File created: 28-May-86 17:27:25 **{ICE}<DENBER>LISP>ACE-PRIM.;3**

changes to: (FNS ACE.SCAN.PRIMBLOCKS ACE.FETCH.BLOCK ACE.MAX.REGIONS ACE.PICK.BEST.REGION ACE.COMPUTE.AREA ACE.COMPILE.FRAME ACE.EXTRACT)

previous date: 20-Jun-85 10:39:15 {ICE}<DENBER>LISP>ACE-PRIM.;2

Read Table: OLD-INTERLISP-FILE

Package: INTERLISP

Format: XCCS


```
(RPAQQ ACE-PRIMCOMS
       ((* COMPILER STUFF)
        (VARS ACE.PIXPERWORD ACE.BITMAP.MASK)
        (* LOW LEVEL COMPILER FNS)
        (FNS ACE.COMPILE.FRAME ACE.EXTRACT ACESETTHRESHOLD)
        (* REGION MAXING ROUTINES)
        (FNS ACE.MAX.REGIONS ACE.PICK.BEST.REGION ACE.COMPUTE.AREA)
        (* LOW LEVEL BITMAP COMPARISON)
        (FNS ACE.SCAN.BITMAPS ACE.SCAN.PRIMBLOCKS ACE.FETCH.BLOCK)))
```

(* * COMPILER STUFF)

```
(RPAQQ ACE.PIXPERWORD 16)

(RPAQ ACE.BITMAP.MASK (READARRAY 16 (QUOTE SMALLPOSP)
                                 0))

(65535 32768 49152 57344 61440 63488 64512 65024 65280 65408 65472 65504 65520 65528 65532 65534 NIL)
```

(* * LOW LEVEL COMPILER FNS)

```
(DEFINEQ

(ACE.COMPILE.FRAME
  [LAMBDA (BM.ORIG BM.CHANGED VERTICAL.BLOCK THRESHOLD)          (* MJD "28-May-86 16:35")
                                                                 (* MAIN ENTRY POINT FOR DIFFERENTIAL BITMAP COMPILING)
                                                                 (* WARNING: NO ERROR CHECKING DONE FROM HERE DOWN!)
                                                                 (* RETURNS A FRAMEPATCH LIST OF ACTUAL BITMAP CHANGES)
                                                                 (* RETURN FORMAT: ((BM X . Y)
                                                                 (BM X . Y) ...))
    (PROG (CHANGES)
          (SETQ CHANGES (ACE.SCAN.BITMAPS BM.ORIG BM.CHANGED VERTICAL.BLOCK))
          (AND CHANGES (SETQ CHANGES (ACE.MAX.REGIONS CHANGES THRESHOLD)))
          (SETQ CHANGES (ACE.EXTRACT CHANGES BM.CHANGED))
          (ACE.UPD.CONTROL.WINDOW (QUOTE OPERATION)
                 "DONE")
          (RETURN CHANGES])

(ACE.EXTRACT
  [LAMBDA (REGIONS BITMAP)                                       (* MJD "28-May-86 13:12")
          (* TAKES LIST OF REGIONS OF CHANGED AREAS AND MAKES INTO ACTUAL FRAMEPATCH LIST BY EXTRACTING FROM NEW BM)
    (PROG (TEMP.BITMAP LEFT BOTTOM WIDTH HEIGHT (FRAMEBLITLIST (CONS)))
          [COND
             ((NULL REGIONS)
              NIL)
             (T (for X in REGIONS do (BLOCK)
                                     (SETQ LEFT (fetch (REGION LEFT) of X))
                                     (SETQ BOTTOM (fetch (REGION BOTTOM) of X))
                                     (SETQ WIDTH (fetch (REGION WIDTH) of X))
                                     (SETQ HEIGHT (fetch (REGION HEIGHT) of X))
                                     (SETQ TEMP.BITMAP (BITMAPCREATE WIDTH HEIGHT 1))
                                     (BITBLT BITMAP LEFT BOTTOM TEMP.BITMAP 0 0 WIDTH HEIGHT (QUOTE INPUT)
                                            (QUOTE REPLACE))
                                     (NCONC1 FRAMEBLITLIST
                                            (create ACE.BLIT
                                                   BITMAP _ TEMP.BITMAP
                                                   XCOOR _ LEFT
                                                   YCOOR _ BOTTOM]
          (RETURN (CDR FRAMEBLITLIST])

(ACESETTHRESHOLD
```

```
  [LAMBDA NIL                                                    (* MD "19-Jun-85 15:46")
    (PROG NIL
      LOOP
          (SETQ ACE.AREA.THRESHOLD (RNUMBER (CONCAT "Compression factor(0-100), currently " ACE.AREA.THRESHOLD
                                                    ": ")))
          (IF (OR (ILESSP ACE.AREA.THRESHOLD 0)
                  (IGREATERP ACE.AREA.THRESHOLD 100))
              THEN (GO LOOP))
          (RETURN ACE.AREA.THRESHOLD])

)


          (* * REGION MAXING ROUTINES)

(DEFINEQ

(ACE.MAX.REGIONS
  [LAMBDA (REGIONS THRESHOLD)                                    (* MJD "28-May-86 13:11")

          (* Merges the changed regions picked out by ACE.SCAN.BITMAPS;
          REGIONS are small areas of change, THRESHOLD specifies how much bitmap area must by "good" for a combination
          (i.e. 100 -
          THRESHOLD is how much space may be wasted in combining two REGIONS);
          Both ARGS required!)
                                                                 (* RETURNS A LIST OF (REGION REGION ...))
    (PROG (BEST.POSS)
      LOOP
          (BLOCK)
          (COND
             [(IGREATERP (LENGTH REGIONS)
                         1)
              (SETQ BEST.POSS (ACE.PICK.BEST.REGION REGIONS))
              (COND
                 ((IGEQ (CADDR BEST.POSS)
                        THRESHOLD)
                  (NCONC1 REGIONS (CONS (UNIONREGIONS (CAAR BEST.POSS)
                                                      (CAADR BEST.POSS))
                                        (CADDR BEST.POSS)))
                  (DREMOVE (CAR BEST.POSS)
                           REGIONS)
                  (DREMOVE (CADR BEST.POSS)
                           REGIONS))
                 (T (GO DONE]
             (T (GO DONE)))
          (GO LOOP)
      DONE
          (RETURN (for X in REGIONS collect (CAR X])

(ACE.PICK.BEST.REGION
  [LAMBDA (REGIONS)                                              (* MJD "28-May-86 13:11")

          (* SLOWest part of animation! Selects the most efficient (i.e. least amount of wasted space resulting from combining two
          regions) combination of two regions from REGIONS; First tries to find 100% match up;
          failing that goes for highest efficiency)

    (PROG (EFFICIENCY BEST.SO.FAR)
          [COND
             [(for X in REGIONS
                 thereis (AND (EQP (CDR X)
                                   100)
                              (for Y in (CDR (MEMB X REGIONS))
                                 thereis (AND (PROG1 T (BLOCK))
                                              (EQP (CDR Y)
                                                   100)
                                              [OR [AND (EQP (fetch (REGION LEFT) of (CAR X))
                                                            (fetch (REGION LEFT) of (CAR Y)))
                                                       (OR [EQP (fetch (REGION BOTTOM) of (CAR Y))
                                                                (ADD1 (fetch (REGION TOP) of (CAR X]
                                                           (EQP (ADD1 (fetch (REGION TOP) of (CAR Y)))
                                                                (fetch (REGION BOTTOM) of (CAR X]
                                                  (AND (EQP (fetch (REGION BOTTOM) of (CAR X))
                                                            (fetch (REGION BOTTOM) of (CAR Y)))
                                                       (OR [EQP (fetch (REGION LEFT) of (CAR Y))
                                                                (ADD1 (fetch (REGION RIGHT) of (CAR X]
                                                           (EQP (ADD1 (fetch (REGION RIGHT) of (CAR Y)))
                                                                (fetch (REGION LEFT) of (CAR X]
                                              (SETQ BEST.SO.FAR (LIST X Y 100]
              (T (SETQ BEST.SO.FAR (QUOTE (NIL NIL -1)))
                 (for X in REGIONS do (for Y in (CDR (MEMB X REGIONS))
                                         do ((SETQ EFFICIENCY (IQUOTIENT [ITIMES 100 (IPLUS (ACE.COMPUTE.AREA
                                                                                             (CAR X)
                                                                                             (CDR X))
                                                                                            (ACE.COMPUTE.AREA
                                                                                             (CAR Y)
                                                                                             (CDR Y]
```

```
                                                    (ACE.COMPUTE.AREA (UNIONREGIONS (CAR X)
                                                                                    (CAR Y))
                                                                      100)))
                                            (AND (IGREATERP EFFICIENCY (CADDR BEST.SO.FAR))
                                                 (SETQ BEST.SO.FAR (LIST X Y EFFICIENCY]
          (RETURN BEST.SO.FAR])
```

(ACE.COMPUTE.AREA

```
  [LAMBDA (REGION EFF)                                          (* MJD "28-May-86 13:10")
    (BLOCK)
    (IQUOTIENT (ITIMES (ffetch (REGION WIDTH) of REGION)
                       (ffetch (REGION HEIGHT) of REGION)
                       EFF)
           100])

)


          (* * LOW LEVEL BITMAP COMPARISON)

(DEFINEQ

(ACE.SCAN.BITMAPS
  [LAMBDA (BM.ORIG BM.NEW BLOCKINGHEIGHT)                       (* PmT "25-Apr-85 15:14")

          (* Compares BM.ORIG and BM.NEW in one word (ACE.PIXPERWORD bits;
          16) by BLOCKINGHEIGHT rectangles. Note masking when get to last word in bitmap and compression of region below
          ACE.PIXPERWORD (16); All ARGS required; BM.ORIG and BM.NEW must have the same dimensions!)
                                                                (* RETURNS A LIST OF TYPE (REGION . 100))
    (PROG [TEMP.ENTRY (BM.WIDTH (ffetch BITMAPWIDTH of BM.ORIG))
                  (CHANGED.REGIONS (CONS))
                  (RASTERWIDTH (SUB1 (ffetch BITMAPRASTERWIDTH of BM.ORIG)))
                  (HEIGHT (SUB1 (ffetch BITMAPHEIGHT of BM.ORIG)))
                  (ALLMASK (ELT ACE.BITMAP.MASK 0))
                  (PARTIALMASK (ELT ACE.BITMAP.MASK (IMOD (ffetch BITMAPWIDTH of BM.ORIG)
                                                          ACE.PIXPERWORD]
          [while (ILESSP Y HEIGHT) bind (Y _ 0)
             do [for HORZ.BLOCK from 0 to RASTERWIDTH
                   do (AND [SETQ TEMP.ENTRY (COND
                                              ((EQP HORZ.BLOCK RASTERWIDTH)
                                               (ACE.SCAN.PRIMBLOCKS BM.ORIG BM.NEW HORZ.BLOCK Y BLOCKINGHEIGHT
                                                      PARTIALMASK))
                                              (T (ACE.SCAN.PRIMBLOCKS BM.ORIG BM.NEW HORZ.BLOCK Y
                                                      BLOCKINGHEIGHT ALLMASK]
                            (NCONC1 CHANGED.REGIONS (CONS [CREATEREGION (ITIMES HORZ.BLOCK ACE.PIXPERWORD)
                                                                        (CAR TEMP.ENTRY)
                                                                        (IMIN ACE.PIXPERWORD (IDIFFERENCE BM.WIDTH
                                                                                                          (ITIMES HORZ.BLOCK
                                                                                                                  ACE.PIXPERWORD
                                                                                                           )))
                                                                        (ADD1 (IDIFFERENCE (CDR TEMP.ENTRY)
                                                                                           (CAR TEMP.ENTRY]
                                                          100]
                (SETQ Y (IPLUS Y BLOCKINGHEIGHT))
                (SETQ BLOCKINGHEIGHT (IMIN BLOCKINGHEIGHT (ADD1 (IDIFFERENCE HEIGHT Y]
          (RETURN (CDR CHANGED.REGIONS])
```

(ACE.SCAN.PRIMBLOCKS

```
  [LAMBDA (BM1 BM2 WORDOFFSET Y0 BLOCKH MASK)                   (* MJD "28-May-86 13:05")

          (* Does the actual comparison of primitive areas in the two bitmaps BM1 and BM2 ;
          WORDOFFSET is the raster word offset; Y0 is the low scanline and
          (IPLUS Y0 BLOCKH) is the hi one; MASK is usually $FFFF, otherwise it is used to ignore extra bits trailing off the end of the
          last raster word)

    (PROG [TEMP1 (MAXY (SUB1 (IPLUS Y0 BLOCKH]
          [SETQ TEMP1 (for Y from Y0 to MAXY thereis (NOT (EQP (LOGAND (LOGXOR (ACE.FETCH.BLOCK BM1 WORDOFFSET Y)
                                                                               (ACE.FETCH.BLOCK BM2 WORDOFFSET Y))
                                                                         MASK)
                                                              0]
          (RETURN (AND TEMP1 (CONS TEMP1 (for Y from MAXY to TEMP1 by -1
                                            thereis (NOT (EQP (LOGAND (LOGXOR (ACE.FETCH.BLOCK BM1 WORDOFFSET Y)
                                                                              (ACE.FETCH.BLOCK BM2 WORDOFFSET Y))
                                                                        MASK)
                                                             0])
```

(ACE.FETCH.BLOCK

```
  [LAMBDA (BITMAP WORDOFFSET VERTICAL)                          (* MJD "28-May-86 13:04")
                                                                (* Nabs a word from bitmap on line VERTICAL with word offset
                                                                WORDOFFSET)
    (BLOCK)
    (\GETBASE (\ADDBASE (ffetch BITMAPBASE of BITMAP)
                        (ITIMES (IDIFFERENCE (ffetch BITMAPHEIGHT of BITMAP)
                                             (ADD1 VERTICAL))
```

```
                            (ffetch BITMAPRASTERWIDTH of BITMAP)))
          WORDOFFSET])

)

(PUTPROPS ACE-PRIM COPYRIGHT ("Xerox Corporation" 1986))
```

FUNCTION INDEX

VARIABLE INDEX