# Improving Handheld Scanner

by Ledger White

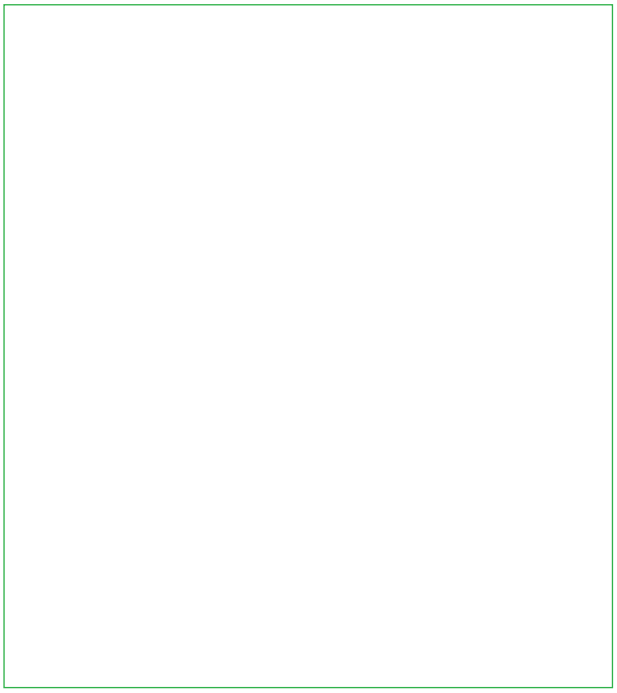
## INTRODUCTION

This article is the result of my experiences with the Scavenger hand held 105mm mono scanner, but the images produced by other makes suggest that they too can be improved. If you believe that mono scanners are not really good enough for producing grey scaled pictures then you are in for a couple of very pleasant surprises. The first is just how good an image you can get from such scanners, and the second is how unexpectedly simple it is. You'll be delighted!

The *Scanty* routine given here improves the output of mono scanners so much that users should feel confident in recommending them as highly worthwhile additions to an Arc system.

## WHY IMPROVE?

If you use one of these scanners you will be aware that when scanning mono images such as line drawings or illustrations the results are very good indeed. But you will also be aware that when converting grey scale images from the mono scan the results are usually disappointing. The results exhibit 'banding' and cannot be zoomed up or down without further degradation, and this makes them pretty well unusable for any DTP or imaging work. You might think that it is after all just a mono scanner and you cannot therefore expect much more. That's exactly what I thought until I reasoned that with equipment of this quality at 400 dpi, surely I should be able to get a much more convincing grey level image. After some experimentation, I discovered that I could get a dramatic improvement from a very simple correction.



Scanned image processed by Scanner

## HOW IT WORKS

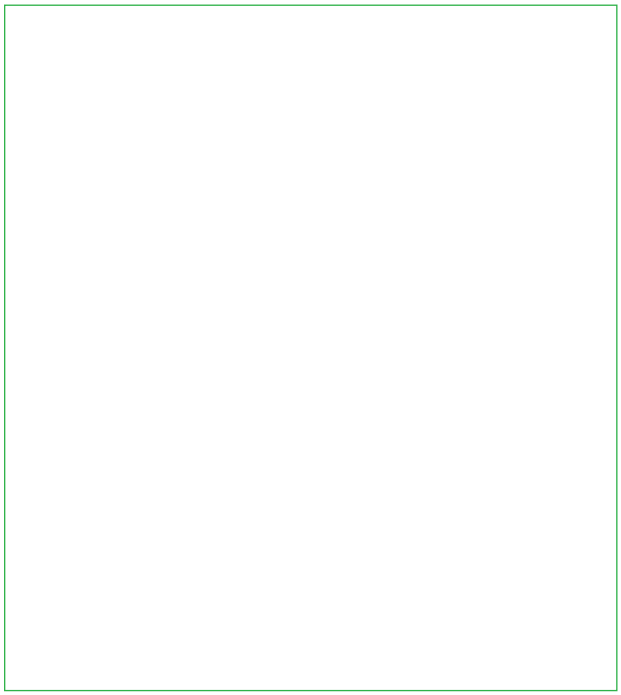
In 'photo' mode my hand held scanner produces an intermediate raw image as a sprite which can then be processed by 'anti-aliasing' into a final image of 16 grey levels. This 'anti-aliasing' process appears to have been misunderstood by some vendors and consequently their routines produce only an inferior image.

To see how the scanner handles images in a 'photo' mode, copy the raw output of your scanner into Paint, zoom up to about six or seven times, and set the grid on in a contrasting colour. Now you can examine the way in which the image is represented using only black and white. There are several things you notice:

1. The image is made up of structured patterns of dots within blocks.
2. A particular block of grey is always represented by the same pattern of dots.

3. The blocks are made up of 9 dots in 3x3 blocks.
4. You never get all of the pixels blacked in.
5. On darker shades the 3x3 blocks are dithered into 6x6 blocks.

All the manufacturers instructions and all the reviews I have seen say that the final 16 grey level image is produced by adding



Same image processed by Scanty

up set and unset pixels in a 4x4 block in the original. It will only take you a few minutes to convince yourself that this is quite the wrong way to do it! Shade in a 3x3 pattern on some graph paper and then 'anti-alias' this by counting pixels in adjacent 4x4 blocks and you will see why a single shade of grey comes out as a jumble which repeats itself every 3 pixels - this is where the familiar 'banding' comes from. The correct technique of course, is simply to anti-alias using multiples of 3x3 blocks. Since you never get more than 7 pixels blacked in, you get 8 grey levels at 133 dpi. Because of the dithering within the 3x3 block, you can use 6x6 blocks to get about 28 grey levels at 67 dpi.

The standard Archimedes can display only 16 grey levels and I find that with normal monitor settings only about 12 of these are readily distinguishable. The darkest four all look the same. My 300 dpi inkjet printer is very similar - it works best on lighter images. I have used this to allow some contrast enhancement in my routine, but despite some claims there is very limited scope with only 16 levels of grey (see the article *Improving the quality of half-tone printing on a dot matrix printer* in RISC User 5:5).

## THE ROUTINE

The *Scanty* anti-aliasing routine given here accepts a parameter for block sizes from 2x2 up to 6x6. Only the 3x3 and 6x6 blocks are really useful, but the others are there for completeness and to show you the result of selecting the wrong size. If you select 4x4 you get an image much the same as the manufacturer's version. It also accepts a contrast parameter low (L), medium (M) or high (H).

To use the routine you will need to type it in and save it. To use it, first scan your image in a 'photo' mode and save the output sprite file without using the anti-alias function. Then run the program (by double-clicking on it in the Desktop), which will ask for the parameters above, and the full pathname of the image to be processed. The routine will use your parameters to produce a new sprite file with the same pathname as the original sprite but with an "X" appended to the filename. You can then use the image just like any other scanner sprite but without the 'banding' problems. An unzoomed 6x6 image comes out about full size and a 3x3 about twice as big. You should pick the size appropriate to your requirements. I have printed many A6 scans on an inkjet printer zoomed to A4 size and the results are very good indeed.

```
 10 REM          >Scanty
 20 REM Program    SCAN TerrificallY
 30 REM Version   A 1.0
 40 REM Author     Ledger White
 50 REM RISC User June 1992
 60 REM Program    Subject to Copyright
 70 REM           Not Public Domain
 80 :
 90 ON ERROR PRINT REPORT$;" at line "
;ERL:END
 100 PROCinitialise
 110 PROCassemble
 120 PROCreadFile
 130 PROCnewFileHeader
 140 PROCconvertOldFile
 150 PROCcompleteNewFile
 160 PROCsaveNewFile
 170 END
 180 :
 190 DEFPROCinitialise
 200 *FX15,1
 210 PRINT"Dithered pattern size (2,3,4
,5,6)?"
 220 REPEAT
 230 dither%=INSTR("123456",GET$)
 240 UNTIL dither%>1
 250 dsquared%=dither%*dither%
 260 *FX15,1
 270 PRINT"Contrast (L,M,H)?"
 280 REPEAT
 290 contrast%=INSTR(" LlMmHh",GET$)
 300 UNTIL contrast%>0
 310 contrast%=contrast% DIV 2
 320 *FX15,1
 330 PROCgetFileName
 340 DIM oldFile% oldFileSizeK%*1024
 350 DIM newFile% newFileSizeK%*1024
 360 ENDPROC
 370 :
 380 DEFPROCgetFileName
 390 PRINT"Enter file name "
 400 INPUT oldFileName$
 410 SYS "OS_File",&05,oldFileName$ TO
r0%,,,,r4%,r5%
 420 IF r0%<>1 THEN VDU7:PRINT"FILE ";o
ldFileName$;" NOT FOUND":END
 430 oldFileSizeK%=r4%/1024+1
 440 PRINT"Old file size is ";oldFileSi
zeK%; "K"
 450 newFileSizeK%=oldFileSizeK%*4/dsqu
ared%+1
 460 PRINT"New file size is ";newFileSi
zeK%; "K"
 470 diff%=(HIMEM-END)-((10+newFileSize
K%+oldFileSizeK%)*1024)
 480 IF diff%<0 THEN VDU7: PRINT"Memory
 too little by ";INT(ABS(diff%)/1024)+1;
"K":END
 490 ENDPROC
 500 :
 510 DEFPROCreadFile
 520 PRINT "LOAD "+oldFileName$+" "+STR
$~(oldFile%+4)
 530 OSCLI "LOAD "+oldFileName$+" "+STR
$~(oldFile%+4)
 540 ENDPROC
 550 :
 560 DEFPROCnewFileHeader
 570 !(newFile%+&04)=&1
 580 !(newFile%+&08)=&10
 590 FOR X%=&14 TO &1F
 600 newFile%?X%=oldFile%?X%
 610 NEXT X%
 620 !(newFile%+&20)=(!(oldFile%+&20)+1
)*4/dither% - 1
 630 !(newFile%+&24)=(!(oldFile%+&24)+1
) DIV dither% - 1
 640 !(newFile%+&28)=0
 650 !(newFile%+&2C)=&1F
 660 !(newFile%+&30)=&AC
 670 !(newFile%+&34)=&AC
 680 !(newFile%+&38)=&14
 690 FORx%=0TO&0F
 700 !(newFile%+&3C+x%*8)=(x%<<4)+(x%<<
12)+(x%<<20)+(x%<<28)
 710 !(newFile%+&3C+x%*8+4)=(x%<<4)+(x%
<<12)+(x%<<20)+(x%<<28)
 720 ?(newFile%+&3C+x%*8) =0
 730 ?(newFile%+&3C+x%*8+4)=0
 740 NEXTx%
 750 ENDPROC
 760 :
 770 DEFPROCconvertOldFile
 780 PRINT"Start conversion..."
 790 oldLineInBytes%=(!(oldFile%+&20)+1
```

```
)*4
 800 newLineInBytes%=(!(newFile%+&20)+1
)*4
 810 oldNumberOfLines%=!(oldFile%+&24)+
1
 820 FOR X%=1 TO oldNumberOfLines%-dith
er% STEP dither%
 830 A%=oldFile%+&4C+X%*oldLineInBytes%
 840 C%=A%+oldLineInBytes%
 850 D%=newFile%+&BC+((X%-1)/dither%)*n
ewLineInBytes%
 860 E%=(!(oldFile%+&20)+1)*4
 870 CALL code%
 880 NEXT X%
 890 PRINT"End conversion"
 900 ENDPROC
 910 :
 920 DEFPROCcompleteNewFile
 930 !(newFile%+&0C)=(!(newFile%+&20)+1
)*4*(!(newFile%+&24)+1)+&BC
 940 !(newFile%+&10)=!(newFile%+&0C)-&1
0
 950 newFileLength%=!(newFile%+&0C)-4
 960 ENDPROC
 970 :
 980 DEFPROCsaveNewFile
 990 PRINT "SAVE "+oldFileName$+"X"+" "
+STR$~(newFile%+4)+" +"+STR$~newFileLeng
th%
1000 OSCLI "SAVE "+oldFileName$+"X"+" "
+STR$~(newFile%+4)+" +"+STR$~newFileLeng
th%
1010 PRINT "SETTYPE "+oldFileName$+"X"+
" "+"FF9"
1020 OSCLI "SETTYPE "+oldFileName$+"X"+
" "+"FF9"
1030 ENDPROC
1040 :
1050 DEFPROCassemble
1060 DIM code% 800
1070 FOR I%=4 TO 6 STEP 2
1080 P%=0
1090 O%=code%
1100 [
1110 OPT I%
1120 STMFD R13!,{R14}
1130 SBC R0,R0,#1
1140 BL getNextSetOfBytes
1150 MOV R1,#2
1160 .loop
1170 MOV R6,#0
1180 BL getNextSixteenBits
1190 ORR R6,R6,R5
1200 BL getNextSixteenBits
1210 ORR R6,R6,R5,LSL #4
1220 BL getNextSixteenBits
1230 ORR R6,R6,R5,LSL #8
1240 BL getNextSixteenBits
1250 ORR R6,R6,R5,LSL #12
1260 BL getNextSixteenBits
1270 ORR R6,R6,R5,LSL #16
1280 BL getNextSixteenBits
1290 ORR R6,R6,R5,LSL #20
1300 BL getNextSixteenBits
1310 ORR R6,R6,R5,LSL #24
1320 BL getNextSixteenBits
1330 ORR R6,R6,R5,LSL #28
1340 STR R6,[R3],#4
1350 CMP R0,R2
1360 BLS loop
1370 LDMFD R13!,{PC}
1380 .getNextSetOfBytes
1390 STMFD R13!,{R14}
1400 ADD R0,R0,#1
1410 MOV R1,R0
1420 LDRB R7,[R1],R4
1430 LDRB R8,[R1],R4
1440 ]
1450 IF dither%>2 THEN [OPT I%:LDRB R9,
[R1],R4:]
1460 IF dither%>3 THEN [OPT I%:LDRB R10
,[R1],R4:]
1470 IF dither%>4 THEN [OPT I%:LDRB R11
,[R1],R4:]
1480 IF dither%>5 THEN [OPT I%:LDRB R12
,[R1],R4:]
1490 [OPT I%
1500 MOV R1,#1
1510 LDMFD R13!,{PC}
1520 .getNextSixteenBits
1530 STMFD R13!,{R14}
1540 MOV R5,#0
1550 ]
1560 FOR X%=1 TO dither%
1570 [OPT I%
1580 TST R7,R1
```

```
 1590 ADDEQ R5,R5,#1
 1600 TST R8,R1
 1610 ADDEQ R5,R5,#1
 1620 ]
 1630 IF dither%>2 THEN [OPT I%:TST R9,R
1 :ADDEQ R5,R5,#1:]
 1640 IF dither%>3 THEN [OPT I%:TST R10,
R1:ADDEQ R5,R5,#1:]
 1650 IF dither%>4 THEN [OPT I%:TST R11,
R1:ADDEQ R5,R5,#1:]
 1660 IF dither%>5 THEN [OPT I%:TST R12,
R1:ADDEQ R5,R5,#1:]
 1670 [OPT I%
 1680 MOV R1,R1, LSL #1
 1690 CMP R1,#&FF
 1700 BLHI getNextSetOfBytes
 1710 ]
 1720 NEXT X%
 1730 IF dither%>4 THEN
 1740 IF contrast%=1 THEN
 1750 [OPT I%:ADD R5,R5,R5,LSL #1:MOVR5,
R5,LSR #3:]
 1760 ENDIF
 1770 IF contrast%=2 THEN
 1780 [OPT I%:MOV R5,R5,LSR #1:]
 1790 ENDIF
 1800 IF contrast%=3 THEN
 1810 [OPT I%:ADD R5,R5,R5,LSL #1:MOV R5
,R5,LSR #2:]
 1820 ENDIF
 1830 ENDIF
 1840 IF dither%<5 THEN
 1850 IF contrast%=2 THEN
 1860 [OPT I%:ADD R5,R5,R5,LSL #1:MOV R5
,R5,LSR #1:]
 1870 ENDIF
 1880 IF contrast%=3 THEN
 1890 [OPT I%:MOV R5,R5,LSL #1:]
 1900 ENDIF
 1910 ENDIF
 1920 [OPT I%
 1930 CMP R5,#15
 1940 MOVHI R5,#15
 1950 EOR R5,R5,#&F
 1960 LDMFD R13!,{PC}
 1970 ]
 1980 NEXT I%
 1990 ENDPROC
```