ChaosPro

Martin Pfingstl

COLLABORATORS			
	TITLE : ChaosPro		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Martin Pfingstl	July 22, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

ChaosPro

1.1 Inhaltsverzeichnis

Inhaltsverzeichnis

1.2 Vorwort

```
I. Einleitung

1.1 Vorwort

Was? Noch ein Programm zur Fraktalerzeugung?

Eigentlich könnte man meinen, daß es schon genug Fraktal\generatoren auf dem Amiga
gibt. Doch riskiert man einen Blick auf andere Computersysteme und testet z.B.
FractInt auf den PC's (nun auch auf dem Amiga...), so erkennt man schnell, daß es ↩
   nicht recht weit her ist mit den
vorhan\denen Fraktal\programmen: Keines von ihnen besitzt so viele Fraktal\typen
und keines von ihnen erlaubt die Ein\stellung von so vielen Fraktal\parametern.
Nun, sicher ist auch FractInt keineswegs perfekt, aber als Anregung allemal zu
gebrauchen...
```

1.3 Warum sollte ich dieses Programm benutzen?

```
1.2 Warum sollte ich dieses Programm benutzen?

Oder anderes gefragt: Was hat dieses Programm denn besonderes zu bieten gegenüber
anderen Fraktalprogrammen? Nun, wenn Sie mit den bisherigen Fraktalprogrammen ↩
   zufrieden
waren und Sie niemals an deren Leistungsfähigkeit gestoßen sind, dann halte ich es ↩
    für
```

```
wahrscheinlich, daß Sie mit einem dieser anderen Fraktalprogramme vielleicht ←
    besser bedient
sind. Dieses Programm erscheint Neulingen durchaus etwas verwirrend, da es relativ ←
    leistungsfähig
ist. Wem es einfach nur darum geht, ein paar Fraktalbilder auf den Monitor zu ←
    bringen,
für den ist dieses Programm schlichtweg etwas überdimensioniert. Man kauft ja ←
    Brilliance und
DPaintIV AGA auch nicht nur, um Icons zu malen...

Folgend ein paar Features des Programms:

(Inspiriert von Mand2000Demo, FractInt, MisterM, MandelMania, Fractal Dynamics,
Slicer, MultiFractals, MandelMountains, Fractal V1.3, MandelPlot 24, Mandelsquare,
SmartFractal, LyapunoviaV1.5, CloudsAGA, KFP)

- Multiwindowing
Alle Fraktale werden in Windows gezeichnet, die man einfach mit dem Sizegadget ←
    vergrößern
kann.

- Multitasking
Für die Berechnung eines jeden Fraktals wird ein Task geschaffen, d.h. es können ←
    mehrere
Fraktale gleichzeitig berechnet werden.

- Echtzeiteffekte
Die Änderung eines Parameters wirkt sich sofort aus.

- Click and Zoom
Durch Doppelklick auf eine Stelle wird hineingezoomt.

-Ausschnittverschiebung
Der Berechnungsausschnitt kann während der Berechnung verschoben werden, einfach ←
    mit der
Maus das Fraktal anklicken, oder Cursortasten benutzen, oder den Joystick in Port
2...

- Systemkonformität
Programm läuft nach Aussage meiner Betatester bisweilen auf:
 - Picasso
 - Piccolo
 - GVP EGS110/24
 - GVP Spectrum
 - ECS/OCS/AGA
 - Merlin

Es läuft ab OS2.0 bis OS3.1, mit einem Screenmoderequester kann man die Auflösung
wählen.

- Formeleditor
Für alle, die mal ein bischen rumprobieren wollen - in meinem Programm allerdings ←
    steht
dafür ein Stringgadget bereit, in das man die Formel einfach eintippt, nicht wie ←
    bei FractInt, wo man
ein ASCII-File ändern und das Programm neu starten muß.
```

- mehrere Fraktaltypen
  - Juliamenge
  - Mandelbrotmenge
  - Bifurkationsdiagramme (Verhulst)
  - Dynamische Systeme
  - Plasma
  - Lyapunov-Räume

- Parameter
Es stehen je nach Fraktaltyp bis zu 3 Parameterwindows bereit.

- logischer Programmaufbau
Schlimmstes Beispiel eines Programms, das so eigentlich nicht existieren dürfte: FractInt...

- 3D-Transformationen
In 3 weiteren Parameterwindows können allerlei Parameter eingestellt werden. Es muß natürlich gesagt werden, daß
die Wahl der richtigen Parameter etwas Geduld erfordert. Doch das Multitasking des Programms (die sofortige Auswirkung
der Änderung eines Parameters) hilft hierbei enorm. Man sieht sofort, ob nun der Wert paßt
oder nicht (einen schnellen Amiga vorausgesetzt...:-)).

- Animationen
Nicht bloß simple Zoom-in Movies, sondern auch Zoom-out-Movies, oder irgendwelche
Animationen bzgl. irgendeines Parameters, der sich dann kontinuerlich ändert. Natürlich
können sich auch mehrere Parameter ändern. Wie wäre es mit einer 3D-Animation in eine
Juliamenge hinein, deren Parameterwert c sich dabei ändert? Nichts ist unmöglich, sofern genug
Fantasie vorhanden ist.

- 24 Bit
Die Fraktale können in 24 Bit Farbtiefe abgespeichert werden.

- Online-Hilfe
Natürlich kontextsensitiv ;-)

- Locale-Unterstützung
Warum auch nicht?

- Arexx-Interface
Ich konnte es einfach nicht lassen ;-)

- und noch ein paar Kleinigkeiten, die mir jetzt so kurz einfallen:
+ Filename-Multiselect (wirklich eine Kleinigkeit)
+ Menu-Multiselect (sollte eigentlich selbstverständlich sein)
+ Farbrad unter OS3.0 beim Palettenbearbeiten
+ Abspeichern von Bildern ins Clipboard
+ Fontsensitivität

usw.

1.4 Systemvoraussetzungen

```
1.3 Systemvoraussetzungen

Als ich dieses Programm geschrieben habe, stand ich oft vor der Entscheidung, ob
ich ein Feature weglassen sollte, damit das Programm auch auf kleinen Amigas läuft ↩
    ,
oder ob das Programm eben höhere Systemanforderungen stellen sollte. Ich habe mich
für letzteres entschieden, da ich finde, daß es im Jahre 1994 an der Zeit ist,
daß die Amigauser einsehen, daß man mit einem 7.09 Mhz-68000 Amiga mit 1 MB RAM
und einem Diskettenlaufwerk nix mehr anfangen kann...
Wenn ich das Programm so geschrieben hätte, daß es auch auf kleineren Amigas liefe ↩
    , dann
würde es in dieser Form einfach nicht existieren, denn die Anpassung hätte viel zu ↩
     viel Zeit
gekostet. Herausgekommen wäre irgendein anderes Fraktalprogramm, das sich kaum von ↩
     den
existierenden unterschieden hätte. So hatte ich Zeit, um endlich einmal 24 Bit,  ↩
    Animationen,
3D-Transformationen, etc. in ein Fraktalprogramm einzubauen.

Das Programm benötigt mindestens einen 68020 mit mathematischem Koprozessor.  ↩
    Aufgrund des
internen Multitaskings kommt es auch zu einer ziemlichen Überfüllung des  ↩
    Bildschirms mit Windows,
was sich leider nicht vermeiden läßt ( oder doch? ). Aus diesem Grund ist eine  ↩
    höhere
Auflösung mit mehr Platz empfehlenswert.
Die vielen Windows tragen natürlich auch nicht gerade zur Beschleunigung des  ↩
    Systems
bei, so daß hier ein schneller Amiga doch auch eine Daseinsberechtigung hat.
Einhergehend mit der Leistung ist auch eine ziemlich hohe Anforderung an den  ↩
    vorhandenen
RAM-Speicher. Ab ca. 2 MB kann man damit arbeiten, doch mehr auch nicht...
Auch hier gilt wieder: Je mehr, desto besser.
Für die Betriebssystemversion gilt natürlich: Ab OS2.0 läuft das Programm, eher
natürlich nicht.
```

1.5 Installation

```
1.4 Installation und Deinstallation

Die Installation wird mittels des Installers von Commodore erledigt.
Wer es per Hand machen will, bitte sehr:
1. Kopieren Sie die reqtools.library ins libs: Verzeichnis, falls sie dort noch
nicht vorhanden sein sollte. Benötigt wird V38 oder höher.
2. Kopieren Sie das Verzeichnis ChaosPro/ mit allen seinen Unterverzeichnissen
irgendwohin. Lassen Sie alles beisammen, kopieren Sie nichts herum.
3. Kopieren Sie den Inhalt des Fonts-Verzeichnisses in ihr FONTS: Verzeichnis..
.

Damit ist die Installation bereits beendet. Zur Anpassung an Ihr System starten  ↩
    Sie bitte
das Preferences-Programm.
```

Falls Sie gerade die Datei 'deutsch.guide' betrachten, wundern Sie sich nicht, ↩
wenn manche "Nodes"
nicht gefunden werden und AmigaGuide deswegen Fehlermeldungen bringt. Diese Datei ↩
wird
vom Preferences-Programm in die Datei ChaosPro.guide konvertiert. Hierbei werden ↩
Links zu unbekannten Nodes in einer "magischen" Weise gelöst (Eigen\ ↩
implementation des
fehlenden Lokali\sierbar\keits\features).

1.6 Autor

1.5 Autor

Adresse:
Martin Pfingstl
Dorfen 16 1/5
84508 Burgkirchen
Tel.: Semesterferien: 0 86 79 / 62 41 Semester: 089 / 28 44 91
(Deutschland)

EMail: pfingstl@informatik.tu-muenchen.de
(während der Semesterferien kaum zu erreichen)

Bin sehr dankbar für:
- Bugreports
- Verbesserungsvorschläge
- Kommentare
- Registrierungen (quatsch! Untersteht euch ;-))

1.7 Konzept

II. Programmbeschreibung

2.1 Das Programmkonzept

...ist nicht neu, aber äußerst praktisch: Multitasking und Multiwindowing überall.
Das heißt, es können so viele Fraktale gleichzeitig berechnet werden wie der ↩
Benutzer
wünscht. Es kann gleichzeitig die Farbpalette bearbeitet, eine Animation berechnet
werden, während ein Arexx-Programm irgendwas anderes im Programm steuert.
Es kann die Online-Hilfe angezeigt werden und während man in ihr was liest, kann
das Gelesene sofort praktisch getestet werden.

1.8 PicTask

2.2 Die verschiedenen Windows

2.2.1 PicTask-Window

Fraktalbilder
- Im Sichtfenster mit der Überschrift Fraktalbilder werden sämtliche dem Programm augenblicklich bekannten Fraktal\bilder angezeigt. Zu jedem dort vorhandenen
Bild existiert eine Datenstruktur, die sämtliche Parameter, die zur Berechnung des Fraktalbildes nötig sind, enthält. Zu Beginn des Programms werden alle Fraktale, ↩ die im Verzeichnis
'FractPic' zu finden sind, eingeladen und hier angezeigt.
Siehe dazu auch Kapitel 2.6 Programmverzeichnisse.

Bildname
- Direkt darunter befindet sich ein String-Gadget, in dem der Name des Bildes geändert werden kann. Damit die Änderung wirksam wird, muß das Gadget durch ↩ Drücken
der Return-Taste verlassen werden.

Bild löschen
- Durch dieses Gadget wird der angewählte Eintrag gelöscht.

Bild berechnen
- Etwas mehr tut sich, wenn man auf den Knopf 'Bild berechnen' klickt. In diesem Fall wird das Bild in die Taskliste aufgenommen, ein Window geöffnet und darin das Fraktal gezeichnet. Wie der Name des Sichtfensters vermuten läßt, wird dazu ein separater Task geschaffen, der die Berechnung komplett übernimmt. Dieser läuft mit einer um 1 niedrigeren Priorität, so daß die Berechnung die Programmsteuerung nicht beeinflußt, wohl aber die Steuerung die Berechnung.

Bild duplizieren
- Unter Umständen will man, ausgehend von einem Fraktalbild und seinen Daten, ein paar Parameter verändern ohne sich die alten Werte merken zu müssen. Aus diesem Grund existiert durch den Knopf 'Bild duplizieren' eine Möglichkeit, ein Bild zu klonen, also einen weiteren Eintrag in der Liste einzufügen, dessen Datenstruktur eine Kopie des gewählten Eintrages ist.

Windows schließen
- Dieses Gadget schließt alle zum aktuellen Fraktal gehörigen Windows, löscht also ↩ den Task, das
Fraktalwindow und alle Parameterwindows.

Previewbreite/-höhe einstellen
- Hiermit wird die Größe des Ausschnitts eingestellt, der vorab berechnet wird.
Der Ausschnitt wird mittig plaziert, eben da, wo mit der größten ↩ Wahrscheinlichkeit
die interessanten Teile des Fraktals versteckt sind. Extreme Werte, seien sie ↩ klein
oder groß, führen dazu, daß kein Preview berechnet wird. Bei manchen Fraktaltypen ↩ ist
die Previeweinstellung wirkungslos, da das Zeichnen eines kleinen Ausschnitts ↩ entweder
keinen Sinn macht oder gar nicht möglich ist.

Bildeinstellungen

3D-Puffertyp
Für Julia- und Mandelbrotmengen kann eine 3D-Transformation gewählt werden. Um nun gutaussehende Bilder zu ermöglichen, ist es möglich, hier einen Puffer für die 3D- ↩ Darstellung

zur Verfügung zu stellen, so daß die 3D-Bilder nach der Berechnung in 24 Bit Farbtiefe
abgespeichert werden können. Zur Beeinflussung des Aussehens der 3D-Bilder existieren
im 3D-Parameterwindow Nr. 3 noch 2 Gadgets. Diese bestimmen, wie sich das einfallende Licht
auf die Farben des Fraktals auswirken.

Puffertyp
- Es stehen 3 verschiedene Puffertypen zur Auswahl:
1. Gar kein Puffer: Diese Methode ist speichersparend. Allerdings können dann keine
3D-Darstellungen berechnet werden, da die Routinen dafür einen Puffer voraussetzen. Auch das Abspeichern als IFF-ILBM-Bild
ist nur in der aktuellen Screentiefe möglich.
2. 16Bit-Int: Hier wird für jeden Punkt ein Wort (16Bit) reserviert, in dem der berechnete Wert abgelegt wird.
Hier kann dann eine 3D-Ansicht gewählt werden. Außerdem ist es möglich, das Fraktal in einer beliebigen Farbtiefe
zu speichern, z.B. in 24 Bit.
3. 32 Bit IEEE Single Precision-Puffer (für Speichermillionäre): Hier wird für jeden Punkt
ein ganzes Langwort reserviert, in dem der genaue Wert des jeweiligen Punktes abgelegt wird.
Hiermit ist es möglich, auch das Innengebiet der Julia-/Mandelbrotmenge als wahre
24Bit-Bilder zu speichern.

Windows
2 Wahlmöglichkeiten:
- 1 Window: Falls eine 3D-Ansicht des Fraktals gezeichnet werden soll, wird diese im selben Window wie das
2D-Fraktal gezeichnet. Das 2D-Fraktal wird einfach übermalt.
- 2 Windows: Falls eine 3D-Ansicht des Fraktals gezeichnet werden soll, wird ein
zweites Window hierfür geöffnet.

3D
Und wieder 2 Möglichkeiten:
- Kein 3D-Bild: Nur das Fraktal selbst wird gezeichnet. Keine 3D-Ansicht davon wird
berechnet.
- 3D-Bild: Nach dem Zeichnen des 2D-Fraktals werden die Daten als Höhen interpretiert und eine
3D Ansicht gezeichnet.

Palettenwahl
Wird ein neues Window aktiviert, versucht das Programm herauszufinden, welche Palette
nun eingestellt werden soll. Dazu untersucht es das Window daraufhin, zu welchem Fraktal
es gehört und stellt die entsprechende Farbpalette ein.
Das Programm hat stets eine globale Farbpalette. Zusätzlich existieren in jeder Fraktalstruktur
noch 2 Einträge für Palettennamen. Der eine Palettenname ist für die Palette gedacht, die für das 2D-Fraktal verwendet werden
soll, der andere Name für das 3D-Fraktal.
Zur Kontrolle über das mögliche Verhalten des Programmes bei der Palettenwahl existieren 2

```
Gadgets. Falls das Checkboxgadget angewählt ist, wird stets die globale ↩
    Farbpalette
angezeigt, völlig unabhängig vom Fraktal und den beiden Palettennamen in der ↩
    Fraktalstruktur.
Dieser Modus ist vor allem für Leute wie mich gedacht, die verwirrt werden, wenn ↩
    bei einer plötzlichen Windowaktivierung
die Farbpalette geändert wird. (Ich benutze 'SunWindow' - kreiert von Bernhard ↩
    Scholz - Werbung! - fürs
automatische Aktivieren meiner Window...)

Falls das Checkboxgadget nicht angewählt ist, bestimmt das Cyclegadget daneben, ↩
    welche
Palette für das Fraktal eingestellt werden soll. 'Eigene Palette' stellt die ↩
    Palette
ein, die durch die Angaben in der Fraktalstruktur definiert ist. 'Globale Palette' ↩
    stellt
die globale Palette ein, wenn immer ein Window aktiviert wird, das zu diesem ↩
    Fraktal
gehört.
Änderungen der Palette wirken sich immer auf die eingestellte Palette aus, das ↩
    heißt,
wenn gerade die globale Palette eingestellt ist und im Palettenwindow eine andere ↩
    Palette
angewählt wird, so wird damit die globale Palette umdefiniert. Soll heißen, es ↩
    passiert eben genau das,
was man erwartet.
```

1.9 Farbpaletten

```
2.2.2 Farbpalettenwindow

Das Farbpalettenwindow enthält sämtliche Paletten, die das Programm beim Start
im Verzeichnis 'Palette' gefunden hat. Dort können sich auch ganze Bilder befinden ↩
    .
In diesem Fall wird aus dem Bild der Farbchunk mit den Farbinformationen ↩
    herausgefilert,
also die Palette des Bildes in die Liste aufgenommen.

Zu Beginn des Programms ist die Palette mit dem Namen 'Default' eingestellt.
Bevorzugt irgendjemand also eine andere Farbpalette, so muß er sie nur unter
dem Namen 'Default' im Palette-Verzeichnis ablegen. Existiert keine Palette mit
diesem Namen, so wird die erste gefundene Palette eingestellt.

Will man nun eine andere Palette einstellen, so muß nur auf den entsprechenden
Eintrag geklickt werden. Die Änderung erfolgt sofort. Damit die Bilddarstellung
nicht leidet, können durch Paletten die Farben 0 bis 3 nicht beeinflußt werden.
Sie werden benötigt, um den 3D-Effekt der graphischen Elemente aufrechtzuhalten
und die Schrift lesbar zu halten. Andere Fraktalprogramme gingen hier ja mit ↩
    schlechtem
Beispiel voraus...

Palettenname
-  Um den Namen einer Palette zu ändern, ist nur ein neuer Name in das ↩
    Stringgadget
einzutragen und Return zu drücken.
```

Palette bearbeiten
- Will man die Farben einer Farbpalette ändern, so klickt man einfach auf den Knopf mit der Aufschrift 'Bearbeiten'. Es öffnen sich 2 weitere Windows, eines in dem die Werte der aktuellen Farbe dargestellt werden und eines, in der ein Palettengadget die Farben anzeigt. Abhängig davon, ob auf dem aktuellen Screen 256 Farben zur Verfügung stehen, wird evtl. ein neuer Screen anhand der im Preferences-Programm zu ChaosPro eingestellten Daten geöffnet.
Siehe Kapitel 2.1.3 Palettenbearbeitungswindow

Palette laden und Palette speichern
- Mit diesem Knöpfen können Paletten eingeladen und abgespeichert werden. Wird beim Abspeichern einer Palette festgestellt, daß eine Datei mit dem gleichen Namen überschrieben würde, so erfolgt zur Sicherheit eine Abfrage. Beim Einladen wird natürlich Multiselect unterstützt.

Palette löschen und Palette duplizieren
- Die Namen sagen alles...

Farboffset
- Mit diesem Gadget stellt man ein, ab welcher Farbe die Farbpalette benutzt werden soll. Beispiel: Jemand hat einen Screen mit 32 Farben, er stellt hier einen Wert von 30 ein. Der Screen erhält nun die Farben der Farbpalette ab Farbe 30 (wird zu Screenfarbe Nr. 4) bis zu Farbe 57 (Screenfarbe 31). Ändert man den Wert kontinuierlich, so erhält man natürlich eine Art Colorcycling-Effekt.

Überspringen
- Ist hier z.B. der Wert 2 eingestellt, so wird bloß jede 2. Farbe der Farbpalette hergenommen. Dies ist sinnvoll z.B. für Farbpaletten, die eigentlich für 256-Farb-Screens ausgelegt sind und die nun für z.B. 32-Farb-Screens hergenommen werden. Hier stellt man diesen Wert einfach auf 8, nimmt somit nur jede 8. Farbe der Palette her und schon hat man eine Vorstellung davon, wie es wohl auf einem 256-Farb-Screen aussieht.

1.10 Palettenbearbeitung

2.2.3 Palettenbearbeitungswindow

Eigentlich sind es ja 3 Windows. Das eine ist für die Steuerung vorhanden, das andere wird zur Anzeige der Farben verwendet. Das 3. evtl. für das Colorwheel und den Gradientslider.

Farbbereich
- Nicht jeder wird die Möglichkeit haben, sich alle 252 Farben - die ersten 4 Farben werden ja vom Programm für sich reserviert - gleichzeitig anzeigen zu lassen.

```
Aus diesem Grund existiert dieses Gadget. Es zeigt mit der Größe und der Position
seines Rollbalkens, wie viele Farben aus welchem Bereich der Palette im anderen
Window angezeigt werden. Wird er verschoben, so verschieben sich gleichzeitig auch die angezeigten Farben im anderen Window
entsprechend der neuen Position des Rollbalkens.

Farbnummer
-  Mit diesem Gadget läßt sich das Farbregister einstellen, gleichzeitig werden
auch alle anderen Werte aktualisiert, wie z.B. der Rollbalken des Farbbereich-\ Gadgets
bzw. die RGB- und FSH- Werte.

Die RGB- und FSH- Regler
-  Mit diesen Reglern kann man die Farbwerte ändern. Gleichzeitig mit einer Änderung
der RGB- Werte werden auch die FSH- Werte aktualisiert und umgekehrt. Wer also
bisher noch nicht so recht den Zusammenhang zwischen RGB und FSH erkannte, der kann
diese Wissenslücke nun schließen.

Kopieren, Austauschen, Farbnuancen
-  'Kopieren' kopiert die aktuelle Farbe an die Stelle, die der Benutzer im Palettengadget
als nächstes anklickt, 'Austauschen' tauscht sie aus und 'Farbnuancen' schafft einen fließenden Übergang
zwischen den beiden Farbregistern und ihren Farben.

Cycling-Modus
-  Etwas unkonventionell ist dieser Modus: klickt man das Gadget an, so befindet man sich im Modus,
in dem man genau festlegen kann, welche Farben beim Colorcycling mitmachen sollen und
welche nicht. Sinnvoll z.B. für die Mandelbrotmenge, wenn man nur das Außengebiet cyclen
lassen will und nicht das (evtl. einfarbige) Innengebiet.
Sämtliche Farben, die sichtbar sind, nehmen am Colorcycling teil, sämtliche Farben ,
die in Grautönen blinken, nehmen nicht teil. Anklicken einer Farbe im Palettengadget
ändert den Zustand desselbigen.
Die 3 Gadgets 'Alle', 'Keines' und 'Invertieren' machen genau das, was man erwartet:
'Alle' läßt alle Farben am Colorcycling teilnehmen, 'Keines' keine Farbe und ' Invertieren'
invertiert den Zustand jedes Farbregisters.

Bereichsfunktionen
- 'Stauchen zu' und 'Stauchen'
Mit diesen beiden Gadgets ist es möglich, die Palette auf weniger Farben zusammenzustauchen.
Dazu wählt man mit dem Schieberegler die Anzahl an Planes, und mit 'Stauchen' löst man
die Aktion aus.
- 'Bereich invertieren'
Damit lassen sich die Farben in einem Bereich umdrehen. Dazu klickt man auf das Gadget, dann auf die
erste Farbe des Bereichs, dann auf die letzte Farbe des Bereichs.
```

```
- 'Bereich kopieren'
Dazu klickt man auf das Gadget, dann auf die erste Farbe des Bereichs, dann auf ↩
    die
letzte Farbe des Bereichs, und nun noch auf die erste Farbe des Zielbereichs. ↩
    Überlappende, überlaufende,
etc. Bereiche werden korrekt behandelt...

Farben einer Farbpalette
- Dieses Window zeigt die Farben der Farbpalette an. Es besitzt ein Sizegadget und
kann daher in der Größe verändert werden.

Farbrad
Leute, die OS3.0 benutzen, können hier ein Farbrad haben, in dem auf recht ↩
    intuitive Art und Weise
die Farben gewählt werden können. Um das Farbrad zu erhalten, muß das ToolType ↩
    COLORWHEEL
angegeben werden. Da es die Hälfte der zur Verfügung stehenden Farben für seine ↩
    Darstellung benötigt,
kann man es auf diese Art und Weise an- und ausschalten.
```

1.11 Animationswindows

```
2.2.4 Animationswindows

Mit diesen Windows werden Fraktalanimationen berechnet. Dazu werden Fraktale,
die sich nur in kontinuierlich veränderlichen Parameterwerten unterscheiden dürfen ↩
    ,
als Keys definiert. Bei der Berechnung werden dann die Zwischenwerte zwischen den
sich unterscheidenden Parametern berechnet (deshalb dieses 'kontinuierlich ↩
    veränderlich')
und einem Fraktal als Parameter übergeben, das dann berechnet wird. Dieses System ↩
    ist
recht leistungsfähig, so kann z.B. der Parameterwert c der Standardjuliamenge ↩
    kontinuierlich
verändert werden, gleichzeitig können sich die Ausschnittwerte des Fraktals und ↩
    die
Iterationstiefe ändern.
Der Effekt: z.B. ein Hineinzoomen in eine sich verändernde Juliamenge, was einem ↩
    effektvollen
Morphing gleichkommt...

Doch nun zur Beschreibung der Gadgets:

Fraktalbilder
- Hier werden  nochmal  die  Fraktale  aus  dem PicTask-Window
angezeigt.

AnimKeys
- Das sind die Schlüsselpositionen. Eine Animation wird als kontinuierlicher
Übergang von einem Key zum nächsten berechnet, bis hin
zum letzten, berechnet.

Aktionen
- neuer Key / als erster
Damit wird das im Fraktalbilder-Gadget des Animationswindows angewählte Fraktal
```

als neuer Key definiert. Der Key wird im einen Fall hinter dem angewählten Key
in die Liste eingefügt, im anderen Fall an die erste Stelle. Beim Einfügen wird
überprüft, ob dieser Key mit den in der Liste befindlichen Keys verträglich ist,
d.h. ob es sich auch um denselben Fraktaltyp, um denselben Subtyp handelt und
ob er sich von den anderen nur in kontinuierlich veränderbaren Parametern unterscheidet.
Andernfalls wird eine Fehlermeldung gebracht, evtl. mit dem Hinweis, welcher Parameter
nicht zu den anderen paßt, verbunden mit dem Angebot, den Parameterwert des neuen Fraktals
entsprechend anzugleichen. Zu bemerken ist noch, daß der neue Key den Namen des Ursprungsfraktals trägt und beim
Einfügen kopiert und NICHT referenziert wird.
Das heißt insbesondere, daß die Änderung eines Parameterwertes des Fraktals sich
nicht auf den gleichnamigen Key auswirkt. Das schafft eine sehr schnelle Möglichkeit,
Animationen zu generieren. Dazu wird das Fraktal berechnet, als Key eingefügt, dann
zoomt man in das Fraktal hinein (oder ändert irgendeinen anderen Wert des Fraktals) und fügt es erneut in die Key-Liste nach dem einen ein.
Fertig ist ein Zoom-In-Movie...
Der erste Key wurde ja kopiert, somit stehen seine Daten unwiderruflich fest und lassen
sich nicht mehr ändern.

- lösche Key
Ich spare mir eine Erklärung...

- Key aufwärts / Key abwärts
Damit läßt sich die Reihenfolge der Keys ändern...

- Key nach Bild
Ein Nachteil des Kopiertwerden der Keys ist, daß auf den Key nicht mehr zugegriffen und nachträglich
Parameter geändert werden können. Es geht sogar so weit, daß überhaupt keine Parameter mehr
betrachtet werden können. Ein Key ist kein Bild, also kann er nicht als Bild berechnet werden, folglich können keine Parameterwindows
geöffnet werden. Wenn man also vom Freund eine AnimData-Datei bekommt, so weiß man nicht einmal, welcher Fraktaltyp hierbei verwendet wird.
Mit diesem Gadget nun ist es möglich, einen Key in ein Bild zurückzuverwandeln, sich die Parameter anzusehen, zu verändern, und dann den alten Key
zu löschen und dieses neue Bild als Key an die alte Position einzufügen. Perfekt ist diese Lösung nicht, aber sie war einfach
... sorry.

- Start / Abbruch
Bei Betätigen des Gadgets 'Start' wird die Animationsberechnung gestartet. Dazu wird
ein neues Fraktalwindow geöffnet, in dem das Fraktal berechnet wird. Natürlich kann dabei
weitergearbeitet werden. Lediglich eine 2. Animation läßt sich nicht starten. Um dies
dem Benutzer klarzumachen, werden alle Gadgets in diesem Window unwählbar, nur das Abbruchgadget
bleibt anwählbar. Was es tut, ist wohl jedem klar...

- Laden/Speichern
Mit diesen Gadgets wird die Liste der Keys abgespeichert (kann dann z.B. bei einem Freund mit einem schnelleren Amiga geladen und
dort berechnet werden) oder geladen.

Zeiteinstellungen
Das Animationssystem ist nun zeitorientierter als vorher. Es existiert eine einstellbare
Zeiteinheit, die als Verweildauer eines einzelnen Frames angesehen werden kann. Man sieht nun
gleich, wie lange eine Animation dauern wird, wann ein bestimmter Frame angezeigt werden wird,
etc.

Moment
Diese beiden Gadgets, die nur lesbar sind, zeigen, zu welchem Zeitpunkt der gerade
aktuelle Keyframe gezeigt wird. Die Angabe erfolgt sowohl als Zeit in Sekunden vom Start an als auch
als Frameanzahl vom Start der Animation an.

relativ zum letzten
Das obere Gadget zeigt die Zeitdifferenz zum letzten Key, d.h. wieviel Zeit zwischen der Anzeige des
letzten Keys und des aktuellen Keys liegt. Das untere der beiden Gadgets ist einstellbar und bezeichnet
die Anzahl der zu berechnenden Frames zwischen dem letzten Key und dem aktuellen.

total
Die beiden Gadgets, beide nur lesbar, zeigen, wie lange die Animation dauern wird, und aus wie vielen Frames sie
derzeit besteht. Die Zeitdauer ist natürlich die Anzahl an Frames multipliziert mit der Zeiteinheit.

Zeiteinheit
Hier kann die Zeiteinheit eingestellt werden. Normal sollte ein Wert von 0.05 sein, d.h. 0.05 Sekunden pro Frame,
also 20 Frames pro Sekunde. Ändert man diesen Wert, so ändern sich alle Zeiten, allerdings keine einzige Framezahl.
Um die Framezahlen, aber nicht die Zeiten zu ändern, muß auf ’Zeit normalisieren’ geklickt werden.

Zeit normalisieren
Dieses Gadget setzt die Zeiteinheit auf 0.05, läßt die Zeitpunkte jedes einzelnen
Keys gleich und berechnet die Frameanzahlen neu.
Angenommen, die Zeiteinheit ist 0.05 und man will die Frameanzahlen erhöhen, die
Animation also quasi ’smoothen’, so stellt man als erstes die Zeiteinheit auf einen
Wert größer 0.05, z.B. 0.1, was die Animationsdauer erhöht (um den Faktor 2), aber
die Framezahlen gleich läßt. Anschließend betätigt man ’Zeit normalisieren’, was die
Zeitdauer wieder auf 0.05 setzt, die Zeitpunkte jedes einzelnen Frames aber gleich läßt, also
auch die Gesamtdauer der Animation. Das erhöht die Frameanzahl jeweils um den Faktor 2.
Ergebnis: Die Animation ist dieselbe, aber sie besteht aus doppelt so vielen Frames.
Entsprechendes gilt, falls man weniger Frames haben will.

Zeit berechnen
Dieses Gadget versucht, eine ideale Anzahl an zu berechnenden Frames zwischen dem letzten Key und dem aktuellen zu berechnen. Dazu ermittelt es alle Parameter, in denen sich die beiden Keys unterscheiden und berechnet anhand der Größenordnungen der jeweiligen Unterschiede eine Zahl.

Alle berechnen
Macht dasselbe wie 'Zeit berechnen', aber für alle Frames.

Sonstiges

Puffer
Hier wählt man den Puffer, der für die Berechnung gewünscht wird. Für 3D-Animationen oder 24 Bit muß ein Puffer vorhanden sein. Für 3D-Animationen empfiehlt es sich außerdem, einen IEEESP-Puffer zu benutzen, sonst kann es zu unschönen Effekten kommen.

Interpolation
Hier kann man wählen, ob die einzelnen Keys linear interpoliert werden sollen oder aber ein Spline (zur Zeit nur ein kubischer Spline) durchgelegt werden soll. Im Falle einer linearen Interpolation kommt es zu gewissen Ruckeffekten, besonders beim Hineinzoomen, was aber meines Erachtens gar nicht so schlecht aussieht. Dies wird bei der Splineinterpolation naturgemäß vermieden.

Speichermodus
Hier kann man wählen, ob die Animation im AnimOpt5-Format abgespeichert werden soll, so daß sie ohne Probleme mit allen gängigen Animationsplayern abgespielt werden kann, oder ob jedes einzelne Bild als Bild gespeichert werden soll. Im letzteren Fall kann man nach dem Start der Animation im Filerequester den Basisnamen angeben, die Einzelbilder bekommen dann noch Nummern angehängt.
Diese Wahl ist nötig, da das AnimOpt5-Format keine 24Bit-Animationen zuläßt. Im Falle von Speichermodus=Bilder kann man bei der Planetiefe auch auf 24 gehen.

Breite / Höhe
- Hier wird die Größe der Animation eingestellt. Keine Angst, Falschmachen kann keiner was. Die Werte werden auf Korrektheit überprüft.

Planes
- Bei manchen Fraktaltypen ist es möglich, eine Animation mit bis zu 256 Farben zu berechnen, obwohl das Programm nur auf einem 16-Farb-Screen läuft bzw. die Hardware dazu überhaupt nicht in der Lage ist. Mit diesem Gadget wird eingestellt, in wievielen Planes Sie die Animation berechnen lassen wollen. Evtl. kann sie dannach ja mit z. B. Clarissa nachbearbeitet, konvertiert, etc. werden.
Der Wertebereich dieses Gadgets reicht normalerweise von 3 bis 8, falls der Speichermodus 'Bilder'

ist, enthält er alle Werte von 3 bis 8 und den Wert 24.

Startframe&Endframe
Hiermit kann man den Start- und den Endframe angeben. Falls ihr Computer während einer Berechnung plötzlich
versagen sollte, können sie nun auch die Restanimation berechnen lassen. Falls für EndFrame 0 eingetragen wird, heißt das,
daß die Animation bis zum maximalen Frame berechnet wird.

3D-Animation
- Manch ein Fraktaltyp kann in 3D dargestellt werden. Mit Hilfe dieses Gadgets wird bestimmt, ob eine 3D-Animation gewünscht wird. Hierzu werden bei
der Berechnung automatisch beide Windows, das 2D- und das 3D-Window geöffnet und der Inhalt des 3D-Windows
jeweils als Animationsframe abgespeichert.

1.12 CycleControl-Window

2.2.5 CycleControl-Window

Dieses Window ist dazu gedacht, die Kontrolle über das Colorcycling leichter zu machen.
Dazu existieren 3 Gadgets:

Colorcycling
- Schaltet Colorcycling an und aus

Speed
- Stellt die Geschwindigkeit ein, Wertebereich von 20 bis 999.

Richtung
- Soll aufwärts oder abwärts gecycled werden?

1.13 Benutzerdefinierte Windows

2.2.6 Benutzerdefinierte Windows

Es können eine beliebige Anzahl an Windows vom Benutzer definiert werden. Alle Windows
bestehen aus einer vertikalen Knopfleiste. Wenn ein Knopf betätigt wird, so wird
das dem Knopf zugeordnete Arexx-Script ausgeführt. Auf diese Art können einzelne nicht
implementierte Funktionen dem Programm hinzugefügt werden.

Der Aufbau der Windows steht in der ASCII-Datei 'Windows.asc', die sich in ChaosPro/Prefs
befinden muß.
Aufgebaut ist sie wie folgt:

```
WINDOW <Windowtitel> <Arexx-Scripts>
GADGET <Gadgetname> <Arexx-Script>
...
```

```
GADGET <Gadgetname> <Arexx-Script>
WINDOW <Windowtitel> <Arexx-Scripts>
GADGET <Gadgetname> <Arexx-Script>
...
GADGET <Gadgetname> <Arexx-Script>
WINDOW <Windowtitel> <Arexx-Scripts>
...
END
```

Das Arexx-Script, das in der Zeile mit dem WINDOW-Schlüsselwort steht, wird jedesmal
beim Öffnen dieses Windows ausgeführt. Zu beachten ist, daß ein Fehler in dieser
Datei bis zum Absturz führen kann, also Achtung.
Ist die Datei geschrieben, so muß sie vom Preferencesprogramm aus übersetzt werden.
Ich hoffe, daß dieses Programm möglichst alle Fehler abfängt, so daß es zu keinem
Absturz bei Fehleingaben kommt...
Die übersetzte Datei wird automatisch in ChaosPro/Prefs/ unter dem Namen Windows.prefs
abgespeichert.

1.14 2D/3D-Fraktalwindows

2.3 Fraktale

2.3.1 Die 2D/3D-Fraktalwindows

Im 2D-Fraktalwindow wird stets das den Parametern entsprechende 2D-Fraktal - bei den
beiden dynamischen Systemen ist es bereits eine Projektion eines dreidimensionalen Fraktals -
angezeigt.
Ändert sich ein Parameter, so wird sofort das Fraktal neu berechnet.

Folgende Aktionen sind im Window möglich (bei Typ Plasma nicht):

1. Cursor-Tasten bzw. Joystick in Port 2
Damit verschiebt man das Fraktal um 8 Bildschirmpixel in die entsprechende Richtung.

2. Leertaste bzw. Feuerknopf Joystick in Port 2
Hineinzoomen in das Fraktal, je nach vorhandenem Speicher werden Zwischenzooms
dadurch "berechnet", daß das ursprüngliche Bild durch Skalierung vergrößert wird.

3. Fraktal anklicken und Maus bewegen
"Packt" das Fraktal und verschiebt es. Hat man einen schnellen (einen sehr schnellen)
Rechner, so kann man auf diese Weise über das Fraktal in Echtzeit hinwegfliegen.

4. Doppelklick auf eine Stelle
Zoomt um den Faktor 2 in das Fraktal hinein und plaziert dabei die Stelle, auf
die gedoppelklickt (wurks) wurde, mittig im Window.

Im 3D-Fraktalwindow wird stets das den Parametern entsprechende 3D-Fraktal ↩
angezeigt.
Eine 3-dimensionale Ansicht ist nur bei Julia-/Mandelbrotmengen möglich. ↩
Prinzipbedingt ist es bei den dynamischen Systemen (das sind ja bereits
3D-Fraktale) und bei Bifurkationsdiagrammen nicht möglich, ein 3D-Fraktalwindow zu ↩
öffnen. (Was sollte man
auch bei den Bifurkationsdiagrammen zeichnen?)

Alle anderen nicht unterstützten Tastendrücke werden vom KeyboardControl-Modul
zuerst an das Datenwindow zum Fraktal weitergeleitet, werden sie auch von ihm ↩
nicht
verstanden, dann an das Parameterwindow1 , dann 2, dann 3. Somit ist es möglich,
im Fraktalmenü den Tastaturshortcut für das Erhöhen der Iterationstiefe zu drücken
und die Iterationstiefe wird tatsächlich erhöht, als ob die Taste im ↩
entsprechenden
Parameterwindow gedrückt wurde. Ich hielt das für sinnvoll, da bei mir immer das ↩
falsche Window aktiv ist.
(Murphys Gesetz, die Nummer hab' ich vergessen)

1.15 Juliamengen: Theorie

2.3.2 Julia- und Mandelbrotmengen

2.3.2.1 Theorie: Juliamengen

weiterhin: 2.3.2.2 Theorie: Mandelbrotmengen

Ich beziehe mich im folgenden auf die 'Standardformel' f(z)=z^2+c.
Bei der Erzeugung von Juliamengen ist die komplexe Zahl c beliebig zu Beginn ↩
wählbar,
aber konstant während der Berechnung. Jeder Punkt im Window entspricht einer ↩
komplexen
Zahl. Welcher, ergibt sich aus den Ausschnittwerten im Parameterwindow Nr. 1. Die
Frage ist nun, was passiert, wenn man z mit dem dem Bildschirmpunkt entsprechenden ↩
komplexen Zahlenwert initialisiert
und dann die Formel iterativ anwendet. Also:
z=dem Bildschirmpixel entsprechende komplexe Zahl
z1=f(z)=z^2+c
z2=f(f(z))=f(z1)=z1^2+c
z3=f(f(f(z)))=f(z2)=z2^2+c
...
Die Juliamenge besteht nun aus allen Punkten z, für die die Punktbahnen ↩
nirgendwohin
führen, oder, anderes gesagt (kontrapositiv), alle Punkte, die NICHT zur ↩
Juliamenge
gehören, haben eine Bahn, die irgendwohin führt, zu einem sogenannten anziehenden ↩
Punkt,
einem Attraktor, bzw. allgemeiner gesagt, zu einer anziehenden Menge, die aus ↩
mehreren
Punkten bestehen kann. Das heißt nun, daß die Juliamenge nicht dieses tolle bunte
Gebilde ist, sondern die - nicht unbedingt sichtbare - schwarze Fläche. Das, was
so bunt eingefärbt ist, sind die Punkte, deren Bahnen zu einem anziehenden Punkt
hinführen und die eben nicht zur Juliamenge gehören.

```
kritischer Punkt: f'(z)=0, also 2z=0, also z=0
Fixpunkte: z=f(z), 0.5\ensuremath{\pm}sqrt(0.25-c) und z=unendlich

- Fixpunkte und Eigenwerte

Wie der aufmerksame Leser bereits bemerkt haben wird, wird die Juliamenge weitgehend
von den anziehenden Punkten bestimmt, Punkten also, für die z=f(z) gilt, den anziehenden
Fixpunkten. Es ist durchaus möglich, daß ein Punkt z0 ein Fixpunkt ist, aber nicht anziehend,
d.h. ist z sehr nahe bei z0, so 'flüchtet' er unter Umständen geradezu von z0 weg. Die
Frage ist nun, wann ist ein Fixpunkt z0 anziehend und wann abstoßend. Dazu braucht
man nur die Ableitung der Formel f(z) zu berechnen, in diesem Fall also f'(z)=2*z.
Nimmt man z aus einer sehr kleinen Umgebung ( eigentlich einer unendlich kleinen )
von z0 und berechnet f(z), so erkennt man, daß ungefähr gilt: |f(z)-z0|=|f'(z0)*(z-z0)|
(vgl. Tangente).
Man erkennt also, daß Anziehung oder Abstoßung von f'(z0), dem sogenannten Eigenwert
des Fixpunktes z0, abhängt. Ist f'(z0)<1, so ist der Abstand von f(z) und z0 kleiner als
der Abstand von z und z0, also ist z0 anziehend. Ist f'(z)>1, so vergrößert sich der Abstand, der
Fixpunkt z0 ist abstoßend Ist f'(z0)=1, so ist der Fixpunkt neutral. Hier können dann viele interessante
Erscheinungen auftreten.
In der Standardformel f(z)=z^2+c sind die Fixpunkte durch Lösen dr Gleichung f(z)=z herausfindbar.
Die Lösungen dieser Gleichung sind:
z1=0.5+sqrt(0.25-c)
z2=0.5-sqrt(0.25-c)

Will man also eine interessante Juliamenge berechnen lassen, so muß man c so wählen, daß die Beträge der Eigenwerte
der Fixpunkte z1 und z2 kleiner als 1 sind.

Außerdem muß man aus theoretischen Gründen den Punkt 'unendlich' auch als Fixpunkt
ansehen, obwohl klar ist, daß dieser 'Punkt' in der Praxis alles andere als anziehend ist.
Dieser Punkt ist in jedem Fall als anziehend einzustufen. Das Berechnen des Eigenwertes ist
hier ziemlich sinnlos. Da der Punkt 'unendlich' immer als anziehend einzuordnen ist, haben
es sich so ziemlich alle Fraktalgeneratoren bisher sehr einfach gemacht, indem sie
nur den unendlich fernen Punkt als einzigen Fixpunkt annahmen. Na ja, richtig ist es nicht, aber
die charakteristischen Bilder der Juliamengen in den verschiedenen Publikationen
lassen sich so berechnen.

Die ganze Sache von den Fixpunkten kann noch ein bischen komplizierter werden.
Zu Beginn habe ich schon erwähnt, daß es nicht nur anziehende Punkte, sondern sogar
anziehende Mengen geben kann, sogenannte Zyklen mit einer Zykluslänge. Dies ist dann z.B. eine
```

Menge aus 3 Punkten (Zykluslänge 3) z1,z2,z3, für die gilt:
f(z1)=z2, f(z2)=z3, f(z3)=z1
Wendet man also f 3 mal auf z1 oder z2 oder z3 an, so kommt wieder z1, z2 oder
z3 heraus. Mit meinem Programm kann man sich auch auf Zyklensuche begeben. Die Theorie verstehe ich allerdings
selbst nicht ganz, also lass ich es lieber...

Weiteres zum Thema Juliamengen findet sich in den folgenden Kapiteln zur Beschreibung der
einzelnen Parameter. Dort wird (hoffentlich) klar, wie mein Programm bei der Berechnung
vorgeht, was die einzelnen Parameter bewirken.

Wenn Sie dieses Kapitel nicht verstanden haben, so kann es durchaus vorkommen,
daß Sie bei der Änderung von Parametern plötzlich (oder sogar recht oft) vor einem
einfarbigen Fraktal oder irgendeinem anderen langweiligen Bild sitzen und nicht wissen,
wie Sie dieses Aussehen interpretieren sollen...

So, und zum Schluß noch etwas vor allem für Mathematiker Interessantes:
Das Newton'sche Verfahren zur Nullstellenbestimmung

Newton (wer auch sonst ?) hat sich stark mit der näherungsweisen Bestimmung der Wurzeln
(Nullstellen) von Polynomen P(z) beschäftigt. Die Formel, die er herausfand und deren iterative
Anwendung eine Näherung für eine Nullstelle ergibt, lautet allgemein:
f(z)=z-P(z)/P'(z)
z wird zu Beginn initialisiert, anschließend die Funktion iterativ jeweils auf das Ergebnis
angewendet, bis sich der Funktionswert mehr oder weniger bei einem bestimmten Wert einpendelt.
Das Problem, mit dem Newton nicht ganz fertig wurde und mit dem man heute noch nicht einmal fertig
wird, ist, für welche Startwerte von z sich eine, und wenn ja, welche Nullstelle ergibt bzw. wie
bekommt man alle Nullstellen des Polynoms P(z), bzw. bekommt man so überhaupt irgendwie
alle?
Hier kommt man nicht weiter. Um sich ein Bild von der Sache zu machen, benutzt man den
Computer. Es handelt sich bei näherer Betrachtung nämlich um simple Juliamengen mit einer
benutzerdefinierten Formel!
Sei nun P(z)=z^3+2, also P'(z)=3*z^2
Als f(z) ist also im Programm z-(z^3+2)/(3*z^2) einzutragen und die Juliamenge hiervon zu berechnen.
Im Parameterwindow 2 muß noch auf endliche Attraktoren (eben die Nullstellen von P(z)) geprüft werden, dieser
Punkt ist also dort anzuwählen. Das sich ergebende Bild ist das sogenannte 'Basin of Attraction' der Funktion.
Öffnet man nun noch das Datenwindow hierzu, so findet man unter dem Punkt 'Ende:' den Endpunkt der Berechnung, also den
endlichen Attraktor, eine Näherung für eine Nullstelle von P(z). Fährt man über das Fraktal mit dem Mauszeiger hinweg, so erkennt man sehr schön, welcher

```
Wert von z zu welcher Nullstelle führt. Bei einem Polynom dritten Grades sind ↩
    stets nur 2 Fälle möglich: Entweder sind alle 3 Nullstellen
reell, oder aber eine reelle Nullstelle und 2 komplexe, die zueinander konjugiert ↩
    komplex sind, existieren. Letzeres
ist hier der Fall, wie leicht nachgeprüft werden kann.
Aus der Komplexität des Fraktals erkennt man, wieso sich die heutige Mathematik ↩
    bei eigentlich so einfach
auszudrückenden Problemen so schwer tut.
Für die Tranformation in die 3. Dimension eignet sich all dies nicht, da hier die ↩
    Continuous Potential
Method in meiner Implementation versagt, was zu Treppenstufen führt.
```

1.16 Mandelbrotmengen: Theorie

```
2.3.2.2 Theorie: Mandelbrotmengen

weiterhin: 2.3.2.1 Theorie: Juliamengen

Grundsätzlich gibt es 2 verschiedene Haupttypen von Juliamengen:
Typ A) Die Juliamenge ist staubartig, d.h. sie besteht aus einer unendlich großen
Anzahl unzu\sammen\hängen\der Punkte
Typ B) Die Juliamenge ist zusammenhängend, d.h. sie besteht aus einer Vielzahl
von Linien, einer Fläche, einer Schleife, oder etwas ähnlichem.
Der Typ einer Juliamenge hängt von ihrem Parameter ab, bei der Standardformel
f(z)=z^2+c also von c.
Die Mandelbrotmenge stellt nun graphisch dar, für welche Werte von c die ↩
    zugehörige
Juliamenge vom Typ A oder vom Typ B ist.
Julia, der Erfinder der Juliamengen, hat sich einen Trick ausgedacht, mit dem
es möglich ist zu entscheiden, zu welchem Typ eine Juliamenge gehört, ohne sie
selbst zu konstruieren. Dazu sind nämlich lediglich die Bahnen der kritischen
Punkte zu verfolgen. Die kritischen Punkte einer Formel sind die Punkte, für
die die Ableitung verschwindet, bei der Standardformel ist also nur der Punkt
0 kritischer Punkt.
Zur Konstruktion der Mandelbrotmenge müssen sämtliche kritischen Punkte beachtet ↩
    werden.
Die vorliegende Programmversion ist dazu nicht in der Lage. Sie kann nur einen
kritischen Punkt verfolgen. Folglich ist das Ergebnis bei Vorhandensein von ↩
    mehreren
kritischen Punkten nicht korrekt. Man kann sich damit behelfen, daß man ↩
    nacheinander
die Bahnen der kritischen Punkte verfolgen läßt, die Bilder jeweils abspeichert
und dann in einem Malprogramm übereinanderlegt. Das Ergebnis ist die korrekte
Mandelbrotmenge.

Eine Mandelbrotmenge wird nun also folgendermaßen konstruiert:
Abhängig vom Ausschnitt aus der komplexen Zahlenebene initialisiert man c mit dem
dem Bildschirmpixel entsprechenden komplexen Zahlenwert, z initialisiert man mit
dem kritischen Punkt, also mit 0.
Anschließend iteriert man das Ganze, d.h. man berechnet f(z), dann f(f(z)), etc.
Führt die Bahn nach unendlich, so ist die zum Punkt c gehörige Juliamenge ↩
    staubartig.
Führt die Bahn zu einem endlichen Attraktor - einem Punkt oder einem Zyklus - so
ist die zugehörige Juliamenge zusammenhängend.
```

Somit ist die Mandelbrotmenge ein Atlas, eine Art grobe Landkarte, für sämtliche Juliamengen zu einer Formel. Sehr oft steht man vor dem Problem, daß sich einfach kein vernünftiger Parameterwert c für die Juliamenge finden läßt. In diesem Fall muß man nur in der zu derselben Formel gehörigen Mandelbrotmenge nachsehen, wo denn dieses c liegt, schon hat man die Erklärung für das langweilige Aussehen der Juliamenge. Eine Lösung ist nun, den Parameterwert in der Nähe des Randes der Mandelbrotmenge zu wählen. Dort ist nämlich zu erwarten, daß die Juliamenge noch nicht so recht weiss, ob sie nun staubartig oder zusammenhängend sein soll (Natürlich weiß sie es ganz genau, aber der Computer nicht...).
Im Inneren der Mandelbrotmenge ist die Juliamenge größtenteils eine unattraktive Fläche. Weit außerhalb der Mandelbrotmenge ist die Juliamenge staubartig, beinahe alle Punkte gehören nicht zur Juliamenge, sondern verlaufen recht geordnet zu einem Attraktor hin. Das heißt, innerhalb kurzer Zeit hat das mein Programm erkannt und färbt den Punkt ein. Resultat: Langweiliges Bild...

Zur einfachen Änderung des Parameters einer Juliamenge siehe den Menüpunkt Juliaparameter setzen

1.17 2.3 Fraktale --- 2.3.2 Julia- und Mandelbrotmengen

2.3.2.3 Parameterwindow 1

Parameter
- Je nach Formel sind 1 oder 2 komplexwertige Parameter wählbar. Diese sind hier einzutragen. Falls 2 Parameter wählbar sind, so ist der obere Parameter dem im Alphabet as erstem vorkommendem Parameter zugeordnet.
Der Parameter ist für Juliamengen entscheidend, da hierdurch die Lage der Fixpunkte und deren Eigenwerte bestimmt werden. Sie sollten deshalb entsprechend gewählt werden. Zum einfacheren Wählen der Parameter bietet es sich an, sich diesen innerhalb der Mandelbrotmenge anzeigen zu lassen. Genau dies kann man durch Anwahl des Menüpunktes Juliaparameter setzen erreichen.
Die Mandelbrotmenge für z.B. z^2+c ist ein 'Atlas' für alle Juliamengen z^2+c !
Interessante Juliamengen finden sich am Rand der Mandelbrotmenge.

Der erste Parameter bei Mandelbrotmengen ist nicht wählbar, da er ja immer mit dem dem jeweiligen Bildschirmpixel entsprechenden komplexen Zahlenwert initialisiert wird.

Iterationen
- Die Qualität einer Julia-/Mandelbrotmenge hängt stark vom Iterationswert ab. Je höher, desto besser, aber auch desto langsamer die Berechnung. Mit dem Slider kann der Iterationswert einfach ohne Tastaturbedienung verändert werden. Klickt man ihn an, so addiert sich sein Wert zum vorhandenen Iterationswert. Läßt man ihn wieder los, so schnappt er in die Ausgangslage, also in die 0-Position zurück. Alternativ läßt sich eine größere Änderung des Iterationswertes auch direkt in das Gadget eintragen. Es ist allerdings dannach die Return-Taste oder die Tab-Taste zu drücken.

Wie in den theoretischen Kapiteln dargestellt, müssen sämtliche Bahnen von Punkten,
die geordet verlaufen, also von einer Menge von Punkten angezogen werden, (Juliamenge),
bzw. sämtliche Bahnen, die von der Unendlichkeit angezogen werden (Mandelbrotmenge)
herausgefiltert werden. Diese 'geordnet' verlaufenden Bahnen werden der Optik wegen
eingefärbt und zwar entsprechend der Anzahl an Iterationen, die es gedauert hat,
bis meinem Program klar war, daß diese Punktbahn angezogen wird. All diese Punkte
gehören zum Aussengebiet der Julia-/Mandelbrotmenge.
Verläuft nach der eingestellten Anzahl an Iterationen eine Bahn eines Punktes immer
noch nicht erkennbar zu einer anziehenden Menge, so wird der Punkt als zur Ju\lia-/Mandel\brot\menge
gehörig eingestuft und gehört somit zum Innengebiet.

Passes
- Hiermit läßt sich die Anzahl der Zeichendurchgänge festlegen. Wegen der Beschaffenheit
der Julia- und Mandelbrotmengen kann man aus gleichen Iterationswerten am Rande
eines z.B. 4x4 Recktecks darauf schließen, daß sich innerhalb des Rechtecks auch
nur Punkte mit derselben Iterationstiefe befinden. Nun, dieser Schluß ist nicht
ganz richtig, er ist völlig falsch für eine staubartige Juliamenge, aber er hilft,
die Zeichengeschwindigkeit erheblich zu steigern.
Und bei staubartigen Juliamengen sieht man auch bei 1-Pass-Zeichnung nichts vom Staub,
da es äußerst unwahrscheinlich ist, daß von der begrenzten Anzahl der Punkte, die
hier gezeichnet werden, auch nur ein einziger exakt in die Juliamenge hineinfällt.
Sie fallen so gut wie immer ein ganz klein bischen daneben, und dieser Punkt gehört
dann ja schon nicht mehr zur Juliamenge. Hier wären andere Berechnungsmethoden
wie z.B. die Abstandsmethode mittels Continous Potential nötig, die mein Programm
jedoch nicht bietet.

Ausschnitt
- Die Juliamenge zeigt, was mit den Punkten der komplexen Zahlenebene passiert,
wenn man in iterativer Weise die Formel auf jeden einzelnen Punkt der Ebene anwendet.
Hier nun kann man den Ausschnitt aus der komplexen Zahlenebene wählen.

- Bei der Mandelbrotmenge wird hierdurch der Ausschnitt aus der komplexen Zahlenebene
definiert, der dann für den Parameterwert c der Formel benutzt wird.

Eliminieren
Ist dieses Gadget angewählt, so markiert das Programm nach jedem Zeichendurchgang alle Bereiche,
deren Eckpunkte jeweils dieselbe Iterationstiefe haben, d.h. alle inneren Punkte des Bereiches
müssen nicht mehr berechnet werden. Dies bringt einen Geschwindigkeitsgewinn von im Durchschnitt
etwa 50 Prozent. Der Nachteil ist natürlich, daß auf diese Art auch Bereiche gar nicht berechnet
werden, die eigentlich doch berechnet werden müßten. Oftmals sind solche Bereiche selten, doch sie
kommen auch vor.

Winkel
Um diesen Winkel wird das 2D-Fraktal noch gedreht. Insbesondere in Verbindung mit dem Animationssystem interessant. Es sind hier 2 Gadgets vorhanden, mit denen der Drehwinkel leicht eingestellt werden kann. Der Wertebereich reicht von -30000 bis 30000.

Theoretisches hierzu:
2.3.2.1 Theorie: Juliamengen
2.3.2.2 Theorie: Mandelbrotmengen

1.18 2.3 Fraktale --- 2.3.2 Julia- und Mandelbrotmengen

2.3.2.4 Parameterwindow 2

Aussenfärbung
- Hier gibt es 3 Möglichkeiten:
1. Farbe
Hierdurch wird das komplette Aussengebiet mit der direkt darunter eingestellten Farbe gezeichnet. Da das Aussengebiet in der Regel für das schöne Erscheinungsbild des Fraktals verantwortlich ist, ist der Sinn dieser Wahl auf den ersten Blick fraglich. Doch damit läßt sich das staubartige Aussehen z.B. der Juliamenge recht schön darstellen, falls es überhaupt möglich ist, denn nun verwirren die vielen Farben den Blick nicht mehr.

2. Iteration
Damit wird jedem Punkt, dessen Bahn geordnet verläuft, also von einem Attraktor angezogen wird, die Anzahl an Iterationen zugeordnet, die es gedauert hat, bis mein Programm die Ordnung der Bahn bemerkt hat. Diesem Zahlenwert, der von 0 bis zur eingestellten Maximalzahl an Iteratinonen-1 gehen kann, wird dann eine Farbe zugeordnet.

3. CPM - ein Akronym für Continous Potential Method
Wer den 2. Punkt betrachtet, dem wird auffallen, daß jedem Punkt ganz offensichtlich nur eine ganze Zahl zugeordnet werden kann. Das ist insbesondere bei Berechnung von 3D-Ansichten der Fraktale nachteilhaft, da es hier zu Treppen in der Darstellung kommt. Das Bild sieht aus wie chinesische Reisterassen. Die Höhenwerte der Punkte springen von einem Wert ganz plötzlich zum nächsten.
Das ändert sich bei Anwahl dieses Modus. Mittels einer recht einfachen Funktion können die Außengebiete von Julia- und Mandelbrotmengen in das Innengebiet eines Einheitskreises transformiert werden. Das besondere dabei ist, daß die Sprünge von einer Iterationsstufe zur nächsten (also die Grenzen der farbigen Bänder) , diese unheimlich komplizierten Kurven, in perfekte konzentrische Kreise mit Mittelpunkt 0 transformiert werden! Diese Kreise haben nun Radien und diesen Radius kann man anstelle der Iteration verwenden. Der Vorteil ist nun, daß ein mitten in einem einfarbigen Band liegender Punkt in den Kreis transformiert wird, und zwar zwischen

```
2 solcher Kreise. Bildet man nun einen Kreis, auf dem dieser Punkt liegt und dreht die
Funktion um, so erhält man ein perfektes Zwischenband!
Diese Methode wird also dazu benutzt, jedem Punkt eine beliebige reelle Zahl zuzuordnen, so
daß Treppenstufen in der 3D-Darstellung der Vergangenheit angehören.
Mit 'Mult.' kann hierbei eingestellt werden, mit welcher Zahl diese reelle Zahl
multipliziert wird. Mein Programm merkt sich keine reellen Zahlen, da dies einfach zu viel
Speicher kosten würde. Also merkt es sich hier ganze Zahlen, aber eben mit 'Mult' multipliziert.
So heißt das, daß ein 'Mult'-Wert von 100 100 Zwischenstufen zwischen 2 Iterationen zuläßt, was
ausreicht, um den Treppeneffekt zu umgehen.
Beim Abspeichern als 24 Bit-Fraktal werden diese 100 Zwischenstufen in Zwischenfarben
umgerechnet, was sanfte Farbübergänge zuläßt.

Innenfärbung
- Nun gibt es sogar 6 Möglichkeiten:
Farbe - Infimum - Infimumsindex - Supremum - Supremumsindex - Betrag von z
Das Innengebiet ist für gewöhnlich recht einfarbig. Doch das muß nicht sein. Es
besteht die Möglichkeit, auch das Innengebiet farbig zu gestalten. Sei (z,z1,z2,z3,z4,...,zn)
die Bahn des Punktes z, n ist also die Maximalzahl an Iterationen.

Infimum
Hier wird das Infimum ( bei Computern ist's das Minimum... ) eines Punktes vom Startwert
berechnet, also das Minimum von |z1-z|,|z2-z|,|z3-z|,...,|zn-z|. Der erhaltende Wert wird
mit 'Multiplikator' multipliziert und als ganze Zahl abgespeichert.

Infimumsindex
Hier wird der Index des Infimums berechnet, also die Anzahl an Iterationen, bei
der das Minimum aufgetreten ist. Ist das Minimum von |z1-z|, |z2-z|, |z3-z|, ..., |zn-z|
gleich |z3-z|, so ist der Index 3

Supremum
Hier nun wird das Supremum ( bei Computern das Maximum... ) eines Punktes vom Startwert
berechnet, also das Maximum von |z1-z|,|z2-z|,|z3-z|,...,|zn-z|. Der erhaltende Wert
wird mit 'Multiplikator' multipliziert und als ganze Zahl abgespeichert.

Supremumsindex
Hier wird der Index des Supremums berechnet, also die Anzahl an Iterationen, bei
der das Maximum aufgetreten ist.

Betrag von z
Nach der Maximalzahl an Iterationen wird die Länge von z berechnet, mit 'Multiplikator'
multipliziert und abgespeichert.

Angeregt worden bin ich hierbei von 'The Beauty of Fractals', Seite 62
```

```
Abbruchbedingungen
Hier kann festgelegt werden, welche Attraktoren von welcher Art als ↩
   Abbruchbedingungen
zugelassen werden.

1. unendlich
Falls angewählt, wird jeder Punkt untersucht, ob er vom unendlich fernen Punkt
angezogen wird.

2. endlich
Falls angewählt, wird jeder Punkt daraufhin untersucht, ob er von irgendeinem ↩
   endlichen
Fixpunkt angezogen wird

3. Zyklussuche
Falls angewählt, wird jeder Punkt daraufhin unterscht, ob er von irgendeinem ↩
   Zyklus
angezogen wird. Dazu muß man noch in 'Start' eine Iterationszahl eintragen, ab der
mit der Suche begonnen werden soll. Sie sollte in etwa die Hälfte der Iterationen ↩
   sein.
Dies ist nötig, da ein Punkt auf seiner Bahn normalerweise erst einmal ziemlich
kopflos herumwandert, bis er sich einmal entschließt, zu einem Attraktor ↩
   hinzuwandern.
Man sollte solchen Punkten also Gelegenheit zu einem Entschluß geben.

4. benutzerdefinierter Punkt
Falls angewählt, wird jeder Punkt daraufhin untersucht, ob er speziell von dem
rechts daneben stehenden Punkt angezogen wird oder nicht.

Bailout
Falls jeder Punkt daraufhin geprüft wird, ob er von dem unendlich fernen Punkt
angezogen wird, so ist natürlich die Frage, wie man das am geschicktesten prüft.
Folgende Methode wird eigentlich überall angewandt (Ausnahme: {"Biomorphie" LINK ↩
   ExpertJM_Bio}):
Man definert einen Kreis um den Ursprung mit Radius 'Bailout'. Fällt ein Punkt auf ↩
    seiner Bahn außerhalb des Kreises,
so schätzt man, daß er ins unendliche flieht.

Bailin
Falls jeder Punkt daraufhin untersucht wird, ob er von einem endlichen Punkt oder
Zyklus angezogen wird, so definiert man einen Kreis um diesen endlichen Punkt mit
dem Radius 'Bailin' und sagt, der Punkt wird von ihm angezogen, wenn er auf seiner
Bahn einmal in den Kreis fällt.

Theoretisches hierzu:
2.3.2.1 Theorie: Juliamengen
2.3.2.2 Theorie: Mandelbrotmengen
```

1.19 2.3 Fraktale --- 2.3.2 Julia- und Mandelbrotmengen

```
2.3.2.5 Parameterwindow 3

Kreisinversion
- Dies ist eine geometrische Transformation. Hierbei wird alles, was außerhalb
des mit 'Mittelpunkt' und 'Radius' definierten Kreises liegt, ins Innengebiet des
```

Kreises transformiert und alles, was innerhalb liegt, nach außen.

Biomorphie
- Normalerweise werden mittels Bailout und Bailin Kreise definiert. Wenn Punkte auf ihren Bahnen außerhalb bzw. innerhalb dieser Kreise zu liegen kommen, so wird abgebrochen. Nun könnte man auf die Idee kommen, keine Kreise, sondern Rechtecke oder sternförmige Gebiete zu definieren und zu testen, ob und wann ein Punkt außerhalb
bzw. innerhalb dieser Gebiete zu liegen kommt. Das Ergebnis sind Fraktale, die ein bischen an Mikroorganismen erinnern, daher der Name. Die genauen Abbruchbedingungen sind
mathematisch nun wie folgt definiert:
(x ist der Realteil von z, y der Imaginärteil)

abs(x)+d*abs(y)>Bailout
und/oder
d*abs(x)+abs(y)>Bailout

bzw.
abs(x)+d*abs(y)<Bailin
und/oder
d*abs(x)+abs(y)<Bailin

Ob diese beiden Ungleichungen mittels 'und' oder 'oder' verknüpft wird, kann man einstellen, die Variable d bezeichne ich im Programm mit Biomorphievariable.

Setzt man d=0, so ergibt sich:
abs(x)<Bailin
und/oder
abs(y)<Bailin

Im Falle von 'und' also ein Quadrat, im Falle von 'oder' ein Kreuz.

Für d=1/10 ergibt sich ein sternförmiges Gebiet, usw.

Dekomposition
- Hier wird das Außengebiet in Winkelfelder eingeteilt. Die Anzahl der Winkelfelder
entspricht dem Wert in 'Kodierung'. Zu jedem Bahnendwert wird nun berechnet, in welchem Winkelfeld von 0 bis 'Kodierung'-1 er zu liegen gekommen ist und so eingefärbt.

Farbverteilung
Mit diesen Gadgets können verschiedene Farbverteilungen eingestellt werden. Das Programm berechnet ja zu jedem
Bildschirmpunkt einen Iterationswert, oder, allgemeiner gesagt, eine Zahl. Mit diesem Cyclegadget und dem Gleitkommagadget
daneben hat man nun eine gewisse Kontrolle darüber, in welcher Weise den verschiedenen Zahlen die Farben zugeordnet
werden. Im folgenden die Aufzählung der Funktionen, der Inhalt des Gleitkommagadgets wird mit 'Shift'
bezeichnet bzw. abgekürzt.

1. Linear: Die Zuordnug geschieht linear, d.h. Farbnummer=Zahl*Shift.

2. Sinus: Farbnummer=abs(sin(Zahl*Shift/100)*AnzahlFarben). Dieser Modus ist auch in Verbindung mit dem Colorcycling interessant, wie

```
man durch ein bischen Nachdenken leicht feststellt, da die Sinusfunktion keine ↩
    monotone Funktion ist, sie fällt mal, dann steigt sie wieder, etc. (ja, ich ↩
    weiß,
der Fachausdruck heißt 'periodisch'...)

3. 0.75: Farbnummer=(Zahl*Shift)^0.75. Dieser Modus ist interessant, da bei tiefen ↩
     Zooms viele stark unterschiedliche
Iterationswerte auftreten, was in anderen Fraktalgeneratoren sehr oft zu einem ↩
    Bild führt, bei dem man den Anschein
hat, daß die Farben per Zufallsprinzip ausgewählt wurden. Mit diesem Modus nun ↩
    werden Iterationswerte auf dieselbe Farbe
abgebildet, wie man leicht nachprüft.
Kurz eine Tabelle:
2.5 - 4.3 auf Farbe 2
...
8.5 - 10.9 auf Farbe 5
...
184 - 189 auf Farbe 50
190 - 194 auf Farbe 51
..
1169 - 1177 auf Farbe 200
...

4. Log: Farbnummer=Log((Zahl/10)+1)*20*Shift Selbiges wie 3, aber in verstärkter
Form, d.h. je höher die Iterationswerte, umso mehr solcher Werte werden auf ↩
    dieselbe Farbe
abgebildet.

5. Arctan: Farbnummer=abs(ArcTan(Zahl*Shift/100))/(pi/2)*AnzahlFarben. Eigentlich ↩
    ähnlich
wie 0.75 und Log, allerdings tritt hier dieser Effekt, daß viele hohe ↩
    Iterationswerte auf dieselbe Farbe abgebildet werden, noch viel stärker auf, da
die Arcustangens-Funktion durch pi/2 beschränkt ist...

Wie man sieht, ist 'Shift' mehr zur Kosmetik, um das Fraktal schön zu machen, das ↩
    Cyclegadget
bestimmt die Qualität. Beide sind sehr wichtig. Fehlt eines der beiden, wäre alles ↩
     ziemlich
sinnlos.

Theoretisches hierzu:
2.3.2.1 Theorie: Juliamengen
2.3.2.2 Theorie: Mandelbrotmengen
```

1.20 2.3 Fraktale --- 2.3.2 Julia- und Mandelbrotmengen

```
2.3.2.6 Das Datenwindow

Mit dem Infowindow können die wichtigsten Daten zu den Julia-/Mandelbrotmengen
ausgegeben werden. Das Window wertet die Mauszeigerposition aus und errechnet ↩
    nochmal
alle Werte an der betreffenden Stelle.
- Im Feld 'Grund' wird angezeigt, warum die Berechnung abgebrochen wurde.
- Im Feld Iterationen, wann das der Fall war.
- Punkt und Start enthalten bei Juliamengen identische Werte. Im Punkt-Feld ist
```

der dem Bildschirmpixel (Felder Pixel x und Pixel y) entsprechende komplexe ↩
Zahlenwert
angegeben. In Start ist die Initialisierung von z angegeben. Bei Juliamengen eben
dasselbe wie bei 'Punkt'. Bei Mandelbrotmengen ist das der kritische Wert, der ↩
durchaus
aus der Auswertung einer Formel erhalten worden sein kann, z.B. bei der 2. Formel
durch Auswertung der Formel m/(2m-2)

- Maxima & Minima geben die maximalen und minimalen Werte im Innengebiet und im
Außengebiet an. Im Außengebiet sind das die Iterationswerte und als Wahl für ↩
LogMin und LogMax
im Parameterwindow 3 geeignet.
Im Innengebiet geben sie die Werte in Abhängigkeit von den gewählten ↩
Einfärbungsalgorithmen an.
Werte von -1 geben an, daß keine vernünftigen Werte vorliegen.

Theoretisches hierzu:
2.3.2.1 Theorie: Juliamengen
2.3.2.2 Theorie: Mandelbrotmengen

1.21 2.3 Fraktale --- 2.3.2 Julia- und Mandelbrotmengen

2.3.2.7 Der Formeleditor

ChaosPro verfügt über einen relativ leistungsfähigen Formeleditor. Mit dessen ↩
Hilfe
ist es möglich, eigene Formeln zu kreieren und die Julia- bzw. Mandelbrotmengen
davon zu untersuchen.

1. Die Gadgets
Formeln
- In diesem Listview kann die Formel gewählt werden, von der die Juliamenge bzw.
Mandelbrotmenge gezeichnet werden soll. Wie in den Kapiteln 2.3.2.1 2.3.2.2 ↩
erklärt,
hängt das Aussehen der Fraktale im wesentlichen von ihren anziehenden Fixpunkten ↩
ab. Die Fixpunkte erhält man durch
Lösen der Gleichung f(z)=z, also z.B. bei f(z)=exp(z)-z-c durch Lösen der ↩
Gleichung
z=exp(z)-z-c. Hat man die Lösungen, so berechnet man die Ableitung, im Beispiel
wäre dies f'(z)=exp(z)-1 und setzte seine Fixpunkte ein. Von der entstehenden ↩
komplexen
Zahl berechnet man den Betrag. Ist der Betrag kleiner 1, so ist der Fixpunkt ↩
anziehend.

Die ersten 6 Formeln sind im Programm eingebaut, daher schneller, als wenn sie
mittels des Formeleditors erstellt würden.

Formelname
Im Stringgadget unter dem Listview wird der Name der Formel angezeigt. Er läßt ↩
sich auch
ändern, was jedoch nicht unproblematisch ist: Werden nämlich die Daten eines ↩
Fraktals mit einer
benutzerdefinierten Formel gespeichert, so wird dabei lediglich der Name der ↩
Formel in der Struktur eingetragen und

mitabgespeichert. Beim Wiedereinladen der Daten sucht dann das Fraktal nach einer Formel
mit dem Namen, den es in seiner Datenstruktur findet. Was aber, wenn diese Formel nun einen
anderen Namen hat? Das Programm weist dem Fraktal dann die Standardformel zu, wenn es die richtige Formel nicht
findet.

Formel
Hier wird die Berechnungsformel angezeigt. Die ersten 6 Formeln sind fest eingebaut, somit kann man sie nicht
ändern. Ansonsten braucht man lediglich eine neue Formel eintippen und die Return-Taste drücken.

Formelinitialisierung
Das zweite Stringgadget ist nur für die Mandelbrotmenge von Bedeutung.
Damit kann man den kritischen Punkt, der ja vom Parameter (der wiederum vom Bildschirmpunkt abhängt und
sich ständig ändert), durch eine Formel in Abhängigkeit vom Parameter ständig zu Beginn
einer neuen Iteration berechnen lassen.

Formel hinzufügen
Eine Formel wird hinzugefügt, indem auf das Feld 'Formel hinzufügen' geklickt wird.
Das Listview erhält daraufhin einen neuen Eintrag mit der Standardformel z^2+c.
Diesen neuen Eintrag kann man anklicken, worauf die Stringgadgets anwählbar werden, falls sie
es nicht waren. Nun kann man so ein Gadget aktivieren und die Formel ändern. Durch Drücken
der Return-Taste wird der Parser gestartet und, falls die Übersetzung glückt, die neue Formel
im Listview eingetragen.

Formel löschen
Na ja, hiermit wird die Formel gelöscht. Jedes Fraktal, das diese Formel benutzte, wird die Standardformel zugewiesen
bekommen und neuberechnet.

Formel laden/speichern
Na ja, damit werden Formeln geladen und gespeichert.

2. Der Parser und seine Funktionen
Der Parser unterscheidet nicht zwischen Groß-und Kleinschreibung. Mehr als 2 Parameter
sind nicht erlaubt - sie haben keinen Platz im Window für die Parameter von Julia-
und Mandelbrotmengen. Mit Parameter 1 ist dort stets der 1. im Alphabet vorkommende
Parameter gemeint.

Der Parser versteht folgende Funktionen:

```
+ - * / ^    - Plus,Minus,Mal,Geteilt,Hoch
sin          - die Sinusfunktion
cos          - die Cosinusfunktion
```

```
tan         - die Tangensfunktion
asin        - die Arcussinusfunktion
acos        - die Arcuscosinusfunktion
atan        - die Arcustangensfunktion
abs         - die Betragsfunktion ( Absolutwert )
ln          - der natürliche Logarithmus
exp         - die Exponentialfunktion
log         - der Logarithmus zur Basis 10
sinh        - der Sinus Hyperbolicus
sqrt        - Square Root, also die Wurzel
tanh        - Tangens Hyperbolicus
cosh        - Cosinus Hyperbolicus
cotan       - Cotangens
cotanh      - Cotangens Hyperbolicus
conj        - das konjugiert komplexe einer Zahl
real        - der Realteil einer komplexen Zahl
imag        - der Imaginärteil einer komplexen Zahl
acot        - der Arcuscotangens
asinh       - Area Sinus Hyperbolicus
acosh       - Area Cosinus Hyperbolicus
atanh       - Area Tangens Hyperbolicus
acoth       - Area Cotangens Hyperbolicus
arg         - das Argument einer Zahl (der Winkel von 0 bis 2*pi)
e           - Konstante: die Eulersche Zahl
i           - Konstante: die Imaginäre Einheit
p           - Konstante: Pi - die Kreiszahl
z           - dieser Wert ist die Variable der Formel und
              sollte deshalb sinnvollerweise vorkommen
12.44       - eine Zahl - oder auch irgendeine andere, wird
              als Konstante behandelt


3. Fehlermeldungen

- "Fehler in der Formel entdeckt. Wahrscheinlich Klammersetzung fehlerhaft. Übersetzung
abgebrochen..."

Hier sollte die Klammersetzung überprüft werden. Der Parser stieß in diesem Fall bei offener
Klammer auf das Ende der Formel oder aber auf eine schließende Klammer, ohne noch
eine öffnende Klammer zur Verfügung zu haben.

- "Fehler in der Formel entdeckt. Ein unbekanntes Zeichen angetroffen. Übersetzung
abgebrochen..."

Hier stieß der Parser zu Beginn seiner Übersetzung auf ein unbekanntes Zeichen und hat
sofort abgebrochen.

- "Fehler in der Formel entdeckt. Irgendetwas stimmt mit den Operatoren nicht... Übersetzung
abgebrochen..."

Dieser Fehler darf nicht auftreten, sonst ist ein Fehler im Parser...
```

```
- "Formel zu komplex. Es sind maximal 2 Parameter erlaubt. Übersetzung abgebrochen ↩
    ..."

Diese Fehlermeldung dürfte klar sein.
Außer z als Variable dürfen nur 2 Parameter vorkommen, bestehend aus einem ↩
   Buchstaben ungleich
e,i,p,z...

- "Fehler in der Formel. Anzahl an Operatoren paßt nicht zur Anzahl an Operanden, ↩
   kurz: entweder zu viele Operatoren oder zu viele Operanden.
Übersetzung abgebrochen..."

Diese Fehlermeldung zeigt an, daß bei einem Testdurchlauf der Formel zum Schluß ↩
   noch Operanden
übrig waren oder aber beim Auswerten plötzlich kein Operand mehr da war, mit dem ↩
   die durch den
Operator bestimmte Aktion ausgeführt werden könnte. Klingt kompliziert, also ein ↩
   Beispiel, das diese
Fehlermeldung provoziert:
a) a**b
Hier versucht er die Formel auszuwerten. Es existieren 2 Operatoren, nämlich 2 mal ↩
    ein Multiplikationszeichen. Multipliziert werden
immer 2 Operanden, man sagt, die Multiplikation ist eine dyadische Operation. Um ↩
   die Formel korrekt auszuwerten, wären aber nun 3
Operanden nötig, es sind aber nur 2 da, nämlich 'a' und 'b'. Hier sind also zu ↩
   viele Operatoren im Vergleich zu der Anzahl an Operanden
vorhanden.
b) b b
Hier ist kein Operator da, aber 2 Operanden. Der Parser startet also einen ↩
   Testdurchlauf, ist sofort fertig und merkt, daß nicht einfach ein
Operand übrigbleibt, den er als Ergebnis interpretieren würde, sondern 2. Also ↩
   sind nun zuviele Operanden
vorhanden...
```

1.22 2.3 Fraktale --- 2.3.3 Bifurkationsdiagramme

```
2.3.3 Bifurkationsdiagramme

2.3.3.1 Theorie

Betrachtet wird das Verhulst-Modell x->a*x*(1-x)

Dieses Modell läßt sich praktisch wie folgt veranschaulichen:
x ist die Population einer Rasse, z.B. von Hasen. Sie ist so skaliert, daß x stets
zwischen 0 und 1 liegt, 0 heißt, es ist kein Hase vorhanden, 1 heißt, es ist alles
voll von Hasen, so voll, daß kein einziger mehr Platz hat. a ist dann die ↩
   Vermehrungsrate,
a=1 würde dann heißen, daß sich die Hasen nicht vermehren. Bleibt noch der Faktor
(1-x) zu erklären. Er ist ein Maß für den Freiraum, der den Hasen bleibt und kann
als Menge des vorhandenen Futters erklärt werden, die auch wieder zwischen 0 und
1 liegt.
Von einem zum nächsten Jahr berechnet sich die Population der Hasen, indem einfach ↩
    aus dem
alten x0 ein neues x1=a*x0*(1-x0) errechnet wird. Betrachtet man das Modell, so ↩
   stellt
```

man fest:
Sei nun a=2 (durchaus denkbar).
Ist in einem Jahr die Population gering, so ist viel Futter vorhanden, was die Population
ansteigen läßt. Ist die Population groß, so ist wenig Futter vorhanden, was die Hasen
sterben läßt.
Die Frage ist nun: Bei welchem Gleichgewicht stellt sich die Population ein?

Ist x0=0.1, a=2, so ergibt sich:
x1=2*0.1*0.9=0.18
x2=2*0.18*0.82=0.2952
x3=2*0.2952*0.7048=0.416
x4=2*0.416*0.584=0.486
x5=2*0.486*0.514=0.50
...
Der Wert erreicht rasch 0.5 und bleibt dort, das heißt die Hasenpopulation würde
im Beispiel zunehmen bis zum Wert 0.5 und dort konstant bleiben.

Nun kann man aber noch die Vermehrungsrate ändern. Die Frage ist nun, was passiert
mit der Hasenpopulation bei verschiedenen Werten von a? Welches Gleichgewicht
wird erreicht? Wird überhaupt ein Gleichgewicht erreicht?

Dieses Modell ist sehr einfach gehalten, trotzdem zeigt es bereits eine überraschende
Vielfalt an Zuständen.
1. Fall: 0<a<=1

In diesem Fall geht x gegen 0, was klar ist, denn a war ja die Vermehrungsrate,
d.h. die Hasen sterben aus.

2. Fall: 1<a<=2

Hier geht die Population rasch in einen Gleichgewichtszustand über, die Folge
der x wächst entweder monoton oder fällt monoton, je nach Anfangswert der Population

3. Fall: 2<a<=3

Auch hier gibt es wieder einen Gleichgewichtszustand, die Folge der x jedoch geht
oszillierend gegen den Grenzwert, nicht monoton.

Nun sei a>3 z.B. 3.1
x1=0.3
x2=0.651
x3=0.704
x4=0.646
x5=...

Rechnet man weiter, so erkennt man, daß x schließlich zwischen 2 Werten hin und
her pendelt, nämlich zwischen den Werten 0.557 und 0.764

Also haben wir hier kein eindeutiges Gleichgewicht, die Hasenpopulation ändert sich
von Jahr zu Jahr sprunghaft.

Vergrößert man a weiter bis zum Wert 3.449489, so pendelt die Population stets

zwischen 2 Werten hin und her. Doch ab diesem Wert passiert es wieder: Eine ↩
Periodenverdoppelung,
d.h. ab hier pendelt der Wert von x schließlich zwischen 4 (!) Werten hin und her.
Bei a=3.5441 geht dann dieser 4-er Zyklus in einen 8er-Zylus über. All diese Werte
von a, bei denen die Zykluslänge verdoppelt wird, heißen Bifurkationspunkte.

Dieser 8er-Zyklus geht in einen 16er-Zyklus, dieser in einen 32er-Zyklus etc. über ↩
, bis
zu einem bestimmten Wert: a=3.569946
Ab diesem Wert bis zum Wert a=4 passiert wieder etwas sehr seltsames:
Erst wird das Ganze einmal chaotisch, d.h. x pendelt zufällig zwischen ↩
irgendwelchen
Werten hin und her, an diesen Stellen ist der Attraktor kein Zyklus irgendeiner ↩
Länge, sondern
ein eindimensionales Fraktal.
In diesem Bereich bis 4 jedoch gibt es noch sogenannte "Fenster", z.B. bei a=3.83, ↩
wo
wieder ein Zyklus der Länge 3 vorherrscht, der in einen Zyklus der Länge 6, dann
12,24,48,... übergeht. Solche Fenster finden sich noch weit mehr.
Wenn man sich das ganze Modell betrachtet und sich vergegenwärtigt, daß wir es uns
am Beispiel der Hasenpopulation deutlich gemacht haben, so überrascht es doch,
daß so seltsame Dinge auftreten können. Ein unwissender Leser hätte doch stets
vermutet, es müsse irgendein Gleichgewicht vorherrschen.

1.23 2.3 Fraktale --- 2.3.3 Bifurkationsdiagramme

2.3.3.2 Parameterwindow 1

Formel
- Im theoretischen Kapitel wurde die Verhulst-Formel a*x*(1-x) besprochen. Analog
kann man jedoch die Gedankengänge auch auf andere Formeln anwenden. Im Programm
wurden 5 der wichtigeren Formeln eingebaut. Von diesem kann man sich das ↩
Bifurkationsdiagramm
zeichnen lassen.

Iteration
- Um das Bifurkationsdiagramm korrekt zeichnen zu können, muß der Anfangswert erst
einmal genügend oft in die Formel eingegeben (iteriert) werden, um ihm Gelegenheit
zu geben, zu einem evtl. vorhandenen Attraktor hinzuwandern. Erst dann kann mit ↩
dem
Zeichnen begonnen werden.
In diesem Programm wird bei den Bifurkationsdiagrammen grundsätzlich so oft ↩
iteriert,
bis die Hälfte des Wertes, der hier angegeben ist, erreicht ist. Dann ist der ↩
Start-
Punkt hoffentlich bei seinem anziehenden Zyklus angelangt. Dann wird bis zum ↩
Erreichen
des hier angegebenen Wertes der Punkt weiter iteriert und nun dabei gezeichnet.
Will man das Bifurkationsdiaramm genauer haben (hat man z.B. eine Verästelung
stärker vergrößert), so erkennt man, daß es sich eigentlich um keine plötzliche
Verästelung handelt, sondern um ein "bandiges" auseinanderdriften der Werte. Dies ↩
ist
auf die Ungenauigkeit des Programms zurückzuführen! Es handelt sich wirklich um ↩
ganz plötzliche

```
Verästelungen. Erhöhen Sie den Iterationswert und das Programm erhöht die ↩
    Ganuigkeit, was
die Sache wieder wie eine Verästelung aussehen läßt.

Variable x/Variable y/beide
- Diese Option ist nur bei Formel Nr. 3 anwählbar. Dort handelt es sich nämlich
um die Formel x=a*x*(1-x-y) und y=a*x*y, es kommen also 2 Variablen vor, also
z.B. Fuchs, Hase (und der Vermehrungsfaktor). Welche Variable gezeichnet werden
soll, bestimmen Sie hier.

a: Minimum - Maximum
- Horizontal im Fraktalwindow wird der Parameter a aufgetragen. Hier bestimmen Sie ↩
    ,
ab welchem und bis zu welchem Wert.

x/y: Minimum - Maximum
- Vertikal im Fraktalwindow wird die Variable x - bzw. bei Formel 3 auch y - ↩
    aufgetragen.
Wieder bestimmen Sie hier den Ausschnitt.

Theorie hierzu:
Kapitel 2.3.3.1
```

1.24 2.3 Fraktale --- 2.3.3 Bifurkationsdiagramme

```
2.3.3.3 Datenwindow

- In den Felder a bzw. x/y werden die der Bildschirmposition entsprechenden Werte
für a bzw. x/y angezeigt.
In den Felder x bzw. y und in den Felder Ende x bzw. Ende y werden die Startwerte,
also die Initialisierungswerte, bzw. die Endwerte nach der darunter eingestellten
Anzahl an Iterationen angezeigt.

- Im Feld Zyklus wird die Länge eines evtl. gefundenen Zyklus angezeigt.

- Durch das Feld 'Zeige Iteration' in Verbindung mit dem Slider wird eingestellt,
nach welcher Iterationstiefe der Wert von x bzw. y in den beiden Feldern Ende x
bzw. Ende y angezeigt wird. Hierdurch kann man untersuchen, welche Werte nach und
nach angenommen werden, ohne den Taschenrechner zu bemühen.

Hinweis:
Durch Drücken der Taste 'I' bzw. Shift+'I' kann dieser Wert aus dem Fraktalwindow
heraus geändert werden.  Es ist  also nicht zwingend nötig, das Datenwindow zu
aktivieren.

Theorie hierzu:
Kapitel 2.3.3.1
```

1.25 2.3 Fraktale --- 2.3.4 Dynamische Systeme

```
2.3.4 Dynamische Systeme
```

2.3.4.1 Theorie

1961 machte der Meteorologe Edward Lorenz Berechnungen mit einem System von mehreren
Differentialgleichungen, d.h. einem System, von dem nicht die konkreten Punkte
bekannt sind, aus denen dann weitere Punkte irgendwie berechnet werden können,
sondern von dem zu jedem beliebigen Punkt die Steigung der Tangente bekannt ist,
also die Ableitung. Aus dieser kann dann eine Näherung für den nächsten Punkt
berechnet werden.
Nun gut, er machte so seine Experimente und stellte fest, daß sein Ergebnis sehr stark
von der Rechengenauigkeit abhängte. Ein sehr kleiner Fehler am Anfang hatte nach
kurzer Zeit ein völlig anderes Ergebnis zur Folge.
Der Titel einer seiner Publikationen war deshalb: "Kann das Flattern eines Schmetterlings
in Brasilien einen Orkan in Texas verursachen?"
Dieser von Lorenz entdeckte Effekt wurde sinnvollerweise dann " Schmetterlingseffekt"
genannt.
Edward Lorenz vereinfachte dann sein Modell stark und experimentierte damit herum.
Dieses Modell bestand dann nur noch aus 3 Differentialgleichungen:

dx/dt=-ax+ay
dy/dt=cx-y-xz
dz/dt=-bz+xy

Die 3 Gleichungen lesen sich so:
Die Ableitung in x-Richtung nach der Zeit t ist -ax+ay
Die Ableitung in y-Richtung nach der Zeit t ist cx-y-xz
Die Ableitung in z-Richtung nach der Zeit t ist -bz+xy

Er gab a den Wert 10, b den Wert 8/3 und experimentierte mit verschiedenen Werten
von c. Die Ergebniskurve ist dreidimensional und nach dem Existenz- und Eindeutigkeitssatz
(das ist nur was für Studenten der Mathematik oder zumindest von solchen, die Mathe
als Nebenfach haben...) aus mathematischer Sicht eindeutig, wenn ein Anfangspunkt
gegeben ist (ein sog. Anfangswertproblem, kurz AWP...). Aber leider eben nur aus
mathematischer Sicht.

Lorenz nahm für c den Wert 28 und berechnete die Kurve für verschiedene Anfangswerte.
Doch er fand, daß nach einer gewissen Zeit stets dieselbe Figur auftauchte. Sie
war zwar kompliziert, nämlich aus unendlich vielen Schleifen aufgebaut, aber seltsam
war das schon. Es schien so, als ob der Raum durch diese 3 Gleichungen eine unendlich
filigrane Struktur aufgeprägt bekam, einen fraktalen Attraktor, zu dem unabhängig
vom Anfangswert der Punkt hingezogen wird. Da diese Figur so merkwürdig war, wurde
sie als "strage attractor" bezeichnet.

Nun, einige Numeriker bezeichnen all dies als "absoluten Schwachsinn" (Zitat!),
da der Schmetterlingseffekt nur auf Rundungsfehler aufbaut, also die ganze Geschichte
nur in der Phantasie der Computerwelt existiert und keinerlei praktische Bedeutung
hat. Dieser Meinung kann ich mich nicht anschließen. Denn Tatsache ist, daß auch

die Natur nur mit Integer-Zahlen rechnet, soll heißen, nichts in der Natur ist unendlich
teilbar. Von allem gibt es ein kleinstes: Die Elementarladung, die Quarks, ein Lichtquant
oder was weiss ich noch was. Das heißt doch aber, daß auch in der Natur zwangsweise
stets Rundungsfehler auftreten. Somit sind die Computer als perfekte Nachahmer der Natur geeignet,
zumindest qualitativ, nicht quantitativ. Die Größenordung der Integer-Zahlen in der
Natur ist doch um einige Äonen größer als die im Computer.

1.26 2.3 Fraktale --- 2.3.4 Dynamische Systeme

2.3.4.2 Parameterwindow 1

Bereich
- Da der Lorenz/Roessler-Attraktor dreidimensional ist, fällt es auf den ersten Blick
schwer, den zu zeichnenden Ausschnitt anzugeben. Ich habe mir dabei auf folgende Art befolgen:
Grundsätzlich sieht man von gerade vorn auf den Attraktor und gibt von diesem Standort
ausgehend die Bereichswerte Links/Oben/Rechts/Unten an, also die x- und y- Koordinaten
des Bereichs.

Ansichtswinkel
- Um nun den Attraktor nicht nur von vorne zu betrachten, sondern aus einer beliebigen
Position, kann man die Ansichtswinkel ändern. Das System, das hier verwendet wird, gleicht dem bei der
Erde verwendeten System: Längen- und Breitenwinkel.
Mittels Alpha wird der Längenwinkel (entspricht dem Längengrad) eingestellt, mit Beta der Breitenwinkel
(also der Breitengrad).

Parameter
- hier können die 3 bei den dynamischen Systemen verwendeten Parameter gewählt werden.
Es empfielt sich, die Parameter langsam zu ändern, da diese Systeme mitunter sehr
stark auf eine Änderung reagieren.

Systemtyp
- Zur Zeit bietet das Programm 2 Typen aus der Klasse der kontinuierlichen dynamischen
Systeme, den Lorenz und den Roessler-Attraktor. Beim Lorenzattraktor hat man als
Ausgangssystem von Differentialgleichungen

dx/dt=-ax+ay
dy/dt=cx-y-xz
dz/dt=-bz+xy

beim Roessler-Attraktor dagegen

```
dx/dt=-y-z
dy/dt=x+ay
dz/dt=b+xz-cz

Theorie hierzu:
Kapitel 2.3.4.1
```

1.27 2.3 Fraktale --- 2.3.4 Dynamische Systeme

```
2.3.4.3 Parameterwindow 2

Startpunkt
- Im Theorieteil wurde erwähnt, daß unabhängig vom Startpunkt die Punktbahn stets
von einem Attraktor, der wegen seines komplizierten Aufbaus als "strage attractor"
gezeichnet wird, angezogen wird. Wer das nicht glaubt, hat hier die Möglichkeit,
unterschiedliche Startpunkte zu wählen. Er kann dann eine Animation berechnen ↩
    lassen und
dort dann bildlich erkennen, welche Auswirkungen eine Änderung des Startpunktes
hat.

Zeiteinstellungen
- Mit 'Zeit' wird die Zeitdauer der Bahnverfolgung festgelegt. Das ↩
    Differentialgleichungssystem
beschreibt ja die Wegeänderung in Abhängigkeit der Zeit. Da aber der Computer mit ↩
    der Ableitung nichts
anfangen kann, muß er dieses dx/dt ( sprich "dx nach dt" ), das gleichbedeutend ↩
    mit dem Grenzwert von
Delta x geteilt durch Delta t für t gegen 0 ist, durch Delta x geteilt durch Delta
t mit hinreichend kleinem Delta t ersetzen. Dieses Delta t wird durch 'Delta' ↩
    wählbar
gemacht.

Zeichengeschwindigkeit
- Der Lorenz- bzw. der Roessler-Attraktor werden recht flott gezeichnet. Das ist ↩
    schön.
Aber zu Demonstrationszwecken bzw. wenn man die Punktebahn mit dem Auge ↩
    mitverfolgen
will, ist es nötig, die Geschwindigkeit zu drosseln. Hiermit ist die ↩
    Geschwindigkeit
einstellbar von 1 (große Verzögerung) bis 100 (keine Verzögerung)

Zeichenmodus
- Hier stehen 3 Möglichkeiten zur Auswahl:
Zeichne als Punkte/zeichne als Linien/Punktansammlung
Die ersten beiden Modi zeichnen den gesamten Lorenzattraktor vom Zeitpunkt 0 bis
zum Zeitpunkt 'Zeit' entweder jeden berechneten Punkt als Punkt oder aber mittles
einer Verbindungslinie zum jeweils letzten Punkt.
Der 3. Modus ist etwas spezielleres:
In Kapitel 2.3.4.1 über die Theorie wurde erwähnt, daß kleine Änderungen am Anfang
zu völlig anderen Ergebnissen führen (Schmetterlingseffekt). Dies kann mit diesem ↩
    Modus
veranschaulicht werden. Dazu wird von einer mehr oder weniger großen Ansammlung ↩
    von Punkten
ausgegangen, die alle recht nahe beieinanderliegen. Dann wird jeder Punkt auf ↩
    seiner
```

Bahn verfolgt. Nach einiger Zeit sieht man, daß die anfangs als einzelner Punkt
erscheinende Punktwolke sich aufteilt und wiederum nach einiger Zeit deckt sie
den gesamten Attraktor ab. Dies verdeutlicht, daß eine mathematische Vorhersage
über die Position eines Punktes zu einer bestimmten Zeit bei diesem System sinnlos
ist, da in der Praxis die Anfangsposition nicht genau bestimmt bzw. festgelegt ↩
werden
kann. Jede noch so kleine Ungenauigkeit am Anfang hat auf Dauer eine ↩
unvorhersehbare
Wirkung. Das einzige, was man sagen kann, ist, daß sich der Punkt auf dem ↩
Attraktor befindet,
wo dort aber genau, ist nicht möglich.
Mittels des Gadgets 'Abstand' kann man den anfänglichen mittleren Punktabstand
der Punkte voneinander einstellen, also den Radius der Punktwolke.
Je näher die Punkte beieinanderliegen, desto länger dauert es, bis der ↩
Schmetterlingseffekt
zu sehen ist.
Im 'Punkte'-Gadget kann die Anzahl der in der Punktwolke vorhandenen Punkte ↩
eingestellt
werden. Das richtet sich hauptsächlich nach der vorhandenen Rechenkapazität und
der Geschwindigkeit der Grafikausgabe, da sehr viele WritePixel-Systemaufrufe hier
nötig sind...
Spaß macht dieser Modus vor allem dann, wenn die Punktwolke aus vielen Punkten
besteht. Das erfordert allerdings viel Rechenkapazität, wenn man systemkonform
bleiben will. Damit man nun nicht einen 200-Mhz-68060 und eine ↩
hardwareunterstützte
WritePixel-Routine braucht, habe ich ein Gadget hinzugefügt, mit dem man einen
Modus einschalten kann, in dem direkt in die Planes hineingezeichnet wird, was ↩
enorme
Geschwindigkeitsvorteile bringt, Grafikkartenbesitzer aber alt aussehen läßt. In
diesem Modus wird radikal alles übermalt, was sich im Einzungsbereich des Windows
befindet, also z.B. das Menü, ein darüberliegendes Window etc. Also bitte darauf ↩
achten,
daß das Window vollständig sichtbar ist...

Theorie hierzu:
Kapitel 2.3.4.1

1.28 2.3 Fraktale --- 2.3.5 Plasma

2.3.5 Plasma

2.3.5.1 Theorie

Plasma ist nichts anderes als eine 2-dimensionale Brown'sche Bewegung. Eine 1- ↩
dimensionale
Brown'sche Bewegung wird folgendermaßen konstruiert: Man geht z.B. von einem Punkt
aus, besorgt sich eine Zufallszahl, wobei der erzeugende Zufalls\zahlen\generator
N(0;1)-normalverteilt sein muß. Man geht von dem Punkt aus horizontal einen ↩
Schritt
nach rechts und um die Zufallszahl nach unten bzw. oben, je nach Vorzeichen ↩
derselben.
Das wiederholt man. Das Ergebnis ist so etwas wie ein Schnitt durch ein Gebirge,
also eine Zickzack-Bewegung mal nach unten, mal nach oben.
Von dieser Methode zur Erzeugung einer Brown'schen Bewegung gibt es mehrere ↩
Varianten.

Die eine ist, daß man z.B. eine Brown'sche Bewegung von Punkt A nach Punkt B ↩
erzeugen
will. Dazu nimmt man die beiden Punkte, verbindet sie durch eine (imaginäre) ↩
Strecke,
holt sich den Mittelpunkt dieser Strecke und versetzt den Mittelpunkt um eine ↩
Zufallszahl,
die noch mit einer Zahl multipliziert wird, die von der gewünschten Dimension der
Bewegung und der Länge des Teilungsintervalls abhägt. Man erhält 2 Strecken, für
die man analog verfährt. Von diesem Algorithmus ist leicht eine 2-dimensionale
Variante denkbar: Man hat ein Viereck und versetzt den Mittelpunkt des Vierecks
um eine Zufallszahl, man macht entsprechendes mit den Randpunkten und erhält ↩
wieder
4 Vierecke, mit denen man analog verfährt. Ungefähr dieser Algorithmus wird in ↩
meinem
Programm angewendet.

1.29 2.3 Fraktale --- 2.3.5 Plasma

2.3.5.2 Parameterwindow 1

Sigma
- Bei jeder Mittelpunktsverschiebung wird eine Zufallszahl geholt, diese mit einer
Zahl multipliziert, die von der Dimension und von der Länge des Intervalls abhängt ↩
.
Diese Zahl ist die Basis dieses Multiplikators, d.h. bei der ersten Mittelpunkts\ ↩
verschiebung
wird die Zufallszahl mit dieser Zahl multipliziert, die weiteren Mittelpunkte der
kleineren Vierecke werden mit Teilen dieser Zahl multipliziert. Die genauen 'Teile ↩
'
hängen von der gewünschten Dimension ab.

H
- Diese Zahl bestimmt direkt die Dimension des Objekts. Die resultierende ↩
Dimension
ist 3-H, d.h. ist H z.B. 0.9, so ist die Dimension 3-0.9=2.1, also ist das ↩
Ergebnis
ein bischen rauher als eine Fläche. Ist H=0.1, so ist die Dimension allerdings ↩
2.9,
was eine sehr rauhe Fläche, die lokal beinahe raumfüllend ist, zur Folge hat. Man
kann sich das Gebilde als einen Berg mit sehr vielen sehr hohen Zacken vorstellen,
eben beinahe raumfüllend...

ColorMult
- Der Ergebniswert wird mit dieser Zahl multipliziert, das Ergebnis als Farbnummer
interpretiert. Dieser Parameter hat eine ähnliche Wirkung wie Sigma, allerdings ↩
ändert
er nicht die Werte im Puffer, sondern veranlaßt eine andere Interpretierung. Das
Plasmafraktal wird bei einer Änderung dieses Parameters nicht neu berechnet, es ↩
wird
lediglich der Puffer neu dargestellt, was die Geschwindigkeit erhöht.

Reihe
- Da mit Zufallszahlen gearbeitet wird und eine Zufallszahlenreihe auf einem ↩
Computer

```
deterministisch ist, ist es nötig, einen Startwert für die Reihe zu wählen, damit ↩
    nicht
immer dasselbe Gebilde herauskommt.

Theorie hierzu:
Kapitel 2.3.5.1
```

1.30 2.3 Fraktale --- 2.3.6 Lyapunov-Raum

```
2.3.6 Lyapunov-Raum

2.3.6.1 Theorie

Der Lyapunov-Raum baut auf den Bifurkationsdiagrammen auf. Dort wurde eine Formel
benutzt, die z.B. die Populationsentwicklung beschreibt. In Abhängigkeit von der
Wachstumsrate wurde graphisch dargestellt, ob , und wenn ja, welches Gleichgewicht ↩
     in der Bevölkerung herrscht ( kein Gleichgewicht, also
chaotisches Verhalten, eindeutiges Gleichgewicht, oder ein Zyklus einer bestimmten
Länge ).
Beim Lyapuov-Raum führt man noch einen 2. Wachstumsfaktor ein, der sich mit dem ↩
    ersten in einer definierten
(aber einstellbaren) Art und Weise abwechselt.
Als Beispielsequenz sei z.B. AAABB gegeben, sei A=3, B=2
Veranschaulicht heißt das, daß meine Hasenpopulation im einen Jahr z.B. mit einem
Faktor von 3 wächst, im nächsten dann wieder um den Faktor 3, dann wieder um den ↩
    Faktor
3, dann plötzlich um den Faktor 2, dann wieder um den Faktor 2, dann beginnt das ↩
    ganze wieder von vorn, also
sie wächst wieder um den Faktor 3 usw.

Das ganze spielt man nun für alle möglichen Werte von A und B als Wachstumsraten
durch, trägt dabei A in der Horizontalen und B in der Vertikalen graphisch auf.
Nun muß man sich noch überlegen, was man an der Stelle macht. Es bietet sich an,
an der durch A und B definierten Stelle einen Bildschirmpunkt in einer bestimmten
Farbe zu zeichnen. Doch in welcher Farbe?
Bei den Bifuraktionsdiagrammen war eine globale Unterscheidung in 2 Hauptklassen
möglich:
1. Klasse: Werte des Wachstumsfaktors, die zu einem Gleichgewicht führt, zu einem
Zyklus irgendeiner Länge
2. Klasse: Werte des Wachstumsfaktors, die zu keinem Gleichgewicht führen.

Diese Klasseneinteilung machen wir hier nun auch und färben einen Punkt ↩
    entsprchend ein.
Die Frage bleibt, wie wir entscheiden wollen, ob ein Gleichgewicht oder Chaos ↩
    vorherrscht.
Nun, betrachten wir die Formel f(x)=a*x*(1-x)
Wir legen fest, daß Ordung vorherrscht, wenn der Durchschnitt aus dem Betrag der
Ableitung der Funktion f(x) nach hinreichend vielen Jahren kleiner 1 ist, sonst ↩
    liegt
Chaos vor. Doch müssen wir, wie bei den Bifurkationsdiagrammen, dem Wert erst ↩
    einmal
Gelegenheit geben, zu einem Attraktor hinzuwandern. In der Praxis bilden wir noch ↩
    kurz den
Logarithmus, so daß wir algorithmisch erhalten:
```

```
X=0.5
( müssen X erst mal Gelegenheit geben, zum Attraktor zu wandern...)
FOR N=1 TO 4000
   (R ist A oder B, je nachdem, wie die Sequenz aussieht)
   X=R*X*(1-X)
NEXT N

Summe=0
FOR N=1 TO 6000
   (R ist A oder B, je nachdem, wie die Sequenz aussieht)
   X=R*X*(1-X)
   Summe=Summe+Ln|R-2*R*X|
NEXT N
Summe=Summe/6000

So, das Ergebnis, in der Variablen 'Summe' ist nun der Durchschnitt aus den Größen
Ln|R-2*R*X|, also dem Logarithmus der Ableitung der Formel f(x) und repräsentiert
die Rate, mit der die Größe von aufeinanderfolgenden Werten der Funktion wächst. Das Ergebnis wird auch Lyapunov-Exponent genannt.
Ist der Durchschnitt negativ ( d.h. also |R-2*R*X| ist kleiner als 1 im Durchschnitt ),
so herrscht Ordnung vor, ansonsten Chaos.

Chaos färben wir einfarbig ein, meistens schwarz. Ich habe testweise das Chaos
eingefärbt  und festgestellt, daß es nicht umsonst als Chaos bezeichnet wird...
Falls ein Wert kleiner 0 herauskommt ( d.h. Ordnung, in der Praxis erhält man Werte meist
so bis -5 ) so multiplizieren wir die Zahl noch geeignet, wandeln es in eine ganze Zahl
um und färben den Punkt mit dieser Farbe ein. Das wars dann schon...
```

1.31 2.3 Fraktale --- 2.3.6 Lyapunov-Raum

```
2.3.6.2 Parameterwindow 1

Formel
- Diese Formeln sind identisch mit den Formeln bei den Bifurkationsdiagrammen,
lediglich Formel Nr. 3 fehlt, da ich nicht wußte, was ich hier mit der Ableitung
der Funktion anfangen sollte.

ExpMin
- Hier kann man den minimalen Exponenten einstellen. Die Farben werden automatisch
bestmöglich auf die Lyapunov-Exponenten von ExpMin bis 0 verteilt. Alle Werte, die einen kleineren Lyapunov-Exponenten
ergeben als ExpMin, werden mit Farbe 4 eingefärbt.

Start x
- Wer den Lyapunov-Raum betrachtet, dem werden gewisse Zacken auffallen, die sich
kreuzen. Sehr bemerkenswert ist, daß die Lage der Zacken, also ob der eine vor dem
anderen oder hinter dem anderen liegt, vom Startwert für x abhängt, den man hiermit
einstellen kann.

Sequenz
```

- Hier kann man das Schema der Abwechslung zwischen den beiden Wachstumsraten angeben.
Damit die Änderung wirksam wird, muß das Gadget mittels Tab-oder Return-Taste verlassen
werden.
Wer also die 7 fetten und die 7 mageren Jahre nachbilden will, muß hier nur AAAAAAABBBBBBB
eingeben...

Passes
- Um einen schnelleren Eindruck vom Aussehen des Lyapunov-Raumes zu bekommen, kann
man erst einmal die Auflösung künstlich verkleinern wie man es schon von den Ju\ lia-/Man\del\brot\mengen
gewohnt ist. Wem sich nun die Nackenhaare aufstellen, weil er meint, daß ich diese
Methode zur Verschnellerung einsetzte (wie bei Julia/Mandel), der darf sich nun gleich
wieder kämmen. Keine Angst, hier wird nur vergröbert, durch Wahl von 3 Zeichendurchgängen
wird das Fraktal langesamer gezeichnet als mit einem (wenn auch nur unbedeutend langsamer),
aber man bekommt sehr viel schneller einen Eindruck vom Aussehen und kann gleich
weiter hineinzoomen.

Chaosfarbe
- Name sagt alles, oder?

Stabilisation
- Dieser Wert gibt an, wie oft die Formel iteriert wird, ohne daß der Exponent berechnet wird. Dies gibt dem Startwert Gelegenheit,
zu seinem Attraktor hinzuwandern.

Iteration
- Die hier angegebene Zahl gibt an, wie oft die Formel dannach noch iteriert wird, um
den Lyapunov-Exponenten zu berechnen.
Der Geschwindigkeit zuliebe rate ich euch, diese Werte erst recht klein zu wählen und sie dann testweise zu erhöhen, um zu
sehen, ob sich was weltbewegendes ändert...

Bereich
- dürfte allen klar sein...
A wird horizontal aufgetragen, B vertikal

Theorie hierzu:
Kapitel 2.3.6.1

1.32 2.3 Fraktale --- 2.3.6 Lyapunov-Raum

2.3.6.3 Datenwindow

Im Datenwindow werden die dem Bildschirmpunkt entsprechenden Wachstumsraten, also A und B, angezeigt und außerdem der
Lyapunov-Exponent nochmal kurz berechnet.

Theorie hierzu:

Kapitel 2.3.6.1

1.33 2.3 Fraktale --- 2.3.7 3D-Ansichten

2.3.7 3D-Ansichten

2.3.7.1 3D-Einführung

Das 3D-Modul ist als recht eigenständiges Modul ausgelegt. Es nimmt keine Rücksicht darauf,
zu welchem Fraktal (wenn überhaupt) die Daten gehören, die es in einem Array übergeben
bekommt. Das hat natürlich Nachteile, da so keine Rückkoppelung zum Fraktal möglich
ist, falls mehr Werte zur korrekten Berechnung benötigt würden. Der Vorteil ist jedoch,
daß in einer späteren Version des Programms ein Außenstehender eine Tabelle mit Höhenwerten
aufbauen und von meinem Programm dann umsetzen lassen kann. So spart man sich eine Menge Arbeit.
Außerdem kann ich irgendwelche Höhenwerttabellen von z.B. Scenery Animator oder
einem ähnlichem Programm kurz in mein Format umrechnen, so daß sie von diesem benutzt
werden können. Wie gesagt, vieles ist möglich, was letztendlich eingebaut wird, entscheidet
sich in der Hauptsache durch eure Reaktionen...

1.34 2.3 Fraktale --- 2.3.7 3D-Ansichten

2.3.7.2 3D-Parameterwindow 1

Projektionsart
Es stehen 2 Projektionsarten zur Verfügung:
1. Orthogonal: Die bevorzugte, gute Methode. Hier wird jeder Punkt einfach per orthogonaler
Projektion auf eine 2D-Fläche abgebildet. Eine Entfernug spielt keine Rolle, die
Implementation eines horizontalen Ansichtswinkels ist nicht möglich, ich habe es
versucht. Man benutze statt dessen eine Rotation des 2D-Bildes. In diesem Modus wird
stets das gesamte Bild in bestmöglicher Qualität gezeichnet.
2. Perspektive: Die alte, schlechte, Methode, nur auf Wunsch eines meiner Lieblingsbetatesters
wieder implementiert. Hier wird das 3D-Objekt auf eine 2D-Fläche projiziert, so
wie es das menschliche Auge macht. Hier können auch noch andere Dinge gewählt werden.
Der große Nachteil bei diesem Algorithmus ist, daß das gesamte Zeichnen in keinster
Weise für den Computer vereinfacht werden kann, was diesen Modus sehr langsam macht.
Der gesamte Programmcode ist auch sehr unübersichtlich, was Erweiterungen zu einer
wahren Qual werden läßt.

Zeichenmodus
Nur bei Projektionsart=Projektion...
Hier kann man wählen, wie die einzelnen Werte dargestellt werden sollen:
1. Punkte: zeichnet nur die Punkte
2. Gridlinien: zeichnet die Verbindungslinien zwischen den Punkten
3. Vierecke: zeichnet aus jeweils 4 Punkten ein Viereck
4. Spitzen: zeichnet Spitzen vom Grund aus.
5. Mosaik: zeichnet einfach Plättchen der Größe 2x2 an den Kreuzungspunkten des Rasters
6. Kreuz: zeichnet an den Kreuzungspunkten Kreuze.
Am besten, man probiert es aus...

Entfernung
Nur bei Projektionsart=Projektion...
Name sagt alles...

H-Winkel
Nur bei Projektionsart=Projektion...
- Der Horizontalwinkel, von dem aus der Beobachter auf das Objekt blickt. Er entspricht
dem Breitengrad der Weltkugel.

V-Winkel
- Der Beobachter steht stets direkt vor dem Fraktal und blickt von einer gewissen Höhe aus auf das Fraktal. Die Höhe wird durch den Vertikalwinkel festglegt, der dem Breitengrad der Weltkugel entspricht.

Licht
- Falls angewählt, ist eine Lichtquelle vorhanden, die unendlich weit entfernt ist. Die Position der Lichtquelle wird durch den Horizontal- und den Vertikalwinkel
definiert, die dem Längen- bzw. Breitengrad auf der Erdkugel entsprechen. Angenommen,
Amerika liegt bei 0, dann wäre Europa ungefähr bei H-Winkel 90, Japan ungefähr bei -90. Falls nicht angewählt, werden
die Originalfarben zur Einfärbung herangezogen.

Diffuse
Umgebung
Reflektion
- Die Helligkeit einer Fläche wird durch diese 3 Werte festgelegt: 'Umgebung' ist eine Zahl
zwischen 0 und 1 und bestimmt, wieviel Licht auf jede Fläche fällt, Betonung auf -jede-,
unabhängig davon, ob nun das Licht daraufscheint oder nicht. Hier einen Wert von 1 einzutragen, ist sinnlos,
da dann jede Fläche mit einer Helligkeit von 1 (volle Lichtstärke) gezeichnet würde.
'Diffuse' bestimmt, in welchem Verhältnis das Licht von der Lichtquelle zu dem reflektierten Licht steht.
Ein Wert von 0.8 heißt, daß 80% der Intensität einer Fläche durch den Winkel, in dem die Lichtquelle darauf
scheint, bestimmt wird und 20% der Intensität einer Fläche durch den Winkel, den der Vektor des
reflektierten Lichtes mit dem Sichtvektor des Beobachters einschließt. Allgemein definiert er, ob das 3D-Bild durch reflektiertes
Licht oder durch Lichtquellenlicht sichtbar wird.

'Reflektion' bestimmt, wie stark die Flächen das Licht reflektieren. 1 heißt stark ↩
, 2 weniger stark, 0.5 sehr
stark, etc.

GridX und GridY
Nur bei Projektionsart=Projektion...
- Hier kann man die Auflösung in X- und in Y-Richtung festlegen. Eine geringere ↩
Auflösung hat eine Verschnellerung
des Zeichenvorganges zur Folge...

1.35 2.3 Fraktale --- 2.3.7 3D-Ansichten

2.3.7.3 3D-Parameterwindow 2

DeltaX/Y
- Das Objekt selber wird um den Nullpunkt herum gezeichnet. Um es zu verschieben,
dienen diese beiden Schieberegler.

DeltaZ
- Hiermit wird das Objekt in der Höhe verschoben.

Invers
- Invertiert alle Höhen. Aus dem ehemaligen Mandelbrotberg wird ein Mandelbrottal.

Autoadjust
- Versucht, das Fraktal so zu verschieben und zu strecken, daß es gut ins Window
paßt.

FrontMult
BackMult
- Bestimmt, mit welchen Zahlen die Höhenwerte vorne bzw. hinten noch multipliziert
werden. Normalerweise wählt man beide Werte gleich. Will man jedoch dem Objekt ↩
mehr
Plastizität verleihen, empfiehlt es sich, BackMult etwas größer zu wählen.
Die Werte in der Mitte werden mit (FrontMult+BackMult)/2 multipliziert, es geht
also linear von vorne nach hinten.

Steigung
- Dieser Wert bestimmt, wie steil die Hänge bzw. Täler des Bildes sein sollen.
Kleinere Werte ==> weniger steil
Größere Werte ==> steiler
Die Funktion lautet exakt:
x^(1/Steigung)
,wobei x die Höhe bezeichnet, d.h. Steigung=2 ==> x^0.5, also Wurzel aus x.

YStretch
Dieser Wert bestimmt den Multiplikationsfaktor für die y-Richtung des 3D-Objektes.
Diese Richtung geht sozusagen nach 'hinten'. Falls das 3D-Bild zu 'kurz' erscheint ↩
,
ist dieser Wert eben größer als 1 zu wählen.

Wasser
Plateau
Mit diesen Werten kann man Höhen abschneiden. Ist eine Höhe kleiner als der durch ↩
'Wasser'

eingestellte Wert, dann wird diese Höhe auf 'Wasser' gesetzt. Ist eine Höhe größer ↩ als
'Plateau', dann wird die Höhe auf 'Plateau' gesetzt. Die Farbe dieser Regionen
wird ganz natürlich durch den Winkel bestimmt, in dem das Licht einfällt.

1.36 2.3 Fraktale --- 2.3.7 3D-Ansichten

2.3.7.4 3D-Parameterwindow 3

zu benutzende Farben
Hier stellt man die Farben ein, die zur Einfärbung benutzt werden sollen. Die ↩ erste 3D-Farbe wird
als 'schwarz' angesehen, die letzte 3D-Farbe als 'weiß' und entsprechend ↩ eingefärbt, d.h. je nachdem,
wie stark das Licht auf die Fläche scheint, wird eine Farbe aus dem Bereich von ' ↩ erste 3D-Farbe'
bis 'letzte 3D-Farbe' genommen.

Hintergrund
Zu Beginn des Zeichenvorganges wird mit dieser Farbe das Window gelöscht. Falls ↩ Flächen freibleiben, so
erscheinen sie eben in dieser Farbe.

Dithering
Es sind 3 Modi wählbar. Der erste, gar kein dithern, der zweite, in dem versucht ↩ wird, mittels
Dithering die doppelte Anzahl an Farben zu simulieren und der dritte, in dem ↩ versucht wird, durch Dithering die
4-fache Anzahl an Farben zu simulieren.

ExtBuffer
Falls ein Fraktal von 2D nach 3D transformiert wird, werden natürlich bloß alle
im Puffer verfügbaren Werte transformiert. Aber das resultiert ↩ höchstwahrscheinlich
in einem Bild, das "vorne" bzw. "hinten" abgeschnitten wirkt. 'ExtBuffer' erlaubt ↩ es
nun, die vertikale Puffergröße zu erhöhen. Die Angabe erfolgt in Prozent der ↩ Originalpuffergröße,
d.h. ExtBuffer=30 heißt, daß ein Puffer derselben Breite, aber der Höhe ' ↩ Normalhöhe + 30% des Normalen'
angelegt und auch berechnet wird. Für ein 2D-Fraktal ist all das absolut sinnlos, ↩ hier sollte man
dem Speicher zuliebe ExtBuffer auf 0 setzen. Für 3D-Transformationen ist ein Wert ↩ so zwischen
30 und 50 sinnvoll.

Sättigung
Helligkeit
Diese Werte haben nur beim Abspeichern des 3D-Bildes in 24 Bit eine Auswirkung. Im ↩ 3D-Puffer stehen die Originalfarben und
die Stärke des Lichtes, das an die Stelle fällt. Wie nun diese beiden ↩ Informationen gekoppelt werden, um zur 24-Bit-Farbe
zu gelangen, wird durch diese beiden Gadgets bestimmt.
Sättigung: Bestimmt, wie stark der Sättigungsgrad der Originalfarbe durch das ↩ Licht beeinflußt wird. Wertebereich reicht

von 0 (Originalsättigung, Licht spielt gar keine Rolle) bis 100 (Originalsättigung ↩
wird komplett durch die Lichtintensität
ersetzt)
Helligkeit: Siehe bei Sättigung, bloß wird nun die Helligkeit der Farbe evtl. ↩
durch das Licht beeinflußt.

Normalerweise setzt man Helligkeit auf 100, Sättigung auf 0, d.h. die ↩
Originalfarbe wird benutzt, ins FSH-Farbmodell umgerechnet,
die Helligkeit durch das Licht, das an exakt diese Stelle scheint, ersetzt, das ↩
ganze nach RGB-Format und dann
abgespeichert. Wer sich für die Wirkung des ganzen interessiert, kann ja eine 24- ↩
Bit-3D-Animation darüber berechnen
lassen ;-)

Riemann
(nicht implementiert!)
- Fraktale sind fast immer Repräsentationen dafür, was für verschiedene komplexe ↩
Zahlenwerte
passiert. Jeder Bildschirmpunkt entspricht einfach einem komplexen Zahlenwert.
Riemann nun hat sich eine andere Darstellung für die komplexen Zahlen einfallen ↩
lassen.
Normalerweise spricht man ja von der komlexen Zahlenebene, die eine Achse ist die ↩
reelle Achse, die
andere die Imaginäre Achse. Riemann hat diese Ebene nun auf eine Kugel ↩
transformiert.
Diese Kugel hat den Radius 1 und berührt mit ihrem Südpol (also grob gesprochen ↩
halt unten...)
die komplexe Zahlenebene im Ursprung. Wenn man nun einen Punkt auf der komplexen ↩
Zahlenebene
hat, dann zeichnet man einfach die Verbindungsgerade zwischen diesem Punkt und dem ↩
Nordpol der
Kugel. Da, wo diese Gerade die Kugel durchstößt, ist der Punkt auf der ↩
Kugeloberfläche zu
finden, der dem komplexen Zahlenwert entspricht.
Alles schön und gut, jedem festen komplexen Zahlenwert wird so genau ein Punkt
auf der Kugeloberfläche zugeordnet, bis auf eine Ausnahme: Dem unendlich fernen
Punkt. Genauer gesagt, gibt es keinen unendlich fernen Punkt, es gibt unendlich ↩
viele
von dieser Sorte. Bloß allen ist eines gemeinsam: Sie werden allesamt auf den
Nordpol der Kugel abgebildet. Dies ist vermutlich auch der Grund dafür, daß in
der Fraktaltheorie der unendlich ferne Punkt fast immer als möglicher Attraktor
in Erwägung gezogen wird. Man beachte den Singular (Punkt). Es sind unendlich ↩
viele
solche da, bloß auf der Riemannkugel sind sie halt alle beisammen am Nordpol.
Nun, mein Programm kann bei der 3D-Darstellung auch die Riemann-Kugel zeichnen.
Allerdings ist es nicht die Standardriemannkugel, denn die berührt die komplexe ↩
Zahlenebene
ja im Ursprung. Dieses Programm allerdings berechnet einen graphisch günstigen ↩
Berührungspunkt
selbst. Der Radius ist auch einstellbar, bei der Originalkugel wäre er ja fest 1.
Man kann sich wohl denken, wie es aussehen würde, wenn man in ein Fraktal ↩
hineinzoomt und
dann sich die Riemanndarstellung betrachtet. Für Leute mit weniger Fantasie: Viel ↩
zu wenig Werte sind
da, hat man also die ganze Kugel vor Augen, so ist darauf evtl. nur ein Punkt ↩
sichtbar, eben der,

der für den Ausschnitt des gerade berechneten Fraktals steht. Kommt man auf die ↩
Idee, sich die Kugel
doch näher heranzuholen, sprich hinzuzoomen, dann ist klar, daß die lokale ↩
Krümmung der Kugel
immer weiter abnimmt, und wenn man das Fraktal dnn vor Augen hat, wird es sich ↩
nicht merklich
von der 2D-Darstellung unterscheiden.
Den Radius einzustellen, kann man auch dem Programm überlassen. Einfach auf ' ↩
Radius setzen' klicken und das Programm berechnet ihn so,
daß das Fraktal schön auf einer Kugel Platz hat.

1.37 2.4 Menüs

2.4 Die Menüs
2.4.1 Systemmenü

Weitere Menüpunkte:
Fraktalmenü
Fraktalwindows
Windows
Extras

Daten laden/speichern
- Damit werden die Fraktaldaten aus einer Datei geladen/in einer Datei gespeichert
Wird die Datei im Verzeichnis ChaosPro/FractPic abgespeichert, so wird sie beim ↩
nächsten
Start des Programms automatisch geladen und im Gadget für die Fraktalbilder ↩
angezeigt.
Wird eine Datei nachträglich geladen, so wird sie nach dem Ladevorgang in das ↩
Listview
eingehängt.
Das Programm merkt beim Abspeichern, ob das 3D-Window offen ist und speichert in ↩
diesem
Fall die Fraktaldaten und die Information, daß es sich um ein 3D-Fraktal handelt. ↩
Wenn
zukünftig dann auf das Gadget 'Bild berechnen' geklickt wird, werden gleich beide ↩
Windows,
das 2D- und das 3D-Window geöffnet.

Bild speichern/ins Clipboard
- Damit kann das Bild als IFF-Bild abgespeichert werden. Bei den Fraktaltypen ↩
Juliamenge/ Mandelbrot/ Plasma/ Lyapunov-Raum ist es möglich, das 2D-Bild
in einer beliebigen Farbtiefe bis 256 Farben unabhängig von der Hardware bzw. vom ↩
Betriebs(bildschirm-)modus
abzuspeichern. In welcher Tiefe Sie es haben wollen, können Sie nach der Wahl des
Dateinamens bestimmen. Damit können dann Besitzer alter Amigas mit ECS-Chipsatz ↩
die Bilder in 256 Farben
speichern und sie von einem anderen Programm in den HAM6-Modus umrechnen lassen.
Außerdem wurde der Wunsch nach 24 Bit geäußerst. Auch das ist hier möglich. Dabei
stehen 2 Möglichkeiten zur Auswahl: 24 (Screen) und 24 (256)
Ersteres speichert das Fraktal so, wie es auf dem Screen angezeigt wird, aber eben ↩
mit
24 Bit. Das Programm berechnet eigentlich mehr Farben als es anzeigen kann und ↩
rechnet dann auf die Anzahl der

zur Verfügung stehenden Farben herunter. Falls nun auf dem Screen Flächen derselben Farbe vorkommen, ist es möglich, daß
das Programm gezwungen war, hier verschiedenen eng beieinander liegenden Werten dieselbe
Farbe zuzuweisen. Wenn das 24 Bit-Bild abgespeichert wird, wird die benutzte Farbpalette 'aufgeblasen'
und so die korrekte Farbe abgespeichert, was die Fläche zu einem (mehr oder weniger) sanften
Farbübergang werden läßt. Beachten Sie bitte, daß bei Verwendung der Iterationseinfärbung bei Julia/Mandel das
24Bit-Bild optisch identisch ist mit dem Bild, das Sie erhalten, wenn Sie es in 256 Farben speichern.
Wollen Sie hier wirklich das, was Sie wahrscheinlich wollen werden, nämlich viele, viele Farben, sanfte
Farbübergänge etc., dann müssen Sie die CPM-Einfärbung einstellen.

Die andere Wahlmöglichkeit, 24 (256), nimmt die komplette Farbpalette von 256 Farben, bläst sie auf und speichert so das Bild ab.
Das entspricht der Anwahl von 8 Planes, nur daß hier eben Flächen durch Farbübergänge ersetzt werden.

Damit keine Mißverständnisse aufkommen: Wenn ein Fraktal keine Flächen enthält, wird das 24 Bit-Fraktal nicht
(merkbar) anders aussehen, als eines mit z.B. 64 Farben. Also ein Plasma-Fraktal mit hoher Granulation als
24 Bit abzuspeichern ist sinnlos. Es muß schon Flächen aufweisen.

Bei den Bifurkationsdiagrammen und den dynamischen Systemen und auch bei den 3D-Ansichten der
Fraktale bestehen all diese Möglichkeit nicht. Das Fraktal wird einfach in der berechneten Tiefe,
also in der Screentiefe, abgespeichert. Ausnahme: Wenn das Fraktal einen 3D-Puffer bekommen
hat, dann kann es in 24 Bit Farbtiefe abgespeichert werden.

SystemInfo
- Dieser Punkt gibt Informationen über den Prozessor/ Koprozessor/ Grafikchips/ Speicher
aus

Über ChaosPro
- Informationen über den Autor, die Programmversion

Ende
- Erklärung nötig?

1.38 2.4 Menüs

2.4.2 Fraktalmenü

Weitere Menüpunkte:
Systemmenü
Fraktalwindows
Windows
Extras

```
Juliamenge
Mandelbrot
Bifurkation
Dynamisches System
Plasma
Lyapunov-Raum
- Damit werden neue Fraktale dieses Typs hinzugefügt. Sie werden mit den Standardwerten
für den jeweiligen Typ initialisiert.

Defaultwerte
- Falls mach sich mit den Parameterwerten vertan hat und gar kein sinnvolles Fraktal mehr erscheinen
will, so kann man diesen Punkt anwählen. Er setzt die Daten des Fraktals auf die im Programm gespeicherten
Defaultwerte.

Windowgröße ändern
- Dieser Menüpunkt zeigt die aktuelle Windowgröße und erlaubt es auch, sie zu ändern.
Die maximale Größe des Windows wird nicht nur durch den Screen bestimmt, sondern ist natürlich
die Screengröße minus dem Rahmen des Windows. Diese Funktion kann derzeit ein normales Window
nicht in ein Backdropwindow umwandeln bzw. umgekehrt. Derzeit finde ich, daß dieses Verhalten
eher ein Feature als ein Fehler ist, da es nun möglich ist, ein Backdropwindow zu machen (Menüpunkt
'Windowtyp ändern') und anschließend dieses Backdropwindow in der Größe zu ändern. (Natürlich mag es
etwas seltsam erscheinen, wenn ein Window ohne Rahmen irgendwo auf dem Screen existiert...)

Zoomen
- Skalieren hinein - Skalieren heraus
- Skalieren hinein entspricht einem Doppelklick in die Mitte des 2D-Fraktalwindows. 'Skalieren heraus' macht das
Gegenteil inklusive der Skalierung, etc.

- Rahmen hinein - Rahmen heraus
Nun kann man irgendwo in das 2D-Fraktalwindow klicken (Maustaste darf wieder losgelassen werden),
kann dann einen Rahmen ziehen und dann nochmal klicken. Wenn hineingezoomt werden soll, wird der
durch den Rahmen bestimmten Ausschnitt zur vollen Windowgröße vergrößert. Beim Hinauszoomen wird das
volle Window in den Rahmen hineinprojiziert.

Undo/Redo
- Unbeschränktes Undo/Redo für jedes Fraktal. Grundsätzlich wird ein Puffer von
10 KB pro Fraktal angelegt, in dem die alten Werte aufbewahrt werden. Mir hat das
bisher in jedem Fall gereicht...

Verschieben nach...
- verschiebt das Fraktal, selbes kann mit den Cursortasten  einfacher  erreicht
werden
```

```
Proportion
- falls das Fraktal stark verzerrt ist, stimmt das Verhältnis der Ausschnittwerte
überhaupt nicht mehr mit dem Verhältnis der Breite zur Höhe des Windows überein.
Mit diesem Punkt kann das Verhältnis wiederhergestellt werden.

Berechnung
- Stopp/Fortsetzen: Da das Programm im Multitasking läuft, kommt es des öfteren ↩
    vor, daß man mehrere
Fraktale berechnen läßt aber nun eben nur eines so schnell wie möglich fertig ↩
    haben will.
Mit diesem Punkt stoppt man die Berechnung des aktiven Fraktals. Der Task wird ↩
    dabei
schlafen gelegt und kann durch Anwahl des Punktes 'Fortsetzen' wieder aufgeweckt ↩
    werden.
- Neustart: Erzwingt eine Neuberechnung des Fraktals.

Picasso
Picasso schließen
- Sollten diese Punkt funktionieren, so würde er das 24 Bit-Bild direkt auf der ↩
    Picasso Grafikkarte
von Village Tronic anzeigen. (nicht implementiert)

EGS ansteuern
- Dieser Punkt öffnet ein Window auf dem EGS-Default-Screen und zeichnet darin das ↩
     Fraktal in 24 Bit, sofern dies
möglich ist. (nicht implementiert)

Zeige Fortschritt
- Es wurde bemängelt, daß man nicht genau sieht, wann das Fraktal fertig ist. Mit
diesem Punkt wird die Anzeige eingeschaltet, wie weit das Fraktal berechnet ist.
```

1.39 2.4 Menüs

```
2.4.3 Fraktalwindows

Weitere Menüpunkte:
Systemmenü
Fraktalmenü
Windows
Extras

Datenwindow
- Einige Fraktaltypen haben Datenwindows zur Verfügung. Wenn man dieses Window
öffnet und mit der Maus über das Fraktal hinwegfährt, so werden hier die Daten
entsprechend der Mauszeigerposition angezeigt.

Parameter 1...
Parameter 2...
Parameter 3...
Formeleditor
- Hiermit wird das Parameterwindow 1/2/3 bzw. das Formeleditorwindow geöffnet/ ↩
    geschlossen. Welche Parameter
einstellbar sind und was sie bedeuten, kann in den entsprechenden Kapiteln 2.3.2 ↩
    bis 2.3.6
```

```
nachgelesen werden

3D-Parameter 1
3D-Parameter 2
3D-Parameter 3
- Die 3D-Parameterwindows werden hiermit geöffnet/geschlossen. Näheres ist in Kapitel 2.3.7
nachzulesen.

Windowtyp als Backdrop/normales Window
- Evtl. will man den gesamten Platz auf dem Screen für ein Fraktal ausnutzen. Wenn
ein Windowrahmen vorhanden ist, so ist das aber nicht möglich. Deshalb kann man ein
Window zum Backdrop-Window erklären. Dabei wird automatisch das Window geschlossen,
der Rahmen, die Titelleiste, die Systemgadgets entfernt, das Window auf volle Größe gebracht
und erneut als Backdrop geöffnet. Das funktioniert mit dem 2D- bzw. 3D-Window.
Aber Achtung: Da man nun plötzlich ohne Depth-gadget da steht, kann man die Reihenfolge
der Windows nicht mehr ändern. Also über Sinn und Unsinn von mehreren Backdrop-Windows
läßt sich streiten...

Zeige Position
- Juliamenge, Mandelbrotmenge, Bifurkation, Lyapunov-Raum haben einen 2D-Ausschnitt. Es kann
hineingezoomt werden. Und plötzlich weiß man nicht mehr ganz genau, wo man denn nun hineingezoomt
ist. Die Parameter in den Parameterwindows geben zwar Auskunft, aber ganz klar ist
es trotzdem nicht. Dieser Menüpunkt öffnet für dieses Fraktal ein Window. In diesem
Window werden alle Fraktale desselben Typs angezeigt. Wählt man einen Eintrag an, so
wird im Fraktalwindow der Bereich, in dem das angewählte Fraktal gezeichnet wurde, als Rahmen
gezeichnet.

Juliaparameter setzen
- Dieser Menüpunkt ist nur bei Mandelbrotmengen anwählbar. Er öffnet ein Window, in dem
sämtliche Juliamengen angezeigt werden. Wählt man eine Juliamenge an, so wird in der Mandelbrotmenge
der Parameterwert der Juliamenge als Kreuz angezeigt. Dieses Kreuz kann man natürlich verschieben
und so den Parameterwert der Juliamenge ändern. Es wurde bereits erwähnt, daß interessante Juliamengen
am Rande der Mandelbrotmenge zu finden sind. Doch wo ist der Rand der Mandelbrotmenge, wenn man nur
eine komplexe Zahl vor sich hat? Mit Hilfe des Kreuzes weiß man es ganz genau.
Aber Achtung: Julia- und Mandelbrotmenge sollten schon zusammenpassen, also sollte die
Juliamenge z.B. vom Subtyp z^2+c sein, dann sollte auch die Mandelbrotmenge vom Subtyp z^2+c sein.
Ansonsten ist dieses Gerede von interessanten Juliamengen am Rande der Mandelbrotmenge ziemlicher
```

Schwachfug...

1.40 2.4 Menüs

2.4.4 Windows

Weitere Menüpunkte:
Systemmenü
Fraktalmenü
Fraktalwindows
Extras

Farbpalettenwindow
Palettenbearbeitung
Animation 1&2
CycleControl
Benutzerwindow

Hier kann ich wieder auf das Kapitel 2.2 verweisen. Dort ist alles Wissenswerte nachzulesen.
Vielleicht noch eines: Über das Menü können nur die ersten 4 benutzerdefinierten Windows geöffnet werden. Wer mehr hat, muß diese dann über den Arexx-Port öffnen.

1.41 2.4 Menüs

2.4.5 Extras

Weitere Menüpunkte:
Systemmenü
Fraktalmenü
Fraktalwindows
Windows

Hilfe
- zeigt das Inhaltsverzeichnis der Online-Hilfe an. Wer schneller zum Ziel gelangen
will, benutzt:

1. Menuhelp
Ihr wählt einen Menüpunkt an, lasst aber die rechte Maustaste nicht los, also eigentlich nicht anwählen, sondern den Menüpunkt
'unterlegen'. Dann drückt ihr die Help-Taste. Das Betriebssystem meldet nun meinem
Programm, daß der Benutzer Hilfe zu einem Menüpunkt will und sagt mir auch, zu welchem.
Das Programm kann nun geeignet reagieren.

2. Selbstgestricktes Gadgethelp
Mein Programm hält umfangreiche Datenlisten aufrecht, in denen die Positionen und Ausmaße
der Gadgets festgehalten werden. Wird die Help-Taste gedrückt, so sucht mein Programm
nach dem Gadget, über dem der Mauszeiger steht und handelt entsprechend. Ist der

```
Mauszeiger über keinem Gadget, so zeigt er den Hilfe-Text an, der als Default für
dieses Window vorgesehen ist.

globaler Stop
globales Fortsetzen
- Stoppt die Berechnung aller Fraktale auf einen Schlag, falls ein anderes ←
   Programm
sämtliche Rechenzeit braucht...
'Fortsetzen' weckt alle Fraktalberechnungstasks auf, falls und nur falls sie ←
   schlafen.

Colorcycling
- An
Damit kann man Colorcycling an-/ausschalten
- Aufwärts
Hiermit kann eingestellt werden, ob aufwärts (zu höheren Farbnummern hin) oder ←
   abwärts
gecycled werden soll.
- Schneller/Langsamer
Geschwindigkeit des Colorcyclings. Zum Colorcyclen wird ein separater Task ←
   geschaffen.
Im Interrupt darf ja laut RKM: Libraries nichts mit den Farbtabellen angestellt
werden.
Da nun das Taskswitching nicht beliebig oft auftritt, sondern nur ca 50 mal in
der Sekunde, so ist dadurch natürlich die Geschwindigkeit des Colorcyclings ←
   begrenzt.
Wer mit 256 Farben arbeitet, muß natürlich auch bedenken, daß es fürs ←
   Betriessystem
ein ganz schöner Happen ist, 256 Farbwerte neu einzustellen, die Copperlisten ←
   hierfür neu zu berechnen, zusammenzulinken mit
den anderen und alles anzuzeigen...

Taskpriorität
- Damit wird die Taskpriorität des Haupttasks und des Colorcycling-Tasks ←
   eingestellt.
Sämtliche Berechnugstasks laufen mit einer um 1 niedrigeren Priorität. Default- ←
   Wert
ist 0, Fraktalberechnung also bei -1, was einen ungestört auf der Workbench oder
mit irgendeinem anderen Programm weiterarbeiten läßt, während ein Fraktal ←
   berechnet
wird.

Window auslagern
- auf Fraktalscreen / auf Parameterscreen / auf Workbench / auf Publicscreen
Damit kann jedes Window bis auf das 2D- und 3D-Fraktalwindow auf den betreffenden
anderen Screen ausgelagert werden. Sehr sinnvoll, da der Platz auf einem noch so ←
   großen
Screen recht schnell knapp wird. Auch für den Speicher gut, denn ein Parameter-
Window auf einem 256 Farb-Fraktalscreen schluckt auch entsprechend mehr Speicher
als wenn es auf einer 4-Farb-Workbench liegt. Durch das Preferences-Programm kann ←
   eingestellt werden, auf welchem
Screen die Windows defaultmäßig aufmachen.

Screenmodes einstellen
Font einstellen
- Ist hier eine umfangreiche Erklärung nötig?
Falls sie angewählt werden, werden alle möglichen Windows geschlossen, die Werte
```

geändert und dann alles erneut wieder geöffnet.
Die minimalen Screentiefen sind:
1. Für den Parameterscreen: 1
2. Für den Fraktalscreen: 3
3. Für den Colorscreen: 4
- Für den Font gibt es keine Einschränkung. Aber: Falls ein Window nicht mehr auf dem Screen Platz hat, ist halt ein kleinerer Font zu wählen. Das betrifft vor allem Leute, die das Programm in einer Auflösung von 640x200 bzw. 640x256 betreiben. Topaz 8 ist da fast schon zu groß.

Einstellungen laden/speichern
- Hiermit können verschiedene Einstellungen geladen bzw. gespeichert werden. Vielleicht
liefere ich in Zukunft ja auch mehrere Dateien für verschiedene Gadgetpositionen/ Anordungen mit.
Sämtliche im Programm verwendeten Positionen/Größen sind nämlich in dieser Datei gespeichert.

1.42 2.4 Menüs

2.4.6 Benutzerdefinietes Menü

Hierfür ist wieder eine ASCII-Datei im Verzeichnis ChaosPro/Prefs/ mit Namen
Menu.asc erforderlich. Diese muß dann mit dem Preferences-Programm übersetzt werden.
Dabei wird die Datei Menu.prefs generiert.

Der Aufbau der Datei ist:

```
MENU <Menütext> <Tastaturshortcut> <Arexx-Script>
ITEM <Itemtext> <Tastaturshortcut> <Arexx-Script>
...
ITEM <Itemtext> <Tastaturshortcut> <Arexx-Script>
MENU <Menütext> <Tastaturshortcut> <Arexx-Script>
...
END <Menütext> <Tastaturshortcut> <Arexx-Script>
```

Dabei muß bei MENU und END natürlich nichts Sinnvolles bei Tastaturshortcut bzw. Arexx-Script
angegeben werden.
für Menütext kann auch die Konstante BARLABEL eingetragen werden. Sie erzeugt einen
Trennstrich.
Für Tastaturshortcut kann auch NONE angegeben werden, falls eben keiner gewünscht wird.

Beispielsweise sieht es so aus:

```
MENU MenüNr1 NONE dummy.rexx
ITEM Data B Daten.rexx
ITEM BARLABEL NONE dummy.rexx
ITEM Nochwas C Nochwas.rexx
MENU MenüNr2 NONE dummy.rexx
ITEM InOut D InOut.rexx
```

```
END BARLABEL NONE dummy.rexx
```

1.43 2.5 Programmverzeichnisse

```
2.5 Programmverzeichnisse

Basisverzeichnis, vom dem aus das Programm die verschiedenen Unter\verzeichnisse
anspricht, ist stets das logische Verzeichnis 'ChaosPro:'. Ist dieses beim ←
   Programmstart
vorhanden, ist alles ok. Ist es nicht vorhanden, so versucht das Programm ←
   herauszufinden,
von welchem Verzeichnis aus es gestartet wurde. Anschließend erzeugt es den Assign ←
    mit
Namen ChaosPro: auf das so gefundene Verzeichnis.

ChaosPro:libs/
hier befinden sich alle Bibliotheken, die das Programm benötigt. Es ist nicht
nötig, sie nach libs: zu kopieren, ganz davon abgesehen, daß sie m.E. dort absolut ←
    nichts
zu suchen haben. Es sind Bibliotheken für ChaosPro, also sollen sie auch da sein, ←
   wo sich
das Hauptprogramm befindet. Daß sich das ganze Programm dadurch so ganz nebenbei ←
   sehr
einfach wieder deinstallieren läßt, ist volle Absicht...

ChaosPro:Guides/
Hier befindet sich die Dokumentation

ChaosPro:Prefs/
Alle Einstellungen des Programms sind dort zu finden

ChaosPro:Palette/
ChaosPro braucht mindestens eine Farbpalette in diesem Verzeichnis, sonst bricht ←
   es mit
einer Fehlermeldung ab. Zum Programmstart liest es selbstständig alle Dateien aus
diesem Verzeichnis ein, filtert eventuell vorhandene Farbinformationen heraus und
verwirft den Rest. Es ist daher auch möglich, hier ganze Bilder zu lagern. In
diesem Fall werden eben die Farbtabellen der Bilder eingeladen.

ChaosPro:Catalogs/
Hier finden sich alle Übersetzungskataloge

ChaosPro:FractPic/
Beim Start des Programms wird dieses Verzeichnis auf mit dem Programm ←
   abgespeicherte Daten von Fraktalen durchsucht.
Alle gefundenen Fraktale werden geladen. Benötigt ein Fraktal eine nicht im ←
   Programm eingebaute Formel, so lädt es sie
ebenso ein (sie wurde in der gleichen Datei in einem anderen Chunk gespeichert ) ←
   und fügt sie der
internen Formelliste hinzu.

ChaosPro:Anims/
ChaosPro:AnimData/
In diesen Verzeichnissen versucht ChaosPro zuallererst, Animationen bzw. ←
   Animationsdaten
```

abzuspeichern.

ChaosPro:Formula/
Dieses Direktory wird beim Programmstart gescannt. Alle dort gefundenen Formeln, ↩
die mit dem Programm
abgespeichert wurden, werden geladen und können verwendet werden.

1.44 2.6 Preferencesprogramm

2.6 Preferencesprogramm

Zur Einstellung von Parametern existiert ein externes Preferences\programm.
Folgende Optionen bieten es an:

Fractalscreen
Durch Druck auf dieses Gadget öffnet sich der Screenmode-Requester der reqtools. ↩
library
und erlaubt es, den Screen, auf dem die berechneten Fraktale erscheinen, zu wählen ↩
.
Dieser Screen muß eine Tiefe von mindestens 3 Planes haben, da das Hauptprogramm ↩
die
Farben 0 bis 3 nicht zum Zeichnen verwendet.

Parameterscreen
Das Hauptprogramm öffnet in jedem Fall den Fraktalscreen. Doch innerhalb kurzer ↩
Zeit merkt man,
daß der Screen recht voll wird. Außerdem benötigen die vielen Windows auch viel ↩
Speicher, da naturgemäß
der Fraktalscreen mit mehreren Planes betrieben wird.
Aus diesem Grund besteht die Möglichkeit, vom Hauptprogramm auch noch einen 2. ↩
Screen öffnen zu lassen,
nämlich den Parameterscreen, auf dem dann einige Fenster ausgelagert werden können ↩
. Dieser Screen darf auf
eine Planetiefe von 2 oder sogar 1 haben, was die (Intuition-)Arbeits\geschwin\ ↩
digkeit
erhöht.

Colorscreen
Wenn der Fractalscreen mit einer Tiefe von 8 Planes (256 Farben) eingestellt ist, ↩
dann
werden die Paletten\bearbeitungs\windows stets auf dem Fractalscreen geöffnet.
Andernfalls ist es nicht möglich, sie dort zu öffnen, da sie evtl. die Farben ↩
verschieben.
Deshalb wird hierfür ein neuer Screen, der Colorscreen, geöffnet. Er sollte so ↩
viele Farben
wie nur irgend möglich bieten.

Font
Damit stellt man den Font ein, der im Programm verwendet wird. Das gesamte ↩
Programm ist
fontsensitiv, d.h. die Gadgetpositionen und Windowgrößen sind nicht wie üblich in ↩
Pixel\einheiten,
sondern in Zeichen\einheiten angegeben. Ein kleinerer Font hat also kleinere ↩
Windows und kleinere

```
Gadgets zur Folge. Da das Hauptprogramm sehr viele Parameter anbietet und sehr ←
   viele Windows öffnet, sollte man
den Font so klein wie möglich halten, um so mehr Platz zu schaffen.

PubScreenName
Das Hauptprogramm kann 4 Bildschirme handhaben: den Fractalscreen, den ←
   Parameterscreen, die Workbench und einen Publicscreen.
Auf jeden dieser Screens können Windows ausgelagert werden, um nicht ständig auf ←
   einem Screen alle Windows zu haben. Hier kann
man den Namen des PublicScreens eingeben, um später dann dort ein Window ←
   auszulagern. Ist der Screen zur Laufzeit nicht vorhanden, so
öffnet sich das Window auf der Workbench.

Compile Userwindows
Benutzerdefinierte Windows werden vorerst in einer ASCII-Datei definiert.
Dannach muß diese noch in ein Format konvertiert werden, das möglichst leicht vom ←
   Hauptprogramm verstanden wird. Dazu dient dieses
Gadget. Einfach drauf klicken und mit Hilfe der ASCII-Datei ChaosPro/Prefs/Windows ←
   .asc wird die Datei
ChaosPro/Prefs/Windows.prefs geschaffen.

Compile Usermenus
Benutzerdefiniete Menüs werden ebenfalls vorerst in einer ASCII-Datei festgelegt, ←
   die
anschließend noch für das Hauptprogramm konvertiert werden muß, was dieses Gadget ←
   erledigt. Es erzeugt aus der Datei
ChaosPro/Prefs/Menu.asc die Datei ChaosPro/Prefs/Menu.prefs.

Die Online-Hilfe
Da das Programm auf beliebigen Screens laufen kann, war es nötig, auch die Online- ←
   Hilfe
entsprechend anzupassen. Denn was nützt es, wenn die Online-Hilfe, die ←
   typischerweise für
einen Screen mit einer Breite von 640 Pixeln formatiert ist, auf einem Screen mit ←
   1024 Pixeln
Breite nur knapp über die Hälfte des Platzes ausnutzt? Was passiert, wenn der ←
   Benutzer sich entschließt, die
Online-Hilfe in einer anderen Schriftart anzuzeigen und deshalb die @FONT- ←
   Direktive in der Guide-Datei ändert?
Ganz klar, es sieht schrecklich aus, da plötzlich jede Zeile eine andere Breite ←
   hat, zumindest wenn man einen
Proportional-Font verwendet.
Aus diesem Grund wird auch die Guide-Datei mit Hilfe des Preferences-Programms ←
   compiliert,
um eine optimale Ausnutzung des Platzes zu erreichen und das Aussehen durch ←
   Verwendung von Blocksatz
(jede Zeile hat dieselbe Breite) ansprechender zu gestalten.
- GuideWidth
Hiermit legt man fest, in welcher Breite der Guide formatiert werden soll. Da der ←
   Fensterrahmen
auch noch Platz beansprucht, sind für gewöhnlich von der Screen-Breite 40 Pixel
abzuziehen.
- Language
Damit legt man fest, in welcher Sprache die Datei geschaffen werden soll. Dies ist ←
    unabhängig davon, ob
das Locale-System installiert ist, denn es existieren (hoffentlich) 2 Guide- ←
   Dateien in ChaosPro/Guides, nämlich deutsch.guide
```

und englisch.guide, die eben entsprechend herangezogen werden.
- Build Guide
Damit wird im Verzeichnis ChaosPro/Guides/ die Guide-Datei ChaosPro.guide aus der
Datei deutsch.guide bzw. englisch.guide geschaffen. Die Originaldateien bleiben
natürlich erhalten. ChaosPro.guide ist eine absolut normale Guide-Datei, die man
sich mit Multiview, Hyper oder AmigaGuide anzeigen lassen kann, die aber eben auf das
richtige Format und Blocksatz gebracht worden ist. Diese Operation kann recht lange
dauern (auf meinem A4000/040 ca 60 Sekunden!), da 1. die Guide-Datei recht groß ist
und 2. ich mir keine Mühe gegeben habe, die Routine schnell zu machen, da ich 3.
denke daß man diese Operation nicht besonders oft ausführen läßt.
Abhängig vom Font kann es vorkommen, daß plötzlich ein ominöser Requester aufgeht mit der Fehlermeldung
"Failed to create a line". Hier kann man nur 'Ok' klicken, und genau das ist auch zu tun. Mit dieser
Meldung hat es folgendes auf sich: Bei Erstellen des Blocksatzes kann es vorkommen, daß ein Wort länger als
eine gesamte Zeile ist und auch nicht getrennt werden kann. Zumindest weiß der Computer nicht, wo er es
trennen soll. In diesem Fall wird diese Meldung gebracht. Doch denke ich, daß diese Meldung in der
Regel mit einem Programmfehler zusammenhängt. Bei einer daraufhin gestarteten Bugsuche habe ich allerdings
keinen Fehler festgestellt, dort waren stets Wörter (es waren irgendwelche komplexe mathematischen Ausdrücke)
wirklich zu lang. Das heißt für die Benutzer: Warten, bis ich da voll durchblicke und evtl. eine genauere Fehlerbeschreibung
(vielleicht sogar mit der Nummer der Zeile, in der der Fehler aufgetreten ist...)
einbaue.

Um den Font der Guide-Datei oder sogar ganze Textpassagen zu ändern, kann man natürlich einfach
die Datei ChaosPro.guide ändern. Doch das ist so ziemlich das Dümmste, was man machen kann.
Denn dann steht man ohne Blocksatz da und außerdem sind die Änderungen beim nächsten Anklicken von
'Build Guide' wieder verloren...
Will man also etwas ändern, so ist dazu deutsch.guide bzw. englisch.guide zu ändern.
Dies sind ebenfalls normale Guide-Dateien für AmigaGuide bzw. MultiView oder auch Hyper, aber eben noch nicht im richtigen
Format. Dort kann man in Zeile 6 die @FONT-Direktive ändern und einen anderen Fontnamen
sowie eine andere Fontgröße einstellen.
Will man Testpassagen ändern, so ist darauf zu achten, daß ein Absatz durch eine
Zeile mit weniger als 76 Zeichen beendet wird. Zeilen mit 76 Zeichen oder mehr
werden mit der nächsten Zeile, diese evtl. mit der übernächsten etc. zu einem Absatz
verbunden. Falls also nach der Konvertierung plötzlich ein neuer 'Absatz' mitten im Absatz
begonnen wird, hat die letzte Zeile weniger als 76 Zeichen...
Falls ein Wort zu lang ist, sieht es evtl. im konvertierten Guide nicht so toll aus,
da um des Blocksatzes willen zu viele Spaces eingefügt werden mußten. Deshalb besteht

```
die Möglichkeit, der Konvertierungsroutine mitzuteilen, wo ein zu langes Wort ←
    getrennt
werden kann. Dies geschieht mit dem Backslash-Zeichen links neben der Backspace- ←
    Taste
rechts oben auf der Tastatur. Also einfach an den Trennstellen dieses Zeichen ←
    einfügen.
Wird das Wort getrennt, so wird ein '-'-Zeichen eingefügt und getrennt, ansonsten ←
    wird
der Backslash einfach entfernt.


Cycle-Gadgets in der rechten Hälfte des Windows
Wie bereits angemerkt, können Windows auf andere Screens ausgelagert werden. Mit
Hilfe dieser Gadgets wird eingestellt, wo das jeweilige Window defaultmäßig ←
    geöffnet
wird. Von dort aus kann es immer noch ausgelagert werden, bloß das AmigaGuide- ←
    Window
nicht. Wo dieses öffnet und dann stets verbleibt, wird hier eingestellt. Es ist ←
    ratsam, dieses Window
auf einem Screen mit möglichst wenig Farben öffnen zu lassen, da so die Text\ ←
    scroll\geschwin\digkeit
stark erhöht wird.

Save und Cancel
Mittels 'Save' werden die Daten abgespeichert, mittels 'Cancel' verworfen. ←
    Aktionen
wie 'Compile Userwindows', 'Compile Usermenus' und 'Build Guide' sind nicht ←
    betroffen.
Einmal ausgeführt, können sie nicht mehr rückgängig gemacht werden.
```

1.45 2.7 Problemecke

```
2.7 Problemecke

1. Problem
Gelegentlich hängt das gesamte System, wenn ich die Online-Hilfe aufrufe.
Lösung:
Keine. Ich habe keine Ahnung, woran das liegen kann. Doch da ich weiß, daß der
AmigaGuide wirklich noch nicht ausgereift ist (man denke daran, daß Commodore
beinahe zeitgleich die lokalisierte Workbench herausbrachte und deshalb das SPEAK ←
    :-Device
nicht mehr mitlieferte, aber nicht daran dachte, daß dieselbe Online-Hilfe in ←
    einer anderen
Sprache andere Zeilennummerierungen hat, was die Lokalisierung einer Online-Hilfe ←
    stark erschwert!), denke ich, daß der Fehler
mehr im AmigaGuide steckt.

2. Problem
Wie kann ich die Größe des AmigaGuide-Windows dauerhaft einstellen?
Lösung:
Durch Anwahl des Menüpunktes 'Vorgaben speichern'...

3. Problem
Bei Druck auf die Help-Taste kommt kein AmigaGuide-Window, die Online-Hilfe ←
    funktioniert also
```

```
nicht.
Lösung:
1. Evtl. ist das AmigaGuide-System nicht richtig installiert, dann sollten Sie ↩
   sich die offizielle Distribution
des AmigaGuide-Pakets (verfügbar bei PD-Händlern oder auf dem Aminet) besorgen und ↩
    es mittels des
Installerscripts installieren.

2. Evtl. ist ChaosPro.guide oder ChaosPro.Topics nicht vorhanden. In diesem Fall ↩
   sollten
Sie das Preferences-Programm starten und auf 'Build Guide' klicken. Mehr ↩
   Informationen hierfür finden sich im Kapitel über das
Preferences-Programm
3. Es existiert ein ToolType, mit dem man die Online-Hilfe abschalten kann, was ↩
   Speicherplatz
spart. Es heißt sinnigerweise NO_AGUIDE und, falls angegeben, wird die Online- ↩
   Hilfe nicht
initialisiert.
```

1.46 2.8 Sonstiges Erwähnenswertes

```
2.8 Sonstiges Erwähnenswertes

Wie gibt man Zahlen ein? Ganz klar, mit Integergadgets. Aber was ist mit ↩
   Fließkommazahlen?
So etwas gibt es bisher noch nicht im Betriebssystem. Deshalb mußte ich mir meine
eigene Hook-Routine schreiben, um aus einem String-Gadget ein Float-Gadget zu ↩
   machen.
In diesem Float-Gadgets werden alle sinnlosen Tastendrücke herausgefiltert. ↩
   Gewisse andere
Tastaturkombinationen wie z.B. RAmiga+X, um das Eingabefeld zu löschen, bewirken ↩
   entsprechend
Sinnvolles für ein Floatgadget, nämlich den Eintrag '+0.0'. Um das Vorzeichen ↩
   einer Zahl zu ändern,
ist an irgendeiner Stelle das '+'- oder '-'-Zeichen zu drücken. Das Vorzeichen an ↩
   allererster Stelle
ändert sich sogleich. Um den Dezimalpunkt an eine andere Stelle zu setzen, drückt ↩
   man an der betreffenden
Position einfach die '.'-Taste. Dadurch wird ein evtl. vorhandener Dezimalpunkt ↩
   gelöscht und an die neue
Stelle gesetzt. Zahlen in Exponentialschreibweise sind in der vorliegenden Version
nicht möglich.

Wer das Programm bereits benutzt hat, dem wird aufgefallen sein, daß die im ↩
   PicTask-Window
angewählten Einträge sich ständig ändern, wenn ein anderes Window aktiviert wird. ↩
   Das ist
Absicht. Das aktive Window bestimmt den aktiven Task. Immer wenn ein neues Window ↩
   aktiv wird,
sucht das Programm nach dem Fraktaltask, zu dem das Window gehört und erklärt ihn
zum aktiven Task. Außerdem geht es noch das gesamte Menüsystem durch und verbietet ↩
    die
Anwahl von unter Umständen für den jeweiligen Fraktaltyp nicht ausführbaren ↩
   Menüpunkten, wie
z.B. das Zeigen des Juliaparameters für Bifurkationsdiagramme.
```

Wenn man nicht mehr weiß, welcher Fraktaltyp nun überhaupt vorliegt, bzw. zu welchem
Task ein Window gehört, dann sollte man das Window aktivieren und sich den Screentitel
ansehen. Dort steht noch einmal kurz, welchen Namen das Fraktal hat und welcher Typ es ist.
Gut z.B. für Julia- und Mandelbrotmengen, wenn man hineingezoomt hat. Dann fällt es recht
schwer, die beiden Typen auseinanderzuhalten.

Ich habe mich bemüht, das Programm style-guide konform zu programmieren, da ich ein
Herz für Grafik\karten\besitzer habe. Aus diesem Grund ist mein Programm auch nicht
unbedingt das schnellste. Vor allem die Benutzer von Mand2000 werden die Geschwindigkeit
des Hineinzoomens als nicht besonders hoch ansehen.

Manch einer will, daß beim Programmstart sofort das Palettenwindow aufgeht oder irgendetwas anderes
gleich mal passiert. Diese Möglichkeit bietet das Programm über den AREXX-Port.
Beim Start wird das Arexx-Programm ChaosPro/Rexx/ChaosProInit ausgeführt. Dort
könnt ihr gleich mal ein Bild berechnen lassen.

Tja, so gut wie alle Programme wollen einen logischen Assign zum Arbeiten. Mein Programm macht
hier keine Ausnahme. Doch habe ich einen anderen Weg als (fast) alle anderen Programme beschritten:
Das Programm sucht beim Programmstart nach dem logischen Assign 'ChaosPro:'. Ist er vorhanden,
dann sucht er ausgehend von dort in den verschiedenen Unterverzeichnissen z.B. ChaosPro:Prefs, ChaosPro:Palette, ChaosPro/Formula etc.
nach den jeweiligen Dateien. Ist dieser Assign aber nicht vorhanden, dann wird er vom Programm kreiert und bei Programmende wieder entfernt.
Das heißt nun, daß z.B. für die Rexx-Scripts als Pfadangabe stets z.B. ChaosPro:Rexx/Default.rexx verwendet werden kann, da zur Laufzeit
immer ein Assign mit Namen 'ChaosPro:' vorhaden ist.

1.47 2.9 Tooltypes

2.9 Tooltypes

Das Programm unterstützt derzeit die folgenden ToolTypes:

NOJOYSTICK
Dieses Tooltype verbietet die Benutzung eines Joysticks in Port zum Zoomen und Herumfahren
im Fraktal. Dies war nötig, da evtl. ein Dongle (z.B. der von Real3D) in diesem Port stecken kann,
was einige Verwirrung stiften kann. Falls das Fraktal, sobald das Window aktiviert wird,
unkontrolliert herumfährt, so ist evtl. so ein Dongle schuld und man sollte dieses ToolType
angeben.

CHUNKYMODE
Dieses ToolType bestimmt die Routine, die zum Skalieren des Fraktals bei einem Doppelklick benutzt wird.
Normalerweise wird dies auf die folgende Art erledigt: Der Windowinhalt wird mittels ReadPixelArray8 in einen
Puffer ausgelesen, dieser dann skaliert, das Ergebnis mit einer eigenen Routine ins Planeformat umgewandelt und
anschließend mittels ClipBlit ins Window kopiert. Wenn nun jemand eine Grafikkarte besitzt und der
Screen, der für das Fraktalprogramm benutzt wird, eigentlich ein ChunkyScreen ist, so wird natürlich in dem
Augenblick, in dem mein Programm ClipBlit aufruft, alles wieder ins Chunkyformat umgewandelt...
Wenn man dieses ToolType angibt, dann benutzt mein Programm nach dem Skalieren einfach WritePixelArray8 und überläßt
es damit dem Grafikkartentreiber, die Werte, die evtl. schon im richtigen Format, eben dem Chunkyformat, vorliegen,
in den Screen zu kopieren.
Bemerkung: Das Programm schreibt nirgendwo direkt in die Planes von Windows oder gar dem Screen.
Wenn alle Programme so geschrieben wären, dann gäbe es heute Grafikkartentreiber, die um einiges
schneller wären.

COLORWHEEL
Dieses ToolType bestimmt, ob beim Paletteneditieren das Farbrad angezeigt werden soll.
Um die korrekte Darstellung des Farbrads zu ermöglichen, sind natürlich viele Farben dafür nötig, genauer gesagt,
werden für die Darstellung die Hälfte der Farben auf dem Screen benutzt. Aus diesem
Grund kann man mit diesem ToolType explizit angeben, ob man das Farbrad auch wirklich will.
Wenn Sie alle Farben auf dem Screen für das Anzeigen der Farben der Palette benutzen wollen, dann
geben Sie dieses ToolType nicht an.

BUILTIN
Falls angegeben, wird die eingebaute Sprache, also englisch, benutzt. Ansonsten natürlich die
Sprache, die im Locale-System voreingestellt ist. Dieses ToolType wird eigentlich nur von
mir zum Testen benötigt.

BACKFILL
Falls gesetzt, dann wird das Window vor dem Hinzufügen durch ein Raster ausgefüllt. Dadurch
hebt es sich besser vom Screen ab und sieht evtl. schöner aus.

PGA_NEWLOOK
Falls gesetzt, werden die Proportional-Gadgets im Aussehen verschönert. GadTools unterstützt dies
jedoch an sich nicht. Das Bit wird also von Hand gesetzt. Es funktioniert, ist jedoch ein
undokumentiertes Feature, das nicht zwingend funkionieren muß, bzw. sogar zu Fehlverhalten führen

kann. Bisher wurden jedoch keinerlei Nebenwirkungen bekannt.

NO_EGS
Falls nicht angegeben, kann das EGS-System, sofern korrekt installiert, durch ↩ einen
Menüpunkt angesprochen werden. Dieser Menüpunkt zeichnet dann das aktuelle Fraktal ↩ auf dem
EGS-Default-Screen.

NO_AGUIDE
Falls nicht angegeben, wird die amigaguide.library geöffnet. Somit ist eine ↩ kontextsensitive
Online-Hilfe verfügbar. Evtl. empfiehlt es sich, aus Speichergründen darauf zu
verzichten.

NO_REXX
Falls nicht angegeben, wird der Arexx-Port des Programms initialisiert und steht ↩ zur Benutzung
bereit. Wer diesen nicht will und den Speicher für etwas anderes besser nutzen ↩ kann, der
gebe dieses ToolType an.

1.48 2.10 Rechtliches

2.10 Rechtliches

Im Laufe der Nutzung und Programmierung dieses Programms passierte es 2 mal, daß
meine Festplatte durch einen Programmfehler einen Defekt aufwies. Ich bin nicht ↩ perfekt.
Deshalb sind mit einer Wahrscheinlichkeit von 100% Fehler in diesem Programm, die
unter Umständen schlimme Folgen haben können.

Deshalb:

Keine Garantie
Es besteht keine Garantie für dieses Program zusätzlich zur augenblicklich ↩ geltenden
Rechtssprechung. Der Copyright-Inhaber und/oder Drittanbieter stellen dieses ↩ Programm "so
wie es ist" zur Verfügung, ohne legliche Garantie, sei es eine Garantie ↩ ausdrücklicher
oder impliziter Art, einschließlich, aber nicht darauf beschränkt, den impliziten ↩ Gepflogenheiten
des Handels und der Tauglichkeit für einen bestimmten Zweck. Das gesamte Risiko, ↩ das
durch die Benutzung des Programms entsteht, tragen allein Sie. Sollte das Programm
fehlerhaft sein, so tragen Sie alle Kosten, die Ihnen durch den/die Fehler ↩ unmittelbar oder
mittelbar entstehen.

Außer die augenblickliche Rechtsprechung fordert es, kann in keinem Fall ein ↩ Copyright-Inhaber
oder irgendein Drittanbieter, der dieses Programm wie oben erlaubt weitervertreibt ↩ , haftbar

gemacht werden für irgendwelche Schäden, seien es allgemeine, spezielle, zufällige ↩
,
direkte oder indirekte Schäden, die durch die Benutzung oder die Fehlbenutzung des
Programms entstehen (eingeschlossen, aber nicht darauf beschränkt, der Verlust von ↩
Daten,
die fehlerhafte Berechnung von Daten, Verluste ihrerseits oder von Drittanbietern, ↩
oder
die Unfähigkeit dieses Programms, mit einem anderen zusammenzuarbeiten), sogar, ↩
wenn
der Copyright-Inhaber oder irgendein Drittanbieter von der Möglichkeit eines ↩
solchen Schadens
unterrichtet wurde.

ChaosPro 1995 Martin Pfingstl

ChaosPro ist Public Domain.

1.49 2.11 Gesucht...

2.11 Gesucht...

Gesucht werden:

1. Übersetzer der Katalogdateien in verschiedene Sprachen, da ich selber nur des
Deutschen und Englischen mächtig bin. Es ist ratsam, mich vorher zu fragen, ob ↩
evtl.
nicht bereits ein hilfreicher Zeitgenosse bereits beschlossen hat, mir zu helfen. ↩
Außerdem
braucht man ja auch die .cd und .ct-Dateien.

2. Falls jemand Vorschläge zur Verbesserung der Dokumentation hat, so kann er es ↩
gerne
tun. Falls jemand meint, ein Kapitel im Programm wäre etwas zu kurz und knapp ↩
geraten,
darf er mir ruhig eine verbesserte Version zuschicken. Dies betrifft natürlich ↩
auch
die theoretischen Teile, also die Kapitel, in denen ich versucht habe, in Kürze ↩
das Wichtigste an theoretischem Wissen zu
vermitteln, aufarbeiten.

3. Was ich mir von Anfang an gewünscht habe, war ein Programm, das nicht einfach
nur Fraktale berechnet, sondern mehr einem Buch über Fraktale gleicht, bloß eben
mit den interaktiven Möglichkeiten eines Computerprogrammes. Ich stelle mir nun
folgendes vor: Ein AmigaGuide-Kapitel über Fraktale, in dem die Knöpfe, die der
Benutzer anklicken kann, keine Links zu Kapiteln darstellen, sondern Arexx-Scripts
ausführen. Dies geht mittels des Schlüsselwortes 'RX' <Arexx-Script-File> statt ' ↩
LINK'
<Nodename>. Da nun ChaosPro ein Arexx-Interface hat und außerdem der AmigaGuide ↩
bei
Druck auf die Help-Taste asynchron innerhalb von ChaosPro läuft, sind alle ↩
Voraussetzungen
geschaffen, über ein AmigaGuide-Dokument durch Knöpfe ChaosPro fernzusteuern. Wer ↩
möchte sich als Autor versuchen? Mir liegt
programmieren mehr als Texte schreiben...

```
4. Wer hübsche Bilder berechnet hat, der kann mir natürlich die Datenfiles davon ↩
   schicken.
Auch an Farbpaletten bin ich interessiert. Aber denkt nicht daran, die Paletten ↩
   von FractInt
oder aus einem Archiv mit Namen FractInt_Extra (oder ähnlich, in der Dosenwelt ↩
   wohl eher
was in der Form FRAXTRA4.ARJ oder FRAXTRA5.ARJ zu konvertieren - das habe ich ↩
   bereits getan, und
werde es auch tun, falls da noch mehr kommt...)

5. An Verbesserungsvorschlägen leglicher Art bin ich natürlich sehr interessiert. ↩
   Sei es ein
Rechtschreibfehler in der Hilfe-Datei in Zeile xxx oder aber der Vorschlag, das ↩
   Programm netzwerkfähig (Stichwort:
verteiltes Rechnen) zu machen...

6. Ich weiß nicht, wo die Freizeit des letzen Jahres geblieben ist, wer kann ↩
   suchen
helfen ;-) ?
```

1.50 Humor

```
Dieses Kapitel ist für all diejenigen gedacht, die sich die Mühe machen, die ↩
   Dokumentation schön brav
durchzulesen. Dieses Kapitel ist mit voller Absicht nicht im Inhaltsverzeichnis ↩
   aufgeführt. Keine Angst, dieses
Kapitel ist auch das einzige versteckte Kapitel...

(Gesammelt aus verschiedenen Newsgroups aus dem Usenet):

- und im uebrigen bin ich der Meinung, daß... AAAAARRRRGGGHH   !!!!!

- Ich *liebe* diese Mentalitaet: 'What does this button do?' *click* -- BOOOOM!


- "Als wir herkamen, hatten wir einen A4000-Tower im Auto"
  "Waaaas???",
  "Ja, fuenf 4000er übereinander!"

- Warum heiraten - leasing ist doch so einfach !

- Wenn Du nicht Teil der Loesung bist, bist Du ein Teil des Problems!

- WindowsError 002: No error . . . yet
- WindowsError 003: Dynamic linking error...Your mistake is now in every file
.
- WindowsError 010: Reserved for future mistakes
- WindowsError 011: Virtual Error, just for fun...
- WindowsError 014: Nonexistent error.  This cannot really be happening.
- WindowsError 056: Operator fell asleep while waiting.
- WindowsError 058: Intel inside
- WindowsError 076: Harter Fehler: Sitzen Sie?
- WindowsError 078: BEAMTEN-ERROR:
Ihr Vorgang ist in Bearbeitung, bitte verwenden Sie beim nächsten Systemaufruf ↩
   Ihre Personalnummer, die
```

wir Ihnen binnen drei Monaten zuteilen.

- Haben Sie Probleme mit Ihrer Hardware? Haben Sie Probleme mit Ihrer Software ?
  DANN BLEIBEN SIE MIR JA VOM LEIB!!

- Warum wird das MSDOS nicht fehlerbereinigt ?
  Damit viele Entwickler sich darauf verlassen können, daß
  die Fehler auch noch in 20 Jahren existieren.

- Was braucht man, um einen durchschnittlichen PC halbwegs vernuenftig nutzen zu koennen ?
eine Grafikkarte, ein CDROM, Lautsprecher, Soundkarte, mehrere 4MB-Simm-Module, LINUX, eine GB-Platte,
viel Geld und einen gewissen Hang zum Masochismus.

- Wer Intelligent ist, braucht keinen Intel mehr. Aber wer ist schon motorolaligent?

- "Windows oder MS-DOS werden auch noch den Intel 868 bremsen.." (Bill Gates)

- Der teuerste Sprengstoff der Welt : Pentium ohne Kühlkörper.

- Worin liegt der Unterschied zwischen MS-DOS 3.0 und 6.2 ?
  Im Speicherplatzbedarf und in der Versionsnummer !

- Wir danken der TELEKOM fuer die tollen Leitu/&%ßd¡opR
  NO CARRIER

- "...seit kurzem stuerzt mein Amiga andauernd ab..!!!"
  "Das liegt an der MERLIN !!!"
  "Aber ich habe gar keine MERLIN !!!!"
  "Trotzdem....."

- Eigentlich wollte ich heut gar nichts tun, aber das habe ich auch nicht geschafft

Die Schlagwörter der Computerindustrie:

"abwaertskompatibel"
  kann genauso soviel wie sein Vorgaenger

"aufruestbar"
  das Grundgeraet alleine ist wertlos

"aussergewoehnlich vielseitig"
  es gibt viele Anwendungen, die das Geraet nicht beherrscht

"beeindruckend"
  niemand haette gedacht, dass wir es wagen

"bewaehrte Technologie"
  veraltetes Geraet

```
"Commodore"
  Lieblingsfirma der Boxfuehrung

"Creativ-Wunder"
  man braucht viel Phantasie um mit dem Schrott das zu machen, was man
  eigentlich wollte

"Denkt mit und denkt weiter"
  wird immer das Gegenteil von dem tun, was es soll

"einfache Bedienung"
  Idiotensicher bis jemand die Tastatur benutzt

"einsatzbereit"
  laeuft noch

"ergonomische Gestaltung"
  der Ausschalter ist ohne Schraubenzieher erreichbar

"eroeffnet neue Dimensionen"
  es kommt alles noch schlimmer

"erwartet"
  aber nicht erfuellt

"erweitert"
  zu den altbekannten Fehlern sind neue hinzu gekommen

"frei programmierbar"
  es ist noch keine Software dafuer vorhanden

"Floppy-Speicher ...  zum Freihalten des Arbeitsspeichers"
  laedt ums Verrecken nicht

"gestochen scharfe Zeichendarstellung"
  fuer Brillentraeger unbedenklich

"handelsueblich"
  wird von uns nicht mitgeliefert

"hochspezialisierte Creativ-Computer-Technik"
  kann absolut nichts, aber das ganz besonders gut

"integriert"
  minderwertige Einzelteile vereint in einem katastrophalem Ganzen

"intensiver"
  gelebter Hass

"keine Programmiersprache noetig"
  es ist keine vorhanden

"komfortabel"
  sturzt bei Eingabefehlern nicht immer sofort ab

"kompakt"
```

```
  alle Geraeteteile, die heiss werden, sind auf einem Punkt konzentriert

"Komplettloesung"
  man bekommt den Muell nicht einzeln, sondern nur im Paket

"konsequente Weiterentwicklung"
  wir haben alle Fehler nochmal gemacht

"meistgekaufte"
  wir haben die beste Marketingabteilung

"Multicolor"
  mehr als eine Farbe

"Multitasking"
  es gibt viele Wege zum Absturz

"Option"
  erfuellt vielleicht irgendwann die Erwartungen

"professionell"
  funktioniert manchmal

"schnelle Fenstertechnik"
  fliegt ziemlich schnell aus demselben

"schoen und repraesentativ"
  alle Vorteile des Geraets in drei Woertern

"Speicherwunder"
  es geht mehr rein, als jemals wieder rauskommen wird

"spezieller Soundprozessor"
  macht viel Laerm um nichts

"sprechend"
  produziert unverstaendliches Kauderwelch

"Standard"
  abgekupfert

"modernste Technologien"
  besser koennen wir es nicht

"schneller"
  am Ende

"ueberraschendes Preis-Leistungs-Verhaeltnis"
  die Leistung des Geraets entspricht dem Preis seiner Verpackung

"ungeahntes Anwendungsspektrum"
  nur fuer abartige Aufgaben bedingt brauchbar

"vereinfachte Arbeitsablaeufe"
  auspacken, einschalten, wegschmeissen

"zukunftsweisend"
```

der Abwaertstrend geht weiter

1.51 Index

III. Index