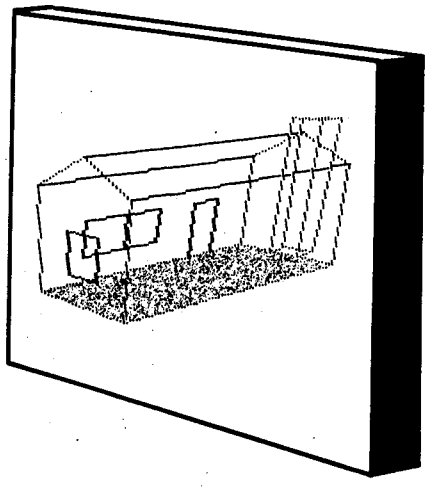
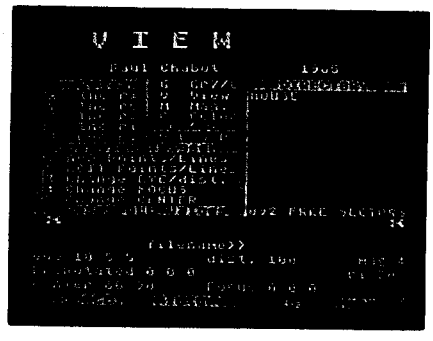
VIEW 3-D



Rotate and zoom 3-D images in ACTION!

by PAUL CHABOT



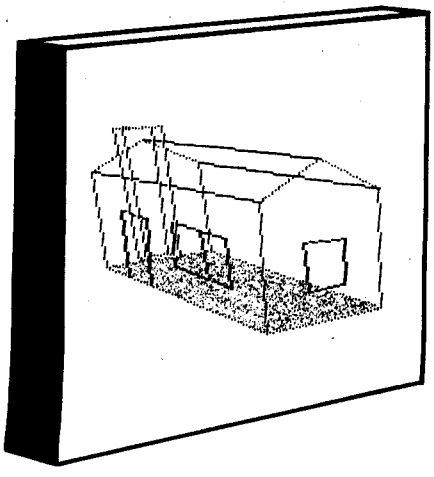
Create 3-D wire-frame outline pictures in your Atari's highest resolutions, Graphics 8 and Graphics 7+. Magnify, shrink, rotate, and otherwise shift your view of the 3-D picture easily and fairly quickly. Requires ACTION! cartridge, disk drive and 48K memory. Antic disk subscribers can run VIEW3D.EXE without the ACTION! cartridge. Disable BASIC and use the L option from DOS 2.0S. Disk or cassette.

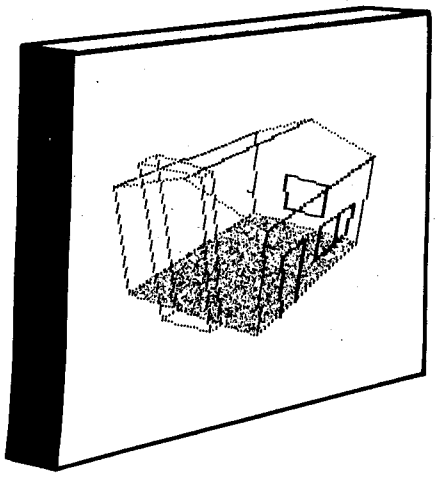
When Paul submitted View 3-D to ***Antic****, we saw it was easily the largest ACTION! program any magazine had considered publishing. But in recent months, we have received so many letters from readers wanting ACTION! that we thought it was time for a monster example of programming in this powerful Atari language.*

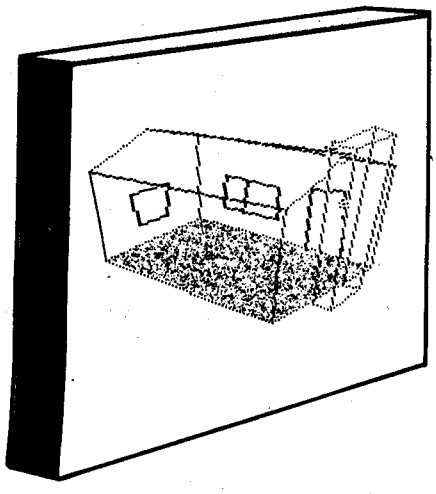
Be warned: there are ten separate program listings, nine of which are dependent on and INCLUDEd into the tenth to form one main program. Because of the nature of ACTION! there is no TYPO II, so type patiently and carefully. The results will be well worth it. —ANTIC ED

There are different approaches to 3-D viewing. You can leave the viewing point ("eye") fixed and rotate the object. Or you can think of the object as fixed and change the location of the eye. These are mathematically equivalent, but conceptually quite different to most people.

Also, should the projection be perspective or orthogonal? Where should the focus be placed? View 3-D will allow any combination of these variations and more. To manipulate a 3-D frame quickly, you need faster number crunching than BASIC provides. The answer is ACTION!, the cartridge-based programming language from Optimized Systems Software, which is becoming increasingly popular with serious Atari programmers.







TYPING IT IN

View 3-D is one program, but it has been split into ten files. Listing 10, called VIEW3D, is the main file which INCLUDEs the other nine. If you look at the beginning of listing 10, you can see the name of the other files.

Type each file in the order they are INCLUDEd in Listing 10. Each subsequent file shares procedures from previous ones, none may be compiled or run independently. You can partially check your work by compiling programs accumulatively in the order in which you type them. For example, GR78M may be compiled alone. After typing in MISC1, create a temporary third program which INCLUDEs GR78M and MISC1. This third program, when compiled, will compile the first two, and so on.

VIEW3D is too large to be compiled and run from the ACTION! editor. When all your files are properly typed in, clear the editor and, from the monitor, type: C "VIEW3D.ACT". After the compilation is complete, type [R] and away you go.

THE PROGRAM

The first thing you should see is the menu screen. View 3-D alternates between two screens—the *menu* screen and the *view* screen. The menu screen has command options and a disk directory. The view screen shows your 3D drawing. Shortly after the menu appears, the colors will alter and the program will switch to the view screen and display a simple 3-D object called "Plane" which is similar to *Figure 1.*

THE COMMANDS

With the exception of [D], any key pressed will take you to the view screen. One-key commands are acted upon immediately. No [RETURN] is needed.

[**B**] Returns you to the menu at any time.

[**G**] Switches you between GR. 7+ and GR. 8. GR. 7+ offers four colors (counting the background), changed with the [C] selection (below).

[**C**] Alters the GR. 7+ color registers. The message line at the bottom will indicate the current color number (0–3), its current hue and luminence values, plus the word Default.

The keys [**C**], [**H**], [**L**] increment the color, hue, and luminence. This can be used while in GR. 8. But the effect may be misleading because the GR. 7+ registers and the current screen registers are being altered but not the GR. 8 default values. The [**D**] key resets all GR. 7+ registers to default values. These values are updated each time you load a data file. Also, none of your playing will affect the menu screen colors, since these are main-

continued on next page

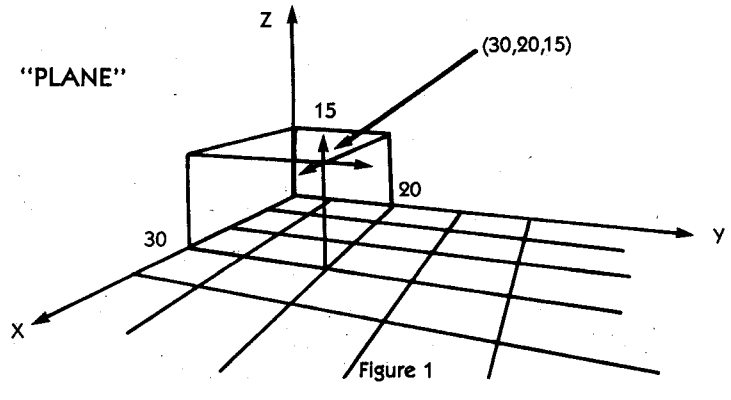


Figure 1

tained separately. Any other key terminates this routine.

[M] Magnifies the object. This is initially set at 4 and wraps to 1 when incremented past 9. You won't see the effect until the picture is redrawn by pressing [SPACE].

[V] Changes the view between perspective and orthogonal. Perspective, which emulates our vision, takes into account the distance from the eye, whereas orthogonal is used in drafting and engineering.

Figures 2 and 3 show the difference between perspective and orthogonal projections.

PREROTATE

These selections let you rotate your object about any of the three X, Y and Z axes. The message line shows the values of **rx** (rotate X), **ry** (rotate Y), **rz** (rotate Z), and **ri** (rotational increment). Each time the [X], [Y] or [Z] keys are pressed, the object rotates about the chosen axis in **ri** increments. The rotations are about axes that pass through the focus point.

The [I]/[J] keys increment/decrement the value of **ri** in degrees. Negative values of **ri** make rotations go in the opposite direction.

POINT OF VIEW

The following commands affect your dimensional view of the object.

[3] Fix EYE/dist. The eye coordinates are controlled by your joystick. Selections [1]-[4] use the same joystick scheme: Left/right alters the X coordinate, up/down alters the Y, and up/down while holding the trigger alters the Z. In selection [3], left/right with the trigger pressed controls the distance. Press [SPACE] to draw your object from this new eye location.

Remember that the eye coordinates are relative to the focus point (see [4] below) and only establish the viewing *direction* in the orthogonal view. The eye-object *distance* is important only in the perspective view. Keep the distance large to avoid distortion.

[4] Change FOCUS. The focus is the point in space at which the eye is aimed and through which all the rotation axes pass. It is normally on or near the object being studied and will be mapped to center screen (**cx,cy**). Move the flashing dot with your joystick. More importantly, watch its coordinates. Use [SPACE] to set your choice.

[5] Change CENTER. This alters **cx** and **cy**, shifting the object. These are actual screen coordinates (0,0 is the upper left). Use [SPACE] to set your choice and see the effect.

[0] Resets the center, eye, focus, magnification, and prerotation values to defaults used at start-up.

I/O

[D] Lists up to 22 data files in the menu window, assuming they have ".V3D" extenders. This is also done automatically at start-up and after each successful save.

[L] Loads a data file from disk. Answer the input prompt with a filename only. The program supplies the "D:" prefix and a ".V3D" extender. Upon hitting [RETURN] you'll see the full filespec. Press [L] again to accomplish the load. Any other key will abort the process.

[S] Saves data to a disk file. The process is the same as the above [L] load.

[P] Outputs to your printer. After pressing [P] you may choose to print the picture data [D] or the picture [P]. The picture is produced by a short screen dump for a Gemini 10X. You'll get best results by printing the GR. 8 picture.

To alter the printout procedure for your own printer, examine the **Prnt** procedure in the PRINTIO.ACT file and adapt accordingly. The **st** array contains printer control codes 26, 51, 16 which, on the Gemini, set the line feed to 16/144 inches. In the **pre** array, the 27, 75, 192, 0 mean print normal-density graphics (60 dots/inch) using 192+256*0 characters. If you have an Epson FX-80, for example, you need only change the line feed commands: Change the 16 to 24 in the **st** array, and later in the procedure at **st(3)=16**. Also, change the 20 to 30 in **st(3)=20**

3-D DRAWING

It's not easy to draw in 3 dimensions. The easiest way to learn is simply to try it. Concentrating on the changing coordinates in the message line may be easier than watching the dots and lines on the screen.

However, before you start, you may wish to save the object currently in memory. The process is easier to understand if you use the EDIT command, [2], to display a blank screen. Each time you press the [SPACE] bar, the screen will step through the drawing process of the object in memory, showing you how to construct a drawing.

To get started on your own, press [0] to use default values. To create a blank screen, press [2] then [1]. The joystick moves a flashing dot, whose coordinates appear in the bottom line. Position the cursor where you want it, and establish that point by pressing the [SPACE] bar. Your current updated point number will be displayed in the bottom line. Next, move the cursor to your second location, press [P] to switch from "Plot" to "LineTo" and press [SPACE] to draw the line.

For starters, keep it simple, or try editing a sample drawing. (The program can take up to 200 data points.) To edit a previous drawing, press [1] to ADD points and lines. As you step through the drawing by pressing [SPACE], you can change any of the values at any point, or you can begin adding to the last points. You can, of course, save your object to disk at any time.

DATA STORAGE

At this point, you need to understand a little about how data for your 3-D object is stored. The INTeger array **P** contains all the information in the following format:

P=[n:x y z:d:x y z:c: . . . . . :x y z:c: . . . . .]

P(0)=n is the number of data points in your object. The next four integers contain EYE data. The first three indicate the direction away from the FOCUS, and the fourth gives the distance.

The following four integers contain the three space coordinates of the focus point and a presently unused

value. These nine integers are followed by n data sets for your object. Each is made up of four integers containing the three space coordinates for a point and a fourth coded message. The encoding of the fourth integer is given by c= color + 16*p, where p=0 for "LineTo" and p=1 for "Plot".

SAMPLE DATA

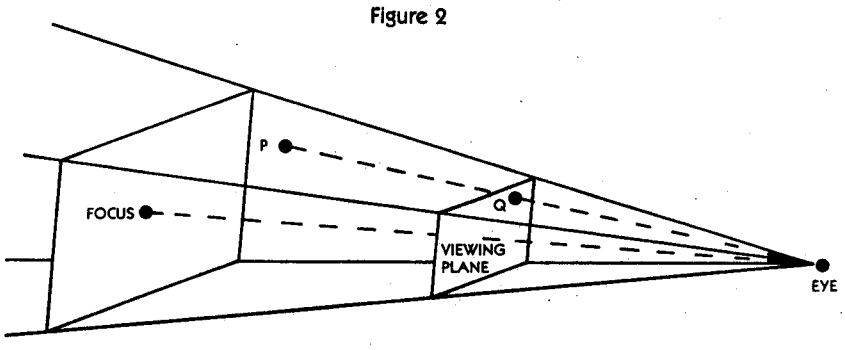
You can enter the data in figure 4 in the ADD mode to create a house with windows and a red chimney. Press [2], then [1] to clear memory. Now use your joystick to get the coordinates in the message line to match those of the first point in the example. Hit [C] and [P] as needed and set the data by hitting [SPACE]. Now do the same for the second point in the example, etc.

You can even do a little at a time. Just save the portion you've done. Next time load in this file, press [1] and continue from where you left off. A couple of examples will show you how to read the notation. "10 20 15:P2" means to Plot (10,20,15) in color 2. Whereas "20 20 30:L3" denotes a color 3 LineTo (20,20,30). Each example has suggested EYE and FOCUS data.

Longtime **Antic** *contributor Paul Chabot is a professor of mathematics and computer science at California State University, Los Angeles. He wrote "Splash In ACTION!" in our April 1985 issue.*

Listing on page 54.

PERSPECTIVE PROJECTION
Figure 2



ORTHOGONAL PROJECTION
Figure 3

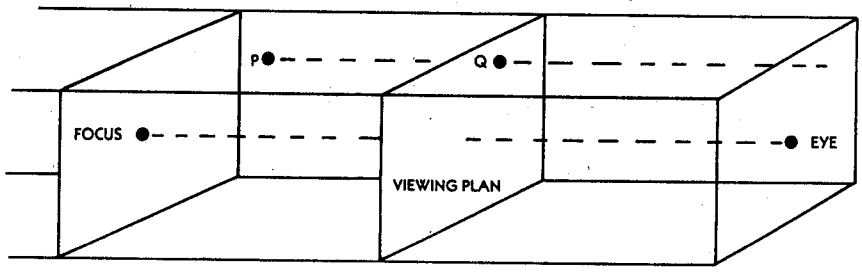


Figure 4

```
HOUSE : eye=(10 3 5:160) focus=(15 30 20)
(0 0 0:P2)      (30 0 0:L2)     (30 60 0:L2)
(0 60 0:L2)     (0 0 0:L2)      (0 0 40:L2)
(15 0 50:L2)    (30 0 40:L2)    (30 0 0:L2)
(30 60 0:P2)    (30 60 40:L2)   (15 60 50:L2)
(0 60 40:L2)    (0 60 0:L2)     (0 60 40:P3)
(0 0 40:L3)     (30 0 40:P3)    (30 60 40:L3)
(15 60 50:P2)   (15 0 50:L2)    (30 10 0:P3)
(30 10 25:L3)   (30 20 25:L3)   (30 20 0:L3)
(30 30 10:P3)   (30 30 25:L3)   (30 50 25:L3)
(30 50 10:L3)   (30 30 10:L3)   (30 40 10:P3)
(30 40 25:L3)   (10 60 10:P3)   (10 60 25:L3)
(20 60 25:L3)   (20 60 10:L3)   (10 60 10:L3)
(10 0 0:P1)     (10 -5 0:L1)    (20 -5 0:L1)
(20 0 0:L1)     (20 0 55:L1)    (20 -5 55:L1)
(20 -5 0:L1)    (10 0 0:P1)     (10 0 55:L1)
(10 -5 55:L1)   (10 -5 0:L1)    (10 0 55:P1)
(20 0 55:L1)    (10 -5 55:P1)   (10 -5 55:L1)
```

rotate and zoom 3-D images in ACTION

VIEW 3-D

Article on page 38.

LISTING 1

```
; GR78M (LISTING 1)

MODULE:INT xnow=[80],ynow=[90]
BYTE cnow=[1],key=764,ram=106,cur=752
BYTE ARRAY mask7=[64 16 4 1],clor=708
  ,mask8=[128 64 32 16 8 4 2 1],mask,row
CARD dlist=560,sa=88
CARD ARRAY adr(192):CARD POINTER mes

PROC Kolor(BYTE c) RETURN
PROC Dot(INT x,y) RETURN
PROC Dit(INT x,y) RETURN

PROC LineTo(INT x,y)
INT dx,dy,xf,yf,a,b,t,i
Dot(xnow,ynow)
IF x=xnow AND y=ynow THEN RETURN FI
IF x>xnow THEN dx=x-xnow:xf=1
ELSE dx=xnow-x:xf=-1 FI
IF y>ynow THEN dy=y-ynow:yf=1
ELSE dy=ynow-y:yf=-1 FI
x=xnow:y=ynow
IF dx>dy THEN a=dy+dy:t=a-dx:b=t-dx
  FOR i=1 TO dx DO x==+xf
    IF t<0 THEN t==+a
    ELSE t==+b:y==+yf
    FI Dot(x,y)
  OD
ELSE a=dx+dx:t=a-dy:b=t-dy
  FOR i=1 TO dy DO y==+yf
    IF t<0 THEN t==+a
    ELSE t==+b:x==+xf
    FI Dot(x,y)
  OD
FI xnow=x:ynow=y:RETURN

PROC Gr78ON():BYTE i:BYTE ARRAY dl
Graphics(8+16):adr(0)=sa:dl=dlist
FOR i= 1 TO 191 DO adr(i)=adr(i-1)+40 OD
dl==-4:dl(0)=112:dl(1)=80:dl(2)=16
FOR i=3 TO 198 DO dl(i)=dl(i+4) OD
dl(199)=16:dl(200)=66:mes=dl+201
dl(204)==-4:dlist=dl:RETURN

PROC Kolor7(BYTE c):BYTE i
c==& 3:cnow=c
FOR i=0 TO 3 DO mask(3-i)=c:c==LSH 2 OD
RETURN

PROC Dot7(INT x,y):BYTE xb,xr
BYTE ARRAY pre=[63 207 243 252]
IF x<0 OR x>159 THEN RETURN FI
IF y<0 OR y>191 THEN RETURN FI
xb=x RSH 2:xr=x AND 3:row=adr(y)
row(xb)==& pre(xr) % mask(xr):RETURN

PROC Dit7(INT x,y):BYTE xb,xr
IF x<0 OR x>159 THEN RETURN FI
IF y<0 OR y>191 THEN RETURN FI
xb=x RSH 2:xr=x AND 3:row=adr(y)
row(xb)==! mask(xr):RETURN

PROC Gr7(BYTE ARRAY d):BYTE i
mask=mask7:Kolor=Kolor7:Dot=Dot7
Dit=Dit7:d(3)=78:d(99)=78
FOR i=6 TO 98 DO d(i)=14 OD
FOR i=102 TO 198 DO d(i)=14 OD:RETURN

PROC Kolor8(BYTE c):BYTE i
cnow=c & 3:IF c>1 THEN c=1 FI
FOR i=0 TO 7 DO mask(7-i)=c:c==LSH 1 OD
RETURN

PROC Dot8(INT x,y):BYTE xb,xr
BYTE ARRAY
  pre=[127 191 223 239 247 251 253 254]
IF x<0 OR x>319 THEN RETURN FI
IF y<0 OR y>191 THEN RETURN FI
xb=x RSH 3:xr=x AND 7:row=adr(y)
row(xb)==& pre(xr) % mask(xr):RETURN

PROC Dit8(INT x,y):BYTE xb,xr
IF x<0 OR x>319 THEN RETURN FI
IF y<0 OR y>191 THEN RETURN FI
xb=x RSH 3:xr=x AND 7:row=adr(y)
row(xb)==! mask(xr):RETURN

PROC Gr8(BYTE ARRAY d):BYTE i
mask=mask8:Kolor=Kolor8:Dot=Dot8
Dit=Dit8:d(3)=79:d(99)=79
FOR i=6 TO 98 DO d(i)=15 OD
FOR i=102 TO 198 DO d(i)=15 OD:RETURN
```

LISTING 2

```
; MISC1 (LISTING 2)

MODULE:BYTE st:INT ARRAY
  jx=[1 1 1 1 1 2 2 2 1 0 0 0 1 1 1 1]
  ,jy=[1 1 1 1 1 2 0 1 1 2 0 1 1 2 0 1]
BYTE ARRAY b="                    "

PROC SetJxJy(BYTE i)
FOR i=0 TO 15 DO jx(i)==-1:jy(i)==-1 OD
RETURN

PROC Pb(BYTE i):b(0)=i:Print(b):RETURN

PROC Wait(CARD w,j)
FOR j=0 TO w DO w==+1:w==-1 OD RETURN

; TRIG
MODULE:BYTE ARRAY si(91)

PROC SetTrig(BYTE t INT y)
FOR t=0 TO 90 DO y=(t*t)/45
  y=(y*t)/5:y=100*t-y:y==/45:si(t)=y
OD RETURN

INT FUNC sin(INT t,y):t==MOD 360
IF t<91 THEN y=si(t)
ELSEIF t<181 THEN y=si(180-t)
ELSEIF t<271 THEN y=-si(t-180)
ELSE y=-si(360-t) FI RETURN(y)

INT FUNC cos(INT t,y):t==MOD 360
IF t<91 THEN y=si(90-t)
ELSEIF t<181 THEN y=-si(t-90)
```

```
ELSEIF t<271 THEN y=-Si(270-t)
ELSE y=Si(t-270) FI RETURN(y)


; VECTOR
INT FUNC ABS(INT x)
IF x<0 THEN x=-x FI RETURN(x)

INT FUNC SQR(INT x):INT y
IF x=0 THEN RETURN(0) FI:x=ABS(x):y=0
DO y==+1:IF y*y+y>x THEN RETURN(y) FI OD

INT FUNC Vdot(INT ARRAY v,w):INT x
x=v(0)*w(0):x==+v(1)*w(1)
x==+v(2)*w(2):RETURN(x)

PROC Vprod(INT ARRAY v,w,u)
u(0)=v(1)*w(2):u(0)==-v(2)*w(1)
u(1)=v(2)*w(0):u(1)==-v(0)*w(2)
u(2)=v(0)*w(1):u(2)==-v(1)*w(0)
RETURN

PROC Normize(INT ARRAY v):INT i,j,s
i=ABS(v(0))
j=ABS(v(1)):IF i<j THEN i=j FI
j=ABS(v(2)):IF i<j THEN i=j FI
IF i>100 THEN j=1+i/100
  FOR i=0 TO 2 DO v(i)==/j OD
FI
FOR j=0 TO 1 DO s=Vdot(v,v):s=SQR(s)
  FOR i=0 TO 2 DO v(i)=v(i)*128/s OD
OD RETURN
```

LISTING 3

```
; COLORS (LISTING 3)

MODULE:BYTE ARRAY dfault(5),CP
  ,C7=[52 24 130 194 0]
  ,C8=[52 26 0 194 208]
  ,CM=[52 24 194 130 80]

PROC IncC():BYTE i
i=cnow:cnow==+1:Position(10,23)
IF i=3 THEN i=4:cnow=0 FI
Kolor(cnow):i=clor(i):PrintB(cnow)
Position(18,23):PrintB(i RSH 4)
Put(' ):Position(27,23)
PrintB(i & 14):Put(' ):RETURN

PROC IncHue():BYTE i,j
IF cnow=0 THEN i=4 ELSE i=cnow-1 FI
j=clor(i) RSH 4
j==+1:IF j>15 THEN j=0 FI
Position(18,23):PrintB(j):Put(' )
clor(i)=(j LSH 4)+(clor(i) & 14)
C7(i)=clor(i):RETURN

PROC IncLum():BYTE i,j
IF cnow=0 THEN i=4 ELSE i=cnow-1 FI
j=clor(i) & 14
j==+2:IF j>15 THEN j=0 FI
Position(27,23):PrintB(j):Put(' )
clor(i)=(clor(i) & 240)+j
C7(i)=clor(i):RETURN

PROC DfaultC():BYTE i
FOR i=0 TO 4 DO C7(i)=dfault(i)
  clor(i)=C7(i)
OD RETURN


PROC SetDfault():BYTE i
FOR i=0 TO 4 DO dfault(i)=C7(i) OD:RETURN

PROC CPon():BYTE i
FOR i=0 TO 4 DO clor(i)=CP(i) OD:RETURN

PROC CMon():BYTE i
FOR i=0 TO 4 DO clor(i)=CM(i) OD:RETURN

PROC FixCol():IncC()
DO WHILE key=255 DO OD
  IF key=18 THEN key=255:IncC()
  ELSEIF key=57 THEN key=255:IncHue()
  ELSEIF key=0 THEN key=255:IncLum()
  ELSEIF key=58 THEN key=255:DfaultC()
  ELSE EXIT FI
OD RETURN
```

LISTING 4

```
; DRAW3D (LISTING 4)

MODULE:BYTE vflag=[0],gflag=[7]
INT sx,sy,mag=[3],cx=[80],cy=[90]
  ,rx=[0],ry=[0],rz=[0],ri=[30]
CARD sa1,d11,d12,lin16,lin17,lin18
  ,lin19,lin20,lin21,lin22,lin23,lin15
INT ARRAY P(809),eye,foc,R(9),E(9),M(9)
  ,Q=[21:10 5 5:100:0 0 0:0:
  50 0 0:18:0 0 0:2:0 50 0:2:
  0 0 0:18:0 0 40:2:10 0 0:17:
  10 50 0:1:20 50 0:17:20 0 0:1:
  30 0 0:17:30 50 0:1:40 50 0:17:
  40 0 0:1:0 10 0:19:50 10 0:3:
  50 20 0:19:0 20 0:3:0 30 0:19:
  50 30 0:3:50 40 0:19:0 40 0:3]

PROC FixP(INT ARRAY Q):INT i,j
Zero(P,1618):j=4*Q(0)+8
FOR i=0 TO j DO P(i)=Q(i) OD
eye=P+2:foc=P+10:RETURN

PROC Rot(INT ARRAY v):INT x,y,z,s,c
y=v(1)
v(1)=y*cos(rx)/128:v(2)=y*sin(rx)/128
x=v(0):z=v(2):s=sin(ry):c=cos(ry)
v(0)=(x*c-z*s)/128:v(2)=(x*s+z*c)/128
x=v(0):y=v(1):s=sin(rz):c=cos(rz)
v(0)=x*c-y*s:v(1)=x*s+y*c:v(0)==/128
v(1)==/128:Normize(v):RETURN

PROC FixR():INT ARRAY v(3),w(3),u(3)
v(0)=128:v(1)=0:v(2)=0:Rot(v)
w(0)=0:w(1)=128:w(2)=0:Rot(w)
Vprod(v,w,u):Normize(u)
R(0)=v(0):R(1)=v(1):R(2)=v(2)
R(3)=w(0):R(4)=w(1):R(5)=w(2)
R(6)=u(0):R(7)=u(1):R(8)=u(2):RETURN

PROC FixE():INT s
E(6)=eye(0):E(7)=eye(1):E(8)=eye(2)
Normize(E+12)
IF E(0)=0 THEN E(3)=0:E(4)=0:E(5)=128
ELSEIF E(6)=0 AND E(7)=0 THEN
  E(3)=0:E(4)=128:E(5)=0
ELSE E(3)=-E(6):E(4)=-E(7)
  E(5)=E(6)*E(6):E(5)==+E(7)*E(7)
  E(5)==/E(8):Normize(E+6)
  IF E(8)<0 THEN E(3)=-E(3):E(4)=-E(4)
    E(5)=-E(5)
  FI
FI Vprod(E+6,E+12,E):Normize(E):RETURN

PROC FixM()
M(0)=Vdot(R,E):M(3)=Vdot(R,E+6)
M(1)=Vdot(R+6,E):M(4)=Vdot(R+6,E+6)
```

continued on next page

```
OD Ufoc(foc):RETURN

PROC JoyE():INT x,y,z
x=eye(0):y=eye(1):z=eye(2)
DO st=Stick(0):Ueye()
  WHILE Strig(0)=0 DO st=Stick(0)
    eye(2)==-jy(st):eye(3)==+jx(st)
    Ueye()
  OD
  IF key=33 THEN key=255
    FixE():FixM():CLR():Draw(P)
    x=eye(0):y=eye(1):z=eye(2)
  ELSEIF key<255 THEN EXIT FI
  eye(0)==+jx(st):eye(1)==-jy(st)
OD eye(0)=x:eye(1)=y:eye(2)=z:Ueye()
RETURN

PROC JoyC():INT x,y
IF cnow=0 THEN Kolor(1) FI
x=cx:y=cy:Dit(x,y)
DO st=Stick(0):Ucen(x,y):Dit(x,y)
  IF key=33 THEN key=255
    cx=x:cy=y:CLR():Draw(P)
  ELSEIF key<255 THEN EXIT FI
  x==+jx(st):y==+jy(st):Dit(x,y)
OD Ucen(cx,cy):RETURN
```

LISTING 7

```
; DISKIO (LISTING 7)

MODULE:BYTE err
BYTE ARRAY fin(16),abort=" ABORTED "

PROC MyErr(BYTE e)
Position(1,16):Print("ERROR ")
PrintB(e):err=1:mes^=lin21
Position(12,21):PrintB(e)
WHILE key=255 DO OD key=21:RETURN

PROC CIO=$E456(BYTE a,x)

PROC IO2(BYTE cmd CARD buf,len)
BYTE IOcmd=866 ;7-LOAD 11-SAVE
CARD IObuf=868,IOlen=872
IOcmd=cmd:IObuf=buf:IOlen=len
CIO(0,32):RETURN

PROC Dir():BYTE i,j,lft=82
BYTE ARRAY a(10),f(9)
lft=22:Position(22,3)
FOR i=1 TO 11 DO Pb(16):PutE() OD
Position(22,3):j=0
Close(2):Open(2,"D:*.V3D",6,0)
FOR i=1 TO 22 DO InputSD(2,a)
  IF a(0)=16 THEN EXIT FI
  SCopyS(f,a,3,10):Print(f)
  IF j=0 THEN j=1:Put(' )
  ELSE j=0:PutE(),FI
OD Position(22,14):Print(a)
Close(2):lft=1:RETURN

PROC Inp(BYTE ARRAY f):BYTE i
BYTE ARRAY a(10)
SCopy(f,"D:                ")
Position(23,16):Pb(15)
Position(23,16):InputS(a)
i=a(0)+3:IF i>11 THEN i=11 FI
SAssign(f,a,3,10):SAssign(f,".V3D",i,14)
Position(23,16):Print(f):RETURN

PROC SaveP():BYTE k:CARD n,t
Position(1,16):Print("S A V E ")
Inp(fin)
Position(1,16):Print("[S]-SAVE ")
WHILE key=255 DO OD k=key:key=255
IF k<>62 THEN Position(1,16):Put('S)
  Print(abort):RETURN
FI t=Error:Error=MyErr:err=0
n=8*P(0)+18:Close(2):Open(2,fin,8,0)
IF err>0 THEN Close(2):Error=t:RETURN FI
IO2(11,P,n):IO2(11,C7,5):Close(2)
Position(1,16):Print(" SAVED  ")
Error=t:Dir() RETURN

PROC LoadP():BYTE k:CARD n,t
Position(1,16):Print("L O A D ")
Inp(fin)
Position(1,16):Print("[L]-LOAD ")
WHILE key=255 DO OD k=key:key=255
IF k<>0 THEN Position(1,16):Put('L)
  Print(abort):RETURN
FI t=Error:Error=MyErr:err=0
Close(2):Open(2,fin,4,0)
IF err>0 THEN Close(2):Error=t:RETURN FI
IO2(7,P,2):n=8*P(0)+16
IO2(7,P+2,n):IO2(7,C7,5):Close(2)
Position(1,16):Print(" LOADED  ")
Error=t:SetDfault():CPon()
Position(30,19):Pb(6):RETURN
```

LISTING 8

```
; PRINTIO (LISTING 8)

PROC Ppt(INT ARRAY v)
PrintD(2,"("):PrintID(2,v(0))
PrintD(2," "):PrintID(2,v(1))
PrintD(2," "):PrintID(2,v(2))
PrintD(2,":"):PrintID(2,v(3))
PrintD(2,") "):RETURN

PROC Prnt():BYTE i,j,k:CARD n,t
BYTE ARRAY a(13),st=[3 27 51 16]
  ,pre=[4 27 75 192 0],s,d(193)
Position(1,16):Print("PRINTING ")
WHILE key=255 DO OD k=key:key=255
IF k<>10 AND k<>58 THEN Position(1,24)
  Put('P):Print(abort):RETURN
FI:t=Error:Error=MyErr:err=0
Close(2):Open(2,"P:",8,0)
IF err>0 THEN Close(2):Error=t:RETURN FI
b(0)=12:SCopy(a,b):SCopyS(a,fin,3,14)
PrintDE(2," "):PrintDE(2,a)
IF err>0 THEN Close(2):Error=t:RETURN FI
IF k=10 THEN st(3)=16:PrintDE(2,st)
  s=sa1:d(0)=192
  FOR i=0 TO 39 DO n=7640+i
    FOR j=1 TO 192 DO d(j)=s(n):n==-40 OD
    PrintD(2,pre):PrintDE(2,d)
  OD
ELSE st(3)=20:PrintDE(2,st)
  PrintD(2,"eye="):Ppt(eye)
  PrintD(2," focus="):Ppt(foc)
  PrintDE(2," "):i=0:j=0:n=P+10
  DO i==+1:IF i>P(0) THEN EXIT FI
    j==+1:n==+8:Ppt(n)
    IF j>2 THEN j=0:PutDE(2) FI
  OD PrintDE(2," ")
FI Position(1,16):Print(" PRINTED ")
Error=t:Close(2):RETURN
```

LISTING 9

```
; MENU3D (LISTING 9)

PROC Menu():BYTE i,lft=82:BYTE ARRAY dl
dl=dl2-5:dlist=dl:dl2=dlist
```

continued on next page

```
M(2)=Vdot(R+12,E):M(5)=Vdot(R+12,E+6)
Normize(M):Normize(M+6)
Vprod(M,M+6,M+12):Normize(M+12):RETURN

PROC Maksxsy(INT ARRAY v):BYTE i
INT px,py,pz,t,d:INT ARRAY w(3)
FOR i=0 TO 2 DO w(i)=v(i)-foc(i) OD
IF vflag=1 THEN px=Vdot(w,M)/128
  py=Vdot(w,M+6)/128
  sx=cx+mag*px/2:sy=cy-mag*py/2
ELSE d=eye(3):t=mag*d/8
  px=Vdot(w,M)/128:py=Vdot(w,M+6)/128
  pz=Vdot(w,M+12)/128
  d==-pz:IF d<4 THEN d=4 FI:d==/4
  sx=t*px/d:sy=t*py/d:sx==+cx:sy=cy-sy
FI RETURN

PROC CLR():Zero(sa1,7680):RETURN

PROC Draw(INT ARRAY P):BYTE i
INT ARRAY pt
pt=P+10
FOR i=1 TO P(0) DO pt==+8 Maksxsy(pt)
  Kolor(pt(3) & 15)
  IF pt(3)<16 THEN LineTo(sx,sy)
  ELSE Dot(sx,sy):xnow=sx:ynow=sy FI
OD RETURN
```

LISTING 5

```
; UPDATES (LISTING 5)

PROC Uview():Position(22,15)
vflag==+1:IF vflag>1 THEN vflag=0 FI
IF vflag=0 THEN Print("[redacted]")
ELSE Print("[redacted]") FI:RETURN

PROC Ueye():BYTE i:Position(5,17)
FOR i=0 TO 2 DO
  IF eye(i)<-10 THEN eye(i)=-10
  ELSEIF eye(i)>10 THEN eye(i)=10 FI
  PrintI(eye(i)):Put(' )
OD Position(24,17)
IF eye(3)>200 THEN eye(3)=200
ELSEIF eye(3)<10 THEN eye(3)=10 FI
PrintI(eye(3)):Put(' ):RETURN

PROC Umag():Position(37,17):mag==+1
IF mag>9 THEN mag=1 FI:PrintI(mag):RETURN

PROC Urot():Position(12,18):PrintI(rx)
Put(' ):PrintI(ry):Put(' )
PrintI(rz):Pb(2)
FixR():FixM():CLR():Draw(P):RETURN

PROC Uri():Position(35,18)
PrintI(ri):Put(' ):RETURN

PROC Ucen(INT x,y):Position(8,19)
PrintI(x):Put(' )
PrintI(y):Put(' ):RETURN

PROC Ufoc(INT ARRAY v):BYTE i
Position(24,19)
FOR i=0 TO 2 DO PrintI(v(i)):Put(' ) OD
RETURN

PROC Upt(INT n):Position(5,22):PrintI(n)
Put(' ):IF n<100 THEN Put(' ) FI
Position(30,22):Pb(8):RETURN

PROC Ucxyz(INT ARRAY pt):BYTE i
Position(15,22):PrintB(pt(3) & 3)
IF pt(3)<16 THEN Print("  LineTo ")
ELSE Print("   Plot  ") FI
FOR i=0 TO 2 DO PrintI(pt(i)):Put(' ) OD
RETURN

PROC Ugr():Position(1,15)
IF gflag=7 THEN gflag=8:cx==+80:CP=C8
  Print("xox GRAPHICS 8 xox"):Gr8(d11)
ELSE gflag=7:cx==-80:CP=C7
  Print("x GRAPHICS 7PLUS x"):Gr7(d11)
FI Ucen(cx,cy)
CPon():CLR():Draw(P):RETURN

PROC UReset():BYTE i
Position(5,17):Pb(12)
Position(12,18):Pb(12)
Position(8,19):Pb(9)
Position(24,19):Pb(14)
FOR i=1 TO 8 DO P(i)=Q(i) OD
mag=3:rx=0:ry=0:rz=0:ri=30:FixE()
Ueye():Ufoc(foc):Umag():Urot():Uri()
cy=90:cx=160:gflag=8:Ugr():RETURN
```

LISTING 6

```
; STICK3D (LISTING 6)

PROC JoyD(INT n) ;0-EDIT,1-ADD
BYTE i,k,f:INT ARRAY pt
IF n=0 THEN f=0 ELSE f=1:n=P(0) FI
n==+1:pt=P+10+8*n:Upt(n):Ucxyz(pt)
Kolor(pt(3) & 15):Maksxsy(pt):Dit(sx,sy)
DO IF n>200 THEN EXIT FI
  WHILE Strig(0)=0 DO
    st=Stick(0):Dit(sx,sy)
    IF st<15 THEN pt(2)==-jy(st)
      Maksxsy(pt):Ucxyz(pt)
    FI Dit(sx,sy)
  OD
  st=Stick(0):Dit(sx,sy)
  IF key<255 THEN k=key:key=255
    IF k=33 THEN n==+1:Upt(n)
      IF pt(3)<16 THEN LineTo(sx,sy)
      ELSE Dot(sx,sy):xnow=sx:ynow=sy FI
      IF f=1 THEN
        FOR i=0 TO 3 DO pt(4+i)=pt(i) OD
      FI pt==+8:Kolor(pt(3) & 15)
      Maksxsy(pt):Ucxyz(pt)
    ELSEIF k=10 THEN pt(3)==! 16:Ucxyz(pt)
    ELSEIF k=18 THEN IncC()
      pt(3)=(pt(3) & 16)+cnow:Ucxyz(pt)
    ELSE key=k:EXIT FI
  FI
  IF st<15 THEN pt(0)==+jx(st)
    pt(1)==-jy(st):Maksxsy(pt):Ucxyz(pt)
  FI Dit(sx,sy)
OD P(0)=n-1:RETURN

PROC JoyF():BYTE i:INT ARRAY pt(3)
IF cnow=0 THEN Kolor(1) FI
FOR i=0 TO 2 DO pt(i)=foc(i) OD
Maksxsy(pt):Dit(sx,sy):Ufoc(pt)
DO
  WHILE Strig(0)=0 DO
    st=Stick(0):Dit(sx,sy)
    IF st<15 THEN pt(2)==-jy(st)
      Maksxsy(pt):Ufoc(pt)
    FI Dit(sx,sy)
  OD
  st=Stick(0):Dit(sx,sy)
  IF key=33 THEN key=255
    FOR i=0 TO 2 DO foc(i)=pt(i) OD
    CLR():Draw(P):Maksxsy(pt)
  ELSEIF key<255 THEN EXIT FI
  IF st<15 THEN pt(0)==+jx(st)
    pt(1)==-jy(st):Maksxsy(pt):Ufoc(pt)
  FI Dit(sx,sy)
```

```
FOR i=0 TO 12 DO dl(i)=dl(5+i) OD
dl(1)=48:dl(2)=48:dl(3)=71:dl(6)=6
dl(8)=48:dl(22)=6:dl(23)=6:dl(24)=16
dl(26)=16:dl(28)=16:dl(30)=16
dl(32)=16:dl(35)=dl&255:lft=1
Print(" v i e w   3 D   ")
PrintE("%0%0%0%0%0%0%0%0%0%0")
PrintE("       Paul Chabot        1985")
PrintE(" PreRotate  G  Gr7/8    DIRECTORY     ")
PrintE(" X  Inc rx  V  View  ")
PrintE(" Y  Inc ry  M  Mag.  ")
PrintE(" Z  Inc rz  C  Color ")
PrintE(" I  Inc ri     I / O     ")
PrintE(" J  Dec ri  D L S P  ")
PrintE("    Joystick Trigger     ")
PrintE(" 1 ADD Points/Lines  ")
PrintE(" 2 EDIT Points/Lines ")
PrintE(" 3 Change EYE/dist.  ")
PrintE(" 4 Change FOCUS      ")
PrintE(" 5 Change CENTER     ")
PrintE(" 0 Reset ALL Values  ")
Print("x GRAPHICS 7PLUS x ")
PrintE("  Perspective View ")
PrintE("              filename>>")
PrintE("eye            dist.          mag")
PrintE("PreRotated                    ri=")
PrintE("Center             Focus")
PrintE("  [SPACE]    Picture      [R]   Menu    ")
PrintE("   ERROR          [any key] - MENU")
PrintE("pt#      color")
Print("  Color   Hue    Lum    Default")
lin15=sa+600
lin16=sa+640:lin17=sa+680:lin18=sa+720
lin19=sa+760:lin20=sa+800:lin21=sa+840
lin22=sa+880:lin23=sa+920:RETURN
```

LISTING 10

```
;  VIEW3D (LISTING 10)
;    by   Paul Chabot
;      ANTIC Magazine

INCLUDE "D:GR78M.ACT"
INCLUDE "D:MISC1.ACT"
INCLUDE "D:COLORS.ACT"
INCLUDE "D:DRAW3D.ACT"
INCLUDE "D:UPDATES.ACT"
INCLUDE "D:STICK3D.ACT"
INCLUDE "D:DISKIO.ACT"
INCLUDE "D:PRINTIO.ACT"
INCLUDE "D:MENU3D.ACT"

PROC Setup():BYTE i
SetTrig():SetJxJy():SetDfault():FixP(0)
Gr78ON():dl1=dlist:sa1=sa:ram==-33
Graphics(0):ram==+33:dl2=dlist:cur=1
CMon():Menu():Dir():UReset():dlist=dl1
mes^=lin20
RETURN

PROC Main():BYTE k:Setup()
DO WHILE key=255 DO OD k=key:key=255
  dlist=dl1:CPon()
  IF k=31 THEN mes^=lin22 ;1
    JoyD(1)
  ELSEIF k=30 THEN mes^=lin22 ;2
    CLR():JoyD(0)
  ELSEIF k=26 THEN mes^=lin17 ;3
    JoyE()
  ELSEIF k=24 THEN mes^=lin19 ;4
    JoyF()
  ELSEIF k=29 THEN mes^=lin19 ;5
    JoyC()
  ELSEIF k=50 THEN mes^=lin20 ;0
    UReset()
  ELSEIF k=22 THEN mes^=lin18 ;X
    rx=(rx+ri+360) MOD 360:Urot()
  ELSEIF k=43 THEN mes^=lin18 ;Y
    ry=(ry+ri+360) MOD 360:Urot()
  ELSEIF k=23 THEN mes^=lin18 ;Z
    rz=(rz+ri+360) MOD 360:Urot()
  ELSEIF k=13 THEN mes^=lin18 ;I
    ri==+1:IF ri>45 THEN ri=45 FI:Uri()
  ELSEIF k=1 THEN mes^=lin18 ;J
    ri==-1:IF ri<-45 THEN ri=-45 FI:Uri()
  ELSEIF k=18 THEN mes^=lin23 ;C
    FixCol()
  ELSEIF k=16 THEN mes^=lin15 ;V
    Uview():CLR():Draw(P)
  ELSEIF k=37 THEN mes^=lin17 ;M
    Umag()
  ELSEIF k=58 THEN dlist=dl2 ;D
    CMon():Dir()
  ELSEIF k=0 THEN mes^=lin16 ;L
    LoadP():Ueye():Ufoc(foc):FixE()
    FixM():CLR():Draw(P):mes^=lin20
  ELSEIF k=62 THEN mes^=lin16 ;S
    SaveP():mes^=lin20
  ELSEIF k=61 THEN mes^=lin15 ;G
    Ugr()
  ELSEIF k=10 THEN mes^=lin16 ;P
    Prnt():mes^=lin20
  ELSEIF k=33 THEN mes^=lin20 ;SPACE
    CLR():Draw(P)
  ELSE dlist=dl2:CMon()
  FI
OD
```

enhance your picture details like NASA does!

PIXEL SCANNER

Article on page 14.

LISTING 1

Don't type the TYPO II Codes!

```
WY 10 REM GTIA PIXEL AVERAGING
ON 20 REM BY LYN BUCHANAN
FW 30 REM (C) 1985, ANTIC PUBLISHING
ML 40 GOTO 300
IK 100 REM PIXEL AVERAGING ROUTINE
RM 110 REM SET AREA TO BE ENHANCED
VS 111 REM Y-COORDINATE RANGE
FV 112 FOR Y=1 TO 88
VP 113 REM X-COORDINATE RANGE
EL 114 FOR X=1 TO 77
FS 115 REM STOP ATTRACT MODE
JR 116 POKE 77,0
SX 120 REM LOCATE EACH PIXEL AND ASSIGN I
   TS VALUE TO A VARIABLE
DN 121 LOCATE X-1,Y-1,A
KP 122 LOCATE X-1,Y,B
DZ 123 LOCATE X-1,Y+1,C
JF 124 LOCATE X,Y-1,D
XF 125 LOCATE X,Y,E
JR 126 LOCATE X,Y+1,F
HV 127 LOCATE X+1,Y-1,G
OL 128 LOCATE X+1,Y,H
```