File created: 24-Sep-2023 14:35:09 **{WMEDLEY}<lispusers>READAIS.;2**

edit by: rmk

changes to: (FNS AISHISTOGRAM)

previous date: 28-Apr-88 17:04:57 {WMEDLEY}<lispusers>READAIS.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
;;
;; Copyright (c) 1982-1988 by Xerox Corporation.

(RPAQQ READAISCOMS
       ((DECLARE%: EVAL@COMPILE DONTCOPY (CONSTANTS (NYBBLESPERWORD 4)))
        ;; fixed INSUREAISFILE, AISBLT, AISBLT8TO8.  nhb 27-Apr-88 01:58:56
        (FNS 24BITCOLORTO8BITMAP AISBLT AISBLT1TO1 AISBLT8TO4MODUL AISBLT8TOLESSFSA AISBLT8TO4TRUNC AISBLT8TO8
             AISBLT4TO4 AISBLT8TO4LESSFSA AISBLT8TO1FSA AISBLT8TO1TRUNC CLOSEST.COLOR GRAPHAISHISTOGRAM
             AISHISTOGRAM SMOOTHEDFILTER SLOW.COLOR.DISTANCE FAST.COLOR.DISTANCE INSUREAISFILE SHOWCOLORAIS
             SHOWCOLORAIS1 WRITEAIS WRITEAIS1 \GETBASENYBBLE \PUTBASENYBBLE)
        (MACROS .GET.4BIT.AND.SPREAD.ERR. .GET.1BIT.AND.SPREAD.ERR. .GET.NBIT.AND.SPREAD.ERR. .GET.LEFTMOST.4BIT
                .GET.LEFTMOST.BIT. .GET.BESTCOLOR.AND.SPREAD.ERR. .4BIT.MODULATE.INTENSITY.VALUE.
                .MODULATE.INTENSITY.VALUE. SQUARE)
        (P (MOVD? 'FAST.COLOR.DISTANCE 'COLOR.DISTANCE))
        (VARS AISDIRECTORIES)
        (GLOBALVARS AISDIRECTORIES)))

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(RPAQQ NYBBLESPERWORD 4)

(CONSTANTS (NYBBLESPERWORD 4))
)
)

;; fixed INSUREAISFILE, AISBLT, AISBLT8TO8.  nhb 27-Apr-88 01:58:56

(DEFINEQ

(24BITCOLORTO8BITMAP
  [LAMBDA (REDSTREAM GREENSTREAM BLUESTREAM WIDTH HEIGHT BASE BYTESPERLINE BITMAPRASTERWIDTH COLORMAP)
                                                                    (* kbr%: "13-Jul-85 19:28")

          (* internal function that puts pixels from an ais file into an 8 bit per pixel bitmap)

    (DECLARE (LOCALVARS . T))
    (PROG (LINEBASE DATABEG NEXTLINEREDERRORTABLE NEXTLINEGREENERRORTABLE NEXTLINEBLUEERRORTABLE
                THISPIXELREDERROR REDERRTABLEPTR THISPIXELGREENERROR GREENERRTABLEPTR THISPIXELBLUEERROR
                BLUEERRTABLEPTR REDBYTE GREENBYTE BLUEBYTE ERR COLOR RGB)
          (SETQ LINEBASE BASE)
          (SETQ DATABEG (GETFILEPTR REDSTREAM))
          (SETQ NEXTLINEREDERRORTABLE (\ALLOCBLOCK (ADD1 WIDTH)
                                                   T))
          (SETQ NEXTLINEGREENERRORTABLE (\ALLOCBLOCK (ADD1 WIDTH)
                                                     T))
          (SETQ NEXTLINEBLUEERRORTABLE (\ALLOCBLOCK (ADD1 WIDTH)
                                                    T))

          (* error tables are 1 larger so no end check is necessary in error propagation code.)
                                                                    (* initialize error tables.)
          (for I from 0 to (ITIMES WIDTH 2) by 2 do (\PUTBASEPTR NEXTLINEREDERRORTABLE I 0))
          (for I from 0 to (ITIMES WIDTH 2) by 2 do (\PUTBASEPTR NEXTLINEGREENERRORTABLE I 0))
          (for I from 0 to (ITIMES WIDTH 2) by 2 do (\PUTBASEPTR NEXTLINEBLUEERRORTABLE I 0))
                                                                    (* set width to width in words.)
          (SETQ WIDTH (LRSH WIDTH 1))
          (for Y from 0 to (ITIMES (SUB1 HEIGHT)
                                   BYTESPERLINE)
             by BYTESPERLINE do (SETQ BASE LINEBASE)
                                (SETQ REDERRTABLEPTR NEXTLINEREDERRORTABLE)
                                (SETQ THISPIXELREDERROR (\GETBASEPTR REDERRTABLEPTR 0))
                                (\PUTBASEPTR REDERRTABLEPTR 0 0)
                                (SETQ GREENERRTABLEPTR NEXTLINEGREENERRORTABLE)
                                (SETQ THISPIXELGREENERROR (\GETBASEPTR GREENERRTABLEPTR 0))
                                (\PUTBASEPTR GREENERRTABLEPTR 0 0)
                                (SETQ BLUEERRTABLEPTR NEXTLINEBLUEERRORTABLE)
                                (SETQ THISPIXELBLUEERROR (\GETBASEPTR BLUEERRTABLEPTR 0))
                                (\PUTBASEPTR BLUEERRTABLEPTR 0 0)
                                (\SETFILEPTR REDSTREAM Y)
                                (\SETFILEPTR GREENSTREAM Y)
```

```
                              (\SETFILEPTR BLUESTREAM Y)
                              (for X from 1 to WIDTH do [\PUTBASE BASE 0 (\PUTBASE BASE 0 (LOGOR (LLSH (
                                                                                    .GET.BESTCOLOR.AND.SPREAD.ERR.
                                                                                                          )
                                                                                                          8)
                                                                                                 (
                                                                                    .GET.BESTCOLOR.AND.SPREAD.ERR.
                                                                                                           ]
                                                    (SETQ BASE (\ADDBASE BASE 1)))
                              (SETQ LINEBASE (\ADDBASE LINEBASE BITMAPRASTERWIDTH)))
        (RETURN NIL])
```

(**AISBLT**

```
  [LAMBDA (FILE SOURCELEFT SOURCEBOTTOM DESTINATION DESTLEFT DESTBOTTOM WIDTH HEIGHT HOW FILTER NBITS
                LOBITADDRESS)                                          ; Edited 28-Apr-88 17:04 by Briggs
```

;; puts an ais image from FILE into a bitmap. The arguments are the same as BITBLTs where possible. HOW specifies how the number of bits per ;; pixel is condensed if reduction is necessary; TRUNCATE is truncate; FSA is Floyd-Steinberg algorithm; MODULATE is modulated with a ;; random function. If NBITS is given, the file is reduced to that number of bits and they are put into the byte so that the low order bit is at ;; LOBITADDRESS.

```
    (PROG (STREAM stodx stody left top bottom right width height DESTDD DESTSTRM SRASTERWIDTH SOURCEBASE
                  DESTRASTERWIDTH DESTBASE BITSPERPIXEL BITSPERSAMPLE SFILEWIDTH SFILEHEIGHT DIRECTION TMP
                  STARTSAMPLELINE STARTPIXEL BITOFFSET)
          (COND
             ((EQ NBITS 0)
              (RETURN)))
          (OR SOURCELEFT (SETQ SOURCELEFT 0))
          (OR SOURCEBOTTOM (SETQ SOURCEBOTTOM 0))
          (OR DESTLEFT (SETQ DESTLEFT 0))
          (OR DESTBOTTOM (SETQ DESTBOTTOM 0))
          (OR HOW (SETQ HOW 'FSA))
          (OR LOBITADDRESS (SETQ LOBITADDRESS 0))
          [COND
             ((OR (SETQ STREAM (FINDFILE FILE NIL AISDIRECTORIES))
                  (SETQ STREAM FILE))
              (SETQ STREAM (OPENSTREAM STREAM 'INPUT]            ; make sure the file is an AIS file and read its bits per sample,
                                                                 ; width and height.
          (SETQ TMP (INSUREAISFILE STREAM))
          (SETQ BITSPERSAMPLE (CAR TMP))
          (SETQ SFILEWIDTH (CADR TMP))
          (SETQ SFILEHEIGHT (CADDR TMP))                         ; convert the words per sample line into bytes
          (SETQ SRASTERWIDTH (CADDDR TMP))
          (SETQ DIRECTION (CADDDR (CDR TMP)))
          (COND
             ((NOT (EQ DIRECTION 3))
              (ERROR "Scan direction is not top-left to bottom-right" DIRECTION)))
          [COND
             [(type? BITMAP DESTINATION)
              (SETQ left 0)
              (SETQ bottom 0)
              (SETQ right (SUB1 (fetch (BITMAP BITMAPWIDTH) of DESTINATION)))
              (SETQ top (SUB1 (fetch (BITMAP BITMAPHEIGHT) of DESTINATION]
             ((\GETDISPLAYDATA DESTINATION)
              (COND
                 ((NEQ BITSPERSAMPLE 1)
                  (ERROR "Sorry, can't AISBLT to window if source is not 1 bpp")))
              (LET ((REGION (DSPCLIPPINGREGION NIL DESTINATION)))
                                                                 ; compute limits based on clipping regions.
                   (SETQ left (fetch (REGION LEFT) of REGION))
                   (SETQ bottom (fetch (REGION BOTTOM) of REGION))
                   (SETQ right (fetch (REGION PRIGHT) of REGION))
                   (SETQ top (fetch (REGION PTOP) of REGION]
```

;; left, bottom, right, top are in destination coordinates, and describe the bounding region

;; right and top are the pixel number counting from 0 of the last useable pixel

;; DESTLEFT and DESTBOTTOM have been transformed into the destination coordinates

```
          (SETQ BITSPERPIXEL (BITSPERPIXEL DESTINATION))
          (COND
             ((NULL NBITS)
              (SETQ NBITS BITSPERPIXEL))
             ((IGREATERP NBITS BITSPERPIXEL)
              (ERROR "Can't put this many bits into this bitmap" NBITS))
             ((IGREATERP (IPLUS LOBITADDRESS NBITS)
                     BITSPERPIXEL)
              (\ILLEGAL.ARG LOBITADDRESS)))
```

;; reduce the region if required by user's DESTLEFT and DESTBOTTOM or WIDTH and HEIGHT

```
          [PROGN (SETQ left (IMAX DESTLEFT left))
                 (SETQ bottom (IMAX DESTBOTTOM bottom))
                 [COND
                    (WIDTH                                       ; WIDTH is optional
                           (SETQ right (IMIN (IPLUS DESTLEFT WIDTH)
                                             right]
                 (COND
                    (HEIGHT                                      ; HEIGHT is optional
```

```
                        (SETQ top (IMIN (IPLUS DESTBOTTOM HEIGHT)
                                        top]
      (SETQ stodx (IDIFFERENCE DESTLEFT SOURCELEFT))
      (SETQ stody (IDIFFERENCE DESTBOTTOM SOURCEBOTTOM))
```

;; reduce by boundary of source (0,0) -- (sfilewidth, sfileheight)

```
      (PROGN (SETQ left (IMAX 0 left))
             (SETQ bottom (IMAX 0 bottom))
             (SETQ right (IMIN (IPLUS stodx SFILEWIDTH)
                               right))
             (SETQ top (IMIN (IPLUS stody SFILEHEIGHT)
                             top)))
```

;; calculate effective width and height

```
      (SETQ width (ADD1 (IDIFFERENCE right left)))
      (SETQ height (ADD1 (IDIFFERENCE top bottom)))
      (COND
         ((OR (ILEQ width 0)
              (ILEQ height 0))                          ; left is past right or bottom is past top;  there is nothing to
                                                        ; transfer.
          (CLOSEF STREAM)
          (RETURN)))
```

;; compute the parameters for the ais file. This assumes the picture is scanned from upper left to lower right.

;; DESTBASE is the start of the TOP row

;; DESTRASTERWIDTH and DESTBASE are not used in the 1 bpp case (which is also the Window case)

```
      [AND (BITMAPP DESTINATION)
           (SETQ DESTRASTERWIDTH (fetch (BITMAP BITMAPRASTERWIDTH) of DESTINATION))
           (SETQ DESTBASE (\ADDBASE (fetch (BITMAP BITMAPBASE) of DESTINATION)
                                    (ITIMES (\SFInvert DESTINATION top)
                                            DESTRASTERWIDTH]
```

;; STARTSAMPLELINE is number of source sample lines to skip to get to correct data

```
      (SETQ STARTSAMPLELINE (IDIFFERENCE bottom stody))
```

;; STARTPIXEL is number of pixels to skip to get to correct data

```
      (SETQ STARTPIXEL (IDIFFERENCE left stodx))
      (SELECTQ BITSPERSAMPLE
           (8 (COND
                 [(EQ BITSPERPIXEL 8)
                  (COND
                     ((AND (EQ HOW 'FSA)
                           (NOT (EQ NBITS 8)))
                      (AISBLT8TOLESSFSA STREAM STARTPIXEL STARTSAMPLELINE SRASTERWIDTH SFILEHEIGHT DESTBASE
                             left DESTRASTERWIDTH width height NBITS LOBITADDRESS))
                     (T (AISBLT8TO8 STREAM STARTPIXEL STARTSAMPLELINE SRASTERWIDTH SFILEHEIGHT DESTBASE left
                               DESTRASTERWIDTH width height NBITS LOBITADDRESS]
                 [(EQ BITSPERPIXEL 4)
                  (COND
                     ((EQ HOW 'FSA)
                      (AISBLT8TO4LESSFSA STREAM STARTPIXEL STARTSAMPLELINE SRASTERWIDTH SFILEHEIGHT DESTBASE
                             left DESTRASTERWIDTH width height NBITS LOBITADDRESS))
                     ((EQ HOW 'TRUNCATE)
                      (AISBLT8TO4TRUNC STREAM STARTPIXEL STARTSAMPLELINE SRASTERWIDTH SFILEHEIGHT DESTBASE
                             left DESTRASTERWIDTH width height NBITS LOBITADDRESS))
                     ((EQ HOW 'MODULATE)
                      (AISBLT8TO4MODUL STREAM STARTPIXEL STARTSAMPLELINE SRASTERWIDTH SFILEHEIGHT DESTBASE
                             left DESTRASTERWIDTH width height NBITS LOBITADDRESS))
                     (T (ERROR "Unknown HOW argument"]
                 [(EQ BITSPERPIXEL 1)
                  (COND
                     ((EQ HOW 'FSA)                     ; default to Floyd-Steinberg algorithm when going to single bit.
                      (AISBLT8TO1FSA STREAM STARTPIXEL STARTSAMPLELINE SRASTERWIDTH SFILEHEIGHT DESTBASE
                             left DESTRASTERWIDTH width height))
                     ((EQ HOW 'TRUNCATE)
                      (AISBLT8TO1TRUNC STREAM width height DESTBASE DESTRASTERWIDTH left STARTSAMPLELINE
                             SRASTERWIDTH BITOFFSET FILTER))
                     ((EQ HOW 'MODULATE)
                      (AISBLT8TO1FSA STREAM STARTPIXEL STARTSAMPLELINE SRASTERWIDTH SFILEHEIGHT DESTBASE
                             left DESTRASTERWIDTH width height))
                     (T (ERROR "Unknown HOW argument"]
                 (T (ERROR "Unknown bit per pixel size"))))
           (4 (COND
                 ((EQ BITSPERPIXEL 8)
                  (ERROR "8 BIT IMAGE FROM A 4 BIT FILE NOT IMPLEMENTED YET."))
                 ((EQ BITSPERPIXEL 4)
                  (AISBLT4TO4 STREAM HOW width height DESTBASE DESTRASTERWIDTH left STARTSAMPLELINE
                         SRASTERWIDTH NBITS LOBITADDRESS))
                 (T (ERROR "Blting from a 4 bit per sample file is only implemented for 4 or 8 bit per pixel
                           bitmaps."))))
           (1 (COND
                 ((EQ BITSPERPIXEL 1)
                  (AISBLT1TO1 STREAM STARTPIXEL STARTSAMPLELINE SRASTERWIDTH SFILEHEIGHT DESTINATION left
                         bottom width height))
                 (T (ERROR "Can only go from a 1 bit sources to a 1 bit destination."))))
           (ERROR "not a 4 or 8 bit per sample file"))
```

```
          (CLOSEF STREAM])
```

(AISBLT1TO1

```
  [LAMBDA (STREAM SOURCEPIXEL STARTSAMPLELINE SRASTERWIDTH SOURCEHEIGHT DEST DESTLEFT DESTBOTTOM WIDTH HEIGHT)
                                                                (* kbr%: "16-Jul-86 20:56")

          (* internal function that puts pixels from an ais file into an 1 bit per pixel bitmap *)
                                                                (* Assumes file has been left pointing at the beginning of the
                                                                data. *)
    (PROG (BYTESPERLINE DATABEG BEG END TEMPBITMAP TEMPBASE MAXX)
          (SETQ DATABEG (GETFILEPTR STREAM))
          (SETQ BYTESPERLINE (ITIMES 2 SRASTERWIDTH))
          (SETQ END (IPLUS DATABEG (UNFOLD (IPLUS (FOLDLO SOURCEPIXEL BITSPERWORD)
                                                  (ITIMES (IDIFFERENCE (SUB1 SOURCEHEIGHT)
                                                                       STARTSAMPLELINE)
                                                          SRASTERWIDTH))
                                           BYTESPERWORD)))
          (SETQ BEG (IDIFFERENCE END (ITIMES (SUB1 HEIGHT)
                                             BYTESPERLINE)))     (* Avoid complications of boundary alignment and clipping by
                                                                using TEMPBITMAP buffer and BITBLT *)
          (SETQ TEMPBITMAP (BITMAPCREATE (IPLUS BITSPERWORD WIDTH BITSPERWORD)
                                         1))
          (SETQ TEMPBASE (fetch (BITMAP BITMAPBASE) of TEMPBITMAP))
          [SETQ MAXX (SUB1 (IDIFFERENCE (FOLDHI (IPLUS SOURCEPIXEL WIDTH -1)
                                                BITSPERWORD)
                                        (FOLDLO SOURCEPIXEL BITSPERWORD]
          (for Y from BEG to END by BYTESPERLINE as BOTTOM from (IPLUS DESTBOTTOM HEIGHT -1) by -1
             do (\SETFILEPTR STREAM Y)
                (for X from 0 to MAXX do (\PUTBASE TEMPBASE X (\WIN STREAM)))
                (BITBLT TEMPBITMAP (IMOD SOURCEPIXEL BITSPERWORD)
                        0 DEST DESTLEFT BOTTOM WIDTH 1])
```

(AISBLT8TO4MODUL

```
  [LAMBDA (STREAM WIDTH HEIGHT BASE BITMAPRASTERWIDTH STARTBYTE STARTSAMPLELINE SRASTERWIDTH NBITS LOBITADDRESS)
                                                                (* kbr%: "16-Jul-86 19:47")

          (* internal function that puts pixels from an ais file into an 4 bit per pixel bitmap modulating the 8 bits.)

    (DECLARE (LOCALVARS . T))
    (PROG (BYTESPERLINE LINEBASE DATABEG WORD BITSTOSET RIGHTSHIFT COMPLMASK MODMAX MODMIN)
          (SETQ LINEBASE BASE)
          (SETQ DATABEG (GETFILEPTR STREAM))                    (* put bits in specified positions)
                                                                (* set the maximum and minimum values for the random
                                                                modulation function.)
          [SETQ MODMAX (IPLUS [SUB1 (LLSH 1 (IDIFFERENCE 6 (OR NBITS 4]
                              (SUB1 (LLSH 1 (IDIFFERENCE 5 (OR NBITS 4]
          (SETQ MODMIN (IMINUS MODMAX))
          (COND
             ((OR (ILESSP (SETQ RIGHTSHIFT (IDIFFERENCE 4 (IPLUS NBITS LOBITADDRESS)))
                          0)
                  (IGREATERP RIGHTSHIFT 4))
              (ERROR "NBITS plus LOBITADDRESS is too large.")))
          (SETQ BITSTOSET (SUB1 (EXPT 2 NBITS)))
          (SETQ BITSTOSET (LOGOR (LLSH BITSTOSET (IPLUS 12 LOBITADDRESS))
                                 (LLSH BITSTOSET (IPLUS 8 LOBITADDRESS))
                                 (LLSH BITSTOSET (IPLUS 4 LOBITADDRESS))
                                 (LLSH BITSTOSET LOBITADDRESS)))
          (SETQ COMPLMASK (LOGXOR BITSTOSET 65535))
          (SETQ BYTESPERLINE (ITIMES 2 SRASTERWIDTH))
          (for Y from 0 to (SUB1 HEIGHT)
             do (SETQ BASE LINEBASE)
                (\SETFILEPTR STREAM (IPLUS DATABEG (ITIMES Y BYTESPERLINE)))
                (for X from 1 to (LRSH WIDTH 2)
                   do (\PUTBASE BASE 0 (LOGOR (LOGAND (\GETBASE BASE 0)
                                                      COMPLMASK)
                                              (LOGAND (LRSH (LOGOR (LLSH (.4BIT.MODULATE.INTENSITY.VALUE. STREAM
                                                                          )
                                                                   8)
                                                                (LLSH (.4BIT.MODULATE.INTENSITY.VALUE. STREAM
                                                                          )
                                                                      4)
                                                                (.4BIT.MODULATE.INTENSITY.VALUE. STREAM)
                                                                (LRSH (.4BIT.MODULATE.INTENSITY.VALUE. STREAM
                                                                          )
                                                                      4))
                                                            RIGHTSHIFT)
                                                      BITSTOSET)))
                      (SETQ BASE (\ADDBASE BASE 1)))
                (SETQ LINEBASE (\ADDBASE LINEBASE BITMAPRASTERWIDTH)))
          (RETURN NIL])
```

(AISBLT8TOLESSFSA

```
  [LAMBDA (STREAM SOURCEBYTE STARTSAMPLELINE SRASTERWIDTH SOURCEHEIGHT DESTBASE DESTPIXEL DESTRASTERWIDTH WIDTH
           HEIGHT NBITS LOBITADDRESS)                           (* kbr%: "16-Jul-86 19:44")
```

```
        (* internal function that goes from an 8 bit file to NBITS in LOBITADDRESS position using a Floyd-Steinberg algorithm.)

        (* assumes starting addresses are all word aligned. Assumes file has been left pointing at the beginning of the data.
        NIL)

    (PROG (BYTESPERLINE DESTRIGHT DATABEG NEXTLINEERRORTABLE THISPIXELERROR ERRTABLEPTR BYTE ERR WORD BEG END
                COMPLMASK VAL DELBITS LOBITSMASK MAXVALUE INTENSITYBASE THREEEIGHTSERR)
           (SETQ DESTRIGHT (IPLUS DESTPIXEL WIDTH -1))
           (SETQ DATABEG (GETFILEPTR STREAM))
           (SETQ NEXTLINEERRORTABLE (\ALLOCBLOCK (ADD1 WIDTH)
                                                 T))
           (SETQ DELBITS (IDIFFERENCE 8 NBITS))
           [SETQ LOBITSMASK (SUB1 (EXPT 2 (IDIFFERENCE 8 NBITS]
           (SETQ MAXVALUE (SUB1 (EXPT 2 NBITS)))
           (SETQ INTENSITYBASE (\ALLOCBLOCK (EXPT 2 NBITS)))      (* BYTE and ERR are used by .GET.4BIT.AND.SPREAD.ERR.
                                                                  macro)
                                                                  (* initialize the intensity values for each color number.)
           (for I from 0 to MAXVALUE do (\PUTBASE INTENSITYBASE I (IQUOTIENT (ITIMES 255 I)
                                                                             MAXVALUE)))
           (for I from 0 to (ITIMES WIDTH 2) by 2 do (\PUTBASEPTR NEXTLINEERRORTABLE I 0))
                                                                  (* NEXTLINEERRORTABLE is 1 larger so no end check is
                                                                  necessary in error propagation code.)
                                                                  (* set width to width in words.)
           (SETQ BYTESPERLINE (ITIMES 2 SRASTERWIDTH))
           (SETQ WIDTH (LRSH WIDTH 1))
           (SETQ END (IPLUS DATABEG SOURCEBYTE (ITIMES (IDIFFERENCE (SUB1 SOURCEHEIGHT)
                                                                    STARTSAMPLELINE)
                                                       BYTESPERLINE)))
           (SETQ BEG (IDIFFERENCE END (ITIMES (SUB1 HEIGHT)
                                              BYTESPERLINE)))
           (SETQ COMPLMASK (LOGXOR (LLSH MAXVALUE LOBITADDRESS)
                                   255))
           (for Y from BEG to END by BYTESPERLINE do (SETQ ERRTABLEPTR NEXTLINEERRORTABLE)
                                                     (SETQ THISPIXELERROR (\GETBASEPTR ERRTABLEPTR 0))
                                                     (\PUTBASEPTR ERRTABLEPTR 0 0)
                                                     (\SETFILEPTR STREAM Y)
                                                     [for X from DESTPIXEL to DESTRIGHT
                                                        do (\PUTBASEBYTE DESTBASE X (LOGOR (LOGAND (\GETBASEBYTE
                                                                                                    DESTBASE X)
                                                                                                   COMPLMASK)
                                                                                           (LLSH (
                                                                                          .GET.NBIT.AND.SPREAD.ERR.
                                                                                                   STREAM)
                                                                                                  LOBITADDRESS]
                                                     (COND
                                                        ((NOT (EQ Y END))
                                                         (SETQ DESTBASE (\ADDBASE DESTBASE DESTRASTERWIDTH])
```

(AISBLT8TO4TRUNC

```
  [LAMBDA (STREAM SOURCEBYTE STARTSAMPLELINE SRASTERWIDTH SOURCEHEIGHT DESTBASE DESTPIXEL DESTRASTERWIDTH WIDTH
                HEIGHT NBITS LOBITADDRESS)                        (* kbr%: "16-Jul-86 19:46")

        (* internal function that puts pixels from an ais file into an 8 bit per pixel bitmap)
                                                                  (* Assumes file has been left pointing at the beginning of the
                                                                  data. NIL)
    (PROG (BYTESPERLINE DESTRIGHT DATABEG WORD BEG END MASK COMPLMASK LEFTSHIFT MAXVALUE)
           (SETQ DESTRIGHT (IPLUS DESTPIXEL WIDTH -1))
           (SETQ DATABEG (GETFILEPTR STREAM))
           (SETQ MAXVALUE (SUB1 (EXPT 2 NBITS)))
           (SETQ BYTESPERLINE (ITIMES 2 SRASTERWIDTH))
           (SETQ END (IPLUS DATABEG SOURCEBYTE (ITIMES (IDIFFERENCE (SUB1 SOURCEHEIGHT)
                                                                    STARTSAMPLELINE)
                                                       BYTESPERLINE)))
           (SETQ BEG (IDIFFERENCE END (ITIMES (SUB1 HEIGHT)
                                              BYTESPERLINE)))
           (SETQ LEFTSHIFT (IDIFFERENCE (IPLUS NBITS LOBITADDRESS)
                                        8))
           (SETQ MASK (LLSH MAXVALUE LOBITADDRESS))
           (SETQ COMPLMASK (LOGXOR MASK 15))
           (for Y from BEG to END by BYTESPERLINE
              do (\SETFILEPTR STREAM Y)
                 [for X from DESTPIXEL to DESTRIGHT do (\PUTBASENYBBLE DESTBASE X
                                                                       (LOGOR (LOGAND (\GETBASENYBBLE DESTBASE X)
                                                                                      COMPLMASK)
                                                                              (LOGAND (LLSH (IDIFFERENCE 255 (\BIN STREAM))
                                                                                            LEFTSHIFT)
                                                                                      MASK]
                 (COND
                    ((NOT (EQ Y END))
                     (SETQ DESTBASE (\ADDBASE DESTBASE DESTRASTERWIDTH])
```

(AISBLT8TO8

```
  [LAMBDA (STREAM SOURCEBYTE STARTSAMPLELINE SRASTERWIDTH SOURCEHEIGHT DESTBASE DESTPIXEL DESTRASTERWIDTH WIDTH
                HEIGHT NBITS LOBITADDRESS)                        ; Edited 27-Apr-88 01:57 by Briggs
```

```
;; internal function that puts pixels from an ais file into an 8 bit per pixel bitmap
;; Assumes file has been left pointing at the beginning of the data.
(PROG (BYTESPERLINE DESTRIGHT DATABEG WORD BEG END MASK COMPLMASK LEFTSHIFT MAXVALUE)
      (SETQ DESTRIGHT (IPLUS DESTPIXEL WIDTH -1))
      (SETQ DATABEG (GETFILEPTR STREAM))
      (SETQ MAXVALUE (SUB1 (EXPT 2 NBITS)))
      (SETQ BYTESPERLINE (ITIMES BYTESPERWORD SRASTERWIDTH))
      (SETQ END (IPLUS DATABEG SOURCEBYTE (ITIMES (IDIFFERENCE (SUB1 SOURCEHEIGHT)
                                                               STARTSAMPLELINE)
                                                  BYTESPERLINE)))
      (SETQ BEG (IDIFFERENCE END (ITIMES (SUB1 HEIGHT)
                                         BYTESPERLINE)))
      (SETQ LEFTSHIFT (IDIFFERENCE (IPLUS NBITS LOBITADDRESS)
                                   8))
      (SETQ MASK (LLSH MAXVALUE LOBITADDRESS))
      (SETQ COMPLMASK (LOGXOR MASK 255))
      (if (AND (EQ NBITS 8)
               (EQ DESTRASTERWIDTH SRASTERWIDTH)
               (EQ SOURCEBYTE 0)
               (EQ DESTPIXEL 0)
               (EQ SRASTERWIDTH (FOLDHI WIDTH BYTESPERWORD)))
          then ;; we will use all the bits of the source bytes
               ;; the source and destination have the same raster width so we can ignore the line boundaries
               ;; there is no offset in the source or destination lines
               ;; the width we desire is the full source raster width modulo a possible slack byte for padding
               ;; SO, we can just slurp up the bytes in one large block
               (\SETFILEPTR STREAM BEG)
               (\BINS STREAM DESTBASE 0 (ITIMES HEIGHT BYTESPERLINE))
        elseif (EQ NBITS 8)
          then ;; we will use all the bits of the source bytes
               ;; but there are pixel offsets or we do not require all the bytes in a line
               [for Y from BEG to END by BYTESPERLINE do (\SETFILEPTR STREAM Y)
                                                         (\BINS STREAM DESTBASE DESTPIXEL WIDTH)
                                                         (COND
                                                            ((NOT (EQ Y END))
                                                             (SETQ DESTBASE (\ADDBASE DESTBASE DESTRASTERWIDTH
                                                                                      ]
          else ;; we are doing some processing of the source bytes
               (for Y from BEG to END by BYTESPERLINE
                  do (\SETFILEPTR STREAM Y)
                     [for X from DESTPIXEL to DESTRIGHT do (\PUTBASEBYTE DESTBASE X
                                                                 (LOGOR (LOGAND (\GETBASEBYTE DESTBASE X)
                                                                                COMPLMASK)
                                                                        (LOGAND (LLSH (\BIN STREAM)
                                                                                      LEFTSHIFT)
                                                                                MASK]
                     (COND
                        ((NOT (EQ Y END))
                         (SETQ DESTBASE (\ADDBASE DESTBASE DESTRASTERWIDTH]))
```

(AISBLT4TO4

```
[LAMBDA (STREAM MODULATIONFLG WIDTH HEIGHT BASE BITMAPRASTERWIDTH STARTBYTE STARTSAMPLELINE SRASTERWIDTH NBITS
          LOBITADDRESS)                                          (* kbr%: "16-Jul-86 19:51")

        (* internal function that puts pixels from a 4 bit ais file into a 4 bit per pixel bitmap)

   (DECLARE (LOCALVARS . T))
   (PROG (BYTESPERLINE LINEBASE DATABEG WORD MASK RIGHTSHIFT COMPLMASK MODMAX MODMIN)
         (SETQ BYTESPERLINE (ITIMES 2 SRASTERWIDTH))
         (SETQ LINEBASE BASE)
         (SETQ DATABEG (GETFILEPTR STREAM))                      (* set width to width in words.)
         (SETQ WIDTH (LRSH WIDTH 2))
         [COND
            [NBITS                                               (* put bits in specified positions)
                   [COND
                      (MODULATIONFLG (COND
                                        ((EQ NBITS 4)            (* turn off modulation; there's enough bits for all information.)
                                         (SETQ MODULATIONFLG NIL))
                                        ((EQ NBITS 3)            (* special case of 3 bits)
                                         (SETQ MODMAX 1)
                                         (SETQ MODMIN 0))
                                        (T                       (* set the maximum and minimum values for the random
                                                                    modulation function.)
                                           [SETQ MODMAX (SUB1 (LLSH 1 (IDIFFERENCE 2 NBITS]
                                           (SETQ MODMIN (IMINUS MODMAX]
                   (COND
                      ((OR (ILESSP (SETQ RIGHTSHIFT (IDIFFERENCE 4 (IPLUS NBITS LOBITADDRESS)))
                                   0)
                           (IGREATERP RIGHTSHIFT 4))
```

```
                        (ERROR "NBITS plus LOBITADDRESS is too large.")))
                    (SETQ MASK (SUB1 (EXPT 2 NBITS)))
                    (SETQ MASK (LOGOR (LLSH MASK (IPLUS 12 LOBITADDRESS))
                                      (LLSH MASK (IPLUS 8 LOBITADDRESS))
                                      (LLSH MASK (IPLUS 4 LOBITADDRESS))
                                      (LLSH MASK LOBITADDRESS)))
                    (SETQ COMPLMASK (LOGXOR MASK 65535))
                    (for Y from (IPLUS DATABEG STARTBYTE (ITIMES STARTSAMPLELINE BYTESPERLINE))
                       to (IPLUS DATABEG STARTBYTE (ITIMES (SUB1 (IPLUS STARTSAMPLELINE HEIGHT))
                                                           BYTESPERLINE))
                       by BYTESPERLINE
                       do (SETQ BASE LINEBASE)
                          (\SETFILEPTR STREAM Y)
                          (for X from 1 to WIDTH
                             do (\PUTBASE BASE 0 (LOGOR (LOGAND (\GETBASE BASE 0)
                                                                COMPLMASK)
                                                        (LOGAND (LRSH [COND
                                                                      (NIL
                                                                       (* not implemented correctly)
                                                                       MODULATIONFLG
                                                                       (LOGOR (LLSH (
                                                                          .4BIT.MODULATE.INTENSITY.VALUE.
                                                                                    STREAM)
                                                                                   8)
                                                                              (LLSH (
                                                                          .4BIT.MODULATE.INTENSITY.VALUE.
                                                                                    STREAM)
                                                                                   4)
                                                                              (
                                                                          .4BIT.MODULATE.INTENSITY.VALUE.
                                                                               STREAM)
                                                                              (LRSH (
                                                                          .4BIT.MODULATE.INTENSITY.VALUE.
                                                                                    STREAM)
                                                                                   4)))
                                                                      (T (LOGOR (LLSH (\BIN STREAM)
                                                                                      8)
                                                                                (\BIN STREAM]
                                                                RIGHTSHIFT)
                                                        MASK)))
                                (SETQ BASE (\ADDBASE BASE 1)))
                          (SETQ LINEBASE (\ADDBASE LINEBASE BITMAPRASTERWIDTH]
                 (T                                                    (* use all of the bits)
                    (for Y from (IPLUS DATABEG STARTBYTE (ITIMES STARTSAMPLELINE BYTESPERLINE))
                       to (IPLUS DATABEG STARTBYTE (ITIMES (SUB1 (IPLUS STARTSAMPLELINE HEIGHT))
                                                           BYTESPERLINE))
                       by BYTESPERLINE do (SETQ BASE LINEBASE)
                                          (\SETFILEPTR STREAM Y)
                                          (for X from 1 to WIDTH do (\PUTBASE BASE 0 (\WIN STREAM))
                                                                    (SETQ BASE (\ADDBASE BASE 1)))
                                          (SETQ LINEBASE (\ADDBASE LINEBASE BITMAPRASTERWIDTH]
        (RETURN])
```

(**AISBLT8TO4LESSFSA**

```
  [LAMBDA (STREAM SOURCEBYTE STARTSAMPLELINE SRASTERWIDTH SOURCEHEIGHT DESTBASE DESTPIXEL DESTRASTERWIDTH WIDTH
                HEIGHT NBITS LOBITADDRESS)                             (* kbr%: "16-Jul-86 19:46")

        (* internal function that goes from an 8 bit file to NBITS in LOBITADDRESS position using a Floyd-Steinberg algorithm.)

        (* assumes starting addresses are all word aligned. Assumes file has been left pointing at the beginning of the data.
        NIL)

    (PROG (BYTESPERLINE DESTRIGHT DATABEG NEXTLINEERRORTABLE THISPIXELERROR ERRTABLEPTR BYTE ERR WORD BEG END
                 COMPLMASK VAL DELBITS LOBITSMASK MAXVALUE INTENSITYBASE THREEEIGHTSERR)
          (SETQ DESTRIGHT (IPLUS DESTPIXEL WIDTH -1))
          (SETQ DATABEG (GETFILEPTR STREAM))
          (SETQ NEXTLINEERRORTABLE (\ALLOCBLOCK (ADD1 WIDTH)
                                                T))
          (SETQ DELBITS (IDIFFERENCE 8 NBITS))
          [SETQ LOBITSMASK (SUB1 (EXPT 2 (IDIFFERENCE 8 NBITS]
          (SETQ MAXVALUE (SUB1 (EXPT 2 NBITS)))
          (SETQ INTENSITYBASE (\ALLOCBLOCK (EXPT 2 NBITS)))            (* initialize the intensity values for each color number.)
          (for I from 0 to MAXVALUE do (\PUTBASE INTENSITYBASE I (IQUOTIENT (ITIMES 255 I)
                                                                            MAXVALUE)))
          (for I from 0 to (ITIMES WIDTH 2) by 2 do (\PUTBASEPTR NEXTLINEERRORTABLE I 0))
                                                                       (* NEXTLINEERRORTABLE is 1 larger so no end check is
                                                                       necessary in error propagation code.)
                                                                       (* set width to width in words.)
          (SETQ WIDTH (LRSH WIDTH 1))
          (SETQ BYTESPERLINE (ITIMES 2 SRASTERWIDTH))
          (SETQ END (IPLUS DATABEG SOURCEBYTE (ITIMES (IDIFFERENCE (SUB1 SOURCEHEIGHT)
                                                                   STARTSAMPLELINE)
                                                      BYTESPERLINE)))
          (SETQ BEG (IDIFFERENCE END (ITIMES (SUB1 HEIGHT)
                                             BYTESPERLINE)))
          (SETQ COMPLMASK (LOGXOR (LLSH MAXVALUE LOBITADDRESS)
```

```
                                15))
          (for Y from BEG to END by BYTESPERLINE do (SETQ ERRTABLEPTR NEXTLINEERRORTABLE)
                                                    (SETQ THISPIXELERROR (\GETBASEPTR ERRTABLEPTR 0))
                                                    (\PUTBASEPTR ERRTABLEPTR 0 0)
                                                    (\SETFILEPTR STREAM Y)
                                                    [for X from DESTPIXEL to DESTRIGHT
                                                       do (\PUTBASENYBBLE DESTBASE X
                                                                 (LOGOR (LOGAND (\GETBASENYBBLE DESTBASE X)
                                                                                COMPLMASK)
                                                                        (LLSH (.GET.NBIT.AND.SPREAD.ERR. STREAM)
                                                                              LOBITADDRESS]
                                                    (COND
                                                       ((NOT (EQ Y END))
                                                        (SETQ DESTBASE (\ADDBASE DESTBASE DESTRASTERWIDTH])
```

(AISBLT8TO1FSA

```
  [LAMBDA (STREAM SOURCEBYTE STARTSAMPLELINE SRASTERWIDTH SOURCEHEIGHT DESTBASE DESTPIXEL DESTRASTERWIDTH WIDTH
                HEIGHT)                                                 (* kbr%: "16-Jul-86 19:49")
```

(* internal function that puts pixels from an ais file into an 1 bit per pixel bitmap propagating error with the Floyd-Steinberg algorithm.)

(* Assumes file has been left pointing at the beginning of the data.)

```
    (PROG (BYTESPERLINE DATABEG NEXTLINEERRORTABLE THISPIXELERROR ERRTABLEPTR BYTE ERR BITPTR BMWORD BEG END VAL
                DESTRIGHT DESTLEFTWORD DESTRIGHTWORD BITOFFSET FIRSTWORDBITS FINALWORDMASK INTENSITYBASE
                THREEEIGHTSERR)
          (SETQ DATABEG (GETFILEPTR STREAM))
          (SETQ NEXTLINEERRORTABLE (\ALLOCBLOCK (ADD1 WIDTH)
                                           T))
          (SETQ DESTRIGHT (IPLUS DESTPIXEL WIDTH -1))
          (SETQ DESTLEFTWORD (FOLDLO DESTPIXEL BITSPERWORD))
          (SETQ DESTRIGHTWORD (FOLDLO DESTRIGHT BITSPERWORD))
          (SETQ BITOFFSET (LOGAND DESTPIXEL 15))
          (SETQ FIRSTWORDBITS (IDIFFERENCE BITSPERWORD BITOFFSET))
          [SETQ FINALWORDMASK (SUB1 (EXPT 2 (IDIFFERENCE BITSPERWORD (LOGAND (IPLUS BITOFFSET WIDTH)
                                                                             15]
          (SETQ INTENSITYBASE (\ALLOCBLOCK 2))                          (* BYTE and ERR are used by .GET.1BIT.AND.SPREAD.ERR. macro)
                                                                        (* NEXTLINEERRORTABLE is 1 larger so no end check is necessary in error propagation code.)
                                                                        (* initialize the intensity values for each color number.)
          (\PUTBASE INTENSITYBASE 0 255)
          (\PUTBASE INTENSITYBASE 1 0)
          (for I from 0 to (ITIMES WIDTH 2) by 2 do (\PUTBASEPTR NEXTLINEERRORTABLE I 0))
          (SETQ BYTESPERLINE (ITIMES 2 SRASTERWIDTH))
          (SETQ END (IPLUS DATABEG SOURCEBYTE (ITIMES (IDIFFERENCE (SUB1 SOURCEHEIGHT)
                                                                   STARTSAMPLELINE)
                                                      BYTESPERLINE)))
          (SETQ BEG (IDIFFERENCE END (ITIMES (SUB1 HEIGHT)
                                             BYTESPERLINE)))
          (for Y from BEG to END by BYTESPERLINE
             do                                                         (* load BMWORD with the bits in the first word that won't be clobbered.)
                (SETQ BMWORD (LRSH (\GETBASE DESTBASE DESTLEFTWORD)
                                   FIRSTWORDBITS))
                (SETQ BITPTR BITOFFSET)
                (SETQ ERRTABLEPTR NEXTLINEERRORTABLE)
                (SETQ THISPIXELERROR (\GETBASEPTR ERRTABLEPTR 0))
                (\PUTBASEPTR ERRTABLEPTR 0 0)
                (\SETFILEPTR STREAM Y)
                [for X from DESTPIXEL to DESTRIGHT do (SETQ BMWORD (LOGOR (LLSH BMWORD 1)
                                                                          (.GET.1BIT.AND.SPREAD.ERR. STREAM)))
                                                      [COND
                                                         ((EQ (SETQ BITPTR (ADD1 BITPTR))
                                                              16) (* store this word and move to next word.)
                                                          (\PUTBASE DESTBASE (FOLDLO X BITSPERWORD)
                                                                 BMWORD)
                                                          (SETQ BITPTR (SETQ BMWORD 0]
                   finally                                              (* get the unset bits from the final word on the line.)
                         (OR (EQ BITPTR 0)
                             (\PUTBASE DESTBASE DESTRIGHTWORD (LOGOR (LLSH BMWORD (IDIFFERENCE 16 BITPTR))
                                                                     (LOGAND (\GETBASE DESTBASE DESTRIGHTWORD)
                                                                             FINALWORDMASK]
                (COND
                   ((NOT (EQ Y END))
                    (SETQ DESTBASE (\ADDBASE DESTBASE DESTRASTERWIDTH])
```

(AISBLT8TO1TRUNC

```
  [LAMBDA (STREAM WIDTH HEIGHT BASE BITMAPRASTERWIDTH STARTBYTE STARTSAMPLELINE SRASTERWIDTH BITOFFSET FILTER)
                                                                        (* kbr%: "16-Jul-86 19:49")
```

(* internal function that puts pixels from an ais file into an 1 bit per pixel bitmap (truncating the error.))

(* Assumes file has been left pointing at the beginning of the data.)

```
    (DECLARE (LOCALVARS . T))
    (PROG (BYTESPERLINE LINEBASE FILTERARRAY DATABEG BYTE BITPTR BMWORD BEG END VAL FIRSTWORDBITS FINALWORDMASK)
          (SETQ LINEBASE BASE)
          (SETQ FILTERARRAY FILTER)
          (SETQ DATABEG (GETFILEPTR STREAM))
          (SETQ FIRSTWORDBITS (IDIFFERENCE BITSPERWORD BITOFFSET))
          [SETQ FINALWORDMASK (SUB1 (EXPT 2 (IDIFFERENCE BITSPERWORD (LOGAND (IPLUS BITOFFSET WIDTH)
                                                                             15]
          (SETQ BYTESPERLINE (ITIMES 2 SRASTERWIDTH))
          (SETQ BEG (IPLUS DATABEG STARTBYTE (ITIMES STARTSAMPLELINE BYTESPERLINE)))
          (SETQ END (IPLUS DATABEG STARTBYTE (ITIMES (SUB1 (IPLUS STARTSAMPLELINE HEIGHT))
                                                    BYTESPERLINE)))
          (for Y from BEG to END by BYTESPERLINE
             do (SETQ BASE LINEBASE)                          (* load BMWORD with the bits in the first word that won't be
                                                                 clobbered.)
                (SETQ BMWORD (LRSH (\GETBASE BASE 0)
                                   FIRSTWORDBITS))
                (SETQ BITPTR BITOFFSET)
                (\SETFILEPTR STREAM Y)
                [for X from 1 to WIDTH do (SETQ BMWORD (LOGOR (LLSH BMWORD 1)
                                                              (.GET.LEFTMOST.BIT. STREAM)))
                                          [COND
                                             ((EQ (SETQ BITPTR (ADD1 BITPTR))
                                                  16)         (* store this word and move to next word.)
                                              (\PUTBASE BASE 0 BMWORD)
                                              (SETQ BITPTR (SETQ BMWORD 0))
                                              (SETQ BASE (\ADDBASE BASE 1]
                   finally                                    (* get the unset bits from the final word on the line.)
                         (OR (EQ BITPTR 0)
                             (\PUTBASE BASE 0 (LOGOR (LLSH BMWORD (IDIFFERENCE 16 BITPTR))
                                                     (LOGAND (\GETBASE BASE 0)
                                                             FINALWORDMASK]
                (SETQ LINEBASE (\ADDBASE LINEBASE BITMAPRASTERWIDTH)))
          (RETURN NIL])

(CLOSEST.COLOR
  [LAMBDA (COLORMAP RED GREEN BLUE)                           (* kbr%: "26-May-85 14:51")
                                                              (* Which color of COLORMAP is closest to RGB? *)
    (PROG (DISTANCE ANSWER)
          (SETQ DISTANCE MAX.FIXP)
          (for COLOR from 0 to (SUB1 (ARRAYSIZE COLORMAP)) when (ILESSP (COLOR.DISTANCE (ELT COLORMAP COLOR)
                                                                                        RED GREEN BLUE)
                                                                        DISTANCE)
             do (SETQ ANSWER (ELT COLORMAP COLOR)))
          (RETURN ANSWER])

(GRAPHAISHISTOGRAM
  [LAMBDA (HISTOGRAM W)                                       (* lmm "13-DEC-82 18:42")
                                                              (* draws a historgram of the intensity levels of a picture.)
    (PROG (W ARSIZE MAX MAXELT)
          [SETQ W (OR W (CREATEW (GETBOXREGION 270 215]
          (SETQ MAX 0)
          (SETQ MAXELT 0)
          (for I from 0 to (SETQ ARSIZE (ARRAYSIZE HISTOGRAM)) by 32 do (DRAWLINE I 10 I 0 1 'REPLACE W))
          [for I from 0 to (SUB1 ARSIZE) do (COND
                                              ((IGREATERP (ELT HISTOGRAM I)
                                                          MAX)
                                               (SETQ MAX (ELT HISTOGRAM I))
                                               (SETQ MAXELT I]
          (for I from 0 to (SUB1 ARSIZE) do (DRAWLINE I 10 I (IPLUS 10 (IQUOTIENT (ITIMES (ELT HISTOGRAM I)
                                                                                           200)
                                                                                   MAX))
                                                             1
                                                             'REPLACE W))
          (RETURN W])

(AISHISTOGRAM
  [LAMBDA (FILE REGION)                                       ; Edited 24-Sep-2023 14:34 by rmk
                                                              (* kbr%: "13-Jul-85 19:28")
                                                              ; returns an array that have the number of pixels in FILE that
                                                              ; have each intensity.
    (PROG (STREAM DATABEG AISHISTOGRAM TMP BITSPERSAMPLE SFILEWIDTH SFILEHEIGHT SFILEBYTESPERLINE LEFT BOTTOM
                  RIGHT TOP WIDTH HEIGHT BEG END)
          [COND
             ((OR (SETQ STREAM (FINDFILE FILE NIL AISDIRECTORIES))
                  (SETQ STREAM FILE))
              (SETQ STREAM (OPENSTREAM STREAM 'INPUT]
          (SETQ TMP (INSUREAISFILE STREAM))
          (SETQ BITSPERSAMPLE (CAR TMP))
          (SETQ SFILEWIDTH (CADR TMP))
          (SETQ SFILEHEIGHT (CADDR TMP))
          (SETQ SFILEBYTESPERLINE (LLSH (CADDDR TMP)
                                        1))
          (SETQ DATABEG (GETFILEPTR STREAM))
```

```
          (SETQ AISHISTOGRAM (ARRAY (EXPT 2 BITSPERSAMPLE)
                                    NIL 0 0))
          [COND
             [REGION (SETQ LEFT (IMAX (IMIN (fetch (REGION LEFT) of REGION)
                                            (SUB1 SFILEWIDTH))
                                      0))
                     (SETQ RIGHT (IMAX (IMIN SFILEWIDTH (fetch (REGION PRIGHT) of REGION))
                                       0))
                     [COND
                        ((IGEQ LEFT RIGHT)
                         (RETURN AISHISTOGRAM))
                        (T (SETQ WIDTH (IDIFFERENCE RIGHT LEFT]
                     (SETQ BOTTOM (IMIN (fetch (REGION BOTTOM) of REGION)
                                        (SUB1 SFILEHEIGHT)))
                     (SETQ TOP (IMIN SFILEHEIGHT (fetch (REGION PTOP) of REGION)))
                     (COND
                        ((IGREATERP BOTTOM TOP)
                         (RETURN AISHISTOGRAM)))
                     (SETQ BEG (IPLUS DATABEG (IPLUS (ITIMES SFILEBYTESPERLINE (IDIFFERENCE SFILEHEIGHT TOP))
                                                     LEFT)))
                     (SETQ END (IPLUS DATABEG (IPLUS (ITIMES SFILEBYTESPERLINE (IDIFFERENCE SFILEHEIGHT BOTTOM))
                                                     LEFT)))
                     (for LINE from BEG to END by SFILEBYTESPERLINE
                        do (\SETFILEPTR STREAM LINE)
                           (for BIT from 1 to WIDTH do (SETA AISHISTOGRAM (SETQ TMP (\BIN STREAM))
                                                             (ADD1 (ELT AISHISTOGRAM TMP]
             (T (for LINE from 1 to SFILEHEIGHT do (for BIT from 1 to SFILEBYTESPERLINE
                                                      do (SETA AISHISTOGRAM (SETQ TMP (\BIN STREAM))
                                                               (ADD1 (ELT AISHISTOGRAM TMP]
          (CLOSEF STREAM)
          (RETURN AISHISTOGRAM])
```

(SMOOTHEDFILTER

```
  [LAMBDA (HISTOGRAM)                                   (* kbr%: "13-Jul-85 15:05")
```

(* returns a 256 to 256 mapping array that maximally distributes the intensity values by looking at the histogram array HISTOGRAM)

```
    (PROG (ARSIZE SMOOTHARRAY TOTALPOINTS POINTSLESS FILEINTENSITY NEWINTENSITY POINTSPAST BUCKETSIZE NTOMOVE
                  NPTS)
          (SETQ ARSIZE (ARRAYSIZE HISTOGRAM))
          (SETQ POINTSLESS 0)
          (SETQ NEWINTENSITY 0)
          (SETQ POINTSPAST 0)
          (SETQ SMOOTHARRAY (ARRAY ARSIZE NIL 0 0))
          (SETQ TOTALPOINTS (for I from 0 to (SUB1 ARSIZE) sum (ELT HISTOGRAM I)))
          (SETQ BUCKETSIZE (IQUOTIENT TOTALPOINTS 256))
          [for I from 0 to (SUB1 ARSIZE) do (SETQ NPTS (ELT HISTOGRAM I))
                                            (SETQ POINTSLESS (IPLUS POINTSLESS NPTS))
                                            (COND
                                               [(IGREATERP POINTSLESS BUCKETSIZE)
                                                (SETQ NTOMOVE (IQUOTIENT POINTSLESS BUCKETSIZE))
                                                (SETA SMOOTHARRAY I (IPLUS NEWINTENSITY (IQUOTIENT NTOMOVE 2)))
                                                [SETQ NEWINTENSITY (COND
                                                                      ((IGREATERP NEWINTENSITY 255)
                                                                       255)
                                                                      (T (IPLUS NEWINTENSITY NTOMOVE]
                                                (SETQ POINTSLESS (IDIFFERENCE POINTSLESS (ITIMES NTOMOVE
                                                                                                 BUCKETSIZE]
                                               (T (SETA SMOOTHARRAY I NEWINTENSITY]
          (RETURN SMOOTHARRAY])
```

(SLOW.COLOR.DISTANCE

```
  [LAMBDA (RGB RED GREEN BLUE)                          (* kbr%: "26-May-85 14:55")
                                                        (* returns a closeness measure for colors.)
    (IPLUS (SQUARE (IDIFFERENCE (fetch (RGB RED) of RGB)
                                RED))
           (SQUARE (IDIFFERENCE (fetch (RGB GREEN) of RGB)
                                GREEN))
           (SQUARE (IDIFFERENCE (fetch (RGB BLUE) of RGB)
                                BLUE])
```

(FAST.COLOR.DISTANCE

```
  [LAMBDA (RGB RED GREEN BLUE)                          (* kbr%: "26-May-85 14:52")
                                                        (* returns a closeness measure for colors.)
    (IPLUS (IABS (IDIFFERENCE (fetch (RGB RED) of RGB)
                              RED))
           (IABS (IDIFFERENCE (fetch (RGB GREEN) of RGB)
                              GREEN))
           (IABS (IDIFFERENCE (fetch (RGB BLUE) of RGB)
                              BLUE])
```

(INSUREAISFILE

```
[LAMBDA (FILE)                                                    ; Edited 26-Apr-88 23:54 by Briggs
```

;; make sure a file is an ais file and put fileptr at beginning of data. Returns a list of bitspersample, width and height

```
(PROG (STREAM HEADERLENGTH WIDTH HEIGHT BITSPERPIXEL RASTERWIDTH DIRECTION)
      (SETQ STREAM (OPENSTREAM FILE 'INPUT))
      (\SETFILEPTR STREAM 0)
      (COND
         ((EQ (\WIN STREAM)
              33962)                                              ; check for AIS password
          NIL)
         (T (ERROR (FULLNAME STREAM)
                   " is not an AIS file.")))
      (SETQ HEADERLENGTH (\WIN STREAM))
      (COND
         ((NOT (EQ (LRSH (\WIN STREAM)
                         10)
                   1))                                            ; unknown raster part type --- ignore the raster part length NIL
          (ERROR "not implemented to handle raster parts of this type.")))
      (SETQ HEIGHT (\WIN STREAM))
      (SETQ WIDTH (\WIN STREAM))
      (SETQ DIRECTION (\WIN STREAM))
      (COND
         ((NOT (EQ (\WIN STREAM)
                   1))
          (ERROR "not 1 sample per pixel.")))
      (COND
         ((NOT (EQ (\WIN STREAM)
                   1))
          (ERROR "Coding type is not 1 - UCA" NIL)))
      (SETQ BITSPERPIXEL (\WIN STREAM))
      (COND
         ((EQ BITSPERPIXEL 0)
          (SETQ BITSPERPIXEL 1)))
      (SETQ RASTERWIDTH (\WIN STREAM))
      (\SETFILEPTR STREAM (ITIMES 2 HEADERLENGTH))
      (COND
         ((NOT (EQ FILE STREAM))
          (CLOSEF STREAM)))
      (RETURN (LIST BITSPERPIXEL WIDTH HEIGHT RASTERWIDTH DIRECTION])
```

(**SHOWCOLORAIS**

```
[LAMBDA (BASEFILE COLORMAPINFO HOW SOURCELEFT SOURCEBOTTOM DESTINATION DESTLEFT DESTBOTTOM WIDTH HEIGHT)
                                                                  (* kbr%: "21-Aug-85 20:46")
```

(* reads a color image from three files -
REDFILE GREENFILE and BLUEFILE If COLORMAPINFO is a colormap, each point is taken into the closed color in colormap. If COLORMAPINFO is a list of numbers totaling the number of bits in the color bitmap)

```
(PROG (UBASEFILE BASENAME REDFILE GREENFILE BLUEFILE)
      (COND
         ((AND (LISTP BASEFILE)
               (EQLENGTH BASEFILE 3))                             (* BASEFILE = (REDFILE BLUEFILE GREENFILE) *)
          (SETQ REDFILE (CAR BASEFILE))
          (SETQ GREENFILE (CADR BASEFILE))
          (SETQ BLUEFILE (CADDR BASEFILE)))
         [(LITATOM BASEFILE)                                      (* BASEFILE = prefix for REDFILE GREENFILE & BLUEFILE *)
          (SETQ UBASEFILE (UNPACKFILENAME BASEFILE))
          (SETQ BASENAME (LISTGET UBASEFILE 'NAME))
          [SETQ REDFILE (INFILEP (PACKFILENAME (APPEND (LIST 'NAME (CONCAT BASENAME "-RED")
                                                             'EXTENSION "AIS")
                                                       UBASEFILE]
          [SETQ GREENFILE (OR (INFILEP (PACKFILENAME (APPEND (LIST 'NAME (CONCAT BASENAME "-GREEN")
                                                                   'EXTENSION "AIS")
                                                             UBASEFILE)))
                              (INFILEP (PACKFILENAME (APPEND (LIST 'NAME (CONCAT BASENAME "-GRN")
                                                                   'EXTENSION "AIS")
                                                             UBASEFILE]
          (SETQ BLUEFILE (OR (INFILEP (PACKFILENAME (APPEND (LIST 'NAME (CONCAT BASENAME "-BLUE")
                                                                  'EXTENSION "AIS")
                                                            UBASEFILE)))
                             (INFILEP (PACKFILENAME (APPEND (LIST 'NAME (CONCAT BASENAME "-BLU")
                                                                  'EXTENSION "AIS")
                                                            UBASEFILE]
          (T (\ILLEGAL.ARG BASEFILE)))
      (COND
         ((LISTP COLORMAPINFO)
          (PROG (REDBITS GREENBITS BLUEBITS)
                (SETQ REDBITS (CAR COLORMAPINFO))
                (SETQ GREENBITS (CADR COLORMAPINFO))
                (SETQ BLUEBITS (CADDR COLORMAPINFO))
                (AISBLT REDFILE SOURCELEFT SOURCEBOTTOM DESTINATION DESTLEFT DESTBOTTOM WIDTH HEIGHT HOW NIL
                        REDBITS (IPLUS GREENBITS BLUEBITS))
                (AISBLT GREENFILE SOURCELEFT SOURCEBOTTOM DESTINATION DESTLEFT DESTBOTTOM WIDTH HEIGHT HOW
                        NIL GREENBITS BLUEBITS)
                (AISBLT BLUEFILE SOURCELEFT SOURCEBOTTOM DESTINATION DESTLEFT DESTBOTTOM WIDTH HEIGHT HOW
                        NIL BLUEBITS 0)))
```

```
            ((ARRAYP COLORMAPINFO)                                    (* KBR%: This is WRONG! All clipping info is being lost.
                                                                      *)
             (SHOWCOLORAIS1 REDFILE GREENFILE BLUEFILE HOW DESTINATION COLORMAPINFO))
            (T (\ILLEGAL.ARG COLORMAPINFO])
```

(SHOWCOLORAIS1

```
  [LAMBDA (REDFILE GREENFILE BLUEFILE HOW COLORBM COLORMAP)      (* kbr%: "13-Jul-85 16:05")

          (* puts a color image into a color bitmap choosing colors that are closest to the ones in COLORMAP.)

    (PROG (REDSTREAM GREENSTREAM BLUESTREAM BITSPERPIXEL BASE BITMAPHEIGHT BITMAPWIDTH BITMAPRASTERWIDTH WIDTH
                 HEIGHT BITSPERSAMPLE BYTESPERLINE)
          (SETQ REDSTREAM (OPENSTREAM REDFILE 'INPUT))
          (SETQ GREENSTREAM (OPENSTREAM GREENFILE 'INPUT))
          (SETQ BLUESTREAM (OPENSTREAM BLUEFILE 'INPUT))
          (SETQ BITSPERPIXEL (fetch (BITMAP BITMAPBITSPERPIXEL) of COLORBM))
          (SETQ BITMAPRASTERWIDTH (fetch (BITMAP BITMAPRASTERWIDTH) of COLORBM))
          (SETQ BITMAPHEIGHT (fetch (BITMAP BITMAPHEIGHT) of COLORBM))
          (SETQ BITMAPWIDTH (fetch (BITMAP BITMAPWIDTH) of COLORBM))
          (SETQ BASE (fetch (BITMAP BITMAPBASE) of COLORBM))
          (SETQ HEIGHT (INSUREAISFILE REDSTREAM))
          (COND
             ((IGREATERP (SETQ WIDTH (CADR HEIGHT))
                         BITMAPWIDTH)
              (ERROR "Can't read AIS files whose width is greater than the target bitmap - yet.")))
          (SETQ BITSPERSAMPLE (CAR HEIGHT))
          (SETQ BYTESPERLINE (LLSH (CADDDR HEIGHT)
                                   1))
          (SETQ HEIGHT (CADDR HEIGHT))
          (INSUREAISFILE GREENSTREAM)
          (INSUREAISFILE BLUESTREAM)
          (COND
             ((AND (EQ BITSPERPIXEL 8)
                   (EQ BITSPERSAMPLE 8))
              (24BITCOLORTO8BITMAP REDSTREAM GREENSTREAM BLUESTREAM (IMIN WIDTH BITMAPWIDTH)
                     (IMIN HEIGHT BITMAPHEIGHT)
                     BASE BYTESPERLINE BITMAPRASTERWIDTH COLORMAP))
             (T (ERROR " can only go from three 8 bit color files into an 8 bit colormap for now.")))
          (CLOSEF REDSTREAM)
          (CLOSEF GREENSTREAM)
          (CLOSEF BLUESTREAM)
          (RETURN T])
```

(WRITEAIS

```
  [LAMBDA (BITMAP FILE REGION)                                    (* kbr%: "16-Jul-86 17:36")
                                                                  (* writes a bitmap on to a file in AIS format.)
    (PROG (STREAM TEMPBITMAP HEADERLENGTH BITSPERPIXEL RASTERWIDTH WIDTH HEIGHT)
          (SETQ BITSPERPIXEL (fetch (BITMAP BITMAPBITSPERPIXEL) of BITMAP))
          (COND
             ((REGIONP REGION)

          (* Get copy of selected REGION of BITMAP into temporary bitmap to avoid having to deal with odd boundary problems
          when writing contents of BITMAP to STREAM *)

              (SETQ TEMPBITMAP (BITMAPCREATE (fetch (REGION WIDTH) of REGION)
                                     (fetch (REGION HEIGHT) of REGION)
                                     BITSPERPIXEL))
              (BITBLT BITMAP (fetch (REGION LEFT) of REGION)
                     (fetch (REGION BOTTOM) of REGION)
                     TEMPBITMAP)
              (SETQ BITMAP TEMPBITMAP))
             (REGION (\ILLEGAL.ARG REGION)))
          (SETQ RASTERWIDTH (fetch (BITMAP BITMAPRASTERWIDTH) of BITMAP))
          (SETQ HEIGHT (fetch (BITMAP BITMAPHEIGHT) of BITMAP))
          (SETQ WIDTH (fetch (BITMAP BITMAPWIDTH) of BITMAP))
          (SETQ STREAM (OPENSTREAM FILE 'OUTPUT))                 (* write AIS password)
          (\WOUT STREAM 33962)                                    (* write header length in words -
                                                                  must be a multiple of 1024)
          (SETQ HEADERLENGTH 1024)
          (\WOUT STREAM HEADERLENGTH)                             (* set type and length of raster part header)
          (\WOUT STREAM (LOGOR (LLSH 1 10)
                               10))                               (* Scan count)
          (\WOUT STREAM HEIGHT)                                   (* ScanLength)
          (\WOUT STREAM WIDTH)                                    (* Scan Dir)
          (\WOUT STREAM 3)                                        (* samples per pixel.)
          (\WOUT STREAM 1)                                        (* coding type -
                                                                  UnCompressedArray)
          (\WOUT STREAM 1)                                        (* bits per sample)
          (\WOUT STREAM BITSPERPIXEL)                             (* words per sample line.)
          (\WOUT STREAM RASTERWIDTH)                              (* Sample lines per block -
                                                                  no blocks is 16 bit -1)
          (\WOUT STREAM 65535)                                    (* padding words per block -
                                                                  -1 if no blocks.)
          (\WOUT STREAM 65535)                                    (* header length is in words.)
          (\SETFILEPTR STREAM (ITIMES 2 HEADERLENGTH))            (* this would be a good place to dump the color map information)
```

```
          (\BOUTS STREAM (fetch (BITMAP BITMAPBASE) of BITMAP)
                 0
                 (ITIMES HEIGHT RASTERWIDTH 2))
          (RETURN (CLOSEF STREAM])
```

(WRITEAIS1

```
  [LAMBDA (STREAM LINEBASE NBYTESPERLINE FIRSTBYTEOFFSET HEIGHT RASTERWIDTH)
                                                                  (* kbr%: "16-Jul-86 17:13")
                                                                  (* dumps the bits from the bitmap with base BASE onto the file
                                                                  OFD.)
    (for Y from 1 to HEIGHT do (\BOUTS STREAM LINEBASE FIRSTBYTEOFFSET NBYTESPERLINE)
                               (SETQ LINEBASE (\ADDBASE LINEBASE RASTERWIDTH])
```

(\GETBASENYBBLE

```
  [LAMBDA (X D)                                                   (* kbr%: "21-Jul-85 23:37")
    (PROG (ANSWER)
          (SETQ ANSWER (\GETBASE X (FOLDLO D NYBBLESPERWORD)))
          (SETQ ANSWER (SELECTQ (LOGAND D 3)
                           (0 (LRSH ANSWER 12))
                           (1 (LRSH ANSWER 8))
                           (2 (LRSH ANSWER 4))
                           ANSWER))
          (SETQ ANSWER (LOGAND ANSWER 15))
          (RETURN ANSWER])
```

(\PUTBASENYBBLE

```
  [LAMBDA (X D V)                                                 (* kbr%: "21-Jul-85 23:40")
    (PROG (N ANSWER)
          (SETQ N (FOLDLO D NYBBLESPERWORD))
          (SETQ ANSWER (\GETBASE X N))
          [SETQ ANSWER (SELECTQ (LOGAND D 3)
                           (0 [LOGOR (LLSH V 12)
                                     (LOGAND ANSWER (LOGNOT (LLSH 15 12])
                           (1 [LOGOR (LLSH V 8)
                                     (LOGAND ANSWER (LOGNOT (LLSH 15 8])
                           (2 [LOGOR (LLSH V 4)
                                     (LOGAND ANSWER (LOGNOT (LLSH 15 4])
                           (LOGOR V (LOGAND ANSWER (LOGNOT 15]
          (\PUTBASE X N ANSWER)
          (RETURN ANSWER])

)

(DECLARE%: EVAL@COMPILE

(PUTPROPS .GET.4BIT.AND.SPREAD.ERR. MACRO [(STREAM)
                                           (PROGN
```

(* returns the 4 most significant bits taking into account the error and spreads the error into the appropriate places.)

```
                                                  (SETQ BYTE (IPLUS (\BIN STREAM)
                                                                    THISPIXELERROR))
                                                  (PROG1 (COND
                                                             ((IGREATERP BYTE 255)
                                                                  (* overflow case)
                                                              15)
                                                             (T (LRSH BYTE 4)))
                                                         (SETQ ERR (LOGAND BYTE 15))
```

(* put |3/8| of error into next pixel, |3/8| to one below and |1/8| to one below and to the right.)

```
                                                                  (* calculate |1/4| of error.)
                                                         (SETQ ERR (LRSH ERR 2))
                                                                  (* |3/8| of error to next pixel plus error from previous line)
                                                         [SETQ THISPIXELERROR (IPLUS (\GETBASE ERRTABLEPTR 1)
                                                                                     (IPLUS ERR (LRSH ERR 1]
                                                                  (* |1/8| of error to next one down to right.)
                                                         (\PUTBASE ERRTABLEPTR 1 (LRSH ERR 1))
                                                                  (* |3/8| to one below)
                                                         [\PUTBASE ERRTABLEPTR 0 (IPLUS (\GETBASE ERRTABLEPTR 0)
                                                                                        (IPLUS ERR (LRSH ERR 1]
                                                         (SETQ ERRTABLEPTR (\ADDBASE ERRTABLEPTR 1)))])

(PUTPROPS .GET.1BIT.AND.SPREAD.ERR. MACRO [(STREAM)
                                           (PROGN
```

(* returns the most significant bit taking into account the error and spreads the error into the appropriate places.)

```
                                                  (SETQ BYTE (IPLUS (\BIN STREAM)
                                                                    THISPIXELERROR))
                                                  (PROG1 [SETQ VAL (COND
                                                                       ((IGREATERP BYTE 255)
                                                                  (* overflow case)
                                                                        0)
                                                                       ((IGREATERP 0 BYTE)
```

```
                                                    (* overflow case)
                                                          1)
                                                         (T (LOGXOR (LRSH BYTE 7)
                                                                    1]
                                                (SETQ ERR (IDIFFERENCE BYTE (\GETBASE INTENSITYBASE VAL)))
```

(* put |3/8| of error into next pixel, |3/8| to one below and |1/4| to one below and to the right.)

```
                                                    (* calculate |1/4| of error.)
                                                (SETQ ERR (IDIFFERENCE (LRSH (IPLUS 256 ERR)
                                                                             2)
                                                                  64))
                                                    (* |3/8| of error to next pixel plus error from previous line)
                                                (SETQ THREEEIGHTSERR (IPLUS ERR
                                                                            (IDIFFERENCE
                                                                             (LRSH (IPLUS 256 ERR)
                                                                                   1)
                                                                             128)))
                                                (SETQ THISPIXELERROR (IPLUS (\GETBASEPTR ERRTABLEPTR 2)
                                                                            THREEEIGHTSERR))
                                                    (* |1/4| of error to next one down to right.)
                                                (\PUTBASEPTR ERRTABLEPTR 2 ERR)
                                                    (* |3/8| to one below)
                                                (\PUTBASEPTR ERRTABLEPTR 0 (IPLUS (\GETBASEPTR ERRTABLEPTR
                                                                                               0)
                                                                                  THREEEIGHTSERR))
                                                (SETQ ERRTABLEPTR (\ADDBASE ERRTABLEPTR 2)))])
```

```
(PUTPROPS .GET.NBIT.AND.SPREAD.ERR. MACRO [(STREAM)
                                           (PROGN
```

(* returns the NBITS most significant bits taking into account the error and spreads the error into the appropriate places.)

```
                                                (SETQ BYTE (IPLUS (IDIFFERENCE 255 (\BIN STREAM))
                                                                  THISPIXELERROR))
                                                (PROG1 [SETQ VAL (COND
                                                                    ((IGREATERP BYTE 255)
                                                    (* overflow case)
                                                                     MAXVALUE)
                                                                    ((IGREATERP 0 BYTE)
                                                                     0)
                                                                    (T (LRSH BYTE DELBITS]
```

(* put |3/8| of error into next pixel, |3/8| to one below and |1/8| to one below and to the right.)

```
                                                (SETQ ERR (IDIFFERENCE BYTE (\GETBASE INTENSITYBASE VAL)))
                                                    (* calculate |1/4| of error.)
                                                (SETQ ERR (IDIFFERENCE (LRSH (IPLUS 256 ERR)
                                                                             2)
                                                                  64))
                                                    (* |3/8| of error to next pixel plus error from previous line)
                                                (SETQ THREEEIGHTSERR (IPLUS ERR
                                                                            (IDIFFERENCE
                                                                             (LRSH (IPLUS 256 ERR)
                                                                                   1)
                                                                             128)))
                                                (SETQ THISPIXELERROR (IPLUS (\GETBASEPTR ERRTABLEPTR 2)
                                                                            THREEEIGHTSERR))
                                                    (* |1/8| of error to next one down to right.)
                                                (\PUTBASEPTR ERRTABLEPTR 2 ERR)
                                                    (* |3/8| to one below)
                                                (\PUTBASEPTR ERRTABLEPTR 0 (IPLUS (\GETBASEPTR ERRTABLEPTR
                                                                                               0)
                                                                                  THREEEIGHTSERR))
                                                (SETQ ERRTABLEPTR (\ADDBASE ERRTABLEPTR 2)))])
```

```
(PUTPROPS .GET.LEFTMOST.4BIT MACRO ((STREAM)                    (* returns the 4 most significant bits)
                                    (LRSH (\BIN STREAM)
                                          4)))
```

```
(PUTPROPS .GET.LEFTMOST.BIT. MACRO ((STREAM)
```

(* returns the most significant bit from an 8 bit sample. It also inverts the sign of the bit since 1 is black and 0 white.
NIL)

```
                                   (COND
                                      ((IGREATERP (COND
                                                     (FILTERARRAY (ELT FILTERARRAY (\BIN STREAM)))
                                                     (T (\BIN STREAM)))
                                              127)
                                       0)
                                      (T 1))))
```

```
(PUTPROPS .GET.BESTCOLOR.AND.SPREAD.ERR. MACRO (NIL (PROGN
```

(* returns the best matching color bits taking into account the error and spreads the error into the appropriate places.)

```
                                                                [SETQ COLOR (CLOSEST.COLOR COLORMAP
```

```
                                                    (SETQ REDBYTE
                                                     (IPLUS (\BIN REDSTREAM)
                                                            THISPIXELREDERROR))
                                                    (SETQ GREENBYTE
                                                     (IPLUS (\BIN GREENSTREAM)
                                                            THISPIXELGREENERROR))
                                                    (SETQ BLUEBYTE
                                                     (IPLUS (\BIN BLUESTREAM)
                                                            THISPIXELBLUEERROR]
                                            (SETQ RGB (ELT COLORMAP COLOR))
                                            (PROGN (SETQ ERR (IDIFFERENCE (fetch (RGB RED)
                                                                             of RGB)
                                                                  REDBYTE))
                                                   [COND
                                                      [(IGREATERP ERR -1)

(* put |3/8| of error into next pixel, |3/8| to one below and |1/8| to one below and to the right.)
                                                         (* calculate |1/4| of error.)
                                                               (SETQ ERR (LRSH ERR 2))
                                                         (* |3/8| of error to next pixel plus error from previous line)
                                                               [SETQ THISPIXELREDERROR
                                                                (IPLUS (\GETBASEPTR REDERRTABLEPTR 2)
                                                                       (IPLUS ERR (LRSH ERR 1]
                                                         (* |1/8| of error to next one down to right.)
                                                               (\PUTBASEPTR REDERRTABLEPTR 2
                                                                      (LRSH ERR 1))
                                                         (* |3/8| to one below)
                                                               (\PUTBASEPTR REDERRTABLEPTR 0
                                                                      (IPLUS (\GETBASEPTR
                                                                                    REDERRTABLEPTR 0)
                                                                             (IPLUS ERR
                                                                                    (LRSH ERR 1]
                                                            (T
                                                         (* error is negative, do things differently.)
                                                         (* calculate |1/4| of error.)
                                                               (SETQ ERR (LRSH (IMINUS ERR)
                                                                               2))
                                                         (* |3/8| of error to next pixel plus error from previous line)
                                                                [SETQ THISPIXELREDERROR
                                                                 (IDIFFERENCE (\GETBASEPTR
                                                                                  REDERRTABLEPTR 2)
                                                                              (IPLUS ERR (LRSH ERR 1]
                                                         (* |1/8| of error to next one down to right.)
                                                                (\PUTBASEPTR REDERRTABLEPTR 2
                                                                       (IMINUS (LRSH ERR 1)))
                                                         (* |3/8| to one below)
                                                                (\PUTBASEPTR
                                                                 REDERRTABLEPTR 0
                                                                 (IDIFFERENCE (\GETBASEPTR
                                                                                  REDERRTABLEPTR 0)
                                                                              (IPLUS ERR (LRSH ERR 1]
                                                               (SETQ REDERRTABLEPTR (\ADDBASE
                                                                                       REDERRTABLEPTR 2
                                                                                       )))
                                            (PROGN (SETQ ERR (IDIFFERENCE (fetch (RGB GREEN)
                                                                             of RGB)
                                                                  GREENBYTE))
                                                   [COND
                                                      [(IGREATERP ERR -1)

(* put |3/8| of error into next pixel, |3/8| to one below and |1/8| to one below and to the right.)
                                                         (* calculate |1/4| of error.)
                                                               (SETQ ERR (LRSH ERR 2))
                                                         (* |3/8| of error to next pixel plus error from previous line)
                                                               [SETQ THISPIXELGREENERROR
                                                                (IPLUS (\GETBASEPTR GREENERRTABLEPTR 2)
                                                                       (IPLUS ERR (LRSH ERR 1]
                                                         (* |1/8| of error to next one down to right.)
                                                               (\PUTBASEPTR GREENERRTABLEPTR 2
                                                                      (LRSH ERR 1))
                                                         (* |3/8| to one below)
                                                               (\PUTBASEPTR GREENERRTABLEPTR 0
                                                                      (IPLUS (\GETBASEPTR
                                                                                    GREENERRTABLEPTR 0)
                                                                             (IPLUS ERR
                                                                                    (LRSH ERR 1]
                                                            (T
                                                         (* error is negative, do things differently.)
                                                         (* calculate |1/4| of error.)
                                                               (SETQ ERR (LRSH (IMINUS ERR)
                                                                               2))
                                                         (* |3/8| of error to next pixel plus error from previous line)
                                                                [SETQ THISPIXELGREENERROR
                                                                 (IDIFFERENCE (\GETBASEPTR
                                                                                  GREENERRTABLEPTR
                                                                                  2)
                                                                              (IPLUS ERR (LRSH ERR 1]
```

```
                                                    (* |1/8| of error to next one down to right.)
                                                          (\PUTBASEPTR GREENERRTABLEPTR 2
                                                                (IMINUS (LRSH ERR 1)))
                                                    (* |3/8| to one below)
                                                          (\PUTBASEPTR
                                                           GREENERRTABLEPTR 0
                                                           (IDIFFERENCE (\GETBASEPTR
                                                                                GREENERRTABLEPTR
                                                                                0)
                                                                        (IPLUS ERR (LRSH ERR 1]
                                                       (SETQ GREENERRTABLEPTR (\ADDBASE
                                                                                  GREENERRTABLEPTR
                                                                                    2)))
                                                (PROGN (SETQ ERR (IDIFFERENCE (fetch (RGB BLUE)
                                                                                  of RGB)
                                                                      BLUEBYTE))
                                                       [COND
                                                          [(IGREATERP ERR -1)

            (* put |3/8| of error into next pixel, |3/8| to one below and |1/8| to one below and to the right.)
                                                    (* calculate |1/4| of error.)
                                                          (SETQ ERR (LRSH ERR 2))
                                                    (* |3/8| of error to next pixel plus error from previous line)
                                                          [SETQ THISPIXELBLUEERROR
                                                           (IPLUS (\GETBASEPTR BLUEERRTABLEPTR 2)
                                                                  (IPLUS ERR (LRSH ERR 1]
                                                    (* |1/8| of error to next one down to right.)
                                                          (\PUTBASEPTR BLUEERRTABLEPTR 2
                                                                (LRSH ERR 1))
                                                    (* |3/8| to one below)
                                                          (\PUTBASEPTR BLUEERRTABLEPTR 0
                                                                 (IPLUS (\GETBASEPTR
                                                                                BLUEERRTABLEPTR 0)
                                                                        (IPLUS ERR
                                                                               (LRSH ERR 1]
                                                         (T
                                                    (* error is negative, do things differently.)
                                                    (* calculate |1/4| of error.)
                                                           (SETQ ERR (LRSH (IMINUS ERR)
                                                                           2))
                                                    (* |3/8| of error to next pixel plus error from previous line)
                                                           [SETQ THISPIXELBLUEERROR
                                                            (IDIFFERENCE (\GETBASEPTR
                                                                                BLUEERRTABLEPTR 2
                                                                                )
                                                                        (IPLUS ERR (LRSH ERR 1]
                                                    (* |1/8| of error to next one down to right.)
                                                          (\PUTBASEPTR BLUEERRTABLEPTR 2
                                                                (IMINUS (LRSH ERR 1)))
                                                    (* |3/8| to one below)
                                                          (\PUTBASEPTR
                                                           BLUEERRTABLEPTR 0
                                                           (IDIFFERENCE (\GETBASEPTR
                                                                                BLUEERRTABLEPTR 0
                                                                                )
                                                                        (IPLUS ERR (LRSH ERR 1]
                                                       (SETQ BLUEERRTABLEPTR (\ADDBASE
                                                                                  BLUEERRTABLEPTR
                                                                                  2)))
                                                COLOR)))

(PUTPROPS .4BIT.MODULATE.INTENSITY.VALUE. MACRO ((STREAM)
                                                 (LOGAND (IMIN 255 (IMAX (IPLUS (\BIN STREAM)
                                                                                (RAND MODMIN MODMAX))
                                                                         0))
                                                         240)))

(PUTPROPS .MODULATE.INTENSITY.VALUE. MACRO ((STREAM)
                                            (IMIN 255 (IMAX (IPLUS (\BIN STREAM)
                                                                   (RAND MODMIN MODMAX))
                                                            0))))

(PUTPROPS SQUARE MACRO [LAMBDA (X)                          (* coded this way because negative arith is not is microcode for
                                                            ITIMES)
                        (COND
                           ((IGREATERP X -1)
                            (ITIMES X X))
                           (T (ITIMES (SETQ X (IMINUS X))
                                      X])
)

(MOVD? 'FAST.COLOR.DISTANCE 'COLOR.DISTANCE)

(RPAQQ AISDIRECTORIES (T {CORE} {DSK} {CYAN}<AIS>))

(DECLARE%: DOEVAL@COMPILE DONTCOPY
```

```
(GLOBALVARS AISDIRECTORIES)
)

(PUTPROPS READAIS COPYRIGHT ("Xerox Corporation" 1982 1983 1984 1985 1986 1987 1988))
```

FUNCTION INDEX

MACRO INDEX

VARIABLE INDEX

CONSTANT INDEX