NOW HEAR THIS

The most popular computer games rely heavily on the exciting visual effects that can be produced by home micros. While many games make a lot of noise, few of them make really creative use of sound. We continue our DIY programming series by suggesting a few ideas for the development of interesting and enjoyable 'audio games'.

```
  10 REM******AUDIO GAME*****
***
  20 PRINT"A SIMPLE AUDIO GAM
E":PRINT:PRINT"MOVE YOUR SPACE
SHIP TO THE BEACON BEFORE
    YOUR FUEL RUNS OUT"
  30 PRINT:PRINT"     THE NEA
RER YOU ARE, THE HIGHER
       THE BEACON'S PITCH"

  40 PRINT:PRINT"          C
ONTROLS ARE:":PRINT" I (UP)  M
 (DOWN)  J (LEFT)  K (RIGHT)"
  50 PRINT:PRINT"  *******PRE
SS ANY KEY TO START*******"
  70 A$=INKEY$(0):IF A$="" TH
EN GOTO 70
  80 P=INT(RND(1)*500)+1:Q=IN
T(RND(1)*500)+1:F=3*(P+Q)/2:X=
0:Y=0
  90 XD=0:YD=0
 100 PRINT "FUEL=";F;" DISTAN
CE=";
 110 IF (ABS(P-X)<2 AND ABS(Q
-Y)<2) THEN PRINT"YOU MADE IT
- WITH ";F;" FUEL UNITS LEFT":
END
 120 A$=INKEY$(0):IF A$="" TH
EN GOTO 120
 130 IF (A$="I" AND YD<3) THE
N YD=YD+1
 140 IF (A$="M" AND YD>-3) TH
EN YD=YD-1
 150 IF (A$="J" AND XD>-3) TH
EN XD=XD-1
 160 IF (A$="K" AND XD<3) THE
N XD=XD+1
 170 X=X+XD:Y=Y+YD
 180 D=SQR((P-X)*(P-X)+(Q-Y)*
(Q-Y))
 190 PRINT D
 200 SOUND 1,-8,(255-D),2
 210 F=F-ABS(XD)-ABS(YD):IF F
>0 THEN GOTO 70
 230 PRINT TAB(10);"*******CR
ASH*******":PRINT TAB(11);"***
OUT OF FUEL***"
 240 PRINT:PRINT"POSITION IS
";D;" SPACE UNITS FROM BASE"
 250 END
```

Arcade games use sound to support their well-developed graphics effects. Although sound effects are used to heighten the realism and excitement of the action, they are rarely the focus of the game itself. However, if we can use our visual sense as the basis for so many varied computer games, there is no reason why we can't generate games that are centred on our sense of sound.

The game we give here is such an 'audio game'. While we do not claim that it is the most exciting game ever developed, it is certainly interesting to play. What seems to be quite a simple task at first soon becomes a fascinating challenge simply because you have to use a different sensory input than that normally used when controlling some brightly coloured screen display.

In our game you are placed at the controls of a spacecraft that is running short of fuel. Your only hope of survival is to dock with a fuel station nearby. Unfortunately, owing to a malfunction in your computer system, you have no visual information to guide you in your efforts at docking; the only method of navigation is to home in on the station using an audible signal. The pitch of the 'navigation beacon' signal becomes higher the nearer you get; so it's just a question of listening carefully and responding precisely with the controls.

Just like a real spacecraft, once you set your ship off in a particular direction, it will keep going until you counteract that motion by using the opposite control. If you use two Us to move up quickly, then you'll need to press two Ds to stop again. This makes the game much harder than it seems.

It will take some practice for you to be able to complete the game using only the sound cues. By adding a few simple lines, however, you could give an indication on the screen of where the ship is in relation to the fuel station. This could simply show the distance between you and the station (the variable D) or it could tell the player the most effective buttons to push (if SGN(p–x)=–1 then you need to go left, for example). These helpful hints should make the game much easier to play.

Having tried our program, you may like to create your own games. Using sound to locate or avoid objects is a field with considerable programming possibilities. How about sonar games for submarines or a minefield that you have to navigate using a detector that emits an audible warning? A safe cracking game shouldn't prove too difficult.

The more advanced sound capabilities of machines like the BBC Micro and the Oric-1 will obviously be an advantage here. Sound games can use several voices simultaneously, or give them slightly different noises, each with its own significance in the game. The programmable volume level available on some machines will be a useful feature.

Basic Flavours

This program was written on a BBC Micro in Mode 7, and is, therefore, almost standard Microsoft BASIC. Its PRINT commands presuppose a 40-column screen display, and will need formatting for different displays. The SOUND command in line 200 is specific to BBC BASIC. The value of the parameter (255-D) is the pitch of the note to be played, while the other parameters control volume, duration and channel. INKEY$(0) in line 120, and the use of RND in line 80 will need attention on other machines:

Spectrum

Insert LET in all assignment statements. Change INKEY$(0) to INKEY$. Change RND(1) to RND. Change line 200 to:

```
200 BEEP 0.4,(255-D)
```

and insert:

```
15 RANDOMIZE
195 IF D>254 THEN LET D=D-254
```

Commodore 64/Vic-20

Change INKEY$(0) to GET A$. Refer to your user manual for Commodore's sound commands. Insert:

```
75 X=RND(-TI)
```

Dragon

Change INKEY$(0) to INKEY$. Change RND(1) to RND(0). Change line 200 to:

```
200 SOUND (255-D),10
```

and insert:

```
195 IF D>254 THEN D=D-254
```

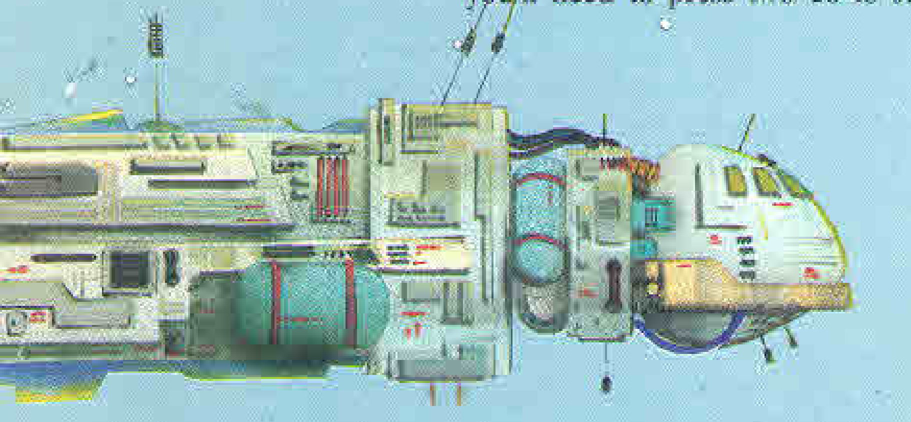
Oric Atmos

Change INKEY$(0) to KEY$. Change line 200 to:

```
200 SOUND 1,(255-D),9:WAIT 40:PLAY0,0,0,0
```

and insert:

```
195 IF D>254 THEN D=D-254
```



DAVE COOPER-SMITH