X.media.press

Wilhelm Burger · Mark James Burge

Digitale Bildverarbeitung

Eine Einführung mit Java und ImageJ

2., überarbeitete Auflage
Mit 255 Abbildungen und 16 Tabellen

Springer

Wilhelm Burger
Medientechnik und -design / Digitale Medien
Fachhochschule Hagenberg
Hauptstr. 117
4232 Hagenberg, Österreich
wilbur@ieee.org

Mark James Burge
National Science Foundation
4201 Wilson Blvd., Suite 835
Arlington, VA 22230, USA
mburge@acm.org

Bibliografische Information der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

Ursprünglich erschienen in der Reihe **eXamen.press**

ISSN 1439-3107
ISBN-10 3-540-30940-3 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-30940-6 Springer Berlin Heidelberg New York
ISBN-10 3-540-21465-8 1. Aufl. Springer Berlin Heidelberg New York

Springer ist ein Unternehmen von Springer Science+Business Media
springer.de

Satz: Druckfertige Daten der Autoren
Herstellung: LE-TEX, Jelonek, Schmidt & Vöckler GbR, Leipzig
Umschlaggestaltung: KünkelLopka Werbeagentur, Heidelberg
Gedruckt auf säurefreiem Papier 33/3100 YL – 5 4 3 2 1 0

Vorwort

Dieses Buch ist eine Einführung in die digitale Bildverarbeitung, die sowohl als Referenz für den Praktiker wie auch als Grundlagentext für die Ausbildung gedacht ist. Es bietet eine Übersicht über die wichtigsten klassischen Techniken in moderner Form und damit einen grundlegenden „Werkzeugkasten“ für dieses spannende Fachgebiet. Das Buch sollte daher insbesondere für folgende drei Einsatzbereiche gut geeignet sein:

- Für Wissenschaftler und Techniker, die digitale Bildverarbeitung als Hilfsmittel für die eigene Arbeit einsetzen oder künftig einsetzen möchten und Interesse an der Realisierung eigener, maßgeschneiderter Verfahren haben.
- Als umfassende Grundlage zum Selbststudium für ausgebildete IT-Experten, die sich erste Kenntnisse im Bereich der digitalen Bildverarbeitung und der zugehörigen Programmiertechnik aneignen oder eine bestehende Grundausbildung vertiefen möchten.
- Als einführendes Lehrbuch für eine ein- bis zweisemestrige Lehrveranstaltung im ersten Studienabschnitt, etwa ab dem 3. Semester. Die meisten Kapitel sind auf das Format einer wöchentlichen Vorlesung ausgelegt, ergänzt durch Einzelaufgaben für begleitende Übungen.

Inhaltlich steht die praktische Anwendbarkeit und konkrete Umsetzung im Vordergrund, ohne dass dabei auf die notwendigen formalen Details verzichtet wird. Allerdings ist dies kein Rezeptbuch, sondern Lösungsansätze werden schrittweise in drei unterschiedlichen Formen entwickelt: (a) in mathematischer Schreibweise, (b) als abstrakte Algorithmen und (c) als konkrete Java-Programme. Die drei Formen ergänzen sich und sollen in Summe ein Maximum an Verständlichkeit sicherstellen.

Voraussetzungen

Wir betrachten digitale Bildverarbeitung nicht vorrangig als mathematische Disziplin und haben daher die formalen Anforderungen in diesem Buch auf das Notwendigste reduziert – sie gehen über die im ersten Studienabschnitt üblichen Kenntnisse nicht hinaus. Als Einsteiger sollte man daher auch nicht beunruhigt sein, dass einige Kapitel auf den ersten Blick etwas mathematisch aussehen. Die durchgehende, einheitliche Notation und ergänzenden Informationen im Anhang tragen dazu bei, eventuell bestehende Schwierigkeiten leicht zu überwinden. Bezüglich der *Programmierung* setzt das Buch gewisse Grundkenntnisse voraus, idealerweise (aber nicht notwendigerweise) in **Java**. Elementare Datenstrukturen, prozedurale Konstrukte und die Grundkonzepte der objektorientierten Programmierung sollten dem Leser vertraut sein. Da Java mittlerweile in vielen Studienplänen als erste Programmiersprache unterrichtet wird, sollte der Einstieg in diesen Fällen problemlos sein. Aber auch Java-Neulinge mit etwas Programmiererfahrung in ähnlichen Sprachen (insbesondere C/C++) dürften sich rasch zurechtfinden.

Softwareseitig basiert dieses Buch auf **ImageJ**, einer komfortablen, frei verfügbaren Programmierumgebung, die von Wayne Rasband am U.S. National Institute of Health (NIH) entwickelt wird.[1] ImageJ ist vollständig in Java implementiert, läuft damit auf vielen Plattformen und kann durch eigene, kleine „Plugin"-Module leicht erweitert werden. Die meisten Programmbeispiele sind jedoch so gestaltet, dass sie problemlos in andere Umgebungen oder Programmiersprachen portiert werden können.

Einsatz in Forschung und Entwicklung

Dieses Buch ist einerseits für den Einsatz in der Lehre konzipiert, bietet andererseits jedoch an vielen Stellen grundlegende Informationen und Details, die in dieser Form nicht immer leicht zu finden sind. Es sollte daher für den interessierten Praktiker und Entwickler eine wertvolle Hilfe sein. Es ist aber nicht als umfassender Ausgangspunkt zur Forschung gedacht und erhebt vor allem auch keinen Anspruch auf wissenschaftliche Vollständigkeit. Im Gegenteil, es wurde versucht, die Fülle der möglichen Literaturangaben auf die wichtigsten und (auch für Studierende) leicht zugreifbaren Quellen zu beschränken. Darüber hinaus konnten einige weiterführende Techniken, wie etwa hierarchische Methoden, Wavelets, Eigenimages oder Bewegungsanalyse, aus Platzgründen nicht berücksichtigt werden. Auch Themenbereiche, die mit „Intelligenz" zu tun haben, wie Objekterkennung oder Bildverstehen, wurden bewusst ausgespart, und Gleiches gilt für alle dreidimensionalen Problemstellungen aus dem Bereich „Computer Vision". Die in diesem Buch gezeigten Verfahren sind durchweg „blind und dumm", wobei wir aber glauben,

[1] http://rsb.info.nih.gov/ij/

dass gerade die technisch saubere Umsetzung dieser scheinbar einfachen Dinge eine essentielle Grundlage für den Erfolg aller weiterführenden (vielleicht sogar „intelligenteren“) Ansätze ist.

Man wird auch enttäuscht sein, falls man sich ein Programmierhandbuch für ImageJ oder Java erwartet – dafür gibt es wesentlich bessere Quellen. Die Programmiersprache selbst steht auch nie im Mittelpunkt, sondern dient uns vorrangig als Instrument zur Verdeutlichung, Präzisierung und – praktischerweise – auch zur Umsetzung der gezeigten Verfahren.

Einsatz in der Ausbildung

An vielen Ausbildungseinrichtungen ist der Themenbereich digitale Signal- und Bildverarbeitung seit Langem in den Studienplänen integriert, speziell im Bereich der Informatik und Kommunikationstechnik, aber auch in anderen technischen Studienrichtungen mit entsprechenden formalen Grundlagen und oft auch erst in höheren („graduate“) Studiensemestern.

Immer häufiger finden sich jedoch auch bereits in der Grundausbildung einführende Lehrveranstaltungen zu diesem Thema, vor allem in neueren Studienrichtungen der Informatik und Softwaretechnik, Mechatronik oder Medientechnik. Ein Problem dabei ist das weitgehende Fehlen von geeigneter Literatur, die bezüglich der Voraussetzungen und der Inhalte diesen Anforderungen entspricht. Die klassische Fachliteratur ist häufig zu formal für Anfänger, während oft gleichzeitig manche populäre, praktische Methode nicht ausreichend genau beschrieben ist. So ist es auch für die Lektoren schwierig, für eine solche Lehrveranstaltung ein einzelnes Textbuch oder zumindest eine kompakte Sammlung von Literatur zu finden und den Studierenden empfehlen zu können. Das Buch soll dazu beitragen, diese Lücke zu schließen.

Die Inhalte der nachfolgenden Kapitel sind für eine Lehrveranstaltung von 1–2 Semestern in einer Folge aufgebaut, die sich in der praktischen Ausbildung gut bewährt hat. Die Kapitel sind meist in sich so abgeschlossen, dass ihre Abfolge relativ flexibel gestaltet werden kann. Der inhaltliche Schwerpunkt liegt dabei auf den klassischen Techniken im Bildraum, wie sie in der heutigen Praxis im Vordergrund stehen. Die Kapitel 13–15 zum Thema Spektraltechniken sind hingegen als grundlegende Einführung gedacht und bewusst im hinteren Teil des Buchs platziert. Sie können bei Bedarf leicht reduziert oder überhaupt weggelassen werden. Die nachfolgende Übersicht (auf Seite VIII) zeigt mehrere Varianten zur Gliederung von Lehrveranstaltungen über ein oder zwei Semester.

1 Semester: Für einen Kurs über *ein* Semester könnte man je nach Zielsetzung zwischen den Themenschwerpunkten **Bildverarbeitung** und **Bildanalyse** wählen. Beide Lehrveranstaltungen passen gut in den ersten Abschnitt moderner Studienpläne im Bereich der

„Road Map“ für 1- und 2-Semester-Kurse	Bildverarb.	Bildanalyse	Grundlagen	Vertiefung
1. Crunching Pixels	☒	☒	☒	☐
2. Digitale Bilder	☒	☒	☒	☐
3. ImageJ	☒	☒	☒	☐
4. Histogramme	☒	☐	☒	☐
5. Punktoperationen	☒	☐	☒	☐
6. Filter	☒	☒	☒	☐
7. Kanten und Konturen	☒	☒	☒	☐
8. Auffinden von Eckpunkten	☐	☒	☐	☒
9. Detektion einfacher Kurven	☐	☒	☐	☒
10. Morphologische Filter	☒	☐	☒	☐
11. Regionen in Binärbildern	☐	☒	☒	☐
12. Farbbilder	☒	☐	☐	☒
13. Einführung in Spektraltechniken	☐	☒	☐	☒
14. Die diskrete Fouriertransformation in 2D	☐	☒	☐	☒
15. Die diskrete Kosinustransformation	☐	☐	☐	☒
16. Geometrische Bildoperationen	☒	☐	☐	☒
17. Bildvergleich	☐	☒	☐	☒
	1 Sem.		2 Sem.	

Informatik oder Informationstechnik. Der zweite Kurs eignet sich etwa auch als Grundlage für eine einführende Lehrveranstaltung in der medizinischen Ausbildung.

2 Semester: Stehen *zwei* Semester zur Vermittlung der Inhalte zur Verfügung, wäre eine Aufteilung in zwei aufbauende Kurse (*Grundlagen* und *Vertiefung*) möglich, wobei die Themen nach ihrem Schwierigkeitsgrad gruppiert sind.

Ergänzungen zur 2. Auflage

Die vorliegende zweite Auflage der deutschen Ausgabe enthält neben zahlreichen kleineren Korrekturen und Verbesserungen zwei ergänzende Abschnitte zu den Themen Histogrammanpassung (Kap. 5) und Lanczos-Interpolation (Kap. 16). Soweit sinnvoll wurden die Beispielprogamme auf Java 5 aktualisiert. Dank der Unterstützung von Seiten des Verlags erhielt diese Ausgabe auch ein moderneres Layout, hochwertigeres Papier und einige wichtige Seiten in Farbe.

Online-Materialien

Auf der Website zu diesem Buch

www.imagingbook.com

stehen zusätzliche Materialien in elektronischer Form frei zur Verfügung, u. a. Testbilder in Originalgröße und Farbe, Java-Quellcode für die angeführten Beispiele, aktuelle Ergänzungen und etwaige Korrekturen. Für Lehrende gibt es außerdem den vollständigen Satz von Abbildungen und Formeln zur Verwendung in eigenen Präsentationen. Kommentare, Fragen, Anregungen und Korrekturen sind willkommen und sollten adressiert werden an:

imagingbook@gmail.com

Ein Dankeschön

Dieses Buch wäre nicht entstanden ohne das Verständnis und die Unterstützung unsere Familien, die uns dankenswerterweise erlaubten, über mehr als ein Jahr hinweg ziemlich schlechte Väter und Ehepartner zu sein. Unser Dank geht auch an Wayne Rasband am NIH für die Entwicklung von ImageJ und sein hervorragendes Engagement innerhalb der Community, an die Kollegen Prof. Axel Pinz (TU Graz) und Prof. Vaclav Hlavac (TU Prag) für ihre sachkundigen Kommentare sowie an die aufmerksamen Leser der ersten Auflage für die zahlreichen Kommentare und Verbesserungsvorschläge. Respekt gebührt nicht zuletzt Ursula Zimpfer für ihr professionelles Copy-Editing sowie Jutta Maria Fleschutz vom Springer-Verlag für ihre Geduld und die gute Zusammenarbeit.

Hagenberg / Washington D.C.
März 2006

Inhaltsverzeichnis

1

Crunching Pixels

Lange Zeit war die digitale Verarbeitung von Bildern einer relativ kleinen Gruppe von Spezialisten mit teurer Ausstattung und einschlägigen Kenntnissen vorbehalten. Spätestens durch das Auftauchen von digitalen Kameras, Scannern und Multi-Media-PCs auf den Schreibtischen vieler Zeitgenossen wurde jedoch die Beschäftigung mit digitalen Bildern, bewusst oder unbewusst, zu einer Alltäglichkeit für viele Computerbenutzer. War es vor wenigen Jahren noch mit großem Aufwand verbunden, Bilder überhaupt zu digitalisieren und im Computer zu speichern, erlauben uns heute üppig dimensionierte Hauptspeicher, riesige Festplatten und Prozessoren mit Taktraten von mehreren Gigahertz digitale Bilder und Videos mühelos und schnell zu manipulieren. Dazu gibt es Tausende von Programmen, die dem Amateur genauso wie dem Fachmann die Bearbeitung von Bildern in bequemer Weise ermöglichen.

So gibt es heute eine riesige „Community“ von Personen, für die das Arbeiten mit digitalen Bildern auf dem Computer zur alltäglichen Selbstverständlichkeit geworden ist. Dabei überrascht es nicht, dass im Verhältnis dazu das Verständnis für die zugrunde liegenden Mechanismen meist über ein oberflächliches Niveau nicht hinausgeht. Für den typischen Konsumenten, der lediglich seine Urlaubsfotos digital archivieren möchte, ist das auch kein Problem, ähnlich wie ein tieferes Verständnis eines Verbrennungsmotors für das Fahren eines Autos weitgehend entbehrlich ist.

Immer häufiger stehen aber auch IT-Fachleute vor der Aufgabe, mit diesem Thema professionell umzugehen, schon allein deshalb, weil Bilder (und zunehmend auch andere Mediendaten) heute ein fester Bestandteil des digitalen Workflows in vielen Unternehmen und Institutionen sind, nicht nur in der Medizin oder in der Medienbranche. Genauso sind auch „gewöhnliche“ Softwaretechniker heute oft mit digitalen Bildern auf Programm-, Datei- oder Datenbankebene konfrontiert und Program-

mierumgebungen in sämtlichen modernen Betriebssystemen bieten dazu umfassende Möglichkeiten. Der einfache praktische Umgang mit dieser Materie führt jedoch, verbunden mit einem oft unklaren Verständnis der grundlegenden Zusammenhänge, häufig zur Unterschätzung der Probleme und nicht selten zu ineffizienten Lösungen, teuren Fehlern und persönlicher Frustration.

1.1 Programmieren mit Bildern

Bildverarbeitung wird im heutigen Sprachgebrauch häufig mit Bild*be*arbeitung verwechselt, also der Manipulation von Bildern mit fertiger Software, wie beispielsweise *Adobe Photoshop*, *Corel Paint* etc. In der Bild*ver*arbeitung geht es im Unterschied dazu um die Konzeption und Erstellung von Software, also um die Entwicklung (oder Erweiterung) dieser Programme selbst.

Moderne Programmierumgebungen machen auch dem Nicht-Spezialisten durch umfassende APIs (Application Programming Interfaces) praktisch jeden Bereich der Informationstechnik zugänglich: Netzwerke und Datenbanken, Computerspiele, Sound, Musik und natürlich auch Bilder. Die Möglichkeit, in eigenen Programmen auf die einzelnen Elemente eines Bilds zugreifen und diese beliebig manipulieren zu können, ist faszinierend und verführerisch zugleich. In der Programmierung sind Bilder nichts weiter als simple Zahlenfelder, also Arrays, deren Zellen man nach Belieben lesen und verändern kann. Alles, was man mit Bildern tun kann, ist somit grundsätzlich machbar und der Phantasie sind keine Grenzen gesetzt.

Im Unterschied zur digitalen Bildverarbeitung beschäftigt man sich in der *Computergrafik* mit der *Synthese* von Bildern aus geometrischen Beschreibungen bzw. dreidimensionalen Objektmodellen [23,29,87]. Realismus und Geschwindigkeit stehen – heute vor allem für Computerspiele – dabei im Vordergrund. Dennoch bestehen zahlreiche Berührungspunkte zur Bildverarbeitung, etwa die Transformation von Texturbildern, die Rekonstruktion von 3D-Modellen aus Bilddaten, oder spezielle Techniken wie „Image-Based Rendering“ und „Non-Photorealistic Rendering“ [64, 88]. In der Bildverarbeitung finden sich wiederum Methoden, die ursprünglich aus der Computergrafik stammen, wie volumetrische Modelle in der medizinischen Bildverarbeitung, Techniken der Farbdarstellung oder Computational-Geometry-Verfahren. Extrem eng ist das Zusammenspiel zwischen Bildverarbeitung und Grafik natürlich in der digitalen Post-Produktion für Film und Video, etwa zur Generierung von Spezialeffekten [89]. Die grundlegenden Verfahren in diesem Buch sind daher nicht nur für Einzelbilder, sondern auch für die Bearbeitung von Bildfolgen, d. h. Video- und Filmsequenzen, durchaus relevant.

1.2 Bildanalyse und „intelligente“ Verfahren

Viele Aufgaben in der Bildverarbeitung, die auf den ersten Blick einfach und vor allem dem menschlichen Auge so spielerisch leicht zu fallen scheinen, entpuppen sich in der Praxis als schwierig, unzuverlässig, zu langsam, oder gänzlich unmachbar. Besonders gilt dies für den Bereich der Bild*analyse*, bei der es darum geht, sinnvolle Informationen aus Bildern zu extrahieren, sei es etwa, um ein Objekt vom Hintergrund zu trennen, einer Straße auf einer Landkarte zu folgen oder den Strichcode auf einer Milchpackung zu finden – meistens ist das schwieriger, als es uns die eigenen Fähigkeiten erwarten lassen.

Dass die technische Realität heute von der beinahe unglaublichen Leistungsfähigkeit biologischer Systeme (und den Phantasien Hollywoods) noch weit entfernt ist, sollte uns zwar Respekt machen, aber nicht davon abhalten, diese Herausforderung unvoreingenommen und kreativ in Angriff zu nehmen. Vieles ist auch mit unseren heutigen Mitteln durchaus lösbar, erfordert aber – wie in jeder technischen Disziplin – sorgfältiges und rationales Vorgehen. Bildverarbeitung funktioniert nämlich in vielen, meist unspektakulären Anwendungen seit langem und sehr erfolgreich, zuverlässig und schnell, nicht zuletzt als Ergebnis fundierter Kenntnisse, präziser Planung und sauberer Umsetzung.

Die Analyse von Bildern ist in diesem Buch nur ein Randthema, mit dem wir aber doch an mehreren Stellen in Berührung kommen, etwa bei der Segmentierung von Bildregionen (Kap. 11), beim Auffinden von einfachen Kurven (Kap. 9) oder beim Vergleichen von Bildern (Kap. 17). Alle hier beschriebenen Verfahren arbeiten jedoch ausschließlich auf Basis der Pixeldaten, also „blind“ und „bottom-up“ und ohne zusätzliches Wissen oder „Intelligenz“. Darin liegt ein wesentlicher Unterschied zwischen digitaler Bildverarbeitung einerseits und „Mustererkennung“ (*Pattern Recognition*) bzw. *Computer Vision* andererseits. Diese Disziplinen greifen zwar häufig auf die Methoden der Bildverarbeitung zurück, ihre Zielsetzungen gehen aber weit über diese hinaus:

Pattern Recognition ist eine vorwiegend mathematische Disziplin, die sich allgemein mit dem Auffinden von „Mustern“ in Daten und Signalen beschäftigt. Typische Beispiele aus dem Bereich der Bildanalyse sind etwa die Unterscheidung von Texturen oder die optische Zeichenerkennung (OCR). Diese Methoden betreffen aber nicht nur Bilddaten, sondern auch Sprach- und Audiosignale, Texte, Börsenkurse, Verkehrsdaten, die Inhalte großer Datenbanken u.v.m. Statistische und syntaktische Methoden spielen in der Mustererkennung eine zentrale Rolle (s. beispielsweise [21, 62, 83]).

Computer Vision beschäftigt sich mit dem Problem, Sehvorgänge in der realen, dreidimensionalen Welt zu mechanisieren. Dazu gehört die räumliche Erfassung von Gegenständen und Szenen, das Erkennen von Objekten, die Interpretation von Bewegungen, autonome Navigation, das mechanische Aufgreifen von Dingen (durch Robo-

ter) usw. Computer Vision entwickelte sich ursprünglich als Teilgebiet der „Künstlichen Intelligenz“ (*Artificial Intelligence*, kurz „AI“) und die Entwicklung zahlreicher AI-Methoden wurde von visuellen Problemstellungen motiviert (s. beispielsweise [18, Kap. 13]). Auch heute bestehen viele Berührungspunkte, besonders aktuell im Zusammenhang mit adaptivem Verhalten und maschinellem Lernen. Einführende und vertiefende Literatur zum Thema Computer Vision findet man z. B. in [4, 26, 37, 76, 80, 84].

Interessant ist der Umstand, dass trotz der langjährigen Entwicklung in diesen Bereichen viele der ursprünglich als relativ einfach betrachteten Aufgaben weiterhin nicht oder nur unzureichend gelöst sind. Das macht die Arbeit an diesen Themen – trotz aller Schwierigkeiten – spannend. Wunderbares darf man sich von der digitalen Bildverarbeitung allein nicht erwarten, sie könnte aber durchaus die „Einstiegsdroge“ zu weiterführenden Unternehmungen sein.

2

Digitale Bilder

Zentrales Thema in diesem Buch sind digitale Bilder, und wir können davon ausgehen, dass man heute kaum einem Leser erklären muss, worum es sich dabei handelt. Genauer gesagt geht es um Rasterbilder, also Bilder, die aus regelmäßig angeordneten Elementen (*picture elements* oder *pixel*) bestehen, im Unterschied etwa zu Vektorgrafiken.

2.1 Arten von digitalen Bildern

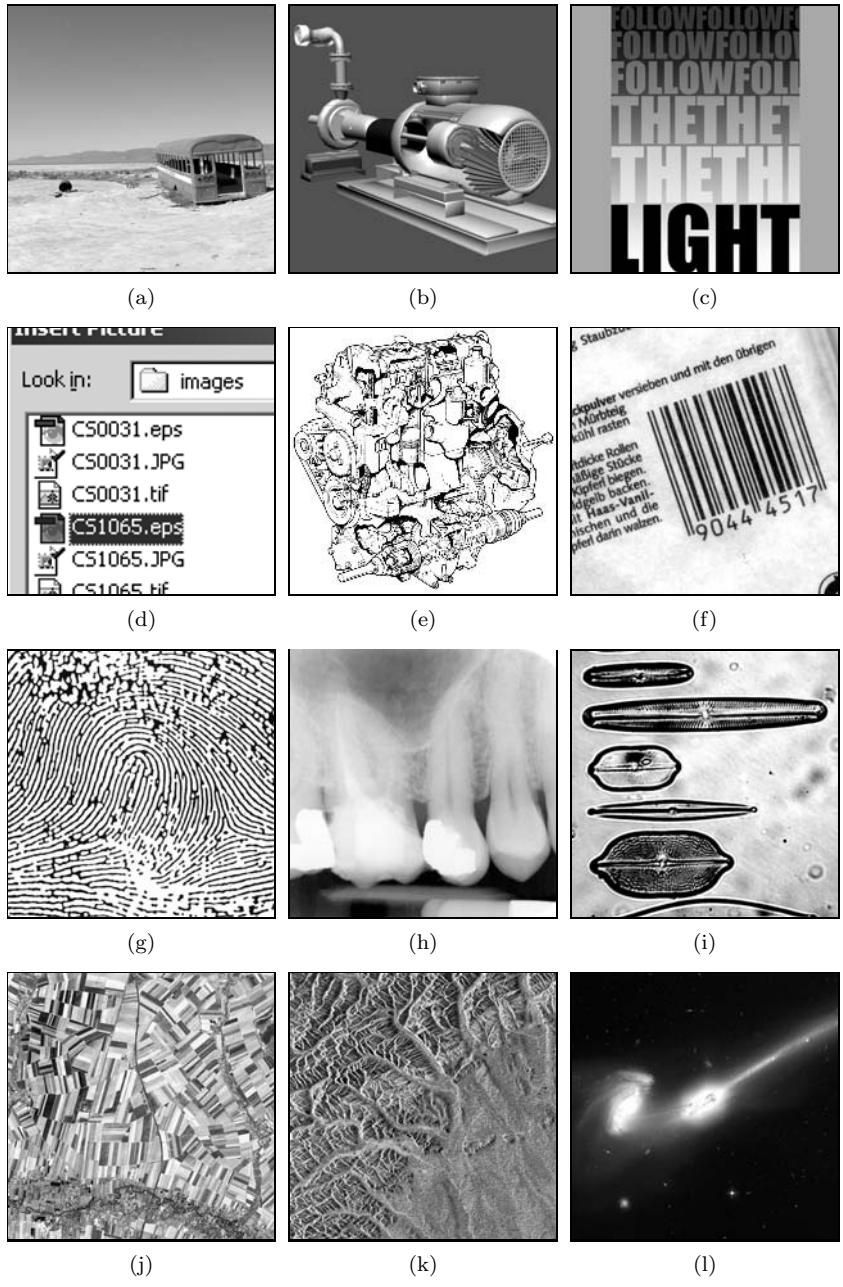
In der Praxis haben wir mit vielen Arten von digitalen Rasterbildern zu tun, wie Fotos von Personen oder Landschaften, Farb- und Grautonbilder, gescannte Druckvorlagen, Baupläne, Fax-Dokumente, Screenshots, Mikroskopaufnahmen, Röntgen- und Ultraschallbilder, Radaraufnahmen u. v. m. (Abb. 2.1). Auf welchem Weg diese Bilder auch entstehen, sie bestehen (fast) immer aus rechteckig angeordneten Bildelementen und unterscheiden sich – je nach Ursprung und Anwendungsbereich – vor allem durch die darin abgelegten Werte.

2.2 Bildaufnahme

Der eigentliche Prozess der Entstehung von Bildern ist oft kompliziert und meistens für die Bildverarbeitung auch unwesentlich. Dennoch wollen wir uns kurz ein Aufnahmeverfahren etwas genauer ansehen, mit dem die meisten von uns vertraut sind: eine optische Kamera.

2.2.1 Das Modell der Lochkamera

Das einfachste Prinzip einer Kamera, das wir uns überhaupt vorstellen können, ist die so genannte Lochkamera, die bereits im 13. Jahrhun-



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Abbildung 2.1
Digitale Bilder: natürliche Landschaftsszene (a), synthetisch generierte Szene (b), Poster-Grafik (c), Screenshot (d), Schwarz-Weiß-Illustration (e), Strichcode (f), Fingerabdruck (g), Röntgenaufnahme (h), Mikroskopbild (i), Satellitenbild (j), Radarbild (k), astronomische Aufnahme (l).

dert als „Camera obscura“ bekannt war. Sie hat zwar heute keinerlei praktische Bedeutung mehr (eventuell als Spielzeug), aber sie dient als brauchbares Modell, um die für uns wesentlichen Elemente der optischen Abbildung ausreichend zu beschreiben, zumindest soweit wir es im Rahmen dieses Buchs überhaupt benötigen.

Die Lochkamera besteht aus einer geschlossenen Box mit einer winzigen Öffnung an der Vorderseite und der Bildebene an der gegenüberliegenden Rückseite. Lichtstrahlen, die von einem Objektpunkt vor der Kamera ausgehend durch die Öffnung einfallen, werden geradlinig auf die Bildebene projiziert, wodurch ein verkleinertes und seitenverkehrtes Abbild der sichtbaren Szene entsteht (Abb. 2.2).

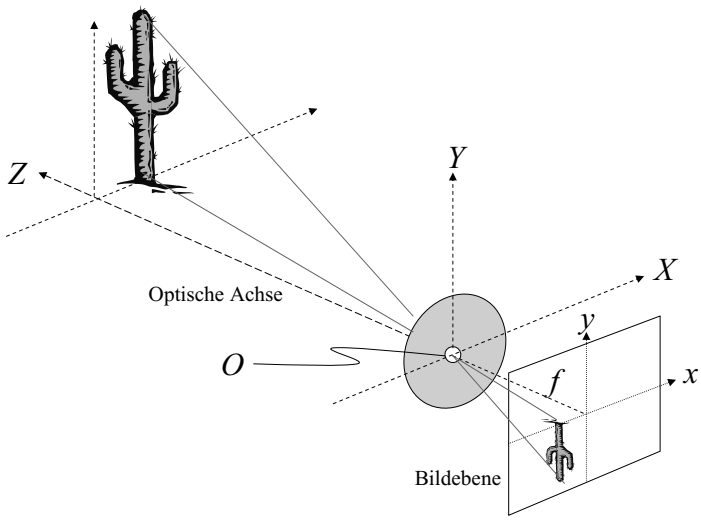


Abbildung 2.2
Geometrie der Lochkamera. Die Lochöffnung bildet den Ursprung des dreidimensionalen Koordinatensystems (X, Y, Z), in dem die Positionen der Objektpunkte in der Szene beschrieben werden. Die optische Achse, die durch die Lochöffnung verläuft, bildet die Z-Achse dieses Koordinatensystems. Ein eigenes, zweidimensionales Koordinatensystem (x, y) beschreibt die Projektionspunkte auf der Bildebene. Der Abstand f („Brennweite“) zwischen der Öffnung und der Bildebene bestimmt den Abbildungsmaßstab der Projektion.

Perspektivische Abbildung

Die geometrischen Verhältnisse der Lochkamera sind extrem einfach. Die so genannte „optische Achse“ läuft gerade durch die Lochöffnung und rechtwinkelig zur Bildebene. Nehmen wir an, ein sichtbarer Objektpunkt (in unserem Fall die Spitze des Kaktus) befindet sich in einer Distanz Z von der Lochebene und im vertikalen Abstand Y über der optischen Achse. Die Höhe der zugehörigen Projektion y wird durch zwei Parameter bestimmt: die (fixe) Tiefe der Kamerabox f und den Abstand Z des Objekts vom Koordinatenursprung. Durch Vergleich der ähnlichen Dreiecke ergibt sich der einfache Zusammenhang

$$y = -f\frac{Y}{Z} \quad \text{und genauso} \quad x = -f\frac{X}{Z}. \tag{2.1}$$

Proportional zur Tiefe der Box, also dem Abstand f, ändert sich auch der Maßstab der gewonnenen Abbildung analog zur Änderung der Brennweite in einer herkömmlichen Fotokamera. Ein kleines f (= kurze Brennweite) erzeugt eine kleine Abbildung bzw. – bei fixer Bildgröße – einen größeren Blickwinkel, genau wie bei einem Weitwinkelobjektiv. Verlängern wir die „Brennweite“ f, dann ergibt sich – wie bei einem Teleobjektiv – eine vergrößerte Abbildung verbunden mit einem entsprechend kleineren Blickwinkel. Das negative Vorzeichen in Gl. 2.1 zeigt lediglich an, dass die Projektion horizontal und vertikal gespiegelt, also um 180° gedreht, erscheint.

Gl. 2.1 beschreibt nichts anderes als die perspektivische Abbildung, wie wir sie heute als selbstverständlich kennen.[1] Wichtige Eigenschaften dieses theoretischen Modells sind u. a., dass Geraden im 3D-Raum immer auch als Geraden in der 2D-Projektion erscheinen und dass Kreise als Ellipsen abgebildet werden.

2.2.2 Die „dünne“ Linse

Während die einfache Geometrie der Lochkamera sehr anschaulich ist, hat die Kamera selbst in der Praxis keine Bedeutung. Um eine scharfe Projektion zu erzielen, benötigt man eine möglichst kleine Lochblende, die wiederum wenig Licht durchlässt und damit zu sehr langen Belichtungszeiten führt. In der Realität verwendet man optische Linsen und Linsensysteme, deren Abbildungsverhalten in vieler Hinsicht besser, aber auch wesentlich komplizierter ist. Häufig bedient man sich aber auch in diesem Fall zunächst eines einfachen Modells, das mit dem der Lochkamera praktisch identisch ist. Im Modell der „dünnen Linse“ ist lediglich die Lochblende durch eine Linse ersetzt (Abb. 2.3). Die Linse wird dabei als symmetrisch und unendlich dünn angenommen, d. h., jeder

[1] Es ist heute schwer vorstellbar, dass die Regeln der perspektivischen Geometrie zwar in der Antike bekannt waren, danach aber in Vergessenheit gerieten und erst in der Renaissance (um 1430 durch den Florentiner Maler Brunoleschi) wiederentdeckt wurden.

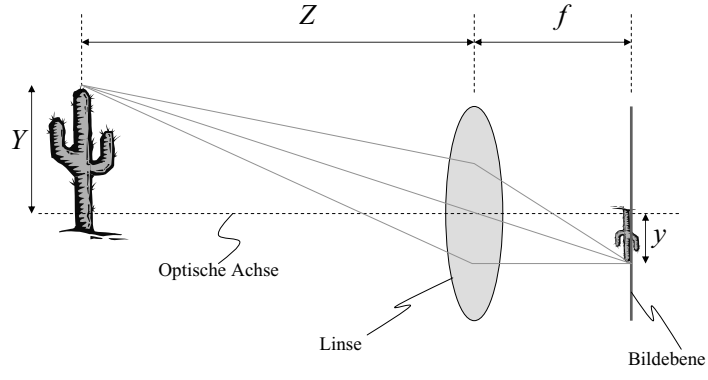


Abbildung 2.3
Modell der „dünnen Linse“.

Lichtstrahl, der in die Linse fällt, wird an einer virtuellen Ebene in der Linsenmitte gebrochen. Daraus ergibt sich die gleiche Abbildungsgeometrie wie bei einer Lochkamera. Für die Beschreibung echter Linsen und Linsensysteme ist dieses Modell natürlich völlig unzureichend, denn Details wie Schärfe, Blenden, geometrische Verzerrungen, unterschiedliche Brechung verschiedener Farben und andere reale Effekte sind darin überhaupt nicht berücksichtigt. Für unsere Zwecke reicht dieses primitive Modell zunächst aber aus und für Interessierte findet sich dazu eine Fülle an einführender Literatur (z. B. [48]).

2.2.3 Übergang zum Digitalbild

Das auf die Bildebene unserer Kamera projizierte Bild ist zunächst nichts weiter als eine zweidimensionale, zeitabhängige, kontinuierliche Verteilung von Lichtenergie. Um diesen kontinuierlichen „Lichtfilm“ als Schnappschuss in digitaler Form in unseren Computer zu bekommen, sind drei wesentliche Schritte erforderlich:

1. Die kontinuierliche Lichtverteilung muss räumlich abgetastet werden.
2. Die daraus resultierende Funktion muss zeitlich abgetastet werden, um ein einzelnes Bild zu erhalten.
3. Die einzelnen Werte müssen quantisiert werden in eine endliche Anzahl möglicher Zahlenwerte, damit sie am Computer darstellbar sind.

Schritt 1: Räumliche Abtastung (*spatial sampling*)

Die räumliche Abtastung, d. h. der Übergang von einer kontinuierlichen zu einer diskreten Lichtverteilung, erfolgt in der Regel direkt durch die Geometrie des Aufnahmesensors, z. B. in einer Digital- oder Viodeokamera. Die einzelnen Sensorelemente sind dabei fast immer regelmäßig und rechtwinklig zueinander auf der Sensorfläche angeordnet (Abb. 2.4). Es gibt allerdings auch Bildsensoren mit hexagonalen Elementen oder auch ringförmige Sensorstrukturen für spezielle Anwendungen.

Schritt 2: Zeitliche Abtastung (*temporal sampling*)

Die zeitliche Abtastung geschieht durch Steuerung der Zeit, über die die Messung der Lichtmenge durch die einzelnen Sensorelemente erfolgt. Auf dem CCD[2]-Chip einer Digitalkamera wird dies durch das Auslösen eines Ladevorgangs und die Messung der elektrischen Ladung nach einer vorgegebenen Belichtungszeit gesteuert.

[2] *Charge-Coupled Device*

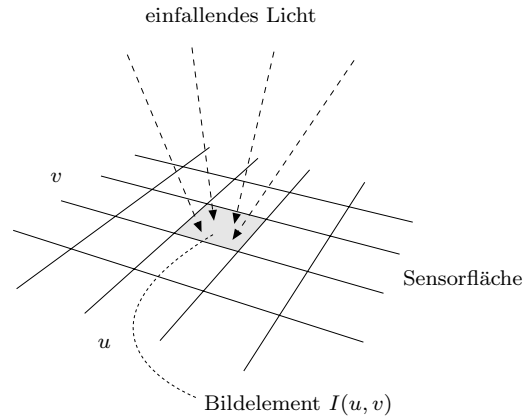


Abbildung 2.4
Die räumliche Abtastung der kontinuierlichen Lichtverteilung erfolgt normalerweise direkt durch die Sensorgeometrie, im einfachsten Fall durch eine ebene, regelmäßige Anordnung rechteckiger Sensorelemente, die jeweils die auf sie einfallende Lichtmenge messen.

Schritt 3: Quantisierung der Pixelwerte

Um die Bildwerte im Computer verarbeiten zu können, müssen diese abschließend auf eine endliche Menge von Zahlenwerten abgebildet werden, typischerweise auf ganzzahlige Werte (z. B. $256 = 2^8$ oder $4096 = 2^{12}$) oder auch auf Gleitkommawerte. Diese Quantisierung erfolgt durch Analog-Digital-Wandlung, entweder in der Sensorelektronik selbst oder durch eine spezielle Interface-Hardware.

Bilder als diskrete Funktionen

Das Endergebnis dieser drei Schritte ist eine Beschreibung des aufgenommenen Bilds als zweidimensionale, regelmäßige Matrix von Zahlen (Abb. 2.5). Etwas formaler ausgedrückt, ist ein digitales Bild I damit eine zweidimensionale Funktion von den ganzzahligen Koordinaten $\mathbb{N} \times \mathbb{N}$ auf eine Menge (bzw. ein Intervall) von Bildwerten $\mathbb{P}$, also

$$I(u, v) \in \mathbb{P} \quad \text{und} \quad u, v \in \mathbb{N}.$$

Damit sind wir bereits so weit, Bilder in unserem Computer darzustellen, sie zu übertragen, zu speichern, zu komprimieren oder in beliebiger Form zu bearbeiten. Ab diesem Punkt ist es uns zunächst egal, auf welchem Weg unsere Bilder entstanden sind, wir behandeln sie einfach nur als zweidimensionale, numerische Daten. Bevor wir aber mit der Verarbeitung von Bildern beginnen, noch einige wichtige Definitionen.

2.2.4 Bildgröße und Auflösung

Im Folgenden gehen wir davon aus, dass wir mit rechteckigen Bildern zu tun haben. Das ist zwar eine relativ sichere Annahme, es gibt aber auch Ausnahmen. Die *Größe* eines Bilds wird daher direkt bestimmt durch

$F(x,y)$

$$\longrightarrow$$

$$I(u,v) = \begin{bmatrix}
148 & 123 & 52 & 107 & 123 & 162 & 172 & 123 & 64 & 89 & \cdots \\
147 & 130 & 92 & 95 & 98 & 130 & 171 & 155 & 169 & 163 & \cdots \\
141 & 118 & 121 & 148 & 117 & 107 & 144 & 137 & 136 & 134 & \cdots \\
82 & 106 & 93 & 172 & 149 & 131 & 138 & 114 & 113 & 129 & \cdots \\
57 & 101 & 72 & 54 & 109 & 111 & 104 & 135 & 106 & 125 & \cdots \\
138 & 135 & 114 & 82 & 121 & 110 & 34 & 76 & 101 & 111 & \cdots \\
138 & 102 & 128 & 159 & 168 & 147 & 116 & 129 & 124 & 117 & \cdots \\
113 & 89 & 89 & 109 & 106 & 126 & 114 & 150 & 164 & 145 & \cdots \\
120 & 121 & 123 & 87 & 85 & 70 & 119 & 64 & 79 & 127 & \cdots \\
145 & 141 & 143 & 134 & 111 & 124 & 117 & 113 & 64 & 112 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots &
\end{bmatrix}$$

Abbildung 2.5
Übergang von einer kontinuierlichen Lichtverteilung $F(x,y)$ zum diskreten Digitalbild $I(u,v)$ (links), zugehöriger Bildausschnitt (unten).

die *Breite* M (Anzahl der Spalten) und die *Höhe* N (Anzahl der Zeilen) der zugehörigen Bildmatrix I.

Die *Auflösung* (*resolution*) eines Bilds spezifiziert seine räumliche Ausdehnung in der realen Welt und wird in der Anzahl der Bildelemente pro Längeneinheit angegeben, z. B. in „dots per inch“ (dpi) oder „lines per inch“ (lpi) bei Druckvorlagen oder etwa in Pixel pro Kilometer bei Satellitenfotos. Meistens geht man davon aus, dass die Auflösung eines Bilds in horizontaler und vertikaler Richtung identisch ist, die Bildelemente also quadratisch sind. Das ist aber nicht notwendigerweise so, z. B. weisen die meisten Videokameras nichtquadratische Bildelemente auf.

Die räumliche Auflösung eines Bilds ist in vielen Bildverarbeitungsschritten unwesentlich, solange es nicht um geometrische Operationen geht. Wenn aber etwa ein Bild gedreht werden muss, Distanzen zu messen sind oder ein präziser Kreis darin zu zeichnen ist, dann sind genaue Informationen über die Auflösung wichtig. Die meisten professionellen Bildformate und Softwaresysteme berücksichtigen daher diese Angaben sehr genau.

2.2.5 Bildkoordinaten

Um zu wissen, welche Bildposition zu welchem Bildelement gehört, benötigen wir ein Koordinatensystem. Entgegen der in der Mathematik üblichen Konvention ist das in der Bildverarbeitung übliche Koordinatensystem in der vertikalen Richtung umgedreht, die y-Koordinate läuft also von oben nach unten und der Koordinatenursprung liegt links oben (Abb. 2.6). Obwohl dieses System keinerlei praktische oder theoretische Vorteile hat (im Gegenteil, bei geometrischen Aufgaben häufig zu Verwirrung führt), wird es mit wenigen Ausnahmen in praktisch allen Softwaresystemen verwendet. Es dürfte ein Erbe der Fernsehtechnik sein, in der Bildzeilen traditionell entlang der Abtastrichtung des Elektronenstrahls, also von oben nach unten nummeriert werden. Aus praktischen Gründen starten wir die Nummerierung von Spalten und Zeilen bei 0, da auch Java-Arrays mit dem Index 0 beginnen.

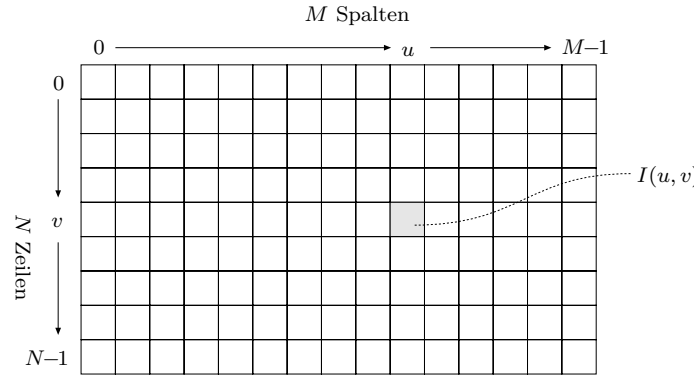


Abbildung 2.6
Bildkoordinaten. In der digitalen Bildverarbeitung wird traditionell ein Koordinatensystem verwendet, dessen Ursprung ($u = 0, v = 0$) links oben liegt. Die Koordinaten u, v bezeichnen die *Spalten* bzw. die *Zeilen* des Bilds. Für ein Bild der Größe $M \times N$ ist der maximale Spaltenindex $u_{\max} = M - 1$, der maximale Zeilenindex $v_{\max} = N - 1$.

2.2.6 Pixelwerte

Die Information innerhalb eines Bildelements ist von seinem Typ abhängig. Pixelwerte sind praktisch immer binäre Wörter der Länge k, sodass ein Pixel grundsätzlich 2^k unterschiedliche Werte annehmen kann. k wird auch häufig als die Bit-Tiefe (oder schlicht „Tiefe“) eines Bilds bezeichnet. Wie genau die einzelnen Pixelwerte in zugehörige Bitmuster kodiert sind, ist vor allem abhängig vom Bildtyp wie Binärbild, Grauwertbild, RGB-Farbbild und speziellen Bildtypen, die im Folgenden kurz zusammengefasst sind (Tabelle 2.1):

Grauwertbilder (Intensitätsbilder)

Die Bilddaten von Grauwertbildern bestehen aus nur einem Kanal, der die Intensität, Helligkeit oder Dichte des Bilds beschreibt. Da in den meisten Fällen nur positive Werte sinnvoll sind (schließlich entspricht Intensität der Lichtenergie, die nicht negativ sein kann), werden üblicherweise positive ganze Zahlen im Bereich $[0 \ldots 2^k - 1]$ zur Darstellung benutzt. Ein typisches Grauwertbild verwendet z. B. $k = 8$ Bits (1 Byte) pro Pixel und deckt damit die Intensitätswerte $[0 \ldots 255]$ ab, wobei der Wert 0 der minimalen Helligkeit (schwarz) und 255 der maximalen Helligkeit (weiß) entspricht.

Bei vielen professionellen Anwendungen für Fotografie und Druck, sowie in der Medizin und Astronomie reicht der mit 8 Bits/Pixel verfügbare Wertebereich allerdings nicht aus. Bildtiefen von 12, 14 und sogar 16 Bits sind daher nicht ungewöhnlich.

Binärbilder

Binärbilder sind spezielle Intensitätsbilder, die nur zwei Pixelwerte vorsehen – schwarz und weiß –, die mit einem einzigen Bit (0/1) pro Pixel kodiert werden. Binärbilder werden häufig verwendet zur Darstellung von Strichgrafiken, zur Archivierung von Dokumenten, für die Kodierung von Fax-Dokumenten, und natürlich im Druck.