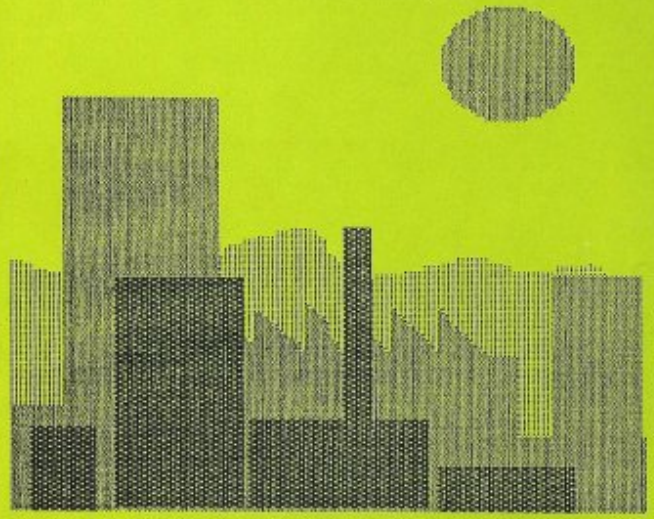
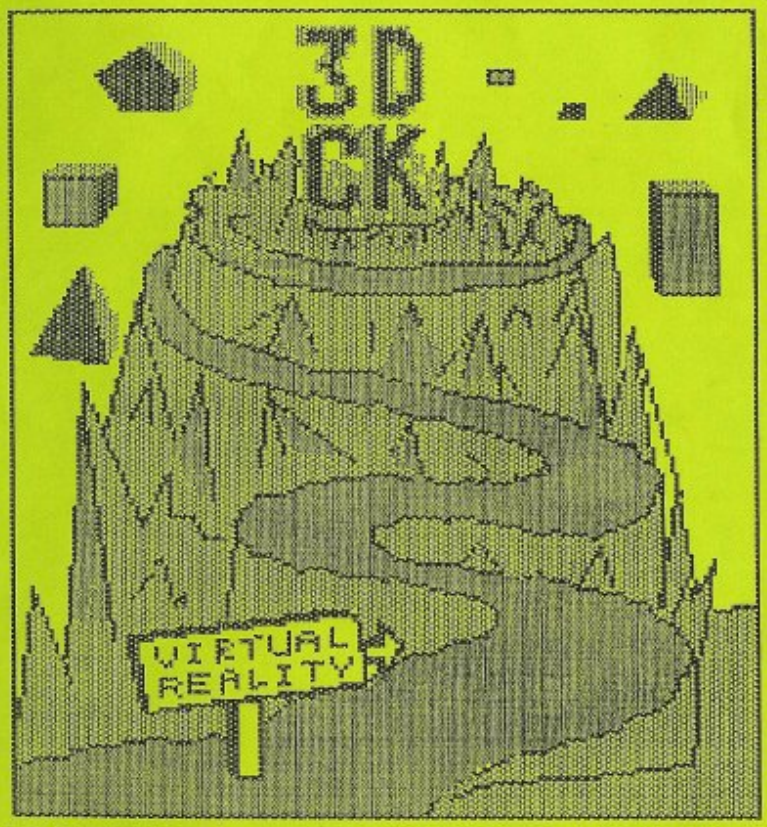
3D CONSTRUCTION KIT
USER GROUP



3D CONSTRUCTION KIT
USER GROUP



MAGAZINE ISSUE 13
JUNE/JULY 1993

EDITORIAL

Welcome to issue 13 of the 3D Construction Kit User Group Newsletter.

You must all be wondering what on earth has been happening for this issue to be so late in getting to you. I owe you all my profound apology for the huge delay. You see, on 10th June whilst I was about to start preparing this issue, a day which started out as a warm and sunny day, we were hit by a freak tropical storm the like of which I have never seen before - or want to see again! We had five inches of rain in just four hours. It is the first time in my life that I have looked through the door at the rain and not been able to see through it! Within a short space of time we all realised that the drains etc were not going to be able to cope with such a deluge and, sure enough, we were flooded. It was a dreadful sight. Anyone who has had the misfortune to experience flooding will know what I mean. We barricaded both the front and the back doors and stood guard with mops and buckets in the vain hope that we could keep it out but the water came up though the floor as well as through the doors and we had no option but to retreat! Unfortunately, the photocopier stood just at the foot of the stairs in the hallway and so prevented us from being able to salvage anything from the ground floor as there was no room to carry anything upstairs around it and it was far too heavy to move. As most of you may know I run the User Group from home. By the time that the storm - we live in a dip in the centre of Llandudno just below sea level - the water on the ground floor had reached a height of three feet and was still rising.

We were under water for five days and, because it wasn't just water that flooded us but also sewage from a sewage system that was also inundated and decided that the only way out was up, everything on the ground floor had to be thrown out. Not only that but the whole of the ground floor had to be completely gutted. We spent the subsequent 11 to 12 weeks camped out on the landing upstairs whilst all the floors, doors, walls down to the brick were ripped out and replaced after drying out. Most of my computer equipment went, together with the photocopier, fax and other items which I used for the User Group. During the repairs it was absolutely impossible to deal with any correspondence or Newsletters so everything had to be put on hold.

Thankfully we are now back to normal and I have been catching up with things, a huge mountain of mail has now been dealt with and thankfully I am now back at the keyboard getting an extremely belated June/July issue ready to send out to you. The August/September issue will be done at the same time so that we will all be up to date again as soon as possible.

You have all been so patient and understanding and the many messages of sympathy that I received from so many of you when you phoned to see what was going on brought tears to my eyes. You have all been so very kind and it was wonderful to know that you are not just User Group members but friends too. A couple of you pointed out that the issue due out during the flood was issue thirteen and I was seriously thinking of calling it issue twelve and a half for a while!

Well, enough of explanations for now - I am sure you all get the picture about our situation and I don't want to sound depressing, so let's get back to the newsletter. I sincerely hope that you all find this issue interesting and helpful - I'll see you all again soon.

Mandy

LETTERS

Dear Mandy

Congratulations on the second anniversary of the 3D Kit User Group. These past two years have really been fun for me using the Kit. I do have a couple of observations to make on Kit 2 for the Amiga. Version V2.07 still has a couple of bugs which are a bit frustrating: when choosing colours from the border or from Freescape, either way you end up with different colours in the edit window and in test mode when the border is present. When you choose, say, green to colour your object in the test window pressing ESC to see the border the same object has a brown colour and so it goes with every colour. I had quite a job adjusting the colours to the border. I was really confused when, upon making a stand alone game I discovered the Freescape colours were back again. Once everyone knows of the problem it can easily be overcome providing that you don't mind, and can ignore, the different colours appearing in test mode when the border is present. My next point is that I find that, although in this version you can increase the buffer size, when you do so you cannot make a stand alone game anymore when your datafile is greater than the default memory amount. I tested this out and I received a runtime error. I found this disappointing as I had only used about 20,000 bytes more than the default and it is difficult to split my game into two parts. All in all, despite all the many new and exciting features of Kit 2 it almost makes me want to go back to programming on Kit 1 as it is bug free and works much more smoothly.

Mieke Van Der Poll **- Kit 1 and 2 AMIGA AND PC**

Hopefully all these niggly bugs will be cleared up with version V2.9, Mieke, and then you will be as happy programming in Kit 2 and you are with Kit 1. Come to think of it - what better person could I find to give V2.09 a thorough bug-test? The disc is on it's way to you, Mieke, if you would be so kind as to help us out.....Mandy

Dear Mandy

Have you seen PC Format this month? There's an article about SUPERSCAPE and the CYBERZONE T.V. programme, plus a round-up of the 3D software available and including an interview with Ian Andrew. The writers don't seem too impressed with Kit 2. This is a pity - and perhaps someone should point out to them that it is the only real-time 3D Interactive' software available to us mere mortals with low bank balances and average computers! How can you slate something when there is nothing to compare it with? There is Superscape true, but that's the professional version of what we use anyway. It is a bit like saying that the Sun isn't up to much, it is a bit yellowy, not very bright, covered in spots... when it is the only one we have and we'd be lost without it! I have only one minor tipette to offer this month. I've recently upgraded my MS-DOS operating system to version 5 and the computer's CPU to a 386SX. With this combination a lot of the operating system can be run from the UMB memory space the new DOS finds. There is also enough room to install a screen-grab utility - I haven't found a screen-save facility within the Kit - into the upper memory also, along with the mouse driver. All this leaves around 609K of free base memory and sidesteps all the juggling with CONFIG.SYS files necessary on some machines.

Nigel Alefounder **- Kit 1 and 2 - PC**

Dear Mandy

I have something which might be covered under your licence from Incentive Software - that is making video films of 3D Kit routines. I

might be something to do with Domark not supporting your computer and the low sales etc., but I think it is because there are only a handful of Archimedes members of the User Group compared with hundreds of other computer users. This doesn't affect routines in the newsletters etc as they apply to all machines but Acorn specific disks with games or routines have yet to surface.....Mandy

Dear Mandy

I am very pleased that you always seem to have material for us 8 bit users in the newsletters, although I would like to see more of them as the majority of writers seem to own the larger computers. Everyone these days seems to be saying that the life of the 8 bit computer is over which is very sad. What do you think? Are there a lot of 8 bit 3D Kit users out there?

Keith Marshall - Spectrum Kit 1

I hate to disappoint you, Keith, but the number of 8 bit owners is dwindling rapidly. Not because they are not re-subscribing to the newsletter but because they are upgrading to 16 bit machines. As far as I know, only Commodore are still producing an 8 bit computer these days and, because of the lack of software for 8 bit machines I have to say that it does, indeed, seem that we are seeing the last days of the faithful old 8 bits. It has always been a struggle to obtain contributions for the 8 bit version of the Kit for the newsletters as, apart from an enthusiastic few members who have kept us going all this time, the response from the majority of 8 bit owners has been very poor. The majority of the 8 bit contributors are now upgrading to 16 bit computers also (four of them this month alone!), and I doubt very much if the 8 bit sections can continue indefinitely. I'll keep on including them as long as I possibly can but if they dry up completely then I'm afraid that even the newsletters will eventually turn out to be 16 bit dedicated only.....Mandy

Dear Mandy

I'm new to 3D Construction Kit and I'm having a number of problems, it might be best if I start by describing the system I am using. I have a PC 286 with 1 Mb of memory, maths co-ori, VGA, soundblaster and a hard drive. I run DRDOS version 6.0 with Superstore and have 3D Kit2 (v2.01). My questions are numerous and no doubt often answered in the newsletter but here we go. Why can Amiga users use sampled sounds and I can't. This may sound like sour grapes but I spent a lot of money on my sound blaster and I am very proud of it. If I tell my machine to use the soundblaster card then 3D edit won't work. It says in issue 11 that the PC can use PCX files (320x200x256 colours) as borders. Well mine doesn't - however, converting this to an LBM file using Graphics Workshop creates a useable file. You mention version 2.03 in issue 11, can I upgrade my version and if so how? If the datafile format is the same across different computing platforms, has anyone tried a straight data connection to transfer the wealth of PD from the Amiga and Atari to the PC? I think that is about my lot except to say that when viewing borders my machine has an annoying habit of hanging - not always but just often enough to cause me real headaches.

Colin Paterson - PC

I think the first thing we have to do is to get you the latest version of the Kit which should eliminate most of your problems - especially with Soundblaster as it was not entirely compatible with early versions of the Kit. Send me your disks and an S.A.E. and I'll swap them for you. Yes, we have tried various ways of converting Amiga and ST datafiles to work on PC. No success whatsoever with Amiga to PC but we have had some success with Atari to PC usually with an urgent

need to change the colours to see exactly what the datafiles look like on the PC. We'll keep trying and let you know. Nigel Alefounder has had some success though - over to you Nigel.....Mandy

Dear Mandy

Thanks very much for the last delivery of indexes and FAST TOWN for the Atari ST. The former are certainly immediately useful and no 3D-Kitter should be without them! Now to report on how I got on with the Atari files. Firstly there was NO problem with the PC reading the Atari disk. I've no idea how the ST version works, as I've never had access to one, so just started by copying all the files onto a PC formatted disk for safety, then I renamed the FASTTOWN datafile so the PC would load it. This meant adding an extension (.KIT) - like Kit1 used to use. After a slight, (but normal) wait the sample world appeared. The palette was completely wrong (unless the author has a very unusual sense of colour!). I notice that on the disk there wasn't the usual triplet of files which the PC version of Kit1 used to use i.e. .KIT .NAM .PAL, so conclude that either there is only one ST palette available or the colour info is stored in the main datafile as in Kit2. Anyway, apart from the psychedelia the only other problem seems to be in the maze. Like you I couldn't resist going where the author warns not to! Trouble is on the PC all that happens is that you are dumped at the beginning of the maze and the computer locks-up solid! Otherwise it was an enjoyable romp around galleons, factories, buses and the curiously nostalgic overhead power cables I remember from my childhood (too many years ago!). On the subject of the individual objects. Well Kit2 doesn't recognise Kit1 objects anyway as far as I can see so there was little hope of joy with the ST ones. The best thing to do would be to load the world, re-colour the objects, group them and save. Ah well, we can't have everything. Still all this DOES mean PC users can have access to the ST worlds/datafiles - which is very useful. Now all I need is a program to convert ST Neopaint files to a PC format...

Nigel Alefounder - Kit1 and Kit2 - PC

Dear Mandy

I am pleased to say that I now have a working copy of Kit2. Thanks for sending it to me. I notice when you run your program after compiling it using 3DMAKE the program displays "Loading a Virtual Reality Studio 2 Virtual World" on the screen. I have found a way to by-pass this by writing an MS-DOS batch file. This comes in useful if you want to either remove this message or add your own message. I contacted Domark recently and asked them if I would be allowed to sell my game as shareware once I had created it. They said yes but only if I stated that it had been created using the Kit. Could you verify this as I am sure other members of the group might find this interesting?

John Clarke - PC Kit2

You can publish, sell, swap, give-away, use as shareware or PD, your finished programs or datafiles. In fact you can do whatever you want to do with them without asking prior permission from Incentive/Domark or paying any royalties etc. The only stipulation to this is that you acknowledge somewhere within the package, disk or documentation that you used the 3D Construction Kit to write the program.....Mandy

Dear Mandy

I am writing in reply to Dominique Watson's idea of a Virtual Reality Visette. He said he hoped there is someone out there who knows how to implement joystick controls using infra red beam. Well there is a company called Spectravideo of Unit 27, Northfield Industrial Estate,

Beresford Avenue, Wembley, Middx, HA0 1NU (Tel: 081 900 0024), who sell walkabout joysticks for just £14.95. This infra red joystick does away with leads and plugs, players can walk around the room and still play their favourite games. It sends infra red rays from as far away as thirty feet, the signals being picked up by a control box which plugs into the back of the computer. It takes the place of the old joystick so it should work with 3D Kit. As for the two miniature TV's inside the visette - they could research development using existing hand held games machines which use low cost screens especially made for computer graphics. You could build part of it yourself. Hand held games in which you can load programs start from around £39.99 for Quickshot Supervision, up to Game Gear at £99.99. The idea is to buy two hand held games machines and build them into your own VR Helmet at low cost and then load the software to simulate the VR World. With this idea you could actually do away with a computer main base on the other side of the room and have a hand held joypad connected to the VR Visette. This low cost idea has never been exploited yet so the idea is up for grabs. Manufacturers would only build new products if they could keep the price down for the finished product. In the Kit 2 manual on page 18 it says that Incentive do not yet possess the knowledge to get a real human being into the world created by the program but with a few more years research and some powerful hallucinogens a solution could soon be at hand. Well I have found a solution to reduce the number of years of research and just research the compass or altitude direction indicator to control the joystick action to the toolkit. Another method of introducing computer graphics to the two or four LCD colour TV's build into the visette is to use a video sender. This, which is connected to the computer, will send 3D Kit sound and vision signals through the air up to 1000 feet radius. You are able to tune in the miniature TV to pick up the Construction Kit sound and graphics. It is alright having a picture but you need something to move about in the 3D World. This is where the infra red joystick is used. With the joystick you are able to walk anywhere in the house and control the environment you can see through the visette. To improve the idea further you could build the joystick into the visette controlled by tilt switches or compass rotation. You could control shooting by connecting a wire down from the head set into your hand. Incentive is only a software company and not a hardware manufacturer so I don't think they could build it, although there is a big sales potential for any manufacturer who could sell it at low cost.

R. Connell - Bolton, Lancs

Dear Mandy

I think the magazine is very good as it has something for all levels of expertise. I found the Beginner's section very interesting as I am self taught and have only had the computer for about three years. One thought I've had, is it possible to obtain a folder on the lines of the ST Format or ST User sort? I find the mag has a tendency to get lost amongst the paper work in my computer drawer.

B. Miles - Trowbridge

I did find the ideal binder a couple of years ago in W.H.Smiths, it was A5 size and had twelve thin metal rods in the centre to allow it to store up to twelve issues - rather like the old TV Times type binders. Unfortunately when I went back for more of the same I was told that it had been a promotional drive offer and there were no more to be had. I will investigate with some companies that I know of who produce various types of binder to see what can be found. Providing the cost is right it might be worth looking into.....Mandy

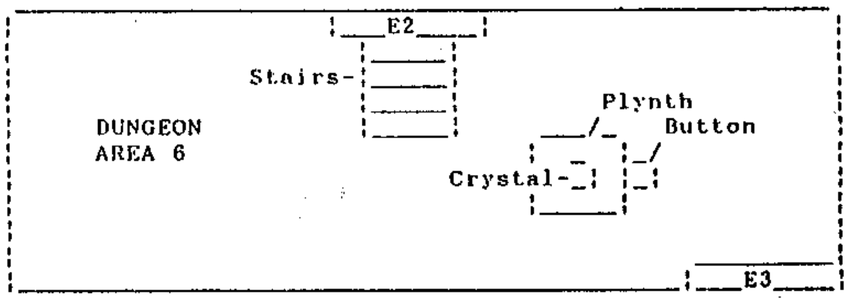
MORE LETTERS NEXT ISSUE!

THE USER GROUP GAME

PART FIVE:

We are going to enhance our game with a couple more puzzles for the player to solve. Basically what we need to do is to create a puzzle within our dungeon area which interacts with a hidden cell nearby. We are going to create a plynth in the dungeon with a crystal on top and also add a button on the bottom of the plynth - not exactly hidden but making it easy to miss if the player isn't searching properly - the button when activated will reveal a hidden door leading to a cell. Within the cell we are going to create another puzzle where, upon entering we will find a deadly wall, complete with nasty spikes, rushing towards us. Just inside the cell will be another plynth upon which we want the player to put the crystal they found in the dungeon. When they have done this the wall will stop moving. Within the cell is a pile of gold ingots hidden within an alcove halfway down the cell. The player has to stop the wall before it covers the alcove so that they can access the treasure.

The first thing we have to do is to create all the objects within the dungeon so that our dungeon area looks something like this:



Firstly we should enter the simple condition needed to allow the player to take the crystal from the plynth by activating the crystal and so making it vanish:

CONDITION FOR CRYSTAL (in my case Pyramid 15)

```
IF ACTIVATED?
THEN INVIS (15)
ENDIF
```

We need to be able to allow the player to take and replace the crystal on the plynth - set initially to vis - and also cover the possibility that the player might also activate the plynth when the crystal is on it, so we will include a "ping" sound effect.

CONDITION FOR PLYNTH (in my case Cube 14)

```
IF ACTIVATED?  -        If activated and
AND VIS? (15)           crystal is there
THEN SOUND (4)          just make a ping sound
ELSE IF INVIS? (15)     but if it isn't there
THEN VIS (15)           then make it appear
SOUND (4)               make a ping sound.
ENDIF                   Note: two endif's as
ENDIF                   we had two if's.
```

Create a rectangle, paint it black to represent a doorway and position it against the wall in the correct place, stretch it to the height you think appropriate for a doorway. Before you enter any conditions for the doorway you should walk right up to it until you are facing it squarely and, making sure you are not touching it but are just in front of it, make a U-turn so that you are standing with your back to it - to give the effect that you have just walked through it from the cell - and CREATE ENTRANCE. This should be entrance 3 and make a note of this so you can come back to this through the door from the cell by sending the player to Entrance 3 in Area 6 (i.e. GOTO (3,6)). When you have done this, turn around so you are facing the doorway - to make things easier - and enter the following condition for the doorway: Which in my case was Rectangle (17):

```
IF COLLIDED? -   If you bump the doorway
THEN GOTO (2,8)  goto E2 in A8 and start the
STARTANIM (1,8)  animation in area 8 - we
ENDIF            will program this later.
```

When you have done this, make sure that you make the doorway INVISIBLE both Initially and Curently via the ATTRIBUTES. Don't worry about the animation just now, that will be programmed when we come to the Cell area.

Now the almost hidden button, a very small cube, should be placed low down on the far side of the plynth, to make things more difficult I coloured mine almost the same shade as the plynth so that it is only when the player goes round the plynth and looks down that it is seen. We now need to enter the condition for the button so that each time the button is activated the doorway will either appear or disappear, depending on its current state. There will also be a sound effect and, remembering that the player is standing with his back to the doorway he/she will have to turn around to see what has happened when they hear the sound effect.

CONDITION FOR THE BUTTON (in my case Cube 16)

```
IF ACTIVATED?  -      If button is pressed
AND VIS? (17)         and doorway is visible
THEN INVIS (17)       then make it invisible
SOUND (2)             and give sound effect.
ELSE IF INVIS? (17)   but if it is invisible
VIS (17)              make it appear and
SOUND (2)             give sound effect.
ENDIF
ENDIF
```

You may have noticed that in all these conditions we have put the "negative" aspects first. I.e. We have told the program what to do in case someone does something twice first. This is simply because that is the way that the program works and we would have great difficulty in doing it the logical way, such as because the doorway was set to invisible we should program it firstly to appear when the button is pressed. With all FCL programming, if a condition isn't working the way we want it is alway wise to check this out first in case we have put something the wrong way round.

Now test everything thoroughly to make sure that everything is working properly. Next issue we will program our final area - the Cell.

TO BE CONTINUED....

BEGINNER'S SECTION

MULTIPLE OBJECTS IN THE TUNNEL AREA

Supplement to the routine in Newsletter 12, page 11
Entitled USING AN AREA MORE THAN ONCE

By *Mieke Van Der Poll* - AMIGA & PC KIT 1 AND KIT 2

This is quite interesting when making a maze in which you have to solve a lot of problems by picking up objects to do so. I will give you an example, using the same entrance variable used in the previous article, making three objects, which each use a variable too.

Create 3 different objects, for instance:

Object 1: a match consisting of two cubes (say cube 2 and cube 3).
Create a group and add the two cubes to it: Group 4 = match.
Choose a variable for this group: Say V35.

Object 2: an arrow consisting of a cube (say 5) and a pyramid (say 6).
Create a group and add the two to it: Group 7 = arrow.
Choose a variable for this group: say V36.

Object 3: a bow consisting of a cube (say 6) and a pyramid (say 9).
In Kit 2 the pyramid could be made wireframe.
Create a group and add the two to it: Group 10 = bow.
Choose a variable for this group: say V37.

Load the three objects into the "tunnel area" and make every part of all three invisible. You may end up with different group numbers, this being only an example.

In the tunnel area (3) create local condition 1. (When you enter the tunnel the first time the entrance variable is set to 1 (1,V45)).
Edit local condition 1:

KIT 1

```
IF VAR=? (V45,1)
THEN IF VAR=? (V35,0)
THEN VIS (4)
INVIS (7)
INVIS (10)
ENDIF
ENDIF
```

KIT 2

```
IF VAREQ? (V45,1)
THEN IF VAREQ? (V35,0)
THEN VIS (4)
INVIS (7)
INVIS (10)
ENDIF
ENDIF
```

The match will become visible when you enter the tunnel through that entrance.

When you enter the tunnel for the second time the entrance variable is set to 2 (2,V45). Create and edit local condition number 2 in the same way as above, this time the first viable is (V45,2), the second one is (V36,0) and object (7) must become visible and the other two invisible, and so on with the third object and, if you wish, many more.

Now when an object is visible in the tunnel you want the player to pick it up. You make an object condition for say cube 2:

KIT 1 AND KIT 2

```
IF ACTIVATED?
THEN PRINT ("YOU PICK UP A MATCH",1) 1=Optional
DELAY (150)
PRINT ("                          ",1)
INVIS (4)
SETVAR (1,V35)   (this is important!)
ENDIF
```

By setting variable 35 to 1 the object has been picked up and you prevent at the same time that the object will become visible again when you enter the area again, **IF VAR=?** or **IF VAREQ? (V45,1)**. In the local condition the variable must be (V35,0) in order for the object to become visible (so not having been picked up yet).

When a certain object that has been picked up, has to be placed in another area again (to solve a problem), be sure to set the variable a number higher than 1 and not to zero because that is the number you reserved in your local condition for it to become visible. You use the above routine in the same way for the other two objects.

In this way you can put a lot of objects in the same area, even if you use it more than 20 times in a maze! You have to make a map when you want to do that. Each time you want a particular object to be visible, choose the entrance variable that has been set when entering the area at that particular place.

Remember: I created three local conditions, one for each object, but it might very well be possible to put all those commands in one local condition. I did not try that out though.

in Kit 2 you could also make a Procedure and put all those commands into it. Then you only have to create one local condition and edit it with the text: proc (1).

CONDITIONS

By *Tony Hartley* - KIT 1 & KIT 2 - ATARI ST

If you are new to any of the 3D Kit programs you may be wondering what the difference is between all the types of conditions and how they all work. There are four types of conditions - INITIAL, GENERAL, LOCAL and OBJECT CONDITIONS in KIT 2 and GENERAL, AREA, OBJECT conditions in KIT 1. The INITIAL CONDITIONS (or in the case of Kit 1 the 1st General Condition), are executed when the program first begins and also when it is reset. You need to firstly CREATE the condition ready for use then EDIT CONDITION to enter your text. These conditions are used for example if you want to load a picture file for use as an information screen in a game, also to load a border (Kit 2 only) and starter variables. In other words any commands that you want to happen when your game first begins. In this example we have one loading screen (drawn on an Art Package, one border and one variable which is used to disble the shooting. The shooting variable can be re-activated once you find a gun. This condition is for Kit 2:

```
LOADSCREEN(".INSTRUC.PI1",0) - Name and type of picture.
DELAY (650)                  - Gives time to view picture.
CLS                          - Clears the picture.
BORDER (1)                   - Loads your border.
SETVAR (0,V20)               - Disables the shooting variable.
```


Next we come to GENERAL CONDITIONS, (from Condition 2 onwards in Kit 1). These are conditions that execute anywhere in your world. They are useful for countdown timers, updating instruments or anything that you want the computer to keep an eye on for any changes throughout the game no matter which area you are in. For example if you wanted to keep track of how long you had been playing your game you could use a general condition to keep track of the time for you. e.g.

TIME (V52,V53,V54) - see page 34 in 3DK2 manual.
This would have to be used with three instruments as well.

Next there are the LOCAL CONDITIIONS or in the case of Kit 1 the AREA CONDITIONS. These are nearly the same as General Conditions but are only activated when you visit the area that they are in. A Local or Area condition could be used to start an animation, set an object fade (Kit 2), start a Brushanim (Kit 2) or anything that you want to apply once you reach that area. Although you can use these conditions to make something happen in another area. An example of this could be:

```
IF INVIS? (25)        - If object 25 is invisible
THEN STARTANIM (1,2)  start animation 1 in area 2
```

Thus the Local/Area condition is checking to see of Object 25 is invisible and when it is it will start animation 1 in area 2 which is another area.

Finally we have OBJECT CONDITIONS. These are only usually activated when an object is collided with, activated or shot. The Object condition could be used to check if someone has bumped into an object and then if this happens the object could vanish, e.g.:

```
IF COLLIDED?     - if you bump into this object
THEN INVIS (45)  then make it (object 45) vanish
```

Or another example could be to create a cube/pyramid into a gun shape then when someone takes the gun they are now able to fire. This would be used with an initial condition for example. after you have created and edited the Initial condition, as mentioned previously, to disable the shooting until the gun is found and picked up, you could enable it again by using these conditions:

```
INITIAL CONDITION:
        SETVAR (0,V20) - disable firing (Kit 1 and Kit 2).

OBJECT CONDITION:
        IF ACTIVATED?    - if you have touched it
        THEN INVIS (45) - make it (object 45) invisible
        SETVAR (15,V20) - enable rapid firing
        ENDIF
```

So that is a brief explanation of the different conditions and what they do. Study the manuals for more help but, in a nutshell -

Initial Conditions Kit 2 and General Condition 1 Kit 1 are for one off settings after a reset, e.g. to set variables, load screens, borders etc before the start of a game.

General Conditions are for general purpose functions that happen every frame, e.g. Timers, setting the time or anything that you want to apply to every area.

Local Conditions Kit 2 or Area Conditions Kit 1 are for making things happen only when you are in the area that triggers the condition. E.g. if you enter an area the condition in that area could start an object FADEBOUNCING (Kit 2 only) or print the name of that area to a text instrument.

Object Conditions are for making something happen if you interact with a particular object.

I hope that helps a little, there are endless possibilities to using conditions and lots of examples in the manuals and the Newsletters so experiment and have fun!

AMIGA SOUND SAMPLE BANKS

By *Mieke Van Der Poll* - AMIGA KIT 2

On the Amiga it is possible to load a sound sample bank and edit the samples within the datafile you created in 3DEdit. The first thing you have to do though is to create your sample bank with the 3DSound Program and save the bank with the extension .3sm. Then start 3DEdit, load in your datafile and load in the sound sample bank. Under the sound menu there is also a command to edit your samples. In the manual nothing has been written on how to do that. It is not so difficult so if you haven't already discovered how to do that I will explain some things below.

When you start to edit the sounds, it seems as if there are only five sounds available, called sound 1, 2, 3, 4, 5, the names of which you can change in the first five of your sound bank samples. In the menu window of the sound editor there are, besides EDIT, two more important buttons: ADD and STORE. If you have edited the first five samples, don't forget to store each one, otherwise you end up with the same sounds you had at the start. When you click on ADD you see sample number 6 appear with the name empty. Change it into your sample name nr.6 and click on EDIT. In the next menu click three times on INSERT. In the first row click on NOP once. In the second row click on NOP twice and in the third row click on NOP three times. Then fill in a number in the first row under value 1, say 500. Fill in a number in the second row, say 64. Then in the third row you fill in the number of your sound sample, in this case it is number 6. Then press PLAY and you will hear your 6th sample sound.

There are lots of ways to change the sound by clicking on the different row buttons and by changing the relevant numbers in the first two rows. Find out for yourself. When you are happy with the sound click on OK and, don't forget! click on STORE in the first window.

In this way you can add all the samples of your sound bank to your datafile - I did so with 26 ones on the one meg Amiga 500 - and save the datafile. The next time you load in your datafile you also have to load in your sound sample bank but then all the edited sounds are present.

We are getting a bit short on contributions for the Beginner's section so I would be grateful if readers could put pen to paper - or pinkies to the keyboard and let me have some more as soon as possible please. I have just enough for the next issue but need more for October- Mandy

8 BIT ROUTINES

DARK AREAS

By Steven Flanagan - C64

I've spotted a tiny error in the 8-bit user group game. The routine at the bottom of page 13 of newsletter issue 11 does not work because it tries to nest IF statements, and this is not allowed on 8-bit machines.
The routine should be:

```
IFHIT (r)
THEN
ELSE
END
ENDIF
IFVIS 11 2
THEN
GOTO 2 7      .     Goto the dark area.
ELSE
GOTO 2 5      .     Goto the light area.
END
```

Now I can tell you that you wont need this routine because there is a better method. With the above routine you need two areas exactly the same but one with everything coloured black. (See last newsletter). This will use up a lot of memory if the area contains many objects.

Using my method, you only need to create one area. When you have done that, select COLOUR AREA in the environment editor and write down the four colours of the area on a piece of paper. I wall call these numbers Col1, 2, 3 and Col4. Next, load the condition editor and enter this routine instead:

```
IFHIT (r)      -     r = door rectangle number.
THEN
ELSE
END
ENDIF
IFVIS 11 2
THEN
SETV 0 1       -     Set flag to 0 if torch not taken.
ELSE
SETV 1 1       -     Set flag to 1 if torch taken.
ENDIF
GOTO 2 5       -     Goto the area.
END
```

In the area you wish to goto (in this case area 5), write this local routine:

```
CMPV 0 1       -     Test if torch is not taken.
```

Continued...

```
IFEQ            -       If not taken...
THEN            -       Then
COLOUR 0 0      -
COLOUR 1 0      -
COLOUR 2 0      -
COLOUR 3 0      -       Colour the whole area black.
REDRAW
ELSE            -       If the torch is taken...
COLOUR 0 Col1-
COLOUR 1 Col2-
COLOUR 2 Col3-
COLOUR 3 Col4-          Colour the area with it's original colours.
REDRAW
ENDIF
SETV 3 1
END
```

The reason why variable 1 is set to 3 at the end is because if it was left at 0 or 1 the whole condition would be repeated every frame (including the REDRAW command), which would take up processing time.

*****

MAKING MORE SOUND EFFECTS
By Steven Flanagan - C64

The few sound effects provided with the kit are a bit limiting when you want the game to make a certain sound such as a car engine running. Here are two simple routines for making different uses of the sounds provides:

Method 1.

```
SOUND (N)
END
```

Method 2.

```
SOUND (N)
DELAY (D)
END
```

Where N is the number of the sound to be sounded. The best sounds to use are numbers 2, 3, 4, 5, 7, 8 and 11. D is the delay time. Experiment by altering the values of N and D until you get the your desired sound. Some of them do give a very good car engine noise.

Unfortunately, the noises created with these routines only work when the 3D world is not being updated. In other words, the sound effect is lost if the player moves.

An simple example of a use for these sound is as follows. A player enters a room to find a coffee machine next to the wall, on activating the machine, the humming sound is produced for a few seconds and his/her cup of coffee appears. This is a pretty useless example but it was all I could think of.

16 BIT ROUTINES
LANDSCAPE TECHNIQUE

By *J. Hayes* - 16 BIT KIT 1

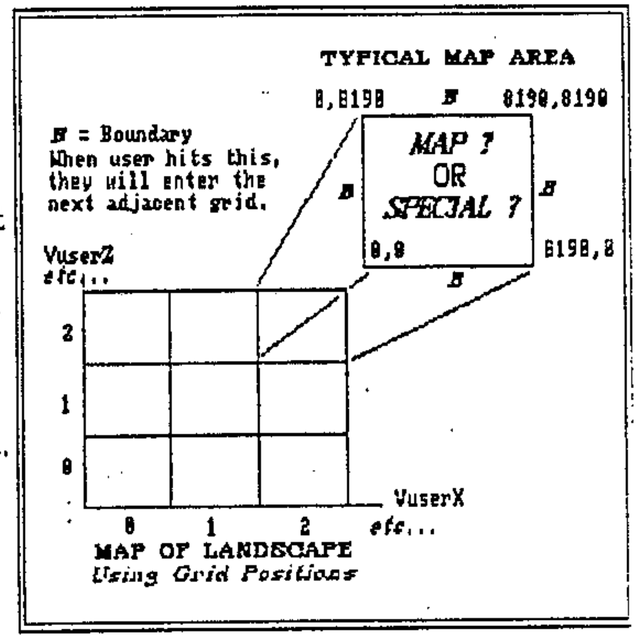
The 3D Kit can create many environments, but supposing there is a need to create an actual landscape. This can take many forms but there is a problem here. How to show a vast landscape without using a number of areas or spend lots of time creating the features to fit into every one.

This idea provides a general way of implementing any size of landscape.

It uses only one area for the main map, leaving lots of areas available for other uses.

To the right is a diagram showing the basic principle.
The landscape is based around the *grid position* the user occupies.

This is also used by the landscape to refer to the features that will appear when the user is at a certain position on the map.



MAP OF LANDSCAPE
Using Grid Positions

The map itself is built from only ONE area. At first this seems a little hard to imagine, but it is easy to give the impression of crossing a vast landscape by using it.

The method used to detect when the user has reached a boundary is described later. For now it is necessary to understand the mechanics behind using just the one area.

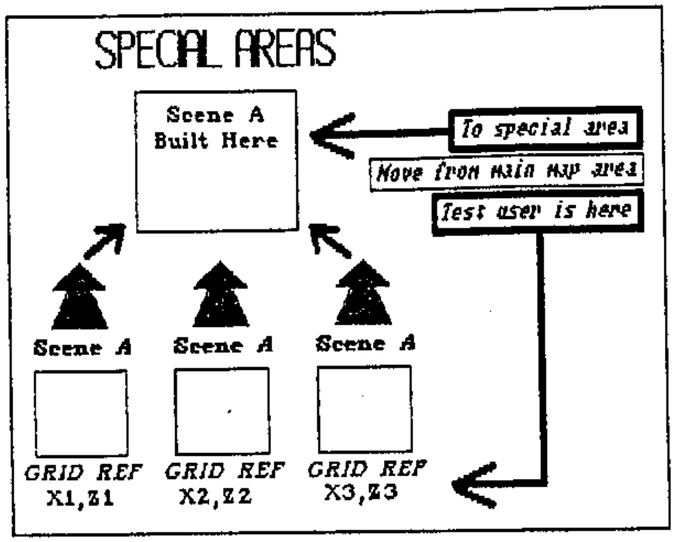
It is a simple illusion. The user remains in one area. When they reach the edge of it, they are transported to the opposite side. Their position on the map in terms of the grid position VuserX & VuserZ is updated to reflect where they now stand on the landscape as a whole.

The grid position is a unique label attached to user. It can be used to enable events or in this case, change the features seen on the landscape.

Another practical use of knowing where the user is on the map, is to move them to certain areas when they reach a specific grid. In this way, set scenes can be built where the main action to a game or environment will occur.

These other areas will obey the same movement rules governing the main map. The user will return to it if they hit a boundary, unless other special areas are detected.

To the right is a diagram showing how a special area will work. These individual scenes will be placed around the landscape at the appropiate grid positions. But remember these positions will only come into existence when the user has `advanced` into them.



The diagram shows a possible case where the same scene can be used. It needs only to be built once in an area.

However, as the grid position they relate to is different for each similar scene, variations can be enforced.

FOR INSTANCE

Suppose SCENE A was a village. At one of them, an object representing food could be made to appear, while at the others, it will not. The same distinction can be used for other features, varying the appearance or layout of the village.

The same approach is used to determine what features will be seen on the map area. Except a more general method is employed, in an attempt to give an apparent `randomness` to the landscape.

VARIABLES -*Refered my name*

Vmapsize	Size of landscape (X & Z)
Vuserx	X position of user *on map*
Vuserz	Z position of user *on map*
Vflag	Status marker
Vnewarea	New area to enter
Vvaryfeat	Determines the features seen
Vposx	User X coordinate *in area*
Vposz	User Z coordinate *in area*
Vlookx	User X rotation *in area*
Vlooky	User Y rotation *in area*

(Method assumes no vertical movement allowed)

STATUS IN Vflag when BIT VALUE is set

VALUE	*REFERENCE*	*PURPOSE*
1	Newgrid	User entered a new grid
2	Viewfeat	OK to view features

DEFAULT SETTING WHEN ENVIRONMENT IS RESET

General Condtion 1

```
SETVAR(Size,Vmapsize)   Define map size
SETVAR(x,Vuserx)        Put user somewhere in it at X
SETVAR(z,Vuserz)        Put user somewhere in it at Z
ORV(newgrid,Vflag)      Tell routine user is in a new grid
```

The default condition will also need to be set to 1

MOVEMENT ACROSS LANDSCAPE

This is the test used to check if the user has reached the boundary of an area. It will apply to every area built, unless the user is prevented from reaching a boundary by a 'wall'.

These can be enabled when the user is at a certain grid. Providing the obstruction was built along with the other features, it can be made to appear. For invisible barriers, colour the obsticle 'invisible'.

General Condition 2

```
IF VAR)?(V2,8190) AND VAR(?(Vuserz,Vmapsize) THEN
    ADDVAR(1,Vuserz)
    SETVAR(1,V2)
    ORV(newgrid,Vflag)
    ELSE IF VAR(?(V2,1) AND VAR)?(Vuserz,0) THEN
    SUBVAR(1,Vuserz)
    SETVAR(8190,V2)
    ORV(newgrid,Vflag)
    ELSE IF VAR)?(V0,8190) AND VAR(?(Vuserx,Vmapsize) THEN
    ADDVAR(1,Vuserx)
    SETVAR(1,V0)
    ORV(newgrid,Vflag)
    ELSE IF VAR(?(V0,1) AND VAR)?(Vuserx,0) THEN
    SUBVAR(1,Vuserx)
    SETVAR(8190,V0)
    ORV(newgrid,Vflag)
ENDIF
```

NOTICE THE INSTRUCTION ORV(newgrid,Vflag)

This sets the bit in Vflag to the value represented by *newgrid.* It is used to find the correct area the user ought to be in when they have crossed a boundary on the map.

FIND CORRECT AREA

This will act on the bit **newgrid** being detected.

The users grid position is used to find any special areas that deviate from the map area , or return the user to the map when leaving special areas.

The map area must be built in AREA 1

General Condition 3

```
IF ANDV(Vflag,newgrid) THEN
    SUBVAR(newgrid,Vflag)  disable this condition.
    IF VAR=?(Vuserx,SpecialX) AND VAR=?(Vuserz,SpecialZ) THEN
        SETVAR(specialarea,Vnewarea)
        ELSE IF VAR=?....etc
         follow with other special areas
         ....
         ....
        ELSE IF VAR)?(V8,maparea) THEN
        SETVAR(maparea,Vnewarea)
         If this becomes true, the user was in another area.
         Put them back on the map !
        ELSE
        ORV(viewfeat,Vflag)
         If this is true, the user is still on the map.
         Just instruct the VIEWING ROUTINE to work.
     ENDIF
ENDIF
```

PUT USER IN CORRECT AREA

This will act when Vnewarea has a valid number in it. To keep the illusion of crossing a landscape, the users coordinates and rotation are transfered also.

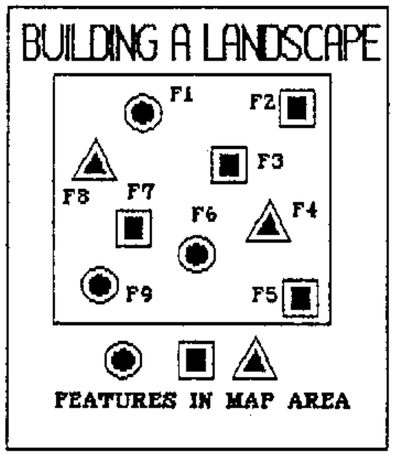
General Condition 4

```
IF VAR)?(Vnewarea,0) THEN
    SETVAR(V0,Vposx)
    SETVAR(V2,Vposz)
    SETVAR(V3,Vlookx)
    SETVAR(V4,Vlooky)
    GOTO(1,Vnewarea)
    SETVAR(Vposx,V0)
    SETVAR(Vposz,V2)
    SETVAR(Vlookx,V3)
    SETVAR(Vlooky,V4)
    ORV(viewfeat,Vflag)
    SETVAR(0,Vnewarea)
ENDIF
```


VIEWING ROUTINE

The purpose of this is to create the features seen in the map area, then let the routine decide what will appear. The advantages are clear with a large landscape - it would be very tedious defining the appearence of objects over countless grids.

First, a word on building features into the map area.



On the left are a collection of objects showing the features on the map area. Make a **note of the groups they belong to.** Due to the limit of *60 objects,* they will need to be carefully designed.

It is necessary though to construct a final group out of all the objects that will change, so ensure there is room.

Leave them all visible. Organise the features into batches, so that they form a priority list.

To each batch, a *bit value* is given. The routine arrives at a figure to test this value with by adding together the users grid position *x & z.*

BATCH	PRIORITY	CONSISTING OF GROUPS	VALUE
0	Always seen	F1,F2,F3	N/A
1	1st seen	F4,F5	1
2	2nd seen	F6,F7	2
3	3rd seen	F8	4
4	4th seen	F9	8
etc until no more groups			

If the bits in this figure match the value for a batch, it will be shown.

User at	Figure	Bits set	Batch seen
4,2	6	4+2	2 & 3
8,3	11	8+2+1	1,2 & 4

In practice, the user can forget what will appear. The intention here is to vary the landscape features without any concern over what they ought to be. This is a `hands off approach` and saves alot of work.

This example is used in the routine itself.

Local Condition 1 For map area

```
IF ANDV(Vflag,viewfeat) THEN
    INVIS(the group selected to vary:) NOT F1,F2,F3
    SETVAR(Vuserx,Vvaryfeat)
    ADDVAR(Vuserz,Vvaryfeat)
    IF ANDV(Vvaryfeat,1) THEN
        VIS(F4)
        VIS(F5)
    ENDIF
    IF ANDV(Vvaryfeat,2) THEN
        VIS(F6)
        VIS(F7)
    ENDIF
    IF ANDV(Vvaryfeat,4) THEN
        VIS(F8)
    ENDIF
    IF ANDV(Vvaryfeat,8) THEN
        VIS(F9)
    ENDIF
    SUBVAR(viewfeat,Vflag)
ENDIF
```

Notice how the values pointing to a batch are detected, then the batch itself is given. If the *batch condition* is true, the selected objects will appear.

Due to the way it is arranged, more than one condition can be true. So a number of batches can exist. It is rather like fitting the pieces into a jigsaw;

SUMMARY

A game can be built around this landscape idea. The user is free to go anywhere, without the limits imposed upon travel in a conventional 3D Kit game. The number of conditions needed to do the same normally, even on a small map, would need some planning.

The landscape can be ridiculously large, but it is wise to populate it with good special areas away from the map area. Being in the wilderness for too long is not a good idea; there has to be an aim to using this method, otherwise the benefit will be lost.

A compass can be added very easily by examining the users 'y' rotation.

This is demonstrated on the disk provided.

The method described is meant for 'external' landscapes. But it can be adapted for 'internal' uses like mazes or passages, by leaving a gap between the walls in order to reach a boundary. If so, each room/passage must occupy a grid on the map and follow a logical route; any exterior views relating to them must match as well.

Putting locked doors in the path of an exit is available. It will be necessary to remember what door(s) are locked or open/closed at certain grids by using variables to test for them and then make the correct item appear.

'Teleports' are possible as well ! Just force the routine to believe the user has entered a new grid. THUS:

SETVAR(TeleX,VuserX) : SETVAR(TeleZ,VuserZ) : ORV(newgrid,Vflag)

The routine will handle the rest ! Use carefully as the user could end up as part of the scenary in another grid !

The marker, *viewfeat in Vflag* will be available in special areas if required to change the scenary. If used it must be reset with *SUBVAR(viewfeat,Vflag)*. Nothing disastrous will happen by forgeting, it only stops features from being re-defined.

For internal scenes, like buildings which also adopt the idea of using the same area, a *viewing marker* must be set on entry (via doors), as viewfeat will not be available for this purpose.

The following is general guidence for using any area more than once for different scenes/features:

1: *Create all the objects in one area. Leave visible.*
2: *In Local condition 1*

```
    If viewmarker set
        Reset viewmarker
        INVIS all objects that vary
        If at grid G1, VIS those that exist here... etc
OR      If at grid G1 AND OK to see, VIS those ok to see
        (Like a door that was shut, must now be shown open)
OR      If at grid G1 AND in building B1...etc
        (Assuming building entered shares the same area and a record of
it was kept on entry)
    Endif
```

I trust all this can be useful ! It works well, much to my astonishment, but took some time to figure out. I enclose a disk with a full description demonstrating these principles for you to study.