Fractal Playtime Vers 1.0

A Program for Exploring the Mandelbrot Set

Richard Hodson

April 3, 1993

Disclaimer

Terms and Conditions

Mail to: rhodson@cix.compulink.co.uk

Contents

Chapter 1

Introduction

1.1 History and Overview

There are already many Mandelbrot set generators in existance, some fast, some slow; some usable, some unusable. This program was started in April 1990 as a simple generator program which took a typed in set of coordinates, and plotted a low-res output screen which was then saved as a NEOchrome file. The reason for writing it was simple: none of the programs I had seen used colour very well. My program remembered the values it had calculated for each pixel, and after drawing the image used it's hindsite to redraw the image with intelligent use of colour. This statistics feature allows the user to zoom into a pattern which looks a real mess, and with a couple of keypresses turn the mess into a wonderful swirling masterpiece.

Features were added as I thought of them and parts of the original 'C' code were rewritten in assembler. Finally, the whole thing was made totally GEM legal, allowing it to print, run on many non standard machines by using GEM screen calls and hopefully making it a lot easier to learn how to use. Statistics freaks may like to know that currently the source code is over 100K bytes of 'C' and a further 12K bytes of assembler. Monochrome users will be pleased to know that most of the development work is done in ST high-res, which really can look good because there are so many pixels on screen.

This program has grown from humble beginnings to what I intend to be the most outstanding Mandelbrot generator available. It was originally written on a 1040STF, with later development on a 4Mb upgraded STE and further writing/debugging on an 8Mb TT.

The idea was to write a *very* fast Mandelbrot generator which can run on *ANY* ST related hardware in *ANY* resolution. So far it can cope with all ST and TT screen modes as well as an ST running Monster screen with a mono monitor. It uses the cookie jar feature of TTs and later STs to find what hardware is available. It can sense whether it has a 68020 or better CPU and whether there is a maths co-processor available. If it finds either of these, it will use them where appropriate. Where possible everything is made to be automatic; the program works out what is best and gets on with it.

1.2 Dive in… (a brief tutorial)

Once you have started the program up, you will be faced with a window which is where the patterns will appear. The program will automatically start calculating an overview of the entire M-set which you can explore.

The mouse pointer will re-appear once the calculation has finished. To zoom in on any part of the image, use the mouse to drag a box around the part that you want to magnify. Press return or click on *Calculate* from the *show* menu to start the calculation. If you decide that you want to abandon a calculation, hold down the right mouse button until the cursor comes back.

When you have calculated a pattern, go to the options menu and click on 'Boundary…' to call up the boundary colouring options form. This allows you to alter the way that colour is used to show the picture (or where the black and white bands are placed in mono) by using statistics collected during the calculation. Try choosing '1' for the cycles box, and 'Equal' from the methods box, then click on 'OK'. This combination usually gives the best results, with the others being useful in certain circumstances. All of the values calculated for the pixels are stored in the computer's memory, so the screen update is much faster than the original calculation. For more details, see section 2.4.2 on boundary selection.

You now have the basics of how to work the package, all of the rest being options and shortcuts. There is a stopwatch which you can switch on and off which tells you how long a picture took to calculate, and an option to use the full screen rather just the inside of the window. Both of these options are available on the Options menu under 'General…'. One handy hint is to size the window down to a small size when trying to find an interesting picture, and then make it full size to view it properly. Generating a smaller image takes a lot less time.

To see the sort of results possible, go to the *Load…* part of the *File* menu and select the file *DEMO.CO*. This is a rolling demo which calculates a series of pictures and then displays them with even and detail statistics settings. The sequence will go back to the beginning when it ends, but you can escape from it by holding the right mouse button.

Chapter 2

The Menus Explained

2.1 The Desk menu

2.1.1 About Fractal

This is an information form. It tells you the version number of the program and also has three strings which it will sometimes display.

- 680x0 Processor mode

  The program fills the 'x' in with a number to describe what type of processor it thinks you have. The *Cookie Jar* is used to get this information. If no cookie jar is found, a normal 68000 is assumed.

- Floating point found

  Should be displayed if you have a TT or Falcon with a floating point co-processor. Again, the *Cookie Jar* is used to find this. If no cookie jar is found, it is assumed that no FPU exists.

- GDOS is present

  Uses a special call to GDOS to test whether it is present. GDOS is only used for printing or for writing to non standard video modes. It should not matter whether GDOS, FONTGDOS or FSMGDOS are used.

2.2 The File menu

2.2.1 Load...

Load a co-ordinate or driver file (a file with a *.co* extension). In simple use this allows you to load the co-ordinates which you have saved previously using the *Save...* option. For more advanced uses, see section 3.5.1 on file formats.

2.2.2 Save...

Saves a co-ordinate file for the currently viewed image.

2.2.3 Image Type...

Brings up the selection form for choosing how images are saved to disc. Currently only IMG, NEOchrome and Degas files are supported. NEOchrome will not accept anything other than ST Low resolution, but some other packages will so Playtime allows you to save NEO files in any ST screen mode.

2.2.4 Save Image...

Allows you to save the currently displayed image in the currently selected graphic file type. If you are saving an IMG file there should be no problems. The other file formats, however, can only save a full screen. If saving in a format other than IMG it is recommended that you use *full screen* mode from the *general options* form to prevent getting a boarder on the right and lower sides.

2.2.5 Quit

Leaves the program. The palette is restored to what it was before the program was started.

2.3 The Show menu

2.3.1 Calculate

Starts a calculation. To stop in the middle of a calculation, hold down the right mouse button until the cursor reappears. Once a calculation is stopped, it cannot be restarted from where it left off, it has to be started again from the beginning.

2.3.2 Co-ords...

Allows you to type in a set of co-ordinates for the next calculation using the co-ordinate system described in the section below “What the numbers mean”. It can also be used so that you can see the co-ordinate values of the current picture. The ‘Clear’ button wipes all of the text out of the text fields to make typing in new values from scratch easier.

2.3.3 Zoom in

This uses the current co-ordinates and increases the magnification by a factor of 2. The result is that you zoom in towards the centre of the screen.

2.3.4 Zoom Out

Similar to Zoom in, except that the magnification is reduced by a factor of 1/2. This zooms you out around the centre of the screen. This can be useful if you select an area, and then realise when it is drawn that you are not exactly where you want to be. Zooming out gives you a picture from which you can usually re-select with the mouse to get the region that you wanted.

2.3.5 Start View

This sets the co-ordinates and magnification back to what they were when you came into the program.

2.4 Options menu

2.4.1 General

- Video writing

  Direct or *GEM.* In direct mode, the gem VDI is bypassed and the program writes directly to video memory. It does this in as clean and portable way as possible, but even so it is a Bad Thing according to Atari. If you have a non-standard video display and tell the program to use Direct, it may sense that something is wrong and use GEM anyway. Only use GEM mode if you experience compatability problems in Direct mode.

- Maths Processor

  Enables or disables a maths Floating Point Unit. If no usable FPU is found then this will be set to *Ignore* and then disabled. See the section *Different Machine Types* for more details.

- Full screen

  This is either on or off. With full-screen switched off the pictures are drawn into the window allowing you to control how much of the screen is used and so allows you to create small images quickly. In full screen mode the window and menu bar are removed and the entire screen is used for the picture. When the picture is complete it is saved into memory and the window and menu bar are restored. Only a portion of the overall picture can be shown in the window, but any manipulations on the image (boundary changes, saving to disc) will happen to the whole image and the whole image can be saved to disc.

- Pre clear

  An on/off option like Full screen above. Controls whether the screen is cleared before starting a new calculation. The final result is identical, so it just depends on what you like to watch while the calculation is being carried out. If you are using monochrome then in the pre-clear mode only the black pixels need to be plotted, so Playtime can speed itself up a little.

- Stopwatch

  The stopwatch gives you an indication of how long the calculation took. This time includes clearing the statistics, screen and pixel value buffers as well as the time for the calculation. The 200Hz clock is used, so the accuracy is about 100th of a second.

  Two readout styles are available. One just works in seconds, the other gives hours, minutes and seconds (and days if you are doing something outrageous).

2.4.2 Boundary

This is where the statistics features of Playtime are controlled. As these features are so central to the program, all of the options in this dialogue are available as single keypresses. See the section on keyboard shortcuts for more on this.

As the image is calculated the values calculated for each pixel are stored in memory. These can then be used to map calculated values to screen colour in several ways. In each case these routines will try to produce as many bands of colour on screen as there are colours available. For example, if you are in a 16 colour screen mode, the image will be given 16 bands of colour. Sometimes this is not enough, so the *Cycles* setting can be used to increase the number of bands on screen.

Cycles is simply the number of times the palette will be used (the number of times the routines will cycle through the available colours). How high cycles is set to depends on the resolution and the number of real colours available. In ST low res cycles will generally be just one or two. In ST high res you may want to use around six or seven.

The way the statistics routines work is given below. Note however that some are sensitive to the type of image they are given to process, and so may not produce acceptable results all of the time. Just experiment until you get something you like.

- None

  No statistics interpretation is carried out, the values are plotted straight to the screen. The *Cycles* setting is ignored in this mode.

- Equal

  Values are grouped together to try to produce bands with equal numbers of pixels in them. If a screen looks a total mess with *None* selected, try it with this.

- Detail

  This is for screens such as the start calculation where there are some very large bands with nothing of interest. The colours are allocated so that the number of pixels in each band gets smaller and smaller, so concentating the colours into the detailed regions.

- Fixed

  The boundary points are placed at fixed intervals with no statistics processing. For example, if the dwell is set at 100 and there are four colours available, then boundaries will be 1–25, 26–50, , 51–75 and 76–100;

- Outline

  Displays only the central Mandelbrot set (the points in the image at which the dwell went to the maximum limit). This was not really included for displaying images, but more for showing where regions of interest might be for zooming into when looking at a cluttered image.

- Exclude

  Designed for monochrome users. In high res the lack of colour can make the region around the Mandelbrot set look untidy. In *exclude* mode the Cycles setting sets how many of the lower levels are displayed. Only the lower levels and the pixels which hit the Mandelbrot limit are displayed, with the lower levels being displayed with no statistics processing.

2.4.3 Dwell

2.4.4 Method

There are two methods of calculation available, and three different precisions at which the calculations are made. The methods and precisions can be mixed in any way.

As a rule, leave the precision setting to automatic and let the computer decide what is best. On an ST the Div-con method is generally the best, though on a TT the TV scan system is nearly as fast. In either case, I think that the div-con method is much more entertaining to watch.

- The TV Scan Engine

  This is the basic traditional way in which a Mandelbrot image is calculated. Named TV scan because it calculates the pixels from left to right, top to bottom and that is the path that the electron beam in a TV set takes when it is drawing a TV picture. Nothing fancy is happening here to optimise the image drawing, so the image is guaranteed to be correct.

- The Div-Con Engine

  Based on an alogorithm by Rico Mariani.

  Div-Con stands for *Divide and Conquer.* The idea behind this is the mathematical proof that if a line is drawn around an area of a Mandelbrot image and all of the points on that line have the same value, then all of the points enclosed by that line will also have the same value. Computationally, it means that you can calculate the values around the perimeter of a box, and if the values on the perimeter are all the same then the box can be filled in with that value without having to calculate anything else, so saving a lot of calculation (see Monochromatic theory in the glossary).

Unfortunately, there is a flaw in doing this. Calculating some points around the perimeter does not prove that *all* of the perimeter is of that value. Imagine that between two of the points calculated to be of the same value there is another point which is not of the same value, then that point will be missed and the area will be filled in even though the perimeter values are not all the same. Fortunately, this does not happen often and the TV-scan and Div-con images tend to look the same.

- 16 Bit Fixed Point

  This is the fastest precision. It uses 16 bit integers to do the calculations, and so can use the 68000's hardware 16 bit x 16 bit multiply instruction to good effect. Unfortunately, 16 bits do not store numbers very accurately, so this precision is only useful at very low magnifications.

- 32 Bit Fixed Point

  This works like the 16 bit routines, but with more accuracy. This is the most useful precision as it works with quite high magnifications and yet is still fast. To do a 32 bit multiply the ST has to do four 16 bit multiplies, so it is not as fast as the 16 bit routine.

  TT users will be pleased to know that the 68030 does have a 32 bit x 32 bit multiply instruction, so there is a TT specific routine as which is almost as fast as the 16 bit routine! Selection of this routine over the ST version is automatic.

- Floating Point

  If 32 bit fixed point starts to look blocky then you will have to use floating point. This is 64 bits on an ST, or 80 bits on a TT with the 68882 maths co-processor. Most users will want to avoid this altogether and the automatic selection system ignores it because it is so slow compared with the 32 bit mode.

- Auto

  In auto precision mode the program will choose between 16 and 32 bit precision at the start on each calculation. The decision is based on the screen resolution and the magnification of the calculation and allows you to forget precision and let the computer work out the fastest way of generating an image. If auto is switched on, then the precision last used will be displayed in the dialogue. This is purely for information in case the user wants to know exactly how their last image was generated.

2.4.5 Edit Palette

The palette editor allows you to manipulate the ten palettes held in memory and maintain a library of palettes on disc. Example palettes come wih the program in a set of directories, one for each of the ST/TT standard screen modes. There is, however, nothing to stop you from loading palettes not meant for your screen. If you load a sixteen colour palette into a four colour mode, then only four colours will be displayed. Similarly, if a four colour palette is loaded into a sixteen colour mode, only the lower four colours will change. Note that in this last case, saving the palette again will only save four colours as the computer remembers what the palette was originally from. Using the *Info...* button will display what the program knows about the palette on display. If a palette is modified, then the info settings are changed to those of the current machine and mode.

In this release, only 256 colours are allowed in a palette. More than this may cause the program to crash. To access more than sixteen colours, a bank system is used similar to the Xcontrol control panel from Atari.

If you should decide that you have a desk accessory which you prefer as a palette editor, then you can use that to edit your palette settings, and then go into the Fractal Playtime editor and use the *Grab* button. This stores the screen palette into the currently selected palette number, and changes the palette name to *Desktop*. The grabbed palette can now be loaded, saved and manipulated like any other.

2.4.6 Print

Printing is only available if you have GDOS installed. Hopefully, you will have got a copy of Hyperpaint or Hyperdraw with your ST, both of which come with a cut down copy of GDOS. Alternatively, programs such as Timeworks DTP and Degas Elite come with full implementations, as does the Atari Laser printer.

There are dwo different styles of printing available. One is a simple screen dump. This does not take very long to perform, but gives a very ragged picture. The other is *Recalculate*, where the calculation is started from scratch and done to the best resolution of the printer. The calculation has to be done with TV scan, and is done with the 32 bit integer routines. Because of the very high resolution that even a 9-pin printer can achieve, this calculation will take a long time. The results, however, can be well worth the time.

The *Screen dump* mode is intended just for rough outlining, and gives some idea as to whether the colour mapping is ok. Monochrome users have an advantage here with a black and white printer as the screen gives a reasonable preview. Colour users should be aware that every colour band on screen will map to either a black or white stripe on a mono printer, so low res images may look cluttered. Use medium res to get finer control of how many bands there are.

The *Setup...* button will take you to the page setup form, which will allow you to position your picture on the page. All measurements here are in millimeters, and are accurate to a few millimeters across the length of a page on most printers. The maximum printable area reported by GEM is given at the top of the form, and then some editable fields with the current settings for widths and offsets. The program should be about right for a 9-pin epson as supplied. Please note that when doing a screen dump, the entire print area will not be filled. Use landscape mode for the fullest picture, but you will only get the full area used in recalculate.

The radio button for *Print Text* can be switched to on or off. When on, the co-ordinates of the view will be printed at the top of the page, whether in landscape or portrait mode.

The *Test* button will print a test pattern to show where the final image should appear on the page. The pattern is simply a right angle at each corner of the image, with the last viewed co-ordinates printed if co-ordinate printing is enabled.

2.4.7 Save setup

The program has an install file called PLAY.PAR which it loads automatically on startup. If there is no file present, then the program will load all of the palettes with the desktop palette, and set all of the options to a set of default values.

Save setup allows you to set the options to what you want them to be from when the program starts. The following items are saved in the PLAY.PAR file:-

- Palette information
- Stopwatch settings
- Fullscreen on/off
- Pre-clear screen on/off
- Boundary choice settings
- The chosen method of calculation
- Size of minimum box used by div-con
- File type for saving images
- Printer settings
- Maths co-processor settings

The file format has been designed so that hopefully regardless of what features are added to it in the future, it will still be possible to load your old files in future versions of the program.

2.4.8 Shortcuts

Displays a form which gives a reminder of the keyboard shortcuts.

Chapter 3

In Detail

3.1 The Co-ordinate System

There are many variations on the Mandelbrot theme, so if there are any terms that you have difficulty with, try looking them up in the glossary section near the end. Some of this manual may expect you to understand complex numbers. If you have difficulty with complex numbers, then get hold of a maths text book. A good book should teach you all you need to know in a couple of hours, and give you a far greater enjoyment of the Mandelbrot set.

The co-ordinate system used here is to give the complex co-ordinate of the center of the screen, and a magnification. The magnification sets the largest size of circle which can be displayed on the screen. For example, a magnification of 1 allows a circle of radius 1 to be displayed in the center of the screen with it's edges just touching the nearest two sides of the rectangular screen. When the program is first started, the initial display is of the whole $\mathcal{M}$ set at magnification 1/2. This allows you to see a circle of radius 2, the escape radius used in the dwell calculation routines. When describing the view, just giving the magnification and location assumes a square screen.

To completely describe a view, you should also give the aspect ratio of the screen, which is the X resolution divided by the Y resolution. In medium resolution this is complicated by the program using different scales for the x and y axis due to the very rectangular pixels, which halves the aspect ratio. This makes the aspect ratio is 1.6 in all ST resolutions when in full screen mode. The TT low res is 0.67 and TT medium 1.33.

The other often used system of top left and bottom right co-ordinates was felt to be less intuitive, and is no better at dealing with changes in resolution.

3.2 Keyboard Shortcuts

Some of the more often used functions of the program have been given keyboard shortcuts. These shortcuts are all single keypresses and do not require the alternate key to be pressed. The shift and Caps Lock keys are both ignored. Only the quit shortcut requires the control key to be held down.

- Statistics Shortcuts

keypress	*function*
1—9	number of times to use palette colours
D	*D*etail enhance
E	*E*qual band sizes
F	*F*ixed band sizes
N	*N*o statistics adjustments
O	*O*utline Mandelbrot set
X	e*X*clude range

- General Shortcuts

keypress	*function*
CNTRL-C	quit program
Help	display shortcuts
Clr/Home	display start image
Return	calculate selected image
C	enter *C*o-ordinates
P	screen *P*re clear mode
S	full *s*creen mode
F1—F10	select palette

3.3 How settings affect speed

There are several optimisations built into the code which tend to come on automatically, so you really do not have to know the next section. If you have any customisations to your machine such as graphics cards and processor accelerators the following may not be true, and you will have to use the stopwatch to work out what is best for you.

- If a known ST/TT screen resolution is found, a direct to screen memory plotting routine is used.

- If full-screen mode is being used and the program is plotting line by line, the fastest plotting mode is used as the program can predict how and when every pixel will be plotted from the start.

  Note that both of the above can be disabled using the 'Video writing' option in the general options form.

- Pre-clear may be faster for monochrome users if direct video writing is being used but not full-screen.

- Avoid floating point calculations. Even on a TT they are a lot slower than the 32-bit fixed point.

- Div-Con is faster than TV scan on a normal ST. This is not as important on a TT because of the processor cache, but is still true.
- Experiment with the Div-Con box size. Use around 6 for an ST and 12 or 14 for a TT, the stopwatch will allow you to find out what is best. The setting depends very much on what is being calculated, and so as the images which I enjoy exploring may be different from yours you may be able to get a speed advantage here.

3.4 Different Machine Types

3.4.1 Notes for 520 ST owners

You should be able to use Low resolution with no problems but may have to boot up without any desk accessories loaded in medium resolution. High resolution will not work on a 1/2 meg machine as 500K is needed for statistics.

3.4.2 Notes for hard disc users

Feel free to install the program on your hard drive. There is no performance increase (except perhaps when printing) but you may find it easier to use.

3.4.3 Notes for STE users

The STE has a better palette than the old STFM. This allows an STE to load palettes from an STFM as well as an STE. The palette editor will recognise your palette and use it. If you have your blitter switched on (there is an option on the GEM Desktop) then it will be used.

3.4.4 Notes for Mega STE users

I have not been able to test for this machine, but there should not be any problems. This should be recognised as an STE, but should be faster.

The Floating point boards produced by Atari and others for the ST cannot work in the same way as the TT works. Blame this on Motorola, not Atari; the plain 68000 was never designed to have an FPU. This means that at the moment your FPU is not being used. I will need to produce a separate program version for the Mega STE, but unfortunately cannot test it if I compile one. This should hopefully be cured in the next release of Playtime.

3.4.5 Notes for TT users

Use of TT features is automatic. The ‘About Fractal’ form will tell you what the program has found and on a TT it should say that you have a 68030 and floating point. The program should load into and use TT ram if you have any.

3.4.6 Notes for upgraded ST users

The TT code is actually compatible with a 68020, so if you have a '020 or '030 processor accelerator then this code will be useful. To detect the processor, the CPU cookie in the cookie jar must be set. I assume that this is done by the accelerator, but if it is not then get in touch and I will try to write a program to set it for you.

The screen routines can currently handle any screen with a GEM driver and 2, 4, 16 or 256 colours on screen at once. Other setups are coded for but have never been tested. More than 256 colours on screen will definitely crash the program. Also the more pixels there are on screen, the more memory will be needed. For each pixel the program will need 2 bytes for statistics and memory for a blit buffer. For example, in TT medium res a pixel takes 1/2 a byte for the blit buffer and 2 bytes of statistics. With 640x480 pixels it needs to find 640x480x2.5 = 750K. ST low needs 156K and a Crazy Dots board will need about 4.5Meg. Most people seem to buy memory before a graphics card, so hopefully this is not a problem.

3.5 File Formats

3.5.1 Co-ordinate file

The co-ordinate file is in ASCII to allow a user to alter them in a text editor. The easiest way to understand this format is to examine a file in an editor or double click on a file from the GEM desktop and use the *Show* option. Note that although the program only saves a simple file, much more can be loaded allowing sequences of images with pauses, statistics control and looping back to the start. Try examining *demo.co* for a good example.

The file consists of lines of text, each with a letter at the start. The letter is like a command, and can be followed by data such as a number. The currently allowed command letters are...

- ; <comment>

  Comment line. The entire line is ignored.

- b <number> <option>

  Boundary settings. Does a screen redraw with the given boundary options. Number is 1 to 9. Option is one of the following...

Option	*Description*
d	Detail enhancing distrubution.
e	Equal band area.
f	Fixed boundary positions.
n	No statistics operations.
o	Outline maximum dwell area.
x	No statistics but with exclusion band.

- c

  Start a calculation.

- d <integer>

  Maximum dwell limit of view.

- i <number>

  Imaginary co-ordinate of view.

- m <number>

  Magnification of view.

- o ({+}|{-}) <option>

  Allows options to be switched on or off. Allowed options are...

Option	*Description*
c	Clear screen before calculation/redraw.
d	Direct video write rather than use GEM.
f	Full Screen.

- r <number>

  Real co-ordinate of view.

- w <integer>

  Wait for <number> of seconds.

3.5.2 Palette file

This file is a binary file consisting of a 40 byte header. followed by however much data is necessary. The last entry in the header field is an extension length. If the palette file header is extended, then this field will say how many bytes have been added to allow upward file compatibility in later versions.

Offset	*Size*	*Description*
0	long	magic number 0xa21e29b7
4	string	name of palette, null terminated
34	short	number of colours defined
36	short	number of rgb levels in palette
38	short	header extension

The data section consists of three 16 bit short integers for each colour; one for red, green and blue. These numbers are in GEM format where each of the RGB intensities is in tenths of a percent, ie from 0 to 1000.

3.6 Compiling the source

First I would like to re-emphasize that this code is for personal use only. The bulk of it is there just for you to read and tinker with. The exception to that is the assembler file '*dwell.s*' which I am placing in the public domain. There are no new ideas there, and it represents a very small fraction of the development effort that went into the program. Besides, if you can understand it, you could probably write it anyway.

The source code as distributed uses many features of the GNU C compiler. You will have difficulty if you wish to convert the program to another compiler, and as GNU seems to be the best available on the ST (and gives you the ability to compile many UNIX programs) I intend keeping this program GNU specific. People with smaller machines may be able to do something with the assembler routines and learning how the program works.

If you really wish to convert it, you will need to convert the assembler routines which are written in Motorola RTL rather than in the more usual assembler found on the ST (this is a UNIX hangover). The next problem facing you will be that I use integers with 32 bits. In GNU you get to choose 32 or 16 bit ints, and 32 made the code easier to write. The next release will probably be 16 bit to make the program a little faster and more compact. Also, I hack the GEM libs to get inline GEM calls where speed is wanted, and normal GEM calls where speed is not important to save space. Good luck converting that.

This distribution was compiled with GNU 'C' version 2.1 using patch level 80 main libraries; patch level 23 GEM libraries and patch level 17 maths libraries. The program could be compiled with lower versions than this, but if you use these versions you should have no problems.

3.6.1 About GNU 'C'

GNU 'C' is a public domain 'C' compiler. It is the main system compiler for the GNU project, a project to get a public domain version of UNIX which is better than the currently available commercial versions. It can produce code optimised for ST, Mega STE or TT with proper support for the FPU. If you buy a UNIX based TT, this is the compiler it comes with. It can switch from K&R compatible to ANSI, and comes with a C++ compiler. A FORTRAN to 'C' convertor is available, and other related compilers are in development.

If you do not have GNU 'C', you will need about 4 to 5 megs of hard disc space to install it and at least 2 megs of RAM to get it to run. If you are serious about your programming, this is the system for you, so upgrade if you have to.

If anyone is still unconvinced about GNU 'C', here is a badly written program which I ran through GNU 'C'.

```
int
main()
{
    int a, b;

    a = b;
    printf( "%s %d\n", a );
}
```

Which produced the folowing warnings...

```
test.c: In function ‘main’:
test.c:7: warning: implicit declaration of function ‘printf’
test.c:7: warning: format argument is not a pointer (arg 2)
test.c:7: warning: too few arguments for format
test.c:4: warning: ‘b’ may be used uninitialized in this function
test.c:8: warning: control reaches end of non-void function
```

To me, this error detection ability alone makes it worth having.

3.7 Glossary

- Dwell

  Dwell is the number of iterations that the calculation goes through for one pixel.

- Dwell Limit

  This is the maximum amount of dwell that a pixel can get. Some coordinates, such as 0i+0, will never break out of the calculation loop unless this artificial limit is imposed.

- Escape Radius

  The magnitude of z at which z is said to be large enough that the calculation must be diverging, so stopping the calculation. This is pre set to 2 in this program and for now cannot be changed. If the escape radius is set to less than two, then you will get a bad approximation and strange results. Anyone writing their own programs may like to try higher escape radius values, such as 100.

- FPU

  *F*loating *P*oint *U*nit. Also known as a floating point co-processor. This is a chip which works alongside the main processor to help with real number calculations such as cos(0.72) or $x = 1.534 + 1/y$. Comes in two types, the 68881 and the 68882.

- Magic number

  A special number which is used in a file to check that the file is what the program is after. TOS uses magic numbers to identify executable programs, Fractal Playtime uses them in it's palette and save setup files. My numbers are generated randomly by tossing a coin 32 times to get a 32 bit number.

- Magnification

  At a magnification of 1 the program will just be able to display a circle of radius 1 in the center of the screen. If you look at the start calculation done by the program, you will see that there is a circle of radius 2 which the Mandelbrot pattern sits inside, and that the magnification is 1/2 in order to show it.

- The Mandelbrot Set ($\mathcal{M}$)

  Named after the discoverer of the set, the mathematician Benoit B. Mandelbrot.

  This is the set of pixels which manage to reach the maximum dwell. Strictly, the true Mandelbrot set is the set of pixels which will never reach the escape radius. It is a bit difficult to prove that a point will do this (though not impossible) so, traditionally, the Mandelbrot set is when the dwell reaches the limit of 1000.

- Monochromatic Theory

  A theory generalised by Rollo Silver in his newsletter on fractals called *Amygdala* in issue 7, November 1 1987. It says that if all the points on the boundary of a region – rectangular or otherwise – have a dwell in the range d. . . D, then all points in it's interior have dwell in that same range, where $0 \leq d \leq D \leq \infty$.

- TT RAM

  A special high speed form of memory only found on the TT. not all programs can run in TT ram, but those that can run noticably faster.