Forschungszentrum Karlsruhe
Technik und Umwelt

Wissenschaftliche Berichte
FZKA 5523

IODA – Ein schnelles, vollautomatisches und flexibles System zur Analyse von Ionenspuren auf Filmen

H. Guth, A. Hellmann
Institut für Angewandte Informatik

Februar 1995

Forschungszentrum Karlsruhe

Technik und Umwelt

Wissenschaftliche Berichte

FZKA 5523

IODA - Ein schnelles, vollautomatisches und flexibles System zur Analyse von Ionenspuren auf Filmen

Helmut Guth, Andreas Hellmann

Institut für Angewandte Informatik

Forschungszentrum Karlsruhe GmbH, Karlsruhe

1995

Als Manuskript gedruckt

Forschungszentrum Karlsruhe GmbH
Postfach 3640, 76021 Karlsruhe

ISSN 0947-8620

IODA - Ein schnelles, vollautomatisches und flexibles System zur Analyse von Ionenspuren auf Filmen

Zusammenfassung

Das System IODA (**Io**n **D**ensity **A**nalysis) wird zur Auswertung der bei Experimenten am Hochspannungspulsgenerator KALIF (**K**arlsruhe **Li**ght **I**on **F**acility) anfallenden ionenempfindlichen Filmen eingesetzt. Es besteht aus einem Prozeßrechner, der ein Mikroskop mit Videointerface und ein Multiprozessor-Rechnersystem steuert, und der Auswerte-Software.

Die Segmentierung der Ionenspuren erfolgt mit Methoden der digitalen Bildverarbeitung und Mustererkennung automatisch. Nach der Definition des Auswerte-Bereichs und Festlegung eines Verfahrens wird die Filmfläche unter dem Mikroskop abgerastert und die Anzahl der Einschläge pro Bild ausgezählt. Je nach Aussehen der Ionenspuren auf dem Film können verschiedene Verfahren gewählt werden. Das Ergebnis, das in einer Matrix abgelegt wird, kann bei gleichmäßigem Rasterabstand als Abbildung der Ionen-Einschlagdichte betrachtet werden.

Durch den Einsatz eines Prozeßrechners und Verteilung der zeitaufwendigen Mustererkennung auf ein Multiprozessor-System, können auch bei hoher Auflösung noch akzeptable Auswertezeiten erreicht werden. Engpaß des Systems ist der Datentransport und die Geschwindigkeit des Mikroskop-Kreuztisches.

Bei der Software wurde Wert auf die einfache Handhabbarkeit gelegt. Durch ein logisch gegliedertes Menue-System mit online-Hilfe konnte dies auch auf einem einfachen Terminal erreicht werden. Das Einrichten des Auswerte-Bereichs, die Auswahl und der Test des Verfahrens erfolgt interaktiv mit Rechnerunterstützung.

Der vorliegende Bericht soll hauptsächlich als Anwenderhandbuch dienen und den Benutzer bei der Bedienung des Systems unterstützen.

IODA - A Fast, Automated and Flexible System for Ion Track Analysis on Film Detectors

Abstract

The IODA System (**Io**n **D**ensity **A**nalysis) is used to analyse detector films, resulting from experiments at the pulse power generator KALIF (**Ka**rlsruhe **L**ight **I**on **F**acility). The system consists of evaluation software and a microcomputer, which controls a microscope, a video interface, and a multiprocessor subsystem.

The segmentation of ion tracks is done automatically by means of digital image processing and pattern recognition. After defining an evaluation range and selecting a suitable analysis method, the film is scanned by the microscope for counting the impacts of the underlying image. According to the appearance of the ion tracks on the film, different methods can be selected. The evaluation results representing the ion density are stored in a matrix. The time needed for an evaluation at a high resolution can be shortened by shipping time consuming pattern recognition calculations to the multiprocessor subsystem. The bottlenecks of the system are the data transfer and the speed of the microscope stage.

Simple handling of the system even on alphanumeric terminals had been an important design issue. This was implemented by a logically structured menue system including online help features.

This report can be used as a manual to support the user with system operation.

Inhaltsverzeichnis

1. Einführung 1
2. Problembeschreibung 2
3. Lösungsansatz 5
4. Realisation der Auswertung 11
    4.1 Vorgehensweise 11
    4.2 Rechner-Konzept 12
5. Systemhandhabung (Manual) 15
    5.1 Aufruf und Bedienung 15
    5.2 Hauptprogramm IODA 16
        5.2.1 Menue PARMS 21
        5.2.2 Menue TISCH 25
        5.2.3 Menue KAMERA 28
        5.2.4 Menue METHODE 30
    5.3 Programm IODA_IF 35
    5.4 Programm IODA_ED 41
        5.4.1 Menue BEREICH 44
        5.4.2 Menue SEGMENT 46
        5.4.3 Menue LAGE 49
6. Beispiel-Auswertung 51
    6.1 Vorbereitung 51
    6.2 Festlegung der auszuwertenden Regionen 52
    6.3 Bestimmung der Auswerte-Methode 55
    6.4 Definition der übrigen Parameter 58
    6.5 Auswertung und Ergebnis 60
7. Bewertung und Ausblick 63
8. Anhang - Implementierungsdetails 65
    8.1 Dateiformate 65
    8.2 Transputer-Farm 68
    8.3 Grundlagen der Menue-Steuerung 69
9. Literaturverzeichnis 74

1. Einführung

Am Institut für Neutronenphysik und Reaktortechnik (INR) des Kernforschungszentrums Karlsruhe werden im Rahmen der Arbeiten zur 'Physik intensiver Ionenstrahlen und gepulster dichter Plasmen' Experimente am Hochspannungspulsgenerator KALIF (**Ka**rlsruhe **L**ight **I**on **F**acility) durchgeführt. Dabei werden intensive Strahlen leichter Ionen benutzt, um in Materie Zustände extrem hoher Energie- und Strahlungsdichte zu erzeugen und Bedingungen herzustellen, unter denen z.B. Kernfusion stattfinden könnte [1]. Die Realisierung der Inertial-Fusion könnte eine Anwendung dieser Strahlen in ferner Zukunft sein. Daneben lassen sich mit diesen intensiven Strahlen auch neue Gebiete der Grundlagenforschung erschließen. Als Beispiel sei die Bestimmung der Zustandsgleichung von Materie in Druck- ($>10^6$Bar) und Temperaturbereichen ($>10^5$ °K) erwähnt, die bis dahin im Labor nicht zugänglich waren.

Um die erforderlichen Teilchendichten zu erreichen, werden am INR Hochstrom-Ionenquellen, sogenannte Dioden, entwickelt. Diese erzeugen intensive Protonenstrahlen, die im Fall des KALIF fokussiert Leistungsdichten von 0.1 bis 1.0TW/cm^2 erreichen. Zur Entwicklung dieser Dioden wird für die Untersuchung und Charakterisierung der von ihnen erzeugten Teilchenstrahlen eine besondere Meßtechnik erforderlich, die unter dem Begriff 'Diagnostik' bekannt ist [2]. Diese Meßmethoden werden auch bei den Experimenten zur Strahl-Target-Wechselwirkung benötigt. Interessant sind z. B. die Aussagen über Größe und Lage des Teilchenfokus oder die Art der beteiligten Teilchen und deren Dichte und Energie.

Anforderungen an die Diagnostik ergeben sich aus den sehr hohen Spannungen, Strömen und Leistungsdichten in Verbindung mit sehr kurzen Impulszeiten. Es müssen Ereignisse untersucht werden, die sich im Nanosekunden-Bereich mit Amplitudenwerten im MV-, MA- und TW-Bereich abspielen. Daher müssen diese Meßgrößen durch geeignete Maßnahmen um mehr als 6 Zehnerpotenzen verringert werden. Außerdem finden die Messungen in einer für empfindliche elektronische Geräte 'feindlichen' Umgebung statt. Daher wurden im KfK Methoden und Apparate entwickelt bzw. weiterentwickelt, die auf optischen Verfahren beruhen und die Schwierigkeiten mit elektrischen Messungen vermeidet [2]. Eine Möglichkeit z.B. ist, die Teilchen durch geeignete Maßnahmen auf einen ionenempfindlichen Film abzubilden und diesen unter dem Mikroskop zu untersuchen.

Ein Ansatz zur Beurteilung der Teilchentrajektorien und damit des Teilchenfokus ist, den Ionenstrahl mit Ein- und Mehrlochkameras abzubilden. Eine Antwort auf die Frage nach der Strahlzusammensetzung bezüglich Teilchenart und Teilchenenergie ergibt sich durch den Einsatz eines speziellen Massenspektrometers. In jedem Fall dient der ionenempfindliche Film als Detektor. Auf der Filmoberfläche entstehen an den Positionen der Teilcheneinschläge nach entsprechender chemischer Behandlung kleine Krater mit Durchmessern im Mikrometer-Bereich. Diese Krater können unter dem Mikroskop nach bestimmten Kriterien ausgewertet werden um Aussagen über die Dichteverteilung des Strahls oder die Art und Energie der Strahlteilchen machen zu können.

Obwohl die Teilchenstromdichten, die auf den Filmen als Einschläge detektiert werden, schon um etliche Zehnerpotenzen reduziert wurden, ist die Dichte am Detektorort noch sehr hoch. Es müssen bis zu 10^5 Krater in mikroskopischer Größe pro mm^2 im Maximum auf dem Film nachgewiesen werden. Der Film selbst weist eine Fläche von bis zu 80cm^2 auf, die unter dem Mikroskop abzurastern ist. Dies macht eine maschinelle Auswertung unbedingt notwendig.

Im folgenden wird das System IODA (**Io**nen**d**ichte **A**nalyse) zur automatischen Auswertung der Filme beschrieben.

2. Problembeschreibung

Zur Charakterisierung der Teilchenstrahlen werden Diagnostikverfahren eingesetzt, bei denen die Teilchen auf einen ionenempfindlichen Film abgebildet werden. Diese ionenempfindlichen Filme bestehen aus CR-39, einem Material, das aus langen organischen Molekülen, sogenannten Polymeren aufgebaut ist [3]. Diese Moleküle bilden ein dreidimensionales Netzwerk. Durch Radiolyse werden Bindungen der Moleküle aufgebrochen, wenn hochenergetische Teilchen auf den Film treffen. Die Wechselwirkung der Teilchen mit den Film-Molekülen ist abhängig von der Art und der Energie der Teilchen. Ein Ätzmittel, z.B. NaOH, greift an den geschädigten Filmpositionen an und läßt kleine, in die Filmoberfläche eingegrabene Krater entstehen. Diese Ätzkegel sind unter dem Mikroskop als Löcher mit unterschiedlichen Durchmessern im Mikrometer-Bereich zu beobachten.

Wie oben schon erwähnt, hängt die Wechselwirkung der Teilchen mit dem Film von dem spezifischen Energieverlust der Teilchen ab, so daß verschiedene Teilchenarten oder verschiedene Energien auf dem Film unterschiedliche Schädigungen hinterlassen. So kann unter Umständen die Größe und Form der Krater auf dem CR-39-Film zur Analyse der Teilchen herangezogen werden. Hochenergetische Teilchen dringen tief in den Film ein, Teilchen mit niedriger Energie schädigen hauptsächlich die obere Filmschicht. Entsprechend werden durch hochenergetische Teilchen tiefe Krater mit kleinem Lochdurchmesser und durch niedrigere Energien flache Krater mit großen Durchmesser erzeugt. Bei gleichen Energien verursachen größere Teilchen Krater mit großem und kleine Teilchen Krater mit kleinerem Durchmesser. Absolute Aussagen lassen sich jedoch nicht machen, da die Ätzkegel-Größen sehr stark von der Dauer, Temperatur und Konzentration des Ätzbades abhängen.

Aufgabe des Systems ist die Unterstützung der Diagnostik-Gruppe des INR bei der Charakterisierung von Teilchenstrahlen bezüglich Leistungsdichte, Teilchenenergie und Teilchenzusammensetzung. Dazu müssen Teilchenspuren auf CR39-Filmen, die bei den KALIF-Experimenten anfallen, ausgewertet werden.

Es handelt sich im wesentlichen um zwei Arten von Bildern, die zu bearbeiten sind.

- Bilder aus der Lochkamera
- Bilder aus dem Thomsonschen Massenspektrometer

Die Bilder aus der Lochkamera dienen zur Charakterisierung der Teilchenstrahl-Qualität. Mit Hilfe der Bilder sind Aussagen über den Strahlfokus möglich. Der Lochkamera kann im Driftrohr der Teilchen eine Mehrloch-Apertur vorgeschaltet werden. Mit dieser *Shadow-Box* können Erkenntnisse über die Strahl-Trajektorien und die gerichtete Strahl-Leistungsdichte gewonnen werden (siehe Abbildung 1a).

Bei der Lochkamera wird ein intensiver Protonenstrahl, bei der Mehrlochapertur kleine Strahlbündel daraus, auf eine Bor-Streufolie abgebildet. In dieser Streufolie findet eine Kernreaktion $^{11}B(p,\alpha)2\alpha$ statt [2] bei der aus Protonen und ^{11}B im Mittel pro Reaktion 3 α-Teilchen freigesetzt werden. Diese α-Teilchen werden als Sekundärstrahl in der Lochkamera auf dem CR39-Film abgebildet. Durch die Streufolie wird auch die Intensität des Strahls auf ein für den Film erträgliches Maß reduziert.

Sind nach der Streufolie vor der Lochkamera im Teilchenstrahl keine Primärteilchen mehr, so wird auf dem Film nur eine Teilchenart abgebildet. Die unterschiedlichen Lochgrößen, die auf dem Film zu sehen sind, werden durch die unterschiedlichen Energien der gleichartigen Teilchen verursacht.

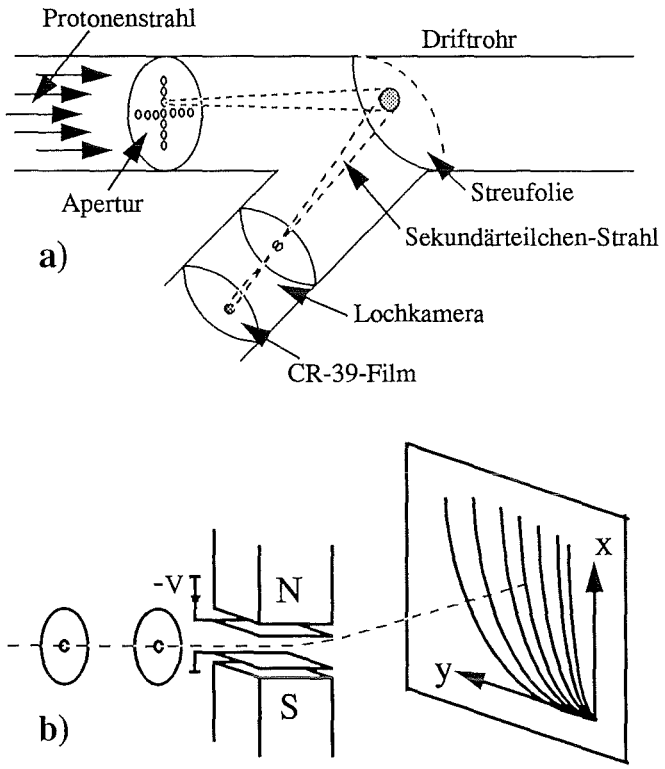


Abbildung 1. Experimentelle Umgebung.

a) Shadowbox-Lochkamera-Anordnung: Die im Driftrohr befindliche Mehrlochapertur für Shadowbox-Aufnahmen blendet aus dem Strahlquerschnitt schmale Strahlbündel aus, die auf einer Streufolie Sekundärteilchen erzeugen. Die Quelldichte der emittierten Sekundärteilchen wird mit Hilfe der Lochkamera auf den Detektor (CR-39-Film) abgebildet.

b) Massenspektrometer nach Thomson [2]: Ein dünner Strahl aus geladenen Teilchen durchquert ein elektrisches und dazu paralleles magnetisches Feld. Das Magnetfeld bewirkt die Ablenkung in y-Richtung, das elektrische in x-Richtung. Die daraus resultierende Ablenkung des Strahls ergibt eine parabelförmige Verteilung, wobei Teilchen mit verschiedenen Ladungs-zu-Massen-Verhältnissen auf verschiedenen Ästen der Parabelschar auftreffen.

Die Bilder aus dem Thomsonschen Massenspektrometer dienen zur Charakterisierung der im Strahl enthaltenen Teilchenarten und ihrer Energie.

Im diesem Massenspektrometer wird, wie in Abbildung 1 zu sehen ist, ein dünner Teilchenstrahl aus geladenen Ionen durch ein elektrisches und in gleiche Richtung zeigendes magnetisches Feld abgelenkt. Ionen gleicher Ladung und gleicher Masse fallen dabei abhängig von ihrer Energie auf einen Parabelast. Hochenergetische Ionen werden weniger stark abgelenkt und treffen näher beim Ursprung (Scheitel der Parabel) auf, niederenergetische werden stärker abgelenkt und liegen weiter außen auf dem Parabelast. Sind verschiedene Teilchenarten beteiligt, so ergeben sich für die unterschiedlichen Ionensorten verschiedene Parabeläste mit gemeinsamem Scheitelpunkt. Das ist der Ort, auf den alle nicht-abgelenkten (ungeladenen Teilchen) fallen.

Da in der Realität die Richtungen des elektrischen und des magnetischen Feldes nicht exakt parallel sind, sondern um ca. 6 Grad gegeneinander verkippt sind, erhält man keine Parabeln, sondern eine Kurvenschar, deren mathematische Beschreibung "etwas" komplizierter ist.

Bei dieser Art von Filmen beobachtet man verschiedene Lochgrößen, die von verschiedenartigen Teilchen und von unterschiedlichen Teilchenenergien herrühren. Allerdings sind die Teilchenarten, außer im Ursprung, örtlich durch die verschiedenen Parabeläste voneinander getrennt, so daß die Lochgrößen-Änderung innerhalb eines Parabelastes durch die sich kontinuierlich ändernde Teilchenenergie verursacht wird.

Wie oben dargelegt, fallen bei verschiedenen Experimenten Filme an, die zwar zu unterschiedlichen Aussagen führen, aber unter dem Mikroskop ähnlich aussehen. In jedem Fall sind kleine Löcher mit Durchmessern im Mikrometer-Bereich zu detektieren. Erst die entsprechende Nachbehandlung und die physikalische Betrachtung der Detektionsergebnisse ergibt eine unterschiedliche Aussage.

Es wird also ein System benötigt, mit dem man unter dem Lichtmikroskop die von den Teilchen verursachten Löcher vollautomatisch erfassen kann. Das System soll in der Lage sein, die Löcher sowohl in Auflicht- als auch in Durchlicht-Betrieb des Mikroskops zu erkennen und die unterschiedlichen Größen, falls nötig, zu selektieren. Durch die unterschiedliche Größe der Löcher ist eine Betrachtung des Filmes mit unterschiedlicher Auflösung nötig, was durch das Einschalten verschiedener Objektiv-Vergrößerungen möglich wird. Der Filmausschnitt, der dabei mit dem Mikroskop jeweils zu sehen ist, ist relativ klein (570µm*570µm bei Objektiv-Vergrößerung 5x und 28.5µm*28.5µm bei 100x). So können je nach auszuwertender Filmfläche leicht 10000 bis 200000 Bilder zu Analyse anstehen. Es muß daher ein hoher Durchsatz möglich sein. Natürlich soll das System auch für den Benutzer einfach zu handhaben sein.

3. Lösungsansatz

Die Analyse der Filme soll zerstörungsfrei durchgeführt werden. Da zudem, wie oben erläutert, die Einschläge auf den Filmen sehr klein sind und zur Beobachtung ein Mikroskop notwendig ist, bietet sich die digitale Bildverarbeitung und Mustererkennung als Methode an [4]. Im folgenden wird das methodische Vorgehen der Bildverarbeitung an Beispielen gezeigt.

Zu Beginn steht die Akquisition der zu verarbeitenden Bilder. Ein Bildgeber, üblicherweise eine Videokamera, liefert ein analoges Signal einer Bildszene. Dieses Signal wird in einem Videointerface digitalisiert und in Bildelemente (Pixel) gerastert. Man erhält damit ein quadratisches Bild mit 512*512 Bildelementen. Jedes Pixel erhält seinem Ort im Bild entsprechend einen Grauwert zwischen 0 und 255, wobei 0 schwarz und 255 weiß zugeordnet wird. Dieses digitalisierte Bild wird in einem Bildspeicher abgelegt und kann mit entsprechenden Operatoren manipuliert werden.

In Abbildung 2 werden beispielhaft zwei Arten von Bildmanipulationen vorgestellt, die dazu geeignet sind, interessierende Information vom Bildhintergrund zu trennen.

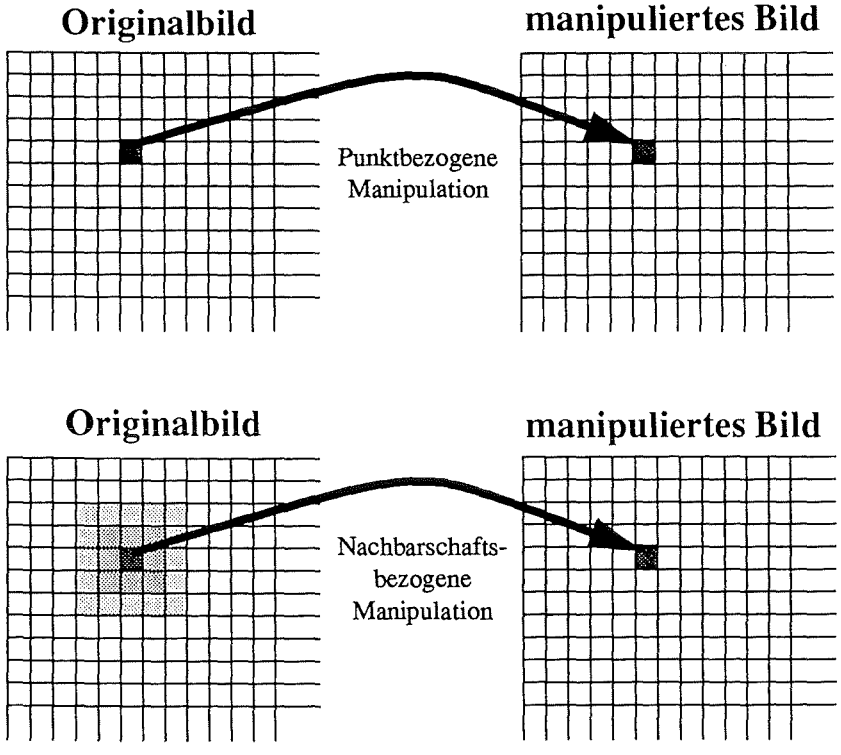


Abbildung 2. Schematische Darstellung allgemeiner Bildmanipulationen.

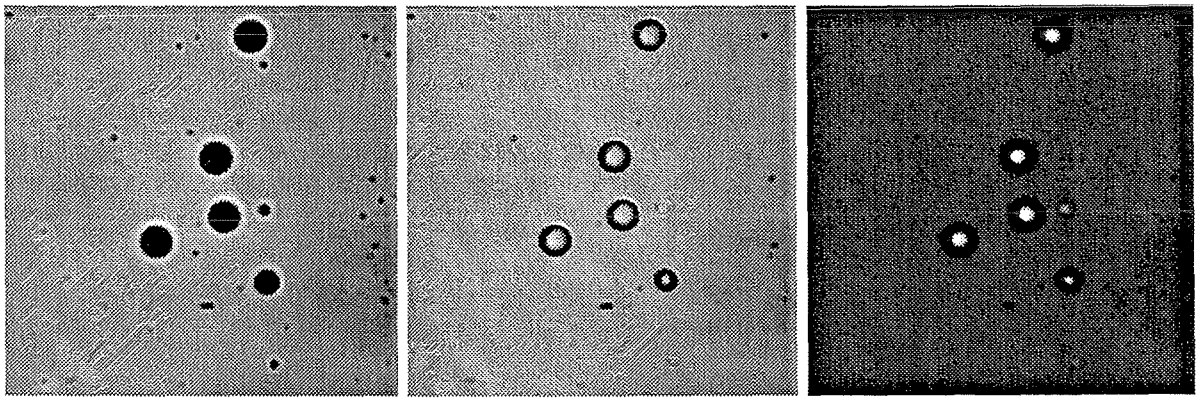
Bei der punktbezogenen Bildmanipulation werden die Pixelwerte des manipulierten Bildes einzig durch die entsprechenden Werte des Ursprungsbildes bestimmt. Beispiele dafür sind: Binarisierung, Skalierung, Addition und Subtraktion von Bildern.

Im Gegensatz dazu berücksichtigt die nachbarschaftsbezogene Manipulation für die Berechnung eines Pixels im Ergebnisbild eine definierte Umgebung des entsprechenden Pixels im Ursprungsbild. Beispiele dafür sind: Gradienten-, Mittelungs- und Rangordnungsfilter und das später beschriebene Kreis- und Sigma-Verfahren.

Während bei der ersten Art, der punktbezogenen Manipulation, für jedes Pixel im Ergebnisbild eine Rechenoperation notwendig ist, müssen bei der nachbarschaftsbezogenen Manipulation umfangreiche, von der Größe der Nachbarschaft abhängige, Berechnungen ausgeführt werden. Im Beispiel sind dazu für ein Ergebnispixel bei einer betrachteten 5*5-Nachbarschaft bei einem einfachen Gradientenfilter 25 Multiplikationen und ebensoviele Additionen notwendig.

Bei einem Bild mit einer Bildkante von 512 Pixeln kann man sich leicht ausrechnen, daß schon bei dieser relativ kleinen betrachteten Umgebung über 13 Millionen arithmetische Operationen ausgeführt werden. Aus dem Beispiel wird deutlich, daß für die Bildverarbeitung leistungsfähige Rechner einzusetzen sind. Da neben den Bildmatrizen der Originalbilder mit 1 Byte pro Pixel auch die der Zwischen- und Ergebnis-Bilder mit je 4 bis 8 Byte pro Pixel im System gehalten werden müssen, sind außer den leistungsfähigen Prozessoren auch große Bildspeicher nötig.

Unter dem Mikroskop besitzen die geätzten Löcher auf dem durchsichtigen CR39-Film je nach Fokuseinstellung unterschiedliches Aussehen. Dabei spielt es keine Rolle, ob das Mikroskop in Auf- oder Durch-Licht-Stellung betrieben wird. In jedem Fall werden, wie in Abbildung 3 zu sehen ist, die Löcher als kreisförmige Flächen abgebildet, die abhängig von der Fokuseinstellung in den Zentren der Kreise mehr oder weniger helle Flecken aufweisen.



a) b) c)

Abbildung 3. Einfluß der Fokussierung auf das Mikroskopbild.

Die geätzten Löcher werden unter dem Mikroskop abhängig von der jeweiligen Fokuseinstellung unterschiedlich abgebildet. Die Löcher erscheinen:

a) bei der Einstellung des Fokus kurz oberhalb der Filmoberfläche als kreisförmige, dunkle Flächen mit hellem Rand,

b) bei Fokuseinstellung auf die Filmoberfläche als dunkle Flächen, die zum Zentrum hin jedoch allmählich heller werden,

c) bei weiterer Fokussierung in die Tiefe als dunkle Ringe mit sehr hellen Zentren, die sich gut vom globalen Hintergrund abheben.

Mit Verstellung des Fokus zeigen sie sich entweder als kreisförmige dunkle Flächen mit hellem Rand, oder bei Fokussierung auf die Ränder als helle Randlinien um kreisförmige Flächen, die jedoch zum Zentrum hin wieder heller werden, oder als dunkle Ringe mit sehr hellem Zentrum. Da die Schärfentiefe beim Lichtmikroskop relativ klein ist, die Filmdicken aber örtlich leicht variieren können, kann der Film während der Auswertung aus dem eingestellten Fokus laufen. Deshalb muß auch mit den entsprechenden Zwischenformen gerechnet werden

Diesen Anforderungen entsprechend wurden Algorithmen entwickelt, um mit Hilfe der digitalen Bildverarbeitung die Löcher auf den Filmoberflächen erkennen zu können. Dabei wurde aufgrund der Menge der auszuwertenden Einzelbilder auch auf die Ausführungszeit geachtet.

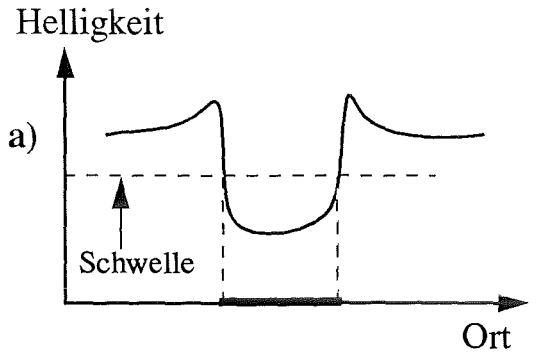
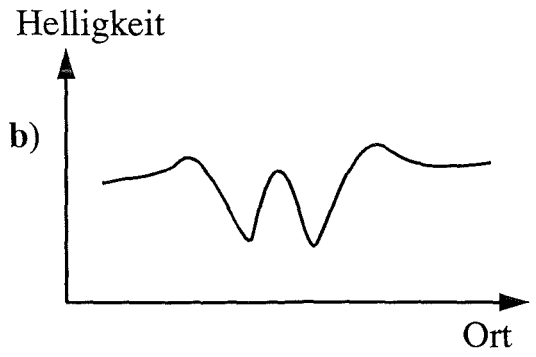


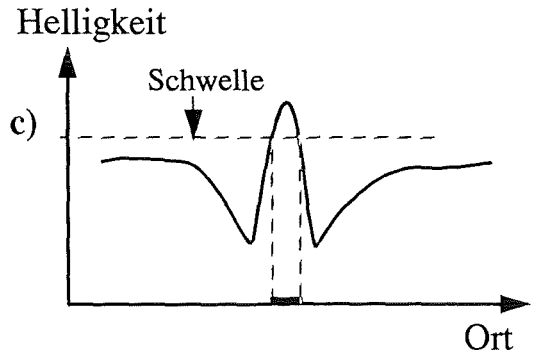
Abbildung 4. Grauwertverteilung der Löcher im Querschnitt.

Die Abbildung zeigt die Helligkeitsverteilungen der Löcher aus der vorangegangenen Abbildung.

a) Das Loch stellt sich als dunkle Fläche dar, so daß eine Schwelle berechnet werden kann, unterhalb der alle Grauwerte der Pixel liegen, die zum Loch gehören. Nach einer Binarisierung ist die Information 'Loch' vom restlichen Hintergrund separiert (Histomin-Verfahren).



b) Das Loch stellt sich als dunkler Ring mit mäßig hellem Zentrum dar. Da keine Schwelle existiert, mit deren Hilfe das Loch vom Hintergrund getrennt werden kann, wurde dafür das Sigma-Verfahren entwikkelt.



c) Das Loch stellt sich als dunkler Ring mit sehr hellem Zentrum dar. So kann eine Schwelle berechnet werden, oberhalb der das helle Zentrum vor dem restlichen Hintergrund als zum Loch gehörend klassifiziert und damit separiert werden kann (Histomax-Verfahren).

Histogramm-Methoden:

Die einfachsten und schnellsten Algorithmen sind diejenigen, die die oben beschriebenen punktbezogene Manipulationen benutzen. Zu diesen Manipulationen zählen die hier benutzten Histogramm-Methoden. In Abbildung 4 sind eindimensionale Schnitte durch die Grauwertver-

teilungen verschiedener unter dem Mikroskop aufgenommener Lochtypen dargestellt. Bei den Typen (a) und (c) unterscheidet sich die Helligkeit der Lochzentren deutlich von der Helligkeit des Bildhintergrundes. Das Histogramm eines Bildes zeigt, wie häufig ein Grauwert im Bild auftritt (siehe Abbildung 5).

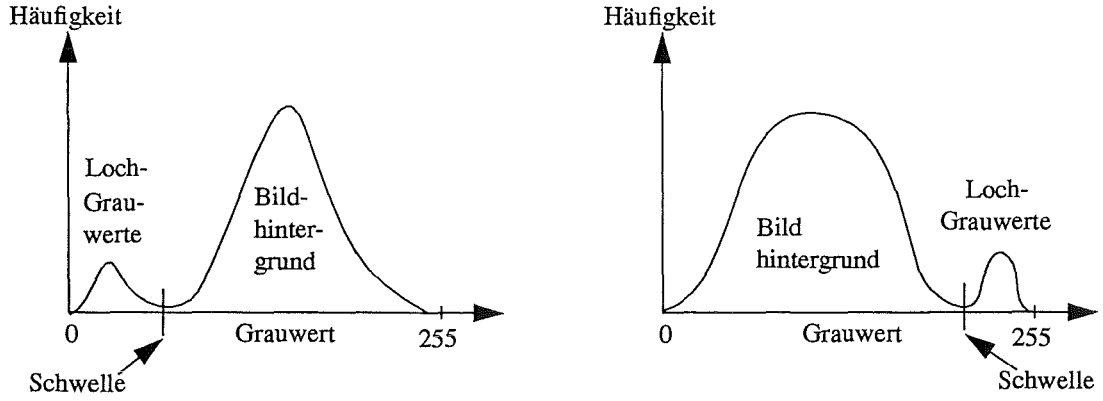


Abbildung 5. Beispiele für Histogramme idealer Bilder.
Links ist ein Histogramm dargestellt, bei dem die Grauwerte der Lochzentren deutlich dunkler sind als der globale Bildhintergrund. Die rechte Skizze zeigt das Histogramm eines Bildes, bei dem die Lochgrauwerte deutlich heller sind als der Bildhintergrund.
In beiden Fällen kann eine Schwelle definiert werden, mit der das Bild binarisiert werden kann.
Dieses Verfahren greift dann gut, wenn die Häufigkeit des Schwellwertes und seiner direkten Umgebung sehr gering ist.

Hat man ideale Verhältnisse, so läßt sich mit statistischen Algorithmen [5], wie in Abbildung 5 dargestellt, eine eindeutige Schwelle berechnen. Diese Schwelle kann zur Binarisierung des Bildes herangezogen werden. Alle Pixel, deren Grauwert kleiner ist als der Schwellwert, erhalten den Wert 0, alle anderen den Wert 1.

In einem nachfolgenden Labelling-Schritt werden die zu einem Loch gehörenden Pixel zusammengefaßt. Bei dem Beispiel mit den dunkleren Lochzentren erhalten jeweils alle zusammenhängenden Pixel mit den Werten 0 ein gemeinsames Label, bei dem Beispiel mit den helleren Lochzentren die zusammenhängenden mit dem Pixelwert 1. Um die Anzahl der Löcher auf einem Bild zu bestimmen, müssen dann nur noch die vorhandenen unterschiedlichen Label gezählt werden.

Sigma-Methode:

Im allgemeinen hat man bei den aufgenommenen Bildern nicht die eben beschriebenen idealen Verhältnisse. Sehr häufig werden die Löcher unter dem Mikroskop so abgebildet, daß das Innere des Loches dunkler als der globale Bildhintergrund ist, das hellere Zentrum jedoch Grauwerte in der Größenordnung des Hintergrundes besitzt. Für diese Fälle wurde die Sigma-Methode entwickelt.

Der Algorithmus gehört zu den nachbarschaftsbezogenen Bildmanipulationen und ist entsprechend zeitaufwendig. Zur Entscheidung, ob ein Pixel zu einem Loch gehört oder zum nicht zu beachtenden Hintergrund wird die Nachbarschaft dieses Pixels folgendermaßen mit herangezogen:

Für jedes Pixel auf dem digitalisierten Bild wird der Mittelwert $\bar{u}$ und die Streuung σ der Grauwerte der jeweiligen Nachbarschaft berechnet (siehe Abbildung 6). Die Gestalt (Ausdehnung und Form) der zu berücksichtigenden Nachbarschaft kann mit einem Template definiert werden. Dieses Template wird in IODA als Maske bezeichnet.

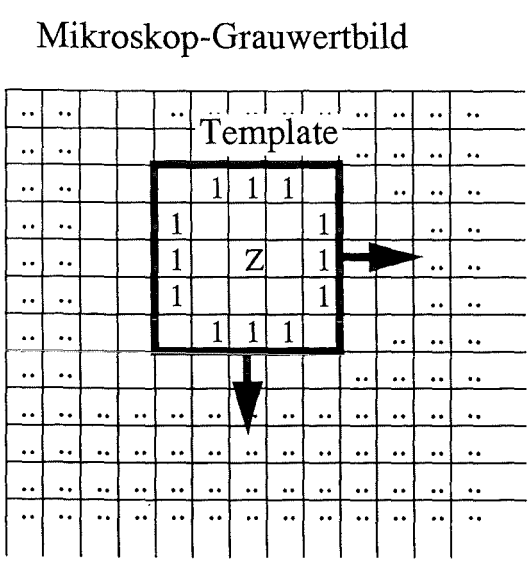


Abbildung 6. Sigma-Verfahren.

Das Sigma-Verfahren geht davon aus, daß ein Loch wie ein kreisförmiges Objekt mit dunklem Ring und leicht hellerem Zentrum aussieht.

Für das zentrale Pixel Z wird durch ein Template, das vom Benutzer frei gewählt werden kann, eine bestimmte Umgebung definiert (1er-Werte des Templates). Entsprechend dieser Umgebung wird aus den dazugehörenden Grauwerten des Mikroskop-Bildes der Mittelwert $\bar{u}$ und die Standardabweichung σ berechnet. Ist der Grauwert des Zentralpixels größer als der Wert $\bar{u} + f*\sigma$ (f ist ein Faktor, der vor der Auswertung experimentell zu bestimmen ist), dann wird das zentrale Pixel als zu einem Loch gehörend klassifiziert.

Auf jedem Mikroskop-Grauwertbild wird auf diese Weise mit einem Template Bildpunkt für Bildpunkt überlagert, $\bar{u}$ und σ berechnet und damit die Löcher klassifiziert.

Diese Methode setzt voraus, daß um das etwas hellere Zentrum eines Loches ein gleichmäßiger dunklerer Ring (mit geringer Streuung) vorhanden ist. Das Zentrumspixel Z wird dann als zum Loch gehörig klassifiziert, wenn die folgende Relation gilt:

$$\textbf{Grauwert von Z} > \bar{u} + f * \sigma$$

mit: $\bar{u}$ = mittlerer Grauwert , σ = Streuung jeweils der definierten Umgebung.

Der Faktor **f** muß vor der Auswertung durch Tests experimentell bestimmt werden. Sein Wert hängt von der Varianz des globalen Hintergrundes ab.

Sind die Pixel bestimmt, die zu einem Loch gehören, werden sie von einem Labelling-Algorithmus als zusammenhängende Flächen mit einem gemeinsamen Wert versehen. Zur Bestimmung der Einschlagdichte werden wieder die unterschiedlichen Label gezählt.

Ein Nachteil des Verfahrens ist, daß mit der Wahl der Größe und der Form des Templates schon eine Auswahl der zu detektierenden Löcher verbunden ist. Außerdem steigt die Rechenzeit mit der Größe des Templates drastisch an.

Kreis-Methode

Eine weitere Methode, die nachbarschaftsbezogene Bildmanipulationen anwendet, ist die sogenannte Kreis-Methode. Sie wurde deshalb so bezeichnet, weil die Löcher in Zwischenergebnis-Bildern in solchen charakteristischen Formen abgebildet werden (siehe Abbildung 7).

Das Verfahren geht davon aus, daß die Löcher so abgebildet werden, daß ein helleres Zentrum

vorhanden ist und die Grauwerte zum Rand des Loches kleiner werden. Die Grauwerte haben Ähnlichkeit mit den z-Werten eines Kegels. Betrachtet man nach der Anwendung eines speziellen Gradienten-Filters die dabei ermittelten Richtungen der maximalen Gradienten, so stellt man fest, daß diese für alle Pixel vom Zentrum des Loches (Kegelspitze) radial zum Lochrand zeigen, was den Löchern auf dem Richtungsbild dieses eigentümliche Aussehen verleiht (Abbildung 7). Mit einem Filter-Kernel, der auf das Richtungsbild angewendet wird, sucht man nach diesen radialen Richtungsformen und findet als Vertreter für ein Loch ein oder mehrere Pixel im Zentrum. Durch ein Labelling-Verfahren werden wieder die zusammengehörenden klassifizierten Pixel eines Loches zusammengefaßt, um dann wie üblich die Anzahl der Löcher im Bild zu bestimmen.

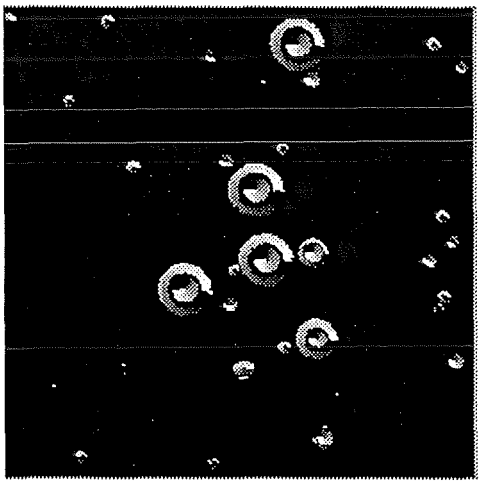


Abbildung 7. Richtungsbild

Mit Hilfe eines Sobel-Filters wird aus den Gradientenwerten in x- und y-Richtung für jede Pixel-Position der Gradienten-Betrag und die Richtung dieses maximalen Gradienten berechnet und jeweils in einer Bild-Matrix abgelegt. In der Abbildung ist das Richtungsbild aus dieser berechnung dargestellt. Die Werte sind so skaliert, daß schwarz 0 und weiß 360 Grad entspricht. An den Pixel-Positionen, die einen nicht signifikanten Gradienten-Wert aufweisen, wird im Richtungsbild 0 eingetragen.

4. Realisation der Auswertung

4.1 Vorgehensweise

Auswertung

Als Ergebnis interessiert die Einschlagdichte der Ionen auf dem CR39-Film. Diese Einschlagdichte ist gleichbedeutend mit der Anzahl der Löcher auf einer Fläche definierter Größe. Deshalb werden möglichst alle Löcher erfaßt und ausgewertet.

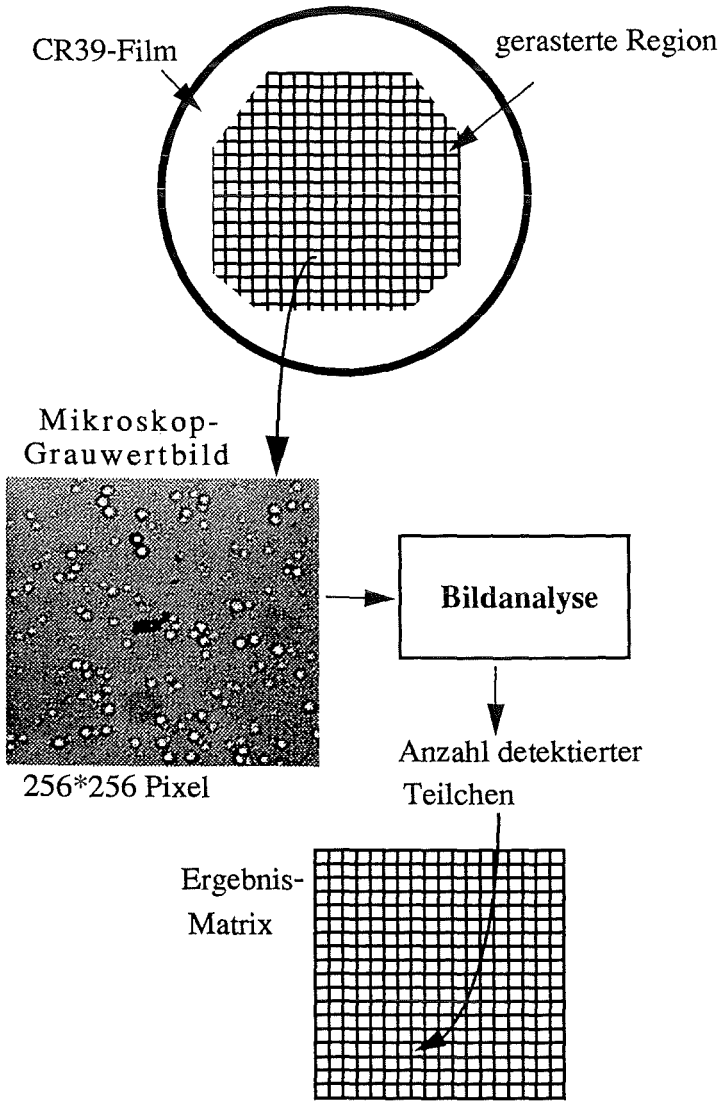


Abbildung 8. Ablauf einer CR-39-Filmauswertung.

Die auszuwertende Fläche des CR-39-Films wird abhängig von der erforderlichen Vergrößerung aufgerastert. Unter dem Mikroskop wird jedes Flächenelement dieses Rasters angefahren und jeweils ein Bild mit 256 Graustufen aufgenommen. Mit Hilfe der digitalen Bildanalyse werden die Löcher auf dem Bild klassifiziert und deren Anzahl für dieses Flächenelement in einer Ergebnismatrix an der entsprechenden Stelle eingetragen.

Bei der Auswertung rastert man den Film in gleichgroße Flächeneinheiten auf, in denen jeweils die Löcher unter dem Mikroskop ausgewertet werden. Die Auswertung erfolgt vollautomatisch: Nach der genauen Positionierung im Mikroskop wird mit der Video-Kamera ein Grauwert-Bild aufgenommen. Dieses 'Einzelbild' wird mit bildanalytischen Verfahren bearbeitet, so daß am Ende nur die Löcher als einzelne Objekte dargestellt sind, die dann gezählt

werden können. Auf die bildanalytische Detektion der Löcher wurde schon eingegangen.
Das Ergebnis der Auszählung wird in einer Matrix entsprechend dem Rasterpunkt abgelegt (siehe Abbildung 8). Bei gleichmäßigem Rasterabstand kann man die Matrix als direkte Abbildung der Einschlagdichte auf dem CR39-Film betrachten. Sie läßt sich unter diesem Gesichtspunkt weiter auswerten.

Rasterung

Die Parameter für die Rasterung hängen von mehreren Faktoren ab. Die Vergrößerung des Mikroskops ist so zu wählen, daß die kleinsten Löcher groß genug sind, um sie als solche zu erkennen. Die Erkennbarkeit ist deshalb ein Kriterium, da das Bild später im Rechner in eine Anzahl Bildpunkte (Pixel) zerlegt wird, und Objekte, die nicht größer als einige Pixel sind, nicht erkannt werden können. Die Fläche, die in einem solchen 'Einzelbild' erfaßt wird, ist damit Grundlage für die Einschlagdichte. Stimmt die Rasterweite mit der Kantenlänge dieser Fläche überein, so kann man alle auf dem Film befindlichen Löcher zählen. Für statistische Aussagen ist man in der Wahl der Rasterweite jedoch frei.
In unserem Fall liegt die Größe der Löcher bei 5-30 µm. Die Flächenkanten sind je nach Vergrößerung 147, 285 oder 570 µm lang. Bei Zerlegung in 256 Bildpunkte ergibt das ca. 0.5 bis 2 µm pro Bildpunkt. Bei einer Filmgröße von 50x50 oder 80x100 mm würde das bei exakter Rasterung zwischen 87x87 und 560x700 Rasterpunkte bedeuten.

4.2 Rechner-Konzept

Architektur

Aus dem Vorgehen für die Auswertung und aus der Diskussion der Rasterung läßt sich leicht erkennen, daß ein erheblicher Rechen- und damit Zeitaufwand nötig ist. Eine Auswertung, die z.B. aus einer 200x200 Rastermatrix besteht, bearbeitet 40000 Einzelbilder. Geht man von der Bearbeitungsdauer 1 sec für ein Einzelbild aus, so sind für die komplette Auswertung mehr als 11 Stunden notwendig. In der Realität dauert die Verarbeitung des Einzelbildes aber eher zehnmal so lange, woraus folgt, daß akzeptable Auswertezeiten nur erreicht werden können, wenn mehrere Rechner die Auswertung gleichzeitig durchführen (Multiprozessor-System).
Um flexibel auf eventuell neue Anforderungen reagieren zu können, wurde eine Multiprozessor-Architektur gewählt, die beliebig erweiterbar ist. So kann die Rechenleistung der jeweiligen Anforderung des Systems gut angepaßt werden. Diese Architektur läßt sich realisieren, indem man das System so aufteilt, daß jeder Prozessor selbständig ein Einzelbild verarbeitet. Man muß dann mit einem Master-Rechner für die Verteilung der Aufträge und das Sammeln der Ergebnisse sorgen.
Die Geschwindigkeit des Systems wird bei diesem Aufbau nur durch die einmalig vorhandenen Ressourcen bestimmt: Positioniergeschwindigkeit des Tisches, Bildeinzug, Verwaltung der Aufträge im Master-Rechner. Momentan liegt die Verarbeitungsgeschwindigkeit im Durchsatz für ein Einzelbild bei unter 0.5 sec, d.h. eine komplette Auswertung dauert weniger als 6 Stunden.

Hardware-Aufbau

Die Rechner-Hardware wurde mit handelsüblichen Standard-Komponenten realisiert, die sich

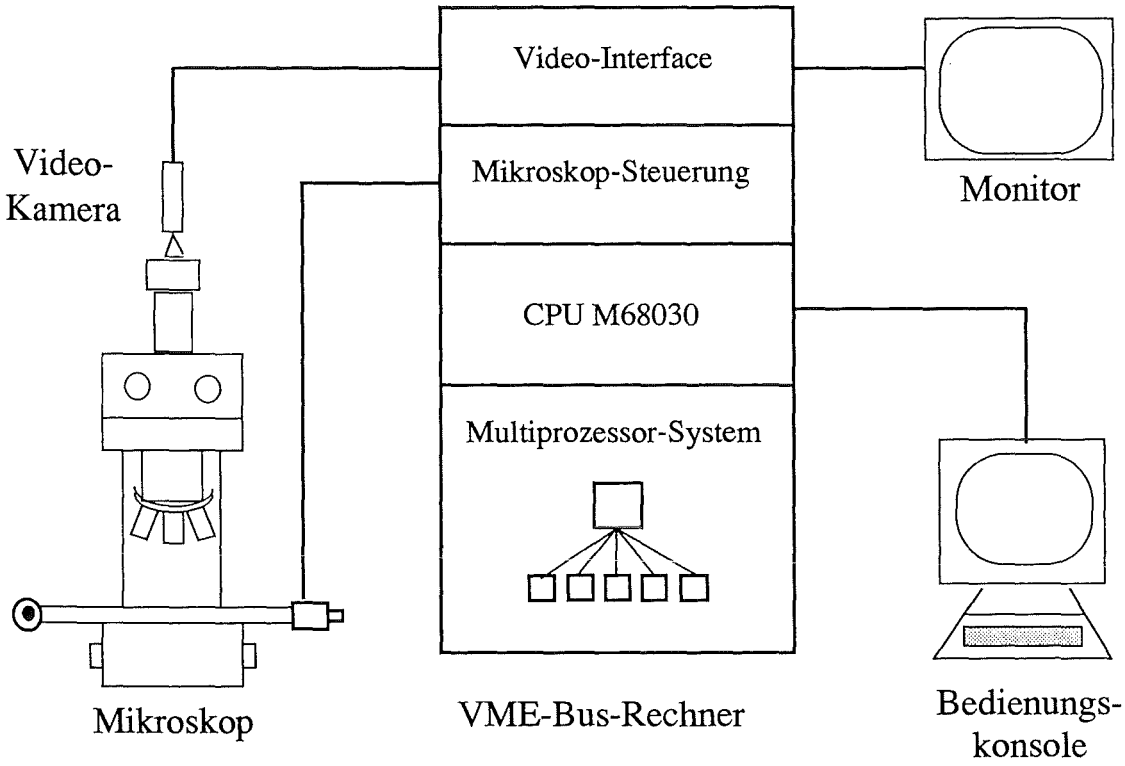


Abbildung 9. Übersicht über den Hardware-Aufbau des Analysesystems.
Das Rückgrat des Systems bildet ein Standard-Industrie-Rechner auf der Basis des 68030-Prozessors von Motorola mit VME-Bus. Die Master-CPU kommuniziert mit dem Benutzer, steuert den Mikroskop-Kreuztische, akquiriert die Video-Bilder und verwaltet die Auswertungsergebnisse. Sie übergibt die Bilddaten und Auswerteparameter als Aufträge an das Multiprozessor-System, das die Einzelbilder bearbeitet.

in einem 19" Rack unterbringen lassen (siehe Abbildung 9). Alle Rechner-Komponenten benötigen zusammen einen Platz von 11 HE. Es handelt sich um einen VME-Bus-Rechner mit 32-Bit-Architektur auf der Basis des Motorola 68030/68882-25 mit 8MB RAM. Außer dem Massenspeicher und seriellen und parallelen Schnittstellen, mit denen der Scanning-Tisch und der Objektiv-Revolver des Mikroskops gesteuert werden, verfügt der Rechner auch über eine Netz-Anbindung mit TCP/IP.

Die Bilder werden mit einer Sony CCD-Kamera mit 2/3" Sensor (756x581 Bildpunkte) aufgenommen und von einer Digitalisierungs-Karte im Format 512x512 mit 256 Graustufen im Rechner abgelegt. Das digitalisierte Bild ist 'live' auf einem Monitor sichtbar.

Die Multiprozessor-Architektur wurde mit Transputern realisiert. Eingesetzt sind Module, die jeweils einen Transputer mit Floatingpoint-Prozessor und 4MB RAM enthalten. Zusätzlich wird ein Transputer als Verbindungs-Rechner verwendet, der mit dual-ported-RAM versehen ist, d.h. auf den Speicher kann auch vom VME-Bus her zugegriffen werden.

Multiprozessing

Wie bereits erwähnt, ist das System in verschiedene Teile untergliedert, die gleichzeitig aktiv

sind. Die Steuerung des Mikroskops, Einziehen des Bildes, und das Abspeichern des Ergebnisses sind Aufgaben des Master-Rechners. Die Auswertung (Auszählung) eines Bildes ist so konzipiert, daß sie parallel dazu in einem unabhängigen Prozessor ablaufen kann. Das bedeutet, der Master-Rechner kann bereits das nächste Bild positionieren, während das aktuelle ausgewertet wird. Halten sich diese Zeiten die Waage, so ist das System gut ausgelastet. In der Praxis dauert die Auswertung aber ungleich länger. Setzt man weitere Prozessoren für die jeweils nächsten Bilder ein, läßt sich ein Zustand erreichen, in dem der Gesamt-Durchsatz der Auswertungen der Positioniergeschwindigkeit entspricht.

Dieser Aufbau erfordert zusätzlichen Verwaltungsaufwand, der aber durch die wesentlich besseren Auswertezeiten ausgeglichen wird. Stoppt man z.B. das System, muß berücksichtigt werden, daß sich noch Aufträge im System befinden, die in Bearbeitung sind. Erst nach 'Leer-Rechnung' aller Prozessoren kann das System ordentlich abschließen.

5. Systemhandhabung (Manual)

5.1 Aufruf und Bedienung

Das System IODA zur Ionen-Dichte-Analyse besteht aus drei Programmteilen:

IODA, dem Hauptprogramm, das die beiden anderen Teile auch automatisch aktivieren kann,

IODA_ED, dem Editor, mit dem Auswerte-Bereiche interaktiv festgelegt werden können, und

IODA_IF, dem User-Interface, das während einer laufenden Auswertung aktiv ist, und Überwachungsfunktionen auch von anderen Terminals aus zuläßt.

Die Bedienungsoberfläche ist menue-gesteuert, d.h. die Programme sind in Funktionsblöcke aufgeteilt, die dem Benutzer die Auswahl einer bestimmten Befehlsmenge gestatten. Der Aufbau und der Umgang mit den Menues ist methodisch gleich und wird im Anhang kurz zusammengefaßt dargestellt.

Voraussetzung für den Start des Systems ist die Anmeldung des Benutzers auf dem VME-Bus-Rechner. Nach der "Login-Prozedur" sollte in das Auswerte-Directory gewechselt werden, das ggf. erst eingerichtet werden muß. Dann kann das System IODA zur Ionen-Dichte-Analyse durch Eingabe von "ioda" und dem Drücken der <RETURN>-Taste gestartet werden.

Nach dem Start erhält man auf dem Monitor die im folgenden gezeigte Zeile und eventuell noch einen Hinweis darauf, daß der Treiber für den Mikroskop-Kreuztisch gestartet wurde. Anschließend erscheint der ioda-Prompt, der die Bereitschaft zum Lesen einer Eingabe in diesem Menue anzeigt. Gleichzeitig hat man auf diese Weise die Information, in welchem Menue man sich gerade befindet.

Beispiel:

```
> ioda

        - - I O D A - -  Version 1.01  v. 23.11.94 --

ioda>
```

In jedem Menue (Haupt- oder Untermenue) hat man die Möglichkeit durch Eingabe eines Fragezeichens "**?**" den entsprechenden Befehlsumfang abzufragen. Das Zeichen ">" zeigt an, daß dieser Befehl zu einem Untermenue führt, wie im folgenden Beispiel "**parms**", "**tisch**", "**kamera**" und "**methode**". Als Eingabe reicht eine signifikante Buchstabenfolge, um den Befehl ausführen zu lassen.

Beispiel:

```
ioda> ?
    #       Kommentar ( blank nach # !! )
    exec    Ausfuehrung einer Datei
    bye     Beenden einer exec-Ebene ( und Programm )
    shell   lokale Shell aufrufen
    dir     Directory anzeigen oder aendern
    free    freien Hauptspeicher anzeigen
    load    Workspace laden
    save    Workspace abspeichern
```

```
    protokoll     Protokoll-Modus schalten
  > parms         Parameter fuer die Auswertung bearbeiten
  > tisch         Bedienung des Tisches - Erstellen der Fahrmaske
  > kamera        Bedienung und Einstellungshilfen fuer die TV-Kamera
  > methode       Zuordnung und Parameter der Methoden
    test          Ausfuehren einer Test-Auswertung
    auswertung    Ausfuehren der kompletten Auswertung
    edit          Editieren einer Bereichs-Datei
    menue         Switch fuer Menue-Mode
    demo          Switch fuer demo-mode
ioda>
```

Es ist auch möglich, aus einem Menue heraus eine Anweisung eines anderen Menues auszuführen, ohne das Menue zu verlassen. Zum Beispiel kann man mit dem Befehl **kamera live** (oder der eindeutigen Abkürzung **k l**) im Menue **kamera** das live-Schalten der Videokarte ausführen, ohne das aktuelle Menue zu verlassen.

5.2 Hauptprogramm IODA

Mit der Folge "Befehlsname ?" erhält man die Aufrufsyntax für den entsprechenden Befehl. Ist der Befehlsname ein Untermenue-Name, so wird der Befehlsumfang des Untermenues angezeigt.

Beispiel:

```
ioda> shell ?
    Syntax: shell [<cmd> [<parm>]...]
ioda>
```

Die Aufrufsyntax aller Befehle erhält man mit "??".

Beispiel:

```
ioda> ? ?
    # [<text>]..
    exec [<datei>]
    bye
    shell [<cmd> [<parm>]...]
    dir [<new_dir>]
    free
    load [<datei>]
    save [<datei>]
    protokoll [<datei> [+]|-]
  > parms
  > tisch
  > kamera
  > methode
    test [vld] [<spot>]
    auswertung [r]
```

```
    edit [<datei>]
    menue [+|-]
    demo [+|-]
ioda>
```

Mit **exec [<dateiname>]** können die Anweisungen aus einer Datei sukzessive abgearbeitet werden. Diese Anweisungen können per Editor erstellt werden oder dadurch, daß durch **protokoll [<dateiname>]** eine IODA-Sitzung oder Teile davon protokolliert werden, wobei durch + oder - der Protokollmodus ein- oder ausgeschaltet werden kann. Mit # hat man die Möglichkeit, Kommentar in den Ablauf einzufügen. Dabei ist darauf zu achten, daß hinter dem Kommentarzeichen mindestens ein Leerzeichen zu stehen hat.

Die Eingabe von **bye** beendet genauso wie das Betätigen der <ESCAPE>-Taste das Programm, wobei jedoch diese Aktion bestätigt werden muß, damit nicht versehentlich abgebrochen wird.

Beispiel:

```
ioda> bye
    Wollen Sie IODA wirklich verlassen ?? (j/n) : n
ioda>
```

Der Aufruf von **shell** bewirkt, daß von IODA ein OS-9-Shell-Kommando gestartet wird. Dabei wird IODA nicht verlassen, sondern wartet auf das Ende des Kommandos. Wird kein Parameter angegeben, so startet die Shell interaktiv. In diesem Fall erscheint ein Prompt mit einer Nummer, worauf alle OS-9-Kommandos ausgeführt werden können. Um die Shell zu beenden und zu IODA zurückzukehren, kann der Befehl **logout** eingegeben oder die ESCAPE-Taste (wird mit eof beantwortet) betätigt werden.

Beispiel:

```
ioda> shell
1.> pd
   /h0/INR/IODA/3357
1.> dir -e
                    Directory of . 20:01:41
   Owner  Last modified  Attributes Sector    Bytecount   Name
   -------  ------------  ----------    ------    ---------   ----
    4.9   92/12/17 1023  ------wr   798          214    3357polygon
    4.9   94/11/11 1644  ------wr   6A2F8        213    3357polygonlage6
    4.9   92/12/17 1407  ------wr   23410     131072    3357sigma7.L0
    4.9   92/12/17 1406  ------wr   22C1C        347    3357sigma7.prot
    4.9   92/12/17 1101  ------wr   7A0          853    3357sigma7.prt
    4.9   92/12/17 1101  ------wr   79C         1444    3357sigma7.save
    4.9   94/11/11 1846  ------wr   6B194     131072    testlage6.L0
    4.9   94/11/11 1846  ------wr   6A318        351    testlage6.prot
    4.9   94/11/11 1648  ------wr   6A314        857    testlage6.prt
    4.9   94/11/11 1648  ------wr   6A30C       1448    testlage6.save
1.> <ESC>
ioda>
```

Mit **dir** kann der Name des aktuellen Verzeichnisses angezeigt werden, oder es kann in ein anderes Verzeichnis gewechselt werden. IODA benutzt dieses dann als aktuelles Directory.

Beispiel:

```
ioda> dir
    /h0/INR/IODA/3357
ioda> dir ../3372
    /h0/INR/IODA/3372
ioda>
```

free zeigt die Größe des aktuell freien Hauptspeichers an.

Beispiel:

```
ioda> free
    Current total free RAM: 6056.00 K-bytes
ioda>
```

Durch den Befehl **load [<dateiname>]** hat man die Möglichkeit, alle bei einer früheren Sitzung definierten und gesicherten Parameter in die aktuelle Sitzung zu laden. Das Sichern der Parameter geschieht beim Start einer Messung automatisch oder kann durch den Befehl **save [<dateiname>]** ausdrücklich veranlaßt werden. Die Extension *.save* wird automatisch angehängt und darf weder bei **load** noch bei **save** angegeben werden. Beim Start der Messung erhält die Sicherungsdatei den Namen der Messung, der im PARMS-Menue als **name** angegeben wurde. Wird beim Befehl **save** kein Dateiname mit angegeben, so heißt die Sicherungsdatei *ioda.save* und wird im aktuellen Directory abgelegt.

Beispiel:

```
ioda> save ?
    Syntax: save [<datei>]
ioda> load testlage6
ioda>
```

Mit dem Kommando **test [v|d] [<spot>]** wird eine definierte Methode benutzt, um an der aktuellen Position des Mikroskopkreuztisches eine Testauswertung eines Bildes durchzuführen (nützlich bei der Ermittlung der Auswerte-Parameter). Dabei sind durch die verschiedenen Argumente, die mit dem Befehl angegeben werden, die folgenden unterschiedlichen Modi möglich.

Das Flag **v** (verify) bewirkt, daß überprüft wird, ob die aktuelle Position des Kreuztisches einem Index-Punkt der erstellten Fahrmaske (siehe Untermenue **tisch**) entspricht und somit einem in **edit** definierten Segment zugeordnet werden kann. Ist dies der Fall, so wird an der Position ein Bild akquiriert und mit der Methode, die dem entsprechenden Segment **connecte**d ist (siehe Untermenue **methode**) ausgewertet. Existiert keine gültige Fahrmaske, oder kann die aktuelle Position keinem Index der Fahrmaske zugeordnet werden, so wird das akquirierte Bild mit der Methode, die aktuell im Untermenue **methode** definiert ist, ausgewertet.

Das Flag **d** (display) bewirkt, daß kein Bild akquiriert wird, sondern das auf dem Display angezeigte Bild mit der in **methode** definierten Methode ausgewertet wird. Dadurch ist es möglich, ein archiviertes Bild, das mit einer externen Prozedur (z.B. Testrahmen-Programm **t** aus einer Tochter-Shell von IODA heraus aktiviert) auf das Display gebracht wurde, auszuwerten (Achtung! Dies ist eine Aktion für den fortgeschrittenen Benutzer).

In jedem Fall kann für **spot** ein Wert 0, 1 oder 2 eingesetzt werden, der angibt, ob das volle 256*256-Bild, ein 128*128-Ausschnitt oder ein 64*64-Spot jeweils um das Zentrum des aufgenommenen Bildes herum ausgewertet werden soll. Keine Angabe entspricht dem Wert 0.

Durch Benutzung der Spot-Option läßt sich die Auswertezeit auf 1/4 bzw. 1/16 gegenüber der Vollbild-Auswertung reduzieren.

Nach der Test-Auswertung wird der bearbeitete Ausschnitt des Grauwertbildes auf dem Display dargestellt und die als Löcher klassifizierten Positionen auf dem Bild als roter Fleck überlagert.

Beispiel:

```
ioda> test
   Prozessor 3; Time = 5382 ms
    0:    191

ioda>
```

Auf dem Terminal-Monitor wird nach Erledigung der Testauswertung angezeigt, welcher Prozessor der Multiprozessor-Hardware die Auswertung durchgeführt hat (zufällig) und wie lange er dazu brauchte. In dem Beispiel benötigte der Prozessor 3 5382 Millisekunden und fand auf dem Bild 191 Löcher.

Mit den Schaltern **menue** und **demo** können die entsprechenden Modi *getoggelt* oder mit + bzw. - gezielt ein- bzw. ausgeschaltet werden.

Eingeschalteter Menue-Mode bewirkt, daß z.B. beim Aufruf eines Untermenues automatisch dessen aktuelle Einstellung (Status) auf dem Monitor ausgegeben wird. Dasselbe geschieht nach einer Parameter-Änderung in dem Untermenue.

Bei eingeschaltetem Demo-Mode wird bei der zu startenden Auswertung, genauso wie bei der **test**-Auswertung das Ergebnis auf dem Display farbig markiert angezeigt. Dies hat allerdings auch zur Folge, daß die Auswertung etwas länger dauert, da durch die Darstellung des Bildes zusätzlich Zeit benötigt wird.

Das Kommando **auswertung** startet die automatische komplette Auswertung. Vorher werden vom System alle Parameter, die für die Auswertung benötigt werden, überprüft und nach erfolgreichem Check unter dem Namen der Messung (siehe Untermenue **parms**) mit der Extension *.save* im aktuellen Directory abgespeichert. Überprüft wird, ob ein Meßbereich definiert und eine Fahrmaske erstellt wurde (siehe **parms, edit, tisch**), ob eine Auswerte-Methode definiert und mit dem zu messenden Filmsegment korreliert wurde (siehe **methode**) und ob für das Meßergebnis ein gültiger Dateiname definiert wurde (siehe **parms**).

Soll eine abgebrochene Auswertung wiederaufgenommen werden, muß das Flag **r** als Argument angegeben werden. Der Benutzer hat in diesem Fall darauf zu achten, daß die Meßparameter der abgebrochenen Auswertung benutzt werden (z.B. durch Laden der entsprechenden gesicherten *.save*-Datei mit dem Kommando **load**).

Kann die Auswertung durchgeführt werden, so wird automatisch das Programm **ioda_if** im 'Master-Mode' gestartet. **ioda_if** übernimmt die Monitor-Anzeige und dokumentiert den Fortgang der Auswertung und handelt den Dialog mit dem Bediener ab, während **ioda** im Hintergrund die Auswertung durchführt.

Beispiel:

```
ioda> aus
   23.11.1994 - 21:01:41  Auswertung wird gestartet fuer 24460 Bilder
   Datei-Initialisierung ..

      - - I O D A - -  Version 1.01  v. 23.11.94  --
   -- Auswertung  --  Userinterface  --  Master
                15730 -  2: ( 54, 105 )     99
```

Das System fährt nun den Mikroskop-Kreuztisch zu der gewünschten Position auf dem Film und veranlaßt, daß die in **edit** spezifizierten Filmsegmente abgerastert werden und für jede Rasterposition eine Bildauswertung durchgeführt wird.

Die Auswertung wird mit Hilfe von Bildverarbeitung und Mustererkennung im Mehrprozessorsystem des Systems durchgeführt. Jeder Prozessor bearbeitet ein ganzes Bild und schickt das Auswertungsergebnis an den Manager-Prozessor (Motorola68030), der die Verwaltung übernimmt. Damit wird eine parallele und schnelle Abarbeitung gewährleistet. Wie beim Beispiel der Testauswertung mit **test** zu sehen war, dauerte die Bearbeitung eines Bildes etwa 5 Sekunden. Durch die gleichzeitige Benutzung von z.B. 8 Prozessoren würde im Durchsatz bei der Auswertung die benötigte Zeit für ein Bild etwa 0.6 Sekunden betragen. Das bedeutet, daß mit einem Takt von ca. 0.6 Sekunden Bilder vom Manager eingezogen und an die Prozessoren weitergereicht werden. Im gleichen Takt kommen die Ergebnisse zurück. Zu beachten ist, daß für ein Bild, das zur Bearbeitung abgeschickt wird, selbstverständlich erst etwa 5 Sekunden später das Ergebnis zurückkommt, so daß ein gewisser Vorlauf entsteht. Bei Verwendung von mehr Prozessoren wird der Durchsatz entsprechend erhöht. Maximal wurde bisher ein Durchsatz von 2.5 Bildern pro Sekunde erreicht, wobei dann der Flaschenhals der Datenübertragung einer weiteren Performancesteigerung entgegensteht. Schneller geht es dann nur noch durch Datenreduktion mit einer Spot-Messung, bei der nur ein Ausschnitt des Bildes übergeben wird. Hierbei wurde ein Durchsatz von 3.5 Bildern pro Sekunde erreicht.

Wie im obigen Beispiel zu sehen ist, wird die Auswertung mit der Angabe des Datums und der Anzahl der auszuwertenden Bilder gestartet. Es wird angezeigt, daß die Meßergebnis-Datei angelegt wird und daß man sich nun im Master-Teil des Benutzer-Interfaces befindet. Weiterhin wird für jedes bearbeitete Bild in einer Resultat-Zeile die laufende Nummer des Bildes und die Nummer des Prozessors, der das Bild bearbeitete, angegeben. In der Klammer stehen die x- und y-Indizes des Matrix-Elementes, in welches das Ergebnis der Auswertung dieses Bildes abgelegt wird. Das Ergebnis dieser Auswertung ist die Anzahl der in dem Bild detektierten Löcher und wird hinter der Klammer angezeigt. Die Angaben aus obigem Beispiel: Auf dem Bild mit der Nummer 15730, das von Prozessor 2 bearbeitet wurde, konnten 99 Ioneneinschläge detektiert werden. Das Ergebnis (die Zahl 99) wird an der Matrixposition x/y = 54/105 einer quadratischen Matrix mit 256*256 Elementen eingetragen. Am Ende der Messung (oder nach gewünschtem Abbruch) wird diese Matrix in der Ergebnisdatei mit dem Namen der in **parms** angegeben wurde mit der Extension *.L0* abgespeichert.

IODA_IF bleibt nun solange in diesem Zustand und zeigt für jedes bearbeitete Bild der entsprechenden Filmposition die oben angegebenen Daten an (Result), bis die Auswertung fertiggestellt ist, oder bis die Tastenkombination <CRTL>-C betätigt wird. Die Messung wird damit nicht abgebrochen, lediglich die Anzeige der Result-Zeile wird beendet. Die Kontrolle des Be-

diener-Terminals wird an den Kommando-Modus des Userinterfaces übergeben. Angezeigt wird dies mit dem Prompt "*if*>". Damit sind einige Kommandos zur Kontrolle oder zum Abbruch der Auswertung möglich. Genaueres ist im Kapitel IODA_IF beschrieben.

Die weiteren Kommandos des IODA-Hauptmenues **parms**, **tisch**, **kamera** und **methode** spielen eine etwas andere Rolle, als die bisher beschriebenen Befehle. Sie verzweigen in Untermenues, die im folgenden ausführlich beschrieben sind, **parms** für die Spezifikation und Bearbeitung der Parameter für die Auswertung, **tisch** für die Bedienung des Mikroskop-Kreuztischs und Erzeugung der Fahrmaske für die Auswertung, **kamera** für die Einstellung der Parameter der Bildakquisition und der Darstellung auf dem Display und **methode** für die Definition der Auswerte-Methoden.

Mit **edit** wird ein weiteres Programm (IODA_ED) gestartet, mit dessen Hilfe die Segmente auf dem Film festgelegt werden, für die eine Auswertung durchgeführt werden soll. **IODA** bleibt dabei im Hintergrund und wird wieder aktiviert, sobald der Editor verlassen wird.

5.2.1 Menue PARMS

Das Menue **parms** dient dazu, einige Parameter zu definieren, die das System benötigt, um eine Auswertung ordnungsgemäß durchführen zu können. In das Menue kommt man durch Eingabe von **parms** im IODA-Hauptmenue oder einem Untermenue. Daß man sich im Untermenue befindet, wird durch den Prompt *parms*> angezeigt. Man erhält auch hier durch Eingabe des Fragezeichens "**?**" eine Übersicht über die in dem Menue möglichen Kommandos.

Beispiel:

```
ioda> parms
parms> ?
    !          Ausgabe der momentanen Einstellung ( Menue )
    schuss     Nummer des Kalif-Schusses setzen
    name       Name der Auswertung festlegen
    vergr      Eingestellte Mikroskop-Vergroesserung angeben
    raster     Rasterweite definieren ( in 10 nm  )
    dim        Dimension des Ergebnisses vorgeben
    scan       Scan-Richtung  horizontal oder vertikal
    spot       Bild-Ausschnitt: ganz, viertel, achtel
    bereich    Name der Bereichsdatei fuer Maske definieren
parms>
```

Durch Betätigen der <RETURN>-Taste nach dem Prompt wird, wie bei allen Untermenues, wieder in das IODA-Hauptmenue zurückgekehrt.

Im folgenden Beispiel ist für die verschiedenen Befehle die jeweilige Syntax aufgeführt.

Beispiel:

```
parms>??
    schuss     <nummer>
    name       <datei>|!
    vergr      <vergr.>
    raster     <weite>
    dim        64|128|256|512
```

```
    scan        h|v
    spot        0|1|2
    bereich     [<datei>|!]
parms>
```

Es kann eine Identifikationsnummer für die spezielle Auswertung mit **schuss** definiert werden (wird in der Regel die Schuß-Nummer sein). Der Wert muß numerisch, positiv und kleiner als 10000 sein.

Mit **name** wird definiert, wie die bei der Auswertung erzeugten Dateien heißen sollen. Die Dateien bekommen als Rumpf den spezifizierten Namen und je nach Eigenschaft eine spezielle Extension. Die erzeugten Files sind: für das Archivieren der Meßergebnisse *name.L0*, für das Protokoll und eine Meßparameter-Beschreibung *name.prot* und *name.prt* und für die Sicherung der Parameter das nur von IODA lesbare File *name.save*. Ein gültiger Dateiname ist bei einer neuen Auswertung ein Name der im Directory noch nicht existiert. Bei der Wiederaufnahme einer Auswertung, muß als Dateiname derjenige der abgebrochenen Messung angegeben werden. Die Überprüfung findet erst beim Start der Auswertung statt, die im Fehlerfall mit einer entsprechenden Meldung abgebrochen wird.

Mit **vergr** wird die Mikroskop-Gesamtvergrößerung (Objektiv*Okular) angegeben. Das System bestimmt dann daraus die Bildweite (Größe der Bildkante des akquirierten Video-Bildes) in Einheiten von 10nm (Nano-Meter). Es wird dabei vorausgesetzt, daß vor der Videokamera ein Okular mit der Vergrößerung 10x benutzt wird. Aufgrund der technischen Gegebenheiten ist diese Bildweite nur ein Ausschnitt dessen, was ein Betrachter mit dem Auge durch das Binokular des Mikroskops sieht.

Die Rastergröße wird mit dem Kommando **raster** festgelegt. Die Schrittweite **weite**, mit der abgerastert werden soll, muß in 10nm angegeben werden, Diese Maßeinheit hat praktische Hintergründe. Sie erlaubt, den gesamten relevanten Größenbereich als ganze Zahl in einer Integer-Variablen abzuspeichern. Die kleinste Schrittweite, die mit dem Mikroskop-Kreuztisch gefahren werden kann, beträgt 250nm (1/4µm). Der Film wird in x- und in y-Richtung mit der gleichen Schrittweite abgerastert.

Bei der Auswertung eines Filmes wird das Ergebnis in den Elementen einer quadratischen Matrix abgelegt. Hierbei sind nur feste Ergebnis-Matrix-Dimensionen zugelassen. Außer der defaultmäßig eingestellten Dimension von **256** kann mit dem Kommando **dim** die Größe in den Dimensionen **64**, **128** oder **512** gewählt werden. Bei der Erzeugung der Fahrmaske im Untermenue **tisch** wird überprüft, ob die Größe des abzurasternden Filmsegmentes und der hier angegebenen Rasterweite zu der Dimension der quadratischen Ergebnismatrix paßt. Eventuell wird dort eine neue Rastergröße berechnet und vorgeschlagen.

Mit dem Kommando **scan** kann bestimmt werden, ob die Step-Richtung, in der die Bilder nacheinander eingezogen werden, horizontal oder vertikal sein soll. Um die Tischfahrzeit zu optimieren, wird bidirektional gefahren, standard ist horizontales Scannen. In diesem Fall entspricht die temporäre Ergebnis-Matrix dem Resultat, im anderen Fall sind x und y vertauscht und werden bei Auswertungsende umgespeichert.

Es gibt die Möglichkeit, bei der Auswertung entweder das gesamte Bildfeld der Kamera zu bearbeiten oder nur Ausschnitte um den Bildmittelpunkt mit den Größen 1/2x*1/2y oder aber 1/4x*1/4y. Die entsprechenden Einstellungen werden mit **spot 0**, **spot 1** oder **spot 2** angegeben.

Die Angaben, welche Segmente des Filmes, zusammengefaßt zu einem Bereich, ausgewertet werden sollen, werden dem System in einer Datei in Form von Polygonen mit dem Befehl **bereich <dateiname>** bekanntgegeben. Diese Segment-Polygone werden mit einem Editor, der mit dem Befehl **edit** aus dem IODA-Hauptmenue gestartet wird, im Zusammenspiel mit dem Mikroskop erstellt.

Mit dem Ausrufezeichen "**!**" wird die aktuelle Einstellung der Parameter dieses Menues angezeigt. Es gibt in dieser Anzeige Parameter, die mit den oben gezeigten Kommandos eingestellt werden können und solche, die aus eingestellten berechnet werden.

Das folgende Beispiel zeigt die Default-Einstellung, nachdem IODA frisch gestartet wurde.

Beispiel:

```
parms> !
   ========================= parms ========================
      schuss :    0;  - film_id :
      name :
      vergroesserung:   100;   - Bildweite:  28500,  28500 *10nm
      raster :  20000,  20000  *10nm
      dimension : 256, 256 ;    - Anzahl der Layer: 1
      scan horizontal
      spot : 0
      bereich :
   ========================= parms ========================
parms>
```

Man hat nun die Möglichkeit, nach oben beschriebenem Muster die Parameter einzeln einzugeben, oder die Einstellungen einer vorangegangenen Auswertung, die ja in einem *.save*-File gesichert wurde, mit dem Befehl **load** aus dem IODA-Hauptmenue zu laden und einige Parameter zu ändern.

Beispiel:

```
parms> load testlage6
parms> !
   ========================= parms ========================
      schuss :  3357;  - film_id : 3357
      name : /h0/INR/IODA/3357/testlage6
      vergroesserung:   100;   - Bildweite:  14250,  14250 *10nm
      raster :  25000,  25000  *10nm
      dimension : 256, 256 ;    - Anzahl der Layer: 1
      scan horizontal
      spot : 1
      bereich : /h0/INR/IODA/3357/polygonlage6
   Maske noch nicht erstellt
   ========================= parms ========================
parms>
```

Wie man sieht, wurden die Einstellungen der Auswertung "testlage6" in das System übernommen, die nun zum Teil geändert werden müssen; zumindest der Name des Auswertungsergebnisses muß geändert werden, wenn im Directory bereits eine gleichnamige Auswertung vorhanden ist.

Üblicherweise erzeugt man für jede Auswertung, zumindest aber für jeden Film ein eigenes Directory. Wird nun aus einem benachbarten Directory mit **load** das *.save*-File geladen oder mit **dir** das Directory gewechselt, so steht anschließend im Pfad sowohl bei **name** als auch bei **bereich** noch der Pfadname des anderen Directory. Möchte man die Dateinamen im neuen Directory beibehalten, so besteht die Möglichkeit, mit **name !** bzw. **bereich !** den Pfad auf das aktuelle Directory umzusetzen.

Beispiel:

```
parms> dir ../3358
parms> load ../3357/testlage6
parms> !
   ========================= parms ========================
      schuss :  3357;  - film_id : 3357
      name : /h0/INR/IODA/3357/testlage6
      vergroesserung:  100;  - Bildweite:  14250,  14250 *10nm
      raster :  25000,  25000  *10nm
      dimension : 256, 256 ;    - Anzahl der Layer: 1
      scan horizontal
      spot : 1
      bereich : /h0/INR/IODA/3357/polygonlage6
   Maske noch nicht erstellt
   ========================= parms ========================
parms>sch 3358
parms>parms> name !
parms>bereich !
   ========================= parms ========================
      schuss :  3358;  - film_id : 3357
      name : /h0/INR/IODA/3358/testlage6
      vergroesserung:  100;  - Bildweite:  14250,  14250 *10nm
      raster :  25000,  25000  *10nm
      dimension : 256, 256 ;    - Anzahl der Layer: 1
      scan horizontal
      spot : 1
      bereich : /h0/INR/IODA/3358/polygonlage6
   Maske noch nicht erstellt
   ========================= parms ========================
parms>
```

Weiterhin fällt auf, daß diese vorangegangene Auswertung offensichtlich mit **spot 1** durchgeführt wurde. Diese Spot-Messung hat selbstverständlich Auswirkungen auf die bearbeitete Bildfeldgröße, die im obigen Beispiel nun entsprechend halbiert angegeben wird.

Im Status wird auch angegeben, ob eine gültige Fahrmaske erstellt wurde. Eine Fahrmaske muß mit **maske** im Menue **tisch** erzeugt werden (kann auch aus dem parms-Menue mit dem Befehl **tisch maske** geschehen). Wird nach Erstellung einer gültigen Maske ein relevanter Parameter neu eingegeben (Raster, Dimension oder Name der Bereichs-Datei), so wird die Maske ungültig und das System meldet bei der nächsten Statusanzeige, daß keine gültige Maske vorhanden ist.

5.2.2 Menue TISCH

Um innerhalb von IODA die Möglichkeit zu schaffen, den Mikroskop-Kreuztisch zu bedienen, und zur Erstellung der Fahrmaske, ist das Untermenue **tisch** vorgesehen. Das Untermenue wird vom IODA-Hauptmenue oder von einem anderen Untermenue aus durch Eingabe des Befehls **tisch** aktiviert. Dies wird - wie bei den übrigen Menues - durch den entsprechenden Prompt *tisch>* angezeigt.

Mit dem Fragezeichen **?** erhält man auch hier wieder eine Liste der in dem Untermenue enthaltenen Kommandos.

Beispiel:

```
tisch> ?
   !          Ausgabe der momentanen Einstellung ( Menue )
   position   Positionierung des Tisches ( x y z )
   index      Positionierung innerhalb der Maske ( x y )
   align      Positionierung auf naechstes Raster
   set        speichere Position
   go         fahre gespeicherte Position an
   label      Label fuer Speicher
   clear      Speicher loeschen
   objektiv   Objektiv wechseln
   speed      Setzen der Fahrgeschwindigkeit
   maske      Bereich laden und Fahrmaske erstellen
   show       Fahrmaske auf dem Display zeigen
   write      Fahrmaske als Datei ausgeben
   reset      Mikroskop zuruecksetzen / kalibrieren
tisch>
```

Die Syntax für jeden Befehl zeigt das folgende Beispiel.

Beispiel:

```
tisch> ??
                position [<x> [<y> [<z>]]]
                index [<xi> <yi>]
                align
                set <memo_nr>
                go [<memo_nr>|+]
                label [<memo_nr> [<string>]..]
                clear [<from> [<to>]]
                objektiv [0|<vergr.>]
                speed [<tisch_speed>]
                maske [all]
                show
                write <datei>
                reset !
tisch>
```

Das Ausrufezeichen **!** als Argument bewirkt das Anzeigen der momentanen Einstellung der Parameter dieses Menues. Man erhält Auskunft über die x-, y- und z-Koordinaten der aktuellen

Kreuztisch-Position, über das eingestellte Mikroskop-Objektiv und darüber, ob eine gültige Fahrmaske existiert. Außerdem wird gegebenenfalls noch angegeben, welche Positionen im Koordinaten-Speicher stehen.

Beispiel:

```
tisch> !
========================== tisch ==========================
  ( 6605475, 7739975,  91042 )
Objektiv: 10 x
Speed:  500000
Maske noch nicht erstellt

  Tisch-Speicher:
   1   6605475  7739975   91042
   2   6780000  7866450   91042

========================== tisch ==========================
tisch>
```

Das Menue enthält einen Koordinaten-Speicher für 20 Tischpositionen, die mit der Speichernummer adressiert werden können. Der Speicher wird beim **save** mit abgespeichert.

Mit dem Befehl **set <nr>** wird die Position, an der der Kreuztisch gerade steht, in den Koordinatenspeicher mit der Nummer **<nr>** gepackt. Mit **label <nr> <String>** kann dem Speicher der Nummer **<nr>** ein Character-String zugeordnet werden, der dann bei einer Status-Anzeige mit ausgegeben wird. Der Befehl **clear** mit einer Nummer als Argument löscht diesen Speicherinhalt, **clear** mit zwei Nummern **<nr1>** und **<nr2>** löscht die Speicher von **<nr1>** bis **<nr2>**.

Beispiel:

```
tisch> label 2 zweite Position
tisch> !
========================== tisch ==========================
  ( 6605475, 7739975,  91042 )
Objektiv: 10 x
Speed:  500000
Maske noch nicht erstellt

  Tisch-Speicher:
   1   6605475  7739975   91042
   2   6780000  7866450   91042  zweite Position

========================== tisch ==========================
tisch>
```

Mit dem Befehl **go <nr>** wird veranlaßt, daß der Kreuztisch an die im Speicher **<nr>** stehenden Koordinaten fährt. Das Argument **+** anstatt der Nummer bewirkt, daß der nächste Speicherinhalt angefahren wird. Ist der letzte Speicher erreicht, wird wieder zum ersten gefahren. Ohne Argument gibt **go** die letzte angefahrene Position zurück.

Die wichtigste Aufgabe dieses Menues ist die Erstellung der Fahrmaske für die Auswertung.

Um dem System die Information zu geben, welche Positionen auf dem Film angefahren und ausgewertet werden sollen, muß für die in **edit** definierten Bereiche eine Fahrmaske erzeugt werden. Die mit **edit** in der Bereichs-Datei (siehe **parms** und **edit**) abgelegten Polygone definieren die Berandung der Segmente, die mit einer bestimmten Rasterweite (siehe **raster** in **parms**) abgescannt werden. Entsprechend der Auswerte-Matrix wird ein Feld als Maske angelegt, das an jeder Rasterposition, die ausgewertet werden soll, einen Wert ungleich 0 enthält. Entsprechend den verschiedenen Segmenten auf dem Film erhalten die Segmente der Fahrmaske als Wert die im Programm **edit** zugeordneten Label. Diesen verschiedenen Labeln kann wiederum im Untermenue **methode** verschiedene Auswertemethoden zugeordnet werden.

Diese Fahrmaske wird mittels **maske** erzeugt. Die minimale rechteckige Fläche, die die Extrempunkte aller definierten Segmente enthält, wird dabei links oben in die Auswerte-Maske positioniert. Sollen auch die im **edit**-Untermenue **bereich** definierten Justierungspunkte berücksichtigt werden, die ja außerhalb dieser Fläche liegen können, muß der Parameter **all** angegeben werden.

Das Kommando **show** stellt die Fahrmaske auf dem Display dar, wobei die einzelnen Segmente entsprechend ihrer unterschiedlichen Label verschiedene Farben zugeordnet werden. Außerdem wird mit **show** für jedes Segment die Anzahl der enthaltenen Rasterpunkte und damit die Zahl der auszuwertenden Mikroskopbilder bestimmt. Aus dieser Anzahl läßt sich die minmale Dauer - schneller geht es prinzipiell nicht - der Auswertung (in h:min:sec) abschätzen.

Beispiel:

```
tisch> maske
    Bereich 'polygonlage6' geladen
    Maske erstellt
tisch> show

        ========== Segmente ================

        rot     4 Ecken:  Label  1     24460 Bilder

        insgesamt 24460 Bilder
        minimale Auswerte-Zeit:  3:03:27.00  bzw. 1:54:08.80 bei Spot 1
tisch>
```

Mit dem Kommando **position** wird, wenn kein Argument mitgegeben wird, die augenblickliche Position angegeben. Ist eine Fahrmaske erstellt, so wird gleichzeitig der dieser Position am nächsten liegende Index der Fahrmaske mit angegeben. Dasselbe geschieht dadurch, daß der Befehl **index** ohne Argument eingegeben wird. Besteht der Wunsch, die Position anzufahren, die exakt dem angegebenen Index entspricht, so kann dies durch **align** geschehen. Beim Ausgeben der Position wird auch mit angegeben, ob die Position einem Index "aligned" ist oder nicht.

Beispiel:

```
tisch> pos
        ( 8035025, 5302150,   91042 ) - Index ( 54, 103 )
tisch> index
        ( 8035025, 5302150,   91042 ) - Index ( 54, 103 )
tisch> align
```

```
tisch> index
     ( 8030475, 5289975,  91042 ) - Index ( 54, 103 ) aligned
tisch>
```

Die Kommandos **position** und **index** mit Argumenten positionieren den Kreuztisch an die gewünschte Position, wobei zu **index** beide Index-Werte anzugeben sind; **index** kann aus verständlichen Gründen nur bei Vorhandensein einer gültigen Fahrmaske benutzt werden. Die Argumente zu **position** sind die x-, y- und z-Koordinaten (in dieser Reihenfolge). Soll eine Richtung nicht gefahren werden, kann als Koordinate -1 angegeben werden, oder sie ganz weggelassen werden, falls es sich um die jeweils letzte handelt.

objektiv ohne Argument liefert die Vergrößerung des am Mikroskop eingestellten Objektivs zurück. Durch die Angabe eines Wertes nach **objektiv** wird das entsprechende Objektiv am Revolver in den Strahlengang geschaltet, falls eine gültige Vergrößerung angegeben wurde. Mögliche Werte sind: **5, 10, 20, 50** und **100**. Eine Sonderfunktion hat der Wert **0**; durch ihn wird die Revolveransteuerung vom Rechner freigegeben, so daß man die Objektive durch Betätigen der entsprechenden Tasten an der linken Handauflage umschalten kann.

Beispiel:

```
tisch> ob
   objektiv = 10
tisch>
```

Durch den Befehl **speed** läßt sich die maximale Tischfahrgeschwindigkeit einstellen, bzw. die aktuell eingestellte abrufen. Der Wert, der ausgegeben bzw. eingegeben wird, ist die Geschwindigkeit für eine Richtung in Tischeinheiten pro Sekunde; eine Tischeinheit beträgt 10nm. Dies gilt aus technischen Gründen nur für die x- und y-Richtung. Die Fahrgeschwindigkeit in z-Richtung ist in jedem Fall geringer. Die minimale Angabe für **speed** ist 500, die maximale 500000, wobei nur bestimmte Einstellungen möglich sind; bei Eingabe einer Zwischengröße wird automatisch auf den nächst niedrigen Wert geschaltet.

Beispiel:

```
tisch> speed
   speed = 500000
tisch>
```

Mit **write <dateiname>** hat man die Möglichkeit, die mit **maske** erstellte Fahrmaske als Bildmatrix (Pixelbild, 1Byte pro Pixel) in die Datei **<dateiname>** zu schreiben.

Falls die Tischsteuerung abgeschaltet war, oder mit dem Reset-Knopf zurückgesetzt wurde, muß das Mikroskop 'kalibriert' werden. Dabei werden alle Achsen an ihre Endschalter gefahren und die Position-Register zurückgesetzt. Für die Kalibrierung sollte der Tisch soweit abgesenkt werden, bis die Objektive frei stehen und nicht mehr kollidieren können. Die Spannungsversorgung für den Objektivwechsel sollte eingeschaltet sein. Dann wird mit **reset !** die Kalibrierung gestartet. Sie kann bis zu 5 min. dauern, und sollte nicht abgebrochen werden.

5.2.3 Menue KAMERA

Im Untermenue **kamera** werden Einstellungen der Parameter zur Bildakquisition mit der Videokamera und zur Bilddarstellung auf dem Display vorgenommen. In das Menue kommt man durch Eingabe von **kamera** oder einer geeigneten Abkürzung aus dem IODA-Hauptmenue oder einem anderen Untermenue; der entsprechende Prompt ist *kamera>*. Mit dem Fragezei-

chen ? wird die Liste der möglichen Kommandos des Menues angezeigt.

Beispiel:

```
kamera> ?
    !          Ausgabe der momentanen Einstellung ( Menue )
    focus      Focus automatisch einstellen
    reset      Video-Interface zuruecksetzen
    live       Kamera live schalten
    shot       Bild einziehen - live beenden
    color      Lookup-table farbig
    grau       Lookup-table grau
    pixel      Setzen der Farbe fuer einen Pixelwert
    keil       Graukeil einblenden
    histo      Bild einziehen und mit Histogramm zeigen
    gain       Verstaerkung des Wandlers einstellen
    offset     Offset des Wandlers einstellen
kamera>
```

Das Ausrufezeichen ! veranlaßt die Anzeige der Parameter des Analog/Digital-Wandlers Gain für die Verstärkung und Offset für die Verschiebung des Videosignals vor der Konvertierung und der Information, ob die Bildakquisition live oder gestoppt ist.

Beispiel:

```
kamera> !
    ======================== kamera ========================

      Gain: 127;    Offset: 128
      Acquisition gestoppt

    ======================== kamera ========================
kamera>
```

Die Syntax der möglichen Befehle wird im folgenden Beispiel aufgelistet.

Beispiel:

```
kamera> ??
                focus
                reset
                live
                shot
                color [<faktor>]
                grau
                pixel <value> [ <r> [ <g> [ <b> ]]]
                keil
                histo
                gain [<gain>]
                offset [<offset>]
kamera>
```

Durch den Befehl **focus** wird eine automatische Fokussierung des Mikroskops veranlaßt. Da-

mit ist die Aufnahme eines Bildes mit gutem Kontrast möglich. Dieser Autofokus [6] arbeitet auf Bildverarbeitungsbasis und benötigt ein ebenes strukturiertes Objekt im Bildfeld.

Mit der Eingabe des Kommandos **reset** wird das Video-Interface in den Grundzustand versetzt und der Akquisitionsmodus auf live geschaltet.

Der Befehl **live** schaltet die Bildakquisition in den live-Modus; das auf dem Display gezeigte Bild ist ein live-Bild der Videokamera. Dieses Kommando ist notwendig nach einer **test**-Auswertung, da dann auf dem Display das Testergebnis dargestellt wird.

Mit **shot** kann ein live-Bild auf dem Display eingefroren werden.

Durch den Befehl **color** [<**faktor**>] werden in Abhängigkeit vom Wert **faktor** Look-Up-Tabellen berechnet und gesetzt, die die Pixelwerte zwischen 0 und 255 auf Regenbogenfarben abbilden. Der Wert von **faktor** bestimmt die "Anzahl der Regenbögen", die zwischen den Pixelwerten 0 und 255 untergebracht werden sollen.

Mit **grau** wird eine Look-Up-Tabelle besetzt, die den Rot-, Grün- und Blau-Anteilen jeweils gleiche Werte zuordnet, so daß auf dem Display ein Graubild angezeigt wird. Der Pixelwert 0 wird schwarz und der Pixelwert 255 weiß dargestellt.

Durch das Kommando **pixel <value>** [<r>[<g>[<b>]]] kann dem Pixelwert der Größe **value** durch geeignetes Mischen der Grundfarben Rot, Grün und Blau eine individuelle Farbe zugeordnet werden. Wird nur Rot benötigt, müssen die Werte der beiden anderen Mischungsanteile nicht angegeben werden, wird jedoch z.B. nur Blau benötigt, so müssen die Werte für Rot und Grün (jeweils als 0) mit angegeben werden. Das Kommando **pixel 254 0 0 255** ordnet dem Pixelwert **254** eine hellblaue Farbe zu.

Der Befehl **keil** erstellt am unteren Rand des Displays im Bild einen Graukeil, der links mit 0 startet und rechts mit dem Wert 255 endet. Es ergibt sich dadurch die Gelegenheit, die Auswirkungen verschiedener Look-Up-Tabellen sichtbar zu machen. Damit wird allerdings das dargestellte Bild eingefroren, und man muß bei Bedarf den live-Modus durch Eingabe von **live** wiederherstellen.

Durch das Kommando **histo** wird veranlaßt, daß das live-Bild eingefroren, das Histogramm (Häufigkeitsverteilung der Grauwerte) dieses Bildes berechnet und auf dem Display überlagert dargestellt wird. Man hat damit die Möglichkeit, sich eine Information über die Dynamik des Videobildes zu verschaffen und gegebenenfalls durch Verändern der Mikroskop-Beleuchtung oder der Parameter **gain** und **offset** auf die Dynamik einzuwirken.

Wird **gain** oder **offset** ohne Argument eingegeben, so wird der jeweils eingestellte Wert ausgegeben. Die Werte der beiden Parameter, die zur Einstellung übergeben werden können, müssen zwischen 1 und 255 liegen. Die Größe **gain** wird zur Verstärkung und **offset** zum Verschieben des Videosignals bei der Analog/Digital-Konvertierung benutzt. Damit die Auswirkungen einer Parameteränderung auf dem Display kontrolliert werden können, sollte die Akquisition auf **live** geschaltet sein.

5.2.4 Menue METHODE

Im Untermenue **methode** wird das für die Auswertung eines Filmsegments benötigte Verfahren ausgewählt und eingerichtet und dem auszuwertenden Segment zugeordnet.

In das Menue kommt man durch Eingabe von **methode** im IODA-Hauptmenue oder einem anderen Untermenue; der entsprechende Prompt ist *methode*>. Mit dem Fragezeichen **?** wird die Liste der möglichen Kommandos des Menues angezeigt.

Beispiel:

```
ioda> methode
methode> ?
    !          Ausgabe der momentanen Einstellung ( Menue )
    mhelp      Einzelheiten der aktuellen Methode ( help )
    select     Selection einer zugeordneten Methode
    new        Selection einer neuen Methode
    connect    Zuordnen der aktuellen Methode zum Segment
    check      Ueberpruefen der Zuordnungen aller Segmente
    set        setzen der Parameter
methode>
```

Durch Eingabe des Ausrufezeichens ! werden die relevanten Einstellungen der aktuellen Methode angezeigt, im folgenden Beispiel die der Sigma-Methode.

Beispiel:

```
methode> !
   ======================== methode ========================
      ---- Sigma ----
      dimension: 256
      faktor:   4.00
      maske:    #7
   ======================== methode ========================
methode>
```

Die Bedeutung der Parameter wird später erläutert. Im folgenden Beispiel ist die Syntax der möglichen Befehle aufgelistet, die man erhält, wenn zum jeweiligen Befehlsnamen das Fragezeichen ? als Argument mitgegeben wird.

Beispiel:

```
methode> ??
    mhelp      [<begriff>] ! !
    select     [<label_nr>]
    new        [<methode_nr>]
    connect    [<label_nr>]
    check
    set        [<name> [<value>]]
methode>
```

Dem Menue wird immer eine Methode, die 'aktuelle Methode', zugeordnet. Sie wird aus den möglichen Methoden ausgewählt, bearbeitet und anschließend einem Segment zugewiesen. Die 'aktuelle Methode' wird auch, wenn nicht anders spezifiziert bei **test** verwendet. Soll sie zur Auswertung eines Segments benutzt werden, muß sie unbedingt mit **connect** mit dem Segment-Label verbunden werden.

Implementiert sind folgende Methoden:

Histo-max und Histo-min

Histo-max und Histo-Min sind Verfahren, die aus dem Histogramm eines Mikroskop-Bildes

mit statistischen Verfahren eine Schwelle berechnen. Dann wird so segmentiert, daß bei Histomax alle Pixel oberhalb der Schwelle, bei Histo-min alle Pixel unterhalb der Schwelle als zu einem Loch gehörend klassifiziert werden. Zusammenhängende Pixel werden dann als ein Loch gewertet entsprechend gezählt.

Diese Verfahren können nur bei sehr einfachen Bildern angewandt werden, bei denen die zu zählenden Löcher sich in ihrer Helligkeit deutlich vom Hintergrund abheben.

Sigma

Die Sigma-Methode ist die wegen ihrer Robustheit wohl am häufigsten ausgewählten Methode. Beim Sigma-Verfahren wird zur Klassifizierung eines Pixels (zu einem Loch gehörend oder nicht) seine Umgebung benutzt. Aus den Grauwerten der Umgebung wird ein Mittelwert und die entsprechende Standardabweichung σ berechnet. Das Pixel wird dann als zu einem Loch gehörend klassifiziert, wenn sein Grauwert größer ist als der Wert, den man erhält, wenn man den Mittelwert der Umgebung um das Produkt aus seiner Standardabweichung σ und **faktor** erhöht. Die zur Mittelwert- und Standardabweichungs-Berechnung herangezogene Umgebung kann frei definiert werden. Dadurch besteht die Möglichkeit, die individuelle Erscheinung der Löcher in die Klassifikation einzubeziehen und unerwünschte zu diskriminieren. Der als Multiplikator der Standardabweichung σ dienenden **faktor** und die **maske**, mit der die Umgebung definiert wird, sind Parameter dieses Verfahrens und müssen durch Tests ermittelt werden.

Kreis

Kreis ist eine Methode, die ausnutzt, daß die Grauwerte im Zentrum eines Loches Ähnlichkeit mit den z-Werten eines Kegels haben. Berechnet man mit dem Sobel-Bildverarbeitungs-Algorithmus die Grauwert-Gradienten, so ist im Inneren eines Loches eine charakteristische kreisförmige Ausrichtung der Gradientenrichtungen zu beobachten. Diese kreisrunde Form wird ausgenutzt, um mit einfacher Mustererkennung die Löcher auf dem Bild zu klassifizieren.

Scan

Die Scan-Methode wurde nicht zur Auszählung von Löchern auf CR-39-Filmen, sondern zur Gradationsuntersuchung von beliebigen Filmen entwickelt. Mit dieser Methode sind nur qualitative Aussagen über die Einschlagdichte der Teilchen möglich, die durch die Berechnung verschiedener Merkmale zustandekommen.

Die 5 Merkmale werden aus dem für das jeweilige Bild erstellten Histogramm (Grauwerthäufigkeit) berechnet. Bei den Bilder wird davon ausgegangen, daß zwei Klassen von Grauwerten existieren, die durch einen Schwellwert voneinander getrennt werden können. Das erste Merkmal ist dementsprechend dieser Schwellwert. Als zweites Merkmal wird der prozentuale integrale Anteil der Grauwertklasse unterhalb der Schwelle benutzt und mit 255 skaliert. Das dritte Merkmal ist der globale Mittelwert aller Grauwerte im Bild und das vierte der Mittelwert der Grauwerte unterhalb der Schwelle. Als fünftes Merkmal wird die globale Varianz aller Grauwerte des Video-Bildes berechnet. Die berechneten Daten [5] werden in 5 verschiedenen Layern abgelegt (Anzahl der Layer ist in parms angezeigt), die wiederum in den entsprechenden Dateien *name.L0, name.L1, name.L2, name.L3* und *name.L4* gespeichert werden.

Die implementierten Methoden lassen sich mit **new** anzeigen und auswählen:

Beispiel:

```
methode> new
        Definierbare Verfahren :
          0  --          nicht auswerten
          1  Histo-max   Segmentierung oberhalb Schwelle
          2  Histo-min   Segmentierung unterhalb Schwelle
          3  Sigma       Klassifizierung anhand der Umgebung
          4  Kreis       Kreis-Detektion mit Sobel-Richtung
          5  Scan        einfaches Histogramm-Scanning

methode>
```

Mit **new <methode_nr>** wird die 'aktuelle Methode' ausgewählt.

Soll ein Segment nicht ausgewertet werden, so mit **new 0** die Nullmethode auszuwählen und mit dem Label zu verbinden.

Beispiel:

```
methode> new 1
    aktuelle Methode: Histo-max
methode> new 2
    aktuelle Methode: Histo-min
methode> !
    ========================= methode ========================
      ---- Histo-min ----
      dimension: 256
    ========================= methode ========================
methode>
```

Eine bereits einem Segment zugeordnete Methode läßt sich mit **select <label_nr>** wieder zur 'aktuellen Methode' machen. Ändert man Parameter und will die Zuordnung beibehalten, muß sie anschließend wieder **connect**ed werden(!). Alle bereits zugeordneten Methoden werden mit **select** ohne Argument gelistet.

check listet alle definierten Segmente mit ihren Auswerte-Methoden. Hier läßt sich erkennen, ob allen Segmenten Methoden zugeordnet sind.

connect <label_nr> verbindet eine Kopie der 'aktuellen Methode' einschließlich der definierten Parameter mit dem Segment **<label_nr>**. Aus dem Wort Kopie geht hervor, daß nach einer Änderung der Parameter erneut ein **connect** erfolgen muß.

Die Parameter einer Methode werden mit dem Kommando **set** gesetzt. **set** ohne Argument listet alle setzbaren Parameter der 'aktuellen Methode' und ihre Kurzbeschreibung. Zum Setzen muß nach **set** der Parameter-Name (eindeutig abgekürzt) und der Wert angegeben werden. Die Parameter werden im folgenden in Abhängigkeit von der Methode diskutiert:

Alle Methoden kennen den Parameter **dimension**. Er kann den Wert **256** oder **512** annehmen und beschreibt die Dimension des zu akquirierenden Bildes. Im Normalfall wird 256 verwendet. Bei notwendig höherer Auflösung kann 512 verwendet werden, die Rechenzeit wird dabei signifikant erhöht.

Das Sigma-Verfahren kennt, wie oben erwähnt, zwei weitere Parameter, **faktor** und **maske**.

Die Werte sind vor dem Start der automatischen Auswertung mit Hilfe von **test** zu ermitteln; der **faktor**-Wert liegt üblicherweise zwischen 3.0 und 6.0. Zur Festlegung der Umgebung gibt es zwei Möglichkeiten. Man benutzt eine der im System voreingestellten Masken oder man definiert sich eine Maske selbst und legt sie in einer Datei ab. Bei der Kreation einer eigenen Maske ist lediglich darauf zu achten, daß als die Dimension ein ungeradzahliger Wert benutzt wird. An den Positionen der Nachbarschafts-Pixel, die zur Mittelwert- und σ-Berechnung beitragen sollen muß eine 1 zu stehen, sonst Null. Der Methode wird dann auf folgende Weise der Name der Datei, in der die Maske abgelegt wurde übergeben: **set maske <dateiname>**. Die Datei liegt im aktuellen Arbeitsverzeichnis. Im folgenden wird beispielhaft das Design einer Maske gezeigt.

Beispiel:

```
maske 7 7
0 0 0 1 1 1 1
0 0 0 0 1 1 1
0 0 0 0 0 1 1
0 0 0 0 0 0 1
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
```

In der ersten Zeile muß das Keywort maske und die x- und y-Dimension (quadratisch) durch Leerzeichen getrennt stehen, in den folgenden Zeilen die Maskenwerte.

Mit der hier als Beispiel kreierten Maske kann mittels Sigma-Methode jedes Pixel eines Bildes markiert werden, dessen Nord-Ost-Nachbarschaft einen signifikanten dunklen Fleck aufweist.

Die vom System voreingestellte Masken sind numeriert und werden der Methode mit **set maske #<nr>** übergeben. Mit **mhelp #** kann man sie sich anzeigen lassen.

Beispiel:

```
methode> mhelp #
        ******** S I G M A ******** in IODA - ioda.sigma help #Masken

   folgende Masken sind voreingestellt:

    #2      #3      #5       #6       #7          #9

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

   methode>
```

Bei dem folgenden Beispiel wurde ein σ-Faktor von 4.5 zusammen mit der vordefinierten 5*5-Maske mit dem Namen #5 benutzt um eine Sigma-Methode für eine Auswertung zusammenzustellen

Beispiel:

```
methode> set fak 4.5
methode> set ma #5
```

```
methode> !
   ========================= methode ========================
      ---- Sigma ----
      dimension: 256
      faktor:   4.50
      maske:    #5
   ========================= methode ========================
methode>
```

Im Verzeichnis */h0/IODA/HELP* des VME-Bus-Rechners, auf dem IODA installiert ist, existieren Help-Files, auf die in IODA mit der Anweisung **help** zugegriffen werden kann. Die Dateien können selbstverständlich bei Bedarf erweitert werden. Für **methode** wurde **mhelp** geschaffen, um Hilfen zu bestimmten Begriffen zur Verfügung zu stellen. So können z.B., wie beim Sigma-Verfahren zu sehen ist, die im System vordefinierten Auswerteverfahrens-Masken angezeigt werden.

5.3 Programm IODA_IF

Bei dem Userinterface handelt es sich um einen menuegesteuerten Programmteil, der nur während der Auswertung aktiv sein kann. Er dient zur Darstellung des Status und der Statistik der Auswertung, und läßt im Master-Mode auch die Beeinflussung der Messung zu. Im Master-Mode befindet er sich automatisch, wenn er von IODA gestartet wird. Zusätzlich ist der gleichzeitige Aufruf von jedem beliebigen Terminal aus möglich, solange eine Auswertung aktiv ist. Die Einschränkungen in diesem Fall betreffen Kommandos, die die Auswertung beeinflussen können.

Der Start von einem externen Terminal sieht so aus:

Beispiel:

```
> ioda_if

     - - I O D A - -  Version 1.01 v. 23.11.94 --

     -- Auswertung -- Userinterface --

if> ?
```

Wird der Versuch gemacht, das Userinterface zu starten, wenn keine Auswertung aktiv ist, so wird diese Tatsache auf dem Monitor angezeigt.

Beispiel:

```
> ioda_if
   Ioda ist nicht im Auswerte-Mode !
>
```

Mit dem Kommando **bye** läßt sich das Programm beenden. Es bricht automatisch ab, sobald keine Auswertung mehr läuft.

Wie bei den übrigen IODA-Menues erhält man durch Eingabe des Fragezeichens **?** eine Auflistung der möglichen Kommandos.

Beispiel:

```
if> ?
    !          Ausgabe des momentanen Status
    shell      lokale Shell aufrufen
    dir        Directory anzeigen oder aendern
    stat       momentane Statistik anzeigen
    result     Resultate ansehen / Resultatmodus
    menue      Switch fuer Menue-Mode
    kill       Auswertung abbrechen
    flush      Datei-flush veranlassen
    demo       Switch fuer Demo-Mode
    single     Single-Resultat-Mode
if>
```

Falls das Userinterface von einem anderen Terminal mit **ioda_if** gestartet wird, ist der Befehlsumfang eingeschränkt, da eine Beeinflussung der Messung nicht möglich ist. Die Liste der möglichen Kommandos sieht in diesem Fall so aus:

Beispiel:

```
if> ?
    !          Ausgabe des momentanen Status
    shell      lokale Shell aufrufen
    dir        Directory anzeigen oder aendern
    stat       momentane Statistik anzeigen
    result     Resultate ansehen / Resultatmodus
    menue      Switch fuer Menue-Mode
    bye        Beenden des Programms
if>
```

Die Befehle **shell**, **dir**, **menue** und **demo** haben genau die gleichen Wirkungen wie im IODA-Hauptmenue. Das Einschalten des Demo-Modes mit **demo** erfolgt aufgrund des oben erwähnten Parallelprozessings verzögert. Das bedeutet: das Bild, bei welchem der Modus geschaltet wurde, (und dann alle folgenden) nimmt ein entsprechendes Flag mit zur Verarbeitung. Das entsprechende Ergebnis wird aber erst nach Ablauf der Bearbeitungszeit auf dem Display farbig angezeigt. Ähnlich verhält es sich beim Ausschalten des Demomodes mit **demo,** auch hier erscheint die Wirkung verzögert.

Um den aktuellen Status anzuzeigen, kann das Ausrufezeichen **!** verwendet werden. Man erhält dann Informationen über den Namen der in **parms** spezifizierten Ergebnis-Datei, die seit dem Start der Auswertung verstrichene Zeit (hier: 1 Minute 26 Sekunden), die bearbeiteten Bilder (hier: 267 Stück) und den mittleren Zeitbedarf für ein Bild (im Durchsatz; hier: 0.32 Sekunden pro Bild). Zu Beginn einer Auswertung ist diese Angabe noch ungenau, da es sich um eine statistische Größe handelt, die Genauigkeit steigt mit der Anzahl der Bilder. Aus dem mittleren Zeitbedarf für ein Bild wird dann mit der Kenntnis der für das auszuwertende Filmsegment anfallenden Gesamtbild-Anzahl die benötigte Restzeit hochgerechnet (hier: 2 Stunden, 9 Minuten und 50 Sekunden).

Beispiel:

```
if> !
    - - I O D A - -  Version 1.01  v. 23.11.94 -- IF-Master --
    laufende Auswertung: /h0/INR/IODA/3357/dddtest.L0
    bisherige Laufzeit:  1:26.00
    Jobs gerechnet:      267
    Zeit / Job :         0.32 s
    verbleibende Jobs:   24193,  Rest-Laufzeit ca. 2:09:50.14
if>
```

Beim Start der Auswertung wird, wie oben schon erwähnt, ein File angelegt, in das die Meßergebnisse in Matrixform eingetragen werden. Dieses File ist während der Auswertung als Temporär-File mit dem Namen *T<Monat><Tag><h><min>.L0* vorhanden und wird am Ende der Auswertung in den in **parms** angegebenen Namen *name.L0* umbenannt. Während der Auswertung sind nun alle, zumindest aber ein wesentlicher Teil der Ergebnis-Daten in einem Puffer des Systems zwischengespeichert. Das Kommando **flush** bewirkt, daß dieser Puffer in die Temporär-Datei geleert wird. Somit besteht die Möglichkeit, eine Kopie dieser Datei z.B. auf eine Workstation zu bringen und mit geeigneten Werkzeugen das bis dahin ermittelte Meßergebnis zu visualisieren, ohne die laufende Messung zu unterbrechen. Wie im Beispiel der Aufruf-Syntax zu sehen ist, kann dem System mit **<offset>** mitgeteilt werden, wieviele Resultate noch abgewartet werden sollen, bevor der Puffer geleert wird.

Beispiel:

```
if> flush ?
    Syntax: flush [<offset>]
if> flush 10
    flush nach 10 Resultaten
if>
```

Mit dem Befehl **result** wird dem Interface mitgeteilt, daß man aus dem Kommando-Modus wieder in den Result-Modus wechseln will, bei dem die Meßergebnisse in der oben beschriebenen Art in sogenannten Result-Zeilen auf dem Monitor angezeigt werden. Das System bleibt dann solange im Result-Modus, bis mit der Tastenkombination CRTL-C wieder in den Kommando-Modus mit dem Prompt *if>* gewechselt wird. Es gibt nun die Möglichkeit, dem Kommando ein Argument mitzugeben. Mit der Übergabe eines Wertes für **anzahl** wird die Anzahl Ergebnisse angezeigt und anschließend automatisch wieder in den Kommandomodus umgeschaltet. Mit **result +** wird eine Result-Modus-Option eingeschaltet, d.h. die Betätigung der ENTER- bzw. RETURN-Taste hinter dem *if>*-Prompt bewirkt das sofortige Wechseln in den Result-Modus, den man nur durch die Tastenkombination CRTL-C wieder verlassen kann. Mit **result -** im Kommando-Modus wird diese Option wieder ausgeschaltet.

Beispiel:

```
if> result ?
    Syntax: result [<anzahl>|+|-]
if>
```

Der Befehl **single** ermöglicht es, das System nach der Anzeige einer Ergebniszeile anzuhalten, um sich dieses Ergebnis auf dem Monitor oder dem Display genauer anzusehen. Durch betätigen der RETURN-Taste wird jeweils das nächste anstehende Ergebnis visualisiert. Aufgrund der asynchronen Bearbeitung im Multiprozessor-System ist die Reihenfolge der Ergebnisse

nicht vorhersagbar. Achtung, da das System in diesem Modus nur so schnell arbeitetet, wie der Benutzer quittiert, geraten alle Zeitstatistiken durcheinander. Aus dem Single-Modus kann man mit CRTL-C wieder in den Kommando-Modus zurückkehren. Indem man dem Befehl **single** eine Anzahl als Argument mitgibt, wird nach erreichen dieser Anzahl automatisch in den Kommando-Modus gewechselt.

Beispiel:

```
if> single
   *s  24375 -  12: ( 26, 162 )      1
if>
```

Das Userinterface erlaubt dem Benutzer neben den oben gezeigten Tätigkeiten, die laufende Auswertung durch **kill** definiert abzubrechen. Mit dem oben beschriebenen Befehl **auswertung r** kann die Messung zu einem beliebigen Zeitpunkt fortgesetzt werden. Wird das Kommando **kill** eingegeben, so verlangt das System noch eine Bestätigung, um einen versehentlichen Abbruch zu verhindern. Der Abbruch erfolgt so, daß keine weiteren Bilder eingezogen werden. Die in Bearbeitung im System befindlichen Bilder werden jedoch zu Ende ausgewertet. Sobald der Abbruch wirksam wird, wird eine Meldung auf dem Terminal ausgegeben. Es folgen eventuell noch weitere Resultate, bis alle Prozessoren ihre Ergebnisse abgeliefert haben. Dann wird das System wie üblich beendet.

Beispiel:

```
if> kill
   Wollen Sie die Auswertung wirklich abbrechen? (j/n) : j
   ******* interrupted ******
      1160 -  1: ( 141,  12 )      5
      1159 -  1: ( 140,  12 )      1

   23.11.1994 - 21:07:37  Datei-Aufbereitung ..
   ..... usw.
```

Das obige Beispiel zeigt die Abbruch-Meldung und die noch folgenden Resultate.

Nach Beendigung der Messung wird, wie auch beim gezielten Abbruch, eine ausführlichere Statistik über die Messung angeboten. Diese Statistik kann während der Auswertung im Kommando-Modus des Userinterface jederzeit durch Eingabe des Befehls **stat** abgerufen werden. Es wird dann die Information angezeigt, die man als Gesamt-Statistik bezeichnet. Sie wird im folgenden ausführlich diskutiert.

Das Ende einer Auswertung könnte wie folgt aussehen:

Beispiel:

```
if>

   23.11.1994 - 16:42:50  Datei-Aufbereitung ..

Prozessoren |   Jobs | [   all,     job,   trans,  wait ]
      8        24460   [  2675,   1277,    663,   1056 ]

Gesamt-Zeit: 1:59:11.45   Jobs: 24460
```

```
Zeit / Job : 0.29 s
Prozessoren: 8   - Transfers: 5
Performance:  54.63%

Transputer-Rechenzeit:  8:40:52.46  -->  Leistungsfaktor = 4.37

Statistik: Prozessor [p], Netz [n], Baum [b], gesamt [g], weiter [w] :w
ioda>
```

Im Beispiel wird nach Beendigung der Messung angegeben, daß 8 Prozessoren an 24460 Bildern gearbeitet hatten und dazu 1h59min11sec benötigten, was einer Bearbeitungszeit von 0.29 Sekunden pro Bild als Durchsatz entspricht. Hätten die Transputer nicht parallel gerechnet, so hätten sie 8h40min52sec benötigt. Durch den parallelen Einsatz konnte eine Zeitersparnis um den Faktor 4.37 erzielt werden, d.h 5 Prozessoren hätten gereicht. Erwartet hätte man einen Faktor von etwas weniger als 8. Daß dieser nicht zustande kommt, kann man schon an der Ausnutzung der Transputerfarm von nur 54.63% und der hohen durchschnittlichen wait-Zeit erkennen. Die Ursache liegt darin, daß dem System die Bilder zur Bearbeitung in keinem kürzeren Takt als etwa 0.29 Sekunden angeboten werden können, weil der Kreuztisch des Mikroskops zum Positionieren und die anschließende Bildakquisition und Übertragung des Bildes in die Farm nicht unter dieser Zeit zu realisieren ist.

Die zusätzliche Statistik, die sich mit der letzten Zeile aktivieren läßt, ist nur für Performance-Untersuchungen interessant, und setzt das Verständnis des Aufbaus der Transputer-Farm (siehe Anhang) voraus.

Mit der Eingabe **b** für Baum erhält man Angaben darüber, wie das Transputernetz aufgebaut ist.

Beispiel:

```
Statistik: Prozessor [p], Netz [n], Baum [b], gesamt [g], weiter [w] : b
   Prozessor-Baum:
   |0|  VME(0)
   |1|  0(1 2 3)
   |2|  1(4-6) 2(7 8 -) 3(--12)

   Prozessor-Nachfolger:
   |0|  0(8)
   |1|  1(2) 2(2) 3(1)
   |2|  4(-) 6(-) 7(-) 8(-) 12(-)
   Transfers: 13
Statistik: Prozessor [p], Netz [n], Baum [b], gesamt [g], weiter [w] :
```

Aus der Angabe unter Prozessor-Baum ist zu ersehen, daß es drei Ebenen (0,1 und 2) gibt. Die nullte Ebene wird durch den Prozessor mit der Nummer 0 repräsentiert, der am VME-Bus angekoppelt ist, und über den die gesamte Kommunikation zwischen Manager (Motorola68030) und der nachgeschalteten Transputerfarm läuft. Er fungiert als Master des Netzes und führt keine Berechnungen für die Vermessungen aus. Auf der zweiten Ebene (Ebene1) sind die Prozessoren mit den Nummern 1, 2 und 3 jeweils an die Links (serielle Verbindungen) des Masters angekoppelt, an denen wiederum Transputer der dritten Ebene (Ebene2) hängen. An dieser

Stelle sei erwähnt, daß jeder Transputer über 4 Links verfügt, von denen einer bei unserer Architektur die Verbindung zu höheren Ebene und die restlichen drei Verbindungen zu nachgeschalteten Ebenen herstellen können; auf diese Weise läßt sich ein trinärer Baum realisieren, der im Prinzip beliebig fortgesetzt werden kann. In unserem Beispiel sind in der dritten Ebene 5 Prozessoren untergebracht, von denen 2 (Nummer 4 und 6; 5 bleibt frei) am Prozessor 1, 2 Stück (7 und 8; 9 bleibt frei) am Prozessor 2 und einer (Nummer 12; 10 und 11 bleiben frei) am Prozessor 3 der zweiten Ebene angekoppelt sind. Die Prozessor-Nummern ergeben sich aus den Links, die bei uns für einen trinären Baum durchnumeriert wurden.

Die Angaben zu Prozessor-Nachfolger informieren, wieviele nachgeschaltete Transputer (Angabe in der Klammer) insgesamt durch den entsprechenden Prozessor (Nummer vor der Klammer) erreichbar sind.

Im folgenden Beispiel werden statistische Angaben über die Auslastung der einzelnen Transputer im Netz gemacht. Außerdem wird nochmal der Aufbau des Netzes angezeigt; z.B. wird unter T die Anzahl der Nachfolger und unter V die Nummer des Vorgängers eines Prozessors angegeben. Unter Jobs wird die Anzahl der von dem entsprechenden Transputer bearbeiteten Bilder gezeigt. Diese Angabe macht deutlich, warum die Performance bei der gezeigten Messung nur knapp über 50% lag.

Beispiel:

Statistik: Prozessor [p], Netz [n], Baum [b], gesamt [g], weiter [w] : p

Prozessor			Time				
Nr.	(T)	V	Jobs	[all,	job,	trans,	wait]
0	(8) >	VME		: -- Master			
1	(2) >	0	: 5692	[2874,	1249,	876,	4]
2	(2) >	0	: 5688	[2890,	1249,	912,	4]
3	(1) >	0	: 5688	[2880,	1249,	889,	4]
4	(0) >	1	: 1231	[2695,	1810,	214,	3965]
6	(0) >	1	: 1231	[2076,	1248,	68,	4554]
7	(0) >	2	: 1232	[2079,	1248,	95,	4550]
8	(0) >	2	: 1233	[2137,	1248,	199,	4547]
12	(0) >	3	: 2465	[2100,	1248,	109,	1649]

Statistik: Prozessor [p], Netz [n], Baum [b], gesamt [g], weiter [w] :

Von den 8 bei der Auswertung beteiligten Prozessoren haben drei (Nummer 1, 2 und 3) jeweils über 5600 Jobs gerechnet, einer bearbeitete noch über 2400 Bilder und die restlichen vier mußten sich mit jeweils etwas über 1200 begnügen. Diese unterschiedlichen Zahlen ergeben sich dadurch, daß die in der Hierarchie weiter oben stehenden Prozessoren, sobald sie bereit sind einen neuen Job zu bearbeiten, keinen zur nachgeschalteten Ebene durchlassen. Dies wird auch deutlich wenn man die unter *wait* angegebenen Werte vergleicht. Was aus dieser Statistik ebenfalls sichtbar wird, ist die Zeit, die ein Prozessor durchschnittlich zur Abarbeitung eines Bildes gebraucht hat. Dabei läßt sich feststellen, daß der Transputer mit der Nummer 4 etwa 50% länger an einem Job gearbeitet hat, als die restlichen. Die Ursache dafür ist die Taktfrequenz dieses Transputers, die sich von derjenigen der anderen unterscheidet.

Das letzte Beispiel gibt an, wieviele Jobs wie häufig gleichzeitig im Transputernetz waren. Wenn man bedenkt, daß jeder Transputer einen Job im Puffer haben kann während einer gerechnet wird, kann man sehen, daß vier Transputer ausgereicht hätten, um das gestellte Pro-

blem (in diesem Fall Spot-Auswertung eines Filmsegments) zu lösen. Dann hätte auch die Performance-Angabe bei etwa 98% gelegen.

Beispiel:

```
Statistik: Prozessor [p], Netz [n], Baum [b], gesamt [g], weiter [w] : n

   Netzbelegung:
 Jobs im Netz | Vorkommen
    4              30
    5            1711
    6           12558
    7            9802
    8             352
    9               3
Durchschnitt: 6.36

Statistik: Prozessor [p], Netz [n], Baum [b], gesamt [g], weiter [w] :w
ioda>
```

5.4 Programm IODA_ED

Im Programm **ioda_ed** wird in Zusammenarbeit mit dem Mikroskop-Kreuztisch die Festlegung der auf einem Film auszuwertenden Segmente getroffen. Hierbei handelt es sich nicht um ein Untermenue von IODA, obwohl es nach der Eingabe der Anweisung **edit** den Anschein hat. Es wird vielmehr ein Prozeß gestartet, dem von IODA aus der in **parms** angegebene Bereichsdatei-Name **bereich** als Parameter mitgegeben wird. Das Programm kann auch von der OS9-Shell aus durch den Befehl **ioda_ed**, dem wahlweise ein Dateiname mitgegeben werden kann, aktiviert werden.

Beispiel:

```
ioda> edit ?
   Syntax: edit [<datei>]
ioda> edit
```

Als Prompt wird in edit *ed>* angezeigt. Der Aufbau und die Wirkungsweise etlicher Anweisungen ist identisch oder doch ähnlich zu denen in IODA. So wird mit dem Fragezeichen **?** auch hier eine Liste der möglichen Befehle und mit einem Fragezeichen hinter einem Befehl die entsprechende Syntax ausgegeben.

Beispiel:

```
ed> ?
   bye        Beenden des Programms
   shell      lokale Shell aufrufen
   dir        Directory anzeigen oder aendern
   new        Einrichten eines neuen Bereichs
   load       Datei laden
   save       Datei abspeichern
```

```
    check       Korrektheit pruefen
    show        Bereich auf dem Display zeigen
  > bereich     Bearbeitung des Auswerte-Bereiches
  > segment     Bearbeitung der einzelnen Segmente
  > lage        Lage an veraenderte Filmposition anpassen
    objektiv    Objektiv wechseln
    speed       Setzen der Tisch-Geschwindigkeit
    position    Positionierung des Tisches ( x y )
ed>
```

Durch das Winkelzeichen > vor dem Befehlsnamen wird hier ebenfalls angezeigt, daß sich hinter dem Namen ein Untermenue verbirgt.

Die Befehle **bye, shell** und **dir** funktionieren hier so wie im IODA-Hauptmenue und werden nicht nochmals erklärt; der einzige Unterschied ist der, daß **bye** beim Aufruf über IODA nicht in die OS9-Shell, sondern zum IODA-Hauptmenue zurückführt.

Zur Klärung der Begriffe sei erwähnt, daß als Bereich ein ganzer Film und die darauf auszuwertenden Gebiete als Segmente bezeichnet werden.

Mit **new** wird veranlaßt, daß ein neuer Bereich (Geometrie eines neuen Filmes) eingerichtet wird; war mit **load** ein Geometrie-Parametersatz einer vorangehenden Sitzung geladen, so wird dieser damit gelöscht. Selbstverständlich wird vor dem Löschen vom Benutzer eine Quittung verlangt, damit nicht versehentlich Daten verloren gehen.

Beispiel:

```
ed> new ?
Syntax: new
ed> new
Wollen Sie ganz neu beginnen ?? (j/n) : j
ed>
```

Die Anweisung **load** lädt den Inhalt der Bereichs-Datei, deren Namen dem **edit**-Aufruf als Argument mitgegeben wurde, oder, falls dies nicht der Fall war, deren Namen im IODA-Untermenue **parms** als **bereich** angegeben wurde. Natürlich ist es auch möglich dem Befehl **load** selbst diesen Namen als Argument mitzugeben.

Ganz ähnlich verhält es sich bei der Speicherung der in den Untermenues festgelegten Geometrie-Parameter mit **save**. Entweder wird der aus IODA bekannte Bereichsdatei-Name benutzt oder der dem **save**-Befehl als Argument beigefügte Namen. Für die Dateinamen gelten die in OS9 festgelegten Konventionen.

Beispiel:

```
ioda> edit ../../dummy/xxtest
      - - I O D A _ E D - -  Version 1.01 v. 23.11.94 --
ed>
                    :
                    :
ed> sav
    Wollen Sie '../../dummy/xxtest' ueberschreiben ?? (j/n) : n
    Save ist nicht erfolgt !
ed>
```

In dem obigen Beispiel wurde der Editor von IODA aus mit einem Datei-Namen als Argument gestartet. Diese Datei (*xxtest*) liegt in dem zum übergeordneten Verzeichnis auf gleicher Hierarchiestufe stehenden Verzeichnis *dummy*. Beim Versuch dieselbe Datei wieder zu beschreiben, wird zuerst nachgefragt, ob das wirklich gewünscht ist.

Mit dem Befehl **check** wird geprüft, ob ein konsistenter Datensatz vorhanden ist. Im folgenden Beispiel wurde mit **new** ein vorhandener Datensatz gelöscht und dann mit **check** die gezeigten Fehlermeldungen ausgelöst.

Beispiel:

```
ed> check
    ** Error **  Lage-Punkt 1 fehlt
    ** Error **  Lage-Punkt 2 fehlt
    ** Error **  Lage-Punkt 3 fehlt
    ** Error **  Kein Segment definiert
    Error #064:071  Objekt nicht definiert
ed> load
ed> check
ed>
```

Die Anweisung **show** zeigt auf dem Display die definierten Segmente, die in IODA ausgewertet werden sollen. Damit kann überprüft werden, ob die Geometrie-Daten in **segment** vernünftig gesetzt wurden. Insbesondere kann man damit sehen, ob die in **segment** als letztes Koordinatenpaar benötigte Position wirklich im Inneren des definierten Polygons liegt (siehe Untermenue **segment**). Diese letzte Position ist Ausgangspunkt eines Füll-Algorithmus, der das definierte Polygon mit dem dazu angegebenen **label**-Wert beim Erstellen der Fahrmaske im IODA-Untermenue **tisch** ausfüllen muß. Wurde diese letzte Position in **segment** ungünstig gewählt, so wird dies bei **show** dadurch sichtbar, daß nicht das Innere des Polygons, sondern der Rest des dargestellten Bereichs ausgefüllt erscheint.

Mit **objektiv** wird die Vergrößerung des sich im Strahlengang des Mikroskops befindlichen Objektivs angezeigt und mit der Übergabe einer Größe **5, 10, 20, 50** oder **100** das entsprechende Objektiv in den Mikroskop-Strahlengang geschaltet. Wie beim IODA-Untermenue **tisch** wird auch hier durch den Befehl **objektiv 0** die Ansteuerung des Objektivrevolvers für den manuellen Betrieb per Taste (auf der linken Handablage) durch den Computer freigegeben

Durch den Befehl **speed** läßt sich die Tischfahrgeschwindigkeit einstellen, bzw. die aktuell eingestellte abrufen. Der Wert, der ausgegeben, bzw. eingegeben wird, ist die Geschwindigkeit in Tischeinheiten pro Sekunde; eine Tischeinheit beträgt 10nm. Aus technischen Gründen wird die Fahrgeschwindigkeit in z-Richtung in jedem Fall verringert. Die minimale Angabe ist 500, die maximale 500000, wobei nur bestimmte Einstellungen möglich sind; bei Eingabe einer Zwischengröße wird automatisch auf den nächst niedrigen Wert geschaltet.

Beispiel:

```
ed> speed
    speed = 500000
ed>
```

Die Anweisung **position** oder eine geeignete Abkürzung veranlaßt, daß die x- und y-Koordinaten der aktuellen Kreuztischposition angezeigt werden. Werden der Anweisung Koordinaten mitgegeben, so wird der Tisch an diese Position gefahren.

Beispiel:

```
ed> pos ?
    Syntax: position [<x> [<y>]]
ed> pos
        position = ( 8030475, 5289975 )
ed>
```

Soll nur in x-Richtung gefahren werden reicht die Angabe dieser Koordinate; wird nur in y-Richtung bewegt, so muß die x-Koordinate mit angegeben werden. Die im Beispiel gezeigten Werte sind absolute Tischkoordinaten in 10nm-Einheiten.

5.4.1 Menue BEREICH

In diesem Untermenue von **edit** werden die Daten eines Auswerte-Bereichs vorbereitet. Dieser Bereich entspricht nach unserem Verständnis einem kompletten Film, die auf dem Film auszuwertenden Regionen sind dann Segmente, für die hier in **bereich** die Verwaltung stattfindet. Das Untermenue wird von **edit** aus durch Eingabe von **bereich** oder einer geeigneten Abkürzung aktiviert. Der Prompt innerhalb des Menues ist *bereich>*. Wie gewohnt, erhält man durch das Fragezeichen **?** die Liste der möglichen Anweisungen des Menues.

Beispiel:

```
bereich> ?
    !          Ausgabe der momentanen Einstellung ( Menue )
    set        speichere Position
    go         fahre gespeicherte Position an
    film       Setzen der Film_id
    list       Liste der definierten Segmente
    delete     Loeschen eines Segmentes
    create     Anlegen eines Segmentes
bereich>
```

Die Syntax der jeweiligen Anweisung wird durch das Fragezeichen **?**, das der Anweisung als einziges Argument mitgegeben wird, angezeigt. Im folgenden Beispiel ist die Liste dazu zusammengestellt.

Beispiel:

```
bereich> ??
    set        [1|2|3]
    go         [1|2|3]
    film       [<Film-Id>]
    list
    delete     <segment_nr>
    create     <segment_nr>
bereich>
```

Zur Filmidentifikation kann mit der Anweisung **film** eine alphanumerische Zeichenkette definiert werden, die im IODA-Untermenue **parms** neben der Schuß-Nummer (nachdem die Fahr-

maske erstellt wurde) angezeigt wird. Damit könnten verschiedene, bei einem Schuß erzeugte Filme, bzw. Auswertungen unterschieden werden.

Das Ausrufezeichen **!** bewirkt, daß die aktuellen Parameter des Untermenues angezeigt werden.

Beispiel:

```
bereich> !
   ========================= bereich ========================
      position = ( 8030475, 5289975 )

      1    4934025  3809175   Punkt 1
      2   10116950  3809175   Punkt 2
      3   10116950  7705625   Punkt 3

      Film-Id: xxtest1

      Segmente:
      Nr. Label Punkte
       1    2    4
       2    1    5
       3    3    4

   ========================= bereich ========================
bereich>
```

Im obigen Beispiel wird zuerst die aktuelle Kreuztischposition in 10nm-Einheiten angezeigt. Anschließend folgen die Koordinaten von drei Punkten, die notwendig sind, um eine reproduzierbare Lage des Films auf dem Kreuztisch herzustellen. Diese drei Punkte sollten Stellen auf dem Film sein, die leicht zu finden bzw. wiederzufinden sind. Es hat sich bewährt, drei Ecken des Films zu benutzen, wenn sie unter dem Mikroskop gut zu erkennen sind. Damit ist man in der Lage, einen schon ausgewerteten Film irgendwann nochmal aufzulegen und die ursprünglich definierten Segmente für die Fahrmaske im Menue **lage** mit Hilfe der drei alten und neuen Punkte auf die aktuelle Filmposition zu transformieren.

Als nächstes wird der in **film** angegebene alphanumerische String zur Identifikation des Bereichs gezeigt.

Am Ende werden die erzeugten und in **segment** mit einem Polygonzug versehenen Auswertesegmente mit Nummer, mit in **segment** zugeordnetem **label** und der jeweiligen Anzahl Polygon-Punkte aufgelistet. Diese Liste wird auch durch den Befehl **list** erzeugt.

Die Anweisungen **set [1|2|3]** und **go [1|2|3]** werden benötigt, um die Koordinaten der drei signifikanten Stellen auf dem Film den entsprechenden Punkten P1, P2 und P3 zuzuordnen, bzw. zur Überprüfung anzufahren. Zum Fahren kann auch der Joystick benutzt werden. Der Befehl **set** ordnet die aktuelle Position dem als Nummer angegebenen Punkt zu.

Mit **create <segment-nr>** wird ein neues Auswerte-Segment erzeugt, das im Menue **segment** noch mit Daten versehen werden muß. Es muß keine fortlaufende Segmentnumerierung stattfinden. Die Anweisung **delete <segment-nr>** löscht das Segment mit der entsprechenden Nummer. Vorher wird noch einmal nachgefragt, um versehentliches Löschen zu verhindern.

Beispiel:

```
bereich> del 2
    Segment  2, Label  1 mit  5  Punkten loeschen? (j/n)  n
bereich>
```

5.4.2 Menue SEGMENT

Im EDIT-Untermenue **segment** werden die in **bereich** kreierten Segmente mit den notwendigen Daten versehen. Diese Daten bestehen aus einer Reihe von Koordinatenpaaren, die einen Polygonzug definieren. Dieses Polygon umschreibt das auszuwertende Segment. Außerdem erhält es zur Identifikation in IODA ein Label.

In das Menue kommt man durch Eingabe von **segment** oder einer eindeutigen Abkürzung im EDIT-Haupt- oder einem gleichwertigen Unter-Menue. Als Zeichen, daß man sich im Untermenue befindet, wird der Prompt *segment*> benutzt.

Wie in jedem Menue wird durch Eingabe des Fragezeichens **?** die Liste der möglichen Anweisungen ausgegeben.

Beispiel:

```
ed> segment
segment> ?
    !          Ausgabe der momentanen Einstellung ( Menue )
    select     selectiere ein Segment
    label      definiere Label fuer Segment
    list       Listen von Punkten
    set        ueberspeichere mit Position
    add        speichere Position
    go         fahre gespeicherte Position an
    delete     Loeschen von Punkten
    clear      Speicher loeschen
segment>
```

Die Syntax der jeweiligen Anweisung wird durch das Fragezeichen **?**, das der Anweisung als einziges Argument mitgegeben wird, angezeigt. Im folgenden Beispiel ist die Liste dazu zusammengestellt.

Beispiel:

```
segment>??
    select     [<segment_nr>]
    label      [<label_nr>]
    list       [<from> [<to>]]
    set        [<pos_nr> [xly]]
    add        [<pos_nr>]
    go         [<pos_nr> [xly]]
    delete     [<from> [<to>]]
    clear
```

Innerhalb des Menues muß vor der Bearbeitung ein gültiges (in **bereich** erzeugtes Segment)

mit **select <segment_nr>** ausgewählt werden. Mit dem Ausrufezeichen wird dann die Information über dieses Segment angezeigt.

Beispiel:

```
segment> select 3
segment> !
========================= segment ========================
   Aktuelle Segment-Nummer : 3

   position = ( 5630475, 7039975 )

   kein Punkt definiert
========================= segment ========================
segment>
```

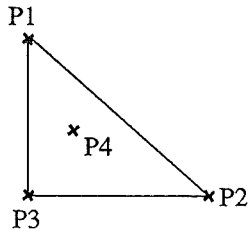
Sind, wie beim obigen Beispiel, noch keine Polygon-Punkte definiert (neues Segment) wird mit dem Joystick der Mikroskop-Kreuztisch an entsprechende Film-Positionen gefahren und diese Position mit **add** jeweils an das Ende der Koordinatenliste angehängt. Wird mit **add** eine Positionsnummer angegeben, so wird die aktuelle Position vor dieser Nummer eingefügt.

Soll ein vorhandenes Koordinatenpaar, das in der Liste die Nummer **pos_nr** hat, durch die augenblickliche Kreuztisch-Position ersetzt werden, so geschieht dies mit **set <pos_nr>**. Es gibt die Möglichkeit, mit **set** nur eine Koordinate durch Angabe des Koordinaten-Namens zu überschreiben, z.B. ersetzt der Befehl **set 3 y** die y-Koordinate der in der Liste an dritter Stelle stehen Position durch den y-Wert der aktuellen Kreuztisch-Position.

Der Befehl **delete** mit einer Positions-Nummer löscht die Eintragungen unter dieser Nummer und numeriert die nachfolgenden entsprechend um. Mit zwei Argumenten löscht **delete** die Eintragungen zwischen den angegebenen Nummern (je inklusive). Die Anweisung **clear** löscht den gesamten Segment-Speicher des selektierten Segments, allerdings erst nach einer ausdrücklichen Zustimmung des Benutzers.

Mit **go <pos-nr>** kann die in **pos-nr** der Koordinaten-Liste des selektierten Segments angegebene Position angefahren werden; durch Angabe eines Koordinaten-Namens wird nur die mit dem Namen spezifizierte Richtung gefahren, die andere Koordinate bleibt erhalten. Damit ergibt sich die Möglichkeit, z.B. rechtwinklige Segmente zu erzeugen, was bei Benutzung des Joysticks alleine nicht so leicht ist. Die folgende Skizze soll dies am Beispiel eines rechtwinkligen Dreiecks verdeutlichen. Wird mit **go** kein Argument übergeben, so werden mit jedem **go** die in der Koordinatenliste stehenden Positionen der Reihe nach angefahren (auf die letzte folgt wieder die erste Position.

Beispiel:



Vorgehensweise:

1. P1 mit Joystick anfahren.
2. add
3. P2 mit Joystick anfahren.
4. add
5. go 1 x (damit P3 anfahren)
6. add
7. P4 mit Joystick anfahren
8. add

Die in der Skizze als P4 angegebene Position muß unbedingt definiert werden. Es ist immer die letzte Position in einer Segment-Liste. Die Position muß so gewählt werden, daß sie innerhalb des durch die restlichen Punkte definierten Polygons liegt. Diese Koordinaten werden benutzt, um im IODA-Untermenue **tisch** für einen Füll-Algorithmus den Startpunkt festzulegen. Von diesem Startpunkt aus wird das Innere des Polygons mit dem zugehörenden **label** aus **segment** ausgefüllt. Diese Information benötigt die Auswertung in IODA, um zu wissen, an welchen Positionen ein Videobild eingezogen und verarbeitet werden soll. Mit dem Befehl **list** werden je nach Wunsch einzelne Positionen oder die gesamte Positionsliste des selektierten Segments ausgegeben.

Beispiel:

```
segment> list
      1   6572275   5696275
      2   7421350   5696200
      3   7421350   4726575
      4   7329500   5270700
segment>
```

Nachdem das selektierte Segment mit den entsprechenden Daten versehen wurde, sieht die Anzeige der mit dem Ausrufezeichen **!** erhaltenen momentanen Einstellung folgendermaßen aus.

Beispiel:

```
segment> !
   ========================= segment ========================
      Aktuelle Segment-Nummer : 3

      position = ( 7329500, 5270700 )

      Punkt 1 - 4 sind definiert
   ========================= segment ========================
segment>
```

Wie oben erwähnt, muß jedem Segment ein Label zugewiesen werden, wobei es möglich ist, mehreren Segmenten dasselbe Label zu geben. Diese Label werden im IODA-Untermenue **methode** benutzt, um verschiedenen Segmenten verschiedene Methoden zuordnen zu können. Label werden mit der Anweisung **label [<label-nr>]** vergeben. Wird der Anweisung **label** kein Argument mitgegeben, so werden alle erzeugte Segmente mit den entsprechenden zuge-

wiesenen Labeln und der Anzahl definierter Punkte ausgegeben.

Beispiel:

```
segment> label
   Segmente:
   Nr. Label Punkte
    1    2    4
    2    1    5
    3    3    4
    4    0    0
segment>
```

Wie im Beispiel zu sehen ist, wurden in **bereich** insgesamt vier Segmente erzeugt, wobei für das letzte bisher weder Polygon-Punkte, noch ein Label definiert sind.

5.4.3 Menue LAGE

Das Untermenue **lage** wurde geschaffen, um bei gleichartigen Filmauswertungen (gleiche Filmabmessungen und gleiche Position der auszuwertenden Regionen auf dem Film) die Bereichs- und Segment-Definitionen vorausgegangener Auswertungen wiederverwenden zu können. Benutzt werden dazu die in **bereich** festgelegten drei markanten Stellen auf dem Film.

In das Menue kommt man aus **edit** oder einem der anderen Untermenues durch den Befehl **lage**; der Prompt des Menues ist *lage*>.

Auch hierbei wird nach Eingabe des Fragezeichens **?** die Liste der im Menue möglichen Befehle angezeigt.

Beispiel:

```
ed> lage
lage> ?
   !          Ausgabe der momentanen Einstellung ( Menue )
   position   Positionierung des Tisches ( x y )
   set        speichere Position
   go         fahre gespeicherte Position an
   exec       rechne auf neue Lage um
lage>
```

Die entsprechende Syntax ist im folgenden Beispiel aufgelistet.

Beispiel:

```
   position   [<x> [<y>]]
   set        [1|2|3]
   go         [<memo_nr>|+]
   exec
```

Die Aufgabe besteht darin, für die beim untenstehenden Beispiel angegebenen Positionen 4, 5 und 6 entsprechende Positionen 1, 2 und 3 zu finden. Bei den Positionen 4 bis 6 handelt es sich um die Koordinaten der in **bereich** einer vorangegangenen Auswertung festgelegten markanten Stellen auf dem Film. Soll der gleiche Film nochmal oder ein neuer Film mit gleichen Eigenschaften ausgewertet werden, so können die einmal definierten Polygone der einzelnen Segmente nach einer Transformation auf die aktuellen Tischpositionen benutzt werden, um die

neue Fahrmaske für die Auswertung zu erstellen.

Beispiel:

```
lage> !
   ========================= lage ========================
      position = ( 6572275, 5696275 )

      1      0        0       Neu 1
      2      0        0       Neu 2
      3      0        0       Neu 3
      4   4934025  3809175    Alt 1
      5  10116950  3809175    Alt 2
      6  10116950  7705625    Alt 3

      Film-Id: xxtest1       Segmente: 4
   ========================= lage ========================
lage>
```

Man geht dabei folgendermaßen vor:

Nachdem der Film aufgelegt wurde, fährt man mit **go 4** zu den Koordinaten der ursprünglich bestimmten markanten 1. Filmstelle (gut sichtbare Film-Ecke oder Markierung). Mit dem Joystick oder dem Befehl **position** muß nun der Mikroskop-Kreuztisch so positioniert werden, daß diese Stelle, wie bei der ersten Einstellung in der Video-Bild-Mitte (durch ein Fadenkreuz auf dem Display gekennzeichnet) plaziert wird. Anschließend mit dem Befehl **set 1** diese Position in den Speicher 1 bringen. Bei den restlichen zwei Positionen muß entsprechend verfahren werden. Nachdem alle Zuordnungen getroffen sind, kann mit der Anweisung **exec** die Transformation der Polygon-Punkte aller Segmente durchgeführt werden. Die dann in **edit** mit **save dateiname** abgespeicherte Polygondatei enthält nun die zur Erstellung der Fahrmaske für die aktuelle Auswertung benutzbaren Polygon-Daten, die die Segmente des aufliegenden Films beranden.

6. Beispiel-Auswertung

Zum besseren Verständnis der Vorgehensweise bei der Auswertung und als ausführliche Anleitung wird im folgenden eine Auswertung am Beispiel erläutert. Es handelt sich dabei um den Kalif-Schuß 3357, der am 17.12.92 ausgewertet wurde. Der Ablauf dieser Auswertung wird hier beispielhaft nachvollzogen.

6.1 Vorbereitung

Beim Betrachten des zu untersuchenden Filmes mit dem bloßen Auge kann unter Umständen schon festgestellt werden, ob eine Auswertung erfolgversprechend ist oder nicht. Sind die Teilcheneinschläge so dicht, daß nach dem Ätzen manche Stellen völlig undurchsichtig sind, oder komplette Gebiete der oberen Filmschicht fehlen, so kann man davon ausgehen, daß bei einer Auswertung keine brauchbaren Ergebnisse erzielt werden können.

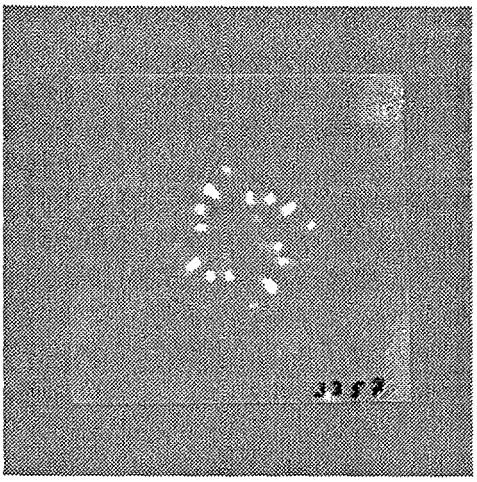


Abbildung 10. Aufnahme eines beschossenen und geätzten CR39-Films.

Das Bild zeigt eine Shadowbox-Aufnahme, die bei dem Schuß mit der Nummer 3357 am KALIF des INR entstand. Die hellen Flecken rühren von Teilcheneinschlägen her. Aufgabe ist, die Einschlagdichte im mikroskopischen Raster zu ermitteln und das Ergebnis als Matrix in einem File zur Verfügung zu stellen.

Soll eine Auswertung durchgeführt werden, dann wird der CR39-Film so auf die Glasplatte des Mikroskop-Kreuztischs aufgelegt, daß die geätzten Löcher zum Objektiv hin zeigen. Auf dem Kreuztisch befindet sich eine Aluminiumplatte, die als Anlegekante für den Film benutzt werden sollte. Da das Aussehen der Löcher im Mikroskop sehr stark von der jeweils eingestellten Fokus-Position abhängt, ist sicherzustellen, daß alle zu erkennenden Löcher bei der Auswertung an derselben Fokusposition zu beobachten sind. Dazu wird eine $10*10cm^2$ große planparallele und optisch reine Glasplatte auf den Film gelegt und mit zwei seitlichen "Klemmbacken" so fixiert, daß der Film plan zwischen den beiden Glasplatten liegt. Eventuell durch das Stanzen des Filmes vor der "Belichtung" hervorgerufene verbogene Filmkanten sollten deshalb beseitigt werden.

Bevor man mit einer Inspektion beginnt, sollte der VME-Bus-Rechner in einen Zustand versetzt werden, daß das Ionendichte-Analyse-System IODA gestartet werden kann.

Das Terminal und das Display werden eingeschaltet. Nach Betätigen der Return-Taste fordert der Rechner zum **login** auf. Nach dem Einloggen für den Benutzer **inr** (Passwort erfragen) befindet man sich in dem Root-Directory von INR. Durch Eingabe von **ll** wird der Inhalt des Ver-

zeichnisses angezeigt. In das dabei angezeigte Sub-Directory IODA kommt man mit **chd ioda** (es gibt keine Unterscheidung zwischen Groß- und Klein-Schreibung). Die Directory-Struktur wurde so angelegt, daß unterhalb des IODA-Verzeichnisses für jede Auswertung eines Filmes ein eigenes Subdirectory erzeugt werden sollte. Üblicherweise erhält dieses Directory die Nummer des (KALIF)-Schusses als Name. Wir benutzen für die beispielhafte Auswertung den Film von Schuß Nummer 3357; entsprechend erzeugen wir mit **makdir 3357a** ein Subdirectory, in dem wir das Programm ablaufen lassen. Mit **chd 3357a** wechseln wir in dieses Arbeits-Directory. Nun können wir mit der Eingabe von **ioda** das System starten und erhalten:

```
> ioda
        --I O D A-- Version 1.01 v. 23.11.94 --
ioda>
```

Für das weitere Vorgehen wird auf die Erklärungen im Manual-Teil verwiesen.

Es folgt eine Inspektion der Regionen, die durch die Teilcheneinschläge und die anschließende Ätzung eine milchige Erscheinung erhalten. Üblicherweise wird mit der Objektiv-Vergrößerung 5x begonnen. Der Mikroskop-Fokus wird so eingestellt, daß die Teilcheneinschläge als dunkle Kreise mit hellem Zentrum zu sehen sind. Es ist prinzipiell egal, ob der Film im Auflicht oder Durchlicht betrachtet wird, solange die Teilcheneinschläge wie oben beschrieben erscheinen. Wird festgestellt, daß die Kreise zu klein sind oder, daß in dichteren Regionen die vielen Einschläge nicht mehr aufgelöst werden können, muß eine höhere Vergrößerung benutzt werden. Läßt sich das Objektiv nicht mit den Tasten an der linken Handauflage des Mikroskops schalten, so kann, nachdem man sich vergewissert hat, daß die Stromversorgung des Objektivrevolvers eingeschaltet ist, das **tisch**-Menue (siehe Manual) von IODA benutzt werden um den Objektivrevolver freizuschalten oder das entsprechende Objektiv per Befehl einzuschalten:

```
ioda> tisch objectiv 0
   objektiv = 5
tisch>
```

Nachdem sich nun die Objektive schalten lassen, kann mit der Inspektion bei verschiedenen Auflösungen fortgefahren werden.

6.2 Festlegung der auszuwertenden Regionen

Wenn man sich nach erfolgter Inspektion entschlossen hat, den Film auszuwerten, müssen mit dem Programm **edit** die auszuwertenden Filmregionen bestimmt werden. Die Daten für diese Filmregionen werden von **edit** in einer Datei (Polygon-Datei) abgelegt. Der Name für diese "Bereichs-Datei" kann im **parms**-Menue mit bereich **dateiname** eingetragen werden und wird beim Start des Bereichs-Editor-Programms **edit** automatisch übergeben.

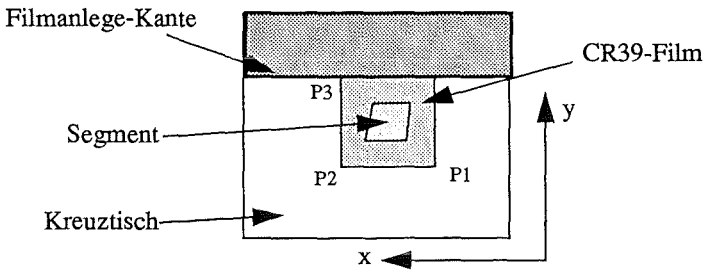
```
parms> bereich polygon
parms> !
========================= parms ========================
      schuss :    0;  - film_id :
      name :
      vergroesserung:   100;   - Bildweite: 28500, 28500 *10nm
      raster : 20000, 20000 *10nm
      dimension : 256, 256 ;    - Anzahl der Layer: 1
```

```
        scan horizontal
        spot : 0
        bereich : /h0/INR/IODA/3357A/polygon
    Maske noch nicht erstellt
    ========================= parms ========================
    parms>
```

Der Befehl **edit** startet nun den Bereichs-Editor, in dem wir die auszuwertenden Filmregionen als Segmente definieren können.

```
    parms> edit
            - - I O D A _ E D - -  Version 1.01 v. 23.11.94 --
    ed>
```

Durch **new** teilt man dem Editor mit, daß ein neuer Bereich eingerichtet werden soll. Nachdem mit **bereich** in das entsprechende Untermenue verzweigt wurde, kann man dem Bereich durch die Anweisung **film schuss3357** den Namen **schuss3357** zuordnen. Nun werden mit Hilfe des Joysticks drei markante Positionen als Film-Justierungspunkte angefahren und jeweils mit **set 1**, **set 2** und **set 3** in die entsprechenden Speicher geladen. Als Justierungspunkte (P1, P2, P3) benutzen wir die unten gezeigten Ecken des Films.



Diese Punkte können benutzt werden, um die Filmlage zu korrigieren, falls zu einem späteren Zeitpunkt die Auswertung eines anderen Filmes mit gleicher Geometrie durchgeführt werden soll. Es ist dann möglich, die hier und im folgenden definierten Filmparameter zu benutzen, indem sie mit **lage** entsprechend der neuen Filmlage transformiert werden. Nun kann man entscheiden, wieviele Segmente man benötigt, um alle interessierenden Teilchen-Einschläge bei der Auswertung zu berücksichtigen. Bei unserem Beispiel reicht ein Segment. Folglich wird mit **create 1** das Segment mit der Nummer 1 angelegt. Das Ausrufezeichen **!** bringt dann die bisher vorhandenen Parameter auf den Monitor.

```
bereich> !
    ======================== bereich ========================
      position = ( 11295000, 4009950 )

      1    4939600  4039900  Punkt 1
      2   11295000  4009950  Punkt 2
      3   11306675  10358775  Punkt 3

      Film-Id: schuss3357

      Segmente:
```

```
   Nr. Label Punkte
    1    0    0
========================= bereich ========================
bereich>
```

Wie im Status zu sehen ist, existiert nun ein Segment, das im Menue **segment** mit den entsprechenden Parametern versehen wird. In das Menue wird mit der Anweisung **segment** gewechselt.

Nachdem durch die Anweisung **select 1** das zu bearbeitende Segment selektiert wurde, wird mit Hilfe des Joysticks die auszuwertende Fläche auf dem Film bestimmt. Diese Fläche kann beliebig geformt sein, weshalb ein Polygon benutzt wird, um die Fläche zu beranden. In unserem Fall reicht ein Viereck, um das Meß-Segment zu beschreiben. Deshalb fahren wir die vier Punkte des Polygons an und fügen mit der Anweisung **add** den jeweiligen Punkt zu der Koordinatenliste des Segments dazu. Wenn das Polygon definiert ist, muß noch ein Punkt im Innern des Polygons an die Koordinatenliste angehängt werden. Dieser dient später bei der Erstellung der Fahrmaske im IODA-Menue **tisch** als Startpunkt für einen Füll-Algorithmus. Müssen die Koordinaten eines Punktes in Liste verändert werden, so können die neuen Koordinaten angefahren werden und anschließend durch **set <pos_nr>** die alten an der Stelle **<pos_nr>** ersetzen. Auch zusätzliche Punkte können an der Stelle mit **add <pos_nr>** eingefügt werden.

Mit dem Befehl **list** kann man die Koordinatenliste ausgeben.

```
segment> list
      1  10088825  9139225
      2   6188875  9139225
      3   6188875  5263425
      4  10088825  5263425
      5   7810525  6601425
segment>
```

Mit **label 1** gibt man dem Segment ein Label, dem zur Auswertung eine Methode zugeordnet werden kann (später in **methode** durch **connect 1**).

Bevor man die Polygondatei in **edit** abspeichert, kann man sich im Menue **bereich** nochmal den Status ausgeben lassen, um zu sehen, daß die Segmentparameter nun definiert sind. Eine Liste mit Angaben über die Segmente erhält man in **bereich** aber auch mit **list**.

```
bereich> list
      Segmente:
      Nr. Label Punkte
       1    1    5
bereich>
```

Mit dem Befehl **save** werden die zuvor in **bereich** und **segment** festgelegten Parameter in eine Datei geschrieben. Der Name der Datei kann zusammen mit dem Befehl **save** angegeben werden. Falls keine Angabe erfolgt, wird der Name benutzt, der vor dem Aufruf des Programms **edit** im **parms**-Menue als **bereich** definiert war; in unserem Fall der Name *polygon*. Nachdem die Polygondatei abgespeichert wurde, kann das Programm **edit** durch **bye** oder das Betätigen der ESCAPE-Taste verlassen werden. Man landet dann wieder in dem Menue von IODA, von dem aus der Editor gestartet wurde.

Der Inhalt der Polygon-datei *polygon* kann angezeigt werden, indem man mit dem **shell**-Kommando eine OS9-Shell startet und, wie im folgenden geschehen, den Datei-Inhalt listet.

```
parms> shell
1.> list polygon
schuss3357
  4939600   4039900  0
 11295000   4009950  0
 11306675  10358775  0
 10088825   9139225  1
  6188875   9139225  1
  6188875   5263425  1
 10088825   5263425  1
  7810525   6601425  1
1.> eof
parms>
```

Die erste Dateizeile beinhaltet die in **bereich** von **edit** festgelegte Film-ID, und die drei folgenden Zeilen die Positionen der drei ebenfalls dort definierten "Justierungspunkte". Anschließend werden die x- und y-Koordinaten der Polygonpunkte angezeigt. In der dritten Spalte werden jeweils die Label zu den entsprechenden Segmenten angegeben; die Justierungspunkte erhalten als Label **0**.

6.3 Bestimmung der Auswerte-Methode

Nun muß überlegt werden, welche Methode zur Auswertung des Filmes benutzt werden kann. Wir entscheiden uns nach der Inspektion für die Sigma-Methode, weil der Bildhintergrund für die übrigen Methoden nicht homogen genug ist, um z.B. bei der Histo-Max-Methode eine für das gesamte Bild benutzbare Grauwertschwelle berechnen zu können. Die Kreis-Methode könnte zwar erfolgversprechend eingesetzt werden, aber damit können gewisse Lochgrößen nicht unterdrückt werden. Wir verzweigen also durch Eingabe von **meth** in das Untermenue **methode** und wählen mit **new 3** als aktuelle Methode die Sigma-Methode aus.

```
methode> new 3
   aktuelle Methode: Sigma
methode> !
======================== methode ========================
        ---- Sigma ----
        dimension: 256
        faktor:   0.00
        maske:
======================== methode ========================
methode>
```

Im angezeigten Status ist zu sehen, daß die Parameter für **faktor** und **maske** noch ermittelt werden müssen.

Bei der eingestellten Vergrößerung (5x) fahren wir auf dem Film eine Stelle an, die gleichzeitig hohe und niedrige Einschlagdichte aufweist. Nach einer Betrachtung der Lochgrößen entscheiden wir uns für die im System vordefinierte Maske **#6.** Der Wert für **faktor** ist stark vom Bildinhalt (z.B. Bildrauschen, Hintergrund-Inhomogenitäten, ...) abhängig und muß mit dem Befehl **test** durch Ausprobieren ermittelt werden. Als Startwert wählen wir **4.0**.

```
methode> set maske #6
methode> set faktor 4.0
methode> !
========================= methode ========================
        ---- Sigma ----
        dimension: 256
        faktor:   4.00
        maske:    #6
========================= methode ========================
methode>
```

Durch wiederholtes Testen (Testauswertung eines Bildes) und Parameter-Ändern können optimale Einstellungen für eine Messung gefunden werden. Zu den Einstellungen gehören neben den Parametern **faktor** und **maske** auch die Mikroskop-Fokussierung, Helligkeit, Blenden, Kondensor-Position und nicht zuletzt die im Menue **kamera** veränderbaren Parameter **gain** und **offset**. Nach dem Ausführen von test wird das dazu benutzte Graubild auf dem Display eingefroren. Diesem Bild wird das Testergebnis überlagert; als Testergebnis wird jedes erkannte Loch im Zentrum mit einem roten Fleck markiert. Deshalb muß nach jedem Ausführen von **test** die Bildakquisition in **kamera** wieder auf **live** geschaltet werden, um die Auswirkungen der Parameteränderungen kontrollieren zu können. Dazu muß man das **methode**-Menue nicht verlassen; mit der eindeutigen Abkürzung **ka l** kann der live-Modus hergestellt werden. Für die Vergrößerung (5x) konnte auf diese Weise als beste Maske **#6** und ein optimaler Wert von **3.35** für **faktor** ermittelt werden. Die Abbildung 11 zeigt als Ergebnis einer Test-Auswertung des als Original bezeichneten Videobildes und das bei **test** zur Begutachtung erzeugte Ergebnisbild.

```
methode> test
   Prozessor 1; Time = 4400 ms
    0:    1361
methode>
```

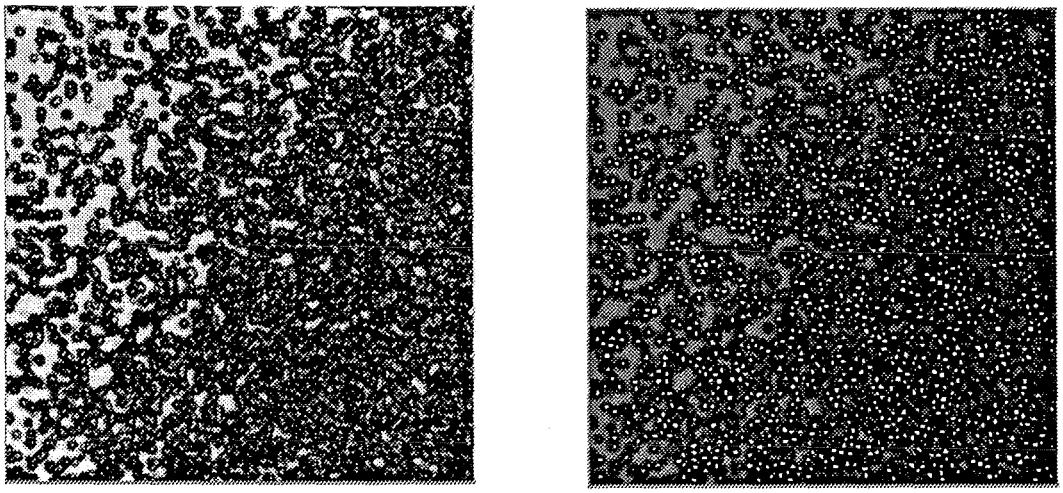


Abbildung 11. Testauswertung bei Objektiv-Vergrößerung 5x

Das linke Bild zeigt das Originalbild einer Mikroskopaufnahme. Die zu detektierenden Löcher sind als dunkle Kreise mit hellerem Zentrum zu sehen.

Das rechte Bild zeigt das Ergebnis einer Auswertung. Die erkannten Löcher werden im Zentrum bei dieser Darstellung mit weißen Punkten markiert.

Wie in Abbildung 11 zu sehen ist, werden die Löcher in dem Originalbild als kleine dunkle Kreise dargestellt. Die zur erfolgreichen Anwendung der Sigma-Methode notwendigen helleren Zentren sind in vielen Fällen nicht sichtbar. Entsprechend wurden, selbst bei der Einstellung der optimalen Parameter, bei der Testauswertung relativ viele Löcher nicht erkannt und in dem Ergebnisbild nicht markiert. Deshalb wurde für die weiteren Tests das Objektiv mit der Vergrößerung 10x benutzt.

Eine Betrachtung der Löcher bei dieser Vergrößerung ergab für die Auswahl einer geeigneten Maske die vom System voreingestellte mit dem Namen **#7**.

Ebenso wie bei dem Objektiv 5x wurden hierbei die oben erwähnten Einstellungen solange variiert, bis optimale Testergebnisse erzielt wurden. Die **kamera**-Parameter **gain** und **offset** wurden mit den Werten **127** bzw. **128** belegt. Weiterhin wurden, wie im folgenden gezeigt, die **methoden**-Parameter eingestellt, um die Ergebnisse in Abbildung 12 zu erhalten.

```
methode> set maske #7
methode> set faktor 4.0
methode> !
========================= methode ========================
        ---- Sigma ----
        dimension: 256
        faktor:   4.00
        maske:    #7
========================= methode ========================
methode>test
    Prozessor 2; Time = 5390 ms
     0:      122
methode>
```

In dem Ergebnisbild in Abbildung 12 ist zu sehen, daß mit den 122 detektierten nahezu alle auf dem Original zu erkennenden Löcher bei dem Test mit den eingestellten Parametern erfaßt wurden.

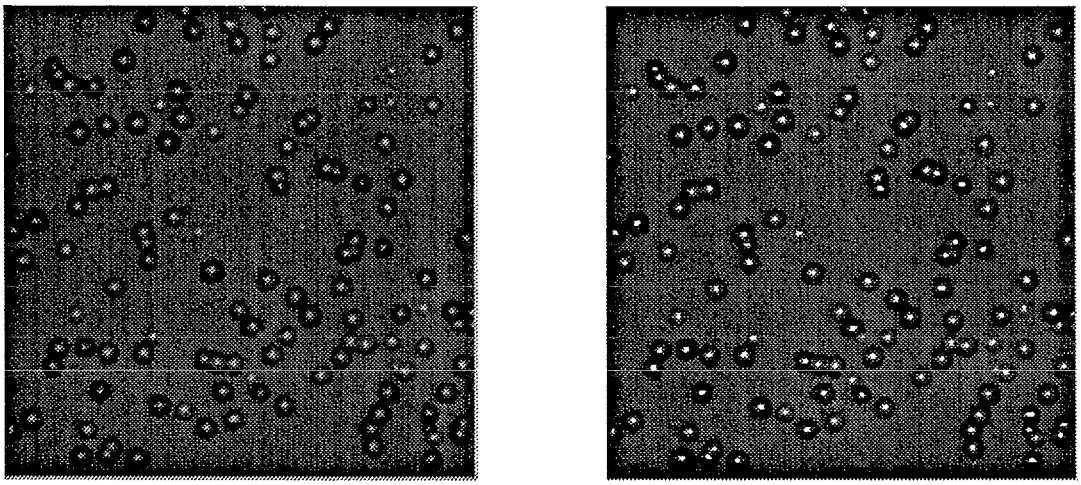


Abbildung 12. Testauswertung mit Objektiv-Vergrößerung 10x.
Das linke Bild zeigt das Originalbild einer Mikroskopaufnahme. Die zu detektierenden Löcher sind als dunkle Kreise mit hellerem Zentrum zu sehen.
Das rechte Bild zeigt das Ergebnis einer Auswertung. Die erkannten Löcher werden im Zentrum weiß markiert

6.4 Definition der übrigen Parameter

Nachdem nun die Methode mit den entsprechenden Werten feststeht, müssen im Menue **parms** die restlichen Parameter definiert werden. Neben der Film-ID, die in **edit** festgelegt wurde, wird hier eine Schuß-Nummer vergeben. Weiterhin wird der Namens-Rumpf der bei der Auswertung entstehenden Dateien angegeben. Die Rasterweite, mit der die zu bearbeitenden Filmflächen abgerastert werden sollen, orientiert sich an der Bildfeldgröße, die durch die Vergrößerung vorgegeben wird. Eine Gesamtvergrößerung von 100 (Objektiv 10x, Okular 10x) führt, wie bei der nachfolgenden Statusanzeige zu sehen ist, zu einer Bildweite von 285µm (28500 *10nm). Um eine flächendeckende Auswertung mit für einen menschlichen Betrachter handhabbaren Rasterweiten durchzuführen, entscheiden wir uns für einen Wert von 250µm (25000 *10nm).

Dazu sind die folgenden Angaben nötig:

```
parms> schuss 3357
parms> name ergebnis
parms> raster 25000
parms> !
========================= parms ========================
      schuss :  3357;  - film_id :
      name : /h0/INR/IODA/3357A/ergebnis
```

```
        vergroesserung:   100;   - Bildweite:  28500,  28500 *10nm
        raster :  25000,  25000  *10nm
        dimension : 256, 256 ;    - Anzahl der Layer: 1
        scan horizontal
        spot : 0
        bereich : /h0/INR/IODA/3357A/polygon
    Maske noch nicht erstellt
    ========================= parms ========================
    parms>
```

Die Ergebnisbild-Dimension von 256 und die horizontale Scanrichtung brauchen nicht verändert zu werden. Auch der Wert von **spot** soll 0 bleiben, damit das gesamte Bildfeld ausgewertet wird. Den Namen der Bereichs-Datei hatten wir schon am Anfang definiert, bevor sie in **edit** mit den Polygondaten bestückt wurde. In der Status-Anzeige wird auch ausgegeben, daß noch keine Fahrmaske vorhanden ist. Diese Fahrmaske kann aus diesem Menue heraus im Untermenue **tisch** mit dem Befehl **maske** erzeugt werden. Der Befehl **show** in **tisch** zeigt die Fahrmaske als Bild auf dem Display und gibt gleichzeitig für alle definierten Segmente die Anzahl der zu bearbeitenden Bilder und die minimale Auswertezeit an.

```
    parms> tisch maske
       Bereich 'polygon' geladen
       Maske erstellt
    parms> tisch show
    ========== Segmente ================
          rot     4 Ecken:  Label  1     24492 Bilder

          insgesamt 24492 Bilder
          minimale Auswerte-Zeit:  3:03:41.40  bzw. 1:54:17.76 bei Spot 1
    parms>
```

Die letzte Überprüfung der Parameter zeigt neben der Angabe, daß eine gültige Fahrmaske erstellt wurde, auch die in **bereich** von **edit** eingegeben Film-ID **schuss3357**.

```
    parms> !
    ========================= parms ========================
          schuss :   3357;   - film_id : schuss3357
          name : /h0/INR/IODA/3357A/ergebnis
          vergroesserung:   100;   - Bildweite:  28500,  28500 *10nm
          raster :  25000,  25000  *10nm
          dimension : 256, 256 ;    - Anzahl der Layer: 1
          scan horizontal
          spot : 0
          bereich : /h0/INR/IODA/3357A/polygon
    Maske ist erstellt.
    ========================= parms ========================
    parms>
```

Bevor die Auswertung gestartet werden kann fehlt nur noch eines: die Zuordnung der in **methode** aktuell eingestellten Methode mit ihren Parametern zu dem auszuwertenden Segment mit dem Label 1. Diese Zuordnung sollte nicht vergessen werden. Wenn man beispielsweise aus einer vorangegangenen Auswertung durch **load** im **ioda**-Hauptmenue mit dem Parameter-

satz auch die Methode übernommen hat, müssen die **methode**-Parameter nach einer Änderung wieder an das entsprechende Auswertesegment "connectet" werden.

```
methode> connect 1
   Sigma    connected to 1
methode>
```

6.5 Auswertung und Ergebnis

Nun wird die Auswertung durch Eingabe des Befehls **auswertung** oder einer eindeutigen Abkürzung gestartet. Die Auswertung beginnt mit der Angabe des Datums und der Initialisierung der notwendigen Dateien.

```
ioda> auswertung
   29.11.1994 - 22:04:03  Auswertung wird gestartet fuer 24492 Bilder
   Datei-Initialisierung ..

        - - I O D A - -  Version 1.01 v. 23.11.94 --

        -- Auswertung -- Userinterface --   Master
```

Nach Beendigung der Auswertung wird das Datum und die Gesamt-Statistik ausgegeben. Man erfährt dabei, daß für die Bearbeitung der 24492 Einzelbilder 13 Prozessoren 3h4min55sec lang beschäftigt waren. Dies ergibt eine durchschnittliche Zeit von 0.45sec pro Bild. Weiterhin wird angezeigt, daß diese 13 Prozessoren zu 99.83% mit Rechenarbeit ausgelastet waren, und daß die Summe der Rechenzeiten aller Prozessoren eine Zeit von 1d15h59min57sec ergibt; dies entspricht dem Zeitbedarf, der nötig gewesen wäre, wenn nur ein Prozessor an dem Problem gearbeitet hätte.

```
   30.11.1994 -  1:09:08  Datei-Aufbereitung ..

   Prozessoren |    Jobs | [   all,   job,  trans,   wait ]
      13        24492    [ 12951,  5879,  6620,     5 ]

   Gesamt-Zeit: 3:04:55.13   Jobs: 24492
   Zeit / Job : 0.45 s
   Prozessoren: 13   - Transfers: 16
   Performance:  99.83%

   Transputer-Rechenzeit:  1-15:59:57.31  --> Leistungsfaktor = 12.98

   Statistik: Prozessor [p], Netz [n], Baum [b], gesamt [g], weiter [w] :
```

Nach der Ausgabe der Gesamtstatistik besteht die Möglichkeit weitere statistische Angaben zu erhalten. Die Anleitung dazu ist dem Manual zu entnehmen.

Im Anschluß an die Auswertung und die Betrachtung der Statistik kann durch Eingabe von **w** die Rückkehr ins **ioda**-Hauptmenue erfolgen. Das Programm wird durch **bye** oder das Betätigen der ESCAPE-Taste verlassen.

```
Statistik: Prozessor [p], Netz [n], Baum [b], gesamt [g], weiter [w] : w
ioda> bye
   Wollen Sie IODA wirklich verlassen ?? (j/n) : j
>
```

Man landet nach Beenden des Programmes in dem Arbeits-Directory, von dem aus man IODA gestartet hat, oder in das man innerhalb des Programms mit dem Kommando **dir** gewechselt hat; in unserem Fall dem Directory */h0/INR/IODA/3357A*. Durch den Befehl **ll** wird der Inhalt des Directory angezeigt.

```
> pd
/h0/INR/IODA/3357a
 >ll
                    Directory of . 10:40:36
   Owner   Last modified  Attributes Sector     Bytecount   Name
   -------  -------------  ----------  ------     ---------   ----
    4.9     94/11/30 0109  ------wr   70E7C       131072    ergebnis.L0
    4.9     94/11/30 0109  ------wr   71080          347    ergebnis.prot
    4.9     94/11/29 2204  ------wr   70E78          856    ergebnis.prt
    4.9     94/11/29 2204  ------wr   70E70         1444    ergebnis.save
    4.9     94/11/29 2039  ------wr   70E6C          220    polygon
>
```

In der Liste der im Directory existierenden Dateien stehen neben der in **parms** definierten und in **edit** mit Daten versehenen Bereichsdatei *polygon* noch die durch die Auswertung erzeugten Files, die den Namensrumpf des in **parms** angegebenen **name** besitzen. Es handelt sich um die Ergebnisdatei *ergebnis.L0*, die Auswerte-Protokolldatei *ergebnis.prt*, die für nachfolgende Weiterverarbeitung der Auswerte-Ergebnisse wichtige Parameter enthaltende Datei *ergebnis.prot* und um die nur von IODA lesbare Date *ergebnis.save*, die den gesamten Datensatz für eine Auswertung enthält.

Mit Hilfe eines speziellen Programmes, das durch den Aufruf von **t** in der OS9-Shell gestartet wird, kann die Ergebnismatrix, die im File ergebnis.L0 abgelegt wurde, als quadratisches Bild auf dem Display zur Ansicht gebracht werden. In Abbildung 13 ist die Ergebnismatrix zusammen mit der zur Auswertung benutzten Fahrmaske dargestellt.

```
> t
t> bild
bild> form 256 w
bild> read ergebnis.l0
bild> write
bild> bye
really 'bye' ?? (y/n) : y
>
```

Das Programm **t** ist genauso strukturiert wie IODA; es ist ebenfalls menuegesteuert, besitzt die entsprechenden Hilfen und gibt als Prompt jeweils den aktuellen Menue-Namen vor dem Winkelzeichen aus. Man muß, nachdem das Programm aktiviert wurde, in das Untermenue **bild** verzweigen und da das Format des einzulesenden Bildes angeben (Dimension 256, 1Wort=2Byte pro Pixel). Mit **read ergebnis.l0** (Groß-Klein-Schreibung wird nicht berücksichtigt) wird die Ergebnisdatei eingelesen und durch **write** auf dem Display ausgegeben. Ver-

lassen kann man das Programm durch **bye** oder das Betätigen der ESCAPE-Taste.

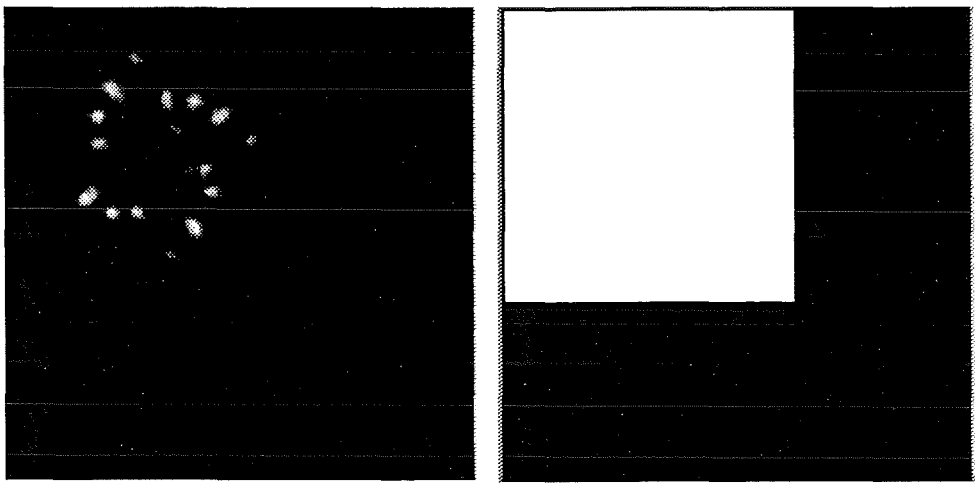


Abbildung 13. Darstellung des Auswertungsergebnisses und der zugehörigen Fahrmaske.
Im linken Bild ist die bei der Auswertung entstandene Ergebnis-Matrix als Grauwertbild dargestellt. Da zur Darstellung nur 1Byte pro Pixel zur Verfügung steht, wurden die Ergebniswerte so skaliert, daß der in der Matrix eingetragene maximale Wert von 977 auf einen Wert von 255 transformiert wird, der im Bild "weiß" entspricht.
Das rechte Bild zeigt die zur Auswertung benutzte Fahrmaske, die aus der Randbeschreibung der Polygon-Datei erstellt wurde. Bei der Auswertung werden auf dem Film nur die Raster-Positionen berücksichtigt, die auf der entsprechenden Stelle der Fahrmaske einen "weißen" Wert haben.

Eine weitere Möglichkeit die Meßergebnisse zu inspizieren oder sogar weiterzuverarbeiten bietet das Programmsystem KHOROS auf der SUN-Workstation [7]. In Abbildung 14 ist die Ergebnismatrix der Auswertung als 3D-Plot mit Hilfe des KHOROS-Programms XPRISM3 dargestellt.

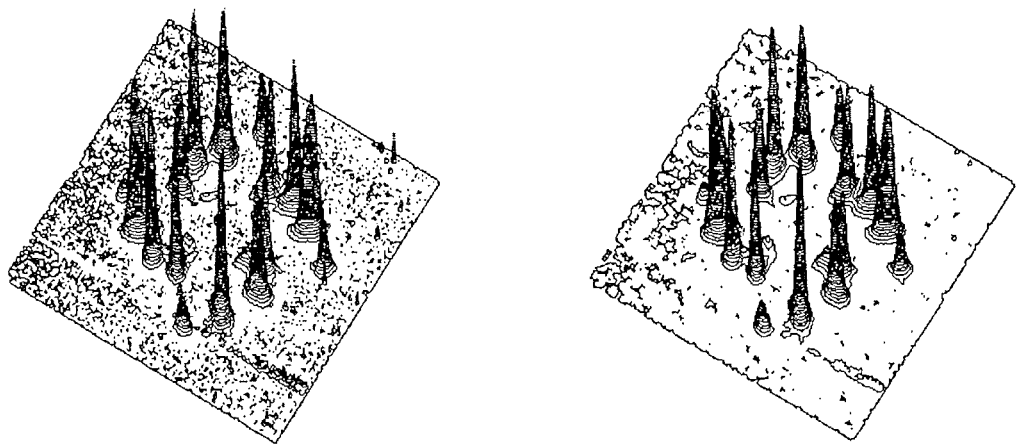


Abbildung 14. Darstellung des Auswertungsergebnisses als 3-dimensionaler Plot.
Im linken Bild wurden die Daten der Auswertung benutzt, um die 3. Dimension des Plots zu erzeugen, während im rechten Bild diese Ionendichte-Werte vor der Darstellung durch das Rangordnungsfilter MEDIAN bearbeitet wurden, um Ausreißer zu eliminieren.

7. Bewertung und Ausblick

Im Einsatz bei weit über hundert Auswertungen hat sich das System gut bewährt. Die zugrunde liegende Konzeption erwies sich als richtig und hielt den Anforderungen der Praxis stand. Sogar in problematischen Fällen, in denen selbst das menschliche Auge Schwierigkeiten mit der eindeutigen Selektion der Lochgrößen hatte, hat es nicht versagt. Ein Beispiel dafür sind die Ergebnisse von zwei Auswertungen eines Films, die in folgender Abbildung dargestellt werden, wobei im einen Fall mit der "Kreis-Methode" alle Löcher und im anderen Fall mit dem "Sigma-Verfahren" nur Löcher einer bestimmten Größe selektiert wurden.

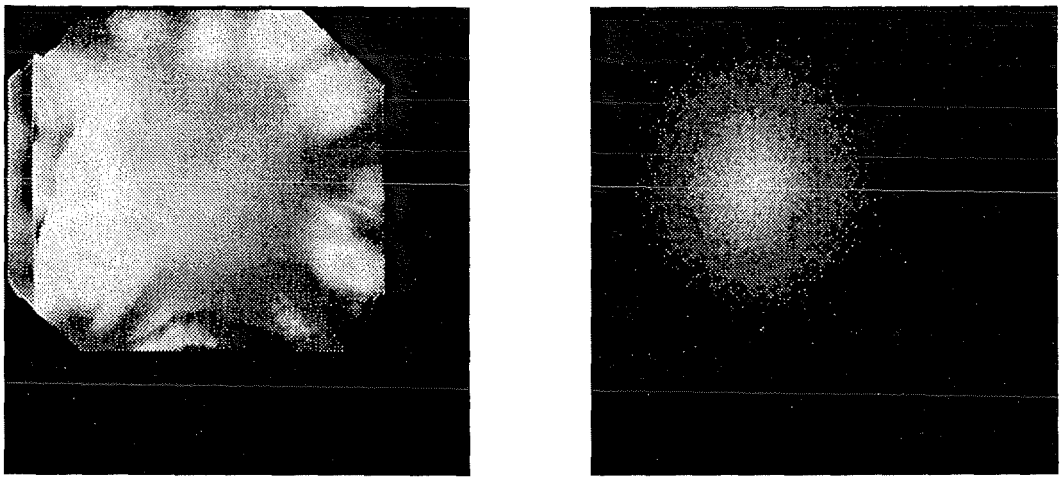


Abbildung 15. Darstellung der Auswertungs-Ergebnisse des Filmes von Schuß 2601.

Das linke Bild zeigt die Ergebnismatrix der Auswertung mit der Kreis-Methode. Diese Methode kann nicht zwischen unterschiedlichen Lochgrößen unterscheiden und klassifiziert alles, was im Zentrum aussieht wie ein Loch als solches.

Das rechte Ergebnis wurde durch Einsatz der Sigma-Methode mit einer Maske und einem ***faktor****-Parameter, womit nur eine definierte Größe selektiert wurde, erzielt.*

Die Bilder wurden zur Darstellung so skaliert, daß die maximalen Einschlagdichten weiß erscheinen. Während beim linken Ergebnis das Maximum bei über 950 Einschlägen pro Mikroskopbild-Fläche liegt, wurden beim rechten Ergebnis im Maximum nur 220 erreicht. Für den Auswertebereich wurde jeweils dieselbe Fahrmaske benutzt.

Bewährt hat sich auch die Handhabung des Systems. Es erlaubt auf einfache Weise, mit Unterstützung des Mikroskops auf einem CR39-Film die für eine Auswertung interessanten Regionen zu bestimmen. Nach Auswahl einer geeigneten implementierten Methode werden die Auswerteparameter durch ausführliche Tests optimiert. Die anschließende bildanalytische Auswertung erfolgt vollautomatisch. Sie besteht aus der Erkennung der Teilcheneinschläge auf der Filmoberfläche und der Bestimmung der Einschlagdichte (Anzahl pro Mikroskopbild-Fläche). Solange die Fokus-Position des Films erhalten bleibt, ändert sich das Erscheinungsbild der Einschläge unter dem Mikroskop nicht und die Löcher können mit ausreichender Sicherheit erkannt werden. Kann selbst durch sorgfältigste Filmpositionierung zwischen den Glasplatten keine einheitliche Fokus-Position für alle Filmregionen erreicht werden, so können für die verschiedenen Fokus-Einstellungen verschiedene Auswertesegmente definiert werden, die mit dem System nacheinander bearbeitet werden können. Auf diese Weise können die Löcher zahlenmäßig vollständig erfaßt werden. Im System ist zwar ein Software-Autofokus

integriert, der auf Bildverarbeitungsmethoden basiert, jedoch wäre der Zeitbedarf zu hoch, um diesen während der Auswertung zu aktivieren. Wird die Dichte so groß, daß die Löcher auf den Bildern zu stark überlappen, kann keine 100%-Auswertung mit den vorhandenen Methoden erwartet werden. Es ist jedoch vorstellbar, in Anlehnung an die in der Physik eingesetzten Zählrohre zur Teilchendetektion, Totzeit-Korrekturen an den Auswertungsergebnissen durchzuführen. Dazu wären zukünftig aber noch umfangreiche Untersuchungen und Tests notwendig.

Zur Verbesserung der gleichzeitigen Klassifikation unterschiedlicher Lochgrößen könnten die begonnenen Untersuchungen zu den Neuronalen Methoden [8] weitergeführt werden.

Durch die menuegesteuerte Benutzer-Schnittstelle ist das System IODA sehr einfach handhabbar. Der logische Menue-Aufbau, die Online-Hilfe, die Abkürzbarkeit der Kommandos und die Status-Abfragen erlauben selbst mit einem einfachen Terminal einen hohen Bedienungskomfort.

Der Aufbau der Hardware aus Standardkomponenten ermöglichte ein kostengünstiges System, das sich auch flexibel aufrüsten ließ. Der Anschluß des VME-Bus-Systems an das Local-Area-Network (LAN) hat sich hervorragend bewährt, da hiermit die Ergebnis-Daten leicht auf eine Workstation zur Weiterverarbeitung und Archivierung gebracht werden können.

Die Aufgabenverteilung im Multiprozessorssystem zur schnellen parallelen Bearbeitung des Problems hat sich als vorteilhaft erwiesen. Durch die Aufteilung in die Bearbeitung jeweils ganzer Bilder ist es möglich, nicht homogene Prozessoren einzusetzen. Wir konnten auch ältere Prozessoren mit niedrigerer Taktrate parallel zu neueren weiter verwenden.

Da in der Zwischenzeit die Entwicklung auf dem Workstation-Markt stürmisch vorangegangen ist, und schnellere Mehrprozessor-Systeme auch hier Einzug gehalten haben, wäre es denkbar, IODA auf Workstations zu portieren. Dies hätte den Vorteil einer homogenen Architektur.

8. Anhang - Implementierungsdetails

8.1 Dateiformate

Dateien der Auswertung

Bei der Auswertung werden eine Reihe von Dateien erstellt. Ihre Namen entsprechen dem Namen der Auswertung, die verschiedenen Dateitypen werden durch die Datei-Extension charakterisiert:

name.L0, name.L1, ...	Layer *n* der Meßergebnis-Matrix
name.prt	Auswertungsprotokoll zum Nachlesen
name.prot	formalisiertes Protokoll zur automatischen Auswertung im INR
name.save	Sicherung der Parameter - nur von IODA lesbar

Meßergebnis-Matrix

Die Meßergebnisse werden in einer quadratischen Matrix der Größe 64, 128, 256 oder 512 abgelegt. Ein Wert ist eine binäre Integer-Zahl von 2 Byte Länge, höheres Byte zuerst. Die Matrix wird zeilenweise in der Datei abgespeichert. Besteht ein Ergebnis aus mehreren Werten, so werden diese layerweise in verschiedenen Dateien abgespeichert, wobei die Datei-Endung den Layer charakterisiert.

Wird die Matrix durch den Meßbereich nicht voll ausgenutzt, liegt das Ergebnis in der linken oberen Ecke.

Auswerte-Protokoll

Alle wichtigen Aktionen einer Auswertung werden in einer Datei protokolliert. Dazu gehören auch alle relevanten Parameter. Wiederaufnahmen werden, falls sich die Datei noch im Directory befindet, hinten angehängt.

Beispiel:

```
1.> list dddtest.prt
   15.11.1994 - 21:01:41  Datei geoeffnet


      - - I O D A - -  Version 0.33 v. 17.9.93 --


      ========== Parms ==========

      schuss :  3357;  - film_id : 3357
      name : /h0/INR/IODA/3357/dddtest
      vergroesserung:  100;  - Bildweite:  14250,  14250 *10nm
      raster :  25000,  25000  *10nm
      dimension : 256, 256 ;    - Anzahl der Layer: 1
      scan horizontal
      spot : 1
```

```
        bereich : /h0/INR/IODA/3357/3357polygonlage6


        ========== Kamera ==========

        Gain: 127;    Offset: 128

      15.11.1994 - 21:01:41  Start Auswertung -- 24460 Bilder

      15.11.1994 - 21:01:42  Start Segment -- Label 1 -- 24460 Bilder

        ========== Methode ==========

        Verfahren: Sigma
        dimension: 256
        faktor:   4.00
        maske:    #7

      15.11.1994 - 21:07:34  ***** Abbruch bei ( 141, 12 ) - Job 1160
      15.11.1994 - 21:07:37  Ende Segment -- Label 1
      15.11.1994 - 21:07:37  Datei-Aufbereitung
      15.11.1994 - 21:08:08  Datei geschlossen
  1.>
```

Protokolldatei zur rechnergestützten Auswertung

Zur rechnergestützten automatischen Weiterverarbeitung der Ergebnisse, werden die Parameter formalisiert in einer lesbaren Datei ausgegeben.

Beispiel:

```
1.> list dddtest.prot
    SCHUSSNR:  3357
    FILM_ID : 3357
    NAME   : dddtest
    DATUM  : 15.11.1994
    COMMENT : ?
    MIK_VERG: 100.0 000.0 000.0
    MIK_PARM: ?
    BILDWEIT:    14250    14250
    RASTER :    25000    25000
    SCAN   : horizontal
    AREA_DAT: 3357polygonlage6
    AREA_PKT:  24460
    LAYERS : 1
    LAYER_NR: 1
    METHODE : Sigma
    DIMS   :  256  256
    FMT    : 2
```

Bereichsdatei

Die Bereichsdatei wird vom Programm ioda_ed aufgebaut. Sie sollte nicht per Hand manipuliert werden. Für Notfälle sei der Aufbau hier angegeben.

Beispiel:

```
film3357
     4939600    4039900  0
    11295000    4009950  0
    11306675   10358775  0
    10088825    9139225  1
     6188875    9139225  1
     6188875    5263425  1
    10088825    5263425  1
     7810525    6601425  1
```

1. Zeile	Filmid : beliebiger String
2-4. Zeile	x-Koordinate y-Koordinate 0 : die drei Justierungspunkte
weitere Zeilen	x-Koordinate y-Koordinate Label : Punkte eines Segments

Fahrmaske

Die Fahrmaske ist eigentlich nur eine interne Datenstruktur, die jedoch auch ausgegeben werden kann. Es handelt sich dann, wie bei der Meßergebnis-Matrix um ein quadratische Matrix der Größe 64, 128, 256 oder 512, die zeilenweise ausgegeben wird. Der Einzel-Wert ist eine binäre 1 Byte-Größe.

Die Maske entspricht der Auswerte-Matrix und enthält an jeder Rasterposition, die ausgewertet werden soll, einen Wert ungleich 0. Die Werte entsprechen dem Label der verschiedenen Segmente, die im Programm **ioda_ed** zugeordnet wurden.

Sigma-Maske

Bei der Kreation einer eigenen Maske ist darauf zu achten, daß die Dimension ein ungeradzahliger Wert ist. An den Positionen der Nachbarschafts-Pixel, die zur Mittelwert- und σ-Berechnung beitragen sollen muß eine 1 zu stehen, sonst Null.

Beispiel:

```
maske 5 5
1 0 0 1 1
0 0 0 0 1
0 0 0 0 0
1 0 0 0 0
1 1 0 0 1
```

In der ersten Zeile muß das Keywort **maske** und die ungerade x- und y-Dimension *n* (gleich, da nur quadratisch!) durch Leerzeichen getrennt stehen, in der oder den folgenden Zeilen die *n*x*n* Maskenwerte 0 oder 1

8.2 Transputer-Farm

Aufbau

Die Multiprozessor-Architektur wurde mit Transputern realisiert. Dies sind 32-Bit-Mikroprozessoren mit Floating-Point-Unit und 4 seriellen DMA-Kanälen. Die 4 Kanäle eignen sich besonders zur Vernetzung der Prozessoren, da Kommunikationsbefehle schon Bestandteile der Maschinensprache sind.

Eingesetzt sind 'TRAM-Module', ca. 10x3cm große Platinen mit 16 externen Pin-Verbindungen, die jeweils einen Inmos T805 25MHz und 4MB RAM enthalten. Aufgrund des kompakten Aufbaus lassen sich jeweils 8 dieser Module auf einem Motherboard in einem VME-Slot unterbringen. Als Motherboards werden 2 INMOS B014-Karten verwendet, die zusammen für 16 Transputer konfiguriert sind (vorhanden sind 8). Zusätzlich wird als Verbindungs-Rechner eine INMOS B016-Karte verwendet, die mit einem T801 und dual-ported-RAM versehen ist. Dies gestattet, daß der Master-Rechner über den VME-Bus direkt in den Speicher des Verbindungs-Transputers zugreifen kann.

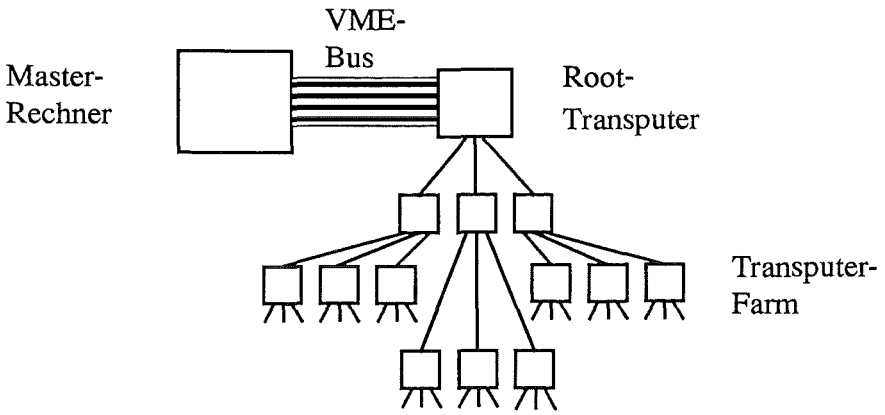


Abbildung 16. Aufbau der Transputerfarm

Funktion

Die Transputer sind so verschaltet, daß der Aufbau einen trinären Baum ergibt (siehe Abbildung 16). Von den vier vorhandenen Verbindungsleitungen (Links) ist eine mit einem übergeordneten Transputer verbunden, während die drei übrigen mit untergeordneten Transputern verbunden sind.

Die Wurzel des Baumes bildet ein spezieller Transputer, in dessen Speicher über den VME-Bus direkt zugegriffen werden kann. Da er die Gesamtlast der Kommunikation des ganzen Baumes trägt, bearbeitet er keine Aufträge, sondern fungiert nur als Verteiler.

Die Transputer enthalten kein übliches Betriebssystem, sondern nur zusätzlich zur Anwendungs-Software ein sogenanntes Routing-Programm, das den Transport von Aufträgen und Ergebnissen steuert. Sie werden am Anfang wellenartig mit identischen Programmen geladen, wobei der Lader automatisch die Konfiguration erkennt. Das Hinzufügen weiterer Transputer

erfordert keinerlei Software-Änderung.
Die Verteilung der Aufträge ist als "Farm"-Konzept realisiert: Ein Auftrag wird sequentiell allen Prozessoren angeboten. Ist ein Prozessor frei, greift er sich den Auftrag, sonst läßt er ihn passieren. Für den Baum wurde der Routing-Algorithmus etwas modifiziert. Jeder freie Transputer schickt eine "Frei"-Meldung in Richtung Wurzel. Durchlaufene Transputer merken sich den Ast, aus dem die Meldung kam. Bekommt nun ein nicht freier Transputer einen Auftrag, so schickt er ihn in einen Ast weiter, aus dem er eine "Frei"-Meldung bekommen hat.

8.3 Grundlagen der Menue-Steuerung

Einführung

Für das Programm-System IODA wurde als User-Interface eine Menue-Steuerung gewählt. Die Struktur dieser Oberfläche ist so allgemein, daß sie sich gut erweitern läßt, und in dieser Form auch für ähnliche Programme einsetzen lassen würde. Deshalb wird sie im folgenden etwas genauer erläutert.
Programm-Systeme, die wie IODA aus mehreren einzeln aktivierbaren Moduln bestehen, benötigen eine komplexe Schnittstelle zum Bediener. Diese soll übersichtlich sein, ungeübten Benutzern genügend Hilfestellung bieten und geübte nicht unnötig behindern.
Diese Forderungen werden am besten durch eine Menue-Steuerung erfüllt. Da jedoch kleinere Rechner mit Betriebssystemen wie OS-9 nur selten mit komfortablen grafischen Schnittstellen ausgerüstet sind, muß sich die Bedienung auf die Standard-Terminals beschränken. Unter diesen Voraussetzungen eine einfache Bedienung zu ermöglichen, war der Leitgedanke für die vorliegende Benutzer-Schnittstelle.

Grundsätzliches

Alle Funktionen des Programmes werden zu Funktionsblöcken zusammengefaßt. Jeder Funktionsblock wird einem Menue zugeordnet. Ein Menue entspricht einer Bildschirm-Seite und enthält alle im Funktionsblock möglichen Kommandos. Jedes Kommando wird kurz in einer Zeile beschrieben.
Alle Menues sind in einem Haupt-Menue zusammengefaßt, in dem sich der Benutzer nach dem Aufruf befindet. Von hier aus kann er die Unter Menues anwählen. Außerdem enthält das Haupt-Menue noch einige allgemeine Funktionen.
Bei sehr großen Programm-Systemen kann das Konzept erweitert werden, indem ein Kommando in das Haupt-Menue eines weiteren Programmes führt. Dies ist dann entsprechend aufgebaut und führt nach Abschluß an die Aufruf-Stelle des ersten zurück.

Prinzipielle Funktion eines Menues

Der Benutzer erfährt vom Programm in welchem Menue er sich gerade befindet durch den Prompt

```
menue-name>
```

Das Kommando '?' erlaubt ihm einen Überblick über die in diesem Menue direkt möglichen Kommandos.
Für das ausgesuchte Kommando kann er sich mit '<kommando> ?' die Syntax zeigen lassen.

Falls dies nicht ausreicht, kann er mit 'help <kommando>' weitere Informationen abrufen.

Auch für den gesamten Funktionsblock des Menues existiert eine Beschreibung, die mit 'help' aufgerufen werden kann. Ist die Information länger als ein Bildschirm-Inhalt, stoppt die Auflistung automatisch. Der Leser kann dann entscheiden, ob und wann er weiterlesen will - mit <CR> -, oder ob er abbrechen will - mit <ESC>.

Alle Kommandos können abgekürzt werden. Entstehen dabei Mehrdeutigkeiten, wird das erste im Menue entsprechende Kommando ausgeführt. Die Eindeutigkeit läßt sich durch '? <Abkz.>' überprüfen.

Beim Aufruf des Programms befindet sich der Benutzer im Haupt-Menue, von dem er in die Unter-Menues gelangt. Diese sind in der Menue-Übersicht durch ein '>' gekennzeichnet. Durch eine Leer-Eingabe im Unter-Menue kehrt er wieder in das Haupt-Menue zurück. Am Prompt kann er jederzeit erkennen, in welchem Menue er sich befindet.

Die Befehle der Unter-Menues können auch vom Haupt-Menue direkt aufgerufen werden, indem als erstes Wort vor dem Kommando der Name des Unter-Menues eingegeben wird.

Ist ein Kommando im Unter-Menue nicht bekannt, so wird es automatisch dem Haupt-Menue übergeben, und von hier aus ausgeführt. Auf diese Weise ist von allen Unter-Menues die Ausführung der Befehle des Haupt-Menues möglich. Ebenso können Befehle von anderen Unter-Menues genau wie vom Haupt-Menue direkt aufgerufen werden.

Alle Kommandos und interaktive Abfragen müssen mit <CR> abgeschlossen werden. Danach sind sie nicht mehr rückgängig zu machen. 'Gefährliche' Kommandos müssen zusätzlich bestätigt werden.

An einigen Stellen lassen sich Kommandos mit '<CTRL> C' abbrechen.

Allgemeine Kommandos

Im folgenden werden einige Kommandos erklärt, die im Haupt-Menue der meisten Programme existieren, wenn sie mit dieser Bedienung ausgestattet sind.

? und ! - Menue-Auskunft

Alle Kommandos, die zum aktuellen Menue gehören, werden nach der Eingabe von '?' zusammen mit Kurztexten aufgelistet. Im Haupt-Menue sind die Unter-Menues durch ein '>' gekennzeichnet.

'? <kommando>' listet den Kurztext für ein Kommando.

Dieser Befehl läßt sich als '? <abkz.>' auch zur Überprüfung der Eindeutigkeit einer Abkürzung einsetzen.

'<kommando> ?' listet die Syntax eines Kommandos. Begriffe, die in '<>' ein- geschlossen sind, sind durch einen aktuellen Wert zu ersetzen. Argumente in '[]' können weggelassen werden.

Die Syntax aller Kommandos zeigt '??'.

Wird ein Programm-Teil zum Setzen von Zuständen (Variablen) verwendet, so lassen sich diese Zustände durch '!' zeigen.

Bye - Beenden des Programms

Das Ende-Kommando im Programm ist 'bye'. Vor Verlassen des Programms wird noch einmal nachgefragt, damit Daten nicht unbeabsichtigt verlorengehen.

End-of-file ist mit 'bye' gleichbedeutend, d.h. auch das Drücken von '<ESC>' oder '<CTRL> C' beendet das Programm.

Dir - Ändern des aktuellen Directories

Mit dem Kommando 'dir <directory>' ist das Ändern des Arbeits-Directories für das Programm möglich. Beim Aufruf ohne Parameter wird der Name des aktuellen Directories ausgegeben.

Shell - Ausführen eines OS-9-Kommandos

Es ist möglich, aus dem laufenden Programm heraus System-Kommandos ablaufen zu lassen. Jedoch sollte darauf geachtet werden, daß keine Kommandos gegeben werden, die das Programm beeinflussen, z.B. Löschen von notwendigen Dateien. Ein System-Fehler ist sonst unvermeidlich.

Mit 'shell <kommando>' wird eine neue OS-9-Shell gestartet und ihr der Befehl zur Ausführung übergeben. Wird kein Kommando angegeben, so wird eine interaktive Shell gestartet.

Exec - Bedienung über eine Datei

Alle nicht interaktiven Kommandos lassen sich auch aus einer Datei heraus ausführen. Dabei muß in dieser Datei jeder Befehl in einer eigenen Zeile stehen. Kommentar-Zeilen sind möglich, sie müssen mit '#' in der ersten Spalte beginnen.

Ablauf- und allgemeine Fehler-Meldungen werden nach 'stdout' geschrieben. Fehler werden für den Ablauf ignoriert.

In diesen Ausführungsmodus gelangt man entweder durch den Aufruf des Programms mit der Datei als Parameter oder aus dem interaktiven Modus mit dem Kommando 'exec <datei>'.

Bei der Ausführung sind weitere 'exec's bis zur Tiefe 8 möglich. Es wird jeweils eine neue Exec-Ebene eröffnet. Dabei ist zu beachten, daß einige globale Zustände nur für die jeweilige und neuere Exec-Ebenen gelten können. Eine Exec-Ebene endet entweder durch das Datei-Ende oder das Kommando 'bye'.

Exec-Ebenen können auch interaktiv eröffnet werden, indem bei 'exec' keine Datei angegeben wird. Jede Ebene muß dann explizit durch 'bye' abgeschlossen werden.

Save und Load - Sicherung des Arbeitszustandes

Das Programm besitzt eine Arbeits-Sicherungsfunktion. Die lokalen Daten aller Menues können abgespeichert und wieder geladen werden. Jedoch werden System-Zustände, die durch das Aufrufen von Funktionen entstanden sind, nicht wiederhergestellt.

Mit den Befehlen 'save <datei>' und 'load <datei>' ist das Sichern und Wiederherstellen der Menue-Daten möglich. Wird der Dateiname nicht angegeben, so legt das Programm die Daten in einer Datei mit default-Namen im aktuellen Directory ab.

Das interne Format der Sicherungs-Datei ist von der Version abhängig. Es ist deshalb nicht selbstverständlich, daß die Datei nach Wechsel auf eine neuere Version ohne weiteres gelesen werden kann !!

Eine weitere automatische Sicherung erfolgt nicht! Bei Verlassen des Programms sind alle Inhalte verloren, beim Aufruf werden wieder default-Werte gesetzt.

Help-Feature und Erweiterung der Texte

Das Programm ist mit einem umfangreichen online-Help ausgestattet. Alle Menues werden mit 'help' unterstützt.

Das Kommando 'help' allein ruft für das jeweilige Menue eine Basis-Übersicht auf. Weiterhin können mit 'help <begriff>' zusätzliche Informationen für den jeweiligen Begriff abgefragt werden. Groß- und Kleinschreibung ist dabei nicht relevant.

Die möglichen Begriffe lassen sich mit 'help !' erfragen. Man erhält eine Liste der Begriffe, wobei die in derselben Zeile aufgeführten Worte die gleiche Hilfs-Information erzeugen. Die Begriffe lassen sich abkürzen.

Alle Texte befinden sich im HELP-Directory der Source und können dort auch direkt mit dem Programm 'info' gelesen werden.

Der Text läßt sich einfach ergänzen und um weitere Begriffe erweitern. Help-Dateien sind normale, lesbare Dateien, die mit einem Editor bearbeitet werden können. Die jeweils zu einem Menue gehörende Datei ist durch spezielle Zeilen, die mit '@' beginnen, in Bereiche aufgeteilt. Jeder Bereich entspricht einem oder mehreren Begriffen, die ab der Spalte 3 in der mit '@' beginnenden Zeile stehen. Bei Aufruf eines dieser Begriffe wird der folgende Text bis zur nächsten mit '@' beginnenden Zeile auf dem Bildschirm gezeigt.

Für die bessere Darstellung beim Kommando 'help !' läßt sich hinter den Begriffen nach einem ';' noch ein Kommentar anfügen.

Beispiel:

```
<datei-beginn>
 ..
 ..... hier steht der Text für 'help' ( ohne Begriff )
 ..
@ begriff1 begriff2        ;Kommentar zu den Begriffen
 ..
 ..... hier steht der Text der bei 'help begriff1' oder
 .. 'help begriff2' ausgeben wird
 ..
@ begriff3                 ;Kommentar zu Begriff 3
 ..
 ..... hier steht der Text für 'help begriff3'
 ..
<datei-ende>
```

Fehler

Beim Auftreten eines Fehlers erfolgt eine Fehlermeldung. Sie folgt dem OS-9-Standard und enthält eine Fehlernummer und einen Kurztext, z.B

```
Error 000:164 no permission
```

Die Fehler-Nummer klassifiziert den Fehler. Die Zahl vor dem ':' gibt die Fehler-Kategorie an.

Es treten zwei Kategorien auf:

- Bedienungsfehler - Fehlerklasse >= 64 :

Sie entstehen durch Fehlbedienung und werden so weit wie möglich abgefangen. Für die Kor-

rektur ist der Benutzer verantwortlich.

- System-Fehler - Fehlerklasse < 64 :

Diese Fehler sollen prinzipiell nicht auftreten. Tritt dennoch ein System- Fehler auf, und kann der Benutzer mit der Meldung nichts anfangen, sollte er nicht weiterarbeiten, sondern einen Spezialisten befragen.

9. Literaturverzeichnis

[1] W.Bauer, H.J. Bluhm, P. Hoppé, H.U. Karow, H. Laqua: *Erzeugung und Fokussierung von Leichtionenstrahlen mit Impulsenergien von 50kJ und Leistungsdichten um 1TW/ cm^2*, KfK-Nachrichten 1/92, S. 3-12 (1992)

[2] W. Bauer, H. Baumung, H.J.Bluhm, H.U. Karow: *Diagnostische verfahren zur Untersuchung intensiver Ionenstrahlenund der Strahl-Target-Wechselwirkungen*, KfK-Nachrichten 1/92, S. 19-28 (1992)

[3] G. Somogyi, I. Hunyadi: *Etching Properties of the CR-39 Polymeric Nuclear Track Detector*, Proc. 10th Int. Conf. of SSNTD, Lyon 1979, Pergamon Press, Oxford, N.Y., S.443-452 (1980)

[4] B. Bürg, H. Guth, A. Hellmann: *Einsatz der Bildanalyse zur Diagnostik Intensiver Ionenstrahlen*, KfK-Nachrichten 2/92, S. 85-94 (1992)

[5] N. Otsu: *A Threshold Selection Method from Gray-Level Histogram*, IEEE Trans., Vol. SMC-9, PP. 62-66 (1979)

[6] H. Guth, A. Hellmann, B. Bürg: unveröffentlichter Bericht, Kernforschungszentrum Karlsruhe (1990)

[7] The Khoros Group: *Khoros Manual, Version 1.0*, Department of Electrical and Computer Engeneering, University of New Mexico, Albuquerque, NM87131 (1990)

[8] B. Flach, H. Guth, R. Osterland: *Lokale Neuronale Filter*, 14. DAGM-Symposium Dresden, 14.-16.10.1992, Informatik Aktuell, Springer Verlag, S. 323-328, (1992)