Entfernungsbestimmungen aus Akkommodation mit einem aktiven Kamerasystem

vorgelegt von

Josef Siemes

Kiel, im August 1998

Diplomarbeit

am Lehrstuhl für Kognitive Systeme
des Instituts für Informatik und Praktische Mathematik
der Christian–Albrechts–Universität zu Kiel
Prof. Dr. G. Sommer

Betreuer:
Dipl. Phys. Michael Hansen

Zusammenfassung

Diese Arbeit entwickelt ein Verfahren, das monokular unter Nutzung der Fokussteuerung einer aktiven Kamera eine Tiefenkarte einer Szene erstellen kann. Es wird das *Depth-from-Focus*-Verfahren angewendet, das zusammen mit anderen Verfahren diskutiert wird. Durch die Fokussierung kommt es zu Bildpunktverschiebungen, die durch ein Kalibrierverfahren ausgeglichen werden. Zum Messen der Fokussierung werden Schärfefilter eingeführt. Im Verfahren wird der sog. Tenengrad verwendet. Das gesamte Verfahren wurde auf einem echtzeitfähigen Rechner implementiert, so daß die Rechenzeit des Verfahrens in der Größenordnung der Fokussierbewegung der Kamera liegt. Insgesamt wird innerhalb von 20 Sekunden eine Tiefenkarte erstellt. Diese Tiefenkarte wird in absolute Entfernungswerte umgerechnet. Die Ergebnisse zeigen, daß die Genauigkeit dieses Verfahrens in der Größenordnung des Tiefenschärfebereichs der Kamera liegt.

Inhaltsverzeichnis

Kapitel 1

Motivation

Ein Ziel von Bildverarbeitungssystemen ist es, Objekte aus Bildern oder ähnlichen Messungen im Rechner in einem Weltkoordinatensystem zu rekonstruieren. In vielen Systemen wird für diese Aufgabe ein Stereokamerasystem genutzt, auch das menschliche visuelle System nutzt dafür zwei Augen.

Die Rekonstruktion dieser Objekte kann jedoch auch monokular durchgeführt werden: Mit Laserscannern kann aktiv die Entfernung einzelner Weltpunkte festgestellt werden, verschiedene sogenannte *Form-von-X*-Verfahren liefern zum Beispiel aus dem Schattenwurf von Objekten Tiefeninformationen. Weiterhin können wir auch die Freiheitsgrade aktiver Kameras ausnutzen, wie die Fokussteuerung. Unter einer aktiven Kamera versteht man ein System, das über regelbare optische und mechanische Freiheitsgrade verfügt.

Im wesentlichen unterscheidet man bei Verwendung des Fokus einer aktiven Kamera zwei Verfahren: *Depth from Focus* und *Depth from Defocus*. Beim *Depth-from-Focus*-Verfahren wird zu einem Weltpunkt die das Bild am schärfsten abbildende Fokusposition bestimmt. Daraus kann ein Entfernungswert berechnet werden. *Depth from Defocus* bestimmt aus zwei mit unterschiedlichen Fokuspositionen aufgenommenen Bildern die Entfernungsinformationen. Im folgenden werden die möglichen Verfahren genauer besprochen.

1.1 Das Stereoverfahren

Das menschliche visuelle System (MVS) nutzt für die Szenenrekonstruktion im Gehirn die Information von zwei Augen, auch in vielen Bildverarbeitungssystemen werden zwei und mehr Kameras für die Aufnahme der Bilder, die zur Rekonstruktion führen sollen, verwendet. Das MVS kann mit einem Auge im Nahbereich nur sehr schlecht Tiefe wahrnehmen, gerade für Greifvorgänge sind zwei Augen unerläßlich. Erst im Fernbereich gewinnt die Hinzunahme von monokular gewonnenen Tiefenhinweisen zunehmend an Bedeutung. Hierzu zählen [2]:

- Bekannte Objektgröße. Durch Vorwissen kennt das MVS die Größe von Objekten, und es kann daraus Entfernung schätzen.
- Schattierungen und Farbeindruck. Die Schattierung eines Objekts läßt Rückschlüsse auf dessen Form und Entfernung zu. Sehr weit entfernte Objekte bekommen einen Blaustich und verlieren Details. Dies heißt in der Malerei „Luftperspektive“ [18].
- Verdeckungen. Nahe Objekte verdecken weiter entfernte.

In Stereokamerasystemen werden ähnlich dem MVS zwei Kameras verwendet. Zur Tiefenmessung müssen die zwei Abbildungen eines Weltpunktes in den beiden Kamerabildern gefunden und zugeordnet werden. Das erste Problem besteht damit in der Identifizierung von Punkten in den Kamerabildern. Man sucht nach Merkmalen, die diese Punkte im Bild auszeichnen, wie z.B. Grauwertecken. Eckendetektoren liefern die gesuchten Punkte. Aufgrund der Eigenschaften dieser Eckendetektoren sind die Ergebnisse aber maximal pixelgenau lokalisierbar. Als Erweiterung zur subpixelgenauen Lokalisation polyedrischer Objekte bieten sich daher Kantendetektoren an. Aus den sich ergebenden approximierten Geraden lassen sich die Schnittpunkte dieser Geraden berechnen. Bei polyedrischen Objekten ist dies mit Subpixelgenauigkeit die Grauwertecke, bei allgemeinen Objekten kann die Genauigkeit aber wesentlich schlechter werden.

Danach müssen die in den beiden Bildern gefundenen Punkte zugeordnet werden. Diese Zuordnung geschieht meist mit Ähnlichkeitsmaßen, die die lokale Umgebung der Punkte betrachtet. Das sichert aber noch nicht die korrekte Zuordnung. Meist werden noch weitere Einschränkungen der Szene vorgenommen, z.B. sollten die beiden korrespondierenden Punkte nicht zu weit in den beiden Bildern auseinanderliegen, also die Disparität beschränkt sein. Damit wird der mögliche Tiefenbereich eingeschränkt, es lassen sich aber Fehler in der Zuordnung leichter vermeiden.

Durch Triangulation der beiden korrespondierenden Bildpunkte wird dann den entsprechenden Weltpunkt berechnet. Es müssen jedoch die internen und externen Parameter der Kameras bekannt sein. Diese Parameter werden in klassischer Weise durch aufwendige Kalibrierungen der Kameras mit Kalibrierungsobjekten ermittelt[6]. In vielen Arbeiten wird die exakte Kalibrierung durch adaptive Verfahren umgangen [17]. Die Triangulation benötigt für genaue Ergebnisse eine gute Kalibrierung und eine subpixelgenaue Detektion der beiden entsprechenden Punkte in den Kamerabildern.

Die Zuordnung durch die Identifizierung der entsprechenden Punkte in den beiden Kamerabildern führt zu vielen Problemen. Da die Kameras die Szene aus verschiedenen Blickwinkeln aufnehmen, kommt es zu Verdeckungen im Bild. Weiterhin wird die Aufnahme mit zwei Kameras durchgeführt, die jeweils eine eigene Optik besitzen und so zum Beispiel gleiche Szenenpunkte mit unterschiedlicher Helligkeit aufnehmen.

Insgesamt sind bei diesem Verfahren zur Rekonstruktion folgende Aspekte zu beachten:

- Die Zuordnung der Punkte muß stabil sein, d.h. es dürfen keine Fehler in der Zuordnung passieren.
- Die Punkte müssen sehr genau, möglichst im Subpixelbereich, in den beiden Bildern lokalisiert werden. Dadurch erhält man geringe Fehler in der Tiefenschätzung.
- Die Zuordnung soll nur für ähnliche Punkte erfolgen, d.h die Umgebung um die zugeordneten Punkte soll sich nicht stark unterscheiden. Diese Zuordnung kann durch verschiedene Einstellungen der beiden Kameras (Iris, Zoom, Fokus) empfindlich gestört werden.
- Spiegelungen und Reflektionen können zu stark unterschiedlichen Abbildungen der Szene in den beiden Kamerabildern führen.
- Die externen und internen Kameraparameter müssen bekannt sein.
- Beide Kameras müssen die aufzunehmenden Objekte im Bild haben. In einem Bild durch weitere Objekte verdeckte Punkte oder außerhalb des Bildes liegende Punkte können im zweiten Bild nicht zugeordnet werden, so daß bei einem Versuch der Zuordnung dieser Punkte es zu Fehlern in der Zuordnung kommen kann.

Die externen und internen Kameraparameter können mit Kalibrierungen gefunden werden. Die Verdeckungen, Zuordnungsprobleme und Spiegelungen führen aber zu Problemen bei der Rekonstruktion der Szenen: Da in einem Kamerabild extrahierte Punkte und Kanten z.B. im Falle von Verdeckungen im anderen Kamerabild nicht zu finden sind, führt dies zu falschen Zuordnungen und daher gegebenenfalls zu falschen Rekonstruktionen der aufgenommenen Objekte.

1.2 Monokulare Verfahren

Es ist wünschenswert ein Verfahren zu haben, mit dem die Weltkoordinaten der interessanten Punkte rekonstruiert werden können und die zuvor geschilderten Probleme umgangen werden. Es bietet sich an, Verfahren mit nur einem aktiven Sensor für die Rekonstruktion der Weltkoordinaten zu nutzen. Mit nur einem Sensor können keine Verdeckungen mehr in der Szene auftreten. Es müssen aber aktive Verfahren der Tiefenmessung verwendet werden. Passive Sensoren reichen nur mit viel Vorwissen über die aufgenommene Szene aus, für ein allgemein verwendbares Verfahren ist ein passiver Sensor nicht ausreichend.

1.2.1 Laserscanner

Eine Möglichkeit Entfernung zu bestimmen, bieten Laserscanner. Bei einem Laserscanner wird keine herkömmliche Kamera genutzt, sondern mit Hilfe eines Lasers wird sehr genau die Entfernung des Zielpunktes gemessen. Handelsübliche Geräte erreichen absolute Genauigkeiten von 4 mm auf 4 Metern Tiefe [9]. Diese Geräte können mit einem Drehspiegel eine horizontale Tiefenkarte erstellen. Mit einem weiteren Spiegel können diese Geräte auch ganze Flächen abtasten, also dichte Tiefenkarten erstellen.

Die Laserscanner berechnen die Entfernung aus der Laufzeit oder Phasenverschiebung des ausgesendeten Laserstrahls gegenüber dem von einem Objekt der Szene reflektierten Signal. Dies impliziert eine aufwendige Meßelektronik in diesen Geräten, die für die geforderte Genauigkeit der Entfernungsmessung sehr teuer ist. Der Laserscanner ist auf einen gewissen Anteil an Reflektionen angewiesen, so daß diese Geräte für polierte Metalloberflächen oder Spiegel keine Entfernung messen können.

Die Reflektion des Laserstrahls kann sich mit der eines weiteren Laserscanners stören. Je nach Farbe des Lasers kann er auch den visuellen Eindruck der Szene stören. Ein roter Laser wird z.B. ein aufgenommenes Kamerabild mit einer roten Linie stören. Um diese Störeffekte zu vermeiden, werden auch Infrarotlaser für diese Messungen verwendet.

1.2.2 *Shape from Shading*

Beim *Shape-from-Shading* (SFS)-Verfahren [10] versucht man, aus einem Bild auf die Oberflächengestalt des aufgenommenen Objektes zu schließen. Als bekannt werden die Reflektionseigenschaften des Objektes und die Beleuchtung angenommen. In erster Näherung kann man sich zum Beispiel das Bild einer Kugel, die aus der Richtung der Kamera beleuchtet wird, selbst als Tiefenkarte vorstellen. In genaueren Verfahren wird das Bild in einzelne Facetten gleicher Reflektanz zerlegt, die dann in Flächen mit entsprechendem Winkel aneinandergehängt werden. Insgesamt ergibt sich damit eine Tiefenkarte.

In allgemein einsetzbaren Systemen will man möglichst wenig Vorwissen über die aufgenommene Szene verwenden. Beim SFS-Verfahren können nur bekannte Objekte unter bekannter Beleuchtung rekonstruiert werden. Dieses Verfahren kann also nicht allgemein eingesetzt werden.

1.2.3 *Depth from Focus*

Entfernungsmessungen von Weltpunkten kann man mit einer Kamera über eine Fokussierung erreichen: Wenn der Weltpunkt scharf auf die Bildebene abgebildet wird, gibt die Fokuseinstellung oder die Bildweite ein Maß für die Entfernung dieses Punktes von der Kamera an. Für einen Weltpunkt wäre dies also das Ergebnis eines Autofokussystems. Autofokussysteme werden z.B. in [1, 25] vorgestellt. Diesen Vorgang der Anpassung des Linsensystems auf die Entfernung des Objektes bezeichnet man allgemein als Akkommodation.

Um für ein ganzes Bild diese Entfernungsschätzungen durchführen zu können, nehmen wir für viele Fokuspositionen ein Kamerabild auf und stellen für jeden Punkt dieser Kamerabilder die Schärfe in der Umgebung dieses Punktes fest. An der Fokusstellung der größten Schärfe haben wir bei voller Ausnutzung des Fokusbereiches eine gute Schätzung für die Entfernung dieses Punktes. Dieses Verfahren nennt sich in der Literatur allgemein *Depth from Focus* [7, 11, 24].

Das in dieser Arbeit vorgestellte Verfahren führt genau diese Fokussierung durch und liefert damit für eine monokular aufgenommene Szene eine Tiefenkarte. Es sind also keine Probleme durch verdeckte Kanten oder Glanzlichter zu erwarten. Durch Verwendung von speziellen Bildverarbeitungsrechnern ist eine schnelle Berechnung dieser Tiefenkarte möglich. Gewünscht ist eine Echtzeitverarbeitung der Bilder. Vorausgehende Messungen lassen dies aber leider nicht ganz erwarten, da ein Fokussierungsvorgang des verwendeten Kamerasystems bis zu 3 Sekunden dauert.

Weiterhin müssen einige Kalibrierungen der verwendeten Kamera durchgeführt werden:

- Kalibrierung der Bildpunktverschiebungen im Kamerabild bei der Fokussierung.

- Kalibrierung der Fokuspositionen zu den realen Entfernungen.

Die Größe der aufgenommenen Kamerabilder ändert sich je nach Fokusposition. Bei genauen Messungen stellen sich für verschiedene Richtungen der Fokussierung unterschiedliche Werte heraus (Siehe Kapitel 4). Die frühen Arbeiten [7] haben dieses Problem gar nicht behandelt. Krotkov [11] erwähnt das Problem, umgeht es aber durch eine große örtliche Ausdehnung der Schärfefilter.

Einen weiteren wichtigen Aspekt dieser Methode stellen die verwendeten Schärfefilter dar, die in Kapitel 3 diskutiert werden.

1.2.4 *Depth from Defocus*

Ein weiteres Verfahren ist das sogenannte *Depth from Defocus* [15, 24]. Es nutzt die Informationen aus zwei mit unterschiedlichen Fokuspositionen aufgenommenen Bildern. Mit speziellen Filtern kann aus diesen Bildern eine Tiefenkarte berechnet werden. Dieses Verfahren geht davon aus, daß die Bilder bis auf den Fokus mit gleichen Parametern aufgenommen werden. Insbesondere können Bildpunktverschiebungen (siehe Kapitel 4) dieses Verfahren erschweren. Alle in der Literatur gezeigten Systeme arbeiten daher mit einer telezentrischen Optik [23], die in Abschnitt 4.1 genauer vorgestellt wird. Diese Optik erzeugt bei allen Fokuspositionen Bilder gleicher Größe. Da wir diese telezentrische Optik nicht zur Verfügung haben, können wir dieses Verfahren mit unserem Kamerasystem nicht anwenden.

Das *Depth-from-Defocus*-Verfahren zeigt gute Ergebnisse. Mit einer speziellen Anordnung zweier CCD-Sensoren in einer Kamera kann man sogar die Echtzeitfähigkeit dieses Verfahrens (33 ms) in einem Entfernungsdifferenz von 30 cm realisieren [15]. Es wird ein Beleuchtungmuster auf die aufzunehmende Szene durch die Kameraoptik projiziert, und zwei CCD-Module nehmen gleichzeitig das mit einem Strahlteiler duplizierte Bild mit zwei Fokuspositionen auf. Es werden also Störungen durch Änderungen in der Szene zwischen den beiden Aufnahmen vermieden. Durch das Beleuchtungsmuster kann auch für texturlose Flächen eine dichte Tiefenkarte berechnet werden. Im aufgenommenen Bereich beträgt der Fehler der Tiefenschätzung ca. 0,3 %.

Dieses Verfahren gibt nur für Entfernungen zwischen den scharfen Fokuswerten der beiden aufgenommenen Bilder sinnvolle Ergebnisse. Daher müssen für große Entfernungen eventuell mehrere Bilder mit den entsprechenden Fokuspositionen aufgenommen werden.

Kapitel 2

Kamerageometrie

In diesem Abschnitt wird die verwendete Kamera vorgestellt und ein Modell für diese Kamera eingeführt.

Die Kamera besteht aus einem Fuji-Objektiv mit Zoom- und Fokusregelung (*close loop*). Die Iris kann nur gesteuert werden (*open loop*), die aktuelle Position der Blende kann also nicht ausgelesen werden. Als Sensor dient ein CCD-Modul. Das CCD-Modul ist fest im Kameragehäuse montiert. Zwei Linsensysteme im Kameragehäuse können für die Zoom- und Fokussteuerungen bewegt werden.

Das Objektiv ist ein Fujinon HGX11.5B-MPX31. Es hat eine Brennweite von 11,5 bis 69 mm, kann also bis Vergrößerungsfaktor 6 zoomen. Der Fokus ist von ca. 1 m bis ∞ regelbar. Der Durchmesser der Objektivlinse beträgt 4,5 cm. Dies ist gleichzeitig der maximale Durchmesser der Iris.

Der CCD-Sensor ist ein XC-77RR-CE der Firma Sony. Die Auflösung beträgt 756×581 Pixel, jedes Pixel hat eine Größe von 11×11 μm. Die Abtastfläche des CCD-Chips ist $8{,}8 \times 6{,}6$ mm groß. Der Adapter DC-77RR-CE von Sony bereitet das Signal des CCD-Sensors auf und gibt das Videosignal des aufgenommenen Bildes aus.

2.1 Linsen und Linsensysteme

Kameralinsen können als Kugelabschnitte modelliert werden. Durch den unterschiedlichen Brechungsindex von Luft und (im allgemeinen) Glas brechen die Linsen das einfallende Licht. Linsen, die als Kugelabschnitte realisiert sind, heißen *dicke Linsen*. Sie bündeln paralleles Licht im wesentlichen in einem Punkt. Nur bei Lichtstrahlen, die in der Nähe des Linsenzentrums die Linse passieren, ergibt sich ein Punkt. Außerhalb dieses Bereiches findet die Brechung durch die Krümmung der Linsen nicht in einer Ebene statt, daher werden Lichtstrahlen, die entfernt vom Linsenzentrum die Linse passieren, vor dem Brennpunkt gebündelt. Genau einen Punkt erreicht man nur bei einer *dünnen Linse*. Diese hat den gleichen Brechungsindex wie die entsprechende *dicke Linse*, hat aber keine Ausdehnung. Sie bricht also das Licht in einer Ebene, nicht mehr auf einem Kugelabschnitt. Eine konvexe *dünne Linse* hat zwei *Brennpunkte*, in denen paralleles Licht gebündelt wird. Die Entfernung dieser Punkte von der Linsenachse ist die *Brennweite*. Diese *dünnen Linsen* gibt es nur in der Theorie, als Näherung dieser Linsen kann man Fresnel-Linsen betrachten. Sie sind sehr flach, aus schmalen Abschnitten einer Kugel aufgebaut und brechen das Licht in einer Ebene. Durch Spiegelungen an den Übergängen der einzelnen Abschnitte erzeugen diese Linsen aber sehr viel Streulicht.

Eine *dicke Linse* kann durch ein Linsensystem zweier *dünner Linsen* dargestellt werden. Jede *dünne Linse* stellt dabei eine Seite der *dicken Linse* dar. Es ergibt sich ein Abstand der beiden *dünnen Linsen*, der der Ausdehnung der *dicken Linse* entspricht. Durch weitere *dünne Linsen* im Linsensystem kann jedes Linsensystem dargestellt werden. Es ergibt sich eine vordere und eine hintere Brennweite des Systems, die nicht notwendigerweise gleich groß sein müssen.

2.2 Modell einer Kamera

Das Modell unserer Kamera besteht aus der Bildebene und einer dünnen Linse. Die Irisblende liegt in der Ebene der Linse, der sogenannten Hauptebene. Abbildung 2.1 zeigt den resultierenden Strahlengang eines Weltpunktes, der scharf auf die Bildebene abgebildet wird.

Zum Fokussieren des Bildes können wir in diesem Modell die Bildebene bewegen. Die Irisblende mit Durchmesser A und die Fokussierung dieses Modells läßt sich regeln. Die Linse hat die Brennweite f. Es wird also ein Weltpunkt mit Abstand g zur Linse in Abstand b zur Linse fokussiert. Ein Objekt der Größe G wird auf ein Bild der Größe B abgebildet. Hierbei gilt:

$$\frac{G}{g} = \frac{B}{b}. \tag{2.1}$$

$$\frac{1}{f} = \frac{1}{g} + \frac{1}{b}. \tag{2.2}$$

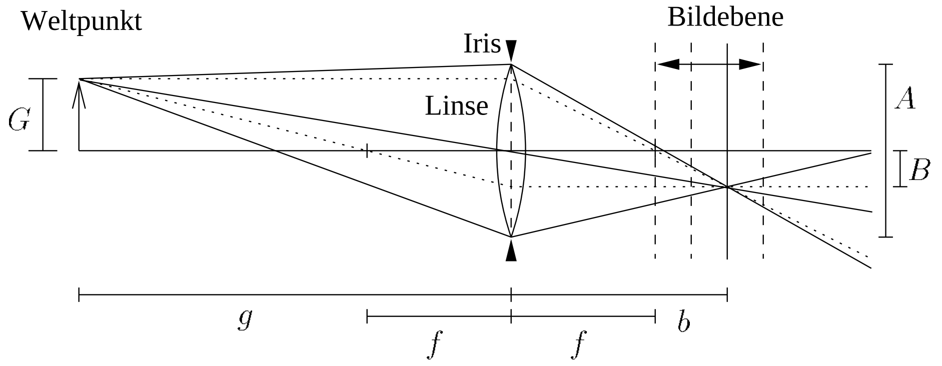

Abbildung 2.1: Modell einer Kamera. Links der Weltpunkt, der durch die Linse und die Iris rechts auf die Bildebene abgebildet wird. Die Bildebene läßt sich zum Fokussieren bewegen, und bildet mit dem Abstand b zur Bildebene den Punkt mit Größe B ab.

In realen Kameras wird die Irisblende wabenförmig realisiert, da eine kreisförmige Iris technisch nicht möglich ist. Im Modell ist die Iris jedoch kreisförmig. Es wird also jeder Weltpunkt auf einen Kreis in der Bildebene abgebildet. Für Gleichung 2.2 wird der Weltpunkt auf einen Kreis mit Radius gegen Null, also einen Punkt abgebildet. Ansonsten ergibt sich für eine um δ verschobene Bildebene und Irisdurchmesser A ein Kreis mit Radius $r = \frac{A\delta}{2b}$ als Abbildung. Die Intensität dieses Kreises beträgt $I = \frac{1}{\pi r^2}$.

Dies führt bei realen Bildern dazu, daß Schwarzweißkanten scharf abgebildet wieder Schwarzweißkanten ergeben, unscharf abgebildet ergibt sich ein weicher (grauer)Übergang zwischen diesen Flächen. Wir werden diesen Effekt näher in Abschnitt 3 betrachten.

Dieses Modell einer realen Kamera berücksichtigt keine Zoomoptik. Diese bewirkt eine Änderung der Brennweite f. Bei unseren Experimenten wird von einer konstanten Zoomstellung ausgegangen.

Dicke Linsen in realen Kameras haben für unterschiedliche Farben des Lichtes unterschiedliche Brennweiten. Das führt zu Farbverfälschungen im Bild. Diesen Effekt nennt man *Dispersion*. Durch geeignete Anordnung mehrerer Linsen läßt sich dieser Effekt vermindern.

Weitere Abbildungsfehler von Kameras können durch ungenau gefertigte Linsen entstehen. Falls die Linsen nicht exakt gefertigt sind, also nur annähernd einen Kugelabschnitt beschreiben, oder nicht exakt parallel zur Bildebene ausgerichtet sind, kann dies zu Abbildungsfehlern führen. In [16] wird daher eine Kalibrierung dieser Linsenfehler vorgestellt. Es wird eine Verschiebung um den Bildhauptpunkt als Effekt dieser Fehler festgestellt.

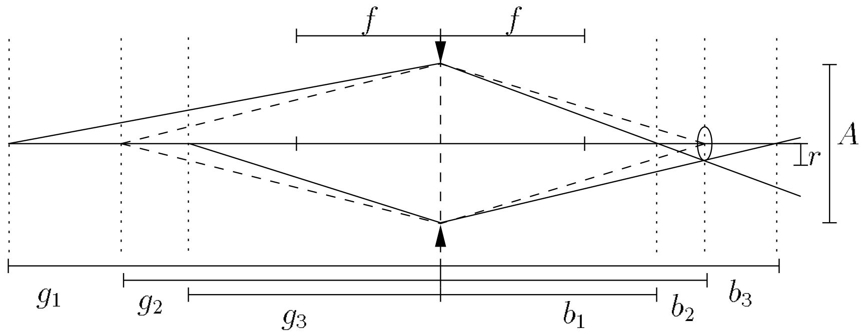

Abbildung 2.2: Tiefenschärfebereich einer Kamera. Der Sensor mit Durchmesser r bildet die Objekte im Abstand g_1 bis g_3, dem Tiefenschärfebereich, scharf ab. Ein entsprechender Bereich im Bildbereich ist b_1 bis b_3.

Durch geeignete Anordnung von Linsen sollen reale Kameras mögliche Linsenfehler, wie die Dispersion und die Effekte von dicken Linsen, vermeiden. Es folgt daraus ein Linsensystem, das wie eine dünne Linse arbeitet. An der verwendeten Kamera wurden keine Linsenfehler festgestellt. Es ergaben sich Größenänderungen um den Bildhauptpunkt bei Fokussierungen. Diesen Effekt kann aus Abbildung 2.1 abgelesen werden, genauer wird dies in Kapitel 4 besprochen.

Üblicherweise wird in realen Kameras die Linse bewegt. Die Bildebene ist fest. Auch unser Kamerasystem hat eine feste Bildebene und ein bewegliches Linsensystem. Bei realen Kameras wird zum Fokussieren die Linse und nicht die Bildebene bewegt. Da die Entfernungsänderung der Linsenebene zum aufgenommenen Objekt nur minimal ist, können wir diesen Fehler vernachlässigen [23].

2.3 Tiefenschärfe

Wir können Objekte nicht mit beliebiger Genauigkeit abbilden. Ein begrenzender Faktor ist z.B. die Korngröße des Films bei Fotokameras oder die Größe eines Pixels der CCD-Einheit bei einer digitalen Kamera. Dadurch erscheinen Objekte nicht in nur genau einer Entfernung scharf, sondern in einem ganzen Entfernungsbereich. Dieser wird als Tiefenschärfebereich bezeichnet.

Den Tiefenschärfebereich können wir auf zwei Arten definieren: Einmal durch den Bereich, in dem wir ein Objekt in der Welt verschieben können, ohne daß das zugehörige Bild dieses Objektes unscharf wird, oder zum zweiten durch den Verschiebungsbereich der Bildebene, in dem das entstehende Bild des Objektes scharf bleibt. Abbildung 2.2 zeigt die beiden Bereiche.

Diese beiden Bereiche können durch einen Wert ausgedrückt werden:

$$\frac{1}{f} = \frac{1}{g} + \frac{1}{b} \tag{2.3}$$

$$= \frac{1}{g} + \delta_T - \delta_T + \frac{1}{b} \tag{2.4}$$

$$= \frac{1}{g'} + \frac{1}{b'} \tag{2.5}$$

für

$$\frac{1}{g'} = \frac{1}{g} + \delta_T \tag{2.6}$$

$$\frac{1}{b'} = \frac{1}{b} - \delta_T \tag{2.7}$$

$$\tag{2.8}$$

Mit $\delta_T = \frac{1}{g'} - \frac{1}{g} = \frac{1}{b} - \frac{1}{b'}$ ergibt sich ein Wert für den Tiefenschärfebereich $|g - g'|$ beziehungsweise $|b - b'|$.

Eine Beispielrechnung für unsere Kamera ergibt:

$$\begin{aligned}
A &= 3,375 \text{ cm} \\
f &= 34,5 \text{ mm} \\
r &= 5,5\ \mu\text{m} \\
g_2 &= 2 \text{ m} \\
b_2 &= \frac{1}{\frac{1}{f} - \frac{1}{g_2}} = 35,106 \text{ mm} \\
b_1 &= \frac{b_2}{1 + \frac{2r}{A}} = 35,094 \text{ mm} \\
b_3 &= \frac{b_2}{1 - \frac{2r}{A}} = 35,117 \text{ mm} \\
\delta_T &= \frac{1}{b_1} - \frac{1}{b_3} = 0,0186\ \frac{1}{\text{m}} \\
d_b &= b_3 - b_1 = 22,88\ \mu\text{m}
\end{aligned}$$

$$
\begin{aligned}
g_1 &= \frac{1}{\frac{1}{f} - \frac{1}{b_1}} = 2,038 \text{ m} \\
g_3 &= \frac{1}{\frac{1}{f} - \frac{1}{b_3}} = \\
&= \frac{1}{\frac{1}{g_1} + \delta_T} = 1,964 \text{ m} \\
d_g &= g_1 - g_3 = 7,43 \text{ cm}
\end{aligned}
$$

Der Tiefenschärfebereich im Bildraum (d_b) und im Objektraum (d_g) ist unterschiedlich groß, kann aber ohne weiteres ineinander umgerechnet werden. Weiterhin sind die Abstände der vorderen und hinteren Ebenen des Tiefenschärfebereichs nicht gleich weit vom Zentrum des Tiefenschärfebereichs (Abstände b_2 bzw. g_2) entfernt.

Das menschliche Auge hat nach [7] einen $\sim 10\times$ größeren Tiefenschärfebereich als eine Kamera, da die Iris entsprechend kleiner ist.

2.4 Überlagerung unscharfer Grauwertkanten

Durch unscharf abgebildete Objekte erhalten wir weiche Kanten in aufgenommenen Bildern. Diese Kanten können sich überlagern, wie z.B. beim Siemensstern. Durch die Überlagerung der einzelnen Kanten des Siemenssterns entstehen kreisförmig wechselnde Muster.

Abbildung 2.3 zeigt diesen Effekt. Die Grauwertkanten überlagern sich so, daß teilweise ursprünglich schwarze Bildteile hell und ursprünglich weiße Bildteile dunkel abgebildet werden.

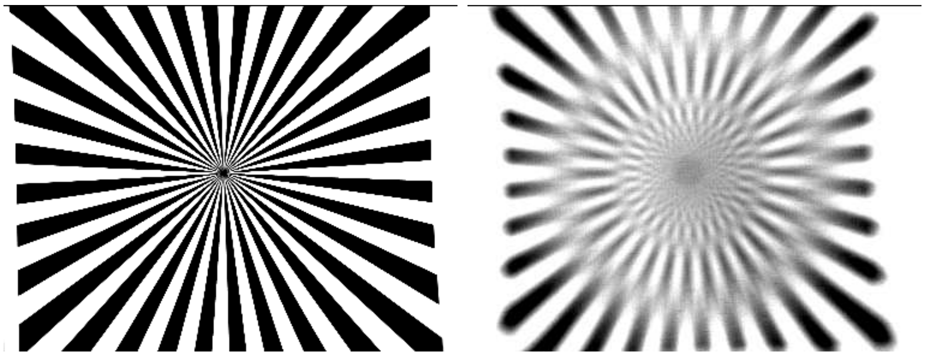

Abbildung 2.3: Überlagerung der Kanten beim Siemensstern. Links das scharf aufgenommene Bild, rechts das unscharfe Bild.

Kapitel 3

Messen von Schärfe

Unschärfe in einem aufgenommenen Bild kommt nach Abschnitt 2.2 durch die Abbildung eines Weltpunktes auf einen Kreis zustande. Eine homogene Grauwertfläche, die als Überlagerung verschiedener Weltpunkte zu verstehen ist, wird dadurch nicht beeinflußt, sie bleibt homogen. Erst an Grauwertkanten macht sich dies bemerkbar. Anstatt harter Grauwertübergänge sehen wir fließende Übergänge zwischen den Grauwerten. Schärfe können wir aus diesem Grund nur an Grauwertübergängen messen. Auf homogenen Bildbereichen läßt sich keine Unschärfe feststellen. Abbildung 3.1 zeigt einen Schwarz-Weiß-Übergang, der durch Faltung mit einer Kreismaske zu einer unscharf abgebildeten Kante modelliert wird. Unscharfe Bildbereiche enthalten einen geringeren Anteil hoher Frequenzen als scharfe Bildbereiche.

Wir können ohne Kenntnis der Szene aus dem Bild keine absoluten Schärfekriterien messen. Ein unscharfes Bild einer realen Szene können wir zum Beispiel nicht von einer scharf aufgenommenen Fotografie dieser unscharf abgebildeten Szene unterscheiden. Beide Bilder könnten identisch sein. Erst beim Vergleich verschiedener Fokusstellungen können wir die ideale Fokusstellung herausfinden. Das Ergebnis von Abbildung 3.1 könnte auch ein Foto der unscharf aufgenommenen Szene darstellen, die scharf abgebildet wird.

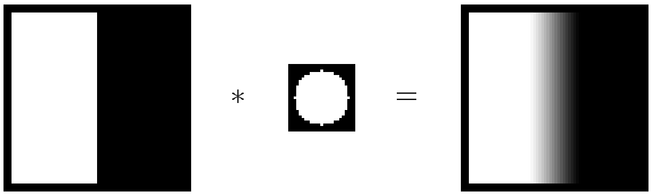

Abbildung 3.1: Unschärfe im rechten Bild durch Faltung mit der Kreismaske.

Ein Schärfefilter wird uns daher einen relativen Wert für die Schärfe des Bildes liefern. Über ein Optimierungsverfahren erhalten wir die ideale Fokusstellung.

Schärfe können wir zum einen für ein ganzes Bild messen, zum anderen in lokalen Umgebungen des Bildes. Da ein unscharfes Bild weiche Grauwertübergänge enthält, sind die meisten Schärfefilter von Kantenfiltern abgeleitet.

Ein idealer Schärfefilter soll ein Maximum an der schärfsten Fokusstellung liefern. Weiterhin sollten lokale Maxima an unscharfen Fokuspositionen signifikant kleiner sein als das globale Maximum. Um die Suche nach der schärfsten Fokusposition zu vereinfachen, sollte das Maximum recht schmal sein. Als Suchalgorithmus wird in [12] die Fibonacci-Suche vorgeschlagen. Diese grenzt von links und rechts das Maximum so lange ein, bis der Suchbereich sehr klein ist. Damit haben wir für diesen Bildpunkt die optimale Fokusposition gefunden. Dieses Verfahren ist nicht sehr rechenaufwendig, benötigt aber durch viele Fokussieroperationen auf der verwendeten Kamera viel Zeit. In [1] wird zum Beispiel für ein Autofokusverfahren die Fibonacci-Suche verwendet. Die Fokussierung eines Objektes benötigt dort eine Zeit von 25 Sekunden.

Wir wollen den Schärfefilter nicht nur auf einen Punkt anwenden, sondern auf alle Bildpunkte. Mit dem vorhandenen Bildverarbeitungssystem haben wir auch genügend Rechenkapazität, die Suche auf dem ganzen Bild und auf vielen Fokuspositionen in kurzer Zeit durchzuführen. Begrenzender Faktor ist hier das verwendete Kamerasystem, da ein Durchlauf der Fokussierung von Entfernung ∞ bis 1 m ungefähr 3 Sekunden dauert. Wir können eine vollständige Suche der Maxima für alle Bildpunkte implementieren. Der Fokus wird einmal komplett durchlaufen, während jeweils die Maxima der Filterantworten gespeichert werden.

3.1 Visuelle Verfahren

Unschärfe in Bildern kann das menschliche visuelle System nur bis zu einem bestimmten Grade feststellen. Die Ortsauflösung des Auges beträgt 2-10 Linien/Grad [19]. In Experimenten an einem Autofokussystem wurden Testpersonen gebeten, ein Bild scharfzustellen [25]. Die Testpersonen stellten den Fokus auf verschiedene Positionen ein, das Ergebnis des Autofokussystems erschien aber allen Probanden scharf. Dies gibt Hinweise darauf, daß das MVS zur genauen visuellen Beurteilung von Unschärfe Hilfsmittel benötigt.

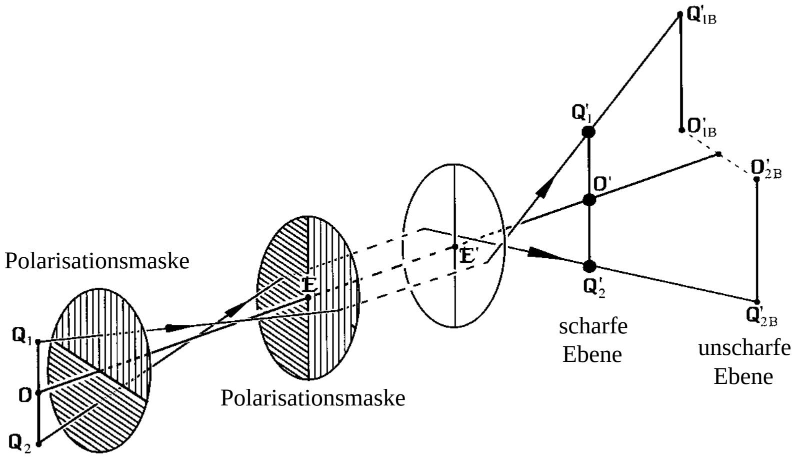

Abbildung 3.2: Polarisationsfilter werden zur Fokussierung genutzt. Die Kante Q_1-O-Q_2 wird durch zwei Polarisationsfilter auf die Kante Q'_1-O'-Q'_2 scharf abgebildet, bei unscharfer Abbildung ergeben sich zwei Kanten Q'_{1B}-O'_{1B} und O'_{2B}-Q'_{2B}.(Aus [19])

3.1.1 Polarisationsfilter

Man kann zur Detektion von Unschärfe Polarisationsfilter einsetzen. Dies führt bei einem unscharfen Bild zu verschobenen Kanten. Da das menschliche visuelle System verschobene Kanten wesentlich besser als Unschärfe im Bild erkennen kann, erhält man eine bessere manuelle Fokussierung der Bilder.

Zur Messung werden zwei Polarisationsfilter in einer Maske verwendet, um jeweils in einem Teil des Bildes die horizontale, im anderen Teil die vertikale Polarisationsebene des Bildes durchzulassen. Im weiteren Strahlverlauf wird diese Maske um 90° gedreht verwendet.

Dies führt dazu, daß bei unscharfen Kanten ein Teil des gestreuten Lichts aus dem Bild gefiltert wird und dadurch sich die abgebildete Kante verschiebt. Durch die zwei Polarisationsfilter wird der Effekt in beide Richtungen ausgenutzt, so daß er sich weiter verstärkt. Abbildung 3.2 zeigt den Aufbau. Die Kante Q_1-O-Q_2 wird auf die scharfe Kante Q'_1-O'-Q'_2 abgebildet. In einer unscharfen Ebene wird diese Kante auf zwei Kanten Q'_{1B}-O'_{1B} und O'_{2B}-Q'_{2B} abgebildet.

In Spiegelreflexkameras wird dieser Effekt mit dünnen Prismen im Sucher erreicht. Bei unscharf abgebildeten Kanten ergeben sich geteilte Kanten.

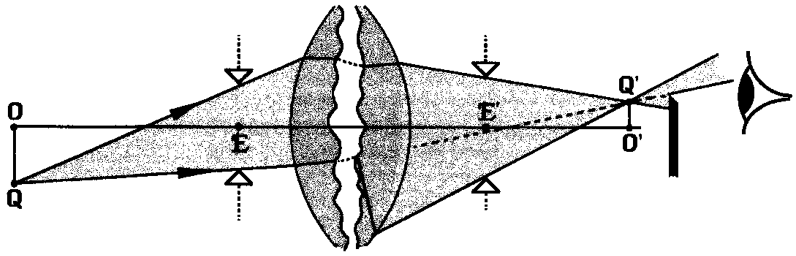

Abbildung 3.3: Mit einer scharfen Blende wird nach der schärfsten Abbildung Q′ eines hellen Punktes Q gesucht. (Aus [19])

3.1.2 Foucault-Methode

Für eine punktförmige Lichtquelle soll die optimale Fokusposition gefunden werden. Mit einer scharfen Blende (*knife edge*) im Bereich der Bildebene kann man die Lichtquelle beobachten. Abbildung 3.3 zeigt den Aufbau des Verfahrens. Solange die Blende sich im unscharfen Bereich befindet, sieht man bei vertikaler Bewegung einen dunklen und einen hellen Teil des abgebildeten Strahlkreises. In der fokussierten Position wird bei vertikalen Bewegungen der gesamte Kreis schlagartig dunkel bzw. hell.

3.2 Rechnerbasierte Verfahren

Die bisher gezeigten Verfahren stellen nur Hilfsmittel dar, um vom MVS als scharf beurteilte Bilder zu erreichen. Unbeaufsichtigte, also automatische Meßverfahren lassen sich aus diesen Verfahren nur schwer entwickeln.

Mit Methoden der Bildverarbeitung lassen sich auf Rechnern die Schärfe eines Bildes wie folgt messen.

3.2.1 Fouriertransformation

Ein in der Literatur vorgeschlagener Schärfefilter leitet sich aus der Fouriertransformation ab [12]. Dabei wird auf das Eingabebild eine Fouriertransformation angewendet. Es wird der hochfrequente Anteil des Betragsspektrums betrachtet und dessen Energie berechnet. Dies ergibt für das *ganze* Bild einen Schärfewert.

Die Funktionsweise dieses Filters ist dadurch charakterisiert, daß ein scharfes Bild durch scharf abgebildete Kanten und andere lokale Bildstrukturen viele hohe Frequenzanteile enthält. In der Praxis ist dieser Filter aber für unsere Anwendung nicht geeignet. Die Fouriertransformation liefert im Betragsspektrum keine Ortsinformation, sondern es wird nur ein globaler Schärfewert für das gesamte Bild berechnet. Dieser Filter liefert für mehrere Bilder unterschiedlicher Schärfe nur ein Maximum, falls die gesamte aufgenommene Szene gleich weit von der Kamera entfernt ist. Sind einige Objekte der Szene in unterschiedlicher Entfernung zur Kamera, liefert die Fouriertransformation an verschiedenen Fokuspositionen lokale Maxima. Wir wollen jedoch in unserem Verfahren keine A-priori-Annahmen über eine Szene machen. Zusätzlich ist die Fouriertransformation auf sequentiellen Rechnern eine sehr rechenaufwendige Operation. Auf unserem Bildverarbeitungssystem ist die Fouriertransformation nicht realisierbar, da das MV200-System kein Modul für eine Fouriertransformation zur Verfügung stellt. Sie kann auch mit den vorhandenen Modulen aufgrund der vorhandenen 8-Bit-Arithmetik nicht genügend genau approximiert werden.

3.2.2 Tenengrad

Ein bekannter Schärfefilter ist der sogenannte Tenengrad[1, 11, 12, 21, 25]. Der Tenengrad ist definiert durch:

$$\begin{aligned}
S(I_{xy}) &= |\nabla I_{xy}| = \sqrt{(Sobel_x * I_{xy})^2 + (Sobel_y * I_{xy})^2} && (3.1)\\
ten(I_{xy}) &= \sum_{x_n=x}^{x+N} \sum_{y_n=y}^{y+N} S(I_{x_n y_n})^2, && (3.2)\\
& \text{wobei} \\
Sobel_x &= \frac{1}{4}\begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}, && (3.3)\\
Sobel_y &= \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}. && (3.4)
\end{aligned}$$

Die Schärfefunktion besteht aus dem Betrag des Gradienten, der über ein Fenster der Größe N aufsummiert wird. In der ursprünglichen Definition von Tenenbaum wird bei der Summierung noch ein Schwellenwert definiert und nur Betragswerte größer als dieser Schwellenwert T werden aufsummiert [21]:

$$ten'(I_{xy}) = \sum_{x_n=x}^{x+N} \sum_{y_n=y}^{y+N} S(I_{x_n y_n})^2 \quad \text{mit } S(I_{x_n y_n}) > T \qquad (3.5)$$

In der Literatur [12] wurde aber festgestellt, daß der Schwellenwert 0 im wesentlichen die gleichen Maxima liefert.

Der Betrag des Gradienten $S(I_{xy})$ gibt eine Antwort größer als Null, falls sich im Einzugsbereich des Gradienten eine Grauwertkante befindet. Dabei reagiert der Sobel in X-Richtung auf senkrechte, der Sobel in Y-Richtung auf waagerechte und beide auf diagonale Kanten. Durch die Summierung über einen in der Regel quadratischen Bereich um den Aufpunkt erhält man den Tenengrad. Dieser Filter liefert 0 für ein konstantes Bild und eine hohe Antwort für eine Grauwertkante im Bild. Je schärfer die Grauwertkante ist, desto besser antworten die beiden Sobel-Operatoren. Durch die Quadrierung wird dieser Effekt noch verstärkt, so daß dieser Filter bei scharfen Kanten die maximale Filterantwort liefert. Die Summierung über das lokale Fenster führt zu einer Unempfindlichkeit gegenüber Bildrauschen.

Das Fenster der Summierung sollte recht klein gewählt werden, damit unterschiedliche Kanten nicht gleichzeitig in diesem Fenster auftreten. Damit vermeidet man mehrere Maxima in der Filterantwort, wenn die Kanten unterschiedlich weit entfernt sind. In [12] wurden Fenstergrößen von 10×10 bis 20×20 Pixel verwendet, da durch die Vergrößerungseffekte der Fokussierung sonst das Objekt aus dem Filterbereich herausläuft. Wir lösen dieses Problem durch eine Kalibrierung (siehe Abschnitt 4.2) und können daher mit kleineren Fenstergrößen arbeiten. Wir haben in Versuchen mit Fenstern der Größe 4×4 und 8×8 gearbeitet und gute Ergebnisse erzielt: Bei hoher Ortsauflösung ergaben sich signifikante Maxima.

Dieser Filter ist ohne große Probleme auf unserem Bildverarbeitungssystem implementierbar (siehe Kapitel 6). Die Rechenzeit ist auch auf herkömmlichen Rechnern gering, auf einem 512×512-Bild benötigt eine SUN Ultra 1 unter Khoros ca. 5 Sekunden. Unser Bildverarbeitungssystem berechnet diesen Filter in nur 0,2 Sekunden. Für $S(I_{xy})^2$ werden zwei Faltungen für den Sobel-Operator, zwei Multiplikationen für die Quadrierungen der Sobelantworten und eine Addition benötigt. Für die Summierung über die Umgebung werden N^2 Additionen für ein $N\times N$-Fenster benötigt.

3.2.3 Graustufenvarianz

Als ein weiterer Schärfefilter kann die Berechnung der Graustufenvarianz angesehen werden. Die Graustufenvarianz σ^2 eines Bildes ist definiert durch:

$$\mu(x,y)_I = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} I(x+i,y+j) \tag{3.6}$$

$$\sigma^2(x,y)_I = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(I(x+i,y+j)-\mu(x,y))^2 \tag{3.7}$$

Wir berechnen den Mittelwert μ einer lokalen Umgebung durch Aufsummieren der Grauwerte in einem Fenster und anschließender Normierung auf die Größe des Fensters. Im zweiten Schritt ziehen wir von unserer ausgewählten Umgebung den Mittelwert ab und summieren diese Werte auf. Eine Normierung auf die Fenstergröße liefert uns die Graustufenvarianz dieser Umgebung.

Dieser Filter liefert für homogene Bildteile dieser Umgebung ebenso wie der Tenengrad keine relevante Filterantwort, da der Mittelwert gleich dem Grauwert im ausgewählten Fenster ist. Sind Grauwertkanten im Fenster vorhanden, liefert die Graustufenvarianz positive Filterantworten. Wie der Tenengrad liefert auch die Graustufenvarianz erwartungsgemäß bei scharfen Grauwertkanten die größten Filterantworten.

Die Graustufenvarianz ist wie der Tenengrad auf unserem Bildverarbeitungssystem implementierbar. Durch die fehlenden Faltungen ist ihr Rechenaufwand auf sequentiellen Rechnern etwas günstiger als der Tenengrad. Die Laufzeit ist in der Größenordnung vergleichbar.

3.3 Vergleich der Filter

Einige der vorgestellten Schärfekriterien sind auf der vorhandenen Ausstattung nicht realisierbar. Die Methode der Polarisationsfilter und die Focault-Methode scheiden aus, da für diese Verfahren die verwendete Kamera modifiziert werden muß.

Die Fouriermethode liefert uns ohne Änderungen an den verwendeten Kameras einen Schärfewert. Dieser läßt sich jedoch nur global für das gesamte Bild bestimmen. Bei verschiedenen Tiefen in der aufgenommenen Szene erhalten wir dadurch mehrere Maxima in der Schärfefunktion. Weiterhin können diese

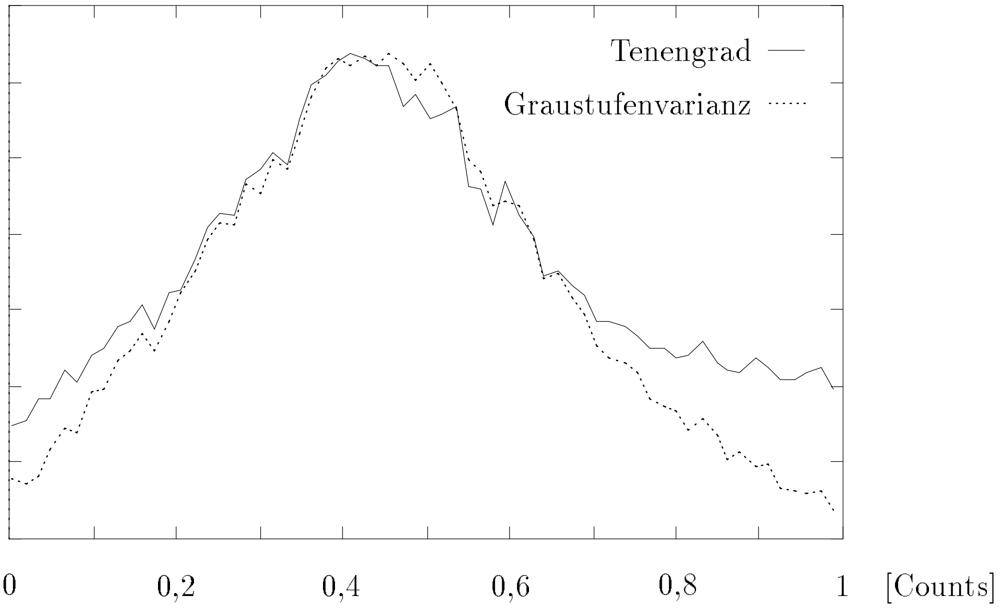

Abbildung 3.4: Filterantworten von Tenengrad und Graustufenvarianz über die Fokusstellung an einem Bild (auf gleiche Größen skaliert). Der Tenengrad zeigt ein schmaleres Maximum als die Graustufenvarianz.

Maxima für kleine Bildstrukturen im hochfrequenten Rauschanteil des Bildes verlorengehen. Diese Methode ist nur für eine Szene mit einer geringen relativen Tiefe zur Kamera anwendbar.

Der Tenengrad und die Graustufenvarianz sind lokale Schärfefilter. Sie lassen sich beide auf unserer Hardware gut realisieren und führen zu einer optimal kurzen Laufzeit. In der Literatur wird der Tenengrad als geeigneter gegenüber der Graustufenvarianz angesehen. Ein Nachteil der Graustufenvarianz sind gelegentlich in Experimenten auftretende globale Maxima an unscharfen Bildteilen [12].

In einem Experiment haben wir beide Filter auf das gleiche Bild angewendet. Das Ergebnis zeigt Abbildung 3.4. Es ergibt sich ein breites Maximum von fast 0,2 Counts bei der Graustufenvarianz und ein schmaleres Maximum von unter 0,1 Counts beim Tenengrad. Wir möchten an Kanten ein möglichst schmales und deutliches Maximum in der Filterantwort haben. Ein schmales Maximum impliziert einen geringen Tiefenbereich als Entfernungswert. Ein hohes Maximum ist besser zu lokalisieren und gibt auch meist schlankere Kurven. Weiterhin sollen die Filter unempfindlich gegen Rauschen in Bildern sein. Da die Graustufenvarianz und der Tenengrad eine lokale Umgebung eines Aufpunktes betrachten, wird vorhandenes Bildrauschen unterdrückt. Weitere, hier nicht vorgestellte, Schärfefilter lieferten in der Literatur schlechtere Ergebnisse als die hier gezeigten [12].

Der Tenengrad ist der geeignetste Schärfefilter. Er liefert in unseren Experimenten schmalere Maxima als die Graustufenvarianz und ist unempfindlich gegen Bildrauschen. Diesen verwenden wir dann auch bei der Implementierung des Algorithmus dieser Arbeit.

Kapitel 4

Bildpunktverschiebungen durch Fokussierung

In Kapitel 2.2 wurde ein Modell der Kamera eingeführt. Bei genauer Betrachtung dieses Modells sehen wir, daß Fokussierung zu einer Änderung der Bildgröße führt. Abbildung 4.1 zeigt dies noch deutlicher: Der Weltpunkt P wird je nach Fokusposition auf Kreise mit den Mittelpunkten P_∞ und $P_{1\,\mathrm{m}}$ abgebildet. Aus Gleichung 2.1 ergibt sich $b = B\frac{g}{G}$. Eine Änderung der Bildweite impliziert eine entsprechende Änderung der Bildgröße. Weitere Änderungen können durch ungenau gefertigte Linsen oder nicht zentrisch in die Kamera eingepaßte Linsen entstehen [16].

Da wir aber bei verschiedenen Fokuspositionen die Abbildungen gleicher Weltpunkte vergleichen möchten, müssen wir diese Verzerrungen kennen, also die Verschiebungen der Weltpunkte in der Bildebene abhängig von der Fokusposition kalibrieren.

Die Bildpunktverschiebungen bestehen aus einer Größenänderung um den Bildhauptpunkt der Kamera. Vorerst nehmen wir an, daß das Zentrum der

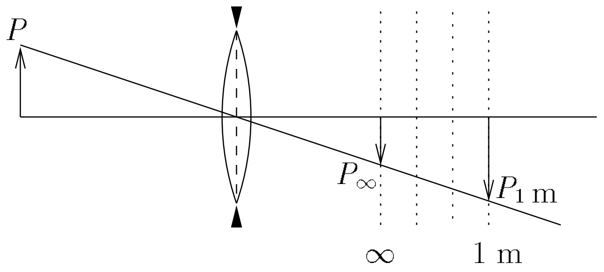

Abbildung 4.1: In Fokusstellung ∞ wird P auf P_∞ abgebildet, dies ist näher am Bildhauptpunkt als $P_{1\,\mathrm{m}}$.

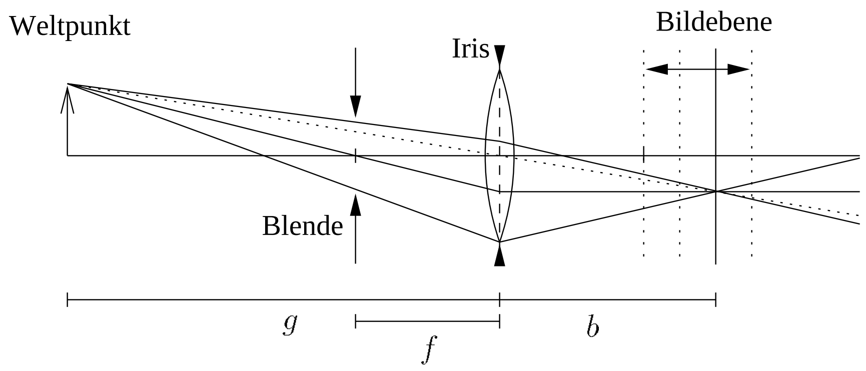

Abbildung 4.2: Telezentrische Optik durch Ergänzung einer Blende im vorderen Brennpunkt der Linse. Eine Änderung der Fokussierung durch Bewegung der Bildebene führt zu keinen Bildpunktverschiebungen (nach [23]).

Vergrößerung bei der Fokussierung nicht fest bleibt. Dies kann z.B. durch ungenau eingepaßte Linsen hervorgerufen werden.

4.1 Telezentrische Optik

Die Vergrößerung des Bildes kann nach [23] durch eine Blende im vorderen Brennpunkt der Kamera verhindert werden.

In Abbildung 4.2 wird durch eine zusätzliche Blende das bisher verwendete Kameramodell telezentrisch gestaltet, das heißt es kommt beim Fokussieren zu keiner Größenänderung. Der Lichtkegel eines Weltpunktes fällt nun senkrecht auf die Bildebene. Eine Fokussierung durch die Bewegung der Bildebene führt nicht mehr zu unterschiedlich großen Abbildungen des Weltpunktes.

Die Größe der Blende bestimmt sich durch die Größe der Iris der verwendeten Kamera. Es muß sichergestellt sein, daß das komplette Strahlenbündel, das durch die vordere Blende tritt, auch die Iris passiert und nicht teilweise ausgeblendet wird. In [23] wurde diese Größe durch Versuchsreihen gewonnen.

Diese Optik kann insgesamt gleich große Bilder bei verschiedenen Fokuspositionen sicherstellen.

Da wir bei der verwendeten Kamera durch den Zoom die Brennweite des Linsensystems ändern können, verschiebt sich auch der vordere Brennpunkt des Linsensystems. Diese Änderungen müßten wir bei der Realisierung einer telezentrischen Optik durch eine Mechanik der Zoomstellung anpassen.

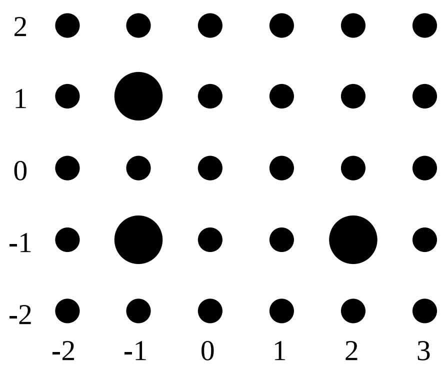

Abbildung 4.3: Ausschnitt des verwendeten Kalibrierungsmusters.

Die Zoom-Linsen haben eine sehr kleine vordere Brennweite, so daß bei diesen Linsensystemen die Blende innerhalb des Kameragehäuses angebracht werden muß. Wir haben bisher den vorderen Brennpunkt unserer Kamera nicht bestimmt, da diese auch eine Zoom-Optik besitzt, wird sich der vordere Brennpunkt vermutlich im Kameragehäuse befinden. Weiterhin können wir bei unserer Kamera die Iris steuern, wir müssen also auch die Öffnung der zusätzlichen Blende abhängig von der Irisöffnung ändern.

Insgesamt ergäbe sich eine komplizierte Mechanik, die je nach Zoom- und Irisstellung die Blendenposition und -öffnung steuert. Da wir die vorhandene Kamera nicht verändern können, müssen wir ein anderes Verfahren verwenden.

4.2 Kalibrierung

Da gleiche Bildgrößen zum Vergleich der Bilder verschiedener Fokuspositionen benötigt werden, werden die auftretenden Verschiebungen gemessen. Dafür wird ein Kalibrierungsbild mit schwarzen Kreisen bei verschiedenen Fokuspositionen aufgenommen. Abbildung 4.3 zeigt einen Teil des Kalibrierungsmusters und die verwendeten Koordinaten. Drei Kreise des Kalibrierungsmusters sind größer als die übrigen Kreise, damit wir ein Koordinatensystem auf dem Kalibrierungsmuster definieren können. Wir haben in ersten Überlegungen eine beliebige Orientierung des Kalibrierungsmusters angenommen. Die drei Kreise sind so angeordnet, daß sich die Position im Koordinatensystem des Kalibrierungsmusters immer eindeutig bestimmen läßt. Für das Muster haben wir Kreise gewählt, da diese bei unscharfer Aufnahme immer noch einfach und vor allem genau zu detektieren sind. Tests mit anderen Mustern, wie z.B. Gittern, führten zu sehr unbefriedigenden Ergebnissen,

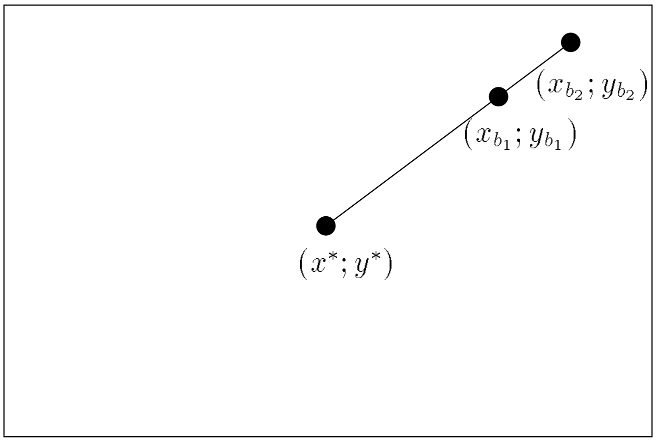

Abbildung 4.4: Im Modell der Verschiebung bleibt $(x^*;y^*)$ bei beiden Fokuspositionen b_1 und b_2 fest, $(x_{b_1};y_{b_1})$ verschiebt sich bei einer anderen Fokusstellung auf $(x_{b_2};y_{b_2})$.

die einzelnen Gitterpunkte waren in der unscharfen Aufnahme nicht mehr zu extrahieren.

Für 30 Fokuspositionen von 0 bis 1 wird das Kalibrierungsmuster aufgenommen. Wir extrahieren die Schwerpunkte und deren Position relativ zum Zentrum des Musters. Damit stellen wir für verschiedene Fokuspositionen die Verschiebung der Punkte fest.

Als Modell der Verschiebung nehmen wird ein Zentrum an, um das in Abhängigkeit von der Fokusposition die Punkte vergrößert werden. Für zwei Fokuspositionen b_1 und b_2 bleibt der Mittelpunkt der Verschiebung $(x^*;y^*)$ konstant, für zwei in den Weltkoordinaten gleiche Punkte $(x_{b_1};y_{b_1})$ und $(x_{b_2};y_{b_2})$ an den Fokuspositionen b_1 und b_2 gilt:

$$(x_{b_1}-x^*)m_x = (x_{b_2}-x^*), \tag{4.1}$$
$$(y_{b_1}-y^*)m_y = (y_{b_2}-y^*), \tag{4.2}$$

mit m_x und m_y als Vergrößerungsfaktoren in x- und y-Richtung.

Für alle entsprechenden Punkte des Bildes $(x_{i,b_j};y_{i,b_j})$ und zwei Fokuspositionen b_1 und b_2 bekommen wir zwei überbestimmte Gleichungssysteme:

$$\begin{pmatrix} x_{0,b_1} - x^* & 0 \\ 0 & y_{0,b_1} - y^* \\ x_{1,b_1} - x^* & 0 \\ 0 & y_{1,b_1} - y^* \\ \vdots & \vdots \end{pmatrix} \begin{pmatrix} m_x \\ m_y \end{pmatrix} = \begin{pmatrix} x_{0,b_2} - x^* \\ y_{0,b_2} - y^* \\ x_{1,b_2} - x^* \\ y_{1,b_2} - y^* \\ \vdots \end{pmatrix}, \tag{4.3}$$

$$\begin{pmatrix} 1 - m_x & 0 \\ 0 & 1 - m_y \\ 1 - m_x & 0 \\ 0 & 1 - m_y \\ \vdots & \vdots \end{pmatrix} \begin{pmatrix} x^* \\ y^* \end{pmatrix} = \begin{pmatrix} x_{0,b_2} - x_{0,b_1} \\ y_{0,b_2} - y_{0,b_1} \\ x_{1,b_2} - x_{1,b_1} \\ y_{1,b_2} - y_{1,b_1} \\ \vdots \end{pmatrix}. \tag{4.4}$$

Damit kann mit Gleichung 4.3 für ein vorgegebenes Bildzentrum $(x^*; y^*)$ die Vergrößerungsfaktoren m_x und m_y berechnet werden. Aufgelöst nach x^* und y^* ergibt sich nach Gleichung 4.4 für vorgegebenes m_x und m_y ein Bildzentrum $(x^*; y^*)$. Als erste Näherung für $(x^*; y^*)$ kann die Mitte des Kamerabildes angenommen werden. Diese liegt bei $(378; 288)$ Pixel.

Mit dem Least-Square-Verfahren können wir diese überbestimmten Gleichungssysteme lösen, wie z.B. gemäß $A^tAx = A^tb$. Da die Vergrößerungsfaktoren nahe 1 sind, ist dieses numerisch instabile Verfahren nicht sinnvoll. Wir verwenden daher die numerisch stabilere QR-Zerlegung des Householderverfahrens in eine unitäre Matrix Q und eine obere Dreiecksmatrix R für die Berechnung [20]. Iterativ werden so m_x, m_y und $(x^*; y^*)$ berechnet.

Als erstes Ergebnis zeigte sich, daß die Vergrößerungsfaktoren m_x und m_y gleich sind. Dies ist bei radialsymmetrischen Linsen zu erwarten, konnte aber aufgrund möglicher Linsenfehler nicht vorausgesetzt werden. Weiterhin könnte das Seitenverhältnis der CCD-Pixel nicht gleich sein, was zu unterschiedlichen Faktoren für die x- und y-Richtung führen würde. Wir können also einen Vergrößerungsfaktor m für die x- und y-Richtung verwenden, und erhalten als Gleichungssystem:

$$\begin{pmatrix} x_{0,b_1} - x^* \\ y_{0,b_1} - y^* \\ x_{1,b_1} - x^* \\ y_{1,b_1} - y^* \\ \vdots \end{pmatrix} \begin{pmatrix} m \end{pmatrix} = \begin{pmatrix} x_{0,b_2} - x^* \\ y_{0,b_2} - y^* \\ x_{1,b_2} - x^* \\ y_{1,b_2} - y^* \\ \vdots \end{pmatrix} \tag{4.5}$$

$$\begin{pmatrix} 1-m \\ 1-m \\ \vdots \end{pmatrix} (x^*) = \begin{pmatrix} x_{0,b_2} - m x_{0,b_1} \\ x_{1,b_2} - m x_{1,b_1} \\ \vdots \end{pmatrix} \tag{4.6}$$

$$\begin{pmatrix} 1-m \\ 1-m \\ \vdots \end{pmatrix} (y^*) = \begin{pmatrix} y_{0,b_2} - m y_{0,b_1} \\ y_{1,b_2} - m y_{1,b_1} \\ \vdots \end{pmatrix} \tag{4.7}$$

Es ist mit Gleichung 4.5 und N Kalibrierungspunkten nur noch ein $1\times 2N$-Gleichungssystem zu lösen, und mit den Gleichungen 4.6 und 4.7 zwei $1\times N$-Gleichungssysteme.

4.3 Implementierung der Kalibrierung

Wir nehmen eine Bildfolge des Kalibrierungsbildes auf. Dabei stellen wir sicher, daß

- das Kalibrierungsmuster gleichmäßig beleuchtet ist,
- das Kalibrierungsmuster möglichst bildfüllend aufgenommen wird,
- das Kalibrierungsmuster orthogonal zur Kameraachse orientiert ist.

Um die Kalibrierungspunkte sicher extrahieren zu können, müssen wir das Muster mit gleichmäßiger Beleuchtung aufnehmen. Da wir mit möglichst vielen Kalibrierungspunkten das gesamte Kamerabild einbeziehen wollen, sollte das Muster das gesamte Bild ausfüllen. Das Muster muß orthogonal zur Kameraachse orientiert sein, da sich sonst unterschiedlich große Punktgrößen in den extrahierten Punkten ergeben. Bei mittlerer Zoomeinstellung ist das Muster ca. 1,5 m von der Kamera entfernt aufgebaut.

Wir nehmen 30 Bilder linear zwischen den Fokuspositionen $b = 0$ (∞) und $b = 1$ (1 m) auf. Bei diesen Aufnahmen führen wir die Bewegung der Linse sowohl von 0 nach 1 als auch von 1 nach 0 durch. Die Bilddateien werden mit Fokus- und Zoomposition indiziert.

Die Auswertung der Kalibrierungsbilder erfolgt unter dem Bildverarbeitungssystem Khoros. Mit einem in die Khorosumgebung integrierten C-Programm werden die Kalibrierungsbilder sequentiell wie folgt verarbeitet:

Zur Detektion der Kreise wenden wir einen Schwellenwert auf das Bild an. Für die Detektion des Mittelpunktes und der Größe der Kreise wenden wir einen auf das Problem angepaßten Schwerpunktoperator an. Dieser liefert uns die Fläche zusammenhängender Bereiche in Pixel und mit Subpixelgenauigkeit den Schwerpunkt dieser Fläche. Über die Fläche finden wir die drei größeren Kreise des Kalibrierungsmusters. Damit wird das Koordinatensystem des Kalibrierungsmusters bestimmt. Durch ihre Anordnung können wir jetzt alle Punkte eindeutig identifizieren. Zu Testzwecken gibt der Schwerpunktoperator noch ein Bild mit den gefundenen Schwerpunkten aus, mit denen wir die gefundenen Schwerpunkte visuell überprüfen können. Die gefundenen Schwerpunkte werden als Ascii-Datei ausgegeben. Jede Zeile enthält die x- und y-Koordinate eines gefundenen Schwerpunktes als Fließkommazahlen und die Größe des zusammenhängenden Bereiches.

Die gefundenen Schwerpunkte eines Bildes werden mit einer weiteren Funktion um das Koordinatensystem des Kalibrierungsmusters ergänzt. Dazu suchen wir in den gefundenen Schwerpunkten nach den drei großen Kreisen und geben diesen die Koordinaten (-1;1), (-1;-1) und (2;-1). Daraus berechnen wir die Vektoren in X- und Y-Richtung, und berechnen für jeden weiteren Punkt dessen Koordinate. Diese wird auf ganze Zahlen gerundet. Wir haben keine Geradenerkennung oder ähnliches zur weiteren Verbesserung des Verfahrens implementiert, da in den Aufnahmen die Koordinaten immer korrekt waren und auch keine falschen Punkte detektiert wurden. Nur Punkte nahe am Bildrand werden nicht in die Kalibrierungsdaten aufgenommen, da sich darunter auch halbe Punkte oder der Hintergrund befinden können, für die wir keinen sinnvollen Mittelpunkt berechnen können.

Die Schwerpunkte sowie die Koordinaten auf dem Kalibrierungsmuster und die Fokusposition aller Kalibrierungsbilder werden als eine Datei realisiert, an die die Ergebnisse der einzelnen Kalibrierungsbilder angehängt werden.

Aus diesen Daten bilden wir die Gleichungen 4.5, 4.6 und 4.7 und lösen diese. Die QR-Zerlegung wurde mit der Bibliotheksfunktion `dgels()` aus der *Sun Performance Library* realisiert. Nach 4 Iterationen zeigte sich keine Änderung mehr an den berechneten m, x^* und y^*. Als Ergebnis erhalten wir eine Textdatei mit den Vergrößerungsfaktoren aller kalibrierten Fokuspositionen. Diese Datei wird in das eigentliche Meßprogramm (siehe Kapitel 6) eingelesen und zur Korrektur der Bildgröße auf die Normgröße genutzt.

Tests der Kalibrierung auf den Kalibrierungsbildern ergeben nur minimale Abweichungen der kalibrierten Vergrößerung: Abbildung 4.5 zeigt das Er-

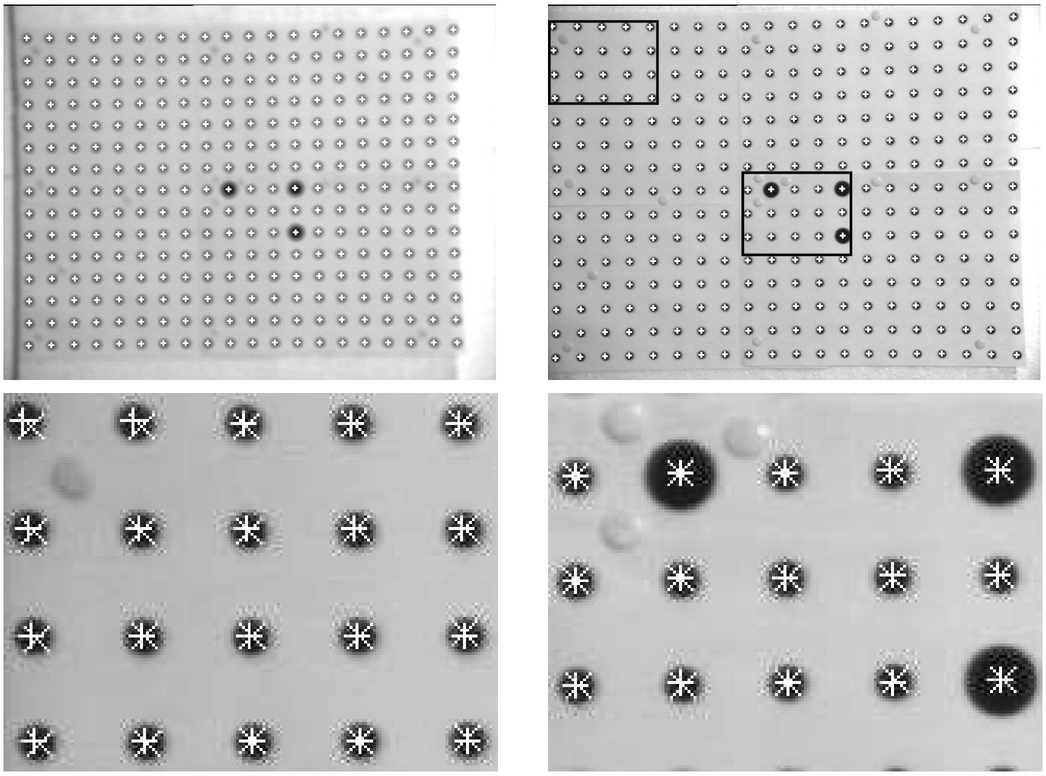

Abbildung 4.5: Test der Kalibrierung. Oben links das bei Fokusstellung 0 (unscharf) aufgenommene Bild, oben rechts bei Fokusstellung 1 (scharf). Gefundene Schwerpunkte sind mit + markiert. Unten rechts und links die markierten Ausschnitte des rechten Bildes. Gefundene Punkte sind wieder mit '+' markiert, '×' zeigt die Punkte des linken Bildes um (378;288) mit Faktor 1,073 vergrößert.

gebnis einer Kalibrierung für Fokusstellung 0 (unscharfes Bild links oben) und 1 (scharfes Bild rechts oben). Das Kalibrierungsmuster steht bei dieser Aufnahme auf dem Kopf. Der Schwerpunktoperator liefert die in den ersten beiden Bildern mit '+' markierten Mittelpunkte. Diese Punkte sind in der Demonstration nur pixelgenau eingetragen. Der Schwerpunkt liefert für die Berechnung die Koordinaten der Schwerpunkte mit Subpixelgenauigkeit. Die Kalibrierung liefert für diese beiden Bilder bei iterativer Anwendung der Gleichungen 4.5, 4.6 und 4.7 einen Vergrößerungsfaktor von 1,073168 um das Vergrößerungszentrum (393;279), bei einmaliger Anwendung von Gleichung 4.5 für das Zentrum (378;288) liefert die Kalibrierung einen Faktor von 1,073053. Die unteren beiden Bilder zeigen die linke obere Ecke und die Mitte des oberen rechten Bildes. Wie auch in den oberen Bildern sind die gefundenen Schwerpunkte mit '+' markiert. Mit '×' haben wir die Punkte des rechten Bildes markiert, die um (378;288) mit Vergrößerungsfaktor 1,073053 vergrößert werden.

Im Bildzentrum liegt der Fehler bei maximal einem Pixel. Im Außenbereich ergibt sich eine Abweichung von bis zu drei Pixeln. In die Kalibrierung gehen die äußersten Punkte in der obersten Zeile und der linken Spalte nicht ein, erst die weiter innen liegenden Punkte werden für die Berechnung der Kalibrierung herangezogen. Die inneren Punkte der oberen linken Ecke sind wie die mittleren Punkte auch mit einem Fehler von maximal einem Pixel behaftet.

Dieses Ergebnis zeigt weiterhin, daß in einem Bereich von 14 Pixeln um den Bildmittelpunkt gar keine Vergrößerung nötig ist, da der Fehler in diesem Bereich kleiner als ein Pixel bleibt:
$x + 1 = 1,073168 \cdot x \Rightarrow x = \frac{1}{1,073168-1} = 13,7$

Die verwendete Kamera hat gut aufeinander abgestimmte Linsen, das heißt es treten keine weiteren Verzerrungen außer der beschriebenen Vergrößerung auf. Der Mittelpunkt bleibt im wesentlichen konstant, die iterative Kalibrierung liefert für den Mittelpunkt Werte von (395;280) bis (391;272). Ausreißer ergeben sich nur für fast identische Bilder, dabei werden zum Beispiel für die Fokuspositionen 0 und 0,034 iterativ das Bildzentrum (408;273) und der Faktor 1,003979 ausgegeben. Für den Mittelpunkt (378;288) nur mit Gleichung 4.5 erhalten wir immer noch sehr gute Ergebnisse, wie Abbildung 4.5 zeigt. Wenn der Mittelpunkt gar nicht berechnet wird, sondern fest auf (378;288) gesetzt wird, verringert dies wesentlich den Rechenaufwand der Kalibrierung. Zum Beispiel benötigen 10 Iterationen der Berechnung des Mittelpunktes einen 20 mal so hohen Rechenaufwand.

Abbildung 4.6 zeigt den Verlauf der Kalibrierungsfaktoren über die Fokuspositionen. Die Kurven beginnen bei 0,060 Counts und enden bei 1,061 Counts,

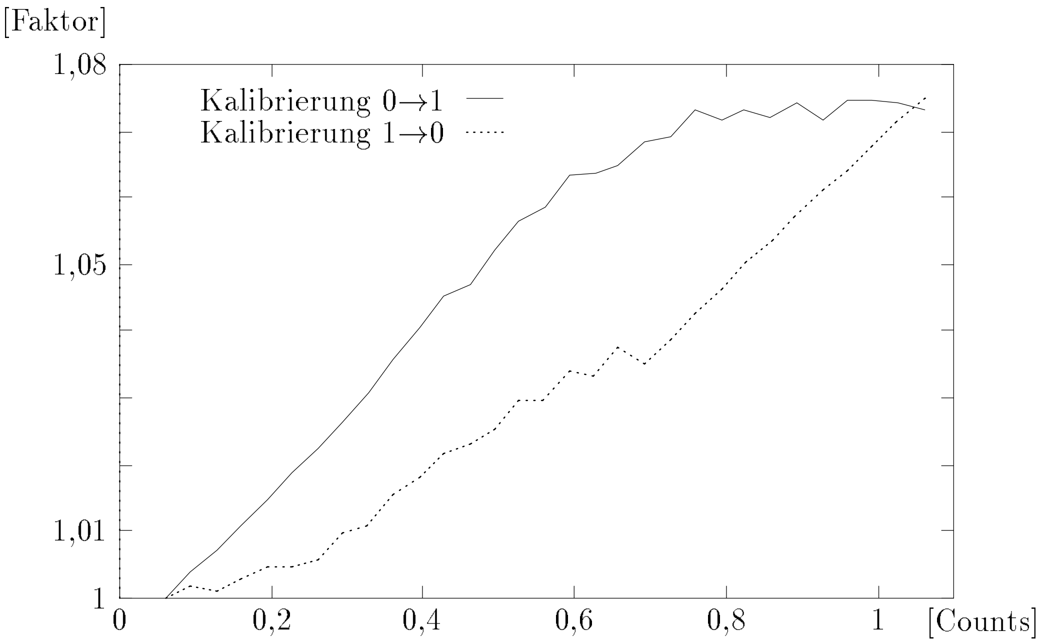

Abbildung 4.6: Kalibrierungsergebnis der Vergrößerungsfaktoren von 0 (∞) nach 1 (1 m) und 1 nach 0. Die Kalibrierungsfaktoren zeigen sehr unterschiedliches Verhalten für die beiden Richtungen der Fokusänderung.

da die angefahrenen Positionen mit `wherelens` (siehe Abschnitt 5.4) ausgelesen wurden, und dies leicht geänderte Fokuspositionen liefert.

Auffällig ist die Abweichung der kalibrierten Vergrößerungen bei der Änderung der Fokussierungsrichtung. Erwartet wird eine Abweichung durch ein Getriebespiel: Die Vergrößerungsfaktoren ändern sich bei einer Richtungsumkehr des Fokusmotors nicht, erst bei weiterer Positionierung dieses Motors läuft die Linse dann mit, und ändert damit auch die Vergrößerungsfaktoren. Beobachtet wird aber genau das Gegenteil: Die Vergrößerungsfaktoren ändern sich direkt bei einer Richtungsumkehr, und bleiben dann gegen Ende des Fokussiervorgangs konstant. In Abbildung 4.6 zeigt sich, daß sich bei der Fokussierung von 0 nach 1 der Vergrößerungsfaktor bis 0,7 Counts beinahe linear ändert. Von 0,7 bis 1 bleibt der Faktor konstant auf 1,75. In der Rückrichtung ändert sich der Vergrößerungsfaktor bis 0,2 Counts wieder ungefähr linear, danach bleibt der Vergrößerungsfaktor konstant.

Vermutlich bewegen sich bei einer Bewegung der Fokuslinsen auch die Zoomlinsen, so daß eine Größenänderung durch die Fokuslinsen durch diese Bewegung der Zoomlinsen vorweggenommen wird. Da keine Dokumentation zur Mechanik der Kamera vorliegt, läßt sich dieser Effekt nicht genauer erklären.

Kapitel 5

Beschreibung der Hardware

Im folgenden Kapitel wird das verwendete echtzeitfähige Bildverarbeitungssystem und die Möglichkeiten dieses Systems vorgestellt.

Bildverarbeitung in Echtzeit stellt sehr hohe Anforderungen an die verwendeten Rechnersysteme. Die Bilder werden in der Regel mit Videokameras aufgenommen und müssen dann im Rechner digitalisiert werden. Die Digitalisierung muß innerhalb eines Bildwechsels erfolgen, bei einem PAL-Signal innerhalb $\frac{1}{25}$ Sekunde. Weiterhin müssen die Bilder mit dieser Bildwiederholfrequenz auch verarbeitet werden können. Herkömmliche Rechner sind mit der auftretenden Datenrate meist überfordert. Erst die letzte Rechnergeneration, wie z.B. SUN Ultrasparc, kann mit dieser Datenmenge umgehen, also Berechnungen in Echtzeit auf Bilddaten durchführen.

Spezielle Bildverarbeitungsrechner können mit voller Videobandbreite die Bilder verarbeiten und somit Echtzeitverarbeitung erst ermöglichen. Spezialisierte Prozessoren übernehmen bei diesen Rechnern die rechenaufwendigen Bildverarbeitungsschritte.

5.1 Bildverarbeitungssystem

Das Bildverarbeitungssystem besteht aus einer SUN-Workstation, im weiteren Hostrechner genannt, die über einen VME-Bus den Pipelinerechner Datacube MaxVideo 200 (MV200-System) steuert. Die Bilder werden mit einer CCD-Kamera aufgenommen. Das MV200-System digitalisiert das Videosignal dieser Kamera und führt die zeitaufwendigen Rechenoperationen durch. Der Hostrechner steuert diese Rechenoperationen. Für die Manipulation des Zoom, des Fokus und der Iris der Kamera ist am VME-Bus eine

Abbildung 5.1: Das verwendete Bildverarbeitungssystem. Vorn steht der Kamerakopf, rechts dahinter das MV200-System und der PMAC. Der Hostrechner steht links, davor ein Monitor des MV200-Systems.

Kontrollerkarte PMAC der Firma Delta Tau angeschlossen. Der Zoom und der Fokus werden geregelt (*close loop*) und die Iris gesteuert (*open loop*). Die Kontrollerkarte steuert zusätzlich den Kamerakopf mit den 4 Freiheitsgraden Pan, Tilt und Vergenz. Abbildung 5.2 zeigt den Aufbau des Systems, und Abbildung 5.1 zeigt den Kamerakopf und die einzelnen Komponenten des Systems. Die größte Datenmenge läuft bei diesem System über das Videokabel der Kamera. Diese liefert 25 Bilder pro Sekunde mit einer Auflösung von 756x576 Pixeln. Sie hat damit eine Bandbreite von ca. 10 MB/s. Der VME-Bus hat demgegenüber eine Bandbreite von ca. 5 MB/s. Der Hostrechner kann über den VME-Bus einzelne Bilder der Kamera herunterladen. Messungen zeigen, daß Bilder der Datacube mit einer Transferrate von maximal 1 MB/s auf den Hostrechner heruntergeladen werden können.

Die Verbindungen des PMAC mit der Kamera sind bis zum Verstärker digital. Zur Kamera laufen analoge Steuersignale.

5.2 Der Hostrechner

Der Hostrechner ist in der aktuellen Konfiguration eine SUN Sparcstation 10/40 mit 64 MB Hauptspeicher. Diese Maschine läuft unter dem Betriebssystem SunOS 4.1.3, welches für die Arbeit mit dem Pipelinerechner unter der mitgelieferten C-Bibliothek „ImageFlow“ angepaßt wurde. Die Arbeit unter einer aktuellen Solaris-Version ist mit dem Pipelinerechner nicht möglich, da Anpassungen an Solaris nötig sind, diese aber nicht stabil laufen.

Die SUN ist mit einer Sbus-Karte mit VME-Bus ausgestattet. An diesem ist das MV200-System und die PMAC-Karte angeschlossen.

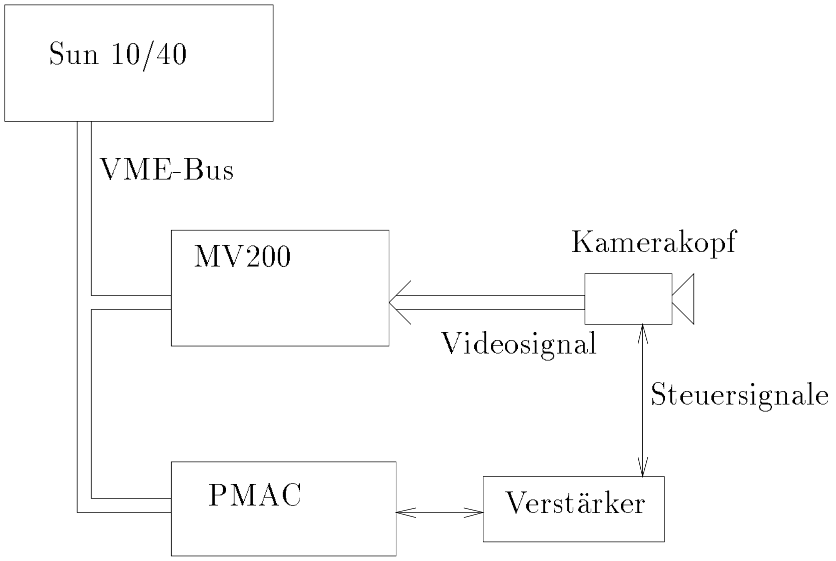

Abbildung 5.2: Prinzipieller Aufbau des Bildverarbeitungssystems

5.3 Das Bildverarbeitungssystem MV200

Die Implementierung des Fokussierungsalgorithmus wurde auf dem Pipelinerechner MV200 der Firma Datacube vorgenommmen. Nun sollen die Einheiten und Möglichkeiten dieses Systems aufgezeigt werden. Eine vollständige Einführung kann und will dieser Abschnitt nicht aufzeigen, dafür sei auf die Dokumentation von Datacube [4, 5] verwiesen. Eine kurze Einführung mit einigen Beispielen bietet [22].

Das MV200-System ist ein Pipelinerechner, der über den VME-Bus mit dem Hostrechner verbunden ist. Die Videoausgabe der Kameras kann in voller Bildwiederholfrequenz ausgelesen werden. Die Ergebnisbilder von Berechnungen können direkt über einen VGA-Monitor ausgegeben werden. Das MV200-System besteht aus insgesamt 6 MV200-Karten. Eine solche Karte beherbergt viele Module, die größtenteils fest verschaltet sind. Es lassen sich jedoch einige Eigenschaften dieser Module programmieren.

Das Herzstück einer MV200-Karte ist der sogenannte *crosspoint switch*, ein Schaltelement auf der Basisplatine (Architectural Adapter Version B, AB), das 32 Ein- und Ausgänge besitzt und diese fast beliebig miteinander verbinden kann. Es gibt einen AD-Wandler für die Digitalisierung von Videobildern (Analog Scanner, AS) und einen DA-Wandler für die Ausgabe von Bildern auf einem angeschlossenen VGA-Monitor (Analog Generator, AG). Sechs Speichermodule (Advanced Memory, AM) dienen zum Zwischenspeichern von

Rechenergebnissen. Eine Arithmetische Einheit (Arithmetic Unit, AU) kann Rechenoperationen ausführen. Ein Faltungsmodul (Advanced Pipeline Processor, AP) kann Faltungen mit einer 8×8-, 64×1- oder 1×64-Faltungsmaske durchführen. Auf dem AP kann ein sogenanntes MaxModule installiert werden. Auf einigen APs ist ein weiteres Faltungsmodul installiert (mmNMAC). Auf den ersten beiden MV200-Karten befindet sich ein sogenanntes Warp-Modul (MiniWarper, WX), das bilineare Bildtransformationen durchführen kann.

Die grobe Implementierung eines Algorithmus beschränkt sich auf das Auswählen der Verbindungen dieser Module durch den Crosspoint Switch. Weiterhin können an den einzelnen Schaltelementen Optionen konfiguriert werden. Ein so konfigurierter Algorithmus nennt sich Pipe. Diese Pipe arbeitet in der Regel mit 10 MHz Pixeltakt und liefert dadurch in kürzester Zeit Ergebnisse, die dem Durchlauf eines Pixels durch die Pipe entspricht.

Das MV200-System wird mit der C-Bibliothek „ImageFlow“ programmiert. Diese Bibliothek nimmt dem Programmierer viel Arbeit ab, so wird unter anderem die zeitliche Synchronisation der einzelnen Module weitestgehend von ImageFlow geregelt. Der Programmierer gibt die Elemente einer Pipe und deren Optionen an, und ImageFlow berechnet dann die Verzögerungen (*pipe delay*), mit der einzelne Elemente der Pipe später die Arbeit aufnehmen sollen.

Pipes können als einmalige Aktionen, als immer wiederkehrende Aktionen und als vordefinierte Pipes (Pipe-op Alternating Threads, PATs) konfiguriert werden. Einmalige Pipes (`one-shot pipe`) werden definiert, und am Ende der Definition dieser Pipe mit dem Kommando `dqArmPipe` werden alle zeitlichen Synchronisationssignale berechnet. Daraufhin wird diese Pipe mit `dqFirePipe` gestartet, und damit die eigentliche Rechnung durchgeführt. Nach Beendigung dieser Pipe, also nach der Verarbeitung eines Bildes, werden die berechneten Synchronisationssignale verworfen. Die Berechnung der Synchronisationssignale findet auf dem Hostrechner statt, so daß diese Berechnung die eigentliche Rechenleistung der MV200 mindert.

Immer wiederkehrende Aktionen (`continuous pipe`) nutzen einmal berechnete Synchronisationssignale kontinuierlich. Die Pipe wird definiert, mit `dqArmPipe` werden die Synchronsiationssignale berechnet, und mit `dqFirePipe` startet die Pipe. Diese Pipe läuft nun kontinuierlich, das heißt sie beginnt die Verarbeitung von neuem, wenn ein Bild verarbeitet wurde.

Die Quelle einer Berechnung in einer Pipe ist ein Speichermodul (wozu auch Digitalisierer und ähnliche Geräte wie z.B. Festplatten zählen), die Senke dieser Pipe liegt wieder auf einem Speichermodul beziehungsweise auf dem

Digital-Analogwandler der Grafikkarte (AG). Eine Pipe darf nie über ein Speichermodul hinweg seine Berechnung erstellen. Mehrere gleichzeitig definierte Pipes dürfen weiterhin nicht die gleichen Verbindungen des MV200-Systems nutzen. Insbesondere darf eine Pipe nicht mehrmals die gleiche Verbindung nutzen. Damit können in einer Pipe nur beschränkte Rechnungen durchgeführt werden. Eine erneute Definition von Pipes für eine Berechnung in mehreren Schritten verbraucht mit der Neuberechnung der Synchronisationssignale viel Rechenzeit auf dem Hostrechner. Eine Lösung für dieses Problem stellen die schon erwähnten PATs (Pipe-op Alternating Threads) dar. Diese werden bei ihrer Definition in einer Softwaresimulation von ImageFlow ausgeführt. Die berechneten Synchronisationssignale werden auf dem Hostrechner zwischengespeichert und bei Ausführung dieser Pipe in das MV200-System konfiguriert. Der Hostrechner berechnet also nicht mehr bei jeder erneuten Ausführung dieser Pipe die Synchronisationssignale neu. Verschiedene PATs dürfen auch gleiche Verbindungen nutzen. Weiterhin läuft die Neukonfigurierung der PAT direkt im Betriebssystem-Kern des Hostrechners ab. Der Kern schaltet nicht mehr für jeden neuen Start einer PAT zwischen Benutzer- und Kernelmodus um, was die Ausführungszeit verringert.

Mit mehreren externen Anschlüssen können Daten zwischen den einzelnen MV200-Karten übertragen werden. Diese können vom Benutzer frei verwendet werden, müssen dann aber in der generellen Konfigurationsdatei des MV200-Systems definiert werden. Weiterhin sind drei bidirektionale Verbindungen zwischen zwei benachbarten MV200-Karten möglich.

Das folgende Beispiel zeigt die Programmierung des MV200-Systems. Ein Ausschnitt des entwickelten Programms (siehe Abschnitt 6.6) ersetzt für alle in einem Binärbild gesetzten Pixel die Fokusposition.

```
/* Definition der Speicherbereiche                                  */
oSubstSrcSurf=dqCreateSurf(oAb00_Am03,AM_MEM_R8,512,512);
oSubstDstSurf=dqCreateSurf(oAb00_Am04,AM_MEM_R8,512,512);
i=32;                                  /*  Nummer der PAT, b=i/256    */

emBegPatDef();                         /*  Beginn der PAT             */
dqAttachSurf(oSubstSrcSurf,AM_XMT);    /*  Gateway bestimmen          */
amSetGateSysClkMult(oAb00_Am03,AM_XMT,2);/*  Option bei AM_XMT setzen   */
dqConnect(oAb00_Am03,AM_XMT,AM_XMT_OUT); /*  AM_XMT an  AM_XMT_OUT      */
dqConnect(oAb00_Am03,AM_XMT_OUT,AM_OP3); /*  AM_XMT_OUT an AM_OP3       */
dqSpecLogic(oAb00_Am03,AM_LOGIC3,0xffffffff,0);/* Identitaet           */
dqConnect(oAb00_Am03,AM_SLDN0_T0,AM_OUTPUT0);/*AM_SLDN0_T0 an AM_OUTPUT0*/

/* Nun weiter ueber AB                                              */
dqConnect(oAb00, DQ_IMX3,AB_OP04);      /* AM03-Ausgabe an AM04-Eingabe */

/* Nun auf AM04                                                     */
```

```
dqConnect(oAb00_Am04,AM_SPUP0,AM_INPUT0);/*  AM_SPUP0 an AM_INPUT0      */
dqConnect(oAb00_Am04,AM_INPUT0_D0,AM_MOSC_RCV_SRC);
/* Bit 0 der AM04-Eingabe an AM_MOSC_RCV weiterreichen                  */
dqSpecLogic(oAb00_Am04,AM_LOGIC4,0x1,0x0);
/* Konstante von AM_LOGIC0 erzeugen, an AM_RCV weiterleiten             */
dqSpecLogic(oAb00_Am04,AM_LOGIC0,0x00,(int)rint(64+i*3.0/4.0));
dqSetALUOp(oAb00_Am04,AM_ALU,DQ_ALU_A);/* Eingabe A fuer AM_ALU waehlen */
amSetGateSysClkMult(oAb00_Am04,AM_RCV,2);  /*  Option bei AM_RCV setzen */
dqAttachSurf(oSubstDstSurf,AM_RCV);

/* Definition ist abgeschlossen, nun wird die Pipe generiert            */
oSubstPipe[i] = dqCreatePipe(oSubstDstSurf, DQ_TRG_ONESHOT);
dqArmPipe(oSubstPipe[i], DQ_DSM_PIPE);          /* Timing berechnen    */
SubstWaitEvent[i]=emFindPipeEvent(oSubstPipe[i]);/* PAT-Event definieren*/
dqFirePipe(oSubstPipe[i]);                      /* Pipe starten        */
emWaitRefEvent(SubstWaitEvent[i],1);      /*  Auf Ende der Pat warten  */
dqFreePipeTimingBus(oSubstPipe[i]);       /*  Bus freigeben */
SubstPat[i]=emEndPatDef();                /*  PAT-Definition bendet    */
```

Diese Pipe kann nun mit `emRunPat(SubstPat[i])` gestartet werden. Abbildung 5.3 zeigt die Verbindungen, die durch das Beispielprogramm erzeugt werden.

5.4 Kontrollerkarte PMAC

Die Kontrollerkarte PMAC ist wie das MV200-System am VME-Bus angeschlossen. Sie enthält einen eigenen Prozessor, der die Kommunikation mit dem Hostrechner und die Steuerung der angeschlossenen Hardware übernimmt. An diese Karte ist ein PTV-Steuerbaustein angeschlossen, der wiederum den Kamerakopf mit Pan (Drehung), Tilt (Neigung) und zwei Vergenzachsen der beiden Kameras bedient. Weiterhin sind wie erwähnt pro Kamera Zoom und Fokus regelbar, die Irissteuerung der Kameras ist nur steuerbar. Ein Verstärker generiert die Steuersignale für die Motoren des Kopfes.

Der PMAC wird mit einer eigenen Kommandosprache vom Hostrechner aus gesteuert. Mit der Karte wurde auch die Steuersoftware `ptv` geliefert, die an die Anforderungen des Lehrstuhls angepaßt wurde. Insbesondere ist `ptv` um eine direkte Steuerung erweitert worden, die zeitoptimaler ist. Normalerweise erhält der PMAC vom Hostrechner Befehle im Ascii-Format, die der PMAC interpretiert und dann ausführt. Diese Interpretierung ist meist recht langsam. Weiterhin wartet ein Programm auf dem Hostrechner, bis der PMAC Vollzug dieses Befehls meldet. Wird zum Beispiel eine Fokussierung mit `movelens` von 0 (∞) nach 1 (1 m) durchgeführt, führt der PMAC diese

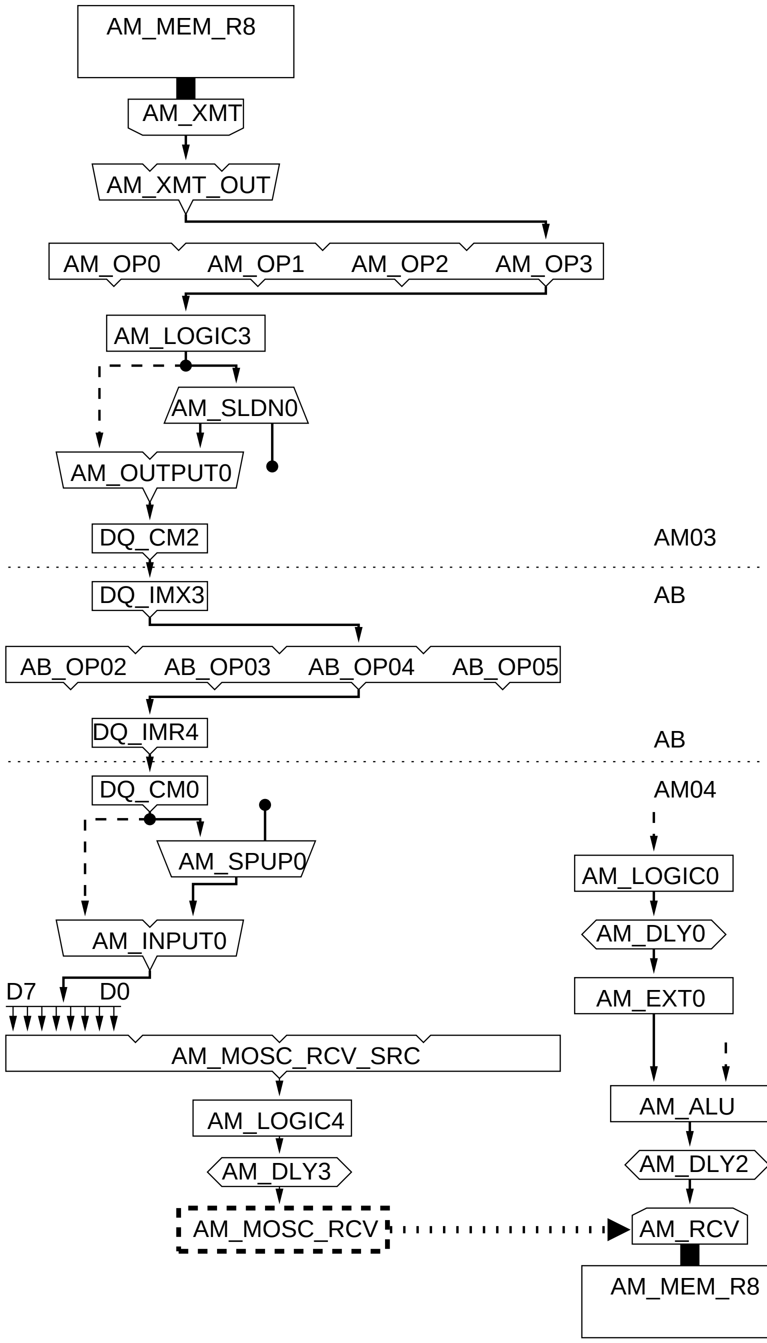

Abbildung 5.3: Verbindungen, die mit dem Beispielprogramm vom Speichermodul AM03 über den *crosspoint switch* des AB auf Speichermodul AM04 erzeugt werden.

Fokussierung aus und gibt erst nach Beendigung der Fokussierung, maximal nach 3 Sekunden, die Kontrolle an das Programm auf dem Hostrechner zurück. Das ganze System muß auf den Ablauf der Fokussierung warten.

Die wesentlichen Befehle für ptv sind:

- up initialisiert den PMAC und fährt den Kopf auf die Ausgangsposition.
- move <pan> <tilt> <vergenz$_{links}$> <vergenz$_{rechts}$> fährt den Kopf auf die angegebene Position. Als Parameter wird der Winkel der Achsen angegeben. Dieses Kommando fährt die angegebene Position an und übergibt erst danach dem ptv wieder die Kontrolle.
- where fragt die aktuelle Position der move-Achsen ab.
- movelens <fokus$_{links}$> <zoom$_{links}$> <fokus$_{rechts}$> <zoom$_{rechts}$> fährt die angegebene Linsenposition an. Fokus und Zoom können von 0 bis 1 gehen, Fokus 0 entspricht einer Fokussierung in eine Entfernung ∞, Fokus 1 entspricht 1 m. Zoom 0 entspricht der maximalen Vergrößerung ($\times 6$) und Zoom 1 entspricht keiner Vergrößerung ($\times 1$). Wie beim move-Befehl wird auch hier zunächst die Position angefahren und erst danach wieder dem ptv die Kontrolle übergeben.
- wherelens liefert die aktuellen movelens-Werte.
- iris L|R 1|-1 <Millisekunden> öffnet (1) oder schließt (-1) die Iris der entsprechenden Kamera (L: linke Kamera, R: rechte Kamera) für die entsprechend Zeit (in Millisekunden).
- dstart 4 startet den Direktmodus und aktiviert die folgenden Befehle:
- dmove <pan> <tilt> <vergenz$_{links}$> <vergenz$_{rechts}$> funktioniert wie move, nur wird die Kommunikation nicht mehr über herkömmliche PMAC-Kommandos ausgeführt, sondern über Register in denen die Achsenregler immer die Sollposition finden. Damit wird erreicht, daß das Programm sofort nach Absetzen dieses Befehls weiterläuft. Des weiteren muß der PMAC nicht mehr die Kommandos interpretieren, sondern kann sie gleich ausführen.
- dmovelens <fokus$_{links}$> <zoom$_{links}$> <fokus$_{rechts}$> <zoom$_{rechts}$> fährt wie movelens die Linsenposition an, die Achsenpositionen werden aber wie bei dmove direkt übergeben.

Speziell für die Fokussierung nutzen wir diesen Direktmodus. Wir geben dem PMAC eine zu erreichende Fokusposition vor, und während der PMAC diese Position einstellt, rechnen wir auf dem Hostrechner und auf dem MV200-System das gerade aufgenommene Bild durch. Mit dem PMAC-Befehl wherelens können wir dann die erreichte Position der Fokuslinse auslesen und für die Normierung auf die Referenzgröße und die eigentliche Entfernungsschätzung nutzen.

Die obigen `ptv`-Befehle sind alle als entsprechende C-Funktionen implementiert.

5.5 Zusammenspiel von Kamera und PMAC

Bei der Entwicklung dieser Arbeit fielen einige Probleme in der Kamerasteuerung auf. Da die meisten Probleme nicht direkt mit dieser Arbeit zu tun haben, sind diese in Anhang A dokumentiert. Ein Problem ist die verzögerte Ausführung von `dmovelens`-Befehlen. Abbildung 5.4 zeigt das gemessene Zeitverhalten. Bei einer Neupositionierung von 0,015 Counts dauert der eigentliche Positioniervorgang nur 0,1 Sekunden. Aber erst 0,2 Sekunden nach dem der Positionierungsbefehl an den PMAC abgesetzt wurde, bewegen sich die Linsen. Die Werte wurden für jeden Meßwert mit jeweils einzelnen Positionierbefehlen angefahren, da sonst die Ausführung des `wherelens` das gemessene Ergebnis weiter verfälscht hätte. Dadurch erklärt sich der nicht ganz glatte Verlauf. Die Fokussierung ist im wesentlichen nach 0,35 Sekunden abgeschlossen.

Viele `wherelens`-Kommandos verlangsamen weiterhin die Ausführung der Neupositionierung. In ersten Messungen für Abbildung 5.4 wurde die Fokusposition so oft wie möglich gemessen. In einer weiteren Messung wurde dann jeweils eine Messung nach angemessener Pause durchgeführt. In der ersten Variante kommen wir auf 0,3 Sekunden Verzögerung und 0,2 Sekunden für die Fokussierung, insgesamt 0,5 Sekunden.

Insgesamt kann frühestens 0,35 Sekunden nach einem Positionierbefehl eine neue Bildaufnahme stattfinden, da sonst die gewünschte Position noch gar nicht erreicht ist. Da wir außerdem die Vergrößerung der Fokussierung im Verfahren über die aktuelle Fokusposition feststellen, können wir während der Positionierung keine Bilder aufnehmen.

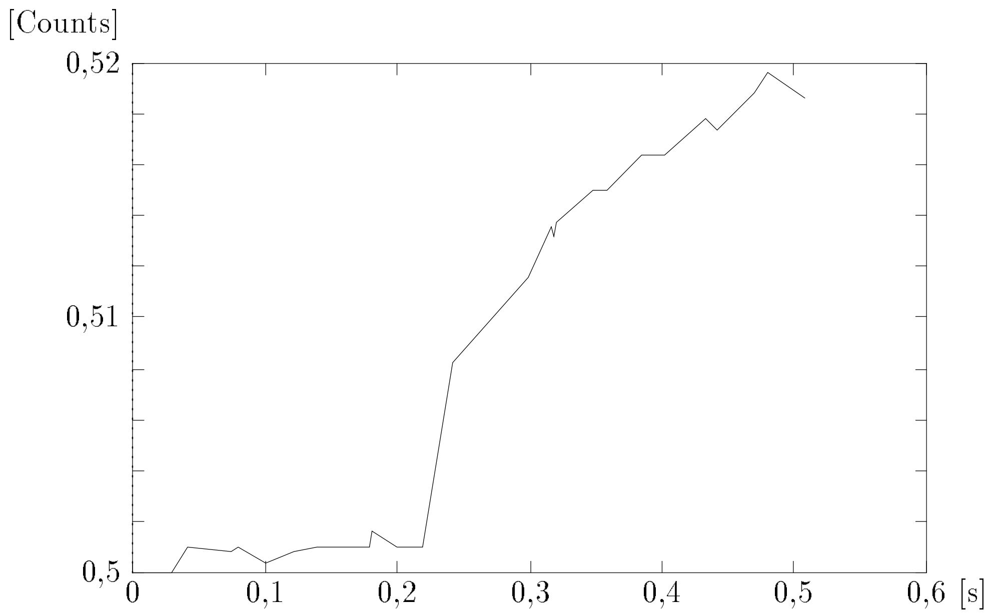

Abbildung 5.4: Zeitverhalten der Kamera bei der Fokussierung von 0,5 auf 0,515. Erst nach einer Verzögerung von 0,2 Sekunden bewegen sich die Linsen, nach 0,35 Sekunden ist die Fokussierung im wesentlichen abgeschlossen.

Kapitel 6

Algorithmus der Entfernungsmessung

Im folgenden soll der Algorithmus beschrieben werden, der der Entfernungsmessung zugrunde liegt. Es werden Tiefenkarten berechnet, die zu einer Szene die „scharfen“, d.h. optimalen Fokuswerte angeben. Die absolute Entfernung wird erst durch eine Kalibrierung der Fokuswerte auf die Entfernungen berechnet.

Zur Berechnung des Schärfebildes einer Kameraszene implementieren wir den Tenengrad auf dem MV200-System. Die Filterantworten werden mit dem Modul 'MiniWarper' (WX) auf die Referenzgröße normiert. Damit kann aus diesen Bildern durch Maximumbildung für jedes Pixel eine schärfste Fokusposition bestimmt werden. Diese Fokuspositionen speichern wir dann als Tiefenwert. Abbildung 6.1 zeigt die einzelnen Programmteile im zeitlichen Ablauf. Der Algorithmus ist wie folgt implementiert:

- Die Sobeloperatoren in X- und Y-Richtung (PATs b,c) realisieren wir auf dem Faltungsmodul als zwei sequentielle PATs. Wir berechnen $f_x := I * S_x$ und $f_y := I * S_Y$.
- Die Normierung erfolgt mit PAT d) auf dem AU-Device. Es ergibt sich $f_n := f_x^2 + f_y^2$.
- Die Summierung (PATs e, f) realisieren wir durch eine Faltung mit einer horizontalen und vertikalen Box-Maske der Größe $N \times 1$ bzw. $1 \times N$. $ten := \sum_{\square_{N\times N}} f_n = 1_{N\times N} * f_n = 1_{N\times 1} * 1_{1\times N} * f_n$
- Danach wird die Filterantwort auf eine Normgröße zurückgerechnet (PATs g_1, g_2). Wir nutzen die Kalibrierungsdaten aus Abschnitt 4.3

und skalieren mit dem MiniWarper-Modul auf die kalibrierte Bildgröße. Sinnvoll ist es hier, auf die kleinste Bildröße zu skalieren, damit der ungenutzte Rand der Filterantwort nicht zu groß wird. Durch die Faltungsoperationen des Tenengrades erhalten wir einen Rand von $\frac{N}{2}+1$ Pixel. Dieser sollte nicht zusätzlich vergrößert werden. Als Interpolationsverfahren der Skalierung wird das *nearest-neighbour*-Verfahren benutzt. Da wir nur mit kleinen Vergrößerungsfaktoren arbeiten, würden z.B. bei bilinearer Interpolation die Werte der Filterantworten verfälscht werden. Dies ist für die Maximumssuche ungeeignet.

Es ergibt sich als Warpergebnis $f_w(x,y) := ten(mx - m \cdot x^* + x^*, my - m \cdot y^* + y^*)$.

Vor dem eigentlichen Programmablauf wird für jede gewünschte Fokusposition eine PAT initialisiert. Damit können bei den Messungen die entsprechenden PATs zu den jeweiligen Fokuspositionen abgerufen werden.

- In PAT h) wird festgestellt, an welchen Pixelpositionen die Filterantworten größer sind als bei bisherigen Fokuspositionen. An diesen Stellen wird in der Entfernungskarte die entsprechende Fokusposition eintragen. $f_s(x,y) := \begin{cases} 1 & : \quad f_w(x,y) \geq f_m(x,y) \\ 0 & : \quad \text{sonst} \end{cases}$
- Die Fokuspositionen wird über vorher für jede gewünschte Fokusposition b vorkonfigurierte PATs i) in die Tiefenkarte f_t eingetragen.

  $$f_t(x,y) := \begin{cases} b & : \quad f_s(x,y) = 1 \\ f_t(x,y) & : \quad \text{sonst} \end{cases}$$
- Danach wird das Maximum f_m der bisherigen und der aktuellen Filterantwort berechnet (PAT j): $f_m(x,y) := \max(f_m(x,y), f_w(x,y))$

Diesen Algorithmus lassen wir für jede angefahrene Fokusposition ablaufen. Eine Iteration benötigt in diesem Programm ca. 0,3 Sekunden. Da die Positionierung des Fokus für einen kompletten Durchlauf 3 Sekunden benötigt und auch bei kleinen Schritten ungefähr 0,3 Sekunden braucht (siehe Abschnitt 5.5), können wir die Fokussierung sofort nach Aufnahme eines Bildes starten. Bis zur nächsten Bildaufnahme rechnet das MV200-System die Filter durch, und der Kontroller PMAC positioniert währenddessen den Fokus auf den neuen Wert.

Wir erhalten damit die Tiefenkarte $f_t(x,y)$ und das Maximum der Filterantworten über alle Meßbilder $f_m(x,y)$. Diese Messung führen wir in beiden Fokusrichtungen ($0 \rightarrow 1$ und $1 \rightarrow 0$) durch. Dadurch erhalten wir unterschiedliche Maxima der Filterantworten, da durch die 8-Bit-Arithmetik des

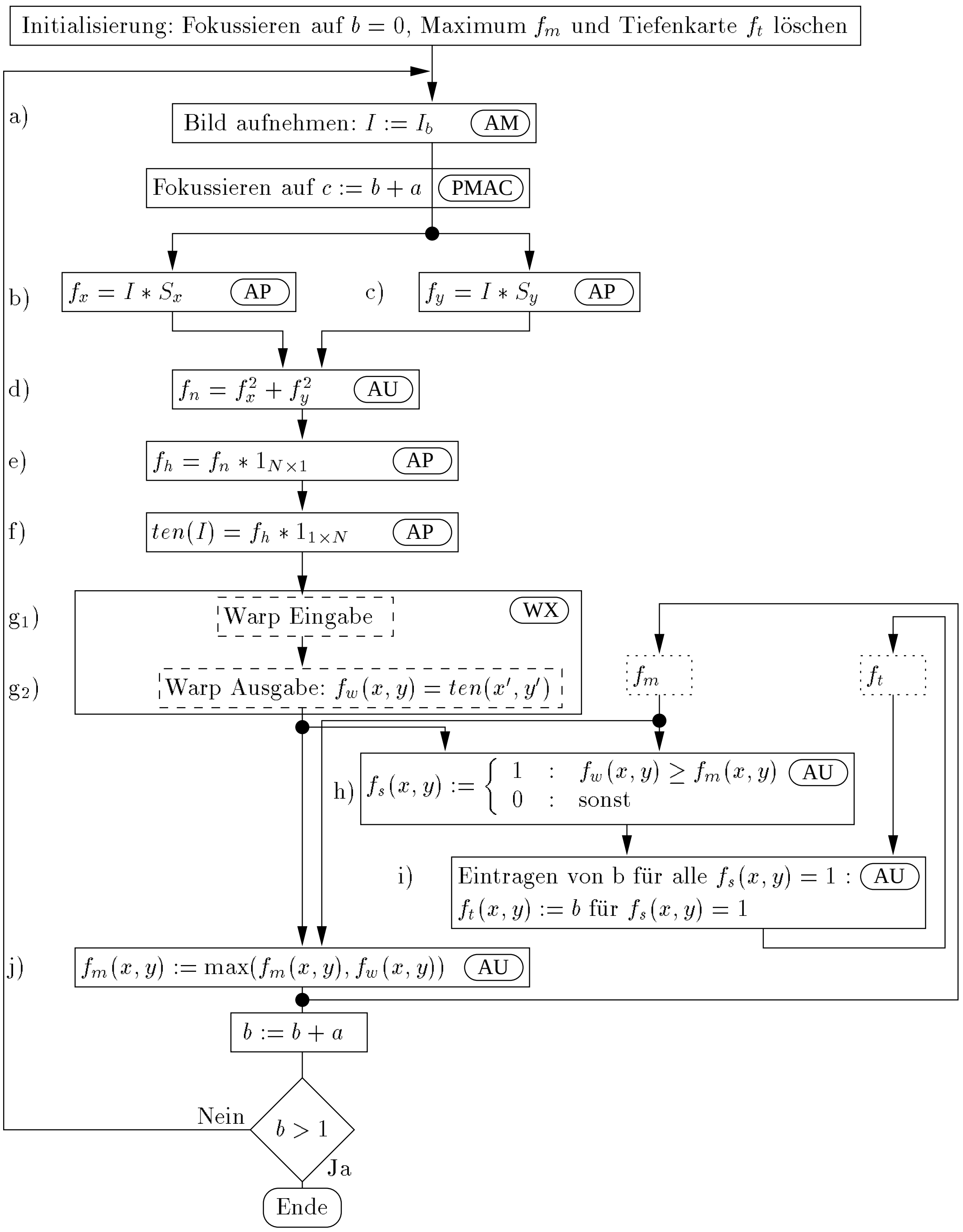

Abbildung 6.1: Ein Durchlauf des Programmes. Die Buchstaben a) bis j) bezeichnen die PATs, in den PATs ist das verwendete Rechenmodul angegeben. Die AM-Module als Zwischenspeicher sind hier nicht angegeben. Die Fokussierung erfolgt direkt nach der Bildaufnahme. Oben wird der Tenengrad berechnet (b-f), dann die Rückvergrößerung auf Normgröße durchgeführt (g_1,g_2), und unten neue Maxima gesucht und in die Tiefenkarte eingetragen (h-j).

MV200-Systems nicht die volle Bandbreite der Filterantwort berechnet wird. Die Maxima werden zu Plateaus, und die beiden Richtungen der Fokussierung berechnen daher die beiden Seiten dieses Plateaus. Diese nennen wir das rechte und linke Maximum der Filterantwort. Dazu erhalten wir entsprechende Fokuswerte. Durch die Breite des Plateaus in der Filterantwort können wir eine Schätzung für die Genauigkeit der Tiefenmessung berechnen. Wir können zu einer Grauwertkante also einen minimalen und maximalen geschätzten Abstand angeben, damit existiert ein Unsicherheitsmaß für die Entfernungsschätzung. Desweiteren haben wir das Maximum der Schärfefunktion und können bei kleinen Filtermaxima von einer größeren Unsicherheit der gemessenen Werte ausgehen. Kleinere Filtermaxima implizieren einen schwachen Grauwertübergang, dieser wird also mehr vom Bildrauschen beeinflußt.

Im weiteren wird die Implementierung auf dem MV200-System genauer beschrieben.

6.1 Bildaufnahme und Ausgabe

Die Bildaufnahme wird komplett vom MV200-System vorgenommen, und erfolgt in Echtzeit (25 Bilder/s). Die Ausgabe auf dem am MV200-System angeschlossenen Monitor kann zur Kontrolle der berechneten Bilder genutzt werden.

Nach der Initialisierung der MV200 definieren und starten wir eine sogenannte *continuous pipe* vom AS-Device (AD-Wandler) zu einem AM-Device (Speichermodul), also eine Pipe die unabhängig von weiteren Pipes sofort nach ihrem Ende neu gestartet wird. Insbesondere für die Ansteuerung der AS- und AG-Devices, d.h. der AD- und DA-Wandler, brauchen wir diese Pipes: Auf dem AS kommt das analoge Kamerasignal an, welches synchron digitalisiert werden muß. Eine asynchrone Digitalisierung des aufgenommenen Bildes ist nicht möglich. Ebenso muß das AG-Device kontinuierlich mit Bildern versorgt werden, da im AG-Device das VGA-Monitorsignal erzeugt wird. Solange die Pipe auf dem AG-Device nicht aktiv ist, kann der angeschlossene Monitor kein Bild anzeigen. Bei nur gelegentlichem Start kann der Monitor das ausgegebene Bild jedoch nicht synchronisieren.

Zur Bildaufnahme wird das Bild in PAT a) aus dem AM-Device in ein weiteres AM-Device kopiert. Dieses verwenden wir als Quelle für die Sobel-Operationen.

6.2 Sobel-Operationen

Zur Berechnung des Tenengrad (siehe Abschnitt 3.2.2) müssen als erstes die Ableitungen in x- und y-Richtung berechnet werden, also eine Faltung mit den entsprechenden Sobel-Masken.

Die beiden Sobel-Operationen führen wir mit zwei PATs auf dem AP-Device (Faltungseinheit) durch. Dieses besitzt ein Faltungsmodul (das NMAC), mit dem eine 8×8-Faltung konfiguriert werden kann. Es könnten also auch größere Sobel-Masken als die 3×3-Masken verwendet werden. Die Masken für den Sobel werden mit dem Kommando `dqWtRect` auf den NMAC geladen. Beide Sobel-Pipes (b, c) sind bis auf die Faltungsmaske und den Ergebnisspeicher identisch.

6.3 Berechnung der Norm

Die Quadrierung und Summierung des Tenengrades haben wir im Programm mit 'Norm' bezeichnet. Insgesamt berechnet diese Pipe $S(I)^2 = f_n = f_x^2 + f_y^2$, da sich die Wurzel aus $S(I)$ und die Quadrierung aus der Summierung aufheben.

Implementiert wurde die gesamte Norm mit PAT d) auf dem AU-Device (Arithmetische Einheit). Der lineare Abschnitt des AU-Device bietet 4 Multiplizierer, deren Ergebnisse in weiteren Schritten baumförmig addiert werden. Wir nutzen von diesen 4 Multiplizierern nur zwei und konfigurieren die anderen als Multiplikation mit 0. Die Multiplizierer bekommen jeweils an beide Eingänge ein Sobel-Ergebnis geschaltet, so daß diese beiden Multiplizierer jeweils das Quadrat des Sobels in x- und y-Richtung berechnen. Die nachfolgenden Addierer berechnen dann den ersten Teil des Tenengrades, den Absolutbetrag oder Norm.

6.4 Zeilen- und spaltenweise Aufsummierung

Die Summierungen können auf dem MV200-System nur durch Faltungen mit entsprechenden Box-Masken erreicht werden. Für eine 4×4-Summierung wird also mit einer 4×4-Maske gefaltet, die nur Einsen enthält.

Wir führen zwei Faltungen mit 1×4- und 4×1-Masken separabel hintereinander aus. Wir benötigen dadurch zwar im Programm eine Pipe mehr, aber da wir den NMAC als 64×1- oder als 1×64-Faltungsmodul nutzen können, sind wir nicht auf kleine Summierungsmasken festgelegt. Wir können deshalb zum Test ohne Änderungen am Programmcode Summierungen bis zur Fenstergröße 64×64 durchführen.

Die beiden Summierungen (PAT e, f) sind bis auf die Filterorientierung (horizontal bzw. vertikal) und die jeweiligen Ein- und Ausgabespeicher gleich. Insgesamt erhalten wir durch diese beiden Summierungen den Tenengrad des aufgenommenen Bildes.

6.5 Korrektur auf Normgröße

In Kapitel 4 wurden schon die auftretenden Bildpunktverzerrungen bei einer Änderung der Fokusposition besprochen. Diese können auf dem MV200-System ausgeglichen werden, so daß die Filterantworten des Tenengrad bei verschiedenen Fokuspositionen vergleichbar werden.

Der MiniWarper (WX) kann biquadratisch Bildverzerrungen berechnen. Er berechnet $w(x, y) = f(a + bx + cx^2 + dy + ey^2 + fxy, g + hx + ix^2 + jy + ky^2 + lxy)$ für eine Eingabe $f(x, y)$. Das sich ergebende Bild kann weiterhin geglättet werden. Wir nutzen nur ein sogenanntes bilineares Warping gemäß $w(x, y) = f(a + bx, c + dy)$. Das Ausgabebild soll nicht geglättet werden, so daß als Interpolationsverfahrender nächste Nachbar (*nearest neighbour*) gewählt wird.

Der MiniWarper besitzt einen eigenen Speicherbereich, den wir im Gegensatz zu den Speichermodulen (AM) nicht gleichzeitig lesen und schreiben können. Daher müssen wir das Laden der Filterantwort in den WX und das eigentliche Verzerren in zwei PATs realisieren.

Das Laden der Filterantwort erfolgt mit PAT g_1. Diese kopiert die Filterantwort des Tenengrad von einem AM-Device in den Speicher des WX-Device. Die eigentliche Verzerrung wird über eine Anzahl von vordefinierten PATs ($g_2(b)$) durchgeführt, denen wir zu jeder der anzufahrenden Fokuspositionen die entsprechenden Vergrößerungsfaktoren vorgeben. Anschließend wird die gewünschte Funktion (bilinear) gewählt.

Aus der Kalibrierung in Kapitel 4 haben wir eine Reihe von Vergrößerungsfaktoren zu Fokuspositionen erhalten. Zur gewünschten Fokusposition b werden aus der Liste der Vergrößerungsfaktoren die 4 umliegenden Meßwerte $(b_1, m_1) \ldots (b_4, m_4)$ gewählt, so daß $b_1 \leq b_2 < b \leq b_3 \leq b_4$. Falls nur ein Meßwert kleiner als b vorhanden ist, wird $b_1 = b_2$ gewählt, ebenso wird bei nur einem Meßwert größer als b $b_3 = b_4$ gewählt. Es wird dann

$m = m_1 + \frac{b-b_1}{(b_3-b_1)(m_3-m_1)} + m_2 \frac{b-b_2}{(b_4-b_2)(m_4-m_2)}$ berechnet. Diese Interpolation kann das Oszillieren der Vergrößerungsfaktoren (siehe Abbildung 4.6) glätten.

Damit erhalten wir einen Vergrößerungsfaktor m für die anzufahrende Position. Als Vergrößerungsmittelpunkt wählen wir die Mitte des Bildes, also $(x^*; y^*) = (378; 288)$. Da das Warpmodul als Aufpunkt (0;0) wählt, muß vor und nach der eigentlichen Skalierung eine Translation auf den Nullpunkt und auf den Mittelpunkt des Bildes durchgeführt werden. Wir erhalten als Funktion für die neuen Pixelpositionen der Filterantwort für den Skalierungsfaktor m:

$$x' = mx - m \cdot x^* + x^* \tag{6.1}$$
$$y' = my - m \cdot y^* + y^* \tag{6.2}$$

Diese Transformation kann nicht am Anfang der Verarbeitung, also vor dem Tenengrad, auf dem Eingabebild ausgeführt werden, da wir sonst Störungen durch das Korrigieren der Bildgröße erhalten. Eine bilineare Interpolation würde das Bild verfälschen, so daß der Tenengrad ein schwach tiefpaßgefiltertes Eingangsbild erhält. Dies führt dann zu einer Reduzierung des hochfrequenten Anteils im Signal und damit zu schlechteren Ergebnissen des Tenengrads.

Insgesamt erhalten wir eine Anzahl von PATs $\mathrm{g}_2(b)$, die um Vergrößerungsfaktoren $m \in [1; 1,1]$ das Bild um den Mittelpunkt $(x^*; y^*)$ vergrößern. Die genauen Werte erhalten wir aus der in Kapitel 4 beschriebenen Kalibrierung. Bei Fokusposition b wird entsprechend $\mathrm{g}_2(b)$ aufgerufen.

Wir erhalten mit diesem Teil des Programms ein auf eine gleiche Größe normiertes Ergebnisbild. Damit können wir nun die Filterantworten $ten(x, y)$ an einzelnen Pixeln (x, y) unterschiedlicher Fokusstellungen vergleichen.

6.6 Berechnung der Tiefenkarte

Die Tiefenkarte berechnen wir mit Hilfe des bisherigen Maximums der Filterantworten. Wir führen in einer ersten PAT (h) den Vergleich des bisherigen Maximums f_m mit der gewonnenen Filterantwort f_w aus. Diese PAT liefert überall da einen Grauwert von 127, wo die Filterantwort größer als dieses Maximum ist, das heißt $ten(x', y') = f_w(x, y) \geq f_m(x, y)$, ansonsten wird 0 ausgegeben. In einem zweiten Schritt (PAT i) können wir dann an diesen Stellen im Bild die aktuelle Fokusposition eintragen. Ähnlich wie bei der Korrektur der Bildgröße führen wir den zweiten Schritt in einer Reihe von PATs i(b) durch, von denen die zur Fokusposition passende ausgewählt wird.

Wir könnten prinzipiell diese beiden Schritte in einer PAT ausführen. Das würde zu einer etwas längeren PAT als der obigen im ersten Schritt führen. Ein Nachteil wäre folgender: Da wir diese PAT für jede Fokusposition duplizieren müssen, geht der vorhandene Speicher (die sogenannte `PatMemSize`) für die Zwischenspeicherung der PATs auf dem Hostrechner schnell aus. Dieser Speicherbereich ist technisch bedingt auf ca. 700 kB begrenzt. Da in unserer Implementierung die zweite PAT sehr kurz ist, kommen wir mit wesentlich weniger Speicher für die vielen PATs aus. Die `PatMemSize` bleibt damit in einem möglichen Rahmen, und das Programm bleibt auch für viele PATs funktionsfähig.

Der erste Schritt ist auf dem AU-Device in der *Non-Linear Processor Section* realisiert. Dieser Teil des AU-Device stellt viele nichtlineare Funktionen zur Verfügung. Wir nutzen ein Maximum und eine Subtraktion.

Wir berechnen in einer arithmetische Einheit (ALU) des AU-Device das neue Maximum aus Filterantwort f_w und dem bisherigen Maximum f_m. Anschließend ziehen wir nun mit einer weiteren ALU das bisherige Maximum davon ab, also $f_m(x,y) - \max(f_w(x,y), f_m(x,y))$. Das Ergebnis ist ein Bild, in dem sich für alle Positionen (x, y) mit einer Filterantwort kleiner gleich dem bisherigen Maximum eine 0 steht und sich für alle Positionen deren Filterantwort größer ist, ein Wert kleiner 0 ergibt.

Mit einem Komparator können wir aus diesem 8-Bit-Bild nun ein logisches Binärbild berechnen: Wir erhalten eine 1, wenn die Eingabe kleiner als 0 ist, die Filterantwort also größer war als das bisherige Maximum, und 0, wenn die Eingabe gleich 0 ist.

Im Ausgabeteil des AU-Device können wir über eine Konstante an einem sogenannten *Multiple Operation State Controller* (MOSC) diese Werte in 127 und 0 ändern. Ein MOSC bekommt im allgemeinen eine 1-Bit-Eingabe und springt abhängig von der Eingabe zwischen zwei Zuständen hin und her. Eine Eingabe 0 schaltet den MOSC auf Zustand 0, Eingabe 1 auf Zustand 1. Wir programmieren diesen Konstantengenerator so, daß eine 0 bei Zustand 0 ausgegeben wird, und der Wert 127 bei Zustand 1. Damit haben wir ein Binärbild generiert und in ein 8-Bit-Grauwertbild konvertiert, das anzeigt, an welchen Stellen in der Tiefenkarte wir die Tiefeninformation ändern müssen.

Das Maximum, das wir schon auf dem AU berechnet und für die Maskierung genutzt haben, können wir nicht wieder vom AU ausgeben, da ein entsprechender Datenpfad nicht vorhanden ist. Dies müssen wir in einem weiteren Schritt (PAT j) durchführen.

Die Ersetzung der Tiefeninformationen führen wir zwischen zwei AM-Devices auf dem AB-Device aus. Dieses Programm wurde auch schon in Abbildung 5.3 gezeigt. Im Zielspeicher nutzen wir ein Bit des Binärbildes für einen MOSC im AM-Device. Dieser bewirkt eine Schreibkontrolle auf den Speicher des AM. Ist der Zustand 1, so wird der Speicher mit den angelieferten Daten überschrieben. Bei Zustand 0 wird der Speicher nicht überschrieben. Nun können wir mit einem logischen Gatter dem Speicher die aktuelle Fokusposition zuführen. Diese Fokusposition wird nun an den Stellen in der Tiefenkarte eingetragen, an denen das Binärbild eins ist.

Insgesamt erhalten wir eine neue Tiefenkarte f_t, die die aktuelle Fokusposition b eingetragen hat. Im folgenden wird die Berechnung des aktuellen Maximums beschrieben.

6.7 Aktualisierung des Filtermaximums

Wie schon bei der Berechnung der Tiefenkarte nutzen wir bei PAT j) die *Non-Linear Processor Section*. Eine ALU berechnet das Maximum der Filterantwort und des bisherigen Maximums, alle weiteren ALUs geben dieses neue Maximum unverändert weiter.

6.8 Ende der Hauptschleife

Insgesamt haben wir mit diesen Schritten einen Durchlauf berechnet. Es ergibt sich ein neues Bild mit dem Maximum der bisherigen Filterantworten und ein dazugehöriges Entfernungsbild mit den Fokuspositionen, an denen diese Maxima auftraten. In einer Schleife werden nun Fokuspositionen von 0 bis 1 angefahren. Nach Ablauf erhalten wir das Maximums- und Entfernungsbild für alle angefahrenen Fokuspositionen als Ergebnis.

Wir führen in der aktuellen Fassung des Programms diesen Algorithmus für insgesamt 64 Fokuspositionen durch. Wir nutzen also für die Vergrößerung auf die Normgröße und für das Ersetzen der Tiefenwerte jeweils 64 PATs. Nach Beendigung der Hauptschleife speichern wir die erhaltene Tiefenkarte und das Maximum der Filterantworten auf dem Hostrechner. Nun können wir diese Schleife auch noch für die Rückrichtung des Fokus durchführen $[b = 1 \to 0]$. Wir benötigen dazu eine andere Kalibrierung der Vergrößerungen und damit eine neue Serie von PATs für die Normierung auf die Standardgröße, ansonsten können die gleichen PATs wie aus der ersten Richtung der Fokusbewegung wiederverwendet werden. Wir erhalten als Ergebnis eine weitere Tiefenkarte und ein Maximum der Filterantwort. Auch dieses speichern wir auf dem Hostrechner ab.

Eine Iteration benötigt ca. 0,3 Sekunden, eine Richtung der Fokussierung dauert damit 19 Sekunden. Wie schon in Abschnitt 5.5 gezeigt, kann ein Schritt der Fokussierung von der Kamera nicht schneller durchgeführt werden. Durch Zusammenfassen von PATs und weitere Optimierungen kann das Programm weiter beschleunigt werden, dies ist aber aufgrund der Verzögerungen der Fokussierung nicht sinnvoll.

Im anschließenden Kapitel sollen Ergebnisse und Erfahrungen mit dieser Implementierung gezeigt werden.

Kapitel 7

Experimente der Tiefenmessung

In den folgenden Experimenten wollen wir das vorgestellte Verfahren hinsichtlich Güte und Genauigkeit testen.

In Abbildung 7.1 (links) wurde von schräg oben eine Tischplatte aufgenommen. Die Zoomstellung ist 0,5. Ein gemusterter Stoff als Auflage der Tischfläche gibt viele Grauwertkanten im Bild. In die Mitte des Bildes liegt ein Zollstock. Rechts unten im Bild befindet sich ein dunkler, sehr naher Bereich (ca. 1 m Abstand zur Kamera), der untere Bildrand ist ca. 1,5 m, der obere Bildrand ca. 3,2 m von der Kamera entfernt.

Abbildung 7.1 zeigt die Schärfefilterantwort des Tenengrad (rechts) für Pixelposition (394;296) (links im Bild markiert) als Funktion der Fokusposition. Die Filterantwort wurde unter Khoros berechnet. Als durchgezogene Linie ist die genaue Berechnung der Filterantwort gezeigt, gestrichelt die auf 8 Bit quantisierte Filterantwort. Die Filterantworten wurden auf gleiche Größe skaliert. Unter b =0,23 Counts berechnet sich die quantisierte Filterantwort zu 0. Das globale Maximum der genau berechneten Filterantwort liegt bei 0,59 Counts, die quantisierte Filterantwort zeigt ein Plateau von 0,53 bis 0,61 Counts als Maximum. Als rechtes Maximum bei der Fokussierung von 0 nach 1 sollten wir bei 0,61 Counts ein Maximum erhalten, bei der Fokussierung von 1 nach 0 bei 0,53 Counts. Tatsächlich erhalten wir auf dem MV200-System bei 0,56 Counts das rechte und bei 0,53 Counts das linke Maximum der Filterantwort.

Abbildung 7.2 zeigt das auf dem MV200-System berechnete Ergebnis des Verfahrens. Im berechneten Maximum der Filterantwort sehen wir, daß der Filter nur an Grauwertkanten im Bild reagiert hat. Weiterhin antwortet der

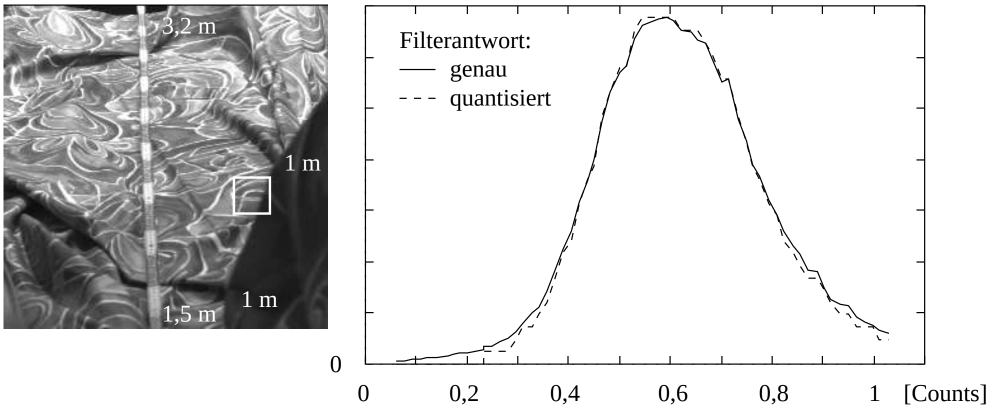

Abbildung 7.1: Filterantwort des Schärfefilters über die Fokusstellung (rechts) an der Pixelposition (394;296) (links im Bild markiert). Die durchgezogene Linie zeigt die genaue Berechnung der Filterantwort, die gestrichelte Linie zeigt die auf 8 Bit quantisierte Filterantwort. Die Kurven wurden auf gleiche Größe skaliert. In der quantisierten Filterantwort zeigt sich ein Plateau als Maximum, während in der genau berechneten Filterantwort ein eindeutiges Maximum vorliegt.

Filter auch abhängig vom Grauwertunterschied der Kanten unterschiedlich stark. Unten links in den Falten (Bereich 1) reagiert der Filter zum Beispiel teilweise gar nicht, obwohl dort Grauwertkanten erkennbar sind. Der Tenengrad liefert bei genauer Berechnung der Filterantworten auch hier eine niedrige Antwort. Durch die verwendete Hardware wird die Filterantwort auf 8 Bit quantisiert. Da die Antwort des Tenengrad wesentlich mehr Bits (bis zu ca. 24 Bit unter Khoros) beansprucht, muß von der Filterantwort ein Teil der Information abgeschnitten werden. Dies lösen wir auf dem MV200-System durch Shiften, also durch sukzessive Division durch 2. Niedrige Filterantworten ergeben auf dem MV200-System also 0. Hohe Filterantworten werden entsprechend quantisiert.

Die Maxima der Filterantworten sind in beiden Richtungen gleich, da das Verfahren den gleichen Algorithmus nutzt. Die dort gefundenen Tiefenwerte sind aber insgesamt niedriger, geben also eine größere Entfernung an.

Die Tiefenkarte hat an den Pixelpositionen, an denen der Tenengrad eine signifikante Antwort liefert, einen entsprechenden Entfernungswert eingetragen. Ein schwarzes Pixel bedeutet hier, daß der Tenengrad nicht (beziehungsweise nicht signifikant genug) geantwortet hat und wir daher keinen stabilen Entfernungswert für dieses Pixel feststellen konnten. Wir erkennen in der berechneten Tiefenkarte am oberen Bildrand einen mittleren Grauwert an

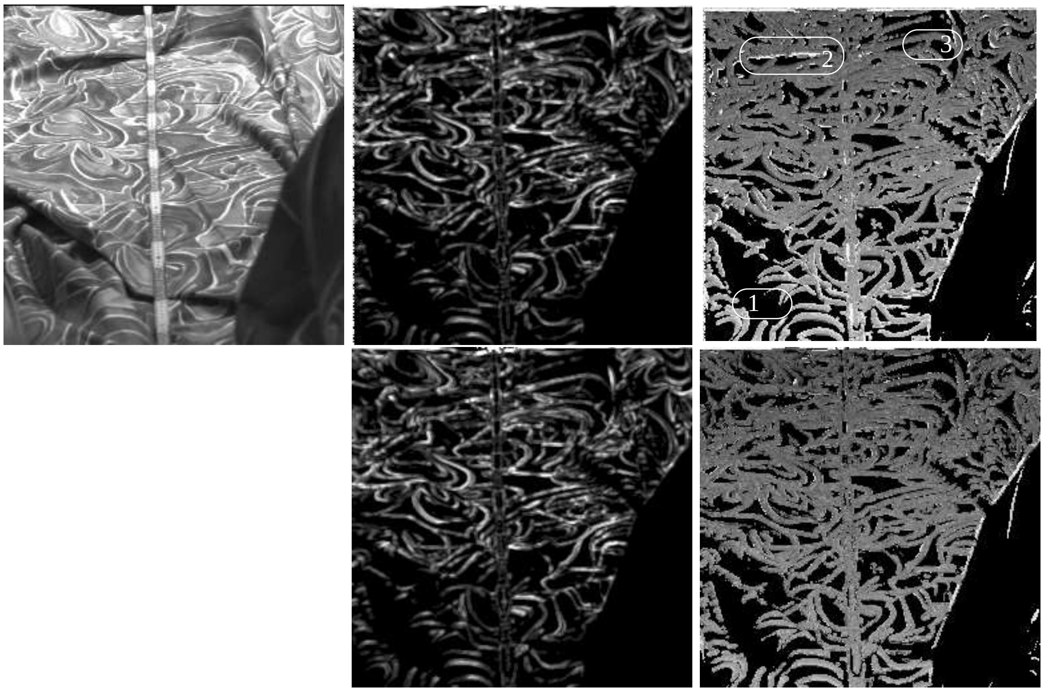

Abbildung 7.2: Ergebnis des Verfahrens. Links die Szene, mittig das Maximum des Schärfefilters, rechts die gemessenen Entfernungen. In den beiden Zeilen sind die verschiedenen Richtungen der Fokussierung der Tiefenmessung gezeigt. Nur für Filterantworten größer Null sind in der Tiefenkarte die Fokuspositionen eingetragen. Bei einer Filterantwort von Null bleibt die Tiefenkarte schwarz. Der untere Rand der Aufnahme ist ca, 1,5 m von der Kamera entfernt, und ergibt in der Tiefenkarte hellen Grauwerte. Der obere Rand der Aufnahme ist 3,2 m von der Kamera entfernt, und gibt in der Tiefenkarte dunklere Grauwerte.

den Stellen, an denen der Filter geantwortet hat. Nach unten hin werden die Grauwerte immer heller. Die rechte untere Ecke zeichnet sich mit sehr hellen Grauwerten ab, das heißt die Entfernung zur Kamera ist gering.

Die Filterantworten des Tenengrad zeichnen sich sehr breit in der Tiefenkarte ab. Die Breite der Antwort des Schärfefilters auf einer Grauwertkante hängt von der Größe des Einzugsbereiches ($\mathcal{U} = N \times N$) ab, in der die Antworten des Gradienten aufsummiert werden. Wir haben eine 4×4-Summierung gewählt, so erhalten wir eine 8 Pixel breite Filterantwort des Tenengrad. Ist der Einzugsbereich zu groß, läßt die Ortsauflösung der Messung nach.

In der oberen linken Ecke der Tiefenkarte (Bereich 2) zeichnet sich ein weißer waagerechter Streifen ab. Dieser entspricht einer sehr schwachen Filterantwort des Tenengrad. Diesen Fehler können wir durch Einbeziehen des Filtermaximums erkennen. Unter Khoros können wir das lokale Maximum auf dem Bild der Filterantwort berechnen, das Maximumsbild also skelettieren, und nur an diesen Pixelpositionen den Entfernungswert berücksichtigen. Der Grund für diese Störung ist das nicht ganz ideale Verhalten der Kamera beim Erreichen extremer Fokusstellungen. Wie wir schon bei der Kalibrierung der Fokussierung gesehen haben, ist die Vergrößerungskurve (Abbildung 4.6) dort keine lineare Funktion mehr. Sie wird fast oszillierend. Diese Störungen führen zu leicht unterschiedlichen Vergrößerungsfaktoren. Da die Verschiebung am Rand des Bildes am größten ist, wirkt sich dies dort signifikant aus.

Insgesamt sehen wir, daß die Grauwerte in Abbildung 7.2 sehr kontinuierlich von oben nach unten zunehmen. Wir erhalten wenige Störungen durch Rauschen im Bild. Auch unterschiedliche Beleuchtungsverhältnisse stören unser Verfahren nicht. Links oben in der Ecke (über Bereich 2) sind zum Beispiel sehr hohe Grauwerte von 180 bis 255 im Bild, die gemessenen Fokuswerte unterscheiden sich mit ca. 0,42 aber nur minimal von denen im rechten oberen Teil des Bildes (Bereich 3) mit ca. 0,41, in denen die Grauwerte mit 80 bis 185 geringer sind. Nur die Bereiche, die sehr geringe Grauwertübergänge zeigen, liefern keine Tiefeninformation.

Abbildung 7.3 zeigt eine Szene mit zwei Tischen in großer Entfernung. Diese Szene ist nicht sehr gut ausgeleuchtet. Das große Lineal auf dem vorderen Tisch ist 2,3 bis 3,25 m von der Kamera entfernt, der Zollstock auf dem hinteren Tisch 2,2 bis 4,1 m. Die Wand mit dem Schachbrettmuster ist ca. 6,2 m entfernt.

Durch die geringe Ausleuchtung erhalten wir nur recht niedrige Filterantworten. Das auf dem hinteren Tisch liegende Muster zeigt hier deutliche Tiefenunterschiede von vorne nach hinten, wie es zu erwarten ist. Da die linke Seite

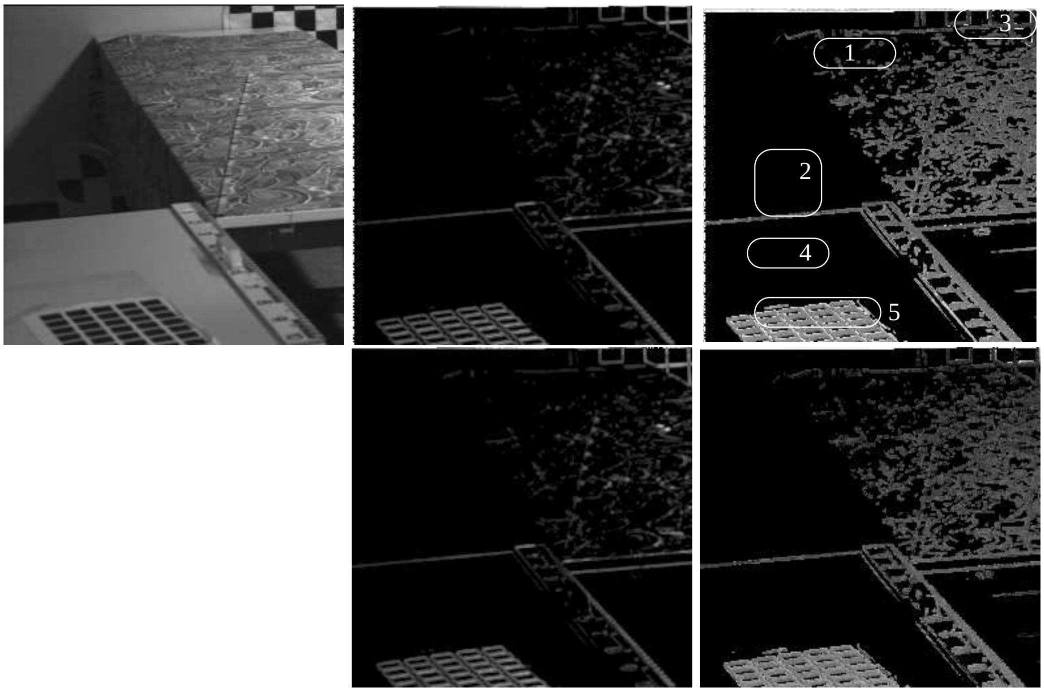

Abbildung 7.3: Testaufnahme mit großen Entfernungen (3 bis 6 m). Die Anordnung ist wie in Abbildung 7.2. Links die Szene, mittig das Maximum des Schärfefilters, rechts die gemessenen Entfernungen. In den beiden Zeilen sind die verschiedenen Richtungen der Tiefenmessung gezeigt. Die gemessenen Fokuswerte sind generell niedriger als in Abbildung 7.2, in der obere linke Ecke des Bildes sind sie sehr niedrig, und werden kontinuierlich bis zum unteren Rand des Bildes größer. Die Filterantworten (mitte) sind durch die geringe Ausleuchtung teilweise sehr schwach, daher liefert das Verfahren z.B. für Bereich 2 keine und für Bereich 1 nur wenige Tiefenwerte.

des Tisches (um Bereich 1) schlechter ausgeleuchtet ist, erhalten wir dort nur noch sporadisch eine Filterantwort. Das Viertelkreismuster (Bereich 2), das im Schatten des hinteren Tisches an der Wand hängt, wird gar nicht erkannt. Das Schachbrettmuster in der rechten oberen Ecke des Bildes (Bereich 3) wird gut erkannt. Hier ergeben sich aber ähnlich wie bei Abbildung 7.2 Streifen falscher Tiefe auf beiden Seiten einer erkannten Kante.

Auf der vorderen Tischplatte (Bereich 4) erhalten wir keine Filterantworten, so daß keine Tiefenangaben möglich sind. Das Quadratmuster auf dem vorderen Tisch (Bereich 5) liefert sehr hohe Filterantworten, so daß wir dort wieder Tiefeninformationen erhalten. Ähnlich wie beim Schachbrettmuster sind diese aber wieder von falschen Tiefeninformationen umgeben.

Die bisher festgestellten falschen Tiefeninformationen aus den beiden Experimenten können wir jedoch sehr einfach aus den Ergebnissen entfernen. Wir müssen dafür die Höhe der Filterantworten betrachten, und nur für die hohen Werte in einem Fenster die Tiefeninformationen beachten. Im einfachsten Fall kann ein Schwellenwert auf dem Maximumsbild und eine anschließende Und-Verknüpfung mit der Tiefenkarte eine Verbesserung liefern. Dadurch werden aber insgesamt niedrige Filterantworten, wie z.B. in Abb. 7.3 (Bereich 1), völlig aus der Tiefenkarte entfernt. Eine weitere Lösung wurde durch eine einfache Heuristik gewonnen: Es wird der Median des Maximumsbildes über ein $5{\times}5$-Fenster berechnet, dann dieses Bild vom Maximumsbild abgezogen und ein Schwellenwert darauf angewendet. Eine Und-Verknüpfung mit der Tiefenkarte liefert ein skelettiertes Bild der Filterantworten. Abbildung 7.4 zeigt die Ergebnisse am Beispiel der Bilder aus Abbildung 7.3. Ein einfacher Schwellenwert löscht zu viele Tiefenwerte. Die Median-Lösung sorgt zwar für sehr schmale Kanten in der Tiefenkarte, es werden jedoch die Maxima einer lokalen Umgebung in der Tiefenkarte belassen. Eine weitere Lösung wäre z.B. ein Filter auf der Basis eines lokalen Maximums.

In einem weiteren Experiment wollen wir untersuchen, wie das Verfahren auf sehr nahe Objekte reagiert. In Abbildung 7.5 haben wir das Kalibrierungsmuster schräg vor die Kamera gestellt. Die linke Seite des Musters ist ca. 0,3 m von der Kamera entfernt, die rechte Seite 0,7 m. Am Rand der rechten Seite (Bereich 1) sehen wir die Kante der Stellwand, an der das Kalibrierungsmuster befestigt ist. Diese Kante hat ca. 1 m Abstand von der Kamera. Die Kreise des Kalibrierungsmusters können wir auf dem gezeigten Bild nur erahnen. Am deutlichsten sind hier noch die drei großen Kreise zur Nullpunktmarkierung (Bereiche 2,3,4) erkennbar. Der Grund ist, daß diese Entfernungen von der verwendeten Kamera nicht scharf abgebildet werden können. Erst ab ca. 1 m Entfernung können wir scharfe Bilder erwarten. Das Ergebnis der Tiefenschätzung zeigt, daß der Filter nur bei Objekten ab

Abbildung 7.4: Verbesserung der Tiefenmessung aus Abbildung 7.3. Durch Einbeziehung des Maximumsbildes nach Anwendung eines Schwellenwertes verschwinden viele Tiefenwerte (mitte), mit Einbeziehung des Medians des Maximumsbildes ergibt sich eine skelettierte Tiefenkarte (rechts), die auch bei niedrigen Maxima einen Tiefenwert liefert.

0,5 m Entfernung geantwortet hat. Hohe Filterantworten ergeben sich erst ab 1 m Entfernung. Nähere Objektpunkte ergeben sehr niedrige Filterantworten. Die Entfernungswerte für die erkannten Bildobjekte weichen für beide Aufnahmerichtungen voneinander ab. Wir erhalten für Bereich 3 den Fokuswert 1 für die Hin-Richtung und Fokuswerte 0,45-0,7 für die Rückrichtung. Wir haben also kein globales Maximum des Tenengrads gefunden, sondern ein Plateau, auf dem für eine Anzahl von Fokuspositionen das gleiche Maximum berechnet wird. Die Maxima der Filterfunktion sind sehr niedrig, daher sehen wir im Maximumsbild kaum Antworten an diesen Stellen. Erst die Kante der Stellwand zeigt große Filterantworten und wird dort auch mit dem geringstmöglichen Abstand, also Fokusposition 1, gemessen.

Wir erhalten für dieses Experiment z.B. im Bereich 2 einen sehr weiten Bereich für die Tiefenschätzung. In realistischen Entfernungen, also weiter als 1 m, erhalten wir einen wesentlich schmaleren Bereich. Daher könnten wir diese fehlerhaften Tiefenschätzungen z.B. über die Differenz der beiden Tiefenkarten ausschließen.

In einem weiteren Experiment (Abbildung 7.6) untersuchen wir die Anfälligkeit für Störungen in der Abbildung von sich überlagernden Kanten. Wir nehmen einen Siemensstern in 1,65 m Entfernung auf. Beim Siemensstern ergeben sich durch die verschiedenen Fokuspositionen Kreise, die unser Verfahren stören können (siehe Abschnitt 2.4). Wir stellen fest, daß die Störungen nicht zu falschen Tiefeninformationen führen.

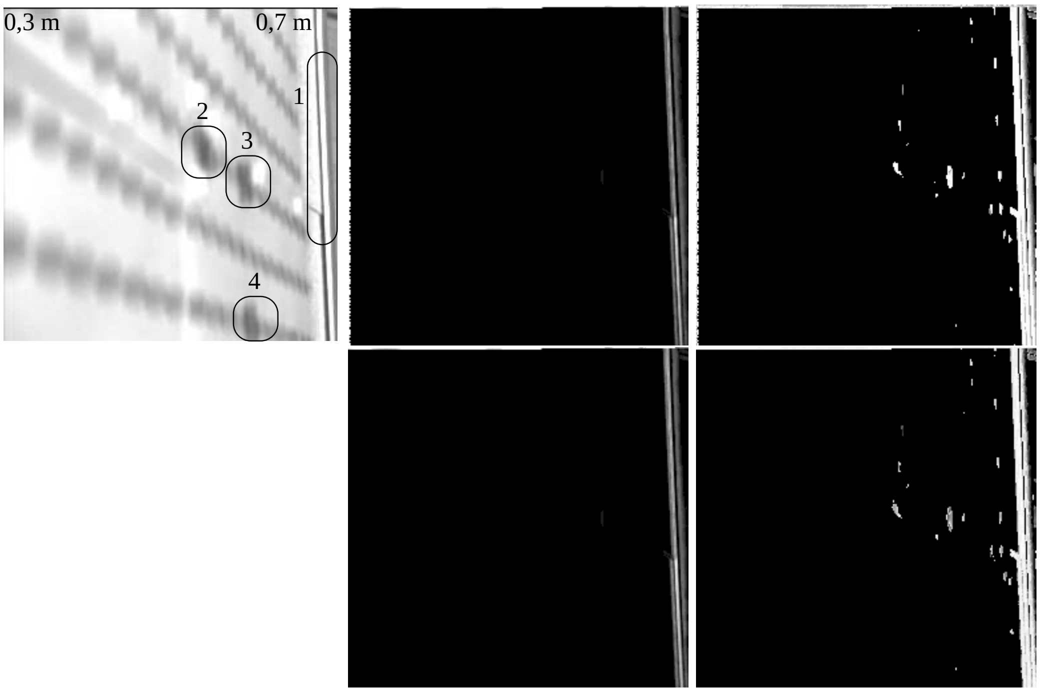

Abbildung 7.5: Testaufnahme mit kleinen Entfernungen. Die Anordnung ist wie in Abbildung 7.2: Links die Szene, mittig das Maximum des Schärfefilters, rechts die gemessenen Entfernungen. In den beiden Zeilen sind die verschiedenen Richtungen der Tiefenmessung gezeigt. Das Maximum der Filterantworten ist fast im ganzen Bild 0. Nur an der Kante (Bereich 1) in 1 m Entfernung ergeben sich hohe Filterantworten. Um die Bereiche 2 und 3 ergibt sich eine schwache Filterantwort. Dementsprechend ist auch die Tiefenkarte fast komplett schwarz, nur im Bereich 1 liefert das Verfahren sinnvolle Tiefenwerte. Um die Bereiche 2 und 3 sind die Tiefenwerte für die beiden Richtungen der Fokussierung sehr unterschiedlich.

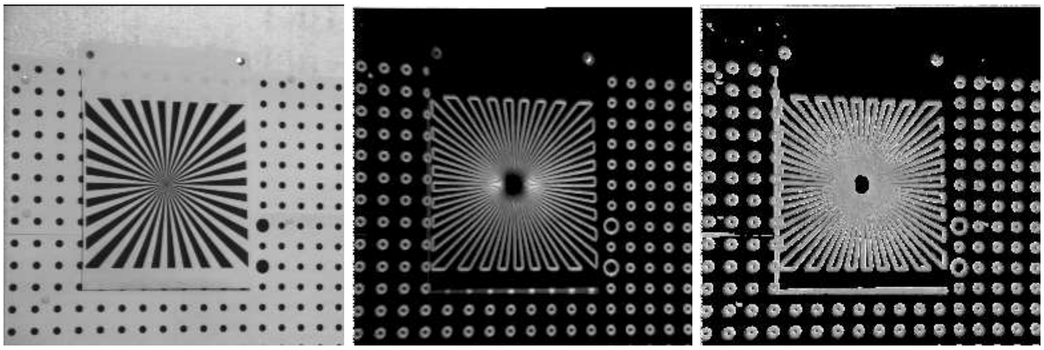

Abbildung 7.6: Aufnahme des Siemenssterns in 1,65 m Entfernung. Links die Aufnahme, mittig die Filterantwort und rechts die Tiefenkarte. Durch die Überlagerungen der Kanten des Siemenssterns treten keine Fehler bei der Tiefenmessung auf.

7.1 Umrechnung der Tiefenwerte in Entfernungen

Bisher wurden nur die Fokuspositionen als Tiefe betrachtet. Mit einer Kalibrierung auf die Entfernung kann die Fokusposition in eine Entfernung umgerechnet werden. Dabei wird ein Schwarz-Weiß-Übergang in verschiedenen Entfernungen aufgenommen. Das Tiefenmeßverfahren liefert dazu die entsprechende Fokusposition. Das Ergebnis zeigt Abbildung 7.7. Die Entfernungskurve ist nicht linear. Dies war zu erwarten, da eine lineare Änderung der Fokusposition eine nichtlineare Änderung der scharf abgebildeten Objektentfernung impliziert.

Zu den Experimenten aus Abbildung 7.2 und 7.3 wurden einige Entfernungswerte gemessen, deren genaue Position in den Bildern jedoch nicht einzeln abgebildet werden soll. Tabelle 7.1 zeigt die Entfernung der Punkte zur Kamera, die in beiden Richtungen des Verfahrens gemessenen Fokuspositionen und die Differenz der Fokuspositionen. Die Entfernungswerte entsprechen nicht denen, die in Abbildung 7.7 gemessen wurden. Diese beiden Messungen liegen zeitlich auseinander, daher könnte sich z.B. die Zoomstellung der Kamera verstellt haben. Die festgestellten Fehler durch Rauschen liegen bei den Fokuswerten durchgehend bei ca. 0,03 Counts, also $\frac{2}{64}$ Counts.

Da sich aus diesen Werten nur schlecht eine Abschätzung der Fehler herleiten läßt, werden die Filterantworten einiger Entfernungen in Abbildung 7.8 betrachtet. Die Filterantworten wurden auf gleiche Größen skaliert. Die Filterantwort bei 1 m zeigt keinen glatten Verlauf, da der Kontrast der betrachteten Kante sehr niedrig ist, und damit die Filterantwort absolut dement-

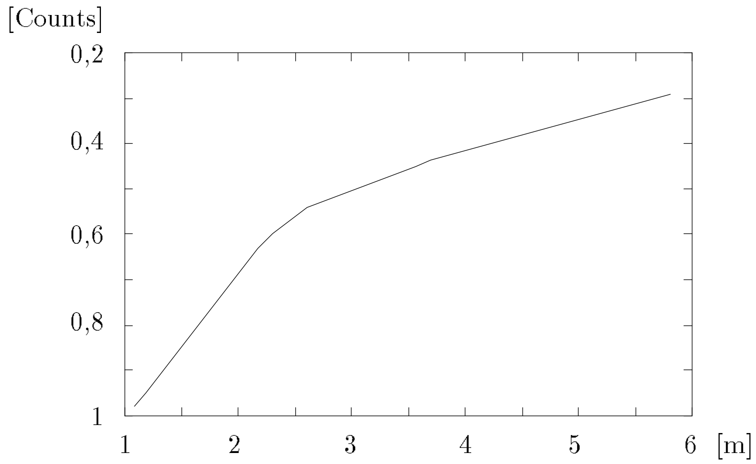

Abbildung 7.7: Kalibrierung der Fokusposition auf die Entfernung.

Nummer	Entfernung	Counts 0→1	Counts 1→0	δ_{Counts}
z2	1.67	0.671875	0.625	0.046875
z3	1.76	0.625	0.578125	0.046875
z4	1.86	0.578125	0.53125	0.046875
z5	2.00	0.5625	0.515625	0.046875
z6	2.19	0.53125	0.484375	0.046875
z1	2.33	0.5	0.46875	0.03125
z7	2.42	0.484375	0.46875	0.015625
z8	2.51	0.453125	0.453125	0
z9	2.56	0.4375	0.421875	0.015625
z10	2.70	0.421875	0.40625	0.015625
z11	2.94	0.421875	0.390625	0.03125
z12	3.08	0.40625	0.390625	0.015625
l1	2.35	0.5	0.46875	0.03125
l2	2.49	0.484375	0.46875	0.015625
l3	2.68	0.46875	0.4375	0.03125
l4	2.92	0.453125	0.40625	0.046875
l5	3.25	0.390625	0.359375	0.03125
w6	3.20	0.421875	0.390625	0.03125
w7	3.39	0.390625	0.328125	0.0625
w8	3.77	0.328125	0.296875	0.03125
w9	4.15	0.3125	0.28125	0.03125
w10	4.53	0.296875	0.265625	0.03125
w11	4.91	0.28125	0.25	0.03125
s12	6.20	0.25	0.234375	0.015625

Tabelle 7.1: Meßwerte des Verfahrens aus Abbildungen 7.2 und 7.3.

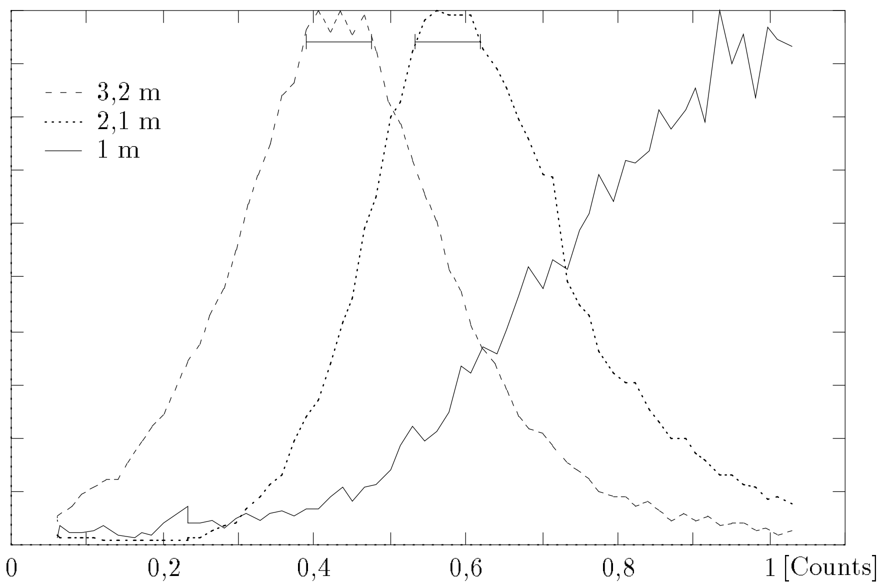

Abbildung 7.8: Filterantworten bei 3,2 m, 2,1 m und 1 m (auf gleiche Größen skaliert).

sprechend niedrig ausfällt. Daher wirkt sich vorhandenes Rauschen im Bild auf diese Filterantwort aus. Die beiden anderen Filterantworten bei 2,1 m und bei 3,2 m sind absolut höher, daher ist die Filterantwort dieser beiden Kurven glatter. Diese Filterantworten wurden unter Khoros berechnet. Da dort bisher keine Korrektur auf die Einheitsgröße berechnet wird, liegen diese Messungen nah am Bildzentrum (x^*, y^*).

Die Filterantworten bei 3,2 m und 2,1 m sind ähnlich schmal. Insgesamt zeigt dies, daß die absolute Breite der Filterantworten in Counts im wesentlichen über die Entfernung gleich bleibt. Vermutet wird, daß die Breite der Filterantworten vom Tiefenschärfebereich abhängt. Zur Abschätzung der Genauigkeit wird vorerst die Breite des Maximums bei 3,2 m betrachtet. Diese beträgt 0,09 Counts. Das Maximum erstreckt sich von 0,38 Counts bis 0,47 Counts. Das sind nach Tabelle 7.1 3,5 m bis 2,7 m (nur Minima betrachtet). In der folgenden Rechnung soll der Radius des Sensors für diesen Tiefenschärfebereich berechnet werden. Mit diesem Radius kann dann für verschiedene Entfernungen der Tiefenschärfebereich berechnet werden, und mit den Werten in Tabelle 7.1 in Counts umgerechnet werden. Der Radius des Sensors für diesen Entfernungsbereich kann nach den Gleichungen aus Abschnitt 2.3 berechnet werden:

$$g_1 = 3,5 \text{ m} \tag{7.1}$$
$$g_3 = 2,7 \text{ m} \tag{7.2}$$
$$b_1 = \frac{1}{\frac{1}{f} - \frac{1}{g_1}} = 34,843 \text{ mm} \tag{7.3}$$
$$b_3 = \frac{1}{\frac{1}{f} - \frac{1}{g_3}} = 34,947 \text{ mm} \tag{7.4}$$
$$r = \frac{A}{2}\frac{b_3 - b_1}{b_3 + b_1} = 25,026 \ \mu\text{m} \tag{7.5}$$

Der Radius ist also 5 mal so groß wie der der verwendeten Kamera mit 5,5 μm (siehe Kapitel 2). Mit r kann nun für die Entfernungen 6,2 m und 1,67 m die Breite des sich ergebenden Maximums berechnet werden:

$$g_1 = 6,2 \text{ m} \tag{7.6}$$
$$b_1 = \frac{1}{\frac{1}{f} - \frac{1}{g_1}} = 34,693 \text{ mm} \tag{7.7}$$
$$b_3 = \frac{b_2}{1 - \frac{2r}{A}} = b_1 \frac{1 + \frac{2r}{A}}{1 - \frac{2r}{A}} = 34,796 \text{ mm} \tag{7.8}$$
$$g_3 = \frac{1}{\frac{1}{f} - \frac{1}{b_3}} = 4,05 \text{ m} \tag{7.9}$$

$$g_3 = 1,67 \text{ m} \tag{7.10}$$
$$b_3 = \frac{1}{\frac{1}{f} - \frac{1}{g_3}} = 35,228 \text{ mm} \tag{7.11}$$
$$b_1 = b_3 \frac{1 - \frac{2r}{A}}{1 + \frac{2r}{A}} = 35,124 \text{ mm} \tag{7.12}$$
$$g_1 = \frac{1}{\frac{1}{f} - \frac{1}{b_1}} = 1,94 \text{ m} \tag{7.13}$$

In Counts ergibt dies: 4,05 m $\widehat{=}$ 0,31 Counts und 1,94 m $\widehat{=}$ 0,57 Counts. Es ergibt sich zu 6,2 m $\widehat{=}$ 0,25 Counts ein Unterschied von 0,06 Counts, und zu 1,67 m $\widehat{=}$ 0,67 Counts ein Unterschied von 0,1 Counts. Dies unterstützt die Beobachtung, daß die Breite der Maxima ungefähr gleich bleibt. Auch in der Tabelle der Meßwerte zeigt sich dieses Verhalten: Bei 1,67 m liegt der Unterschied der gemessenen Counts bei ca. 0,05 Counts, bei ca. 2,5 m bei 0,04 Counts, und fällt dann bis 6,2 m (bis auf einige Ausreißer in der Messung) auf

0,3 Counts. Dies ist proportional zu der theoretisch berechneten Filterbreite von 0,1 Counts bei 1,67 m und 0,06 Counts bei 6,2 m.

Es folgt aus dieser Rechnung, daß die erzielbare Genauigkeit des Verfahrens vom Tiefenschärfebereich abhängig ist. Die Breite der Maxima läßt sich in einen konstanten Durchmesser auf der Bildebene umrechnen, der ca. 2,5 mal größer ist als ein Pixel des CCD-Moduls der Kamera. Da der Tiefenschärfebereich im Nahbereich (1-2 m) wesentlich kleiner als im Fernbereich ($>$ 4 m) ist, kann das Meßverfahren hier auch genauere absolute Ergebnisse liefern. Die Breite der Maxima der Filterantworten ist insgesamt $<$ 0,05 Counts, also ist auch der Meßfehler $<$ 5 %.

Die örtliche Breite der signifikanten Filterantwort beträgt ca. 10 Pixel um eine Grauwertkante. Der mittlere Bereich von ca. 5 Pixeln zeigt im Maximumsbild das Maximum als Plateau. Damit werden kleinere Strukturen also vom Schärfefilter nicht mehr erkannt. Die Ortsauflösung des Verfahrens liegt damit bei 5 Pixeln. Dies entspricht der Ausdehnung des Filters durch die Summierung über die 4×4-Umgebung. Örtlich kann noch ein weiterer Fehler beobachtet werden: In Abbildung 7.3, Bereich 3 zeigen sich falsche Tiefenwerte rechts und links der senkrechten Kanten. Diese kommen durch den großen Grauwertunterschied der Grauwertkante zustande. Da die Filterantwort des Schärfefilters an einer scharfen Grauwertkante eine kleinere Ausdehnung hat als die Filterantwort an einer unscharfen Grauwertkante, antwortet der Schärfefilter neben der Grauwertkante nicht bei einer scharfen Abbildung dieser Kante, aber relativ niedrig bei einer unscharfen Abbildung dieser Kante. Bei sehr starken Grauwertkanten antwortet dann der Schärfefilter neben der Kante bei einer unscharfen Fokusposition signifikant. Dieser Effekt kann aber durch die oben beschriebene Verbesserung des Verfahrens korrigiert werden.

Kapitel 8

Diskussion und Ausblick

Es sollen die vorgestellten Verfahren kurz zusammengefasst und die Ergebnisse diskutiert werden. In einem Ausblick werden Hinweise auf mögliche Weiterentwicklungen geben.

Das vorgestellte *Depth-from-Focus*-Verfahren wurde auf dem MV200-System implementiert. Wir haben den Tenengrad als Schärfefilter realisiert. Die Filterantworten des Tenengrads wurden mit Hilfe eines Normierungsverfahrens auf eine einheitliche Größe gebracht, so daß zu den Maxima der Filterantworten die aktuelle Fokusposition festgestellt werden konnte. Dadurch erhalten wir eine flächige Tiefenkarte der betrachteten Szene. In verschiedenen Experimenten haben wir dieses Verfahren getestet. Es arbeitet in normalen Entfernungsbereichen ab 1 m gut. Unterhalb von 1 m erhalten wir keine Tiefeninformationen, da die verwendete Kamera nur bis 1 m fokussieren kann.

In Kapitel 4.2 wurde festgestellt, daß durch die Fokussierung sich die Bildgröße der aufgenommenen Bilder verändert. Da eine konstante Bildgröße für den Vergleich der Schärfefilterantworten benötigt wird, ist eine Kalibrierung der Bildgröße in Abhängigkeit von der Fokusposition durchgeführt worden. Es wurde festgestellt, daß sich die Bildgröße bei einer Fokussierung um bis zu 7% um den Bildhauptpunkt $(x^*; y^*) = (378; 288)$ vergrößert.

Die möglichen Schärfefilter wurden in Kapitel 3 vorgestellt. Als geeigneter Filter hat sich hier der Tenengrad herausgestellt. Er läßt sich auf der verwendeten Rechnerplattform implementieren und gibt gute Ergebnisse im Verfahren, d.h. gut detektierbare Maxima.

Die Breite der an einer Kante berechneten Tiefenwerte liegt bei ca. 10 Pixel. Das Maximum der Filterantwort ist quantisiert ca. 5 Pixel breit. Die Ortsauflösung des Verfahrens liegt demnach bei 5 Pixeln. Feinere Strukturen werden vom Schärfefilter nicht erkannt.

Weiterhin stellt das Verfahren an kontrastarmen Grauwertkanten keine Tiefenwerte fest. Aufgrund der 8-Bit-Arithmetik des verwendeten Bildverarbeitungsrechners fallen die Filterantworten für diese Grauwertkanten so niedrig aus, daß die auf 8 Bit quantisierte Filterantwort gleich Null ist. Möglich wäre es hier, das Verfahren auf andere Weise auf dem Bildverarbeitungsrechner zu implementieren. Durch Erweiterung des Algorithmus auf mit 16 Bit quantisierten Filterantworten würde ein geringerer Fehler erreicht. Sehr niedrige Filterantworten würden nicht mehr zu Null quantisiert, und das Verfahren liefert auch bei niedrigen Filterantworten die Maxima und Tiefenwerte.

Am Rand der Tiefenkarten stellen wir bei Experimenten falsche Tiefeninformationen fest. Diese kommen durch eine nicht ganz exakte Kalibrierung der ermittelten Vergrößerungsfaktoren zustande. Beim Anfahren einer Fokusposition erreicht die Kamera die angefahrene Position nicht exakt, und dadurch erhalten wir für die aufgenommene Szene einen anderen Vergrößerungsfaktor. In der Bildmitte beeinflußt der Fehler des Vergrößerungsfaktors in der Kalibrierung die berechneten Tiefenkarte nicht so stark, erst am Rand der Tiefenkarte wirken sich diese Fehler aus. Wir könnten z.B. einen Teil dieser Fehler nur durch Betrachten des entsprechenden Maximums der Fehlerantwort eliminieren. Dadurch wird die Ortsauflösung des Verfahrens weiter verschlechtert. Die Tiefe kann nur noch für große Bereiche von ca. 20×20 Pixeln festgestellt werden.

Die Laufzeit des Verfahrens ist mit ca. 0,3 Sekunden pro Fokusstellung erheblich. Insgesamt kommen wir damit bei 32 angefahrenen Fokuspositionen zu einer Laufzeit von 10 Sekunden für eine Bewegungsrichtung des Verfahrens. In Abschnitt 5.5 haben wir gesehen, daß die Kamera eine einzelne Fokussierung auch in ca. 0,3 Sekunden durchführt. Das Rechnenverfahren könnte ohne weiteres beschleunigt werden, indem mehrere Rechenschritte zu einer PAT zusammengefasst werden. Trotzdem müßte das Verfahren auf die Kamerabewegung warten. Eine weitere Möglichkeit wäre es, der Kamera einen Fokussierbefehl von 0 auf 1 zu geben und während der laufenden Fokussierung das Verfahren anzuwenden. Wir erhielten damit eine Laufzeit von 3 Sekunden für eine Richtung. Ein Nachteil wäre es, daß die genaue Fokusposition während der Laufzeit des Verfahrens nicht mehr festgestellt werden könnte. In Kapitel 7 wurde schon angemerkt, daß das Verfahren im äußeren Bereich der Tiefenkarten empfindlich auf falsche Fokuspositionen reagiert. Weitere Probleme würden sich durch die unbekannten Fokuspositionen während der direkten Fokussierung ergeben. Die Vergrößerungsfaktoren lassen sich aufgrund der unbekannten Fokusposition nicht mehr genau angeben. Weiterhin wurde bei Experimenten beobachtet, daß die Fokusposition sich nicht kontinuierlich ändert, sondern sprunghaft von einer Position auf die andere wechselt.

Die Implementierung des Verfahrens auf dem Bildverarbeitungsrechner stellt sich durch die aufwendige Programmentwicklung insgesamt als anspruchsvoll dar.

8.1 Verwendbarkeit des Verfahrens

Das vorgestellte Verfahren liefert monokular Tiefeninformationen. Diese Tiefeninformationen sind metrisch nicht sehr genau, aber in der Größenordnung sehr zuverlässig. In Experimenten wurde ein Fehler von ca. 0,05 Counts festgestellt, also ein Fehler von 5 %. Da der verwendete Kamerakopf zwei Kameras besitzt, kann das Verfahren auch für die zweite Kamera implementiert werden. Es könnte damit ein Stereoverfahren unterstützt werden. In Abschnitt 1.1 wurden schon kurz die Probleme von Stereoverfahren angesprochen, insbesondere die Zuordnung der segmentierten Punkte ist dort ein Problem. Die Tiefeninformationen können monokular bestimmt werden, und die Zuordnung mit diesen Informationen unterstützen.

In [8] wird ein Teil dieses Verfahrens für eine Vergenzsteuerung des Kamerakopfes genutzt. Dort wird ein phasenbasiertes Stereoverfahren vorgestellt, das als Voraussetzung die Überlappung beider Kamerablickfelder annimmt. Durch eine monokulare Tiefenmessung kann die Entfernung des aufzunehmenden Objektes bestimmt werden, woraus sich der Vergenzwinkel für die zweite Kamera ergibt. Dabei erweist es sich als vorteilhaft, daß die monokulare Tiefenmessung für kleine Entfernungen recht genaue Werte liefert und für große Entfernungen eher ungenau ist, denn der Vergenzwinkel muß für kleine Entfernungen recht genau sein. Für große Entfernungen dagegen sind die Änderungen des Vergenzwinkels sehr klein. Erleichternd kommt bei diesem Verfahren hinzu, daß die Tiefenwerte nur in der Mitte des Bildes gemessen werden, wo die Größenunterschiede bei unterschiedlichen Fokuspositionen nicht auftreten. Eine Vergrößerung auf die Normgröße der Filterantworten des Tenengrad entfällt.

Als Anwendung wäre auch ein Autofokussystem mit Teilen dieses Verfahren ohne Probleme realisierbar. Schon die Tiefenkarten liefern zu jedem Tiefenwert eine Fokusposition, an der die entsprechende Grauwertstruktur scharf abgebildet wird. Die genaue Fokusposition dieser Kante ergibt sich durch Anwenden der Fibonacci-Suche zwischen den beiden Seiten der Maxima, also den Tiefenkarten der Hin- und Rückrichtung. Dafür müssen aber dann mehr als die verwendeten oberen 8 Bit der Filterantwort berechnet werden. Eine Fibonacci-Suche auf dem gesamten Fokusbereich ist mit dem verwendeten Kamerasystem nicht sinnvoll, da die Fokussierung wesentlich langsamer als die Berechnung der einzelnen Filterantworten ist.

Anhang A

Technische Eigenheiten des Kamerasystems

Während der Entwicklung dieser Arbeit ergaben sich einige technische Eigenheiten des verwendeten Kamerakopfes, die im folgenden Abschnitt zur Dokumentation beschrieben werden sollen:

- Schwingen des Kopfes,
- Positionsänderungen des Kopfes bei `movelens`-Befehlen,
- Fokus- und Zoomposition werden nicht genau angefahren,
- Lange Ausführungszeit von `wherelens`-Befehlen,
- Verzögerte Ausführung von `dmovelens`-Befehlen. Dies wurde schon in Abschnitt 5.5 besprochen.

A.1 Schwingen des Kopfes

Der Kamerakopf wird mit 4 Linearmotoren auf seiner Position gehalten. Diese Linearmotoren benötigen eine ständige Nachregulierung der Position. Bei den beiden Vergenzachsen ist dies kein Problem, da die Kameras mittig unter diesen Achsen montiert sind. Das Gewicht der Kameras wirkt also nicht auf die Vergenzachsen zurück.

Die generelle Schwenkachse dreht horizontal den kompletten Kopf. Da der gesamte Kopf recht schwer ist, wirkt der ganze Aufbau wie ein Pendel: Der Kopf wird in eine bestimmte Position gebracht, schwingt durch die Masse nach und muß dann wieder auf die richtige Position nachgeregelt werden. Je nach Position des Kopfes führt dies zu horizontalen Pendelbewegungen mit ca. 10 Hz. Diese treten vor allem dann auf, wenn der Kopf nach vorne geneigt ist, meist aber auch schon in der waagerechten Stellung.

Die Neigungsachse liegt über der Schwenkachse. Diese Achse hält also das gesamte Kopfgewicht, vor allem wenn der Kopf nicht waagerecht steht, sondern nach vorn oder hinten geneigt ist. Beobachtungen zeigen, daß es hier jedoch nicht zu einem Aufschwingen kommt.

Der Aufbau des Kamerakopfes führt damit zu einem merklichen Zittern im Bereich von $\pm$ 5 Pixeln in horizontaler Richtung. In vertikaler Richtung ergeben sich nur minimale Schwankungen. Diese Schwingung ist zum einen von der Position des Kopfes abhängig, zum zweiten tritt sie bei der gleichen Position nicht immer auf. Dieses Verhalten ist auch nicht ohne weiteres vorhersagbar, und die Schwingung ist nicht gleichmäßig, sondern ändert Frequenz und Amplitude. Für alle Messungen dieser Arbeit haben wir deshalb eine Kamera auf ein Stativ gesetzt, damit dieser Effekt für die Messungen ausgeschlossen werden kann.

A.2 Der `movelens`-Befehl

Bei den ersten Versuchen, die Vergrößerung beim Fokussieren zu messen (siehe Kapitel 4), wurde eine Position mit dem Kamerakopf angefahren und dann mit dem Befehl `movelens` die Kameralinsen bewegt. Die Kamera war bei diesen Messungen nicht auf dem Stativ montiert. Es kamen unerwartete Ergebnisse bei diesen Versuchen heraus: Der Mittelpunkt $(x^*; y^*)$ verschob sich nach links oben im Kamerabild, und wanderte sogar bis zur Pixelposition $(-1000; -500)$. Die Lösung war, daß sich der Kamerakopf bei jedem `movelens`-Befehl minimal bewegt.

Erste Vermutungen zu dem Problem waren:

- Die Achsen des Kamerakopfes sind auf eine Weise miteinander verknüpft, daß eine Bewegung der Linsen eine Bewegung der anderen Kopfachsen impliziert.
- Die Achsen des Kamerakopfes bewegen sich bei jedem `movelens`-Befehl unabhängig von der Position der Linsenachsen um einen definiert Winkel.

In weiteren Tests wurde eine Position mit dem Kopf angefahren und nur diverse `movelens`-Befehle auf eine feste Position der Linsen gegeben. Ein weiterer `move`-Befehl auf die alte Position fuhr dann sichtbar den Kopf wieder auf die alte Position zurück. Nach Schätzungen läuft bei jedem `movelens`-Befehl der Kopf um 0,01 Grad aus der Schwenk- und 0,02 Grad aus der Neigungsposition. Eine Abfolge von 10 `movelens`-Befehlen bewirkt also eine Neupositionierung des Kopfes um 0,1 Grad in der Schwenkachse und 0,2 Grad in der Neigungsachse. Diese Winkel zeigen sich auch bei einer Abfrage der Achsen mit dem Kommando `where`. Bei einem einzelnen `movelens`-Befehl fällt diese Neupositionierung nicht auf, erst bei einer Abfolge von `movelens`-Befehlen summiert sich die Fehlpositionierung zu nicht einer nicht vernachlässigbaren Höhe auf. Nach jedem `movelens`-Befehl sollte daher ein `move`-Befehl auf die alte Position durchgeführt werden.

Im Direktmodus (`dmovelens` und `dmove`) tritt dieses Problem nicht auf, da der Kopf durch das Programm auf dem PMAC immer wieder angewiesen wird, seine alte Position zu halten. Falls dieser Fehler also auftritt, wird er durch die Regelung im PMAC gleich wieder korrigiert.

A.3 Ungenauigkeiten der Fokus- und Zoompositionen

Bei der Kalibrierung der Vergrößerung fielen unterschiedliche Ergebnisse für eine Fokusänderung von 0 nach 1 bzw. von 1 nach 0 auf. Weitere Tests zeigten, daß die Kameralinsen jeweils beim Hochfahren des Kopfes mit `up` eine leicht unterschiedliche Position bei Angabe der gleichen Linsenposition anfahren. Diese unterschiedliche Position kann mit `wherelens` ausgelesen werden. So ist z.B. bei dem Experiment aus Abbildung 7.1 eine Abweichung von 0,06 Counts festzustellen, beim Anfahren von Fokusposition 0 wird 0,06 Counts erreicht, beim Anfahren von Fokusposition 1 wird 1,061 erreicht. Da die mit `wherelens` ausgelesenen Fokuspositionen besser die tatsächliche Fokus- und Zoomposition zeigt, werden in allen Grafiken auch die ausgelesenen Positionen betrachtet. Auch bei der Zoomposition wird dieses Verhalten beobachtet.

A.4 Der Befehl `wherelens`

Das Kommando `wherelens` gibt die aktuellen Motorpositionen der Zoom- und Fokuslinsen aus. Dieses Kommando nutzt ein `ptv`-Kommando und liest damit für beide Kameras jeweils die Zoom- und Fokusposition aus. Im Direktmodus kann es auch während der Positionierung der Linsen angewendet werden und gibt die tatsächliche Position der Motoren aus. Nach Abschnitt

A.3 ist dies die aktuelle Linsenposition, mit einem eventuellen Offset. Implementiert ist dieses Kommando als konventionelles PMAC-Kommando. Es wird also aus dem Befehl ein Ascii-Befehl an den PMAC gemacht und auf dem PMAC ausgeführt. Auch die Positionen werden als Ascii-Zeichenkette zurückgegeben. Dieses Kommando benötigt ohne aktiven Direktmodus auf dem PMAC ca. 10 ms, mit aktivem Direktmodus ca. 60 ms. Der PMAC wird also durch das laufende Direktmodus-Programm bei der Ausführung von konventionellen PMAC-Kommandos recht langsam. Die Reihenfolge der Abfrage der Linsenpositionen können wir festlegen, wobei wir davon ausgehen, daß jede Abfrage gleich lang dauert. Eine Abfrage dauert ca. 15 ms. Ob der zurückgelieferte Wert die Position am Anfang, in der Mitte oder am Ende einer Abfrage liefert, können wir nicht feststellen. Als Konvention können wir aber die Mitte der Abfragezeit annehmen, da durch die Kommunikation des PMAC mit dem Hostrechner in beide Richtungen ähnlich viel Protokolloverhead auftritt.

Ein weiterer Test, der nur die Fokusposition der linken Kamera abfragt, gibt noch weitere Ergebnisse. Eine Abfrage dieser Position dauert im Programm ca. 30 ms. Mit der Abfrage der Systemzeit kommen wir also wieder auf die ca. 15 ms für ein `wherelens`-Kommando.

Literaturverzeichnis

[1] U. Ahlrichs, J. Denzler, R. Kompe und H. Niemann. Sprachgesteuerte Fovealisierung und Vergenz. *Proceedings in Artificial Intelligence*, 4:52–59, 1996.

[2] H. Büllthof und H.P. Mallot. Integration of stereo, shading and texture. In J.L. Crowley und H.I. Christensen, Hrsg., *AI and the Eye*, Seiten 119–146. John Wiley & Sons, Chichester, 1990.

[3] S. Das und N. Ahuja. A comparative study of stereo, vergence, and focus as depth cues for active vision. *IEEE Conf. Computer Vision and Pattern Recognition*, Seiten 194–199, 1993.

[4] Datacube, Inc., Danvers, Massachusetts. *MaxVideo 200 Hardware Reference Manual*, Oktober 1993.

[5] Datacube, Inc., Danvers, Massachusetts. *Image Flow Programmer's Manual*, Oktober 1996.

[6] J. Ernst. *Kalibrierung der internen Parameter einer aktiven Kamera.* Diplomarbeit, Christian-Albrechts-Universität zu Kiel, 1996.

[7] P. Grossmann. Depth from focus. *Pattern Recognition Letters*, 5:63–69, 1987.

[8] M. Hansen. *Stereosehen: Ein verhaltensbasierter Zugang unter Echtzeitbedingung.* Dissertation, Christian-Albrechts-Universität zu Kiel, 1998.

[9] R.A. Jarvis. A laser time-of-flight scanner. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 5:505–512, 1983.

[10] R. Klette, A. Koschan und K. Schlüns. *Computer Vision.* Vieweg Verlag, Braunschweig/Wiesbaden, 1996.

[11] E. Krotkov. Focusing. *International Journal of Computer Vision*, 1:223–237, 1987.

[12] E. Krotkov. *Active Computer Vision by Cooperative Focus and Stereo.* Springer Verlag, Berlin et al., 1989.

[13] E. Krotkov und R. Bajcsy. Active vision for reliable ranging: Cooperating focus, stereo, and vergence. *International Journal of Computer Vision*, 11:187–203, 1993.

[14] M. Li und J.-M. Lavest. *Some aspects of zoom-lens camera calibration.* Technical Report ISRN KTH/NA/P–95/03–SE, Computational Vision and Active Perception Laboratory, Royal Institute of Technology, Stockholm, Schweden, 1995.

[15] S. K. Nayar, M. Wantanabe und M. Noguchi. Real-time focus range sensor. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18:1186–1197, 1996.

[16] S. I. Olsen. Image point motion when zooming and focusing. In *Proc. 10th Scandinavian Conference on Image Analysis*, Seiten 65–70, 1997.

[17] A. Schmidt. *Manipulatorregelung durch fortwährende visuelle Rückkopplung.* Diplomarbeit, Christian-Albrechts-Universität zu Kiel, 1998.

[18] G. Schmidt. *Kleine Geschichte der modernen Malerei.* Friedrich Reinhard Verlag, Basel, 1982.

[19] G. Smith und D. A. Atchinson. *The eye and visual optical instruments.* Cambridge University Press, 1996.

[20] J. Stoer. *Numerische Mathematik 1.* Springer Verlag, Berlin et al., 1989.

[21] J. M. Tenenbaum. *Accommodation in Cmputer Vision.* Dissertation, Stanford, 1970.

[22] J. Vallino. *Datacube MV200 and ImageFlow User's Guide.* Rochester, 1995. http://www.cs.rochester.edu:80/u/vallino/research/datacube/.

[23] M. Watanabe und S. K. Nayar. Telecentric optics for computational vision. *Lecture Notes in Computer Science*, 1065:439–444, 1996.

[24] Y. Xiong und S.A. Shafer. Depth from focusing and defocusing. *IEEE Conf. Computer Vision and Pattern Recognition*, Seiten 68–73, 1993.

[25] T. T. E. Yeo, S. H. Yong, Yayasooriah und R. Sinniah. Autofocusing for tissue microscopy. *Image and Vision Computing*, 11:629–639, 1993.

Danksagung:

Ich bedanke mich bei Dipl. Phys. Michael Hansen und Prof. Dr. Gerald Sommer für die geduldige Betreuung und Unterstützung dieser Arbeit. Weiterhin möchte ich mich bei allen Kollegen, Freunden und Bekannten für Lob und Kritik bedanken. Für das Korrekturlesen danke ich Sven Paul. Mein Dank gilt vor allem meinen Eltern, die diese Arbeit durch Finanzierung meines Studiums erst ermöglicht haben.

Erklärung:

Hiermit versichere ich, daß ich die vorliegende Arbeit selbständig verfaßt und ausschließlich die angegebenen Hilfsmittel und Quellen verwendet habe.

Kiel, den