Wasteroids Help Contents

Introduction

Wasteroids (pronounced Waste-er-oids) is a Windows based asteroids game which features the following:

Fast moving 256 color graphics using WinG (well it's fast on my 66-DX2 PCI)
Realistic tumbling asteroids.
Interesting backdrops.
Some alien nasties to kill
Correct momentum calculations for bouncing off objects.
Various ship control mechanisms.
Fun demo mode, watch it.

I can be contacted at gsummers@physics.ox.ac.uk

Reasons for Writing Wasteroids

Primarily Wasteroids is a game to test the feasibility of producing decent games that run under Windows in native mode. It utilizes the WinG library to maximize the speed of copying data to the users screen. Off screen drawing into the WinG bitmap is performed by low level assembler routines (these routines are not particularly optimized at present).

Controls

The controls are user redefineable. The following represents the default settings.

Keyboard

'A'	:	Rotate Left
'S'	:	Rotate Right
'	:	Thrust
'/'	:	Retro thrust
<Return>	:	Fire weapon
<Space>	:	Shield

Mouse

The ship will rotate to face towards the mouse pointer, with these additional controls applying:

Left Button	:	Fire weapon
Right Button	:	Thrust
Middle Button	:	Shield

Joystick

Joystick Left	:	Rotate Left
Joystick Right	:	Rotate Right
Joystick Forward	:	Thrust
Joystick back	:	Retro thrust
Button 1	:	Fire Weapon
Button 2	:	Shield

Auto

The ship will fly under computer control. Note you can play against the computer using this option.

You can select the same control option for both player's ships, this can also be fun but rather pointless.

Additional keys

F2	:	Start One Player Game
F3	:	Start Tow Player Game
F4	:	Open Preferences dialog box
F5	:	Toggle Sound
'P'	:	Pause/Unpause
'Q'	:	Quit current Game

(The game will pause if you switch to another application, or open a dialog box.)

Future of Wasteroids

Wasteroids will most likely propagate to Win32.

Expect to see a massively enhanced version with hundreds of levels accessible by some kind of Galaxy map, loads of really nasty aliens, more lumps of rock, a plethora (yes plethora) of power up options and much, much more.

Some of the graphics are rather tacky at present, I'll run some through a raytracer I've just written to spruce things up a bit.

Eventually I might get WAVEMIX.DLL to work and the sound will become bearable.

A network version, gee that'll be fun.

Pronouncement

Pronounce it any way you like. I originally wanted a name that was a simple addition to Asteroids, but Atari got the only one I can think of that sounds any good (Blasteroids). Wasteroids=W(in)-asteroids, I suppose. If you have any suggestions please let me know. Also the name Waste-er-oids is meant to signify what a waste of time it has been writing this game, and what a waste of time it is for you to play it. I should be off doing my real job.

Wasteroid Internals

The program is written mostly in C++ plus a small amount of assembler. The game kernel took about 2 weeks to write and consists of about 5000 lines of code. I have also written several utility programs to facilitate the automatic generation of many game objects.

AST.DAT is a LZW compressed file container that is the receptacle for all the files necessary for running the game, including data files which describe the game objects, how they interact and a level database. In principal this file could be replaced and the entire nature of the game could be changed.

The program uses an off-screen WinG bitmap. Assembler code it utilized to draw into the off-screen bitmap, both bottom-up and top-down code has been written. WinGStretchBlt() is used with variable scale factor to copy rectangles which need updating to the screen.

Game objects are handled as C++ classes, more complex objects are derived from the simple base object class. Classes exist for special purposes like target acquisition, a object derived from these additional classes automatically gains its functionality. Multiple copies of these objects can be created without any chance of overwriting data. For an example watch the two ship auto play demo mode.

The asteroids were modeled by taking a number of points distributed about a sphere (typically 4096 for the large asteroids), and applying numerous craters and additional roughness to the surface. The 3D objects were then rendered and anti-aliased to create the 2D bitmaps. Other rotating objects were generated and anti-aliased from single bitmaps.

All objects have mass and this is used to calculate the momentum change when certain objects bounce off each other. Not all objects will bounce off each other as this would take too many collision checks and slow down the program.

How to Play

Press F2 to start a one player game.
Press F3 to start a two player game.

Use the controls to control your ship(s). Shoot the asteroids and aliens. When everything is dead you go on to the next level. The game is over when you run out of ships (or if the game runs out of asteroids, unlikely!)

When a new level starts the center of the screen is generally the safest place to be.

The ship's status indicator shows score, lives left and shield strength available. When the shield strength is used up it will become deactivated. While not in use the shield will recharge.

After you have died your next ship will wait until it is safe to materialize. This is quite a loose definition of safe. In general there will be room to appear but a rock might be about to hit you in the face, be ready with your shield! To force the ship to materialize while it is waiting for space, just press the shield button and you will appear with the shield on. What will happen if you appear within an asteroid is not known.

WinG

WinG is Microsoft's high performance graphics library enabling bandwidth limited memory transfers to the display.

Notes on Performance

The speed of the program is likely to be limited by the time taken by WinG copying regions of the off screen buffer to display memory. It is thus essential to have WinG correctly installed on your system and to be running in a suitable graphics mode (256 colors).

For local bus graphics systems (32 bit, 25 or 33MHz) the program should run quite smoothly. However, slower ISA graphics cards (16 bit, 8 or 11MHz) could cause the program to run very slowly. For this reason I have allowed the number of objects to be modified (this is a number which effects how many asteroids will be created on each level). The default value is 8, setting this value to 4, for example, would approximately double the speed of the game. Also the overall scaling factor applied when copying the image to the screen can be modified, the default value is 2 but by setting this to 1 the game will run significantly faster.

For slower machines I suggest you use the lowest resolution 256 color screen mode available (e.g. 640x480), and maybe use a scale factor of 1 (this will however make the playing area appear very small).

For faster machines any 256 color mode should be sufficient.

If WinG seems to be running sluggishly, check that the line: device=dva.386 appears in the [386 Enhanced] section of your system.ini.