicher Parameter des Balles bestimmt werden kann, ist der Aufwand sehr gering. Die Ausführungszeiten werden in Abschnitt 5.13.2 näher untersucht.

Modellierung anderer geometrischer Formen:

Die Modellierung anderer geometrischer Formen wurde im Rahmen dieser Diplomarbeit nur am Rand untersucht. Ein Beispiel ist die Detektion von Rechtecken, die allerdings einer perspektivischen Verzerrung unterworfen sind, die nicht berücksichtigt wird, da im Falle der Tore nur eine Richtung aber keine genaue Bestimmung von Nöten ist, zumal durch partielle Verdeckung wie es vor allem bei den Toren häufig vorkommt, die Messung ohnehin sehr ungenau ist. Rechteckige Farbsegmente werden charakterisiert durch die Koordinaten der vier Eckpunkte. Diese lassen sich sehr einfach ermitteln. Anhand der Differenz aus rechter oder linker, oberer Ecke und der dazu gehörenden unteren Ecke lässt sich eine ungefähre Entfernung der Kamera zur Fläche berechnen, sofern die tatsächliche Höhe bekannt ist. Die Höhe eignet sich aus dem Grund besser als die Differenz der x-Koordinaten sprich der Breite, da sie nicht von einer eventuellen partiellen Verdeckung betroffen ist.

Die vier Eckpunkte können sogar schon im Zuge der Farbklassifizierung mit abgespeichert werden, womit sich eine nachträgliche Suche erübrigt und sofort nach der Segmentierung eine Entfernungsbestimmung erfolgen kann.

5.9.7 Bildkoordinaten

Nach erfolgreicher Objektidentifikation werden Objekte nur mehr durch ihre Bildkoordinaten repräsentiert. Für Kreise sind dies die Koordinaten des Mittelpunktes und der Radius. Für Rechtecke die Koordinaten der vier Eckpunkte wie es Abbildung 5.16 zeigt. Eingezeichnet ist auch das zugrunde liegende karthesische Koordinatensystem selbst. Der Ursprung liegt in der linken oberen Ecke.

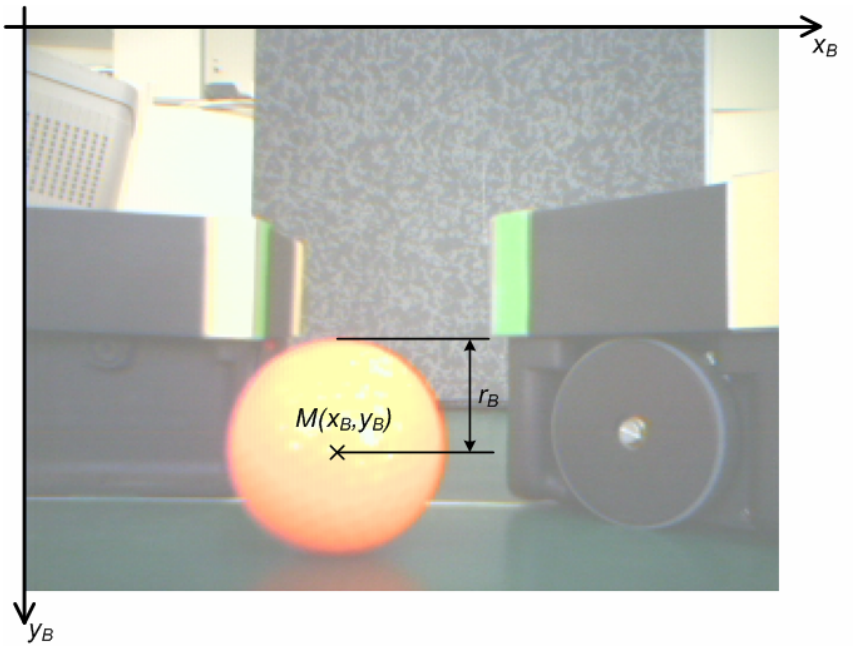

Abbildung 5.16: Bildkoordinatensystem

Diese Bildkoordinaten müssen nun in zweidimensionale Koordinaten des Weltbildes des Roboters transformiert werden.

5.10 Transformation der Bild- in Roboterkoordinaten

Als Ergebnis des bisher dargestellten Ablaufs der Objekterkennung erhalten wir *x*- und *y*-Koordinaten von Objekten im Bildkoordinatensystem aus Abbildung 5.16. Für die Steuerung des Roboters ist aber die Position der Objekte relativ zum Roboter selbst von Interesse. Aus diesem Grund ist es erforderlich die Bildkoordinaten in das Roboterkoordinatensystem (Abbildung 5.17) zu transformieren. Dabei stehen aber für unterschiedliche Objekte unterschiedliche Ausgangsgrößen zur Verfügung. Beim Ball sind dies der Radius und die x- und y- Koordinate des Mittelpunktes im Bild. Bei den Toren sind es die vier Eckpunkte. Zur Vereinfachung werden beide Objekte auf dieselben Ausgangsgrößen zurückgeführt. So wird aus den Eckpunkten der Tore der Flächenmittelpunkt und die Höhe bestimmt, wodurch wir auf dieselben Größen wie für den Ball kommen. Letztendlich bedeutet dies eine Abbildung dreier Ausgangsgrößen x_B, y_B und r_B bzw. h_B auf die zwei Größen x_R und y_R bzw. α_R und r_R.

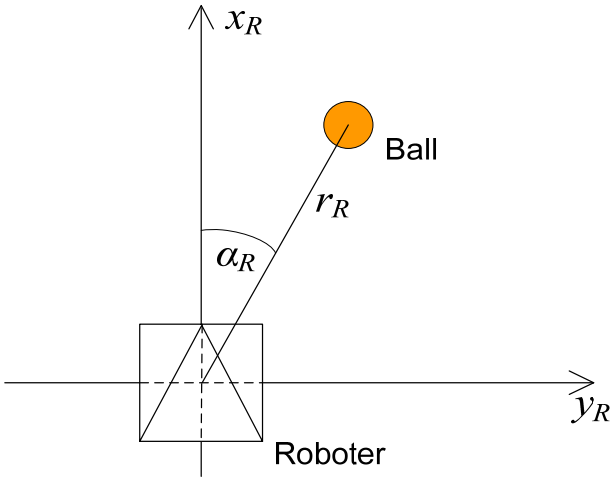

Abbildung 5.17: Roboterkoordinatensystem

Die Wahl der Ausgangsgrößen der Transformation ist dabei nur bezogen auf das Roboterfußballspiel interessant. Im Sinne anderer Anwendungen könnten auch andere Informationen interessant sein, etwa eine Vermessung der Objekte nicht nur in der Ebene sondern im Raum. In diesem Fall wären drei Ausgangsgrößen zu berechnen x, y und z.

5.10.1 Koordinatentransformation

Zu transformieren sind somit die drei Ausgangsgrößen x_B, y_B und r_B auf die zwei Größen x_R und y_R. Die Abbildungen Abbildung 5.18Abbildung 5.19 zeigen die Abhängigkeit der Roboterkoordinaten α_R und r_R in Abhängigkeit der Bildkoordinaten. Die Polarkoordinaten r_R und α_R können leicht auf die karthesischen x_R und y_R umgerechnet werden.

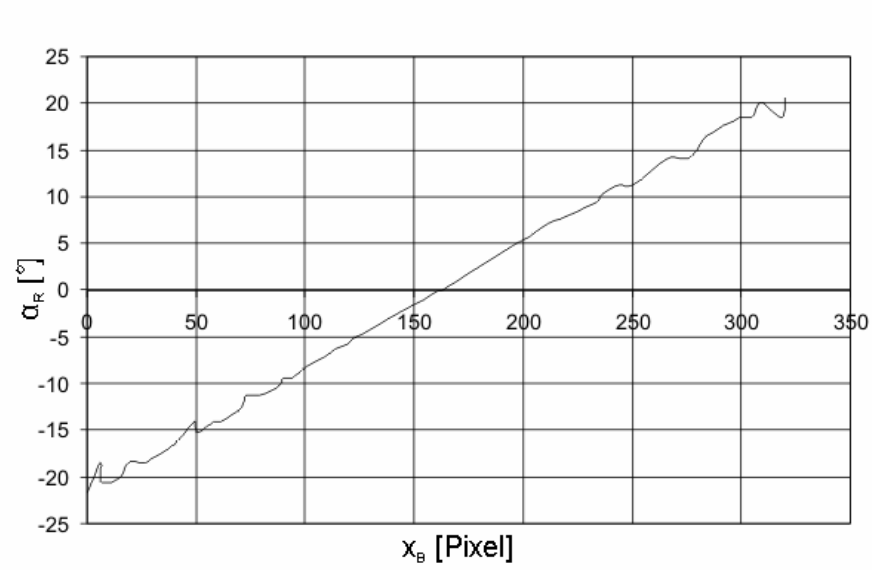

Abbildung 5.18: Winkel α_R in Abhängigkeit von x_B

Deutlich zu erkennen ist, dass sich der Winkel des Roboters zum Ball α_R sehr leicht durch die x-Koordinate des Ballmittelpunktes im Bild x_B ausdrücken lässt, da ein nahezu linearer Zusammenhang besteht. Der Abstand des Roboters r_R zum Ball drückt sich im Bild durch den Radius des Balles aus. Abbildung 5.19 zeigt die entsprechende Abhängigkeit. Die y-Koordinate des Ballmittelpunktes im Bild kann vernachlässigt werden, da sie kaum relevante Informationen enthält (siehe Abbildung 5.20). Der Abstand ließe sich zwar aus y_B berechnen, die Auflösung und damit die Genauigkeit wären aber sehr gering, ersichtlich am anfänglich steilen Abfall der Kurve r_R. Über den Winkel lassen sich überhaupt keinerlei Aussagen machen. Dies ist auf die speziell gewählte Montageposition der Kamera zurückzuführen.

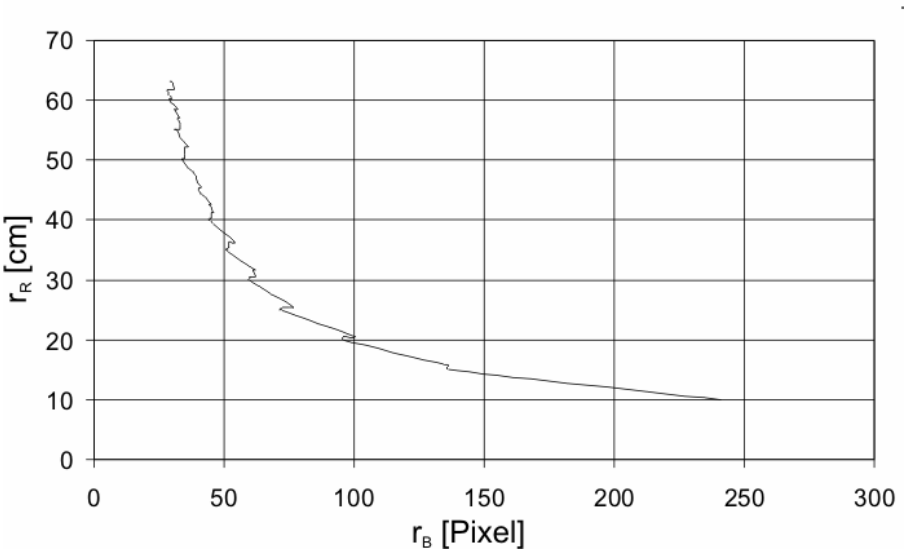

Abbildung 5.19: Ballabstand r_R in Abhängigkeit von r_B

Die Koordinatentransformation lässt sich somit auf zwei unabhängige funktionale Zusammenhänge reduzieren: $r_R = f(r_B)$ und $\alpha_R = f(x_B)$.

Die beiden Funktionen lassen sich aus den empirisch gewonnenen Werten aus Abbildung 5.18 und Abbildung 5.19 durch Interpolation gewinnen. Zwei mögliche Interpolationsverfahren werden in den folgenden beiden Abschnitten vorgestellt.

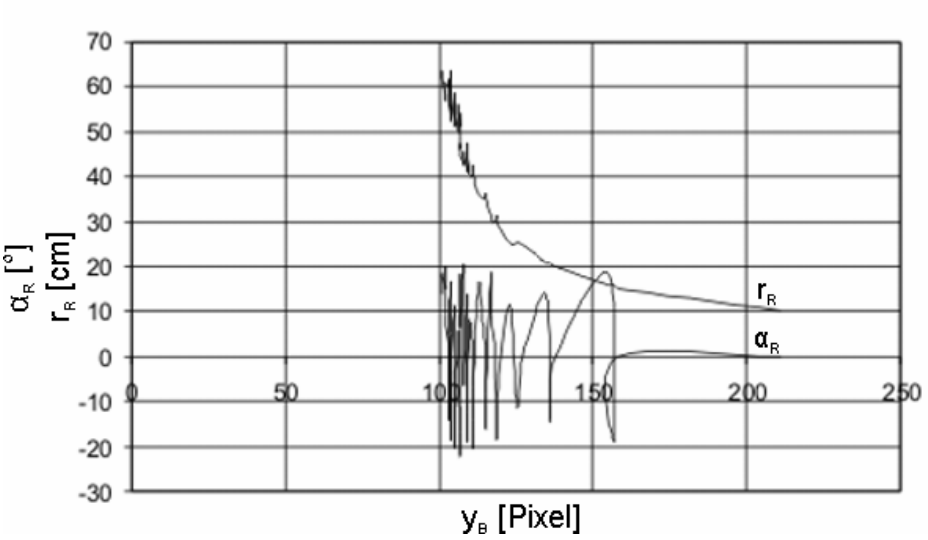

Abbildung 5.20: Ballabstand r_R und Winkel α_R in Abhängigkeit von y_B

Kamerakalibrierung:

Bereits in einem vorigen Abschnitt wurde der Punkt Kamerakalibrierung angesprochen. Dazu noch ein paar Anmerkungen: Verzerrungen im Linsensystem führen zu Versatzpunkten im Bild. Das bedeutet dass Bildpunkte im aufgenommenen Bild nicht an der Stelle erscheinen an dem sie nach den Gesetzen der Optik erscheinen sollten. Diese Nichtlinearitäten werden im Allgemeinen durch die Aufnahme eines Bildes mit markierten Testpunkten und der daraus folgenden Ermittlung einer Transformationsmatrix von den tatsächlichen Positionen im Bild auf die Sollpositionen umgerechnet. Dies hat für jedes Bild zu erfolgen. Da im vorliegenden Projekt die Transformationsgleichungen zur Ermittlung der Roboterkoordinaten aus den Bildkoordinaten empirisch ermittelt worden sind, ist die Korrektur der Nichtlinearitäten des Kameramoduls in diesen Gleichungen implizit enthalten. Eine laufende Korekktur der Bildkoordinaten der gesuchten Objekte vor einer Transformation in Roboterkoordinaten entfällt. Dies hat aber zur Folge, dass die Transformationsgleichungen für jede Kamera eigens ermittelt werden sollten, eine Tatsache die aber ohnehin für alle Formen der Kamerakalibrierung gilt.

5.10.2 Testaufbau

Die emprische Ermittlung der Werte aus obigen Diagrammen erfolgte mittels dem Testaufbau aus Abbildung 5.21. Der Ball wurde dabei im gesamten sichtbaren Bereich des Kameramoduls um jeweils 5 cm weiterbewegt und ein Foto geschossen. Anschließend wurde aus den Bildern der Radius und Mittelpunkt des Balles bestimmt.

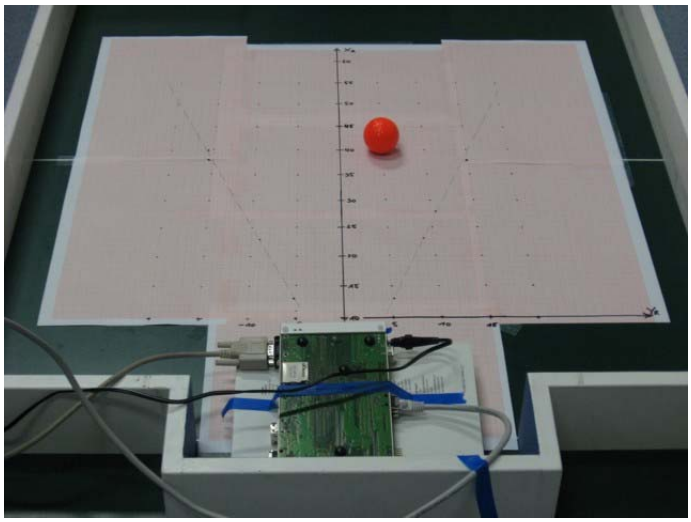

Abbildung 5.21: Testaufbau zur Bestimmung der Transformationsgleichungen

Aufgrund der leichteren Handhabung erfolgten die Messungen nicht mit dem Roboter direkt, sondern mit einer Entwicklungsplatine die so positioniert wurde, dass die Höhe der Kamera exakt mit der Einbauhöhe im Roboter übereinstimmt.

5.10.3 Methode der kleinsten Quadrate

Mit Hilfe der Methode der kleinsten Quadrate (engl. *„least square“*) kann eine Funktion ermittelt werden, die eine Menge von empirisch gewonnenen Punkten in einem Diagramm möglichst gut approximiert. Man unterscheidet grundsätzlich zwischen einer linearen und einer nichtlinearen Modellfunktion, wobei in letzterem Fall meistens ein Polynom n-ten Grades zu Grunde gelegt wird. Eine Darstellung der Methode der kleinsten Quadrate findet sich in [OPF02] oder in [BSM01].

Wendet man dieses Verfahren mit einem linearen Ansatz auf die Werte aus Abbildung 5.18 an, so ergibt sich folgendes:

Als Ansatz dient die Funktion $\alpha_R = a \cdot x_B + b$ mit den beiden zu bestimmenden Parametern *a* und *b*. Im vorliegenden Fall ergeben sich für *a*: 0,144 und für *b*: – 22,4. Damit lässt sich der Winkel des Roboters zum Ball aus der x-Koordinate des Ballmittelpunktes im Bild wie folgt berechnen:

$$\alpha_R = 0{,}144 \cdot x_B - 22{,}4\,. \tag{5.11}$$

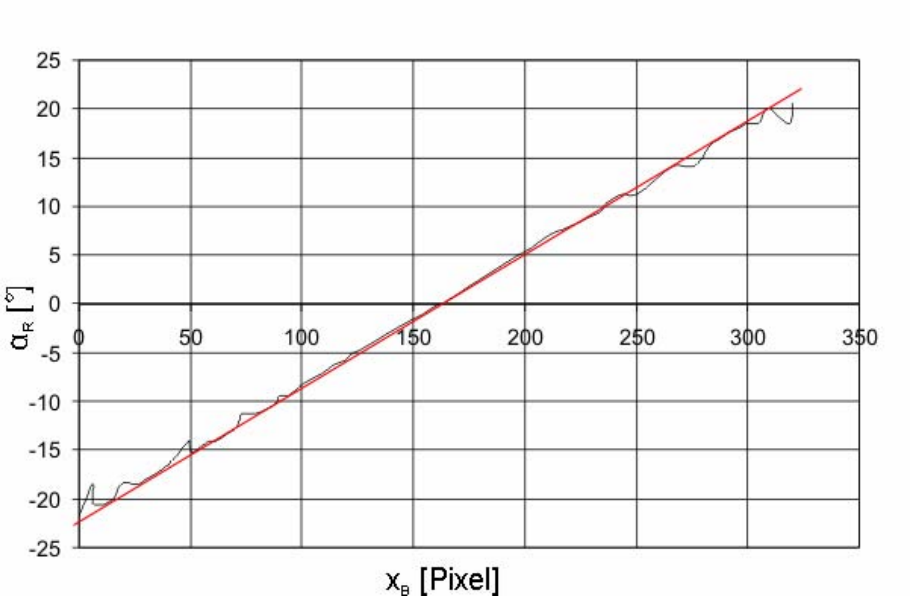

Abbildung 5.22: Mathematische Interpolation mittels *„least square“*

Die rote Linie in der Abbildung 5.22 zeigt die grafische Repräsentation der Gleichung (5.11). Die reale schwarze Kurve wird dabei sehr gut angenähert. Nur in den Randbereichen ergeben sich Ungenauigkeiten. Diese sind auf die Verzerrungen des Linsensystems an den Rändern des Objektivs zurückzuführen.

5.10.4 Stückweise lineare Approximation

Wie in obigen Abschnitt erwähnt, ließen sich durch einen Ansatz mittels einer Polynomfunktion n-ten Grades auch nicht annähernd lineare Verteilungen mittels der Methode der kleinsten Quadrate approximieren. Für die Verteilung aus Abbildung 5.19 wird aber eine Näherung mittels einer stückweise stetigen Funktion gewählt. Die Transformation des Abstandes lässt sich damit wieder auf die einfache Berechnung einer linearen Funktion zurückführen, ohne die Notwendigkeit eine komplexe Polynomfunktion berechnen zu müssen. Dies verkürzt die Ausführungszeit. Der empirisch ermittelte Kurvenverlauf wird dazu in 8 Abschnitte unterteilt, die jeweils durch eine Gerade angenähert werden wie Abbildung 5.23 zeigt. Die beiden Geradenparameter *a* und *b* wurden dabei direkt aus dem Diagramm aus Abbildung 5.19 heraus gelesen.

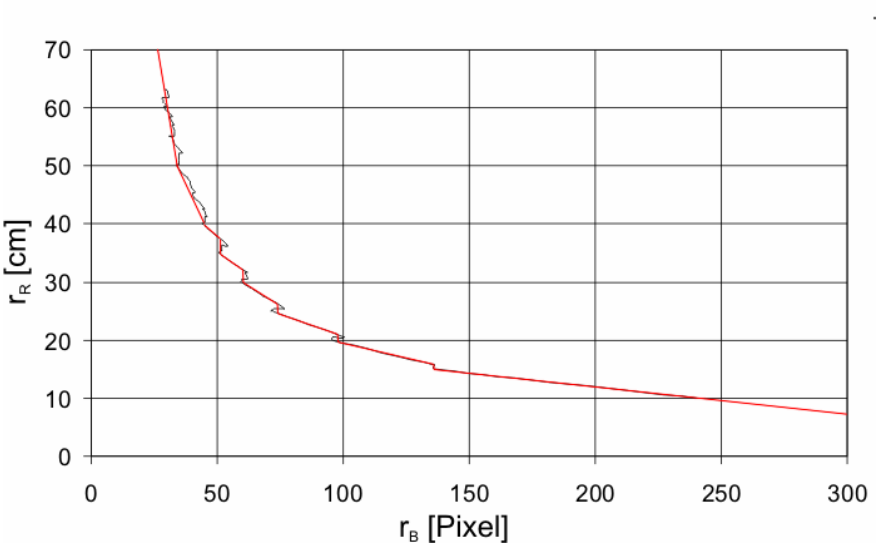

Abbildung 5.23: Stückweise lineare Approximation

Die nachfolgende Tabelle zeigt die den 8 Abschnitten entsprechenden Gleichungen.

Tabelle 5.2: Transformationsfunktionen für den Ballabstand

Bereich (r_B in [Pixel])	**Funktion**
0 - 35	$r_R = -1{,}66 \cdot r_B + 130$
36 - 45	$r_R = -r_B + 90$
46 - 51	$r_R = -0{,}3125 \cdot r_B + 56$
52 - 61	$r_R = -0{,}298 \cdot r_B + 50$
62 - 75	$r_R = -0{,}2623 \cdot r_B + 46$
76 - 99	$r_R = -0{,}16 \cdot r_B + 36$
100 - 135	$r_R = -0{,}11 \cdot r_B + 31$
ab 136	$r_R = -0{,}0498 \cdot r_B + 22$

In den vorhergehenden Abschnitten wurde stets darauf geachtet Fließkommaoperationen zu vermeiden, da dies zu Lasten der Ausführungsgeschwindigkeit geht. Man beachte aber, dass obige Operationen nur einmal pro Frame ausgeführt werden. Dies bedeutet zwei bis vier Fließkommaoperationen pro zu transformierenden Punkt eines Objekts. Weshalb es nicht ins Gewicht fällt.

5.11 Ablauf und Timing des Algorithmus

Den gesamten Ablauf der Objekterkennung für ein Frame zeigt Abbildung 5.24. Die gelb hinterlegten Methoden sind jene die in Assembler ausgeführt wurden (siehe Abschnitt 5.12). Dieser Funktionsblock enthält die Farbklassifizierung, die Kantendetektion und die SLD. Nicht enthalten ist die Farbraumkonvertierung vom YUV in das HSI Format. Eine Analyse des Timings hat gezeigt, dass die Transformationsgleichungen aufgrund der dafür notwendigen Divisionen zuviel Zeit benötigen und somit eine Framerate von 60 fps nicht mehr möglich wäre. Die Konvertierung allein würde für das gesamte Frame 18 ms benötigen. Aus diesem Grund wird sie für sehr hohe Frameraten nicht verwendet und die Farbklassifizierung stattdessen auf die Cromakanäle des YUV-Formates durchgeführt.

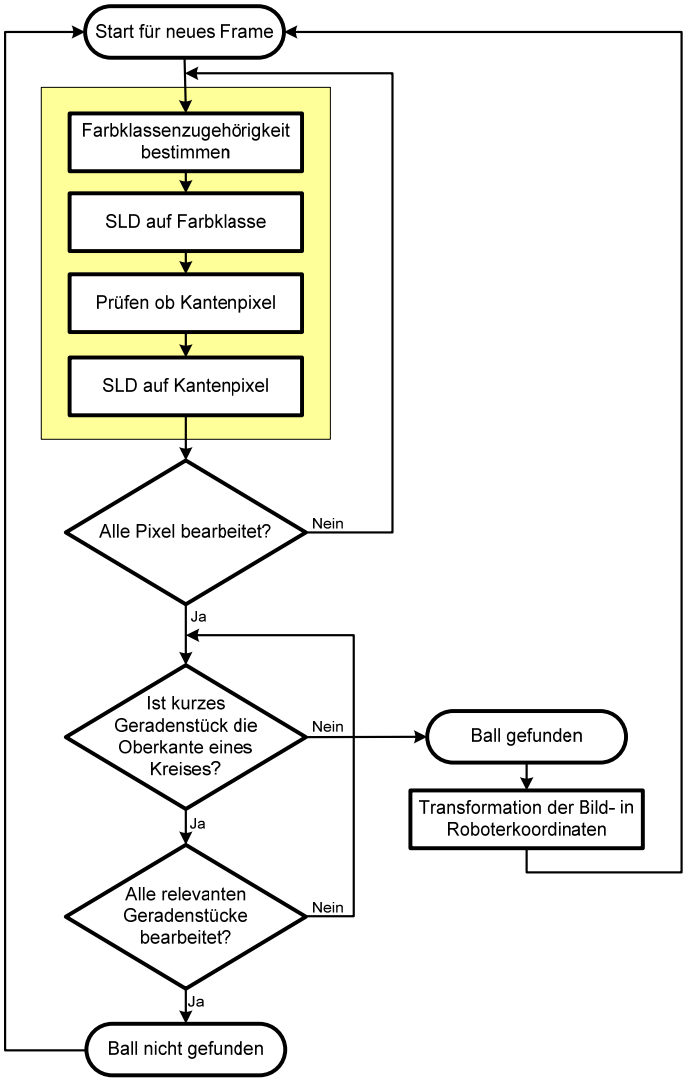

Abbildung 5.24: Ablauf der Bilderkennung für ein Frame

Die Abarbeitung des obigen Ablaufs benötigt für ein Frame zwischen 13 und 15 ms, je nach Lage und Anzahl der gefundenen kurzen Geradenstücke und dem Balldurchmesser im Bild. Das dem Ablauf zugrunde liegende Timing ist das VSYNC-Signal (Framesynchronisationssignal) des Kameramoduls. Dieses bestimmt den Datentransfer der Bildaten von der Kamera in das SDRAM und somit das Timing des gesamten Algorithmus. Die Framerate beträgt hier 67,7 fps. Dies ist auf die Verwendung eines Oszillators von 27 Mhz für die Taktung des Kameramoduls zurückzuführen womit sich die Framerate etwas erhöht. Besonders deutlich sieht man im Timing-Diagramm aus Abbildung 5.25, dass die Kantendetektion, Farbklassifizierung und SLD bereits während die Bilddaten in das SDRAM kopiert werden, erfolgt. Gestartet wird der Vorgang unmittelbar nachdem die ersten 4 Bildzeilen mittels des PPI-DMA kopiert wurden.

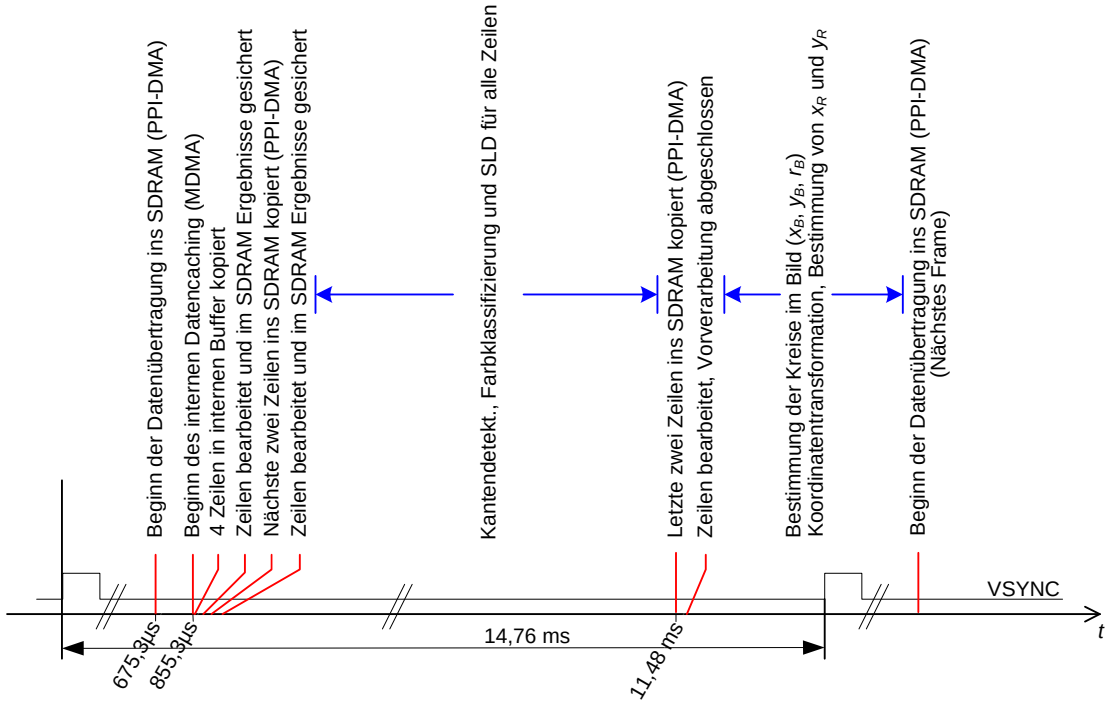

Abbildung 5.25: Timing der Bilderkennung

Zum besseren Verständnis des Diagramms ist in Abbildung 5.26 das Kopieren zweier Zeilen ins SDRAM über die PPI-Schnittstelle, deren Caching im internen Speicher und die Abarbeitung sprich Kantendetektion und Farbklassifizierung detailierter aufgezeichnet. Man beachte, dass die Kantendetektion, die Farbklassifizierung und die SLD zusammen ungefähr halb so viel Zeit in Anspruch nehmen, wie das Kopieren der Bildzeilen von der Kamera in das SDRAM. Diese hohe Effizienz des Algorithmus ist nur durch die Verwendung von Assembler als Programmiersprache möglich (siehe Abschnitt 5.12). Deutlich erkennbar ist auch der parallele Ablauf. Zwei Zeilen werden in das SDRAM kopiert, während gleichzeitig die zwei vorhergehenden Zeilen in das interne SRAM kopiert, bearbeitet und die Ergebnisse wiederum mittels DMA in das SDRAM zurück geschrieben werden.

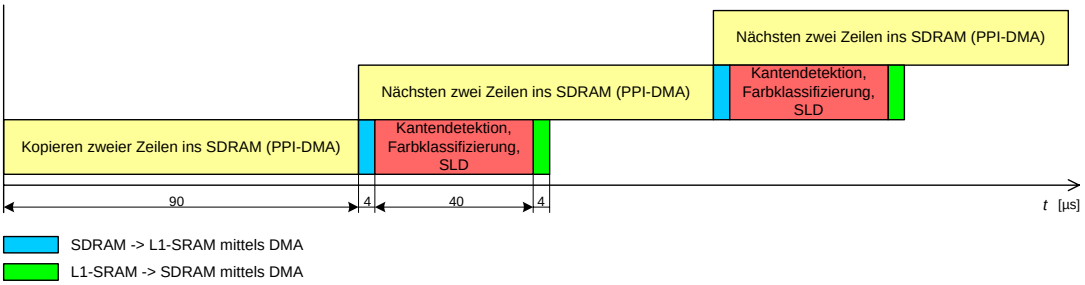

Abbildung 5.26: Detailiertes Timing einiger Algorithmusteile

5.12 Performance optimiertes Kodieren

Die Programmiersprache für den Algorithmus der Bilderkennung is C und Assembler. C++ war für das vorliegende Projekt nie ein Thema, da keine Vorteile ersichtlich sind und es im Gegenteil nur Nachteile bei den Ausführungszeiten gibt. Die Frage ist also welche Teile sollen in Assembler und welche in C geschrieben werden? C bietet natürlich den Vorteil, dass Algorithmen sehr einfach zu implementieren sind, wogegen sogar einfache Berechnungen in Assembler schon eines sehr hohen Aufwandes bedürfen. Außerdem werden Assembler Programme sehr schnell unübersichtlich und sind nur schwer wartbar. Dafür bietet besonders der Assembler des Blackfin sehr mächtige Befehle wie etwa Hardwareschleifen ohne den Overhead von Zählschleifen oder die Möglichkeit zur parallelen Ausführung von bis zu zwei MAC-Operationen (*„multiply accumulate operations“*) mit einer zusätzlichen Lade- oder Speicheroperation. Die meisten solcher Assemblerbefehle werden vom C-Kompiler nicht unterstützt. Zudem unterscheiden sich einfachste Algorithmen sehr stark in ihrer Ausführungszeit je nachdem ob sie in C oder in Assembler geschrieben worden sind. Tabelle 5.3 gibt einen Vergleich der Ausführungszeiten eines Farbraumkonvertierungsalgorithmus einmal in C und einmal in Assembler geschrieben, ausgeführt auf einem Blackfin BF533 mit 600 Mhz Corefrequenz. Konvertiert wurde eine Bild mit QVGA-Auflösung vom YUV 4:2:2 in den RGB 24 Bit Farbraum. Beidesmal mit demselben Datensatz inklusive Daten-Caching, das heißt, keine Zugriffe auf das SDRAM.

Tabelle 5.3: Vergleich C und Assembler

2 Zeilen QVGA von YUV 4:2:2 in RGB 24 Bit	
Assembler	3,1 ms
C	16,6 ms

Die Gegenüberstellung zeigt, dass im Durchschnitt mit einem in Assembler geschriebenen Algorithmus ein Geschwindigkeitsvorteil von bis zu einem Faktor 5 gegenüber denselben in

C erreichbar ist. Dies fällt besonders bei häufig ausgeführten Operationen wie etwa Pixel-Manipulationen ins Gewicht. Aus diesem Grund wurden die gelb hinterlegten Methoden aus dem Ablaufdiagramm von Abbildung 5.24 in Assembler geschrieben. Ruft man sich das Diagramm aus Abbildung 5.26 in Erinnerung so erkennt man, dass bei einer Verwendung von C als Programmiersprache für die rot hinterlegten Arbeitsschritte sich die Ausführungszeit nach obiger Feststellung auf ca. 200µs erhöhen würde. Damit wäre es nicht mehr möglich, die Verarbeitung der Daten parallel zum Kopieren in das SDRAM durchzuführen. Die Folge wäre, dass Frames verloren gehen.

Für die Farbklassifikation, die Kantendetektion und die SLD wurde soweit möglich die Parallelisierung von Assembler-Befehlen genutzt, sowie die Möglichkeit der Hardware-Schleifen und der automatisierten zirkulären Datenpufferung.

Optimierungen in C:

Auch die in der Programmiersprache C geschriebenen Teile wurden weitestgehend im Sinne einer Senkung der Ausführungszeit optimiert. Es gibt eine Reihe von Untersuchungen und Veröffentlichungen zum Thema optimiertes Kodieren für die Sprache C, weshalb auf Optimierungen des C-Programmcodes nicht näher eingegangen werden soll.

5.13 Evaluierung der Performance

Im Folgenden soll nun untersucht werden inwieweit die Resultate der Objekterkennung den Anforderungen entsprechen. Es sollen die Ausführungszeit und die Genauigkeit gemessen und deren Gültigkeitsbereich untersucht werden.

5.13.1 Genauigkeit

Es ist schwierig generelle Aussagen über die Genauigkeit zu machen, da sie von vielen Faktoren abhängig ist. In besonderem Maße natürlich von der Art und der Position der Beleuchtung. Es wurde zwar versucht, die Erkennung weitgehend unabhängig vom Umgebungslicht zu machen aber besonders beim Golfball ist dies durch seine spiegelnde Oberfläche nur eingeschränkt möglich. Bei günstigen Bedingungen, sprich einer nahezu homogenen Beleuchtung von oben, erreicht man im Schärfebereich des Fokus des Kameramoduls durchaus Genauigkeiten von +/- 1 mm. Dies liegt bereits unterhalb der Genauigkeit der Positionierung des Balles am Raster. Verwendet wurde für die Messungen der Testaufbau aus Abbildung 5.21.

Für eine Positionierung des Balles entlang der x_R-Achse des Roboterkoordinatensystems ergeben sich die Werte aus Abbildung 5.27. Der Ball wurde dabei jeweils um 5 cm weiter bewegt und das Ergebnis, das die Erkennung liefert, aufgezeichnet.

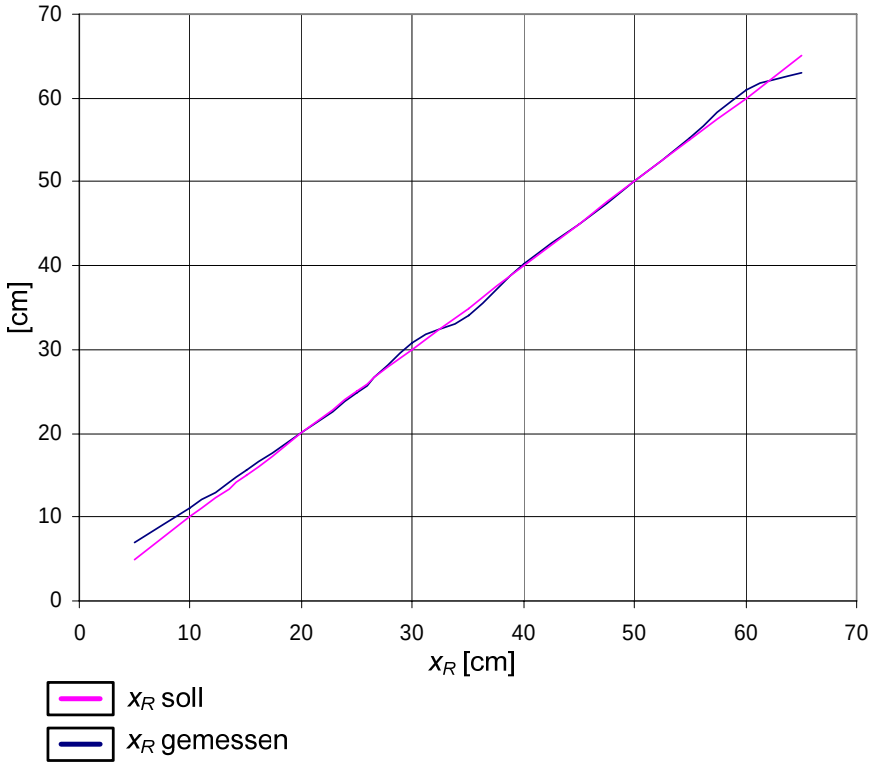

Abbildung 5.27: Genauigkeit der Messung von x_R

Den geringsten Fehler haben wir im Bereich zwichen 40 und 50 cm. Dort liegt die Genauigkeit bei den erwähnten +/- 1 mm. In diesem Bereich liegt auch der Fokus. Zu den Randbereichen der Erkennung hin nimmt der Fehler zu. Im Nahbereich liegt dies zum einen natürlich daran, dass der Ball die Kamera zum großteil verdeckt und damit nur mehr ein kleiner Ausschnitt des Balles zur Bestimmung des Radius herangezogen werden kann zum anderen aber auch daran, dass durch die Nähe des Balles zur Kamera die Oberkante im Bild sehr unscharf und damit der Farbverlauf verfälscht wird (siehe Abbildung 5.28). Es fällt weniger Licht in die Kamera wodurch die Einstellungen der AEC und AWB stark beeinflusst werden, was wiederum zu Farbverfälschungen führt. Auch im Kantenbild ist keine durchgehende Kante mehr detektierbar.

Im Fernbereich der Kamera, ab 60 cm Entfernung ist der Radius des Balles im Bild schon sehr klein; die Auflösung der Entfernungsbestimmung nimmt damit ab. Das heißt, der Durchmesser des Balles im Bild verkleinert sich im Verhältnis immer weniger mit zunehmender Entfernung (siehe Abschnitt 5.7.2). Dies erklärt den Knick der blauen Kurve am Ende des Diagramms.

Die Messungen zur Genauigkeit wurden idealerweise unter denselben Bedingungen durchgeführt wie jene zur Bestimmung der Transformationsgleichungen (siehe Abschnitt 5.10.2).

Abbildung 5.28: Ball im Nahbereich der Kamera

Die Genauigkeit der y_R-Koordinate ist hingegen nicht solchen Schwankungen unterworfen. Sie liegt über dem gesamten sichbaren Bereich bei +/- 1 mm. Nur in den Randbereichen, in denen nicht mehr der gesamte Ball im sichtbaren Bereich der Kamera liegt, verschlechtert sich die Genauigkeit sehr rasch.

5.13.2 Ausführungszeit

Die Ausführungszeiten der einzelnen Teile des Algorithmus können aus den Timing-Diagrammen der Abbildungen Abbildung 5.25 und Abbildung 5.26 abgelesen werden, weshalb hier nicht näher darauf eingegangen wird. Unter der Voraussetzung, dass die Farbraumkonvertierung in das HSI-Format nicht durchgeführt wird, was nur zu marginalen Einbusen in der Genauigkeit führt, kann eine Framerate von 67,7 fps erreicht werden (siehe Abschnitt 5.11). Damit wird die geforderte Framerate aus Abschnitt 1.2 von 50 fps sogar überschritten.

5.13.3 Leistungsaufnahme

Die durchschnittliche Leistungsaufnahme des Prozessormoduls mit dem Kameramodul beträgt 1,5 Watt während der Ausführung des Algorithmus. Gemessen wurde dieser Wert bei einer Prozessortakt-Frequenz von 600 Mhz und einem SDRAM-Takt von 120 Mhz. Die Core-Spannung betrug 1,25V. Dies sind die maximal zulässigen Werte für die verwendete Version des Blackfin BF533. Ein Vergleich mit Tabelle 1.1 zeigt, dass auch in Puncto Leistungsaufnahme die Anforderungen erfüllt wurden.

5.14 Erkenntnisse und Problembereiche

Autobuffermodus:

Bei manchen Kameramodellen kann es vorkommen, dass die automatische Datenübertragung mittels DMA über die PPI-Schnittstelle zu einem Verlust der Synchronisation führt, was zu einem „Durchlaufen“ des Bildes führt. In diesem Fall muss die Datenübertragung nach jedem Frame per Software gestoppt und gestartet werden.

Erkennung an den Rändern des Sichtwinkels:

Befindet sich der Ball an den Rändern des für die Kamera sichtbaren Bereiches, so kann es zu starken Schwankungen in den Ergebnissen der Positionsbestimmung kommen. Der Grund dafür ist die teilweise Verdeckung des Balles über die Hälfte des Balldurchmessers hinaus, wodurch sich das Ergebnis der Berechnung des Radius schlagartig von einem Frame zum nächsten ändern kann. Dasselbe gilt für die partielle Verdeckung durch andere Objekte.

spontane Änderung der Lichtverhältnisse:

Bei schneller Änderung der Lichtverhältnisse braucht die AEC und AWB einige Zeit bis sie die Werte angepasst hat. Aus diesem Grunde kann es im Falle von schnellen Helligkeitsänderungen für die Dauer der Anpassung an die neuen Lichtverhältnissen zu Fehlerkennungen führen.

Die Objekterkennung mittels Bildverarbeitung ist ein sehr weit verbreitetes Gebiet. Für Problemstellungen in diesem Bereich gibt es keine Patentlösungen. Geforderte Genauigkeiten und Bildwiederholraten können nur erreicht werden, wenn das System genau auf die Gegebenheiten angepasst ist.

Kapitel 6 Weitere Entwicklungen

Die Entwicklung am Tinyphoon insgesamt und auch am Bildverarbeitungssystem schreitet stetig voran. In diesem Kapitel soll nun ein kurzer Ausblick auf zukünftige oder bereits laufende Entwicklungen gegeben werden mit dem Fokus auf das bildverarbeitende System des Roboters. Dies bedeutet sowohl die Entwicklung neuer Konzepte als auch die Verbesserung des bestehenden Monokamerasystems.

6.1 FPGAs

Die ständige Weiterentwicklung im Bereich der Prozessoren für Embedded Systems ermöglicht die Integration von immer mehr Rechenleistung. Dies erlaubt auch die Implementierung von komplexeren Algorithmen. Werden FPGAs im Hinblick ihrer Leistungsaufnahme optimiert, wäre in Zukunft die Verwendung eines solchen für eine etwaige Bildvorverarbeitung denkbar.

6.2 Drehbarer Stereokopf

Mit einem Monokamerasystem ist eine Vermessung, bzw. Entfernungsbestimmung nur bei Objekten bekannter Abmessungen möglich, sollen auch Objekte unbekannter Größe detektiert und die Entfernung zu denselben bestimmt werden, ist ein Stereokamerasystem unumgänglich. Dies bringt vor allem bei der Erkennung von beweglichen Objekten für eine etwaige Kollisionsdetektion (engl. *„collision detection“*) erhebliche Vorteile. Die Entfernung zu Objekten wird bei einem Stereosystem aufgrund der Parallaxe in den Bildern der linken und rechten Kamera ermittelt. Die beiden Kameras sollen auf einem drehbaren Kopf montiert werden, um eine Rundumsicht zu ermöglichen. Angedacht sind dafür Antriebe mit Schritt- oder Servomotoren. Servomotoren haben zwar einen eingeschränkten Stellwinkel, können aber durch Umbau auf einen Rundumlauf erweitert werden.

Die Entfernungsbestimmung und Objektdetektion anhand der Parallaxe ist um einiges aufwendiger als das im Rahmen dieser Arbeit vorgestellte. Aus diesem Grund wird die Entwicklung einer neuen Hardwarearchitektur auf Kosten der Leistungsaufnahme von Nöten sein.

6.3 Neue Prozessoren

Es werden von den einschlägigen Firmen ständig neue Prozessoren auf dem Markt gebracht, mit oft auch speziell für die Bildverarbeitung entwickelten Einheiten. Auch Analog Devices bringt neue Varianten des Blackfin auf den Markt, die besonders im Bereich der Bildverarbeitung für den automotiv Bereich zugeschnitten werden. Diese sind natürlich auch für den Roboter interessant.

Besonders interessant im Hinblick auf die Entwicklung des Stereokopfes ist der Blackfin ADSP-BF561 von Analog Devices. Es handelt sich dabei um die Dualcore-Variante des ADSP-BF533 mit getrenntem Code- und L1-Data-RAM, aber einem gemeinsamen 128 kByte großen L2-RAM. Das SDRAM und die restliche Peripherie werden gemeinsam genutzt. Der BF561 besitzt zwei PPI-Interfaces, womit direkt zwei Kameramodule angeschlossen werden können. Somit kann das Kamerabild einer Kamera von einem Core und jenes der zweiten Kamera vom zweiten Core ausgewertet werden. Dies ist interessant sowohl für die Berechnung der Parallaxe als auch für eine zweifache Ausführung des Monokamerasystems und mit der damit verbundenen Möglichkeit zur Verdoppelung der Framerate (siehe Abschnitt 6.6.1).

6.4 Neue Kameramodule

Eine stetig weiterlaufende Entwicklung gibt es auch bei den Kameramodulen, so werden Kameras in der Größe der im Rahmen dieser Arbeit verwendeten, mit hochauflösenden CMOS-Chips bis zu einigen Megapixeln entwickelt. Größere Auflösungen können unter Umständen die Genauigkeit der Erkennung erhöhen.

6.5 Autofokus

Die Diskussion über Autofokus ja oder nein wurde bereits im Abschnitt 4.3.1 ausführlich geführt, zugunsten von Objektiven mit Fixfokus. Die Firma Phillips hat ein ein Linsensystem vorgestellt mit einer flüssigen Linse für die Brennweiteneinstellung, ähnlich dem Verfahren das auch das menschliche Auge anwendet um die Schärfeneinstellung vor zu nehmen. Die flüssige Linse verändert dabei ihren Brechungsindex mittels Anlegen eines elektrischen

Feldes. [INR04] Ein solches Linsensystem würde die zwei wesentlichen Punkte die gegen die Verwendung eines Autofokussystems sprechen, widerlegen. Zum einen die benötigte Baugröße. Philips strebt eine Verwendung in Handys und PDAs an womit sie im Rahmen der Baugröße für den Roboter wären und zum zweiten die erforderliche Geschwindigkeit. Da mechanische Bauelemente entfallen und „nur“ elektrische Felder zur Steuerung vewendet werden, ist die Nachstellzeit erheblich kürzer als bei herkömmlichen Autofokussystemen. Eine Verwendung im Vision System des Tinyphoon könnte somit durchaus in Betracht gezogen werden. Zum Zeitpunkt der Erstellung dieser Arbeit waren jedoch noch keine Systeme mit flüssiger Linse verfügbar.

6.6 Verbesserungen des Monokamerasystems

Einige mögliche Ansatzpunkte zur Verbesserung des Algorithmus zur Monobilderkennung sind im folgenden Abschnitt beschrieben.

6.6.1 Verdoppelung der Framerate

Wie in Abschnitt 6.3 kurz angedeutet, besteht die Möglichkeit die Software für das Monokamerasystem auf je einem Core des Blackfin BF561 auszuführen, an dem zwei Kameramodule angeschlossen sind, die dieselbe Szene. Jeder Core bearbeitet dabei das Bild einer Kamera. Durch eine Synchronisation der beiden Kameras, die einerseits über den integrierten Powerdown-Pin oder über das Steuerinterface SCCB erfolgen kann, wäre es denkbar die Aufnahme der Bilder um die Hälfte eines Frames zu verschieben, wie es Abbildung 6.1 zeigt. Das heißt, die zweite Kamera beginnt mit der Aufnahme sobald die erste Kamera die Hälfte des Frames ausgelesen hat. Dadurch würde eine theoretische Verdoppelung der Framerate erreicht.

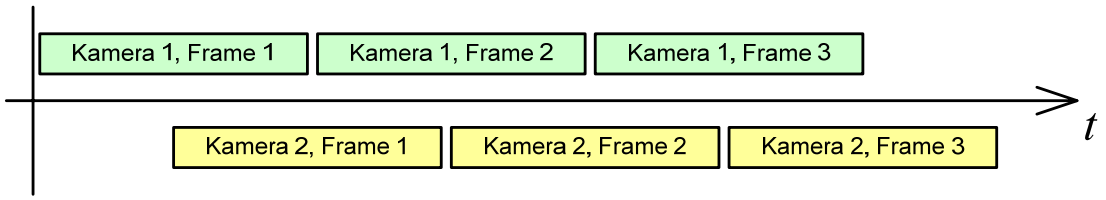

Abbildung 6.1: Synchronisation zweier Kameras

6.6.2 Verringerung der Empfindlichkeit gegen Verdeckung

Um die Einschränkung zu umgehen, dass 50 % der Balloberkante sichtbar sein müssen um eine Zuverlässige Detektion zu ermöglichen, wäre es denkbar die Shortline Detection auch auf vertikale Kanten auszudehnen um somit nicht nur die Ball ober- sondern auch die linke und rechte Kante des Balles detektieren zu können.

6.6.3 Verbesserte Objektdetektion

Stehen in Zukunft noch leistungsstärkere Prozessoren zur Verfügung wäre es möglich die Detektion von Kreissegmenten im Kantenbild auch mittels einer Houghtransformation durchzuführen, was möglicherweise eine Erhöhung der Erkennungsrate mit sich bringen würde.

6.6.4 Weniger Empfindlichkeit gegenüber Helligkeitsänderungen

Ein großes Problem für jeden Bildverarbeitungsalgorithmus stellt nach wie vor die Beleuchtung dar. Durch die spiegelnde Oberfläche des Balles wirkt dieses Problem im vorliegenden Fall noch viel gravierender. Die Durchschnittshelligkeit eines jeden Frame wird bereits mitberechnet. Eine „manuelle“ Einstellung der Belichtungszeit ist aber noch nicht implementiert, diese würde aber einige Vorteile bringen. Derzeit wird die Belichtungszeit automatisch von der Kamera eingestellt (AEC siehe Abschnitt 5.7.1). Eine adaptive Einstellung je nach Umgebungsbeleuchtung wäre von Vorteil. Vielleicht kann ein verbesserter Algorithmus für die Exposure Control im Blackfin implementiert werden, der die Probleme der Beleuchtung verringert und somit eine Erkennung bei einer größeren Bandbreite an Beleuchtungsvarianten gewährleistet und den Algorithmus noch unempfindlicher gegen Beleuchtungsschwankungen macht.

6.6.5 Verringerung der Ausführungszeit

Die Kamera liefert die Bilddaten derzeit im YUV-Farbraum. Für die Kantenerkennung erfordert dies keine Konvertierung. Diese kann direkt auf den Y-Kanal durchgeführt werden. Für die Farbklassifizierung wird der YUV-Raum über den RGB-Raum in den HSI-Raum konvertiert. Die Kamera ist in der Lage die Daten im RGB565-Format zu liefern (16 Bit RGB). Dies würde den Umweg bei der Transformation in den HSI-Raum ersparen. Für die Kantendetektion wäre dann ein Helligkeitskanal zu ermitteln. Dies könnte der Mittelwert aus R, G und B, der Maximalwert oder eine gewichtetete Summe aus RGB sein (siehe Abschnitt 2.3.6). Auch ein allein aus den G-Werten gebildetes Kantenbild wäre denkbar, da dieser Farbton zu ca. 60 % zum Helligkeitsempfinden des menschlichen Auges beiträgt (siehe Abschnitt 2.3). Inwieweit dies eine Verringerung der Ausführungszeit mit sich bringt, gilt es zu untersuchen.

6.7 Intelligenter Bildverarbeitungssensor

Für zukünftige Anwendungen werden „intelligente“ Bildverarbeitungssensoren immer größere Bedeutung erlangen. Das Monokamerasystem könnte im Sinne eines solchen intelligenten Bildverarbeitungssensors weiterentwickelt werden. In diesem Falle wäre die

Erweiterung der Erkennung auf andere geometrische Grundformen notwendig. Aus diesen Grundformen könnten dann komplexere Formen zusammengesetzt werden um somit Modelle zu bilden, die die Erkennung komplexerer Strukturen ermöglichen. Eine noch intensivere Kombination von Farb- und Kantenerkennung wäre dafür wünschenswert. Mit der Integration eines Steuerbusses könnte der Sensor dann für die jeweilige Aufgabe konfiguriert werden.

Bildverarbeitungssysteme sind derzeit sehr genau auf die jeweilige Aufgabe zugeschnitten, um alle Möglichkeiten zur Optimierung ausschöpfen zu können. Die Software ist dann auch meistens auf die jeweiligen Umgebungsbedingungen, für die sie entwickelt wurde, beschränkt. Bis zur eben geschilderten Vision eines universell einsetzbaren Sensors zur Objekterkennung ist es deshalb noch ein weiter Weg.

Abbildungsverzeichnis

Tabellenverzeichnis

Literatur- und Quellenverzeichnis

[ADS06] Datenblatt zu Analog Devices Blackfin ADSP-BF533, Rev. C, Analog Devices, Mai 2006

[ADVD02] Datenblatt zu Analog Devices ADV7183, Videodecoder with 10 bit ADC, Rev 0, Analog Devices, 2002

[BSM01] I. N. Bronstein ; K. A. Semendjajew ; G. Musiol ; H. Mühlig, *"Taschenbuch der Mathematik"*, Deutsch Verlag: ISBN 3-8171-2005-2 , Thun 2001

[BVN96] Bevan James: *„Der menschliche Körper: Anatomie und Physiologie"*, 1. Auflage: Verl. Gesundheit: ISBN 3-333-00752-5, Berlin 1996.

[BBS91] H.Bässmann, Ph. W. Besslich: *„Bildverarbeitung Ad Oculus"*, Springer Verlag, 1991

[DBD04] Dietmar Bruckner, *„Plattform zur Bildbearbeitung autonomer Kleinstroboter"*, Diplomarbeit, TU-Wien, 2004

[DGV04] Deutscher Golf Verband EV Wiesbaden: *„Offizielle Golfregeln 2004-2007"*: Albrecht Golf Verlag Gmbh: ISBN 3-87014-186-7, Juli 2004

[DSW02] C. Demant, B. Streicher-Abel, P. Waszkewitz: *„Industrielle Bildverarbeitung, Wie optische Qualitätskontrolle wirklich funktioniert"*, ISBN 3-540-41977-2, Springer Verlag, 2002

[FBS86] David S. Falk; Dieter R. Brill; David G. Stork: *"Seeing the light: Optics in nature, photography, color, vision an holography"*: ISBN 0-471-60385-6, Chichester: Wiley 1986

[GLW81] Neal C. J.R. Gallagher, Gary L. Wise: *„A Theoretical Analysis of the Properties of Median Filters"* in *„IEEE Transactions on Acoustics, Speech, and Signal Processing"*, ASSP-29(6), Seite 466-471, Dezember 1981

[HBL98] D. H. Hubel: *"Auge und Gehirn"* in *"Neurobiologie des Sehens"*, Spektrum der Wissenschaft, Heidelberg 1989

[HBR91] Prof. Peter Haberäcker: *„Digitale Bildverarbeitung, Grundlagen und anwendungen"*: Carl Hanser Verlag: ISBN 3-446-16339, München Wien, 1991

[HOU62] P. V. C. Hough, *„Methods and Means for Recognizing Complex Patterns“*, U.S. Patent 3069654, 1962

[HRM06] Analog Devices, *„ADSP-BF533 Blackfin Processore Hardware Reference Revision 3.2“*, Juli 2006

[DRS92] Hans Jörg Dirschmit: *„Mathematische Grundlagen der Elektrotechnik“*: ISBN 3-528-33034-1, Vieweg Verlagsgesellschaft, Auflage 4, 1992

[GHR02] Daniel Göhring: *„Digitalkameratechnologien, Eine vergleichende Betrachtung, CCD kontra CMOS“*, Humboldt Universität Berlin

[HFR03] Heinz Haferkorn: *„Optik, physikalisch technische Grundlagen und Anwendungen“*: Wiley-VCH-Verlag: ISBN 3-527-40372-8, Weinheim, 2003

[ITU94] International Telecommunication Union (www.itu.ch): *“ITU-R Recommendation BT.601-4: Encoding parameters of digital television for studios“*, 1994

[JHG99] Bernd Jähne, Horst Haußecker, Peter Geißler: *„Handbook of computer vision and applications, Volume 1, Sensors and Imaging“*: Academic Press: ISBN 0-12-379770-5, 1999

[JHN91] Bernd Jähne: *„Digitale Bildverarbeitung“*: ISBN 3-540-53768-6, Springer Verlag, Berlin Heidelberg, 1991

[JSM02] M. Jamzad, B.S. Sadjad, V.S. Mirrokni, M. Kazemi, H. Chitsaz, A. Heydarnoori, M.T. Hajiaghai, E. Chiniforooshan, *“A Fast Vision System for Middle Size Robots in RoboCup“* aus den Proceedings zu Robocup 2001, Seite 71 bis 80, Springer Verlag, Berlin, Heidelberg, 2002

[KHL04] Dietrich Kühlke: *„Optik, Grundlagen und Anwendungen“*: Deutsch Verlag: ISBN 3-8171-1741-8, Frankfurt am Main, 2004

[KSC96] R. Klette, a. Koschan, K. Schlüns: *„Computer Vision, Räumliche Information aus digitalen Bildern“*: Friedr. Vieweg & Sohn VerlagsgesmbH: ISBN 3-528-06625-3, Braunschweig/Wiesbaden, 1996

[MCR05] Datenblatt des Micron MT9V131, Micron Technology, 2006

[MNO04] S. Mahlknecht, G. Novak, R. Oberhammer: „*A Real-time Image Recognition System for Tiny Autonomous Mobile Robots“*: aus den Proceedings zu IEEE Real-Time and Embedded Technology and Applications Symposium, Toronto, Mai 2004

[NTL03] Datenblat der Intel StrataFlashes 28FxxxJ3, Intel, Februar 2003

[NVK02] Novak Gregor: *“Multi Agent Systems Robot Soccer”*: Dissertation: TU-Wien, April 2002

[OVT04] Datenblatt des Omnivision OV7660FSG V1.1, Omnivision, September 2004

[OPF02] Opfer Gerhard, *„Numerische Mathematik für Anfänger“*, Vieweg Verlag:

ISBN 3-528-37265-6 , Braunschweig 2002

[RDG93] Prof. Bernd Radig (Hrsg.): *„Verarbeiten und Verstehen von Bildern“*: R. Oldenbourg Verlag: ISBN 3-486-20787-3, München Wien 1993

[SCC02] Omnivision *„Serial Camera Control Bus Functional Specification“,* Version 2.0, 2002

[STB88] Erich Steinböck: *„Die Extraktion der wesentlichen Strukturen aus einem pixelorientierten Grauwertbild als Vorbereitung zur Bilderkennung“*: Diplomarbeit: TU-Wien, September 1988

[STM06] Daniel Steinmair: *„Tinyphoon. Entwicklung einer Hardwareplattform für einen autonomen Fußballroboter“*: Diplomarbeit: TU-Wien, 2006

[STN93] R. Steinbrecher: *“Bildverarbeitung in der Praxis”*: R. Oldenbourg Verlag GmbH: ISBN 3-489-22372-0, München 1993

[TMS06] Datenblatt des Texas Instrument TMS320DM642 V 2006, Texas Instruments, Juli 2006

[VDU06] Analog Devices: *„VisualDSP++ 4.5 Users Guide Revision 2.0“*, April 2006

Internet:

[DMN06] Hersteller von Optiken: http://www.edmundoptics.com zuletzt abgerufen am 19.12.2006

[FIRA06] http://www.fira.net/soccer/mirosot/overview.html, zuletzt aufgerufen am 08.09.2006

[FIRR06] http://www.fira.net/soccer/mirosot/MiroSot.pdf, Regelwerk für das FIRA Mirsosot Spiel, zuletzt abgerufen am 18.12.2006

[INR04] http://www.innovations-report.de/html/berichte/verfahrenstechnologie/ bericht-26811.html, Informationen zur flüssigen Linse von Phillips, zuletzt abgerufen am 13.01.2007

[ITU656] http://www17.cds.ne.jp/~stray/ccir656.html, ITU656 Standard, zuletzt abgerufen am 06.09.2006

[ITUCCU04] ITU 2004: Entstehung des ITU656 Standards entnommen von http://www.itu.int/aboutitu/overview/history.html, abgerufen am 06.09.2006

[RBC06] http://www.robocup.org/overview/23.html (history robocup) (31.08.2006)

[SCL06] Hersteller von Optiken: http://www.schael-optik.de zuletzt abgerufen am 19.12.2006

[WAD06] Informationsseite von Analog Devices Inc: www.analog.com, zuletzt aufgerufen am 26.09.2006

[WCM01] Informationen über CMVision entnommen von http://www.cs.cmu.edu/~jbruce/cmvision/, zuletzt abgerufen am 15.10.2006

[WIC02] *„I^2C Bus Specification Version 2.1 Jannuary 2000“*, entnommen von http://www.esacademy.com/faq/i2c/I2C_Bus_specification.pdf, abgerufen am 23.09.2006

[WIN05] Informationen über OpenCV entnommen aus http://www.intel.com/technology/computing/opencv/overview.htm, zuletzt abgerufen am 15.10.2006

[WJF01] *„Biographie: Jean-Baptiste Joseph Baron de Fourier“*, entnommen von http://www.chemgapedia.de, abgerufen am 15.09.2006.

[WSC03] Informationen zum Icebear JTAG entnommen von http://section5.ch/icebear, abgerufen am 26.09.2006

[WWP04] „Erklärungen zu Betriebssystemen“ entnommen von http://de.wikipedia.org/wiki/Betriebssystem , abgerufen am 26.09.2006