Eine Kommandosprache zur Bildverarbeitung

H. P. Meinzer und U. Engelmann

Institut für Dokumentation, Information und Statistik, Deutsches Krebsforschungszentrum, Heidelberg

Stichworte: *Dialog, Befehlssprache, Bildverarbeitung*

Zusammenfassung: *Die Sprache PIC ist eine interpretative Kommandosprache. Sie beinhaltet wichtige Funktionen, die in der Bildverarbeitung notwendig sind. Die Sprache enthält z.Z. über 90 Befehle zur Bildspeicherung und -verwaltung, geometrische Bildmanipulationen, Histogramm – Modifikationen, Gradientenoperatoren, Glättungsalgorithmen, Fourierfunktionen, arithmetische und Boolesche Operationen und Funktionen zur Bilddarstellung.*

PIC eignet sich zur Ausbildung von Anfängern auf dem Gebiet der Bildverarbeitung genauso wie für Entwickler und Benutzer von Systemen. Komplexe Problemlösungen und Anwendungssysteme können dadurch realisiert werden, daß die speziellen problemabhängigen Algorithmen zu den allgemein verfügbaren hinzugefügt werden. Da die Routinen in PIC allgemein und grundlegend sind, spielt es keine Rolle aus welchen Problemkreisen die Bilder stammen.

Key-words: *Dialog, Command Language, Image Processing*

Abstract: *PIC is an interpretative command language which can be applied to digitized images. It includes the major operations in the field of image processing. Today it is a collection of more than 90 commands for dataset management, storage, simple image manipulation, histogram modification, gradient operators, smoothing algorithms, fourier functions, arithmetic and Boolean operations and display routines.*

It is suited for the training of beginners in the field of image processing, as well as for designers and users of systems in this area. If some problem-dependent specialized programs are additionally included, sophisticated application programs can be constructed. The routines collected in the language PIC are all quite general. Therefore, there is no restriction to the origin of the images.

1 Einleitung

Gewöhnlich hat jeder Systementwickler in der Bildverarbeitung heute immer noch seine eigene Sammlung von Unterprogrammen grundlegender und spezieller Funktionen. Mit diesen wird eine spezielle Lösung eines speziellen Problems geschneidert. Dieser Ansatz ist teuer und personalintensiv. Eine bessere Lösung ist es, eine gemeinsame Unterprogrammsammlung für mehr als einen Programmentwickler zu schaffen, um Doppelarbeiten zu vermeiden. Auch bei der Verwendung allgemeiner Unterprogramme muß für jedes Problem ein eigenes Hauptprogramm geschrieben werden.

Nützlicher und billiger ist die Verwendung eines dialogfähigen Systems, in das alle gängigen Bildverarbeitungsprozesse eingebettet sind. Ein solches System kann nicht nur vom Entwickler sondern auch vom Anwender benutzt werden, falls die Befehle einfach und plausibel sind. Für den Entwickler ist darauf zu achten, daß Erweiterungen einfach in das bestehende Paket integriert und dadurch spezielle Lösungen mit kleinem Aufwand verfügbar gemacht werden können.

Überlegungen zu Bildverarbeitungssprachen gab es schon vor ca. 10 Jahren [9, 10]. Es sind viele verschiedene Systeme entwickelt worden, allein etwa 50 in den USA [24]. Es befinden sich darunter Erweiterungen klassischer Programmiersprachen, Neuschöpfungen zu kompilierender Sprachen und mehr oder weniger komfortable Interpreter, also Kommandosprachen, mit zum Teil schrecklicher Syntax [24]. Andere sind einfach Unterprogrammbibliotheken in FORTRAN, wie z.B. SPIDER [30].

2 Anforderungen an Dialogsysteme

Pretschner [25] hat folgende Anforderungen an Bildverarbeitungssysteme in der Nuklearmedizin gestellt, die als allgemeingültig in der Bildverarbeitung betrachtet werden können:

- Interaktivität,
- Dialogfähigkeit,
- leichte Erlernbarkeit,

0013-5704/83/10 0434–06 $ 02.00/0

© Friedr. Vieweg & Sohn Verlagsgesellschaft mbH

- Flexibilität,
- Transparenz und
- Anwenderfreundlichkeit.

Es wird eine Dialogsprache gefordert, die direkt auf Eingaben des Benutzers reagiert (conversational computing) und Eigenschaften aufweist, die mit höheren Programmiersprachen vergleichbar sind [25].

Resultate solcher Überlegungen sind kommandoverarbeitende Systeme. Diese basieren auf Konzepten, die bereits seit längerem von den Kommandosprachen in Betriebssystemen her bekannt sind [19, 20, 21, 31]. Mit Kommandosprachen werden nicht primär Algorithmen formuliert, sie sind vielmehr als Werkzeug anzusehen, um sekundär andere Programme oder Funktionen zu aktivieren – ursprünglich Betriebssystemfunktionen [12].

3 Der Interpreter PIC

Kennzeichen des Interpreters PIC sind:

- *Klar verständliche Kommandosprache:* Die Kommandos sind sehr einfach und verständlich gehalten. Die Funktion eines Kommandos erklärt sich schon durch den Namen bzw. die Schlüsselworte der Parameter.
- *leichte Erlernbarkeit* durch eine klar verständliche Kommandosprache und einfache Kommandos.
- *Transparenz:* Das System kennt nur zwei Grundzustände, entweder READY oder WORKING. Es gibt nur eine Eingabeebene.
- *Robustheit gegenüber Fehlern* wird durch die Verarbeitung des Inputstrings durch einen LL(1)-Parser und Lexical Analyser sowie Prüfungen der semantischen Integritätsbedingungen erreicht. Der Übergang in undefinierte Zustände kann damit weitgehend verhindert werden.
- *Abkürzungen und Synonyme:* Vom System wird eine große Anzahl von Abkürzungen und Synonymen angeboten. Diese können vom Benutzer erweitert oder geändert werden.
- *Defaults:* Um möglichst kurze Kommandos zu erreichen, wurden für die meisten Systemparameter Defaults vorgesehen. Diese können per Kommando geändert werden.
- *Verketten von Kommandos:* Kommandos werden zeilenweise interpretiert, also verarbeitet. Mehrere Kommandos können in einer Zeile mit Semikolon oder dem Wort "AND" verknüpft werden.
- *Benutzerhilfen:* Es steht ein HELP-Kommando zur Verfügung, das alle Kommandos und deren Syntax anzeigt. Über einen sog. MENUE-MODUS wird dem Benutzer geholfen, wenn er ein Kommando falsch eingegeben hat. Die fehlerhafte Stelle im Inputstring wird markiert und die möglichen richtigen Alternativen ausgegeben. Über ein STATUS-Kommando kann sich der Benutzer über den Systemzustand orientieren. Mit einem RESET-Kommando wird der Anfangszustand wieder hergestellt.
- *Semantische Integritätsprüfungen:* In einem interaktiven Kommandosystem hat der Benutzer die Möglichkeit, Kommandos wahllos zu kombinieren. Deshalb wird automatisch die semantische Integrität, d.h. hier insbesondere die Präzedenz der Kommandos geprüft. Ist diese verletzt, so wird das aktuelle Kommando und alle folgenden gestoppt und der Benutzer wird mit einer klaren Meldung auf die Fehlerursache aufmerksam gemacht.
- *Fehlermeldungen* werden im Sinne von Shneiderman [29] realisiert. Das bedeutet, daß dem Benutzer ganz explizit in Worten gesagt wird, was er falsch gemacht hat und was er machen muß, damit der Fehler vermieden wird.
- *Prozeduren:* Häufig vorkommende Kommandofolgen können sehr einfach in Prozeduren zusammengefaßt werden. Die Übergabe von Parametern und die Benutzung von Kontrollstrukturen ist möglich.
- *Conversational/Batch:* Der Interpreter kann interaktiv am Bildschirm benutzt werden. So kann der Benutzer sofort die Ergebnisse überprüfen bzw. wird sofort auf Fehler aufmerksam gemacht. Komplexere Operationen an großen Bildern dauern sehr lange. Deshalb wurde die Möglichkeit vorgesehen, Kommandos in einer Datei zusammenzufassen und im Rechnerhintergrund abarbeiten zu lassen.
- *Erweiterbarkeit:* PIC bietet nur bildunabhängige Operationen. Deshalb haben Benutzer nach kurzer Zeit Wünsche, die über das Funktionsspektrum von PIC hinaus gehen. Der Interpreter kann mit dem Kommando "CALL Name" erweitert werden. Hiermit wird ein vom Benutzer geschriebenes FORTRAN-Unterprogramm mit dem Entry „Name" dynamisch geladen und ausgeführt. Die Subroutine muß eine Standardparameterliste besitzen, über die der Variablen- und Datenaustausch mit PIC realisiert wird. Über COMMON-Bereiche können die benutzereigenen Unterprogramme miteinander Daten austauschen.

Die interpretative Sprache PIC [7] dient der Bildbearbeitung und besteht aus über 90 Befehlen, die sich in neun Gruppen zusammenfassen lassen:

- Dateiverwaltungsfunktionen,
- Darstellungsfunktionen,
- geometrische Bildmanipulationen,
- Histogramm-Modifikationen,
- Gradientenoperatoren,
- Glättungsalgorithmen,
- Fourier-Operatoren,
- arithmetische und Boolesche Operatoren,
- Hilfsfunktionen.

Aus jeder Gruppe werden einige typische Befehle näher beschrieben. Ausdrücke in Klammern können in der Regel auch weggelassen werden: das Zeichen '/' entspricht einem logischen ODER. Zur Gruppe der Dateiverwaltungsfunktionen gehören:

LOAD Name,
STORE Name und
ERASE Name.

Alle Bilder (Bildmatrizen) werden auf Magnetplatte gespeichert. Die Größe eines Bildes kann beliebig gewählt werden bis zu maximal 900 x 900 Bildpunkten (Pixels). Wenn ein Bild von der Platte in den Arbeitsbereich von PIC geladen wird, werden auch die Bildgröße und der Bildname mit transferiert. Auf diese Weise spricht der Benutzer ein Bild nur durch seinen Namen an und muß sich nicht um die aktuelle Größe kümmern. Die folgenden Befehle arbeiten auf einem Bild im Arbeitsbereich, mit ganz wenigen Ausnahmen, wo Operationen auf zwei gleichgroßen Bildern ausgeführt werden. Zur Darstellung von Bildern und Ergebnissen auf einem dafür geeigneten Gerät dienen u.a. die folgenden Befehle:

PRINT (REV) (ON WHITE PAPER)
PLOT HISTO
PLOT TOPO
PLOT 3D (PHI, THETA)
PLOT ISO IVALUE
PLOT CONTOUR
PLOT GRAY
DISPLAY

PRINT erlaubt eine Ausgabe minderer Qualität mit maximal 14 Graustufen auf einem normalen alphanumerischen Drucker. Die sechs PLOT Funktionen brauchen alle ein graphisches Terminal TEKTRONIX 4014/4015 für die Darstellung von Kurven und Matrixbildern. Die nützlichsten Kommandos sind hier TOPO und 3D, welche ein pseudo-dreidimensionales Topogramm eines Bildes ausgeben. Beide enthalten einen Hidden-line-Algorithmus und produzieren verständliche Darstellungen. ISO und CONTOUR zeichnen Isophoten und Höhenlinien, PLOT GRAY erzeugt ein bescheidenes Grauwertbild. DISPLAY stellt ein Bild auf einem Rastersichtgerät TEKTRONIX 4112 dar.

Die Bilder 1 und 2 zeigen Ausschnitte aus dem Colon einer Ratte. Sie entstanden im Rahmen der Bemühung, die Mitosen in der Krypte automatisch zu erkennen.

Vier geometrische Bildmanipulationen sind:

WINDOW IX, IY, IDELTAX, IDELTAY
SCALE N (/M)
ROTATE (90/180/270)
REFLECT (HOR/VER/DIAG1/DIAG2).

Mit diesen Befehlen werden die Bildgröße und die Orientierung eines Bildes beeinflußt. Histogramm – Modifikationen sind möglich mit:

NORM GRAYSCALE
FLAT
SELECT N-M (SET IVALUE).

Das erste Kommando transformiert die Grauwerte eines Bildes so, daß die volle Bandbreite von 0 bis 255 ausgenutzt wird. FLAT arbeitet ähnlich, strebt jedoch zusätzlich eine gleichmäßige Verteilung der Grauwerte an. Das letzte Kommando selektiert einen Grauwertebereich von besonderem Interesse und setzt alle anderen Werte auf Null oder einen vorgegebenen Wert.

Beispielsweise kann ein Bild, das zwei Zellen nach der Mitose beinhaltet, durch LOAD CELLS geladen werden.

Bild 1 Das Topogramm einer Krypte aus dem Colon einer Ratte

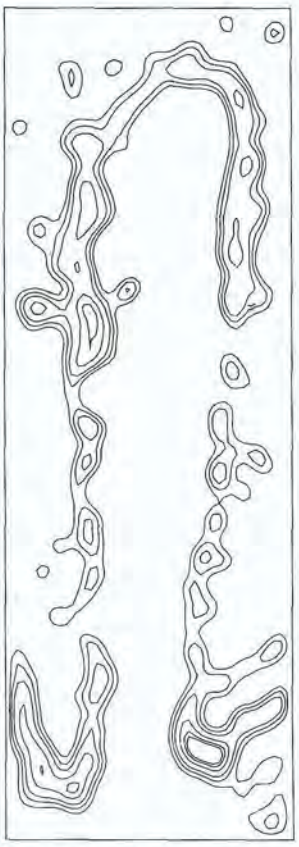

Bild 2 Die Konturlinien des gleichen Bildes

Dann wird mit PLOT TOPO das entsprechende Topogramm gezeichnet, mit SELECT 50–150 ein Grauwertebereich ausgewählt (alle verbleibenden Bildpunkte werden auf Null gesetzt) und das entstandene Bild wieder mit PLOT TOPO dargestellt (vgl. Bilder 3 bis 5).

PIC enthält acht Gradientenoperatoren, darunter:

SOBEL
ROBERTS
LAPLACE.

Alle diese Funktionen arbeiten auf einem 3x3 Pixel großen Teilbild und berücksichtigen verschiedene Gradienten. Für die Glättung von Bildern gibt es Befehle wie:

MEDIAN N
SMOOTH.

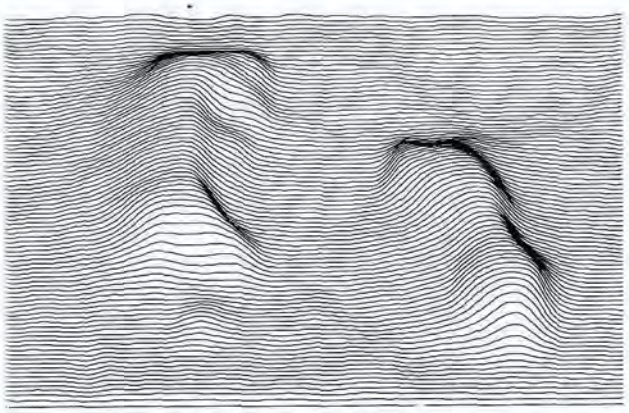

Bild 3 Topogramm von zwei Zellen nach der Mitose

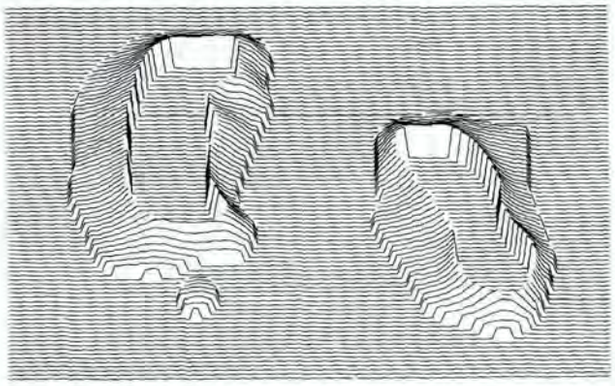

Bild 4 wie Bild 3, nur alle Pixel außerhalb 50–150 auf Null gesetzt.

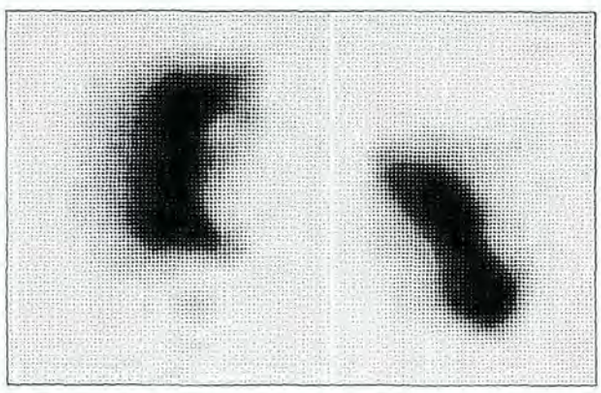

Bild 5 wie Bild 3, aber mit PLOT GRAY gezeichnet

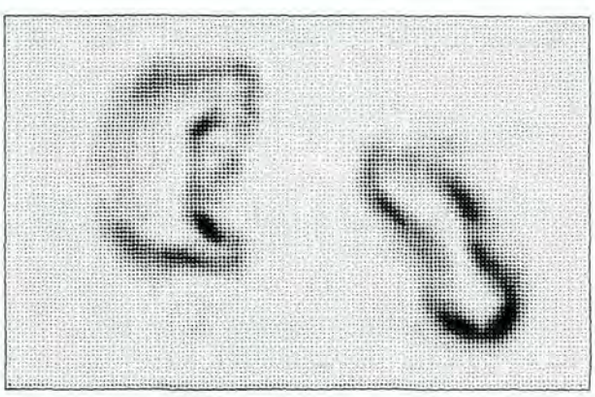

Bild 6 wie Bild 5 nach dem Gradientenoperator ROBERTS

MEDIAN ist ein Medianfilter auf dem Teilbild der Größe N*N und SMOOTH ist ein gewichteter Mittelwertfilter auf einem 3*3 Fenster. Es gibt eine Reihe von Fourier-Operatoren:

FOURIER (INVERSE)
CONV
CORR
POWER.

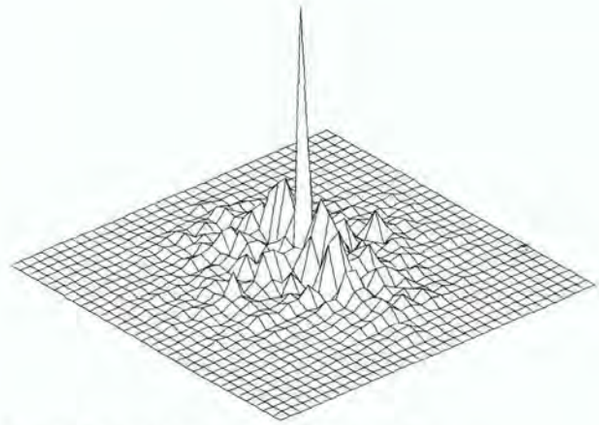

Bild 7 3D-Darstellung eines Powerspektrums

FOURIER berechnet die schnelle Fourier-Transformation (FFT) eines gegebenen Bildes und die Rücktransformation. In der Fourierebene stehen daneben Funktionen wie Hoch- und Tiefpaßfilter zur Verfügung. CONV und CORR berechnen die Faltung und Korrelation zweier Bilder im Frequenzbereich. POWER berechnet das Powerspektrum eines Bildes.

Grundlegende arithmetische Operationen sind

ADD (CONST)
SUB (CONST)
MULT (CONST)
DIV (CONST).

Wenn der konstante Faktor weggelassen wird, so wird die Operation nicht auf einem, sondern auf zwei Bildern durchgeführt, die beide im Arbeitsbereich gespeichert sind. Die Booleschen Operatoren AND, OR und XOR arbeiten punktweise ausschließlich auf zwei Bildern. Zuletzt gibt es einige Hilfsfunktionen wie:

STATUS
HELP
END.

STATUS gibt am Terminal alle verfügbaren Information über das geladene Bild aus und zeigt die Werte einiger systemimmanenter Variablen. HELP und END sind selbsterklärend.

Gewöhnlich werden die Befehle zeilenweise einzeln eingegeben und im Dialog sofort ausgeführt. Zusätzlich ist das Verbinden von Befehlen durch Semikolon (;) oder ein AND erlaubt, z.B.:

LOAD IMAGE; SMOOTH AND DISPLAY.

Obwohl der Interpreter PIC bereits eine ganze Menge nützlicher, allgemein einsetzbarer Operationen enthält, und obschon PIC erweitert wird, sooft sich ein neuer, brauchbarer Operator findet, wird für die Lösung einer speziellen bildverarbeitenden Aufgabe immer ein spezieller Algorithmus entwickelt werden müssen. Spezialisierte selbstentwickelte Programme beliebiger Anzahl können über den Befehl

CALL subroutine

in PIC integriert werden. Die Unterprogramme haben eine vordefinierte Parameterliste und werden zur Lauf-

zeit auf Anforderung zusätzlich geladen. Damit wird PIC beliebig individuell erweiterbar. Der Befehl CALL benutzt eine besondere Fähigkeit unseres Betriebssystems TSS [31]. Solche Erweiterungen setzen eine solide Kenntnis von TSS, FORTRAN und der internen Struktur von PIC voraus. Durch die hier erläuterte Möglichkeit kann der Anwender sich auf seine ganz speziellen Probleme konzentrieren und hat als Vorleistung nicht nur sehr viele gängige Programme, sondern auch eine benutzerfreundliche interpretative Sprache zur Verfügung.

PIC kann auf verschiedene Weisen benutzt werden. In der Regel wird man es im Dialog an einem graphischen Terminal benutzen, wo das Ergebnis einer Operation sofort optisch verifizierbar ist. Für zeitaufwendige oder Routineanwendungen können Befehle auch in einer Datei zusammengefaßt und im Rechnerhintergrund (Batch) abgearbeitet werden.

Das System erlaubt auch, mit Abkürzungen der meisten Befehlsworte zu arbeiten. Dies ist nützlich für Anwender, die die Sprache gut beherrschen. Zusätzlich können Synonyme definiert werden. Auch das Zusammenfassen von Befehlsfolgen zu einem neuen Kommando (Prozedur) ist möglich. Abkürzungen, Synonyme und Prozeduren können lokal für einen oder global für alle Benutzer vereinbart werden. Hierfür wird der im Deutschen Krebsforschungszentrum entwickelte Inputprozessor verwendet [28].

4 Implementierung

PIC läuft auf einer IBM 3032 mit dem Betriebssystem TSS (Time Sharing System). Es benötigt mindestens ein graphisches Terminal, z.B. eine TEKTRONIX 4014 oder 4112. Die Software ist in FORTRAN IV geschrieben. Die Analyse der Eingabezeilen wird von einem Parser durchgeführt, der mit den Generatoren PAULA [3] erzeugt wurde. Die Bilddarstellung erfolgt mit den graphischen Softwarepaketen TCS und AG-II [14]. Ein Teil der Routinen wurde aus der Literatur und von anderen Personen und Instituten übernommen. Aufgrund des Betriebssystems TSS und der Generatoren PAULA ist das Hauptprogramm nicht auf andere Maschinen transportierbar. Die Unterprogrammbibliothek kann jedoch mit kleinem Aufwand auf anderen Rechnern bzw. unter anderen Betriebssystemen implementiert werden.

5 Schlußbetrachtung

Die Erfahrung zeigt, daß ein Anwender mit PIC leicht umzugehen lernt. Ein Anfänger kann bildverarbeitende Routinen lernen und anwenden.

Parallel zum didaktischen Wert kommt die einfache Erweiterbarkeit, die es ermöglicht, ohne großen Aufwand anwendungsorientierte Systeme zu entwickeln. Gewöhnlich reicht die Erweiterung um wenige spezielle Routinen zur Lösung komplexer Probleme aus [8]. Der Entwickler kann sich um die speziellen Anforderungen kümmern und braucht sich nicht mit den Standardmethoden zu beschäftigen, diese sind in PIC bereits enthalten.

PIC wird erweitert, wann immer wir Algorithmen finden, die für mehr als einen Anwender von Nutzen sein können. Hierbei ist die Unterstützung durch die Generatoren PAULA [3] nicht zu unterschätzen.

Danksagung

Für Ideen und Programme haben wir unseren Dank auszusprechen: J. Dengler, M. Jaksch und G. Zinser steuerten Unterprogramme bei. Dr. Komitowski war bei der Digitalisierung von Bildern behilflich. Spezieller Dank gebührt Prof. Dr. G. Zaijcek für kreative Anregungen und unseren Kollegen im Institut, die bei der Generierung des Systems behilflich waren.

Literatur

[1] *Aho, A. V., Ullman, J. D.:* The Theory of Parsing, Translation and Compiling. Band 1 und 2. Englewood Cliffs, N. J.: Prentice Hall 1972

[2] *Aho, A. V., Ullman, J. D.:* Principles of Compiler Design. Reading: Addison-Wesley Publishing Company 1977

[3] *Becker, N., Osterburg, G., Schadewaldt, K.:* PAULA – Generator für LL(1)-Parser und lexikalische Analyseprogramme. Heidelberg: DKFZ 1977. (ZDV Technical Report Nr. 10)

[4] *Becker, N., Osterburg, G.:* Generierung interpretativer Systeme. Heidelberg: DKFZ 1978. (ZDV Technical Report Nr. 11)

[5] *Castleman, K. R.:* Digital Image Processing. Englewood Cliffs, N. J.: Prentice Hall 1979

[6] *Enderle, G., Giese, J., Krause, M., Meinzer, H. P.:* The AGF-Plotfile – Towards a Standardization for Storage and Transportation of Graphic Information. In: SIGGRAPH-ACM Comp. Graph. Vol. 12, No. 4 (1978) 92–113

[7] *Engelmann, U., Meinzer, H. P.:* Bildbearbeitungssystem PIC – Kommandobeschreibung. Heidelberg: DKFZ 1982. (ZDV Technical Report Nr. 25)

[8] *Engelmann, U., Meinzer, H. P.:* Analysis of Electrophoresis Gels by an Image Processing System. In: van Bemmel, J. H., Ball, M., Wigertz, O. (Eds): MEDINFO 83. Amsterdam: North Holland 1983

[9] *Fink, B.:* Untersuchungen über eine Programmiersprache für die Verarbeitung matrixförmiger und hierarchischer Datenstrukturen. FIM/FGAN-Bericht Nr. 3. Karlsruhe: Forschungsinstitut für Informationsverarbeitung und Mustererkennung 1973

[10] *Fink, B.:* Überlegungen zu einer Programmiersprache für die automatische Bildverarbeitung. FIM/FGAN-Bericht Nr. 5. Karlsruhe: Forschungsinstitut für Informationsverarbeitung und Mustererkennung 1973

[11] *Gonzalez, R. C., Wintz, P.:* Digital Image Processing. Reading: Addison-Wesley 1977

[12] *Gram, C., Hertweck, F.:* Command Languages: Design Considerations and Basic Concepts. In: Unger, G. (Ed.): Command Languages. Amsterdam: North-Holland 1975, pp. 43–69

[13] *Hahne, H.:* DKFGRAF – Graphisches Programmpaket. Heidelberg: DKFZ 1971. (ZDV Technical Report Nr. 1)

[14] *Hahne, H.:* TEKTRONIX: TCS und AG-II. Heidelberg: DKFZ 1976. (ZDV Technical Report Nr. 8).

[15] *Hall, E. L.:* Computer Image Processing and Recognition. New York: Academic Press 1979

[16] *Herschel, R.:* Einführung in die Theorie der Automation, Sprachen und Algorithmen. München: Oldenbourg 1974

[17] *Hofele, G., Schlifkowitz, U.:* Bewertung von Standardmethoden der Bildvorverarbeitung. Forschungsauftrag T/R 624/90024/92661 (Abschlußbericht). Karlsruhe: Forschungsinstitut für Informationsverarbeitung und Mustererkennung 1981

[18] *Kazmierczak, H.:* Erfassung und maschinelle Verarbeitung von Bilddaten. Berlin: Springer 1980

[19] *Meinzer, H. P.:* An Interpreter for Matrix Graphics. In: O'Moore, R. R., Barber, B., Reichertz, P. L., Roger, F. (Eds): Medical Informatics Europe 82. Berlin: Springer 1982, pp. 256–258.

[20] *Meinzer, H. P.:* Dialog zwischen Mensch und Maschine. In: C. O. Köhler, K. Böhm und R. Thome (Hrsg.): Aktuelle Methoden der Information in der Medizin. Festschrift zum 65. Geburtstag von Prof. Dr. med. Gustav Wagner. Landsberg: Ecomed Verlag 1983, S. 169–179

[21] *Meinzer, H. P., Engelmann, U.:* A Language for the Interactive Manipulation of Digitalized Images. In: Lang, M. (Ed.): 6th International Conference on Pattern Recognition, Munich, 1982. Silver Spring, MD: IEEE, Computer Society Press 1982, pp. 68–70

[22] *Newman, W. M., Sproul, R. F.:* Principles of Interactive Computer Graphics. New York: McGraw Hill 1973

[23] *Pratt, W. K.:* Digital Image Processing. New York: Wiley 1978.

[24] *Preston, K.:* Image Manipulative Languages: A Preliminary Survey. In: Gelsema, E. S., Kanal, L. N. (Eds): Pattern Recognition in Practice, Amsterdam: North-Holland 1980, pp. 5–20.

[25] *Pretschner, D. P., Pfeiffer, G.:* Erzeugung einer Kommandosprache für Nuklearmedizinische Signal- und Bildverarbeitung aus einem allgemeinen Dialogsystem. In: Poeppl, S. J., Pretschner, D. P. (Hrsg.): Systeme und Signalverarbeitung in der Nuklearmedizin. Berlin: Springer 1981, S. 187–204

[26] *Rosenfeld, A.:* Picture Processing by Computer. New York: Academic Press 1969

[27] *Rosenfeld, A., Kak, A. C.:* Digital Picture Processing. New York: Academic Press 1976

[28] *Schadewaldt, K., Merx, R., Kynast, W.:* Der Inputprozessor. Heidelberg: DKFZ 1980. (ZDV Technical Report Nr. 18)

[29] *Shneiderman, B.:* Designing Computer System Messages. In: Commun. ACM Vol. 25 (1982), pp. 610–611

[30] *Tamura, H., Tomita, F., Sakane, S., Yokaya, N., Sakaue, K., Kaneko, M.:* A Transportable Image Processing Software Package: SPIDER. In: Lang, M. (Ed.): Proceedings of the 6th International Conference on Pattern Recognition, Munich, 1982 Silver Spring, MD: IEEE Computer Society Press 1982, pp. 75–78

[31] TSS COMMAND SYSTEM USER'S GUIDE. White plains: IBM 1976. (IBM Form No. GC28–2001–9)

[32] *Wirth, N.:* Compilerbau – Eine Einführung. Stuttgart: Teubner 1977.

Eingegangen am 11.07.1983

Virtuos auf dem Personal Computer

Entnommen mit freundlicher Genehmigung aus *Wukasch:* Computer sind auch Menschen. Verlagsgesellschaft Rudolf Müller, Köln 1983