Voxel

Alastair M. Robinson

COLLABORATORS

	TITLE : Voxel	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Alastair M. Robinson	June 24, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Voxel

1.1 A Voxel Engine - © 1997 Black Five SoftDesign

```
                  A currently unnamed Voxel Engine
  by Alastair M. Robinson

  Copyright © 1997 Black Five SoftDesign.

  This is a work in progress demo of an engine which is likely to
be used in a racing game.

  Bear in mind that this is a very early preview, so don't be surprised if
you hate the control method, think the car looks crap or think the grass and
sunset colour schemes clash - you're probably right, but the engine has the
potential  to become something good!

  THIS SOFTWARE IS PROVIDED AS IS, WITHOUT WARRANTY OF ANY KIND.  ALL USE IS
AT YOUR OWN RISK, AND THE AUTHOR CANNOT BE HELD RESPONSIBLE FOR DAMAGES OF ANY
KIND, ARISING FROM THE USE OR MISUSE OF THE PROGRAM OR ANY OF ITS ACCOMPANYING
FILES.

  This preview is freely distributable but copyrighted.  This means that you
may not make any personal gain from distributing it, and you should not use
any of the files for any purpose other than playing with the demo!


               Requirements

               Installation

               Running

               Credits
```

1.2 About the Chunky to planar libraries....

```
  The GSChunky2Planar libraries were written in 1997 by me (Alastair Robinson),
```

in an attempt to replace the CopperChunky screenmode I used in my Wolfenstein clone a year or so ago. (AMRWolf, just a demo released in Aminet and CUCD III.)
Basically, I wanted to migrate to Chunky2Planar conversion for higher resolutions, without losing the advantage of TrueColour rendering.

That was when I downloaded ASA/Cirion's TrueColour HAM8 routine. This gives a resolution of 160 $\times$ 100, but has the small disadvantage that each pixel must be in RBGB format - which prevents the chunky data being interchangable with a similar routine for graphics cards.
Then a friend gave me a disk full of Chunky2Planar routines he'd collected from various sources, and I began to experiment.

Peter McGavin's C2P4.s was the first routine I attacked: Out came the chunky comparison buffer, (you'd need two anyway if you're double-buffering - three chunky buffers as well as two planar buffers is a bit excessive!) I re-wrote the CPU portion to use the most significant 4 bits instead of the least significant and added support for region restriction.
This has evolved into the C+B-12-?x?.libraries.

The next routine to be attacked was C2P8.s, by Peter McGavin, and modified by Conrad Sandersson. Once again, I removed the chunky comparison buffer and added support for region restriction. I also modified the blitter routines so that only six bitplanes are converted, and split one of the blitter passes in half to allow a larger buffer to be converted. (This was needed for the 1.5$\times$1 library) Most importantly of all, I changed the name to C2P6.s! This routine was used in the C+B-18-?x?.libraries.

Lastly, I wrote a library around Kalms_C2P.s, which I'm pleased to report needed no modification. (Optimising it for 6 planes instead of 8 was a task I couldn't face - but this routine is blindingly fast anyway!) This became the 020-18-?x?.libraries, and these are the best ones to use if you have a 68030/50Mhz or better.

These libraries will eventually be released for general use, along with some equivalents for CyberGrafX, and possibly Picasso'96, but writing these equivalents is not an easy task while I don't own a graphics board!

Contact me if you'd like to furnish me with a suitable routine - but don't expect great wealth as a result; the C2P routines I've used so far are (to the best of my knowledge) public domain, so the libraries themselves will be released as freeware.

1.3 Installing the VoxelEngine...

Just de-archive the entire lha file and copy the entire directory onto a hard-disk.

The voxel engine relies on my
GSChunky2Planar libraries
, also © 1997 by
Black Five Softdesign, which are in the GSChunky2Planar drawer. The specification of these libraries is still subject to change (and it will do

when I support graphics cards), so it would be advisable not to place them
in libs:, and even more advisable not to try to use them in your own projects!
(Public release soon, I promise!)

1.4 What is required for the VoxelEngine...

The VoxelEngine requires a 68020 or higher and (with the current C2P libraries) the AGA Chipset.

I have only tried this version on machines with 8 megs of fast ram, but it may run with less. As usual, the more ram you have, the better.

The engine was written on a 68030/25MHz machine, and has been tested on a 50MHz '030 and a 25MHz '040.
On the '040 it was no faster than on my '030/25, probably due to copyback caches being disabled or something. This problem will probably disappear, along with several others when I ditch AMOSPro and make the engine 100% machine code!

The Configuration GUI requires MUI 3.6 or higher.

1.5 Thanks go to the following people / organisations:

This VoxelEngine was written in 1997 by Alastair M. Robinson and is Copyright © 1997 Black Five SoftDesign.

The engine was written partly in AMOSPro, and uses features from my GameSupport extension, and the GSChunky2Planar libraries (which are still subject to change and so haven't been properly released yet.)

All trademarks mentioned in this documentation are acknowledged as being the property of their respective owners.

Thanks to:
Steve and Chris Hall of Chroma for their encouragement and continual selfless netsurfing.
Stefan Stuntz for the incredible
MUI
system.

Contact Address:

Alastair M. Robinson
The Old Chapel
Syderstone
Kings Lynn
Norfolk
PE31 8SD

1.6 About the MUI system...

The IDEZipTools GUI utilizes

MUI - MagicUserInterface

(c) Copyright 1993/94 by Stefan Stuntz

MUI is a system to generate and maintain graphical user interfaces. With the aid of a preferences program, the user of an application has the ability to customize the outfit according to his personal taste.

MUI is distributed as shareware. To obtain a complete package containing lots of examples and more information about registration please look for a file called "muiXXusr.lha" (XX means the latest version number) on your local bulletin boards or on public domain disks.

If you want to register directly, feel free to send

DM 30.- or US$ 20.-

to

Stefan Stuntz
Eduard-Spranger-Straße 7
80935 München
GERMANY

1.7 Running the VoxelEngine...

To run the engine, just double click on the big Track1 Icon. After a short pause the engine will display a landscape. Moving the mouse left and right will rotate the viewpoint, and holding the left mouse button will make you travel forward.
Don't be surprised if the car behaves strangely on occasions; It will quite often turn 270\textdegree{} clockwise when it should turn 90\textdegree{} ↩
anticlockwise!
I'll fix this soon.

You will start on some grass, you'll have to find the road for yourself!

So what make this engine special?

» The engine runs on screen in TRUE COLOUR, even on AGA machines (HAM8). In theory the 12 bit (HAM6) 3$\times$1 mode could run on ECS chips, but the blitter routines would need to be re-written. Support for graphics cards is planned, but this cannot be done until I re-write the AMOSPro stuff in assembler.

» All graphical routines are resolution-independent, supporting up to 1280$\ ↩
times$1024
as a hard-coded maximum size. The view can be shrunk and expanded at will.

» Two Megabytes are devoted to the landscape, one for altitude and the other for shading. The shading and terraforming are performed by my own track editor which will probably be released in the future.

» A roadway is carved into the hills! (or more correctly, the hills were generated around the road!)

» The road path is constructed from spline curves, giving the track designer extreme freedom.

» Provision has been made for objects to be integrated and properly clipped with the landscape.

» A car object is already implemented (but needs a lot of work!)

Keys:

» F1 - F6 will select the colour scheme for the landscape.

» F7 - F8 will choose between two different sky textures.

» F9 - F10 will select high and low quality rendering modes respectively.

» + & - will change the size of the window.

» Delete and Help will toggle between third-person perspective and overhead views. (The car is not visible in the overhead view!)

» S will save a screenshot into the ram disk. The produced file will be in ImageStudio's native format, VMEM. This was chosen because it is the easiest 24-bit format to save in existence! Saving the file might take a minute or so, depending on the resolution, because the saver is an AMOSPro routine!

» Esc will quit the program.

Resolution cannot be changed from within the engine, you will have to quit and run...
The Configuration Program

This is nowhere near finished, but it gives an indication of my plans for the future. The Save and Cancel gadgets on the first page work, as does the Chunky2Planar page, but everything else is for show only!

To change the Chunky2Planar module, just choose the desired mode from the list, go back to the other page and press save.

Please note: The Benchmark gadget will attempt to gauge the speed of the selected Chunky2Planar converter. This has nothing to do with the speed of the voxel engine itself; the time taken by the converter is (roughly speaking) in addition to the time taken to render the frame! Bear in mind also that those converters which support region-restriction will be sped up considerably if the engine runs in a small window!