3D PRECISION

BY

ARNO HASNAES

DOCUMENTATION BY

Dr. HELMUT AIGNER & Dr. ANDY AIGNER

Published by F. A. Vachha for DP, 222 The Avenue, London E4 9SE

C O N T E N T S

1. INTRODUCTION

1.1 What Does 3D Precision Do for You?

Thank you for buying 3D PRECISION.

3D PRECISION is a suite of programs which permits the creation and rapid manipulation (i.e. size changes as well as linear and rotary movement in space) of three-dimensional objects in any colour or combination of colours on the QL and Thor computers. It comprises:

- a 3D Editor (treated in detail in Chapter 2) that will let you define, test, edit and save objects under menu and cursor control if your computer has at least some extra memory;
- a SuperBASIC Toolkit (Chapter 3), which adds dozens of new commands to the SuperBASIC language, allowing you easy manipulation of objects if you are at all proficient in that language;
- an Assembler Toolkit (Chapter 4), offering many subroutines for superfast manipulation of objects - all accessible to your programs written in assembler code (you supply the assembly-language know-how).

Your objects, together with an imaginary camera which views them, are placed in a three-dimensional 'world' with integer x, y and z co-ordinates ranging from -32768 to 32767. (In practice you may be slightly more restricted, as no object must be more than some 32000 'world units' away from the camera. Lines should also be shorter than 16000 units and object dimensions less than 15000 units, or you may get lines going beyond their intended endpoints or deformed or disappearing objects.)

There are a few further restrictions:

- Objects are of the wire-frame type, i.e. composed exclusively of lines and points. They are viewed as though anything but the lines and points were transparent; in other words, there is no hidden line removal (unless you provide for it in your own program). You may, however, define for each object a cut-off plane ('backplane') parallel to the screen, which will make everything on its off-camera side disappear.

- Curves on the surface of objects (e.g. on cylinders, cones, and spheres) must be approximated by line segments. (The SuperBASIC CIRCLE command does nothing else; but in 3D PRECISION you have the option of specifying the number of corners to be used in the approximation.)

You may save individual objects or whole worlds on microdrive/disk and reload them singly or in combination.

You may also reconfigure nearly all of the default values that the program suite uses on start-up.

Please do not be put off by the technical appearance of this manual. Many parts of it are intended for use by BASIC and machine code programmers - if all you want to do is design and manipulate objects in 2D or 3D, ignore sections 3 and 4 of the manual.

While on the subject of the manual, Digital Precision is happy to acknowledge the skill and expertise of Dr Helmut Aigner, who designed and wrote this treatise virtually single-handedly. 3D Precision is one of those rare programs whose production involved technical skills from more than one country (Denmark, Austria & the UK). So did Professional Astrologer, incidentally - Germany, India, Japan & the UK!

In all, your master medium(s) comprises the following files:

Part 1	DEFAULTS_DEF3D	Defaults for 3D_PRECISION_BIN - preset to the values of DEFCM_DEF3D.
	BOOT	LRUN this to start Editor.
	BOOT2	This is LRUN by BOOT and will EXECUTE 3D_PRECISION_BIN.
	CLONE_BIN	EXEC this to make a working copy (see Section 1.3).
	DEFTV8_DEF3D	Copy this to DEFAULTS_DEF3D if you use a TV in Mode 8 for display.
	DEFTV4_DEF3D	Copy this to DEFAULTS_DEF3D if you use a TV in Mode 4 for display.
	DEFBW_DEF3D	Copy this to DEFAULTS_DEF3D if you use a black-and-white monitor for display.
	DEFCM_DEF3D	Copy this to DEFAULTS_DEF3D if you use a colour monitor for display.
	GL_PRT	A plotter driver for plotters that use the GL plotter language. (set plotter for ISO, A4 paper)
	GL_BIN	GL_PRT compiled by TURBO. Same as PLOTTER_BIN.
	PLOTTER_BIN	This is called by 3D_PRECISION when PLOT PIC. is used.
	3D_TOOLKIT_BIN	The BASIC Toolkit (21128 bytes).
	RUNTIME_EXTS	Toolkit used by BOOT.
	3D_TOOLKITASM_BIN	The Toolkit without the BASIC definitions (13472 bytes).
	HELPFILE	The Help file; you may add notes, using an Editor, if you like.
Part 2	3D_PRECISION_BIN	The 3D Object Editor.
Part 3	BASICDEMO	A BASIC program that demonstrates the use of the BASIC Toolkit.
	BASICDEMOBOOT	LRUN this to start BASICDEMO.
	BASICEXT	An Assembler file (METACOMCO) which demonstrates how to make further BASIC extensions. It also shows the use of DRAWTYPE.
	BASICEXT_BIN	BASICEXT in machine code.
	BASICEXTDEMO	A BASIC prog to demo BASICEXT.
	BASICEXTDEMOBOOT	LRUN this to see a demonstration.
	ASMDEMO	An Assembler file (METACOMCO) which shows how to make Assembler progs.
	ASMDEMOMDV_BIN	ASMDEMO in machine code; uses MDV2_.
	ASMDEMOFLP_BIN	ASMDEMO in machine code; uses FLP1_.
	ASMDEMOMDVBOOT	LRUN this to see ASMDEMO (if on microdrive) run.
	ASMDEMOFLPBOOT	LRUN this to see ASMDEMO (if on floppy disk) run.
	DRAWTYPE	Draw routine for making points in different sizes.
	DRAWTYPE_BIN	DRAWTYPE in machine code.
	BASICDEMO_ALL3D	Objects for demonstrations.
	BASICDEMO1_CAM3D	"
	BASICDEMO2_OBJ3D	"
	BASICDEMO3_OBJ3D	"
	BASICDEMO4_OBJ3D	"
	BASICDEMO5_OBJ3D	"
	SPACE_OBJ3D	"
	LANDER_OBJ3D	"
	STATION_OBJ3D	"

.2 Definitions

n all the following chapters you will come across the following erms:

A WORLD is a group of three-dimensional objects (or a single object if you insist) that can be resident in memory at the same time and saved and loaded as a whole. The definition of a world also includes the location of the three axes and of the camera.

An OBJECT is any combination of lines and points that you define as belonging together. You may create or place an object anywhere in a world, move it, enlarge or reduce it, rotate it, recolour it, and delete it.

The CAMERA is invisible but behaves like an object in many respects. It always exists in any given world, but it can be moved and rotated. This will produce an apparent motion or size change of the objects on the screen: moving or turning the camera in any direction will make the whole world (i.e. all objects in it) appear to move or turn in the opposite direction. This is of course as it should be but still takes some getting used to.

CO-ORDINATES are a set of three numbers (named x, y and z) used to describe locations in three-dimensional space. In the WORLD CO-ORDINATES used in 3D PRECISION, the greater the x co-ordinate of a point, the farther right it will be located on the screen. The greater the y co-ordinate of a point, the higher it will be located on the screen - at least if you forgo rotating the camera about a line parallel to the z axis. Increasing the z co-ordinate of a point will have no apparent effect on the screen (except for slight changes caused by perspective) until you rotate the object or the camera.

The GRID is an invisible pattern of gridpoints in the world which have the same mutual distance in all directions. This distance can be set by you, or you may accept the default; you may then move the cursor to the nearest gridpoint at any time.

A POINT is a dot, defined by a triplet of x, y and z co-ordinates (e.g. 5,-10,20) and drawn on the screen as part of an object.

A LINE is, strictly speaking, a line segment (as in SuperBASIC). It is defined by two triplets of co-ordinates for its endpoints and drawn on the screen as part of an object. The endpoints of a line are not considered to be POINTS as defined above.

An AXIS is one of three invisible lines defined in every world. The endpoints of the x axis have co-ordinates (-32768,0,0) and (32767,0,0), of the y axis have co-ordinates (0,-32768,0) and (0,32767,0) and of the z axis have co-ordinates (0,0,-32768) and (0,0,32767).

A VERTEX (plural: vertices) is an invisible point which may serve as a location for a visible point, for one of the two end points of a line, or for the intersection of several lines.

ample: A wire-frame cube consists of 12 lines, 8 vertices and no ints. (Remember: points must be isolated, not part of a line.)

The CENTRE of an object is a reference point from which all the vertices of the object are calculated. If you change the centre of an object, the whole object is recalculated. If you rotate an object, the axis of rotation (parallel to the x, y or z axis as specified by you) passes through its centre. All rotation angles must be expressed in degrees, not in radians as in SuperBASIC.

1.3 Making a Backup Copy - What Goes Where

Before you run any of the programs, you MUST make a backup copy by running the CLONE program. Murphy's Law is insistent on loss of files happening to YOU sooner or later if you fail to do this.

Follow the instructions given here and on the screen. Make sure that your destination medium is not write-protected and that it contains no files bearing any of the names listed in Section 1.1 (they would be overwritten) - if you are using cartridges, they should be entirely blank as there is no room on them for anything other than the 3D Precision files.

The clone program has be designed to be able to make a copy from (and/or to) disk or cartridge. In order to do this the copying has been split into 3 parts: each part can fit onto a cartridge that gives at least 212 sectors on formatting (or a 212 sector ramdisk). If you are using ramdisk as an intermediate stage for copying to floppy disk, it is most convenient to format just one ramdisk, but with at least 636 sectors so it can hold all three parts of the system. If you are using ramdisk as an intermediate stage for copying to cartridge, it is best to format 3 ramdisks (ram1_, ram2_ and ram3_) with at least 212 sectors each.

If you have the Supertoolkit command WCOPY available you can bypass our backing up process and do a direct WCOPY: there is no messy copy protection system waiting to cause trouble for you!

First ensure that you have formatted the target medium(s).

The clone program is invoked by entering:

```
EXEC device_CLONE_BIN
```

where device is MDV1_, FLP1_ or whatever.

Press CTRL + C if on the QL so that the onscreen cursor flashes.

Now you are asked to enter the source (copy from), target (copy to) and the drive which is ultimately going to hold the copy (copy to run from) for part 1 of the copying.

Below are some examples of what you might enter in reply to the questions for part 1:

WHAT YOU HAVE AVAILABLE:	1 DRIVE	2 DRIVES	1 MDV	2 MDVS
Copy from	flp1_	flp1_	mdv1_	mdv1_
Copy to	ram1_	flp2_	ram1_	mdv2_
Copy to run from	flp1_	flp1_	mdv1_	mdv1_

Note how ramdisk may be used as an 'intermediate' storage space if you do have only one disk drive or one microdrive (as opposed to one microcartridge!).

If you are making a copy to run from disk, repeat these answers for parts 2 and 3.

:f you are making a copy to run from cartridge, the corresponding ınswers for both parts 2 and 3 will be:

Copy from	mdv1_	mdv1_
Copy to	ram1_	mdv2_
Copy to run from	mdv2_	mdv2_

lote the change between part 1 and parts 2/3 for microdrives. It is .ecessary because cartridge 1 is intended to be run from drive 1 while ›oth the others are intended to be run from drive 2.

ow you are asked to enter the name of the default file you want to se:

Filename	For use with
DEFCM	COLOUR MONITOR
DEFBW	MONOCHROME MONITOR
DEFTV8	TV 8 COLOUR MODE
DEFTV4	TV 4 COLOUR MODE (MORE ROOM IN WINDOWS)

nsure the source medium (if the source is cartridge, this means artridge 1) is in the drive you specified as the source drive for art 1. Ensure that it is not write protected, as we will be adjusting he default file on it. Press a key, and the DEFAULT_DEF3D file will ow be set to the one you have just selected.

nce you have answered the above questions the copying process will tart, pausing at the start of each part, where you will be asked to ress a key when ready. Before you answer each part you must ensure nat the source and target media appropriate for that part are present ı the drives you specified.

ıe copying may take a while - especially on cartridges. Pray be ıtient.

ıen the backup is completed, put the master medium away in a safe lace and use only the backup medium (except for making a new backup : something nasty happens to your working copy).

se the following instructions to run the programs.

› run 3D_PRECISION_BIN enter:

```
LRUN FLP1_BOOT
```

· if on microdrive, insert cartridge 1 in MDV1_ and cartridge 2 in)V2_, and enter:

```
LRUN MDV1_BOOT
```

To run the demonstrations enter:

```
LRUN FLP1_BASICDEMOBOOT
```
or
```
LRUN FLP1_BASICEXTDEMOBOOT
```
or
```
LRUN FLP1_ASMDEMOFLPBOOT
```
or
```
LRUN FLP1_BASICDEMOBOOT
```

or if on microdrive, insert cartridge 1 in MDV1_ and cartridge 3 in MDV2_ , and enter:

```
LRUN MDV2_BASICDEMOBOOT
```
or
```
LRUN MDV2_BASICEXTDEMOBOOT
```
or
```
LRUN MDV2_ASMDEMOMDVBOOT
```
or
```
LRUN MDV2_BASICDEMOBOOT
```

The computer should be reset before running a demonstration.

You have noticed that there is no copy protection on 3D PRECISION. Against the trend on top quality products, we have taken a gamble on your honesty. Please don't let us down. Only if you do distribute copies do we have to track you down and take - or get the authorities to take - whatever action is required under civil and criminal law. We still offer rewards for information which helps us to track down pirates. Please realise we would far prefer to be able to spend all our time more productively!

If you somehow find yourself in the possession of a pirated copy of 3D PRECISION, then please do the honest thing: order a legitimate copy. This will entitle you to software and documentation upgrades, and support from us if you run into problems.

Please note that we cannot undertake to answer technical queries on 3D PRECISION over the phone: please put your queries into writing. Often you will find that the very act of putting ink to paper (or fingers to keyboard) concentrates the mind wonderfully, and the "problem" vanishes. About 90% of telephone calls to us result from either not making a backup copy (if you screw up while making a backup, don't call us - just return the original medium to us with an SAE - we can't unscramble cartridges over the phone as yet!) or from not reading the manual (have you used the index or contents pages?). We are on your side - so please help us to help you.

If we have anything to add to this manual, you will find it on a file called UPDATES_DOC, which is intended to be read with Quill. This file will be present on cartridge 3 if you have ordered 3D PRECISION on cartridge. Please read it.

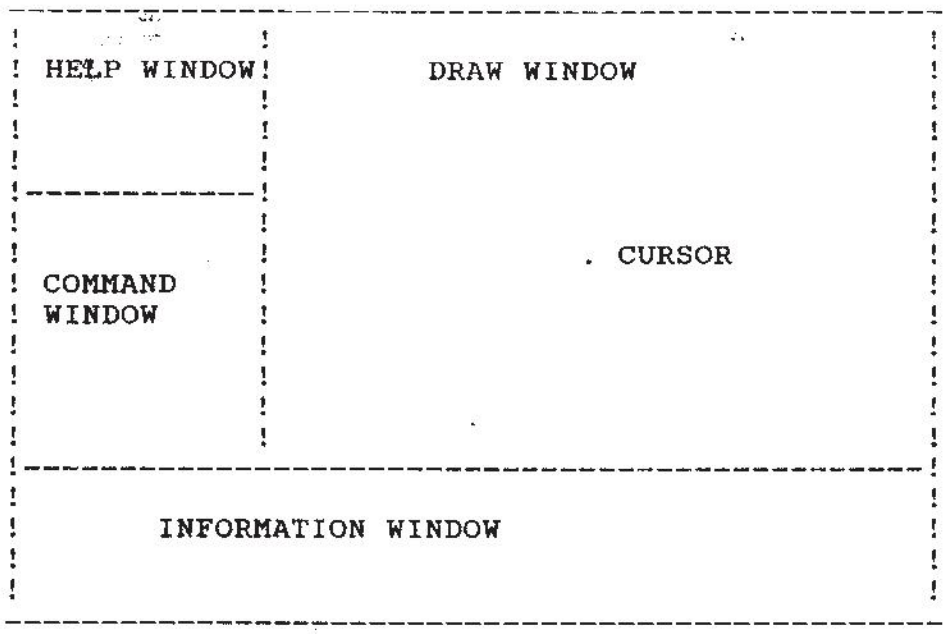
. USING THE 3D OBJECT EDITOR

.1 Screen Windows

› run the editor, insert cartridges 1 and 2 in the appropriate slots, ˙ the disk in slot 1, and type:

LRUN FLP1_BOOT or LRUN MDV1_BOOT

›u will then see the windows below. You must now select the program ˙ pressing CTRL C on the QL; no action is required on the Thor.



e names of the windows are for reference only; they do indicate eir main use, but part of the time the windows may be used for other ings.

HELP WINDOW:

Shows the current step sizes (in world units) for moving (MOVE) and rotating (TURN) objects.

INFORMATION WINDOW:

Shows the positions and rotation angles of the camera, cursor, points and lines and of any objects currently in memory, followed (if there is enough room left in the window) by the amount of memory used (the plot area is part of the work area); and the amount of space allocated can be changed with EDIT DEF - see below). For a detailed description of memory usage see 4.3.1.

2.2 Control Keys

When the command window contains a menu of commands, the following keys will have special functions:

←→↓↑F2F1	Changes cursor co-ordinates (x-+ y-+ z-+)
CTRL + ←→↓↑F2F1	Rotates cursor (if cursor is on an object or on the camera)
ALT + ↓↑	Changes move step (-+)
CTRL + ALT + ↓↑	Changes rotate step (-+)
SHIFT + F4F5	If there is no room in the information window, this will scroll the list of objects. This also works when you choose a work object and the cursor. The keys are also used to control Box and Size.
F4 F5	Selects command (-+)
SPACE, ENTER	Executes command
ESC	Usually takes you back to where you came from
F3	Updates draw window
CTRL + F3	Changes screen mode
CTRL + F4	Changes default cursor colour
SHIFT + F1	Makes the draw window fill the whole screen
SHIFT + F2	Makes the computer ask where to put the cursor
SHIFT + F3	Shows camera views from all 6 possible directions. The direction number is shown before the work area.
CTRL + 1	Moves the cursor to the closest vertex in the work object (if it is closer than the move step)
CTRL + 2	Makes the cursor jump to the nearest point on the grid
CTRL + CAPSLOCK	Shows the Help pages in the draw window

You needn't learn all these keys now; you can always come back to this list when you need it.

The Editor cannot multitask, but provided that you exit through STOP PROGRAM (see later), it can be restarted by:

EXECUTE MDV2_3D_PRECISION_BIN or EXECUTE FLP1_3D_PRECISION_BIN

Note that from within the program an EXECUTE command is used to start the plotter driver.

3.3 An Example

3.3.1 DEFINE

First let's define an object. Use the F4 and F5 keys and press SPACE (or ENTER) to select DEFINE. You will be asked for the object name: type 'A' and press ENTER. (The program will print everything in upper case.) Now you must decide how complex an object you are going to define:

```
OBJECT:
A
VTCS:10      (number of vertices)
PNTS:10      (number of points)
LINES10      (number of lines)
```

Remember that you must use one vertex for each isolated point and for each endpoint of a line; but if the same triplet of co-ordinates is to be used several times, a single vertex will do.

If you just press ENTER, the default (0) will be assumed. If you want to change any value, use the usual keys (←→↓↑CTRL←→). When you have answered the last question, (LINE), you will be returned to the main menu. Try to define a new object with the same name - it will show the definition of the object and the memory used by the definition. On the QL you can use about 100 to 200 lines before the program gets too slow, depending on the speed of your expansion hardware and, of course, on the length of the lines drawn on the screen.

3.3.2 INSERT

So far our object has a name (A), and memory is reserved for it, but it is still empty. Now we will 'insert' something into it. The default is a point: select INSERT, using the F4 F5 keys, and press SPACE. You will now see a list of colours. Select the ink colour you want, using the F4 F5 keys, and press SPACE. You have now inserted a point into the object, which will become visible as you move the cursor away with the ←→keys. As you will see presently, it is also possible to insert whole objects.

2.3.3 CURSOR

What about lines? Well, select CURSOR (F4 F5). You will see the following list:

CAMERA
POINT
LINE
A

Yes - 'A' is the object you have defined before, but it will have to wait. Select LINE (F4 F5), then select Insert. When you now move the cursor (←→↓↑ F2 F1), you will see a ('rubber') line going from the previous position to the current cursor position. When you select Insert again, you will be asked to choose the colour of the line. This question is asked twice, not because the program is deaf but because lines can have stipples, though only in standard drawing mode. As it happens, the Editor is loaded as a default in fast drawing mode, and you will have to perform some minor contortions to get into standard drawing mode for stipples. There is an error in the way MG ROM QLs handle points. Determine which QL version you have (PRINT VER$ from SuperBASIC accomplishes this): if it is an MG you must enter GDRAWTYPE 1,0 . If it is any other version (AH, JM or JS - if you have any other ROM, you are in real trouble!) then enter GDRAWTYPE 1,1 instead. The MG ROM point error can cause the 3D_PRECISION editor to stop with an error message if a point is plotted. If you then choose two different colours, you must also select the stipple pattern to be used. If you want a solid colour - or if you stayed in fast drawing mode - just select the same colour again.

2.3.4 OBJECT

First define a new object called 'B'. When you then select OBJECT, you will see:

A
B

Select 'B'. B is now the object you work on. Then change the cursor: select CURSOR and the 'A' object. You can now move the object with the cursor control keys. Try also to rotate it by holding the CTRL key down and then pressing one of the cursor control keys. Now select INSERT: You will be asked to select colours or to press ENTER (instead of SPACE). This can be used to make copies of objects in different colours and to build a large object from smaller ones. Just press ENTER for now and move the object away. Press F3 to update the world. If there are too few vertices, points or lines, an error message (or several) will be displayed on a red background in the information window. Select CAMERA under CURSOR. You can now move and rotate the camera.

2.3.5 REMOVE ONE

Select REMOVE, type 'A' and press ENTER. You will observe that the object 'A' is removed and that the 'B' object takes its place in the information window. Repeat this with 'B', and you have removed both objects you had defined. You could of course have done this by using the REMOVE MULtiple command in 2.4.2. If there are many and/or large objects after the last object to delete, the command will take a little time to be executed.

.4 Further Menu Commands

ou may read what follows as needed. It is organised on the principle f indentation of sub-menus. Some of the headings are abbreviated on he screen.

.4.1 FILE I/O

Five different file types are used:

_OBJ3D	an object.
_ALL3D	a small file containing the number of cameras (always 1) and the number of objects saved as a world under this name by SAVE ALL.
_CAM3D	the camera definition saved by SAVE ALL.
_DEF3D	A file containing the defaults used by this program.
_PIC	A common screen file.

Some of these are discussed below under DEF-I/O and PIC-I/O. Only the name, i.e. without the device name (e.g. flp1_, mdv1_) and the above extensions, must be typed in answer to prompts generated by the following commands. You may want to change the device before trying this. (See DEFAULT EDIT)

LOAD OBJECT

First you will be asked for the file name of the object and then for its name when loaded. If the former cannot be found, the usual error message will be displayed.

SAVE OBJECT

This is the opposite of load. You are asked first for the name of the object and then for the file name. If a file with the same name (and extension) exists, you are asked whether to replace or not (type Y or N and press ENTER).

DIR OBJECT

This shows the filenames of all objects (_OBJ3D) on the default device. If you get tired of waiting, you can stop the listing by pressing ESC.

DELETE OBJECT

This will ask you for the name (remember, the name only) of the file you want to delete.

LOAD ALL

This will load a whole world at once, provided it has been saved by SAVE ALL. If some objects have already been defined or loaded you are asked if you want to delete them (Y/N).

SAVE ALL

If you have just made 30 objects and want to save, you needn't wear out your fingers by 30 SAVE OBJECT commands. Just type the 'world' name, and all objects, as well as the camera position, will be saved under that name. You can then reload them all by LOAD ALL, or use load LOAD OBJECT to load objects one by one under the world name with the object's number at the end; e.g. 'BASICDEMO2'.

DIR ALL

Displays all the world names (_ALL) on the device. If you get tired of waiting, you can stop the listing by pressing ESC.

DELETE ALL

Use this to delete a world saved by SAVE ALL.

2.4.2 OBJECT CONTROL

This menu contains functions which work on the object selected by OBJECT.

CHANGE VERTEX

This can be used to change a vertex's co-ordinates. You must first select the vertex (shown by a flashing dot) to change. The information window will show the number of the vertex and its x, y, z co-ordinates. After that you can move the vertex with the usual keys. If the vertex is shared by several lines, these lines will follow the vertex.

VERTEX CLEAR

When you have deleted some points or lines, you can delete their vertices. If you try to delete a vertex which is still in use, an IN USE error message will be displayed.

POINT INK

Select the point to change (F4 F5, SPACE). The point is 'highlighted' (!) by changing its colour to black. Then select the new ink colour (as for INSERTing points). You must select another point before you can see the change. In the information window you can see the colour of the point and the vertex it uses.

POINT CLEAR

Select the point to clear (F4 F5), press SPACE, and the point will be removed. In the information window you can see what used to be the colour of the point and the vertex it used.

LINE INK

Select the line to change (F4 F5,SPACE). The line will be 'shown' by disappearing. Then select the ink colour (as for INSERTing lines). You must select another line before you can see the change. In the information window you can see the colour of line and the vertices it uses.

LINE CLEAR

Select the line to clear (F4 F5), press SPACE, and the line is removed. In the information window you can see what used to be the colour of the line and the vertices it used.

REMOVE MUL

It works as REMOVE ONE except, that it will remove multiple objects. You must type in the identifier for the first and the last object that you want to delete: see the list of objects in the information window. If there are many and/or large objects after the last object to delete, it will take some time.

BOX

First select the colour of the box. Then you can select the box size along the x axis by F4 F5. You can select other axes by SHIFT + F4 F5 (you can only select z when x > 0 and y > 0). You can change the position and rotation of a box with the usual keys: (CTRL +←→↓↑F2F1).

CIRCLE

First you must decide whether you want the circle to consist of points or of lines (P/L). Then you must select its colour (as in INSERT point or line). Now, how many vertices (points or lines) should be used? The more vertices, the rounder the circle will become (there must be at least 5). You can now move, rotate and change size (SHIFT←→↓↑) until you are satisfied, and then press SPACE.

SIZE

You can use this to change the size of an object. Pressing F4 F5 will change the integer part of the step factor, and SHIFT F4 F5 the fractional part. Now comes the fun: using the usual move keys, you can change the size along the different axes. If you don't want to wreck your object, press ESC; if it's all right, press SPACE.

CENTRE

When you have selected this function, any movement of the cursor keys will affect the reference point from which all vertices of the current object are calculated.

MIRROR

This will make the negative x (or y, or z - it's your option) co-ordinates of all or selected vertices of an object positive, and positive ones negative. Remember, all co-ordinates are defined with respect to the centre of the object - so if you want to shift the mirroring plane, use CHANGE CENTRE first. You can choose whether to mirror from the current cursor position or from the work object (but mirroring will always affect the work object). You will only be prompted if necessary, i.e., if there is a cursor object and the cursor and work objects are not the same. Then you must decide about which axis to mirror and whether you want to 'mirror all' or choose the points or lines to mirror (F4/F5, followed by SPACE/ENTER). The old vertices disappear as their mirror images are formed; there is no way to have an object and its mirror image on the screen at the same time.

BACKPLANE

This is used to set a z co-ordinate behind which any lines and points will disappear. If you want to remove the backplane, press ESC.

2.4.3 PROGRAM CONTROL

This menu contains facilities to change the defaults in the DEFAULTS_DEF3D file and to save the screen.

The defaults are as follows:

ISL The crosses in the columns at left indicate whether these defaults are effective immediately, after START, or after LOAD DEFAULTS.

XXX	STORAGE DEVICE.....FLP1_	Device used in LOAD, SAVE, DIR, DELETE.
XXX	INSERT IN EDIT(1/0).....1	Mode used when editing text (Insert / Overwrite).
XXX	OVER MODE(-1/0).....0	Mode used when plotting objects (XOR or OVER).
XXX	GRID.....10	Distance between gridpoints. When you press the 'TO GRID' key, the cursor will be moved to the nearest gridpoint.
XXX	PLOTTER DEVICE.....SER1	The device the plotter is connected to or a filename to direct output to file.
XX	MODE.....8	Screen mode 4 or 8. If set to 0, the current mode is used.
X	NO. OF OBJECTS.....30	The maximum number of objects that can be loaded. The theoretical maximum is 1024, but there is not enough room in the dataspace of the Editor, which is a Turbo-CHARGED SuperBASIC program.
X	SIZE OF WORK AREA.....4226	The work area used by the Editor.
X	SIZE OF PLOT AREA.....16	The plot area is part of the work area; it is used for the plot lists.
X	LOAD ALL ON START.....	The objects saved under this name are loaded.
XX	CURSOR COLOUR.....7	Colour of the cursor.
X	CAMERA POS. X.....0	Camera position.
X	Y.....0	
X	Z.....-500	
X	CAMERA ANGLE X.....0	Camera angle.
X	Y.....0	
X	Z.....0	
X	SCALE YL.....192	Height of the screen window.
X	ZL.....288	Distance into the screen corresponding to YL.
X	SCREEN FACTOR.....17128	Change this if a circle is not round.
XX	MOVE STEP.....50	Cursor move step.
XX	ROTATE STEP.....10	Rotate step.
XXX	ERROR INK.....7	Colour of error messages.
XXX	ERROR PAPER.....2	

EDIT DEFAULTS

When you select this, you will see a list of parameters in the information window. The changes will first be effective on restart or when you use LOAD DEFAULTS, except for the first four defaults: device, insert/overwrite, XOR mode and grid, which will be changed immediately). If you enter an illegal value, it will be changed to an acceptable value on exit (by ESC).

WINDOW

First you must select which window to change (F4 F5). Then if you are in a mode 4, you can change the size of characters (and the window) by SHIFT ←→. To move the window, use the arrows; to change the size of the window, ALT and the arrows; to change size in steps of one character, CTRL and the arrows. When finished, press ENTER; you will be prompted for changing the colour of the border, of the background and of the ink for the window. You can press ESC if you don't want any changes.

SCALE

This is used to change the height of the window (YL) and its equivalent into the screen (ZL). If a circle does not look round on you monitor or TV, you should change the x axis (SF for Screen Factor).

WARNING:
DO NOT EXPERIMENT WITH ANY EXTREME COMBINATIONS OF YL,ZL,SF. There is an ever so slight chance that you may crash the program.

KEYBOARD

You can change the function of keys by altering this list of ASCII values. The list will be displayed in the draw window, and editing will work as for EDIT DEFAULTS. Changes will first be effective after a restart or a LOAD DEFAULTS.

DEFAULT-I/O

This should be very easy to follow.

LOAD DEFAULTS

Type the name of a _DEF3D file, e.g. DEFTV8, and press ENTER. You will now see the window change. Some of the above defaults will also be changed.

WARNING: The program cannot recover from a 'bad or changed medium' error in LOAD DEFAULTS.

SAVE DEFAULTS

Type the name of the file, e.g. DEFAULTS, and press ENTER. If a file of that name already exists, you will be asked whether you want to replace the file (Y/N). When the program is (re)started (or LOAD DEFAULTS selected), the windows (and some defaults) will be changed to the settings in force when they were saved by SAVE DEFAULTS (see above).

DIR DEFAULTS

Lists the _DEF3D files on the default device. If you get tired of waiting, you can stop the listing by pressing ESC.

DELETE DEFAULTS

If you enter the name of a _DEF3D file, it'll be deleted.

PICTURE-I/O

PLOT PICTURE

You are asked whether you want to plot all or only one object (type A/O). If you type 'O', you are asked to type in the identifier for the object you want to plot. The work object is suggested as default. Then you are asked which form the colour information must have. You can choose between Normal, B&W and Convert colours (N/B/C). The default is normal which does nothing. B&W will change all colours to black = 0. Convert colours will convert whatever colours you want. You are asked to type the number of the colour to convert, and the number of the colour it must be converted to. You can repeat until all colours are specified to your satisfaction. Press enter when you are through. You may then type in a label for the drawing. When you press ENTER it will start plotting. You can stop by pressing ESC. You must have inserted a disk or tape in the memory device, with room for the PLOT3D file.

SAVE PICTURE

First you are asked for the filename; then whether you want 'this screen' (which means all the windows) or the draw window to fill all the screen. If you choose 'this screen', you must decide which of the 8 menu items you want. If the file name already exists, you will be asked whether you want to replace it (Y/N).

DIR PICTURES

Lists the _PIC files on the default device. If you get fed up, you can stop the listing by pressing ESC.

DELETE PICTURE

This deletes a _PIC file on the default device.

DIR

Lists all files on the default device. If you get tired of waiting, you can stop the listing by pressing ESC.

DELETE

This will delete any file on the default device.

STOP PROGRAM

This will end the program, i.e. remove it from memory.
You can restart it with:

EXECUTE MDV2_3D_PRECISION_BIN

or

EXECUTE FLP1_3D_PRECISION_BIN

PROGRAMMING WITH THE BASIC TOOLKIT

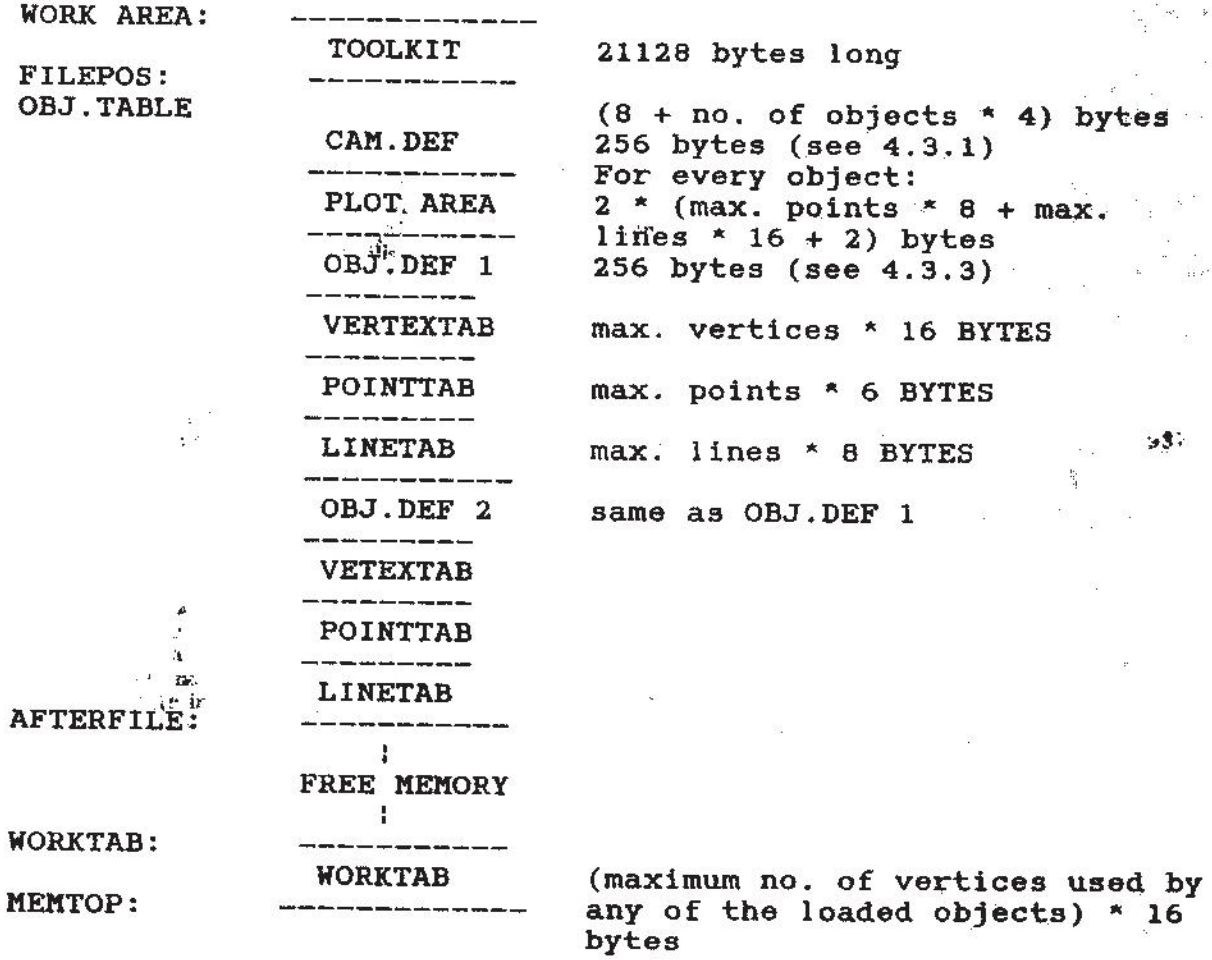
is chapter does not assume that you have read Chapter 2; thus some formation is the same. You are, however, assumed to know perBASIC. To use the Toolkit most effectively, you should also be epared to learn about some of its inner workings.

1 How to Go About It

fore you can use the BASIC Toolkit, you must load it - and at the me time reserve workspace - with a short SuperBASIC program along e following lines:

```
100 workarea = 40                          : REMark set workarea = 40 K
110 a = RESPR(workarea * 1024)             : REMark reserve area
120 LBYTES flp1_3d_toolkit_bin, a          : REMark load BASIC toolkit
130 CALL a,a+workarea*1024                 : REMark set memtop, link in
140                                        : REMark BASIC keywords
```

e only parameters that may change in this program depending on your quirements are the value of workarea and the device from which you ad the toolkit. Normally it is advisable to reserve a lot of rkspace. If space is at a premium, the following memory map may lp you to calculate exact memory requirements. All space shown on is map is by default placed after the toolkit, but can also be located in QDOS' common heap - see 4.1.



n you have run the above program, you are ready to use the new erBASIC extensions in your own programs. (The demonstration gram BASICDEMO has its own boot program, BASICDEMOBOOT, to take e of the initialisation.) Let's look at an even shorter sample gram that will show us the most important features:

```
200 GSTART 3,5                                  : REMark start system:
202                                             : REMark plot area = 3 K;
205                                             : REMark max. 5 objects
210 GWINDOW 512,256,0,0,256,512,17128           : REMark open window over
212                                             : REMark whole screen and
215                                             : REMark set YL, ZL, SF
218                                             : REMark (screen factor)
220 GCLS                                        : REMark clear the screen
230 GPLACE 0,200,0                              : REMark place the camera
235                                             : REMark (only 3 params.)
240 GLOADOBJ flp1_basicdemo2_obj3d              : REMark load object
250 FOR T=1 TO 90
260   GPLACE 1,0,0,7000-T*50                    : REMark place the object
270   GROTATE 1,0,T*4,0                         : REMark rotate the object
280   GMAKE : GCLEAR : GPLOT                    : REMark update the object
290 END FOR T
```

In order to understand the commands in the loop, it is important to know that for each vertex two sets of co-ordinates are stored (to prevent loss of precision): definitions and results; but only the latter are used directly for calculation and subsequent plotting. When an object is GROTATEd, new results are produced from the definitions, whereas GCHANGEOBJ duplicates the definitions into the results. Other commands, e.g. GSIZEOBJ and GCENTREOBJ, only change the definition, so that the changes they cause will not become evident until after the next GROTATE or GCHANGEOBJ.

In analogy to the two co-ordinate sets, the program also maintains two plotting lists for each object, a current list and a reserve list. GMAKE reads from the 'result' part of the vertex variables and places the values it has calculated in the current plot list. GPLOT plots from this list and swaps the two plot lists, so that the above values remain available in the reserve list while a subsequent GMAKE writes new values into the first list. This makes it possible to toggle between two states of an object without the need for recomputation.

In the above program, however, each set of computed values is used only once. GCLEAR makes the last image of the object disappear by replotting it in the background colour (from the reserve list, which is then used for next plot list), and GPLOT plots the next image from the current list. The first time through the loop, of course, the reserve list is still empty, so there is nothing for GCLEAR to act on.

There is a lot more to learn if you take the time to analyse the BASICDEMO program as a larger example of using the BASIC Toolkit.

The BASIC Toolkit is not re-entrant, i.e., only one program must use it at any time. Thus programs using it will not multitask, unless you can link the toolkit to a compiled program.

3.2 The New Keywords

FuNctions are marked as such. All other keywords refer to PROCedures.

Below are some names used frequently in the syntax descriptions. Any suitably-ranged numeric expression may be used in their place.

o:	1 TO 1024	Object number
v:	1 TO 32767	Vertex number
p:	1 TO 32767	Point number
l:	1 TO 32767	Line number
on/off:	0/1	Turn something on or off
col:	0 TO 255	Colour
	-32768 TO 32513	If the clear flag is set (bit 15)
x, y, z:	-32768 TO 32767	

GBACKPLANE o, on/off [, clipplane]

clipplane: -32768 TO 32767 Default for clipplane is zero

Turns backplane clipping of an object on or off and sets its clipping plane on the z axis.

GCAMADDR

A function that returns the address of the camera definitions (see 4.3.1).

GCAMCEN x, y, z

Sets the centre of rotation for the camera relative to its position.

GCAMCENX

A function that returns the x co-ordinate of the camera's centre of rotation relative to its position.

GCAMCENY

A function that returns the y co-ordinate of the camera's centre of rotation relative to its position.

GCAMCENZ

A function that returns the z co-ordinate of the camera's centre of rotation relative to its position.

GCAMX

A function that returns the x co-ordinate of the camera position.

GCAMXA

A function that returns the rotation angle of the camera about a line through its centre and parallel to the x axis.

GCAMY

A function that returns the y co-ordinate of the camera position.

GCAMYA

A function that returns the rotation angle of the camera about a line through its centre and parallel to the y axis.

GCAMZ

A function that returns the z co-ordinate of the camera position.

GCAMZA

A function that returns the rotation angle of the camera about a line through its centre and parallel to the z axis.

GCENTREOBJ o, x, y, z

Changes the centre of an object by adding x, y, z to the definitions. The change will first be effective after GROTATE or GCHANGEOBJ.

GCHANGEOBJ o

Moves the definitions to the results (i.e., the object is not rotated). If you have just changed the definition of an object (e.g. by GCENTRE, GSIZEOBJ) and you must plot the object without rotation, your changes will first be effective after this command. If rotation is required, use GROTATE.

GCLEAR [o [TO o2]]

Clears the object(s) by plotting them (from the reserve plotting list, which is subsequently cleared) in the background colour (or in their previous colour when in XOR mode; see GOVER). If two object numbers are given, the two objects and all objects with in-between numbers will be cleared. If no object is specified, all objects will be cleared.

GCLOSE

Closes the graphics window.

GCLS [col]

Clears the screen. If no colour is given, the last GCLS or paper colour is used.

GDATAADDR

A function that returns the address of certain parameters (see 4.3.8).

GDEFINEOBJ nv, np, nl

```
nv: 0 to 32767      Number of vertices
np: 0 to 32767      Number of points
nl: 0 to 32767      Number of lines
Maximum size depends on available memory.
```

Defines a new object.

GDRAWTYPE point, line

```
point: 0 to 1,( TO 32767)    0: QDOS
line:  0 to 1,( TO 32767)    1: new fast algorithm,
                                but no stipples
                          >=  2: reserved for your
                                own algorithms
```

Sets the algorithm for drawing. If you choose 2 or larger, you must have linked in your own assembly plotting routine. You can use values larger than 2 to mean something special; you can make a whole set of different drawing modes (see the DRAWTYPE file).

GERROR

A function that returns the number of a trapped error (see GTRAPERROR).

The following error codes may occur in addition to those of QDOS:

1 Work area used up
2 Plot area used up
3 Too many objects; maximum set by GSTART

GFACTOR sf

sf: 8000 to 32767 screen factor (default 17128)

Changes the x/y ratio of the screen by changing the screen size (in world units) in the x direction.

GFINDLINE (o, l, col, v1, v2, dummy)

dummy:	Must always be set to -1
l:	Line number from which to start the search
col:	Colour of the line (use -1 to search for a line of any colour)
v1,v2:	Numbers of the vertices the line is using (in any order; use -1 once or twice if one or both vertex numbers are to be ignored) in the search)

A function that returns the lowest number of a line conforming to the specifications given in the command, or, if no such line is found, zero.

GFINDMAX (o, v1, v2)

A function that returns the number of the vertex (ignoring vertex numbers lower than v1 or higher than v2) of an object which has the largest absolute value in any co-ordinate (x, y or z). This largest absolute value can then be found by GOBJMAX.

GFINDPOINT (o, p, col, v, dummy)

dummy:	Must always be set to -1
p:	Point number from which to start the search
col:	Colour of the point (use -1 to search for a point of any colour)
v:	Number of the vertex the point is using (use -1 if vertex numbers are to be ignored in the search)

A function that returns the lowest number of a point conforming to the specifications given in the command, or, if no such point is found, zero.

GFINDVERTEX (o, v, x, y, z, range)

range:	Search limits are +-range on x, y, z. If set to -32768, the whole world will be searched.
v:	Vertex number from which to start the search
x, y, z:	Vertex co-ordinates

A function that returns the number of a vertex or, if no vertex has been found, zero. If a range other than -32768 is specified, the search will be made in the rotated results; else in the definitions.

GINK o, col

col:	If col is -1, the old colours will return.

Sets the colour of an object.

GINSERTOBJ col, o1, o2

col:	If col is -1, the colours of o1 will be used.

Inserts object o1 in object o2. Duplicated points and lines are not inserted. If a positive error number results, it should be read as follows: bit 0 - too few vertices; bit 1 - too few points; bit 2 - too few lines.

GLINE col, xp, yp [, xp2, yp2]

It will plot a line using the algorithm set by GDRAWTYPE. Pixel co-ordinates are used when in fast line drawing mode. Graphic co-ordinates are used when in QDOS drawing mode. The values in the plotting lists are, of course, compatible with the requirements of this routine.

GLINE3D col, xp, yp, zp [, xp2, yp2, zp2]

It will plot a line in the 3D world. You specify its position just as an object (see GPLACE).

GLOADOBJ "filename"

filename:	Name of 'object' file to be loaded

Loads an object and assigns it the number given by (number of objects previously loaded +1).

GMAKE [o [TO o2]]

Calculates the object(s) from the results areas of its vertices and places the calculated values in the current plot list.

GMARKLINE o, l, [l2,] on/off

Marks or unmarks one or more lines. Marked lines will not be calculated during a GMAKE.

GMARKOBJ o [TO o2], on/off

Marks or unmarks one or more objects. GMAKE, GPLOT, GCLEAR and GSWAP will not take any action on marked objects.

GMARKPOINT o, p, [p2,] on/off

Marks or unmarks one or more points. Marked points will not be calculated during a GMAKE.

GMARKVERTEX o, v, [v2,] on/off

Marks or unmarks one or more vertices. Marked vertices will not be calculated during a GMAKE.

GMODE mno

mno: 4/8 mode number

Sets the screen mode in the Toolkit. This command has to be preceded by the analogous SuperBASIC command, thus:
MODE 4 : GMODE 4

GMOVE [o,] forward/backward, right/left, up/down

Moves the object by the number of world units specified along each of the three axes. If no o is specified, the camera is moved as if it were an object.

GNOOBJ

A function that returns the number of objects in memory.

GNOOBJMAX

A function that returns the maximum number of objects as defined by GSTART.

GOBJADDR (o)

A function that returns the address of the object definitions (see 4.3.3).

GOBJMAX

A function that returns the maximum absolute value found by the most recently executed GFINDMAX.

GOBJX (o)

A function that returns the x co-ordinate of the centre of the object with respect to the camera position.

GOBJXA (o)

A function that returns the angle of rotation of the object about a line through its centre and parallel to the x axis.

GOBJY (o)

A function that returns the y co-ordinate of the centre of the object with respect to the camera position.

GOBJYA (o)

A function that returns the angle of rotation of the object about a line through its centre and parallel to the y axis.

GOBJZ (o)

A function that returns the z co-ordinate of the centre of the object with respect to the camera position.

GOBJZA (o)

A function that returns the angle of rotation of the object about a line through its centre and parallel to the z axis.

GOVER drawmode

drawmode 0, -1 0 = normal -1 = XOR

Sets the drawing mode.

GPAPER [o,] col

Sets the colour of the background. If an object is specified, it will be cleared by being redrawn in that colour. The background colour - and hence the clearing action - is turned off if col = -1. This can be used to make patterns: as the object is not cleared, it will leave an image every time it is moved or rotated.

GPLACE [o,] xp, yp, zp

xp, yp, zp: -16384 TO 16383 x, y, z positions of object

Note: -32768 TO 32767 will be accepted, but the distance from the camera must not exceed 32767.

Sets the object at a given position. If no object is specified or o = 0, the camera will be set at the given position.

GPLOT [o [TO o2]]

Plots the object(s). After an object has been plotted, the plotting lists are swapped (see GSWAP).

GPOINT col, xp, yp

It will plot a point using the algorithm set by GDRAWTYPE. Pixel co-ordinates are used when in fast point drawing mode. Graphic co-ordinates are used when in QDOS drawing mode. The values in the plotting lists are compatible with the requirements of this routine.

GPOINT3D col, xp, yp, zp

It will plot a point in the 3D world. You specify its position just as an object (see GPLACE).

GREMOVELINE o, l [TO l2]

Removes the line(s) from the description.

GREMOVEOBJ o [,TO o2]

Removes one or more objects.

GREMOVEPOINT o, p [TO p2]

Removes the point(s) from the description.

GREMOVEVERTEX o, v [TO v2]

Removes the vertex/vertices, unless used by a point or line, in which case an 'in use' error will be generated.

GROTATE [o,] xa, ya, za

xa, ya ,za: -32768 TO 32767 xa, ya, za are the angles of rotation of the object about lines through its centre parallel to the x, y, z axis respectively.

Prepares rotation of an object by calculating new 'result' co-ordinates of its vertices from the vertex definitions and the command parameters. If no object is specified or o = 0, the camera will be rotated. GMAKE and GPLOT must follow if the rotation is to become visible.

GROTATEVERTEX no, v [TO v2] [,xa, ya, za]

xa, ya ,za: -32768 TO 32767 xa, ya, za are the angles of rotation of the vertex about lines through its centre parallel to the x, y, z axis respectively

Prepares rotation of some selected vertices in an object by calculating new 'result' co-ordinates from the vertex definitions and the command parameters. GMAKE and GPLOT must follow if the rotation is to become visible.

GROTATEVERTEXX no, v [TO v2] [,xa]

xa: -32768 TO 32767

Prepares rotation of some selected vertices in an object about a line through its centre and parallel to the x axis by calculating new 'result' co-ordinates from the current 'result' co-ordinates and the command parameters. GMAKE and GPLOT must follow if the rotation is to become visible.

GROTATEVERTEXY no, v [TO v2] [,ya]

ya: -32768 TO 32767

Prepares rotation of some selected vertices in an object about a line through its centre and parallel to the y axis by calculating new 'result' co-ordinates from the current 'result' co-ordinates and the command parameters. GMAKE and GPLOT must follow if the rotation is to become visible.

GROTATEVERTEXZ no, v [TO v] [,za]

za: -32768 TO 32767

Prepares rotation of some selected vertices in an object about a line through its centre and parallel to the z axis by calculating new 'result' co-ordinates from the current 'result' co-ordinates and the command parameters. GMAKE and GPLOT must follow if the rotation is to become visible.

GROTATEX x, y, z, xa

x, y, z: -32768 to 32767 Variables - NOT expressions - whose current values (-32768 to 32767) represent the x, y, z co-ordinates of a point before a rotation of xa degrees about the x axis, and are changed by the command to the co-ordinates after rotation

xa: -32768 to 32767 Angle of rotation

Rotates x, y, z about the x axis.

Note: If you use a compiler which does not support parameter passing by reference (say Supercharge) an alternative method of finding the changed co-ordinates will be found in the GRXVALUE, GRYVALUE and GRZVALUE functions.

GROTATEY x, y, z, ya

x, y, z: -32768 to 32767 Variables - NOT expressions - whose current values (-32768 to 32767) represent the x, y, z co-ordinates of a point before a rotation of ya degrees about the y axis, and are changed by the command to the co-ordinates after rotation

ya: -32768 to 32767 Angle of rotation

Rotates x, y, z about the y axis.

Note: If you use a compiler which does not support parameter passing by reference (say Supercharge) an alternative method of finding the changed co-ordinates will be found in the GRXVALUE, GRYVALUE and GRZVALUE functions.

GROTATEZ x, y, z, za

x, y, z: -32768 to 32767 Variables - NOT expressions - whose current values (-32768 to 32767) represent the x, y, z co-ordinates of a point before a rotation of za degrees about the z axis, and are changed by the command to the co-ordinates after rotation

za: -32768 to 32767 Angle of rotation

Rotates x, y, z about the z axis.

Note: If you use a compiler which does not support parameter passing by reference (say Supercharge) an alternative method of finding the changed co-ordinates can be found in the GRXVALUE, GRYVALUE and GRZVALUE functions.

GRXVALUE

A function that returns the x co-ordinate computed by the last GROTATEX, GROTATEY or GROTATEZ command.

GRYVALUE

A function that returns the y co-ordinate computed by the last GROTATEX, GROTATEY or GROTATEZ command.

GRZVALUE

A function that returns the z co-ordinate computed by the last GROTATEX, GROTATEY or GROTATEZ command.

GSAVEOBJ o, "filename"

Saves an object. If an object with that name already exists, it is overwritten.

GSCALE yl, zl

yl:	32-512	height of the screen window in world co-ordinates
zl:	32-2048	distance into the screen equivalent to yl in world co-ordinates

Sets the yl, zl parameters.

GSETLINE o, l, col, v1, v2, dummy

dummy: Must always be set to zero.

Defines a line. If l leaves a gap after the previously highest line number, intervening lines are marked.

GSETPOINT o, p, col, v, dummy

dummy: Must always be set to zero.

Defines a point. If p leaves a gap after the previously highest point number, intervening points are marked.

GSETVERTEX o, v, x, y, z

Defines a vertex. If v leaves a gap after the previously highest vertex number, intervening vertices are marked.

GSIZEOBJ o, intx, fracx, inty, fracy, intz, fracz

intxyz: Integer part of magnification factor
fracxyz: Fractional part of magnification factor (denominator 32768 implied, numerator can be 0 to 32767)

Changes the size of an object. As it changes the definition, it will first come into effect after GROTATE or GCHANGEOBJ. Magnification factors are given along each axis. Observe that the size is changed in a fixed world co-ordinate system. You can see that the object becomes 'deformed' when it is rotated.

GSTART pa, no

pa: plot area, i.e. area used to hold the plotting lists
no: number of objects to be used

The first command in every program that uses the Tolkit. It sets the size of the plot area and the number of objects that can be loaded or defined. It can only be used once in a program.

GSTOP

Resets the graphics toolkit.

GSWAP [o [TO o2]]

Swaps the plotting lists. Remember that the plotting lists are swapped by every GPLOT; so if you want to replot identically after a GPLOT, GSWAP has to be used first, in order to swap the lists back.

GTRAPERROR on/off Errors are trapped when this is set to 1.

Traps errors so that program execution continues. The error number can be read by GERROR.

GWINDOW xl, yl, xp, yp [, yl, zl, sf]

xl, yl Must be at least 16

Sets window position and size. The screen parameters can also be set (see GSCALE, GFACTOR).

PROGRAMMING WITH THE ASSEMBLER TOOLKIT

.1 How to Go About It

iis chapter assumes that you have read the chapter on BASIC ·ogramming.

ien using the Assembler, you may load the BASIC Toolkit as described ı section 3.1. You can, however, save some memory by using)_TOOLKITASM_BIN, which does not contain any BASIC definitions. If ju clear the link flag the BASIC definition will not be linked in, ı the basic toolkit.

ıu can change the position of the work area. Set FILEPOS to the :art and MEMTOP to the end. The change must be made after a call to ıe start of the toolkit and before the start of the Assembler program :START).

allow QDOS points and lines, some dummy variables must be defined. ı normal assembler, GBV.RIP must be set to point to a stack in the b and GBV.VVBAS to a reasonably large area. In BASIC extensions V.RIP should be set to BV.RIP and GBV.VVBAS to BV.VVBAS. This is ly necessary when using those commands that use the QDOS scale, dot d line traps (i.e. window, dot, line, clearobject and plotobject). LOADOBJECT and/or SAVEOBJECT are used, use BP.ABSIO to set dresses.

u can make your own drawing routines. D0-D4, A0, A3 need not be eserved. On entry A2 will point to the parameters.

r an example of drawing routines see DRAWTYPE and BASICEXT.

r an example of assembly see ASMDEMO.

r an example of BASIC extensions see BASICEXT, BASICEXTDEMO and SICEXTDEMOBOOT.

2 Assembler Routines

```
A5  (often) object number
A0  (often) camera address
A2  is always preserved or set to data address
A3  (often) object address
A4  (usually) undefined; use for call of routines
A5  is always preserved or set to address of EXECLIST

START    $00
```

This call is like GSTART.

INPUT		OUTPUT
D1.W	plot area in K	all undefined.
D5.W	number of objects max 1024 every object uses 4 bytes	
D6.W	number of cameras (always 1)	

POINT $01

Plots a point, using one of the algorithms set by DRAWTYPE.

INPUT		OUTPUT	
D0.W	colour	D0.L	undefined
D1.L	X position	D1.L	undefined
D2.L	Y position	D2.L	undefined
		D3.L	undefined
		D4.L	undefined
		A3.L	undefined

PLACE $02

Sets the line start.

INPUT		OUTPUT	
D1.W	X1 position	D1.L	preserved
D2.W	Y1 position	D2.L	preserved

LINE $03

Plots a line, using one of the algorithms set by DRAWTYPE.

INPUT		OUTPUT	
D0.W	colour	D0.L	undefined
D1.W	X2 position	D1.L	undefined
D2.W	Y2 position	D2.L	undefined
		D3.L	undefined
		D4.L	undefined
		A3.L	undefined

POINT3D $05

Plots a point in 3D, using one of the algorithms set by DRAWTYPE.

INPUT		OUTPUT
D0.W	colour	all undefined
D1.W	x position	except: D7, A4
D2.W	y position	A0=camera address
D3.W	z position	

PLACE3D $06

Sets the start of a line in 3D.

INPUT		OUTPUT	
D1.W	x position	D1.L	preserved
D2.W	y position	D2.L	preserved
D3.W	z position	D3.L	preserved

LINE3D $07

Plots a line in 3D, using one of the algorithms set by DRAWTYPE.

INPUT		OUTPUT
D0.W	colour	all undefined,
D1.W	x position	except:
D2.W	y position	A0 camera address
D3.W	z position	

CONV16 $09

Calculates the x, y positions when z = 0.

INPUT		OUTPUT	
D1.W	x position	D1.W	x pos. in 2D
D2.W	y position	D2.W	y pos. in 2D
D3.W	z position	D3.L	undefined

ROTATEINIT16 $0A

Calculates rotate information.

INPUT		OUTPUT	
D1.W	X angle(0 to 360)	D1.W	undefined
D2.W	Y angle(0 to 360)	D2.W	undefined
D3.W	Z angle(0 to 360)	D3.W	undefined
A3.L	addr. of calculated info (9 words)	A3.L	preserved
		A4.L	undefined

ROTATE16 $0B

Rotates x, y, z about 0, 0, 0 using rotate information from ROTATEINIT16.

INPUT		OUTPUT	
D1.W	x position	D1.W	rotated x
D2.W	y position	D2.W	rotated y
D3.W	z position	D3.W	rotated z
		D4.L	undefined
		D5.L	undefined
A3.L	addr. of calculated info	A3.L	preserved

ROTATEX16 $0C

Works like ROTATE16 but rotates about the x axis only.

ROTATEY16 $0D

Works like ROTATE16 but rotates about the y axis only.

ROTATEZ16 $0E

Works like ROTATE16 but rotates about the z axis only.

ADDRCAMOBJ $21

Returns the addresses of the camera and object.

INPUT		OUTPUT	
D5.W	object number	D5.L	preserved
D6.W	camera number (always 1)	D6.L	preserved
		D7.L	undefined
		A0.L	camera address
		A3.L	object address

WINDOW $22

Changes the window size and position.

INPUT		OUTPUT
D1.W	x length min. 16	all undefined
D2.W	y length min. 16	
D3.W	x pos.	
D4.W	y pos.	

DRAWTYPE $23

Sets the algorithm to be used in making points and lines.

INPUT		OUTPUT	
D1.B	point type 0,1,2(to 32767)	D1.L	preserved
D2.B	line type 0,1,2 (to 32767)	D2.L	undefined
		A0.L	undefined
		A1.L	undefined
		A3.L	undefined
		A4.L	undefined

MODE $24

Changes the internal screen-mode setting.

INPUT		OUTPUT	
D1.B	mode 4/8	D1.L	preserved
		D2.L	undefined
		A0.L	undefined
		A1.L	undefined
		A3.L	undefined
		A4.L	undefined

CLS $25

Clears the window.

INPUT	OUTPUT	
	D1.L	undefined
no input	D3.L	undefined
	A1.L	undefined
	A4.L	undefined

PAPER $26

Works like GPAPER, i.e., sets the screen colour. If D5 <> 0, it will be CLEARed by drawing the object in that colour.

INPUT		OUTPUT	
D1.B	colour	D1.L	preserved
		D3.L	undefined
D5.W	if 0 then screen	D5.L	preserved
		A1.L	undefined
A3.L	object address	A3.L	preserved
		A4.L	undefined

INK $27

Works like GINK, i.e., PLOTs the object by drawing it in that colour.

INPUT		OUTPUT	
D1.B	colour	D1.L	preserved
		D3.L	undefined
		A1.L	undefined
A3.L	object address	A3.L	preserved

SCALE $28

Sets YL (the window height) and ZL (the depth corresponding to this), both in world co-ordinates.

Warnings: If low values of YL are used, precision suffers, resulting in strange deformation of objects as they move. Clipping at screen borders may not work at extreme combinations of YL, ZL, SF, so that system information may be overwritten, resulting in a crash (see GVARSF in 4.3.1).

INPUT			OUTPUT
D1.W	YL	32 to 512	all undefined
D2.W	ZL	32 to 2048	

OVER $29

Sets the plotting mode.

INPUT		OUTPUT	
D1.W	mode 0 normal	D1.L	undefined
	-1 XOR	D3.L	undefined
		A1.L	undefined
		A4.L	undefined

CLOSE $2B

Close closes the window. The window can be reopened by WINDOW.

INPUT	OUTPUT	
no input	A4.L	undefined

STOP $2C

Stop resets the toolkit: if the window is open it will closed.

INPUT		OUTPUT	
no input		A4.L	undefined

ROTATECAMERA $2D

Works like GROTATE with 3 parameters, i.e., rotates the camera about its centre (centre of window).

INPUT		OUTPUT	
D1.W	XA(0 to 360) not -XA	D1.L	preserved
D2.W	YA(0 to 360) not -YA	D2.L	preserved
D3.W	ZA(0 to 360) not -ZA	D3.L	preserved

MOVECAMERA $2E

Moves the camera.

INPUT		OUTPUT	
D1.W	+-forward	D1.L	undefined
D2.W	+right/-left	D2.L	undefined
D3.W	+up/-down	D3.L	undefined
		D4.L	undefined
		D5.L	undefined
A0.L	camera address	A0.L	preserved

ADDRCAMERA $30

Returns the address of the camera definition (see 4.3.1).

INPUT		OUTPUT	
D6.W	camera number (always 1)	D6.L	preserved
		A0.L	address of camera

ROTATEOBJECT $31

Works like GROTATE with 4 parameters, i.e., rotates the object about its centre.

INPUT		OUTPUT	
D1.W	XA(0 to 360) not -XA	D1.L	undefined
D2.W	YA(0 to 360) not -YA	D2.L	undefined
D3.W	ZA(0 to 360) not -ZA	D3.L	undefined
A3.W	object address	A3.L	preserved

MOVEOBJECT $32

Moves an object.

INPUT		OUTPUT	
D1.W	+-forward	D1.L	undefined
D2.W	+right/-left	D2.L	undefined
D3.W	+up/-down	D3.L	undefined
		D4.L	undefined
		D5.L	undefined
A3.L	camera address	A3.L	preserved

ADDROBJECT $34

Returns the address of the object definition (see 4.3.3).

INPUT		OUTPUT	
D5.W	object number	D5.L	preserved
		D7.W	undefined
		A3.L	object address

MAKEOBJECT $35

Calculates an object and produces a plot list.

INPUT		OUTPUT
D1.W	to object number if 0 then one object.	all preserved except: A0 camera address
D5.W	object number (not if D1 = 0)	
A3.L	object address (only if one object)	

PLOTOBJECT $36

Draws an object from the plot list.

INPUT		OUTPUT
D1.W	to object number if 0 then one object	all preserved except: A0 camera address
D5.W	object number (not if D1 = 0)	
A3.L	object address (only if D1 <> 0)	

CLEAROBJECT $37

INPUT	OUTPUT

Draws an object in background colour from a plot list.

INPUT		OUTPUT
D1.W	to object number if 0 then one object	all preserved except: A0 camera address
D5.W	object number (not if D1 = 0)	
A3.L	object address	

SWAPOBJECT $38

Swaps the plot lists so an object can be plotted repeatedly.

INPUT		OUTPUT
D1.W	to object number if 0 then one object	all preserved except: A0 camera address
D5.W	object number (not if D1 = 0)	
A3.L	object address	

DEFINEOBJECT $39

Defines a new object's size, i.e. the number of vertices, points and lines it can contain.

INPUT		OUTPUT
D1.W	number of vertices	all undefined
D2.W	number of points	except: D0=error
D3.W	number of lines	1: WA 2: PA 3: TO
		A3.L object address

REMOVEOBJECT $3A

Deletes an object from memory.

INPUT		OUTPUT
D1.W	to object number	all undefined
D5.W	object number	

LOADOBJECT $3B

Loads an object into memory. The device and filename in standard format must be pointed to by A0. (Errors: WA, PA, TO, FF)

INPUT		OUTPUT
A0.L	pointer to name	all undefined except: D0 = error

SAVEOBJECT $3C

Saves an object from memory. The device and filename in standard format must be pointed to by A0.

INPUT		OUTPUT
D5.W	object number	all undefined
A0.L	points to the filename	

INSERTOBJECT $3F

Inserts an object into another object.

INPUT		OUTPUT
D0.W	colour if -1 then copy colour	all undefined except: D0 = error code
D1.W	to object number	IF positive then
D5.W	from object number	bit 0: too few vertices 1: too few points 2: too few lines

CENTREOBJECT $40

Changes the centre of the object by adding the values to the definition. ROTATEOBJECT or CHANGEOBJECT must be used before the change will come into effect.

INPUT		OUTPUT	
D1.W	x change	D1.L	preserved
D2.W	y change	D2.L	preserved
D3.W	z change	D3.L	preserved
		D4.L	undefined
A3.L	object address	A3.L	preserved
		A4.L	undefined

SIZEOBJECT $41

Changes the size of the object by multiplying the values into the definition. ROTATEOBJECT or CHANGEOBJECT must be used before the change will come into effect.

INPUT		OUTPUT	
D1.W	integer	D1.L	preserved
D2.W	part 0 to 32767	D2.L	preserved
D3.W	x: 1, y: 2, z: 3	D3.L	preserved
		D4.L	undefined
		D5.L	undefined
		D6.L	undefined
A3.L	object address	A3.L	preserved
		A4.L	undefined

MARKOBJECT $42

Marks one or more objects. If an object is marked, GMAKE, GPLOT, GCLEAR and GSWAP will have no effect on it.

INPUT		OUTPUT	
D1.W	to object number	D1.L	preserved if zero
D2.W	FLAG: 0 OR 1	D2.L	preserved
D5.W	object number	D5.W	preserved
		D7.W	undefined
		A0.L	undefined
		A3.L	undefined

CHANGEOBJECT $43

Moves the definition to the result.

INPUT		OUTPUT	
		D1.L	undefined
		A1.L	undefined
A3.L	object address	A3.L	preserved

WORKFREE $49

Returns the number of free bytes in the work area.

OUTPUT	
D0.W	free bytes

PLOTFREE $4A

Returns the number of free bytes in the plot area.

OUTPUT

D0.W free bytes

FINDMAX $4B

Finds the largest value of x, y, z vertex definitions in the specified range.

INPUT		OUTPUT	
		D0.W	error or vtx number
D1.W	vertex number	D1.L	undefined
		D5.W	max. value
D6.W	to vertex number	D6.L	undefined
		D7.L	undefined
		A1.L	undefined
A3.L	object address	A3.L	preserved
		A4.L	undefined

ROTATEVERTEX $4D

Rotates the vertex/vertices.

INPUT		OUTPUT	
D0.W	If 0, angle from object	D0.W	error or vtx number
D1.W	vertex number	D1.L	undefined
D2.W	XA	D2.L	undefined
D3.W	YA	D3.L	undefined
D4.W	ZA	D4.L	undefined
		D6.L	undefined
		A1.L	undefined
A3.L	object address	A3.L	preserved
		A4.L	undefined

ROTATEVERTEXX $4E

Rotates the vertex/vertices about a line through the centre of the object and parallel to the x axis.

INPUT		OUTPUT	
D0.W	If 0, angle from object	D0.W	error or vtx number
D1.W	vertex number	D1.L	undefined
D2.W	XA	D2.L	undefined
D3.W	0	D3.L	undefined
D4.W	0	D4.L	undefined
		D6.L	undefined
		D7.W	undefined
		A1.L	undefined
A3.L	object address	A3.L	preserved
		A4.L	undefined

PLOTFREE $4A

Returns the number of free bytes in the plot area.

OUTPUT

D0.W free bytes

FINDMAX $4B

Finds the largest value of x, y, z vertex definitions in the specified range.

INPUT

D1.W vertex number

D6.W to vertex number

A3.L object address

OUTPUT

D0.W error or vtx number
D1.L undefined
D5.W max. value
D6.L undefined
D7.L undefined
A1.L undefined
A3.L preserved
A4.L undefined

ROTATEVERTEX $4D

Rotates the vertex/vertices.

INPUT

D0.W If 0, angle from object
D1.W vertex number
D2.W XA
D3.W YA
D4.W ZA

A3.L object address

OUTPUT

D0.W error or vtx number
D1.L undefined
D2.L undefined
D3.L undefined
D4.L undefined
D6.L undefined
A1.L undefined
A3.L preserved
A4.L undefined

ROTATEVERTEXX $4E

Rotates the vertex/vertices about a line through the centre of the object and parallel to the x axis.

INPUT

D0.W If 0, angle from object
D1.W vertex number
D2.W XA
D3.W 0
D4.W 0

A3.L object address

OUTPUT

D0.W error or vtx number
D1.L undefined
D2.L undefined
D3.L undefined
D4.L undefined
D6.L undefined
D7.W undefined
A1.L undefined
A3.L preserved
A4.L undefined

REMOVELINE $53

Removes a line.

INPUT		OUTPUT	
D1.W	vertex number	D1.L	undefined
D2.W	to vertex number	D2.L	undefined
		D7.L	undefined
		A1.L	undefined
A3.L	object address	A3.L	preserved
		A4.L	undefined

4.3 Assembler Definitions

4.3.1 Camera definition

```
GVARRESTAB      EQU     18    Start of plot table
GVARCAMCENX     EQU     32    Defaults = 0, which is at the
GVARCAMCENY     EQU     34      centre of the screen
GVARCAMCENZ     EQU     36

Size of screen in world co-ordinates:
GVARXL          EQU     48    Calculated from yl, zl
GVARYL          EQU     50    Default 256
GVARZL          EQU     52    Default 512
GVARXLH         EQU     54    Centre of screen
GVARYLH         EQU     56          "
GVARSF          EQU     58    Screen factor; default = 17128

Size and position of screen in pixel coordinates:
GVARX1          EQU     64    For window clipping of
GVARY1          EQU     66    drawing routines
GVARX2          EQU     68    Mode8 x:0 to 255
GVARY2          EQU     70    Mode4 x:0 to 511
```

4.3.2 Object and camera definition

```
GVARX           EQU     0
GVARX16         EQU     2
GVARY           EQU     4
GVARY16         EQU     6
GVARZ           EQU     8
GVARZ16         EQU     10
GVARXA          EQU     12    Read only
GVARYA          EQU     14    Read only
GVARZA          EQU     16    Read only
```

4.3.3 Object definition

Where a name is followed by '16' (e..g GVARX16), only this part of the 32 bits should be used in this version.

GVARSIZE	EQU	22	The size of an object. Used to clip objects. Object will not be clipped if set to zero.
GVARHLIM	EQU	24	Position of the backplane
GVARFLAG	EQU	30	Low byte: If flag 2 = 1, size calc. is on If flag 3 = 1, backplane clip on High byte: If flag 0 = 1, object is on screen If flag 1 = 1, object is marked
GVARDISTX	EQU	32	Distance from camera to centre of object
GVARDISTX16	EQU	34	Calculated if backplane clip or size <> 0
GVARDISTY	EQU	36	
GVARDISTY16	EQU	38	
GVARDISTZ	EQU	40	
GVARDISTZ16	EQU	42	
GVARSX	EQU	44	Size of object (if flag set): Low word: integer High word: fraction
GVARSY	EQU	48	
GVARSZ	EQU	52	
GVAROBJLEN	EQU	60	Length of the definition of the object when in memory (only 1 to GVARNV, P, L are saved)
GVARNVM	EQU	64	Max. defined vertices
GVARNPM	EQU	66	Max. defined points
GVARNLM	EQU	68	Max. defined lines
GVARNV	EQU	80	Number of vertices used
GVARNP	EQU	82	Number of points used
GVARNL	EQU	84	Number of lines used
GVARVERTEXTAB	EQU	96	Address of vertex table
GVARPOINTTAB	EQU	100	Address of point table
GVARLINETAB	EQU	104	Address of line table
GVARPLOTTAB2	EQU	128	Offsets pointing to the objects's plot lists
GVARPLOTTAB1	EQU	132	GVARRESTAB+GVARPLOTTAB(1/2) = the plot table (1/2) address

4.3.4 Vertex table

VERTEXXD	EQU	0	Definition of object
VERTEXYD	EQU	2	
VERTEXZD	EQU	4	
VERTEXX	EQU	6	Rotated definition
VERTEXY	EQU	8	
VERTEXZ	EQU	10	
VERTEXINFO	EQU	12	Set to zero when created; some unused space here
VERTEXFLAG	EQU	15	If bit 7 = 1 then not calculated
	EQU	16	

4.3.5 Point table

COLOUR	EQU	0	If flag 15 = 1 then no plot
VERTEX	EQU	2	Starts from vertex 0
RESERVED	EQU	4	and some more
	EQU	6	

3.6 Line table

```
COLOUR          EQU     0   If flag 15 = 1 then no plot
VERTEX 1        EQU     2   Starts from vertex 0
VERTEX 2        EQU     4   Starts from vertex 0
RESERVED        EQU     6   and some more
                EQU     8
```

3.7 Plot table

```
COMMAND         EQU     0   1 = point     2 = place
                            3 = line     -1 = stop
COLOUR          EQU     2
X               EQU     4
Y               EQU     6
                EQU     8
```

It is conceivable that the plot table format may change in future versions of 3D PRECISION.

3.8 Other parameters

```
ADDREXECLIST    EQU     0
NOOBJN          EQU     4
NOOBJMAX        EQU     6
ERRORNO         EQU     8
CHANNEL         EQU     10  Used for QDOS points/lines
CAMADDR         EQU     14
GBV.RIP         EQU     18  Set to stack address
GBV.VVBAS       EQU     22
FLAGS           EQU     26  8: 1 = trap error
                            9: 1 = after start
                           10: 1 = after window
                           11: 1 = link BASIC
OBJMAX          EQU     28
FILEPOS         EQU     30  Start of file (the default is the
                            address after the toolkit (when
                            loaded)). (For changing the address
                            see 4.1).
MEMTOP          EQU     34  DEFAULT:start of toolkit + WORK AREA
AFTERFILE       EQU     38  Pointer to first free byte after
                            last object
WORKTAB         EQU     42  First byte not free (used by make)
                            (length = (max number of vertices in
                            any object) * 16)
BASICDEF        EQU     46  Start of BASIC definitions (offset
                            from start of 3D_TOOLKIT_BIN
POINTTYPEADDR   EQU     50  Address of linked draw routine for
                            point
LINETYPEADDR    EQU     54  Address of linked draw routine for
                            line
MODE            EQU     62  Mode8: 0 mode4: 1
POINTTYPE       EQU     63  Type of point: 0, 1, 2 to 32767
LINETYPE        EQU     64  Type of line: 0, 1, 2 to 32767
OVER            EQU     66  Highest byte = 1: OVER mode will be
                            used when drawing
DRAWSTACK       EQU     70  Address of co-ordinates put by the
                            place routine for use in line draw
                            routine
END             EQU     74
```

.9 Extra error codes

```
ERR.WA      1   Work area used up
ERR.PA      2   Plot area used up
ERR.TO      3   Too few objects allocated in 'START'
```

5. INDEX

Entries referring to the new keywords in section 3.2 are arranged alphabetically within that section. Many other keywords for which a reference to 3.2 is given will be found in that section under the corresponding keyword with initial G; e.g., SCALE is discussed in section 3.2 under GSCALE.

GCAMY 3.2
GCAMYA 3.2
GCAMZ 3.2
GCAMZA 3.2
GCENTREOBJ 3.2
GCHANGEOBJ 3.1 3.2
GCLEAR 3.1 3.2
GCLOSE 3.2
GCLS 3.1 3.2
GDATAADDR 3.2
GDEFINEOBJ 3.2
GDRAWTYPE 2.3.3 3.2
GERROR 3.2
GFACTOR 3.2
GFINDLINE 3.2
GFINDMAX 3.2
GFINDPOINT 3.2
GFINDVERTEX 3.2
GINK 3.2
GINK . . . . . 3.2 4.2.$27
GINSERTOBJ 3.2
GLINE 3.2
GLINE3D 3.2
GLOADOBJ 3.1
GLOADOBJECT 3.2
GMAKE . . . . . 3.1 3.2 4.2.$42
GMARKLINE 3.2
GMARKOBJ 3.2
GMARKPOINT 3.2
GMARKVERTEX 3.2
GMODE 3.2
GMOVE 3.2
GNOOBJ 3.2
GNOOBJMAX 3.2
GOBJADDR 3.2
GOBJMAX 3.2
GOBJX 3.2
GOBJXA 3.2
GOBJY 3.2
GOBJYA 3.2
GOBJZ 3.2
GOBJZA 3.2
GOVER 3.2
GPAPER . . . . . 3.2 4.2.$26
GPLACE 3.1 3.2
GPLOT . . . . . 3.1 3.2 4.2.$42
GPOINT 3.2
GPOINT3D 3.2
GREMOVELINE 3.2
GREMOVEOBJ 3.2
GREMOVEPOINT 3.2
GREMOVEVERTEX 3.2
GRID 1.2 2.4.3
GROTATE . . . . . 3.1 3.2 4.2.$2D 4.2.$31
GROTATEVERTEX 3.2
GROTATEVERTEXX 3.2
GROTATEVERTEXY 3.2
GROTATEVERTEXZ 3.2
GROTATEX 3.2
GROTATEY 3.2
GROTATEZ 3.2
GRXVALUE 3.2
GRYVALUE 3.2
GRZVALUE 3.2
GSAVEOBJ 3.2
GSCALE 3.2