



**FACHHOCHSCHUL-BACHELORSTUDIENGANG  
AUTOMATISIERUNGSTECHNIK-INIF**

---

**Anbindung und Adaptierung einer mikrocontrollerge-  
steuerten Kamera an einen autonomen Roboter**

**ALS BACHELORARBEIT EINGEREICHT**

**zur Erlangung des akademischen Grades**

**Bachelor of Science in Engineering**

**von**

**Bernhard Muckenhumer**

**04/2010**

---

Betreuung der Bachelorarbeit durch

Prof. (Fh) DI Walter Rokitansky

Ich erkläre ehrenwörtlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benutzt, die den benutzten Quellen entnommenen Stellen als solche kenntlich gemacht habe und dass diese Arbeit mit der vom Begutachter beurteilten Arbeit übereinstimmt.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

.....  
Bernhard Muckenhumer

Peuerbach, 03.04.2010

## KURZFASSUNG

Die Eurobot ist ein internationaler Roboterwettbewerb, der einmal im Jahr in Europa stattfindet. Dafür müssen die Teilnehmer eine vorgeschriebene Aufgabe mit einem autonomen Roboter lösen. Das Robo-Racing-Team<sup>1</sup> der Fh Wels nimmt seit dem Jahr 2007 aktiv an diesem Bewerb teil. Eurobot<sup>2</sup> wird auf der Homepage wie folgt beschrieben:

*„Der Event EUROBOT ist aus dem französischen Nationalen Wettbewerb entstanden. An EUROBOT können Teams aus allen Ländern der Welt teilnehmen, obwohl der Schwerpunkt und der Austragungsort des Wettbewerbs in Europa liegt. Es nehmen pro Land bis zu drei Teams teil, in den vergangenen Jahren ist die Anzahl der teilnehmenden Mannschaften auf circa 50 gewachsen. EUROBOT findet Ende Mai statt.“*

Für die Eurobot 2010 ist eine Hinderniserkennung zur Fahrtroutenplanung des Roboters erforderlich. Um eine Fahrtroutenplanung durchzuführen bevor sich der Roboter in Bewegung setzt, müssen die Positionen der Hindernisse erfasst werden. Dafür eignen sich stationäre Kameras, die jeweils einen bestimmten Bereich der Aktionsfläche erfassen können. Die möglichen Positionen an denen sich ein Hindernis befinden kann, sind bekannt und werden von der jeweiligen Kamera ausgewertet. Die Daten werden bei einer Anfrage an den Roboter gesendet. Um diese Forderungen zu erfüllen sind drei mikrocontrollergesteuerte Kameras erforderlich, die die nötigen Grundfunktionalitäten liefern und deren Software adaptiert werden kann.

Nachfolgende Arbeit beschäftigt sich mit der Erläuterung der Inbetriebnahme, Adaptierung und Anbindung des ausgewählten Kameratyps.

---

<sup>1</sup> [Robo-Racing-Team, 2010]

<sup>2</sup> [Austrobot, 2010]

# INHALTSVERZEICHNIS

1 MOTIVATION .....	1
2 KRITERIEN FÜR DIE KAMERAUSWAHL .....	2
2.1 Notwendige Funktionalität .....	2
2.2 Baugröße und Energieversorgung.....	3
2.3 Programmierbarkeit .....	4
2.4 Datenkommunikation .....	4
2.5 Wiederverwendung.....	4
3 CMUCAM3 CMOS KAMERASYSTEM.....	5
3.1 Bereitgestellte Kamerafunktionen .....	6
3.2 Hardware.....	7
3.2.1 Hardwarecharakteristik.....	7
3.3 Technische Daten.....	8
3.4 Systemvoraussetzungen zur Verbindung mit dem PC.....	9
3.5 Softwaretools .....	10
3.5.1 Cygwin .....	10
3.5.2 Philips Programmer .....	12
3.5.3 Framegrabber.....	13
3.6 Inbetriebnahme und Programmierung .....	13
3.6.1 Flashen des Philips LPC2106 Mikrocontrollers.....	14
3.6.2 Verbindung mit dem PC/Roboter.....	14
3.6.2 Steuerkommandos der CMUCAM3.....	15
3.7 Nachteile der CMUCAM3.....	16

4 ADAPTIERUNG DER SOFTWARE.....	17
4.1 Beispielprogramme der CMUCAM3.....	17
4.2 Anpassung und Erweiterung.....	18
4.2.1 Sichtbereiche für die Eurobot 2010.....	18
4.2.2 Vergleich der Farbbereiche RGB und YCrCb.....	19
4.2.3 Programmablaufdiagramm.....	20
4.2.4 Zustandsautomat.....	22
4.2.5 Datentransferprotokoll.....	23
4.3 Implementierung.....	25
5 ERSTELLUNG EINES FRAMEGRABBERS.....	26
5.1 Roboterfernsteuerung.....	27
5.2 Einbindung des Framegrabbers in die Fernsteuersoftware.....	28
6 ZUSAMMENFASSUNG UND AUSBLICK.....	29
7 LITERATUR.....	30
7.1 Bücher.....	30
7.2 Datenblätter der Kamera.....	30
7.3 Internetquellen.....	31
7.4 Relevante Bachelorarbeiten.....	31
8 ANHANG.....	32
8.1 C-Code der Messung des Farbmittelwertes.....	32
8.2 C-Code für das Setzen des virtuellen Bildbereichs.....	33
8.3 C-Code für das Erstellen des Datenpaketes.....	33
8.4 C-Code für das Senden des Datenpaketes.....	34

## ABBILDUNGSVERZEICHNIS

Abb. 1: Zugehörige Beacons zu den Spielerfarben.....	3
Abb. 2: Gesamtansicht der Aktionsfläche mit den Beacons .....	3
Abb. 3: Funkmodul ER400TRS von Easy Radio .....	4
Abb. 4: Kameramodul CMUCAM3.....	5
Abb. 5: Blockdiagramm des Hardwareaufbaus der CMUCAM3 .....	7
Abb. 6: Hardwarecharakteristik der CMUCAM3 .....	7
Abb. 7: Cygwin - Compileroberfläche.....	10
Abb. 8: Flash Tool für den LPC2106 Mikontroller .....	12
Abb. 9: Verbindung via serieller Schnittstelle .....	14
Abb. 10: Unterstützte Kommandos der CMUCAM3 .....	15
Abb. 11: Sichtbereiche der Kameras.....	18
Abb. 12: Bildaufnahme Kamera zwei im RGB Farbbereich .....	19
Abb. 13: Bildaufnahme der Kamera zwei im YCrCb Farbbereich.....	19
Abb. 14: Flussdiagramm Kamerasteuerung.....	20
Abb. 15: Flussdiagramm des Prozesses „Check and execute setting commands“ .....	21
Abb. 16: Zustandsautomat des Prozesses „Measure mean colour of positions“.....	22
Abb. 17: Oberfläche des programmierten Framegrabbers .....	28
Abb. 18: Funktion cmucam2_get_meanOfAllPositions .....	32
Abb. 19: Funktion set_positionOfElement .....	33
Abb. 20: Funktion cmucam2_fill_positions .....	33
Abb. 21: Funktion cmucam2_send_positions.....	34

# 1 MOTIVATION

Das Robo-Racing-Team hat bereits zahlreiche autonome Roboter für verschiedene Bewerbe entwickelt und dies erfolgreich eingesetzt. Eine Auswertung der Umgebungsverhältnisse durch ein einfaches und kompaktes Kamerasystem, das stationär oder direkt auf einem Roboter angebracht ist, kann dafür sehr nützlich sein.

Das erste Einsatzgebiet des Kamerasystems ist die Eurobot 2010. Es wird aber dennoch großer Wert darauf gelegt, die Möglichkeit zu schaffen, dieses Kamerasystem für mehrere Aufgaben in verschiedenen Robotersystemen auszulegen und damit eine Wiederverwendung sicherzustellen

## **2 KRITERIEN FÜR DIE KAMERAUSWAHL**

In autonomen Robotern werden meist Mikrocontroller für die Steuer- und Regelaufgaben eingesetzt, welche nur eine begrenzte Rechenleistung zur Verfügung stellen können. Die Kamera muss daher ein in sich geschlossenes System sein, um die Robotersteuerung zu entlasten. Für die Auswahl der Kamera wurden folgende Punkte in Betracht gezogen.

### **2.1 Notwendige Funktionalität**

Das Kamerasystem muss mindestens die folgenden Funktionen bereitstellen:

- Einfache Bildverarbeitungsaufgaben
- Stationäre Bildaufnahme und Pufferung
- Einstellung des aufzunehmenden Bildbereichs durch Koordinatenübergabe
- Möglichkeit zur zeilenweise Auswertung eines Bildes um hohe Verarbeitungsgeschwindigkeiten zu ermöglichen
- Über ein serielles Interface parametrierbare Eigenschaften
- Flexibilität durch einstellbaren Farbbereich

## 2.2 Baugröße und Energieversorgung

Da die Kameras für die Eurobot 2010 auf den Beacons<sup>3</sup> am Rand der Aktionsfläche platziert werden und dafür eine Abmessungsbeschränkung von 80x80x160 besteht, darf die Kamera inklusive Elektronik eine Abmessung von 60x60x60 nicht überschreiten. In Abb. 1 und Abb. 2 sind die Positionen der Beacons ersichtlich. Beacons sind Plattformen die sich am Rand der Aktionsfläche befinden. Auf diesen Plattformen kann jedes Team Geräte platzieren, um zum Beispiel eine Positionsbestimmung des eigenen oder des gegnerischen Roboters zu ermöglichen.

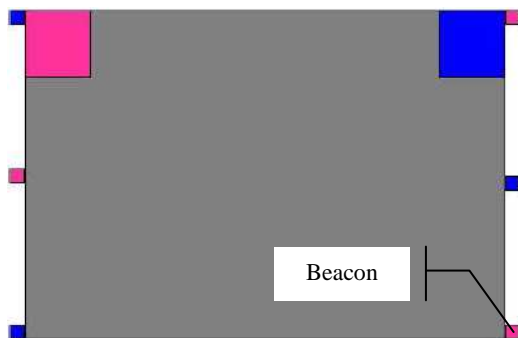


Abb. 1: Zugehörige Beacons zu den Spielerfarben

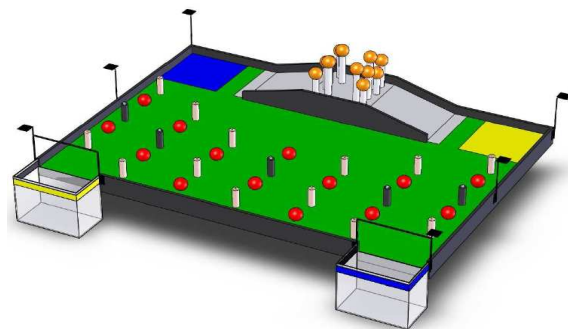


Abb. 2: Gesamtansicht der Aktionsfläche mit den Beacons

Zusätzlich zum Kamerasystem muss noch ein Funkmodul und die Elektronik für die Gegnererkennung in den Beacon-Aufsätzen integriert werden.

Die Energieversorgung der drei erlaubten Beacon-Aufsätze erfolgt durch eine Lipozelle, was bei der Auswahl der Kamera (Spannung, Stromaufnahme) ebenfalls berücksichtigt werden muss. Um die Eignung für zukünftige Einsätze nicht einzuschränken, sollte jedoch die Möglichkeit vorhanden sein, die Kamera softwaremäßig in einen „Sleep-Modus“ zu versetzen.

---

<sup>3</sup> [Regeln Eurobot, 2010, S7-8]

## 2.3 Programmierbarkeit

Um die Kamera auf spezifische Aufgaben anzupassen, muss der Programmcode zugänglich und durch entsprechende Tools programmiert und erweitert werden können. Vorzugsweise soll der Code mit der Programmiersprache C erstellt werden können.

## 2.4 Datenkommunikation

Da die Kameras auf die stationären Beacons aufgesetzt werden, erfolgt die Datenkommunikation zwischen den Kameras und dem autonomen Roboter via Funkmodul(Abb. 3).

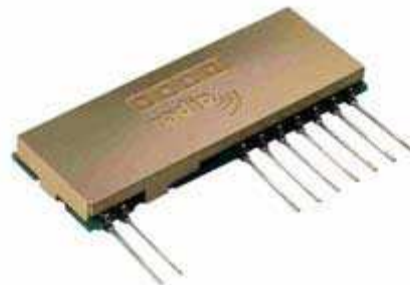


Abb. 3: Funkmodul ER400TRS<sup>4</sup> von Easy Radio

Das Funkmodul ER400TRS wird vom Robo-Racing-Team bereits eingesetzt und daher existieren bereits das nötige Wissen und die Hardware für dessen Verwendung. Als Sendeeingangsinterface verfügt es über ein RS232 Interface<sup>5</sup>. Die ausgewählte Kamera sollte ein solches standardisiertes Interface zur Verfügung stellen, da dieses ebenfalls für andere Bereiche und Aufgaben sehr einfach verwendet werden kann. Es verursacht somit keinen zusätzlichen Hardwareaufwand.

## 2.5 Wiederverwendung

Werden die vorangegangenen Punkte betreffend der Kameraauswahl erfüllt, so ist ein hohes Maß an Wiederverwendbarkeit gegeben. Durch einen Aufbau als Slave-Modul eines übergeordneten Systems kann die Kamera über das serielle Interface Befehle empfangen und diese ausführen. Diese können zum Beispiel ein Startbefehl zur Erfassung des Bildbereiches oder die Anforderung eines Datenpaketes sein. Durch eine spezifische Anpassung der Software können verschiedenste Aufgaben erfüllt werden.

---

<sup>4</sup> [Easy Radio, 2010]

<sup>5</sup> [Tietze, Schenk, 1999, S1115-1118]

### 3 CMUCAM3 CMOS KAMERASYSTEM

Da diese Arbeit eine Lösung für die Eurobot 2010 bereitstellen muss, wird auf eine zeitaufwendige Vergleichsstudie verzichtet. Es werden jedoch zwei in Frage kommende Kamerasysteme in Betracht gezogen:

- POB EYE II<sup>6</sup>
- CMUCAM3<sup>7</sup>

Aufgrund der Tatsache, dass die CMUCAM3 über weitaus mehr Funktionen verfügt als die POB EYE II, eine höhere Auflösung besitzt und preiswerter ist, wurde die CMUCAM3 ausgewählt. Die CMUCAM3 ist ein äußerst kompaktes und vielschichtiges Kamerasystem, welches mit ca. 178,5€ recht preiswert ist. Es handelt sich dabei um eine Entwicklung der Carnegie-Mellon-Universität<sup>8</sup> in Pittsburgh und erfüllt alle Anforderungen die im Vorfeld spezifiziert wurden. Die CMUCAM3 ist der Nachfolger der CMUCAM1 und CMUCAM2. Für die Datenverarbeitung wird der sehr leistungsfähige 32 Bit Mikrocontroller LPC2106<sup>9</sup> von Philips mit ARM7TDMI Kern verwendet. Die Bilddaten mit einer Auflösung von 352x288 werden mit dem Kameramodul CV3088 in den Framebuffer AL440B von Averlogic zwischengespeichert und können dann vom LPC2106 verarbeitet werden. In Abb. 4 ist die CMUCAM3 dargestellt.



Abb. 4: Kameramodul CMUCAM3

Es ist zu beachten, dass die CMUCAM3 nicht die Leistungsfähigkeit einer computergestützten Bildverarbeitung erreicht. Dies wird jedoch für diesen Anwendungsfall auch nicht erwartet und soll an dieser Stelle nur der vollständigen Information dienen.

---

<sup>6</sup> [Pob-Technology, 2010]

<sup>7</sup> [CMUCAM, 2010]

<sup>8</sup> [Carnegie Mellon University, 2010]

<sup>9</sup> [LPC2106, Datasheet, 2004, S8]

### 3.1 Bereitgestellte Kamerafunktionen

Folgende Funktionen werden durch die CMUCAM3 realisiert<sup>10</sup>:

- Verfolgen von benutzerdefinierten Farbobjekten mit bis zu 26 Bildern pro Sekunde
- Berechnung des Schwerpunktes eines Bildobjektes
- Ermitteln der durchschnittlichen Farbe und Varianz eines Bildbereichs
- Übermitteln eines Binärbildes der verfolgten Bildobjekte in Echtzeit (Line-Mode)
- Setzen beliebiger Bildausschnitte
- Einstellung der Bildparameter des Omnivision-Kameramoduls
- Übertragen eines Rohbildes mit Auflösung von 80x143 Pixel
- Schnittstellengeschwindigkeit: 115.200, 38.400, 19.200, 9.600 baud
- Möglichkeit der parallelen Bildverarbeitung mit einem untergeordnetem Prozessorboard über einen gemeinsamen Kamerabus
- Automatische Verfolgung einer Farbe mit Ansteuerung eines Modellbauservos
- Anschluss eines RC-Modellbauservos oder Nutzung des Ports als 1 Kanal Digital I/O
- Open Source für Windows und Linux
- MMC Flash Slot für die Aufnahme einer SD Speicherkarte für das Datalogging

---

<sup>10</sup> [Hardware, CMUCAM3\_datasheet, 2007, S1]

## 3.2 Hardware

Der Hardwareaufbau<sup>11</sup> wird in Abb. 5 beschrieben.

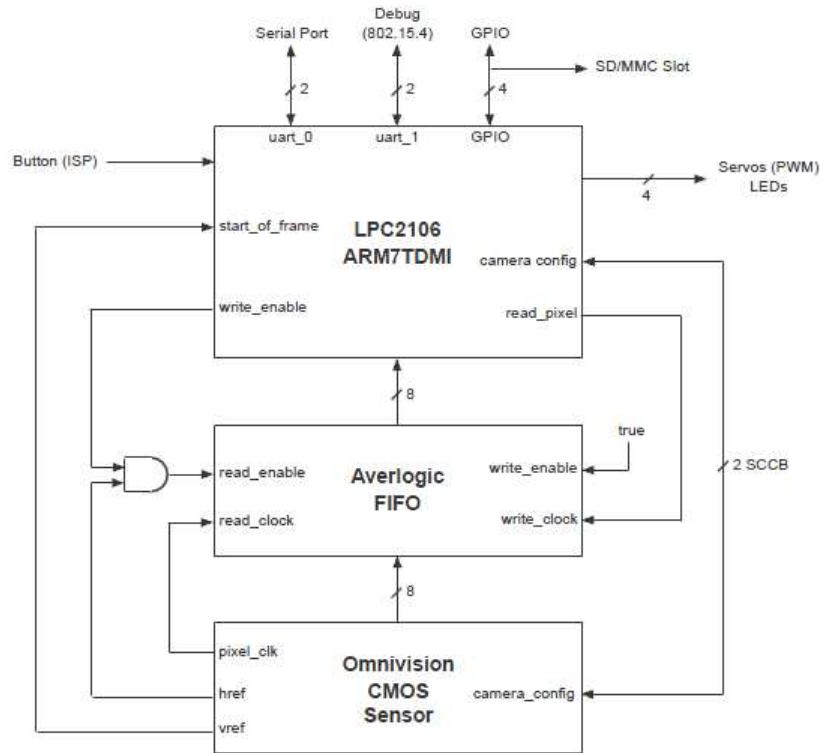


Abb. 5: Blockdiagramm des Hardwareaufbaus der CMUCAM3

### 3.2.1 Hardwarecharakteristik<sup>12</sup>

Abb. 6 zeigt die Ströme und Spannungen der einzelnen Kamerakomponenten im aktiven und inaktiven Zustand.

Power State	Active Current	Idle Current	Voltage
All Active	130mA	25mA	5 V
External Regulator Disabled	n/a	0.01uA	n/a
CPU (@60Mhz)	30mA	10uA	1.8 V
CPU Peripherals	30	10uA	3.3 V
CMOS camera	25mA	10uA	5 V
MAX232	8mA	n/a	3.3 V
FIFO	52mA	14mA	3.3 V
MMC card	4mA	4mA	3.3 V
misc	10mA	10mA	3.3 V

Abb. 6: Hardwarecharakteristik der CMUCAM3

<sup>11</sup> [Hardware, CMUCAM3\_datasheet, 2007, S3-5]

<sup>12</sup> [Hardware, CMUCAM3\_datasheet, 2007, S14]

### 3.3 Technische Daten<sup>13</sup>

- Auflösung: 352x288 (CIF)
- Spannungsversorgung: 5-15VDC
- Stromaufnahme: max. 200mA
- Mikrocontroller: LPC2106 mit ARM7TDMI Kern
- FIFO Buffer Speicher: AL440B von Averlogic
- Kamera: Omnivision CMOS C3088
- Interfaces: 2xRS232, SPI

Die für die Eurobot verwendeten CMUCAM3 Module werden vom Onlineanbieter Robotikhardware<sup>14</sup> bezogen.

---

<sup>13</sup> [Hardware, CMUCAM3\_datasheet, 2007, S14]

<sup>14</sup> [robotikhardware, 2010]

### 3.4 Systemvoraussetzungen zur Verbindung mit dem PC

Nach Versuchen mit der CMUCAM3 ergaben sich folgende Erkenntnisse:

- Wird die CMUCAM3 in Verbindung mit einem seriellen RS232 Comport betrieben, ist die Verwendung aller mitgelieferten Tools, wie zum Beispiel der Framegrabber<sup>15</sup> und der Programmiersoftware LPC Flash utility<sup>16</sup> mit dem Betriebssystem Windows XP professional problemlos möglich.
- Wird die CMUCAM3 in Verbindung mit einem RS232 zu USB Konverter betrieben, was einen virtuellen Comport zur Folge hat, treten einige Probleme bei der Inbetriebnahme der Tools auf. Da aber in den meisten Fällen bei Notebooks keine seriellen RS232 Anschlüsse vorhanden sind, muss dieses Problem gelöst werden. Im Folgenden ist daher eine Liste mit Herstellern angegeben mit deren Produkte die Verwendung laut Hersteller funktionieren sollte:
  - Airlink 101 - USB SERIAL Adapter (kein Erfolg mit WinVista)
  - I (dot)connect USB - SERIAL CABLE (kein Erfolg mit WinVista)
  - Belkin USB-to-Serial Portable Adapter (Erfolg mit WinVista)
  - ATEN UC232A USB-to-Serial-Adaptor (Erfolg mit WinXP)

Diese Angaben wurden im Onlineforum<sup>17</sup> des Herstellers veröffentlicht und wurden im Zuge dieser Arbeit nicht nachgewiesen. Für den Aufbau der Verbindung für diese Arbeit wird ein USB to Serial Converter der Firma Prolific verwendet. Als Betriebssystem dient Windows XP professional, das in einer virtuellen Maschine (VM Ware Workstation) unter Windows Vista Home Premium läuft.

Die Lösung dieses Problems hat sich als sehr zeitaufwendig und problematisch erwiesen und es besteht durchaus die Gefahr, dass oben angeführte Produkte nicht mit jedem Notebook zum Erfolg führen.

---

<sup>15</sup> [CMUCAM2GUI\_Overview, CMUCAM3\_sdk\_guide S19, 2010]

<sup>16</sup> [Hardware, CMUCAM3\_datasheet, 2007, S15-17]

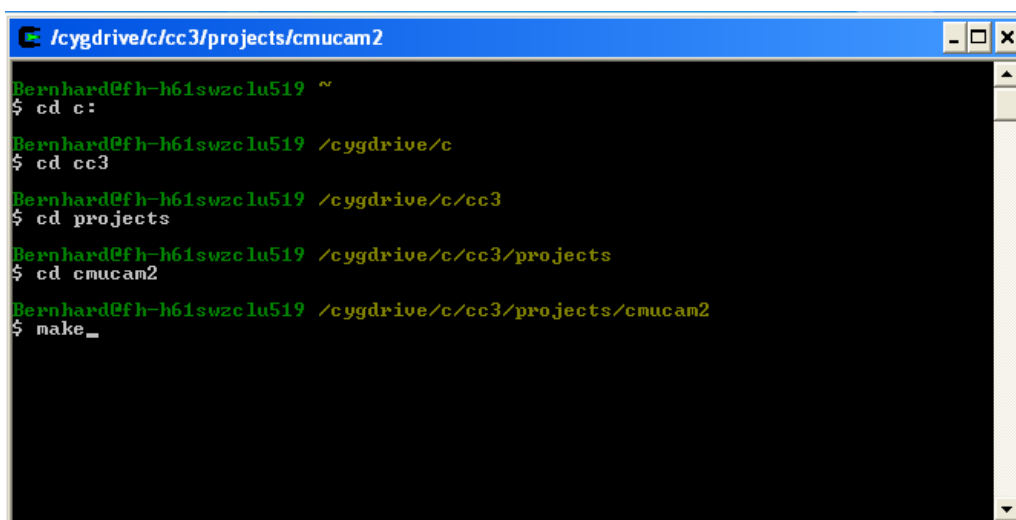
<sup>17</sup> [cc3 Froum, 2010]

## 3.5 Softwaretools

Die im Lieferumfang der CMUCAM3 enthaltenen Softwaretools werden im Folgenden näher beschrieben. Alle Softwaretools<sup>18</sup> können jederzeit kostenlos herunter geladen werden. Sie sind erforderlich, um die CMUCAM3 in Betrieb zu nehmen, zu programmieren und zu parametrieren.

### 3.5.1 Cygwin<sup>19</sup>

Cygwin wurde von der Firma Cygnus Solutions entwickelt und stellt eine Möglichkeit dar, Programme, die unter POSIX (portable operating system interface) entwickelt wurden, auf Microsoft Betriebssystemen zu portieren. Cygwin besteht aus der cygwin1.dll, welche als so genannte Kompatibilitätsschicht agiert und Linux API Funktionalitäten bietet und aus einer Sammlung von Tools, mit der man diese Funktionalitäten anwenden kann. In Abb. 7 ist die Oberfläche ersichtlich.



```
icydrive/c/cc3/projects/cmucam2
Bernhard@fh-h61swzclu519 ~
$ cd c:
Bernhard@fh-h61swzclu519 /cygdrive/c
$ cd cc3
Bernhard@fh-h61swzclu519 /cygdrive/c/cc3
$ cd projects
Bernhard@fh-h61swzclu519 /cygdrive/c/cc3/projects
$ cd cmucam2
Bernhard@fh-h61swzclu519 /cygdrive/c/cc3/projects/cmucam2
$ make_
```

Abb. 7: Cygwin - Compileroberfläche

---

<sup>18</sup> [Cmucam Tools, 2010]

<sup>19</sup> [Cygwin, 2010]

Mit dem Standard DOS-Befehl `cd` (change directory) wird in das Verzeichnis navigiert, in welchem sich die zu kompilierende C-Datei befindet. Der Compiler benötigt für die Kompilierung einige Informationen, die dieser in der Datei Makefile findet. Unter anderem muss hier der Dateiname angegeben werden, um dem Compiler mitzuteilen, welche Datei zu kompilieren ist. Dies ist insbesondere zu beachten, wenn neue, eigene C-Dateien erstellt werden. Der Dateiname muss dann unter `CSOURCES=filename.c` in der Datei Makefile geändert oder eingetragen werden. In den mitgelieferten Beispielpogrammen der CMUCAM3 existieren ebenfalls solche Dateien, die bei Bedarf angepasst werden müssen. Zur Anwendung kommt hier ein GNU ARM Compiler<sup>20</sup>.

---

<sup>20</sup> [Gnu Arm Compiler, 2010]

### 3.5.2 Philips Programmer

Für das Übertragen des kompilierten Programms in den Mikrocontroller steht das in Abb. 8 ersichtliche LPC200 Flash Utility der Firma Philips zur Verfügung.

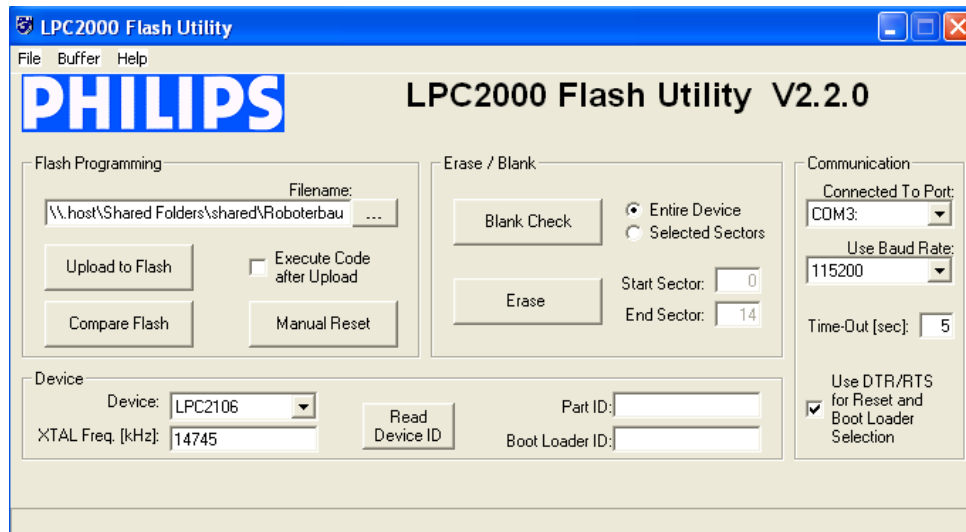


Abb. 8: Flash Tool für den LPC2106 Mikrocontroller

Mit diesem Programm, kann ein kompiliertes .hex file ausgewählt werden und über das serielle Interface in den Programmspeicher des Mikrocontrollers geladen werden. Die Programmierung des Mikrocontrollers über das serielle Interface erfordert einen Bootloader, welcher bereits bei der Auslieferung im Mikrocontroller vorhanden ist. Zu Beachten ist, dass in Verbindung mit dem Betriebssystem Microsoft XP professional nur die Programmversion V2.2.0 des LPC Flash Utilities eine funktionierende Verbindung zuließ.

### **3.5.3 Framegrabber**

Für die CMUCAM3 stehen im Lieferumfang zwei Framegrabber für den PC zur Verfügung. Der Framegrabber „CMUCM2GUI“ ist in der Programmiersprache Java erstellt worden und ist der Standard-Framegrabber des Vorgängermodells CMUCAM2. Für die CMUCAM3 steht der Framegrabber „CMUCAM3 Framegrabber“ zur Verfügung, welcher in Visual Basic programmiert wurde. Grundsätzlich kann die CMUCAM3 mit beiden Anwendungen bedient werden, es wird jedoch der Framegrabber „CMUCAM3 Framegrabber“ empfohlen, da dieser ein stabileres Verhalten aufweist, was Programmabstürze betrifft. Außerdem steht der Programmcode zur Verfügung, was in Bezug auf die Einbindung eines eigenen Framegrabbers in die Roboterfernsteuersoftware (siehe Punkt 5 Erstellung eines Framegrabbers) von großem Vorteil ist. Jedoch muss an dieser Stelle bemerkt werden, dass der Framegrabber „CMUCAM2GUI“ eine größere Anzahl an Funktionen bietet.

## **3.6 Inbetriebnahme und Programmierung**

Es ist unbedingt erforderlich, dass die Tools genau nach den der Kamera beiliegenden Dokumenten installiert werden, um eine reibungslose Funktion zu gewährleisten.

Nach der Installation müssen die Systemfunktionen der CMUCAM3 kompiliert werden, um die Projekte kompilieren zu können. Dazu muss mit dem unter Punkt 3.5.1 Cygwin erläuterten Tool in das Verzeichnis C/cc3/hal navigiert werden und der Befehl „make“ ausgeführt werden. Erst danach können die einzelnen Projekte kompiliert werden.

### 3.6.1 Flashen des Philips LPC2106 Mikrocontrollers

Nach dem Kompilieren steht eine Datei mit der Endung (\*.hex) zur Verfügung, welche mit dem unter Punkt 3.5.2 Philips Programmer angeführten Programmierwerkzeug aufgerufen werden kann. Zu beachten ist, dass die Einstellungen am Programmierwerkzeug identisch mit denen in Abb. 8 sein müssen, ausgenommen die Verbindungseinstellungen der Schnittstelle. Der Name des virtuellen Comports kann im Windows-Gerätemanager eingesehen werden. Die Baudrate ist immer auf 115200baud einzustellen bzw. eingestellt zu lassen, da der Bootloader im Mikrocontroller nur diese Baudrate unterstützt.

### 3.6.2 Verbindung mit dem PC/Roboter

In Abb. 9 ist die Pinbelegung der seriellen Schnittstellenverbindung dargestellt. Mit einem im Lieferumfang enthaltenen zehnpoligen Flachbandkabel ist die direkte Verbindung mit einem neunpoligen D-Sub-Stecker am PC oder dem USB-To-Serial-Adapter sofort möglich. Für die Eurobot 2010 müssen diese Signale jedoch über Funk übertragen werden, um sie zum Roboter zu senden oder von diesem zu empfangen. Dazu werden TTL Pegel benötigt, die durch das Entfernen des „level shifted serial jumper“ eingestellt werden.

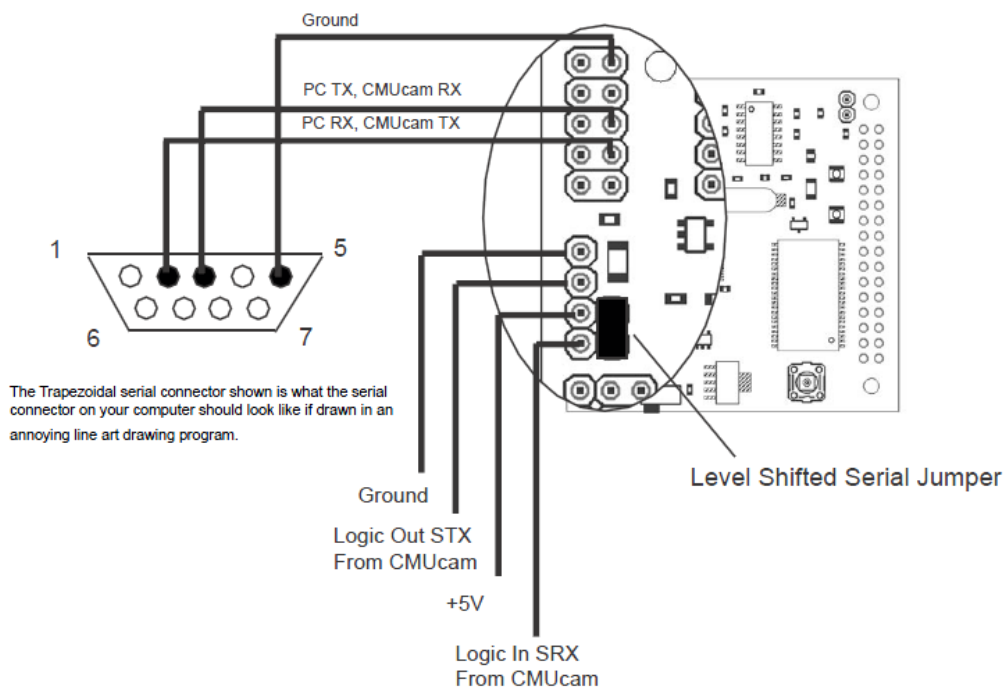


Abb. 9: Verbindung via serieller Schnittstelle

### 3.6.2 Steuerkommandos der CMUCAM3

In Abb. 10 sind alle von der CMUCAM3 unterstützten Kommandos ersichtlich. Beispiele für die Anwendung finden sich in den beiliegenden Dateien<sup>21</sup>

#### Buffer Commands

BM	Buffer Mode	30
RF	Read Frame	47

#### Camera Module Commands

CR	Camera Register	31
CP	Camera Power	32
CT	Camera Type	32

#### Data Rate Commands

DM	Delay Mode	33
PM	Poll Mode	46
PS	Packet Skip	47
RM	Raw Mode	48
PF	Packet Filter	46
OM	Output Packet Mask	45

#### Servo Commands

SV	Servo Position	53
SP	Servo Parameters	52
GS	Get Servo Position	36
SM	Servo Mask	51
SO	Servo Output	51

#### Image Windowing Commands

SF	Send Frame	50
DS	Down Sample	33
VW	Virtual Window	55
FS	Frame Stream	34
HR	HiRes Mode	38
GW	Get Window	38
PD	Pixel Difference	46

#### Auxiliary I/O Commands

GB	Get Button	35
GI	Get Auxiliary I/O	35
L0(1)	LED control	39

#### Color Tracking Commands

TC	Track Color	53
TI	Track Inverted	53
TW	Track Window	54
NF	Noise Filter	44
LM	Line Mode	40
GT	Get Tracking Parameters	37
ST	Set Tracking Parameters	52

#### Histogram Commands

GH	Get Histogram	35
HC	Histogram Config	38
HT	Histogram Track	39

#### Frame Differencing Commands

FD	Frame Difference	34
DC	Difference Channel	32
LF	Load Frame	39
MD	Mask Difference	44
UD	Upload Difference	55
HD	HiRes Difference	38
LM	Line Mode	40

#### Color Statistics Commands

GM	Get Mean	36
LM	Line Mode	40

#### System Level Commands

SD	Sleep Deeply	49
SL	Sleep	50
RS	Reset	49
GV	Get Version	37

Abb. 10: Unterstützte Kommandos der CMUCAM3<sup>22</sup>

<sup>21</sup> [cmucam.pdf, 2010, S18-27]

<sup>22</sup> [cmucam2\_manual, 2010, S28]

### 3.7 Nachteile der CMUCAM3

Durch die Arbeit mit der CMUCAM3 ergaben sich folgende Nachteile:

- Der Programmcode ist wenig beziehungsweise überhaupt nicht kommentiert, was eine äußerst zeitaufwendige Codeanalyse nach sich zieht, wenn Erweiterungen eingefügt werden.
- Die Dokumentation der Codefunktionen<sup>23</sup> bezieht sich zum Großteil nur auf Übergabeparameter, was nützlich ist, jedoch nicht ausreichend.
- Bei den beiliegenden Framegrabbern lässt sich eine Baudrate von 19200baud nicht auswählen, wodurch alleine deswegen die Notwendigkeit eines selbst programmierten Framegrabbers besteht. Die verwendeten Funkmodule zur Datenübertragung zwischen Roboter und Kameras unterstützen eine maximale Baudrate von 19200baud.
- Im Onlineforum des Herstellers werden Fragen bezüglich der CMUCAM3 erst sehr spät oder gar nicht beantwortet.

---

<sup>23</sup> [cc3api\_refman, 2007]

## 4 ADAPTIERUNG DER SOFTWARE

Für die Eurobot 2010 müssen zwei Kameras, die stationär installiert sind, in der Lage sein, an festgelegten Positionen festzustellen, ob es sich um einen weißen oder einen schwarzen Zylinder (ear of corn<sup>24</sup>) handelt. Mit der dritten Kamera wird festgestellt, ob der gegnerische Roboter bereits auf der Rampe war und die Bälle (oranges<sup>25</sup>) eingesammelt hat oder nicht.

Für diese Funktionen muss die Software der Kameras angepasst und erweitert werden. Grundsätzlich sind die Erweiterungen für alle drei Kameras identisch, es sind ausschließlich die interessierenden Bildausschnitte und die versendeten Datenpakete verschieden.

### 4.1 Beispielprogramme der CMUCAM3

Die CMUCAM3 bietet eine Vielzahl an zugänglichen Beispielprogrammen, um die Anwendungsmöglichkeiten zu verdeutlichen.

Um eigene Programme zu entwickeln, empfiehlt der Hersteller das Beispielprogramm `hello-world.c`<sup>26</sup> zu kopieren und abzuändern. Für eine Grundlage der kommandobasierten Programmsteuerung eignet sich jedoch das Beispielprogramm `cmucam2.c`<sup>27</sup> besser.

Für die Ansicht und Erweiterung der Programme wird ein Standardtexteditor verwendet.

---

<sup>24</sup> [Regeln Eurobot, 2010, S9]

<sup>25</sup> [Regeln Eurobot, 2010, S9]

<sup>26</sup> [Software CC3 Source Tree, 2010]

<sup>27</sup> [Software CC3 Source Tree, 2010]

## 4.2 Anpassung und Erweiterung

Für die Eurobot 2010 werden im Folgenden die Anpassungen und Definitionen für die CMUCAM3 erläutert.

### 4.2.1 Sichtbereiche für die Eurobot 2010

Jede der beiden Spielfeldhälften wird in vier Sektoren eingeteilt (siehe Abb. 11), in denen jeweils ein schwarzer Zylinder (fake corn<sup>28</sup>) vorhanden ist. Die restlichen Zylinder sind weiß (ears of corn) und können vom Roboter angefahren und eingesammelt werden. Die Position des schwarzen Zylinders in jedem Sektor ist variabel. Kamera eins und zwei erfassen jeweils die Zylinderfarbe der eingekreisten Position, wohingegen Kamera drei die Bälle auf der Gegnerrampe erfasst und detektiert ob sich diese noch dort befinden. Abb. 11 zeigt die Kamerastandorte, wenn die Spielfarbe Blau ist. Durch die Sektoreinteilung wird es möglich von den gemessenen Zylindern auf die Farbe der unerfassten zu schließen, da nur ein Zylinder schwarz sein kann. Aufgrund dessen, dass die Spielhälften spiegelverkehrt angeordnet sind, kann von der Messung von Kamera zwei auf die Elemente in der anderen Hälfte geschlossen werden und umgekehrt. Kamera eins und Kamera zwei erfassen jeweils vier Zylinder, was den Vorteil bietet, dass bei Ausfall einer der beiden Kameras noch ein Großteil der Zylinderfarben bestimmt werden kann. Insgesamt gibt es 36 verschiedenen Konstellationen der.

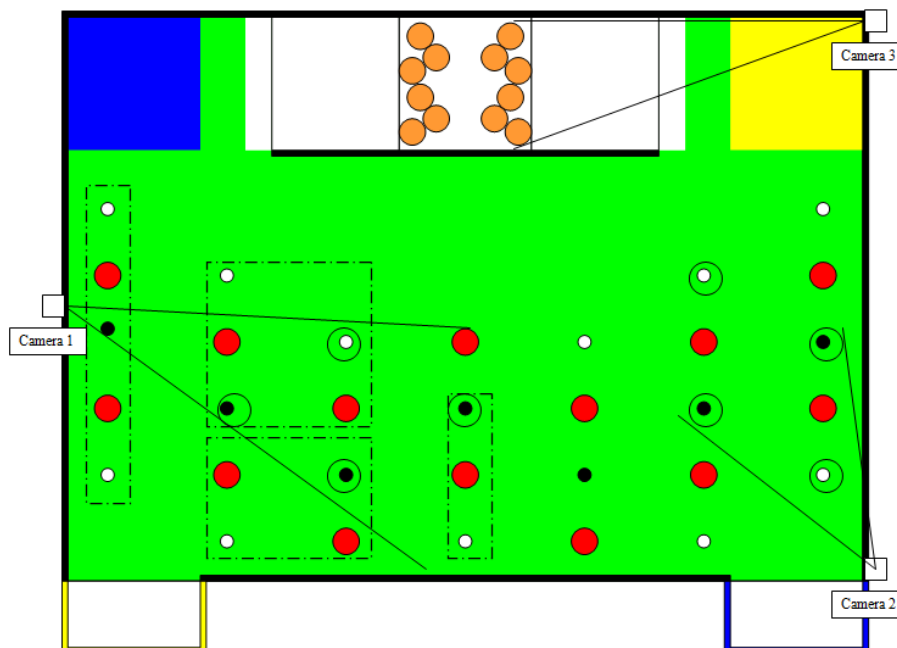


Abb. 11: Sichtbereiche der Kameras

---

<sup>28</sup> [Regeln Eurobot, 2010, S10]

#### 4.2.2 Vergleich der Farbbereiche RGB und YCrCb

Um eine möglichst sichere und stabile Auswertung der Elemente sicherzustellen, wurden Versuche mit den Farbbereichen RGB<sup>29</sup> und YCrCb<sup>30</sup> durchgeführt. In Abb. 12 und Abb. 13 sind die Bildaufnahmen dargestellt. Da die Kamera als Farbsensor eingesetzt wird, liegt die Vermutung nahe, dass eine Farbbereichsumschaltung in YCrCb einen Vorteil bringt. Diese Vermutung wird jedoch wie unten angeführt nicht bestätigt.



Abb. 12: Bildaufnahme Kamera zwei im RGB Farbbereich

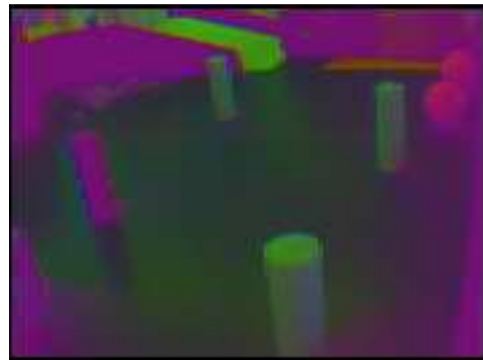


Abb. 13: Bildaufnahme der Kamera zwei im YCrCb Farbbereich

Farbwerte Kanal Blau (B):

Schwarz: 16

Weiß: 93

Farbwerte Kanal Blau (Cb):

Schwarz: 100

Weiß: 70

Die Auswertung der Zylinderfarbe erfolgt ausschließlich mit dem Blauanteil, da dieser im Spielfeld nicht vorkommt. Der Grünanteil ist ungeeignet, da Grün von den Weißen Zylindern vom Boden stark reflektiert wird und dadurch die Messung verfälscht. Da die Bälle (tomatoes<sup>31</sup>) rot sind, ist auch dieser Farbanteil ungeeignet. Aufgrund der technischen Gegebenheiten der CMUCAM3 liegt ein Farbwert immer innerhalb von 16 – 250. Dies liegt daran, dass 0 und 255 für die Synchronisation der Datenübertragung über die Schnittstelle im Pollmode<sup>32</sup> verwendet wird. Wie man in der Auswertung der Abbildungen 12 und 13 erkennen kann, ist die Differenz der Farbwerte im RGB Bereich viel größer als im YCrCb Bereich, was einen toleranteren Threshold erlaubt. Aus diesem Grund wird für die Messung der Farbmittelwerte des Blauanteils der RGB Bereich verwendet.

---

<sup>29</sup> [Neumann, 2004]

<sup>30</sup> [cmucam.pdf, 2003, S9]

<sup>31</sup> [Regeln Eurobot, 2010, S10]

<sup>32</sup> [cmucam.pdf, 2003, S22]

### 4.2.3 Programmablaufdiagramm

In Abb. 14 ist der grundsätzliche Programmablauf der Kamera ersichtlich. Eine Auflistung der Kommandos und ihre Bedeutung ist unter Punkt 4.2.4 Datentransferprotokoll angegeben.

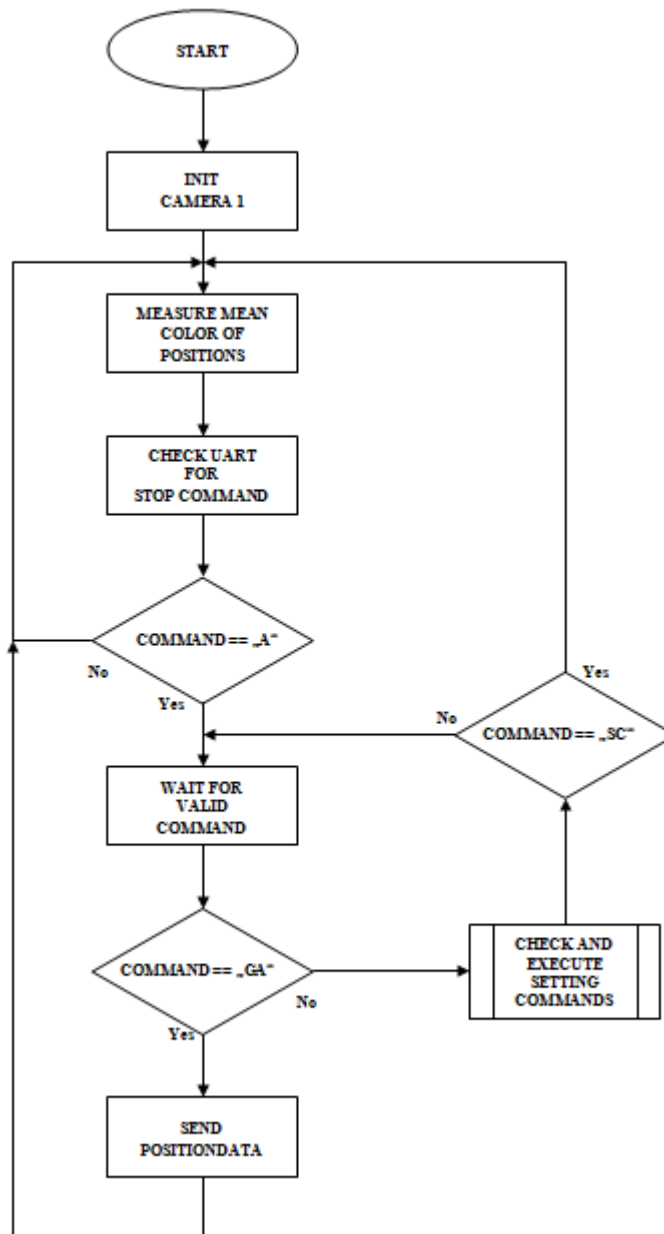


Abb. 14: Flussdiagramm Kamerasteuerung

In Abb. 15 wird der Prozess „Check and execute setting commands“ genauer erläutert:

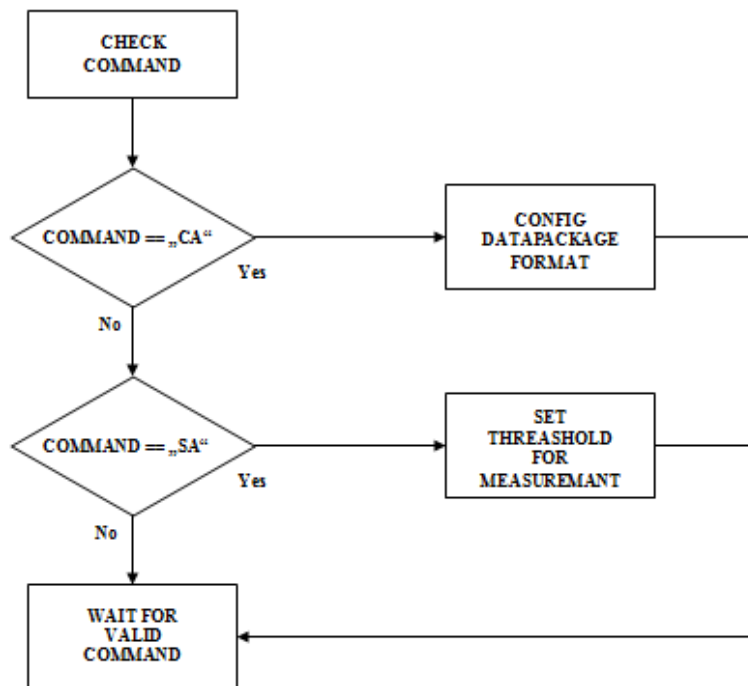


Abb. 15: Flussdiagramm des Prozesses „Check and execute setting commands“

#### 4.2.4 Zustandsautomat

Der Prozess „Measure mean colour of positions“ wird in einem Zustandsautomaten modelliert und ist in Abb. 16 ersichtlich:

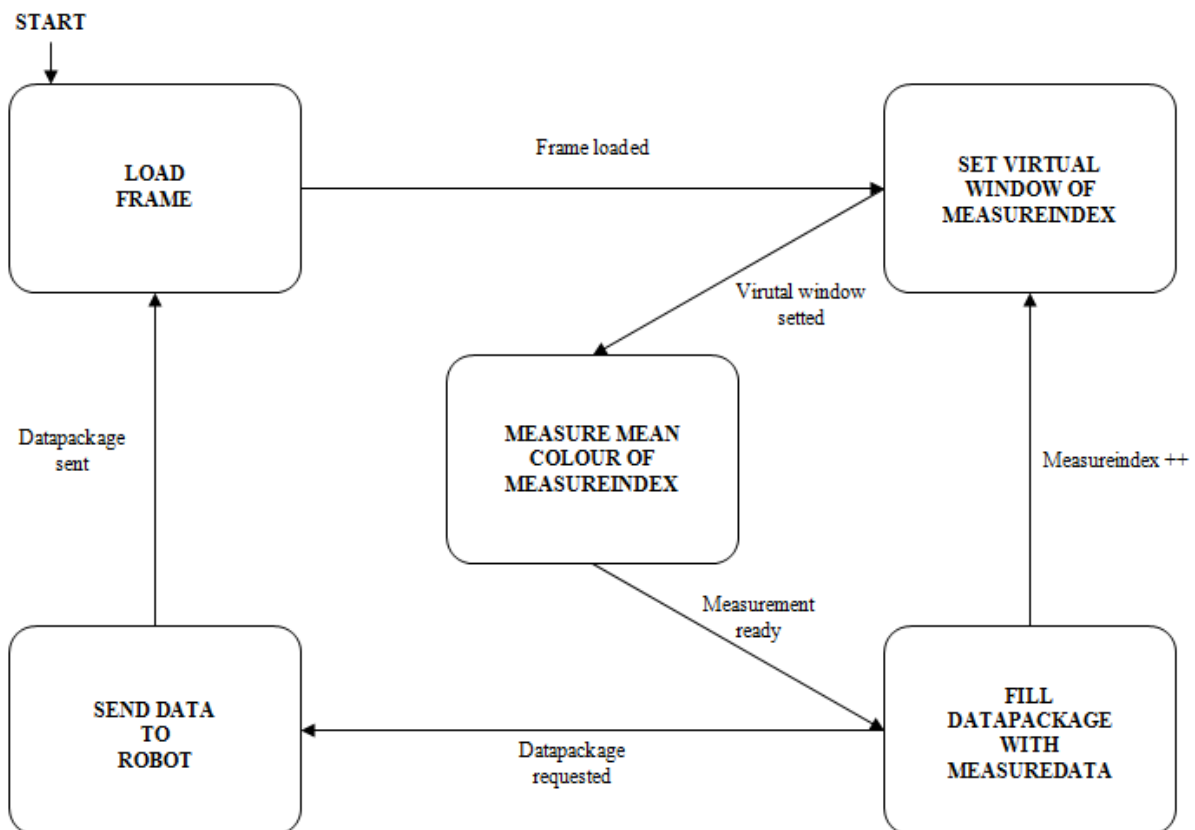


Abb. 16: Zustandsautomat des Prozesses „Measure mean colour of positions“

#### 4.2.5 Datentransferprotokoll

Für die Kommunikation werden folgende Befehle definiert, um den Datenaustausch mit den drei Kameras zu ermöglichen:

- **SC:** start check position cam 1, cam 2, cam 3:  
Startet die Farbmittelwertsmessung für alle Kameras
- **A:** stop measure camera 1  
Stoppt die Messung von Kamera 1
- **B:** stop measure camera 2  
Stoppt die Messung von Kamera 2
- **C:** stop measure camera 3  
Stoppt die Messung von Kamera 3
- **GA:** get datapackage cam 1 and start check position  
Datenpaket von Kamera 1 anfordern und anschließend Messung fortsetzen
- **GB:** get datapackage cam 2 and start check position  
Datenpaket von Kamera 2 anfordern und anschließend Messung fortsetzen
- **GC:** get datapackage cam 3 and start check position  
Datenpaket von Kamera 3 anfordern und anschließend Messung fortsetzen
- **CA:** config cam 1  
“CA 0”: Kamera 1 gibt von nun an nach „GA“ die binären Werte zurück  
“CA 1”: Kamera 1 gibt von nun an nach „GA“ die Farbmittelwerte zurück
- **CB:** config cam 2  
“CB 0”: Kamera 2 gibt von nun an nach „GB“ die binären Werte zurück  
“CB 1”: Kamera 2 gibt von nun an nach „GB“ die Farbmittelwerte zurück
- **CC:** config cam 3  
“CC 0”: Kamera 3 gibt von nun an nach „GC“ die binären Werte zurück  
“CC 1”: Kamera 3 gibt von nun an nach „GC“ die Farbmittelwerte zurück

- **SA:** set cam 1 threshold  
“SA 40”: setzt den Farbmittelwert, ab dem der binäre Wert auf 0 gesetzt wird
- **SB:** set cam 2 threshold  
“SB 40”: setzt den Farbmittelwert, ab dem der binäre Wert auf 0 gesetzt wird
- **SC:** set cam 3 threshold  
“SA 40”: setzt den Farbmittelwert, ab dem der binäre Wert auf 0 gesetzt wird
- **RS:** reset cam 1, cam 2 cam 3  
Reset aller Kameras und Neustart der Messung

Die Datenpakete, die von den Kameras zum Roboter übertragen werden, sehen wie folgt aus:

**C1 X X X X:** Datenpaket der Kamera eins. **X** kann 0 für „Weiß“ oder 1 für „Schwarz“ annehmen

**C2 X X X X:** Datenpaket der Kamera zwei. **X** kann 0 für „Weiß“ oder 1 für „Schwarz“ annehmen

**C3 X X X X X X:** Datenpaket der Kamera drei. **X** kann 0 für „Orange erkannt“ oder 1 für „Orange nicht erkannt“ annehmen

### Beispielsequenz:

- I. Roboter an Kamera 1: „A“
- II. Roboter an Kamera 1: „GA“
- III. Kamera1 an Roboter: „C1 0 1 1 0“
- IV. Roboter an Kamera 2: „B“
- V. Roboter an Kamera 3: „GB“
- VI. Kamera2 an Roboter: „C2 0 1 0 0“

Obige Sequenz beschreibt die Anfrage des Roboters an Kamera eins nach dem Datenpaket der Farbmessung. Die Kamera eins sendet das Datenpaket (C1 0 1 1 0) mit den Informationen der Zylinderfarben. Anschließend wird die Anfrage des Roboters für die Kamera zwei gesendet und auch hier erhält der Roboter wieder ein Datenpaket. Der Roboter kann anschließend auf die restlichen Zylinderfarben an den jeweiligen Positionen schließen.

### 4.3 Implementierung

Die Implementierung der Programmerweiterungen erfolgte nach oben angeführten Spezifikationen in der Programmiersprache C. Das Beispielprogramm cmucam2.c wurde für die Eurobot 2010 um folgende Funktionen erweitert:

```
static void cmucam2_get_meanOfAllPositions (cc3_color_info_pkt_t * s_pkt,  
                                           bool poll_mode, bool line_mode,  
                                           bool buf_mode,  
                                           uint8_t sw_color_space,  
                                           bool quiet);
```

Aufgabe: Messung des Farbmittelwertes des aktuellen Bildbereiches.

```
static void set_positionOfElement(int index);
```

Aufgabe: Setzen der virtuellen Bildbereiche auf die Elementpositionen (ears of corn)

```
static void cmucam2_fill_positions (cc3_color_info_pkt_t * pkt, uint i);
```

Aufgabe: Messergebnis in das zu sendende Datenpaket einfügen

```
static void cmucam2_send_positions (void);
```

Aufgabe: Datenpaket mit den Informationen der gemessenen Elemente an den Roboter senden

Der C-Code der Funktionen ist in Punkt 8 Anhang ersichtlich.

## **5 ERSTELLUNG EINES FRAMEGRABBERS**

Im Lieferumfang der CMUCAM3 sind zwei Framegrabber enthalten, die eine Konfiguration der Kamera ermöglichen und durch Sendekommandos auch Daten empfangen können. Da im Zuge der Entwicklung des Roboters für die Eurobot 2010 ein Fernsteuerungsprogramm mit der Programmiersprache C# erstellt wird, mit dessen Hilfe Kommandos an den Roboter gesendet und empfangen werden können, liegt es nahe einen Framegrabber in dieses Programm einzubinden, der die benötigte Funktionalität bietet. Die mitgelieferten Framegrabber haben den wesentlichen Nachteil, dass eine Baudrate von 19200baud nicht ausgewählt werden kann. Da die Datenübertragung aber mittels Funkmodul erfolgt, welches ausschließlich 19200baud unterstützt, besteht ein zwingender Grund, einen eigenen Framegrabber zu entwickeln.

Diese Anwendung soll in weiterer Folge in der Lage sein, auf Anforderung ein Bild einer in der Kamera definierten Grundposition zu empfangen und anzuzeigen. Weiters soll es möglich sein, aufgrund der durch die Kamera erfassten Daten die aktuelle Konstellation der Zylinder anzuzeigen, um die Richtigkeit der Messung zu testen.

## 5.1 Roboterfernsteuerung

Auf Basis eines bereits existierenden Fernsteuerungsprogramms, das in der Programmiersprache LabView<sup>33</sup> erstellt wurde, wird im Zuge der Eurobot 2010 eine neue und erweiterte Anwendung für diesen Zweck erstellt. Eine Aufgabe dieses Programms liegt in der Konfiguration der Kameras eins bis drei. Mit Hilfe der Fernsteuersoftware kann neben den Grundkonfigurationen der Kameras folgendes eingestellt werden:

- Es kann ein Datenpaket angefordert werden, das Auskunft über die Zylinderfarbe an der jeweiligen Position gibt.
- Dieses Datenpaket kann so konfiguriert werden, dass es entweder den Mittelwert der gemessenen Farbwerte jeder Position oder einen definierten Wert zurückgibt (1 = Schwarz; 0 = Weiß; 2 = nicht definiert)
- Der Farbwert, ab dem ein Zylinder als weiß markiert wird kann nachträglich eingestellt werden, um so auf die jeweiligen Verhältnisse zu reagieren.
- Es wird ein Bild einer vordefinierten Grundposition angezeigt, mit dessen Hilfe die Kameraposition justiert werden kann, da es durchaus möglich ist, dass die Bewerbungsspieltische gewisse Toleranzen in den Bemaßungen aufweisen.

---

<sup>33</sup> [LabView, 2010]

## 5.2 Einbindung des Framegrabbers in die Fernsteuersoftware

Da das Fernsteuerungsprogramm den Inhalt einer anderen Bachelorarbeit<sup>34</sup> darstellt, wird nur die Funktion eines Framegrabbers implementiert, welcher anschließend in das Fernsteuerungsprogramm eingebunden und erweitert wird. Der Framegrabber wird natürlich ebenfalls in der objekt-orientierten Programmiersprache C# erstellt. Abb. 17 zeigt die Bedienoberfläche des erstellten Framegrabbers.

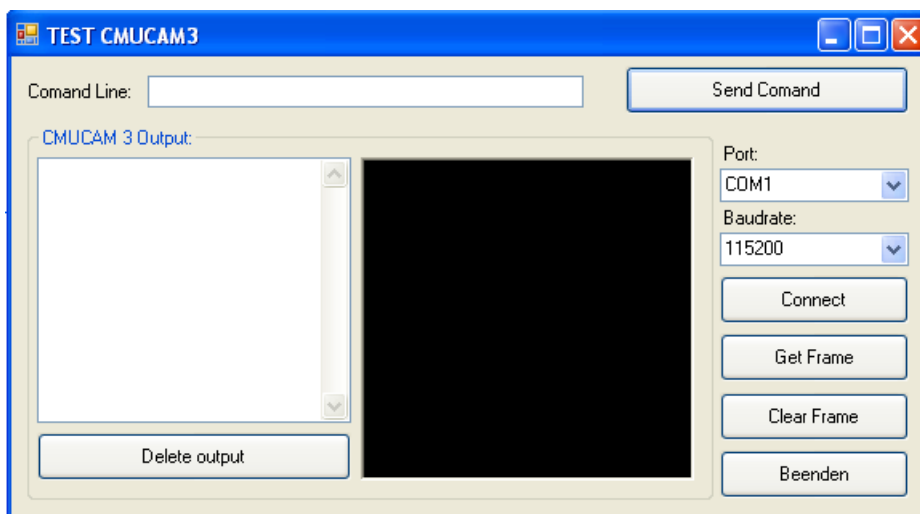


Abb. 17: Oberfläche des programmierten Framegrabbers

---

<sup>34</sup> [Mitterndofer, 2010]

## 6 ZUSAMMENFASSUNG UND AUSBLICK

Die CMUCAM3 ist sehr gut für den Einsatz als Farbsensor geeignet. Für die Eurobot 2010 ist sie ein wichtiger Bestandteil um Kollisionen mit fest am Boden verschraubten fake corns zu verhindern. Somit kann der Roboter bereits eine optimale Fahrtroute berechnen, ohne dass sich dieser in Bewegung setzen muss. Für einfache Anwendungen, die keine hohe Bildqualität voraussetzen, können hohe Verarbeitungsgeschwindigkeiten erzielt werden. Mit dem seriellen RS232 Interface ist eine Anbindung an ein bestehendes System sehr einfach möglich. Durch das Definieren von spezifischen Kommandos kann die CMUCAM3 äußerst flexibel eingesetzt werden. Mit der CMUCAM3 steht ein wiederverwendbares, erweiterbares und kompaktes System zur Verfügung.

Durch die Möglichkeit der zeilenweisen Auswertung der Bilder, ist die Verwendung für einen Linienverfolgerroboter denkbar. Eine weite Anwendungsmöglichkeit wäre ein Bewegungsmelder für eine einfache Hinderniserkennung bei bewegten Hindernissen.

Da moderne Notebooks oft nicht mehr über eine RS232 Schnittstelle verfügen, müssen dafür USB to serial Converter verwendet werden. Es hat sich als schwierig erwiesen, das Flashtool von Philips (siehe Punkt 3.5.2 Philips Programmer) mit verschiedenen Betriebssystemen und verschiedenen Convertern einzusetzen. Eine Alternative bietet das Flashtool Flash Magic<sup>35</sup>, welches unter den Betriebssystemen Microsoft XP, Vista und Win7 mit verschiedenen USB to serial Converter erfolgreich eingesetzt werden kann. Dieses Tool kann ebenfalls kostenlos heruntergeladen werden und bietet zusätzlich ein Terminalprogramm zum Senden und Empfangen von Daten.

---

<sup>35</sup> [Flash Magic, 2010]

## 7 LITERATUR

### 7.1 Bücher

[Neumann, 2004] Neumann, Burkhard: Bildverarbeitung für Einsteiger,

Hrsg. Springer 2004

[Tietze, Schenk, 1999] Tietze, Ulrich ; Schenk, Christoph: Halbleiterschaltungstechnik,

Hrsg. Springer, 1999

### 7.2 Datenblätter der Kamera

[CMUCAM2GUI\_Overview, CMUCAM3\_sdk\_guide, 2010] [http://www.cmucam.org/](http://www.cmucam.org/wiki/Documentation)

[wiki/ Documentation](http://www.cmucam.org/wiki/Documentation),

Stand 29.04.2006

[cmucam.pdf, 2003] [http://www.cmucam.org/wiki/ Documentation](http://www.cmucam.org/wiki/Documentation), Stand 2003

[cmucam2\_manual, 2003] [http://www.cmucam.org/wiki/ Documentation](http://www.cmucam.org/wiki/Documentation), Stand 2003

[cc3api\_refman, 2007] [http://www.cmucam.org/wiki/ Documentation](http://www.cmucam.org/wiki/Documentation), Stand 19.05.2003

[Hardware, CMUCAM3\_datasheet, 2007] <http://www.cmucam.org/wiki/Documentation>,

Stand vom 22.09.2007

[LPC2106, Datasheet, 2004] [http://www.keil.com/dd/docs/datashts/philips/lpc2104\\_2105\\_2106.pdf](http://www.keil.com/dd/docs/datashts/philips/lpc2104_2105_2106.pdf)

Stand vom 01.03.2010

## 7.3 Internetquellen

- [austrobot, 2010] <http://austrobot.info/>, Stand vom 01.03.2010
- [CMUCAM, 2010] <http://www.cmucam.org/>, Stand vom 01.03.2010
- [Carnegie Mellon University, 2010] <http://www.cmu.edu/>, Stand vom 01.03.2010
- [cc3 Forum, 2010] <http://www.cmucam.org/discussion/topic/142>
- [Cmucam Tools, 2010] <http://www.cmucam.org/wiki/Windows-Quick-Start>, Stand vom 01.03.2010
- [Cygwin, 2010] <http://www.cygwin.com/>, Stand vom 01.03.2010
- [Easy-Radio, 2010] <http://www.roboter-teile.de/Shop/index.php>, Stand vom 01.03.2010
- [Flash Magic, 2010] <http://www.flashmagictool.com/> Stand vom 01.03.2010
- [Gnu Arm Compiler, 2010] [http://www.codesourcery.com/gnu\\_toolchains/arm/download.html](http://www.codesourcery.com/gnu_toolchains/arm/download.html)  
Stand vom 01.03.2010
- [LabView, 2010] <http://www.ni.com/labview/d/>, Stand vom 01.03.2010
- [Robo-Racing-Team, 2010] <http://rrt.fh-wels.at/>, Stand vom 01.03.2010
- [Pob-Technology, 2010] <http://www.pob-technology.com/>, Stand vom 01.03.2010
- [Regeln Eurobot, 2010] [http://www.planete-sciences.org/robot/data/file/coupe/2010/  
E2010\\_rules\\_and\\_drawing\\_EN.pdf](http://www.planete-sciences.org/robot/data/file/coupe/2010/E2010_rules_and_drawing_EN.pdf), Stand vom 22.09.2009
- [robotikhardware, 2010] <http://www.shop.robotikhardware.de/>, Stand vom 01.03.2010
- [Software CC3 Source Tree, 2010] <http://www.cmucam.org/wiki/Downloads>,  
Stand vom 01.03.2010

## 7.4 Relevante Bachelorarbeiten

- [Mitterndofer, 2010] Entwurf einer Benutzeroberfläche zur Steuerung der Aktoren und Auslesen  
der Sensoren des Eurobot-Roboters



## 8.2 C-Code für das Setzen des virtuellen Bildbereichs

```
////////////////////////////////////  
//Funktion zum Einstellen des virtuellen Fensters (Bereiche in denen sich die Elemente befinden)  
//21.10.2009 Muckenhumer  
////////////////////////////////////  
void set_positionOfElement(int indexelement)  
{  
    if(indexelement == 0)  
    {  
        //virtuelles Fenster auf die Position des ersten Elementes von Interesse setzten  
        cc3_pixbuf_frame_set_roi (26 * 2,  
                                41, 27 * 2, 59);  
    }  
    else if(indexelement == 1)  
    {  
        //virtuelles Fenster auf die Position des zweiten Elementes von Interesse setzten  
        cc3_pixbuf_frame_set_roi (55 * 2,  
                                75, 58 * 2, 111);  
    }  
    else if(indexelement == 2)  
    {  
        //virtuelles Fenster auf die Position des dritten Elementes von Interesse setzten  
        cc3_pixbuf_frame_set_roi (54 * 2,  
                                48, 56 * 2, 71);  
    }  
    else if(indexelement == 3)  
    {  
        //virtuelles Fenster auf die Position des vierten Elementes von Interesse setzten  
        cc3_pixbuf_frame_set_roi (16 * 2,  
                                62, 18 * 2, 90);  
    }  
}  
}
```

Abb. 19: Funktion set\_positionOfElement

## 8.3 C-Code für das Erstellen des Datenpaketes

```
////////////////////////////////////  
//Funktion zum Erstellen des Datenpaketes  
//20.10.09 Muckenhumer  
////////////////////////////////////  
void cmucam2_fill_positions (cc3_color_info_pkt_t * pkt, uint i)  
{  
    uint8_t pkt2[6];  
  
    pkt2[0] = pkt->mean.channel[0];  
    pkt2[1] = pkt->mean.channel[1];  
    pkt2[2] = pkt->mean.channel[2];  
    pkt2[3] = pkt->deviation.channel[0];  
    pkt2[4] = pkt->deviation.channel[1];  
    pkt2[5] = pkt->deviation.channel[2];  
  
    //checken ob schwarz oder weiß  
    if((pkt2[2] <= colorvalue))  
    {  
        if(datainbool)  
        {  
            //Diese Position wird als schwarz markiert  
            position[i] = 1;  
        }  
        else  
        {  
            position[i] = pkt2[2];  
        }  
    }  
    else  
    {  
        if(datainbool)  
        {  
            //Diese Position wird als weiß markiert  
            position[i] = 0;  
        }  
        else  
        {  
            position[i] = pkt2[2];  
        }  
    }  
}  
}
```

Abb. 20: Funktion cmucam2\_fill\_positions

## 8.4 C-Code für das Senden des Datenpaketes

```
////////////////////////////////////  
//Funktion zum Senden des Datenpaketes  
//20.10.09 Muckenhumer  
////////////////////////////////////  
void cmucam2_send_positions (void)  
{  
    if (raw_mode_output)  
    {  
        putchar('#');  
    }  
    else  
    {  
        printf("#");  
    }  
  
    printf ("c1");  
    for (int i = 0; i < 4; i++)  
    {  
        printf (" %d", position[i]);  
        //putchar (position[i]);  
    }  
    printf ("*");  
}
```

Abb. 21: Funktion cmucam2\_send\_positions