PROGETTO ETIS: CURVE, PROCEDURA ELABORATIVA DI SEGMENTAZIONE MEDIANTE RICONOSCIMENTO DI CONTORNI.

Enrico Fantini

B4-06

1988

CURVE: PROCEDURA ELABORATIVA DI SEGMENTAZIONE MEDIANTE RICONOSCIMENTO DI CONTORNI

E. Fantini
Istituto di Elaborazione della Informazione - C.N.R. - Pisa

INTRODUZIONE

Questa relazione illustra e documenta la procedura elaborativa CURVE che e' stata sviluppata nell'ambito di un contratto di collaborazione tecnico-scientifica stipulato fra l'Istituto di Elaborazione della Informazione e la Societa' Aeritalia, Gruppo Velivoli da Trasporto.

Questo pacchetto software fa parte delle procedure elaborative per l'analisi di immagini ed e' stato particolarmente orientato verso l'applicazione del controllo non invasivo di materiali e prodotti aeronautici.

CURVE e' stato sviluppato in due versioni: una piu' ampia operante nel sistema di sviluppo per il controllo non distruttivo ETIS progettato dall'I.E.I. e fornito all'AERITALIA; una piu' ridotta operante su personal IBM AT o compatibile in ambiente MS/DOS.

In quest'ultima versione CURVE fa parte di un insieme di moduli elaborativi, progettati e realizzati dall'I.E.I., che dovra' essere inserito in un sistema software di gestione attualmente in fase di avanzata progettazione.

DESCRIZIONE DELLA PROCEDURA CURVE

La procedura elaborativa CURVE e' stata sviluppata per l'esecuzione di operazioni di segmentazione regionale al fine di estrarre dalle strutture discriminate i parametri necessari ad una loro eventuale classificazione.

Il tipo di segmentazione prescelta si basa sulla determinazione dei contorni delle strutture in esame

mediante l'assegnazione di una soglia di discrimanazione individuabile in modo interattivo all'interno della procedura, oppure in modo automatico mediante l'impiego di moduli elaborativi previsti nelle librerie del sistema GEPETIS.

CURVE e' costituita da tre moduli richiamabili anche individualmente: ISOF, DISCU, CARPE. Questi moduli sono scritti in linguaggio FORTRAN-77 ad esclusione di alcune routines scritte in ASSEMBLER; i moduli possono essere selezionati da un menu che viene presentato sullo schermo del terminale alfanumerico del sistema di calcolo.

Nella versione 1.0, sviluppata sul sistema di calcolo IBM AT, la procedura CURVE richiede la seguente configurazione hardware e software:

IBM XT/AT o compatibile
640 KByte RAM
CGA con relativo monitor
Scheda MATROX PIP-1024 con relativo video
H.D. almeno 20 MByte
Stampante IBM Proprinter

S.O. MS-DOS 3.2 o successivi
Libreria PIP_EH per primitive PIP 1024 MATROX

Descrizione moduli

ISOF: MODULO PER LA DETERMINAZIONE DELLE CURVE DI CONTORNO

Questo modulo serve per la determinazione delle aree interessate, la loro eventuale memorizzazione su disco e copia grafica su stampante IBM Proprinter.

Il programma parte dal presupposto che in una memoria di quadro della scheda Matrox PIP-1024 sia residente un'immagine precedentemente caricata.

Il modulo elaborativo richiede una serie di parametri quali: la memoria video dove risiede l'immagine da trattare,

il sottoinsieme di interesse, il valore di quantizzazione dell'area che interessa rilevare, la memoria video su cui si vuole visualizzare l'area discriminata e i parametri per il controllo della tracciatura delle curve di contorno determinate.

A questo punto il programma inizia a rilevare le aree richieste che vengono visualizzate sulla memoria di quadro selezionata.

Terminato questo compito viene chiesto all'operatore se si vuole una copia su carta delle aree rilevate e se si desidera l'archiviazione di queste ultime su file di disco.

DESCRIZIONE DELL'ALGORITMO USATO PER LA DETERMINAZIONE DI CURVE DI LIVELLO

L'informazione presente in un'immagine fotografica in B/N si presenta come una gradazione continua di toni di grigio.

Matematicamente e` una funzione continua f(x,y) rappresentante il valore del livello di grigio del punto (x,y) del piano dell'immagine.

Se nell'immagine tutti gli oggetti hanno le stesse variazioni di grigio e sono tutti distinti dal fondo, cioe` a fuoco, lungo i loro contorni la funzione variera` rapidamente.

E` possibile in questo caso determinare un livello di grigio K che rappresentera` il livello dei punti di contorno degli oggetti presenti nell'immagine.

Generalmente pero`, sia perche` gli oggetti di un'immagine non sono tutti a fuoco sia perche` spesso si vogliono determinare i contorni di oggetti con diversi livelli di grigio, e` necessario rilevare contorni con differenti valori di livello.

La ricerca delle curve di livello K della superficie immagine equivale a tagliare l'immagine ad un livello K

cioe` prelevare le coordinate soltanto dei punti con tale livello.

A questo punto e` necessario ricordare che un'immagine viene sempre acquisita, con un processo di campionamento e di quantizzazione, sotto forma discreta cioe` con l'estrazione dall'insieme continuo dei valori della funzione f(x,y) di un sottoinsieme discreto di campioni.

Dato che le linee di livello di una superficie discreta sono poco significative, tale superficie viene approssimata con una superficie poliedrica continua per poi approssimare le linee di livello di quest'ultima alle curve di livello discrete.

Se tale superficie poliedrica e` vista come una superficie continua composta dall'unione di tanti triangoli, la ricerca delle linee di livello K puo` essere ricondotta alla ricerca della intersezione fra i triangoli componenti la poliedrica e il piano Z=K, cioe` visto che tale intersezione sara` un segmento e` sufficiente determinare l'intersezione dei lati di ogni triangolo con il piano di livello. Ad operazione conclusa ogni curva di livello K sara` definita da un insieme di spezzate aventi per vertici dette intersezioni.

Entriamo ora piu` dettagliatamente nell'algoritmo.

Per ogmi terna di punti continui [due punti continui (X_i,Y_j,Z_{ij}) e (X_k,Y_h,Z_{kh}) distinti si dicono contigui se $|k-i| \leq 1$ e $|j-h| \leq 1$] (X_k,Y_h,Z_{kh}), (X_i,Y_j,Z_{ij}) e (X_l,Y_m,Z_{lm}) rappresentanti i vertici di un triangolo della poliedrica, ordinati in modo che $Z_{hm} \leq Z_{ij} \leq Z_{lm}$, viene controllato il verificarsi o meno delle seguenti relazioni:

$$Z_{hn} \leq K \leq Z_{ij}$$
$$Z_{ij} \leq K \leq Z_{lm}$$
$$Z_{hm} \leq K \leq Z_{lm}$$

Le possibilita` sono:

1) Nessuna relazione e` verificata, cioe` il triangolo non interseca il piano di livello.
2) Soltanto due relazioni sono verificate, cioe` il triangolo interseca il piano in un segmento che non e` nessuno dei suoi lati.
3) Tutte le relazioni sono verificate con due uguaglianze, $Z_{hn}=Z_{ij}=K$ oppure $Z_{ij}=Z_{lm}=K$, cioe` un lato del triangolo e` interamente sul piano di livello.
4) Tutte le relazioni sono verificate con uguaglianza, $Z_{hn}=Z_{ij}=Z_{lm}=K$, cioe` l'intero triangolo e` sul piano di livello.

Al verificarsi del caso 1 si passa a considerare il successivo triangolo.

Nel caso 2 si vanno a considerare i lati che intersecano il piano di livello.

Ammettiamo che uno di tali lati sia il segmento di estremi (X_h,Y_n,Z_{hn}) e (X_i,Y_j,Z_{ij}), viene effettuato l'interpolazione lineare fra le coordinate:

$$X_t = X_h + \frac{K - Z_{hn}}{Z_{ij} - Z_{hn}} (X_i - X_h)$$

$$Y_t = Y_n + \frac{K - Z_{hn}}{Z_{ij} - Z_{hn}} (Y_i - Y_n)$$

Il punto (X_t,Y_t,K) rappresenta l'interpolazione tra il lato e il piano di livello.

Riportando l'operazione per l'altro lato viene individuato il secondo estremo del segmento di intersezione tra il triangolo e il piano di livello.

Nel caso 3 gli estremi del segmento di intersezione tra triangolo e piano sono gli estremi stessi del lato che giace sul piano. Infatti applicando l'interpolazione per gli altri

due lati del triangolo si determinano proprio gli estremi del terzo lato.

Al verificarsi del caso 4 le formule di interpolazione non sono definite per nessun punto di vertice del triangolo, comunque i tre segmenti di intersezione vengono determinati tramite l'applicazione dell'algoritmo sulle terne (triangoli) contigue a quelle in esame.

Iterando il procedimento per tutti i triangoli che compongono la poliedrica e` possibile individuare tutti i segmenti delle curve di livello K.

Una volta concluso l'intero rilevamento delle curve a livello K e` necessario poter distinguere le singole curve per poter poi effettuare su di esse tutti quei rilevamenti di parametri necessari per ulteriori applicazioni.

Per fare questo e` necessario ordinare i segmenti rilevati con il procedimento precedentemente descritto.

Se indichiamo un segmento con A --> B dove A e B sono gli estremi e --> indica l'orientamento da A a B, il procedimento di ordinamento puo` essere riassunto nel modo seguente:

1) Viene individuato un segmento A --> B considerato come segmento iniziale della curva da inseguire.
2) Viene ricercato un segmento con estremo B, questo puo` essere C --> B oppure B --> C; nel primo caso vengono scambiati gli estremi per poi proseguire, come per il secondo caso, alla ricerca di un segmento con estremo C.
   Con questo metodo vengono congiunti segmenti consecutivi, cioe` con due estremi coincidenti, precedentemente acquisiti in maniera non ordinata scandendo da sinistra a destra e dal basso all'alto la superficie immagine.
   Tramite questa tecnica di scansione vengono determinati segmenti appartenenti a differenti curve, comunque il loro ordinamento e` possibile dal fatto che ogni lato di ciascun triangolo appartiene a due triangoli contigui ed

e` quindi esaminata due volte, percio` ciascun segmento della curva di livello avra` un suo estremo coincidente con l'estremo del segmento successivo.

3) Al termine dell'inseguimento si possono verificare due casi:
   - viene trovato un segmento un cui estremo coincide con A e percio` la curva e` chiusa
   - non vengono piu` trovati segmenti in grado di soddisfare il punto 2 e pertanto la curva e` aperta.

   In entrambi i casi viene considerato concluso l'inseguimento per quella curva e si puo` iniziare l'ordinamento dei segmenti di un'altra curva.

DISCU: MODULO PER LA VISUALIZZAZIONE SU VIDEO DELLE CURVE (AREE) ARCHIVIATE SU DISCO

Con questo programma e` possibile visualizzare su video le curve archiviate su disco con il modulo precedentemente descritto.

E` possibile effettuare una numerazione delle curve rilevate e una loro copia su carta una volta visualizzate su video.

Il programma inizia con la richiesta del nome del file che contiene le curve, vengono poi chiesti i parametri necessari per determinare la memoria di quadro su cui devono essere visualizzate le curve.

A questo punto vengono tracciate le curve con l'eventuale numerazione dopodiche` puo` essere effettuata la copia su carta.

CARPE: MODULO PER LA DETERMINAZIONE DEI PARAMETRI DELLE CURVE

Nella revisione della procedura in uso al momento della stesura di questa nota (Rev.1.0) e` possibile, con questo programma, calcolare soltanto area e perimetro delle curve archiviate precedentemente su disco.

E` comunque previsto per la revisione successiva un ampliamento di queste funzioni.

Il programma inizia con la richiesta del nome del file su cui sono state memorizzate precedentemente le curve, poi viene richiesto il dispositivo (video alfanumerico o stampante) di uscita per i dati calcolati.

L'uscita consiste in una tabella dove sono riportate le curve numerate in ordine crescente con i relativi parametri calcolati dal programma.

```
echo off
:loop
cls
menu
cls
if errorlevel 3 goto exit3
if errorlevel 2 goto exit2
if errorlevel 1 goto exit1
if errorlevel Ø goto exit
:exit3
carpe
goto loop
:exit2
discu
goto loop
:exit1
isof
goto loop
:exit
```

```
      implicit integer*2 (i-n)
1     call cls
      write(*,2)
2     format(///27x'####  PROCEDURA ISOF  ####',///,
     -10x,'1 - Determinazione delle curve',/,
     -10x,'2 - Visualizzazione curve archiviate',/,
     -10x,'3 - Determinazione area, perimetro curve archiviate',//,
     -10x,'0 - Fine',//,' [0] >>>> '$)
      read(*,3) i
3     format(i1)
      if(i.lt.0.or.i.gt.3) then
          call bel
          goto 1
      end if
      if(i.eq.0) stop 0
      if(i.eq.1) stop 1
      if(i.eq.2) stop 2
      if(i.eq.3) stop 3
      end
```

```
c*****************************************************************************
c***                                                                       ***
c***          Determinazione delle isofote di un'immagine                  ***
c***          su frame memory della PIP-1Ø24.                              ***
c***          L'uscita e' costituita da linee tracciate sulla              ***
c***          immagine stessa o su altra memoria di quadro.                ***
c***                                                                       ***
c***    Rev. 1.Ø - Dicembre 1987                                           ***
c***                                                                       ***
c***    Autore : FANTINI Enrico - I.E.I.  C.N.R.  Pisa                     ***
c***                                                                       ***
c*****************************************************************************
      program tisof
      implicit integer*2 (i-n)
      integer*4 i3
      integer*2 quad,quadu,buffer(512),soglia
      character ok*1,nop*2
      common b3(4ØØØØ),b4(4ØØØØ),i3
      write(*,222)
222   format(/2Øx,'****     Isofote di una immagine      ****'/////,16x
     *,'Programma che determina le isofote di una immagine'/,16x,
     *'presente sulla memoria di quadro della PIP-1Ø24'/,16x,
     *'L''uscita e'' costituita da linee tracciate sulla',/,16x,
     *'stessa memoria o su altra memoria della PIP.',/,16x,
     *'E` possibile archiviare le curve su disco.',///,13x,
     *'**** richiesta dei parametri per la elaborazione  ****'/)
      call setind(Ø)
46    lun=512
      nrec=512
      npr=6
      lro=256
43    write(*,42)
42    format(//' Quadrante di ingresso (Ø-3) [Ø] ( < Ø = FINE) ? :_ '$)

      read(*,58) quad
58    format(i2)
      if(quad.lt.Ø) goto 9999
      if(quad.gt.3) then
          call bel
          goto 43
      end if
      call displ(quad)
44    write(*,45)
45    format(/' o.k. (s/n) [s] ? :_ '$)
      read(*,'(a)') ok
      if(ok.eq.' ') ok='s'
      if(ok.eq.'n'.or.ok.eq.'n') goto 43
      if(ok.ne.'s'.and.ok.ne.'s') then
          call bel
          goto 44
      end if
81    write(*,8Ø)
8Ø    format(/' Colonna iniziale,colonna finale,passo [Ø,511,1] :_ '$)
      read(*,79)ixi,ixf,ipx
79    format(3i4)
```

```
      if(ixi.eq.0) ixi=0
      if(ixf.eq.0) ixf=lun-1
      if(ipx.eq.0) ipx=1
      if(ixi.lt.0.or.ixf.gt.lun-1.or.ixi.gt.ixf)then
          call bel
          goto 81
      end if
83    write(*,82)
82    format(/' Riga iniziale,riga finale,passo [0,511,1] :_ '$)
      read(*,79)iyi,iyf,ipy
      if(iyi.eq.0) iyi=0
      if(iyf.eq.0) iyf=nrec-1
      if(ipy.eq.0) ipy=1
      if(iyi.lt.0.or.iyf.gt.nrec-1.or.iyi.gt.iyf) then
          call bel
          goto83
      end if
c********** determinazione del valore delle linee di livello ricercate
21    write(*,20)
20    format(/' Valore isofota ? :_'\)
      read(*,*) ivis
      if(ivis.lt.0.or.ivis.gt.255) then
          call bel
          goto 21
      end if
      vis=float(ivis)+0.1
30    write(*,31)
31    format(//' Quadrante di uscita (0-3) [0] ? :  '$)
      read(*,58) quadu
      if(quadu.lt.0.or.quadu.gt.3) then
          call bel
          goto 30
      end if
      call displ(quadu)
32    write(*,45)
      read(*,'(a)') ok
      if(ok.eq.' ') ok='s'
      if(ok.eq.'n'.or.ok.eq.'n') goto 30
      if(ok.ne.'s'.and.ok.ne.'s') then
          call bel
          goto 32
      end if
33    write(*,34)
34    format(/' Cancello schermo (s/n) [s] ? :  '$)
      read(*,'(a)') ok
      if(ok.eq.' ') ok='s'
      if(ok.eq.'s'.or.ok.eq.'S') then
35        write(*,36)
36        format(/' Valore di cancellazzione (0-255) [0] :_ '$)
          read(*,375) ivz
375       format(i3)
          if(ivz.lt.0.or.ivz.gt.255) then
              call bel
              goto 35
          end if
```

```
            call set(ivz)
        endif
991     write(*,990)
990     format(/' Valore di tracciatura (0,255) [0] :_ '$)
        read(*,375)ivt
        if(ivt.lt.0.or.ivt.gt.255) then
            call bel
            goto 991
        end if
37      write(*,38)
38      format(/' Risoluzione alta, bassa (a,b) [a] :_ '$)
        read(*,'(a)') ok
        if(ok.eq.' ') ok='a'
        if(ok.eq.'a'.or.ok.eq.'A') then
            it=1
        else
            if(ok.eq.'b'.or.ok.eq.'B') then
                it=2
            else
                call bel
                goto 37
            end if
        end if
c********** scansione della immagine per la ricerca delle coordinate
c********** dei punti componenti le isofote
78      continue
        nerr=0
        call cerca(quad,quadu,vis,ixi,ixf,ipx,iyi,iyf,ipy,ivt,it,nerr)
        if(nerr.ne.0) goto 43
630     write(*,631)
631     format(/' Copio su stampante le curve rilevate (s/n) [n]: '$)
        read(*,'(a)') ok
        if(ok.eq.' ') ok='n'
        if(ok.eq.'s'.or.ok.eq.'S') then
            soglia=ivt-1
            inter=2
            call setscr(quadu)
        call fp_tp(soglia,inter,ixi,ixf,ipx,iyi,iyf,ipy,buffer,nerr)

        else
            if(ok.ne.'n'.and.ok.ne.'N') goto 630
        endif
603     write(*,602)
602     format(/' Provo un altro valore di isofota (s/n) [n] ? :_ '$)
        read(*,'(a)')ok
        if(ok.eq.' ') ok='n'
        if(ok.eq.'s'.or.ok.eq.'S')goto 43
        if(ok.ne.'n'.and.ok.ne.'N') then
            call bel
            goto 603
        end if
610     write(*,611)
611     format(/' Archivio le curve (s/n)  [s] ? : '$)
        read(*,'(a)') ok
        if(ok.eq.' ') ok='s'
```

```
      if(ok.eq.'n'.or.ok.eq.'N') goto 616
      if(ok.ne.'s'.and.ok.ne.'S') then
          call bel
          goto 610
      end if
612   write(*,613)
613   format(/
     -' Archivio anche le curve che intersecano i bordi (s/n) [n]? : '$)

      read(*,'(a)') ok
      if(ok.eq.' ') ok='n'
      if(ok.eq.'s'.or.ok.eq.'S') then
          nop='SI'
      else
         if(ok.eq.'n'.or.ok.eq.'N') then
             nop='NO'
         else
             call bel
             goto 612
         end if
      end if
614   write(*,615)
615   format(/' Numero minimo di punti per ogni curva : '$)
      read(*,*) nn
      if(nn.lt.1) then
          call bel
          goto 614
      end if
      call ordm(nop,nn,ixi,ixf,iyi,iyf,lro,npr)
616   write(*,617)
617   format(///' Ancora (s/n) [n] ? :_ '$)
      read(*,'(a)') ok
      if(ok.eq.' ') ok='n'
      if(ok.eq.'s'.or.ok.eq.'S') then
          goto 46
      else
          if(ok.ne.'n'.and.ok.ne.'N') then
              call bel
              goto 616
          end if
      end if
9999  end
c
c*************************************************************************
c********** scansione della immagine per la ricerca delle coordinate
c********** dei punti componenti le isofote
c*************************************************************************
c
      subroutine cerca(quad,quadu,vis,ixi,ixf,ipx,
     -                 iyi,iyf,ipy,ivt,it,nerr)
      implicit integer*2 (i-n)
      integer*4 i3
      dimension b1(512),b2(512),f(3),x(3),y(3)
      integer*2 ia(512),quad,quadu
      common b3(40000),b4(40000),i3
```

```
      ierr=0
      I3=1
      lun=512
      npu=ixf-ixi+1
      call setscr(quad)
      call rowri(ixi,iyi,npu,ia(ixi+1))
      do 1 j=1,512
1         b1(j)=float(ia(j))
      do 1000 icr=iyi+ipy,iyf,ipy
      call rowri(ixi,icr,npu,ia(ixi+1))
      do 2 j=1,512
2         b2(j)=float(ia(j))
      call setscr(quadu)
      goto(23,22)it
22    do 20 k=ixi,ixf-ipx,ipx
      f(1)=b1(k)
      f(2)=b1(k+ipx)
      f(3)=b2(k)
      x(1)=k
      x(2)=k+ipx
      x(3)=k
      y(1)=icr-ipy
      y(2)=icr-ipy
      y(3)=icr
      if(ista(f(1),f(2),f(3),vis).ne.0) then
          call trian(x,y,f,vis,ivt,ierr)
          if(ierr.ne.0) nerr=1
      end if
      f(1)=b1(k+ipx)
      f(2)=b2(k)
      f(3)=b2(k+ipx)
      x(1)=k+ipx
      x(2)=k
      x(3)=k+ipx
      y(1)=icr-ipy
      y(2)=icr
      y(3)=icr
      if(ista(f(1),f(2),f(3),vis).ne.0) then
          call trian(x,y,f,vis,ivt,ierr)
          if(ierr.ne.0) nerr=1
      end if
20    continue
      goto 28
23    do 24 k=ixi,ixf-ipx,ipx
      xm=float(k)+(float(ipx))/2.
      ym=float(icr)-(float(ipy))/2.
      b=(b1(k)+b1(k+ipx)+b2(k)+b2(k+ipx))/4.
      f(1)=b
      x(1)=xm
      y(1)=ym
      f(2)=b1(k)
      x(2)=k
      y(2)=icr-ipy
      f(3)=b1(k+ipx)
      x(3)=k+ipx
```

```
      y(3)=icr-ipy
      if(ista(f(1),f(2),f(3),vis).ne.0) then
          call trian(x,y,f,vis,ivt,ierr)
          if(ierr.ne.0) nerr=1
      end if
      f(2)=b2(k+ipx)
      x(2)=k+ipx
      y(2)=icr
      if(ista(f(1),f(2),f(3),vis).ne.0) then
          call trian(x,y,f,vis,ivt,ierr)
          if(ierr.ne.0) nerr=1
      end if
      f(3)=b2(k)
      x(3)=k
      y(3)=icr
      if(ista(f(1),f(2),f(3),vis).ne.0) then
          call trian(x,y,f,vis,ivt,ierr)
          if(ierr.ne.0) nerr=1
      end if
      f(2)=b1(k)
      x(2)=k
      y(2)=icr-ipy
      if(ista(f(1),f(2),f(3),vis).ne.0) then
          call trian(x,y,f,vis,ivt,ierr)
          if(ierr.ne.0) nerr=1
      end if
24    continue
28    do 21 i=1,lun
21    b1(i)=b2(i)
      call setscr(quad)
1000  continue
      b3(i3)=30000.
      b4(i3)=30000.
      return
      end
c
c*************************************************************************
c*****  SUBROUTINE PER LA RICERCA DELLE ISOFOTE A LIVELLO SUB-PIXEL
c*************************************************************************
c
      subroutine trian(x,y,f,vis,ivt,ierr)
      implicit integer*2 (i-n)
      common b3(40000),b4(40000),i3
      integer*4 i3
      dimension x(1),y(1),f(1),xp(3),yp(3),ix(3),iy(3)
      character ris*1
      k=0
      do 1 i=1,3
      j=i+1
      if(i.eq.3)j=1
      if(f(i)-f(j))2,1,3
2     if(vis.lt.f(i).or.vis.gt.f(j))go to1
      k=k+1
      xp(k)=(vis-f(i))*(x(j)-x(i))/(f(j)-f(i))+x(i)
      yp(k)=(vis-f(i))*(y(j)-y(i))/(f(j)-f(i))+y(i)
```

```
      goto1
3     if(vis.lt.f(j).or.vis.gt.f(i))goto1
      k=k+1
      xp(k)=(vis-f(j))*(x(i)-x(j))/(f(i)-f(j))+x(j)
      yp(k)=(vis-f(j))*(y(i)-y(j))/(f(i)-f(j))+y(j)
1     continue
      if(k-2)10,11,12
10    return
11    if(xp(1).eq.xp(2).and.yp(1).eq.yp(2))return
      goto13
12    if(xp(1).ne.xp(2).or.yp(1).ne.yp(2))goto13
      xp(2)=xp(3)
      yp(2)=yp(3)
13    continue
c************** visualizzazione su video grafico del segmento
c************** di isofota trovato
      do 33 jj=1,2
      ix(jj)=xp(jj)+.5
      iy(jj)=yp(jj)+.5
      b3(i3)=xp(jj)
      b4(i3)=yp(jj)
      i3=i3+1
      if(i3.gt.40000) then
          if(ierr.eq.0) then
              write(*,34)
34            format(///' ATTENZIONE. Buffer archiviazione curve pieno.',/,
     -        ' Il programma prosegue esclusivamente per via grafica.',/,
     -        ' non e` possibile archiviare le curve.',//,
     -        ' #####  PREMERE UN TASTO PER CONTINUARE  #####'$)
              call letcar(ris)
              ierr=1
          end if
          i3=1
      end if
33    continue
      call line(ix(1),iy(1),ix(2),iy(2),ivt)
      return
      end
c*******************************************************************************
      function ista(f1,f2,f3,f)
      implicit integer*2 (i-n)
      ista=-1
      if(f1.le.f)goto1
      if(f2.le.f)return
      if(f3.le.f)return
      goto2
1     if(f2.ge.f)return
      if(f3.ge.f)return
2     ista=0
      return
      end
c
c*******************************************************************************
c*****  SUBROUTINE PER L'ORDINAMENTO IN X, Y CRESCENTI DELLE
c*****  COORDINATE DELLE ISOFOTE DA ARCHIVIARE
```

```
c*********************************************************************
c
      subroutine ordm(nop,nn,ixi,ixf,iyi,iyf,lro,npr)
      implicit integer*2 (i-n)
      common b3(40000),b4(40000),i3
      dimension x(512),y(512),itab(2560)
      character nop*2,nfile*12,ris*1
      integer*4 i3
500   write(*,510)
510   format(//' Quale e` il nome del file di archiviazione',/,
     -' delle curve (max. 12 caratteri) ? : '$)
      read(*,'(a)') nfile
      if(nfile.eq.'            ') then
          call bel
          goto 500
      end if
      open(20,file=nfile,access='direct',status='new',recl=lro*4,
     -iostat=ios)
      if(ios.ne.0) then
          call bel
          write(*,501) ios,nfile
501       format(///' E` stato riscontrato un errore (num. ',i6,')',/,
     -    ' nella apertura del file -',a,'-',//,
     -    ' -----  premere un tasto per continuare  -----'$)
          call letcar(ris)
          close(20)
          goto 500
      end if
      do 1 k=1,2560
          itab(k)=0
1     continue
      krec=npr
      ic=1
      iic=1
      kk=1
      m=0
      if(nop.eq.'no') goto20
      kk=0
      rex=float(iyf)
      rux=float(ixf)
      ax=float(ixi)
      ay=float(iyi)
16    l=0
      k1=0
      i3=iic
      do 77 k=1,lro
          x(k)=0
          y(k)=0
77    continue
      krn=krec
3     if(b3(i3).ne.30000.)goto 4
      kk=1
      ic=1
      goto20
4     if(b3(i3).gt.0.) goto30
```

```
31      i3=i3+1
        goto3
3Ø      if(b3(i3).eq.ax)goto55
        if(b4(i3).eq.ay)goto55
        if(b4(i3).eq.rex)goto55
        if(b3(i3).eq.rux)goto55
        goto31
2Ø      l=Ø
        k1=Ø
        i3=ic
        do 37 k=1,lro
            x(k)=Ø
            y(k)=Ø
37      continue
        krn=krec
23      if(b3(i3).ne.3ØØØØ.)goto 24
        do 23Ø kn9=1,5
        k9=lro*(kn9-1)+1
        write(2Ø,rec=kn9) (itab(k),k=k9,k9+lro-1)
23Ø     continue
        nrec=krec-1
        close(2Ø)
        return
24      if(b3(i3).ge.Ø.)goto5
        i3=i3+2
        goto 23
55      iic=i3
        ic=1
        goto56
5       ic=i3
56      i=1
        x(i)=b3(i3)
        y(i)=b4(i3)
        xi=x(i)
        yi=y(i)
7       b3(i3)=-1.
        b4(i3)=-1.
        if(mod(i3,2).eq.Ø)i3=i3-2
        i=i+1
        if(i.le.lro)goto8
        l=l+lro
        k1=k1+2
        write(2Ø,rec=krec) (x(k),k=1,lro)
        krec=krec+1
        write(2Ø,rec=krec) (y(k),k=1,lro)
        krec=krec+1
        do 377 k=1,lro
            x(k)=Ø
            y(k)=Ø
377     continue
        i=1
8       x(i)=b3(i3+1)
        y(i)=b4(i3+1)
        b3(i3+1)=-1.
        b4(i3+1)=-1.
```

```
      if(xi.eq.x(i).and.yi.eq.y(i))goto88
      i3=ic
12    if(b3(i3).eq.30000.)goto14
      if(x(i).eq.b3(i3).and.y(i).eq.b4(i3)) goto7
      i3=i3+1
      goto 12
14    if(nop.eq.'si') goto88
      goto17
88    l=l+i
      if(l.lt.nn)goto17
      m=m+1
c************ messaggio relativo al numero d'ordine dell'isofota
c************ inseguita e al numero dei punti che la compongono
      write(*,200) m,l
200   format(/' Curva n.',i4,2x,'di',i7,2x,'punti'/)
      write(20,rec=krec) (x(k),k=1,lro)
      krec=krec+1
      write(20,rec=krec) (y(k),k=1,lro)
      krec=krec+1
      icu=(m-1)*3+1
      itab(icu)=1
      itab(icu+1)=krn
      itab(icu+2)=krec-1
17    if(kk.eq.0)goto16
      goto20
      end
```

```
c*************************************************************************
c***                                                                   ***
c***   PROGRAMMA DISCU.FOR. VISUALIZZA E NUMERA SU PIP-1Ø24 LE CURVE   ***
c***   RICAVATE ED ARCHIVIATE CON IL PROGRAMMA ISOF.FOR.               ***
c***                                                                   ***
c***   Rev. 1.Ø - Dicembre 1987                                        ***
c***                                                                   ***
c***   Autore : FANTINI Enrico - I.E.I.  C.N.R.  Pisa                  ***
c***                                                                   ***
c*************************************************************************
c
      program discu
      implicit integer*2 (i-n)
      dimension x(32ØØØ),y(32ØØØ),itab(255Ø),ntab(3,85Ø)
      character nop*2,nfile*12,num*1,ok*1
      integer*2 buffer(512),soglia
      equivalence (itab(1),ntab(1,1))
      lro=256
1     call cls
      write(*,2)
2     format(/3x,73('*'),/3x,'***',67x,'***',/3x,'***   ',
     -'PROGRAMMA DISCU.FOR. VISUALIZZA E NUMERA SU PIP-1Ø24 LE CURVE',
     -3x,'***',/3x,'***   ',
     -'RICAVATE ED ARCHIVIATE CON IL PROGRAMMA -ISOF-.              ',
     -3x,'***',/3x,'***',67x,'***',/3x,73('*'))
5ØØ   nfile='            '
      write(*,51Ø)
51Ø   format(///' Quale e` il nome del file di archiviazione',/,
     -' delle curve (max. 12 caratteri)  (Return=FINE) ? : '$)
      read(*,'(a)') nfile
      if(nfile.eq.'            ') goto 7777
      open(2Ø,file=nfile,access='direct',status='old',recl=lro*4,
     -iostat=ios)
      if(ios.ne.Ø) then
          call bel
          write(*,5Ø1) ios,nfile
5Ø1       format(///' E` stato riscontrato un errore (num. ',i6,')',/,
     -    ' nella apertura del file -',a,'-',//,
     -    ' -----  PREMERE UN TASTO PER CONTINUARE  -----'$)
          call letcar(ris)
          close(2Ø)
          goto 5ØØ
      end if
      do 99 k=1,5
          k1=lro*(k-1)+1
          read(2Ø,rec=k) (itab(j),j=k1,k1+lro-1)
99    continue
      k=1
      icc=Ø
2Ø    if(itab(k).eq.Ø) goto 21
      k=k+3
      icc=icc+1
      if(k.le.2547) goto 2Ø
21    write(*,22)
22    format(//' Quadrante di uscita (Ø-3) [Ø] ? : '$)
```

```
      read(*,222) iquad
222   format(i1)
      if(iquad.lt.0.or.iquad.gt.3) then
          call bel
          goto 21
      end if
      call displ(iquad)
23    write(*,24)
24    format(/' o.k. (s/n) [s] ? : '$)
      read(*,'(a)') ok
      if(ok.eq.' ') ok='s'
      if(ok.eq.'n'.or.ok.eq.'N') goto 21
      if(ok.ne.'s'.and.ok.ne.'S') then
          call bel
          goto 23
      end if
724    write(*,25)
25    format(/' Cancello schermo (s/n) [s] ? :'$)
      read(*,'(a)') ok
      if(ok.eq.' ') ok='s'
      if(ok.eq.'n'.or.ok.eq.'N') goto 28
      if(ok.eq.'s'.or.ok.eq.'S') then
26        write(*,27)
27        format(/' Valore di cancellazione (0-255) [0] ? : '$)
          read(*,227) ival
227       format(i3)
          if(ival.lt.0.or.ival.gt.255) goto 26
          call set(ival)
      else
          call bel
          goto 724
      end if
28    write(*,29)
29    format(/' Valore di tracciatura (0-255) [0] ? : '$)
      read(*,227) ival
      if(ival.lt.0.or.ival.gt.255) then
          call bel
          goto 28
      end if
30    write(*,40) icc,icc
40    format(/' #### Sono presenti ',i4,' curve ####',/,
     -' #### Quante ne tratto ? (da, a) [1,',i4,'] (9999=Fine) :_ '$)
      read(*,444) ici,icf
444   format(2i4)
      if(ici.eq.0) ici=1
      if(icf.eq.0) icf=icc
      if(ici.eq.9999.or.icf.eq.9999) then
          close(20)
630       write(*,631)
631   format(/' Copio su stampante le curve disegnate (s/n) [n]: '$)
          read(*,'(a)') ok
          if(ok.eq.' ') ok='n'
          if(ok.eq.'s'.or.ok.eq.'S') then
              soglia=ival-1
              inter=2
```

```
                call fp_tp(soglia,inter,0,511,1,0,511,1,buffer,nerr)

            else
                if(ok.ne.'n'.and.ok.ne.'N') goto 630
            endif
            goto 1
        end if
        if(ici.lt.0.or.icf.lt.0.or.ici.gt.icf.or.icf.gt.icc) then
            call bel
            goto 30
        end if
41      write(*,42)
42      format(/' Numero le curve (s/n) [s] ? : '$)
        read(*,'(a)') num
        if(num.eq.' ') num='s'
        if(num.ne.'s'.and.num.ne.'S'.and.num.ne.'n'.and.num.ne.'N') then
            call bel
            goto 41
        end if
        do 1000 k=ici,icf
        j=1
        do 900 n=ntab(2,k),ntab(3,k),2
            k1=lro*(j-1)+1
            k2=k1+lro-1
            read(20,rec=n) (x(i3),i3=k1,k2)
            nn=n+1
            read(20,rec=nn) (y(i3),i3=k1,k2)
            j=j+1
900     continue
        n9=ntab(1,k)
        do 800 l=1,n9-1,2
        ixi=x(l)+0.5
        ixf=x(l+1)+0.5
        iyi=y(l)+0.5
        iyf=y(l+1)+0.5
800     call line(ixi,iyi,ixf,iyf,ival)
        if(num.eq.'s'.or.num.eq.'S') then
            call numera(x(1),y(1),k,ival)
        end if
1000    continue
        goto 30
7777    end
c
c       ROUTINE PER NUMERARE LE CURVE VISUALIZZATE
c
c       x,y - coordinate primo punto curva
c       k   - numero della curva
c       ival- valore di scrittura numero
c
        subroutine numera(x,y,k,ival)
        implicit integer*2 (i-n)
        integer*1 ia(3)
        character ca*1(3)
        equivalence (ia(1),ca(1))
        j=k
```

```
      ia(1)=j/100
      j=j-(ia(1)*100)
      ia(1)=ia(1)+48
      ia(2)=j/10
      ia(3)=j-(ia(2)*10)+48
      ia(2)=ia(2)+48
      nx=x+0.5-27.
      ny=y+0.5-3.
      do 10 j=1,3
          if(ia(j).gt.48.or.j.eq.3) then
              call text(nx,ny,ca(j),1,1,0,ival)
          end if
          nx=nx+9
10    continue
      return
      end
```

```
C*****************************************************************
C***                                                           ***
C***   PROGRAMMA CARPE.FOR DETERMINA AREA E PERIMETRO          ***
C***   DELLE CURVE RILEVATE CON IL PROGRAMMA -ISOF-            ***
C***                                                           ***
C***   Rev. 1.0 - Dicembre 1987                                ***
C***                                                           ***
C***   Autore : FANTINI Enrico - I.E.I.  C.N.R.  Pisa          ***
C***                                                           ***
C*****************************************************************
      PROGRAM CARPE
      IMPLICIT INTEGER*2 (I-N)
      CHARACTER NFILE*12,TS*1,DEV*3,RIS*1
      DIMENSION X(40000),Y(40000)
      DIMENSION TAB3(850,2)
      INTEGER*2 TAB1(2550),ITAB1(3,850)
      EQUIVALENCE (TAB1(1),ITAB1(1,1))
4     NPR=6
      LRO=256
      NOK=0
      CALL CLS
      WRITE(*,10)
10    FORMAT(12X,56('*'),/,12X,'***',50X,'***',/,
     -12X,'***  PROGRAMMA CARPE.FOR DETERMINA AREA E PERIMETRO  ***',/,
     -12X,'***  DELLE  CURVE  RICAVATE  ED  ARCHIVIATE CON  IL  ***',/,
     -12X,'***  PROGRAMMA -ISOF- ,  L` USCITA  CONSISTE IN UNA  ***',/,
     -12X,'***  TABELLA CHE  RIPORTA IL  NUMERO DI  CURVA  E I  ***',/,
     -12X,'***  DATI CHE LA RIGUARDANO.                         ***',/,
     -12X,'***  TALE TABELLA PUO` ESSERE INVIATA SU VIDEO O SU  ***',/,
     -12X,'***  STAMPANTE.                                      ***',/,
     -12X,56('*'))
500   WRITE(*,2)
2     FORMAT(///' Quale e` il nome del file di archiviazione',/,
     -' delle curve (max. 12 caratteri)  (Return=FINE) ? : '$)
      READ(*,'(A)') NFILE
      IF(NFILE.EQ.'            ') GOTO 7777
      OPEN(10,FILE=NFILE,ACCESS='DIRECT',STATUS='OLD',RECL=LRO*4,
     -IOSTAT=IOS)
      IF(IOS.NE.0) THEN
          CALL BEL
          WRITE(*,501) IOS,NFILE
501       FORMAT(///' E` stato riscontrato un errore (num. ',i6,')',/,
     -     ' nella apertura del file -',a,'-',//,
     -     ' -----  PREMERE UN TASTO PER CONTINUARE  -----'$)
          CALL LETCAR(RIS)
          CLOSE(10)
          GOTO 500
      END IF
      DO 3 K=1,5
          K1=LRO*(K-1)+1
          READ(10,REC=K) (TAB1(J),J=K1,K1+LRO-1)
3     CONTINUE
      K=1
      ICC=0
20    IF(TAB1(K).EQ.0) GOTO 29
```

```
      K=K+3
      ICC=ICC+1
      IUR=TAB1(K-1)
      IF(K.LE.2547) GOTO 20
29    DO 78 K=1,850
          DO 78 J=1,2
              TAB3(K,J)=0.
78    CONTINUE
      DO 1000 K=1,ICC
      J=1
      DO 900 N=ITAB1(2,K),ITAB1(3,K),2
          K1=LRO*(J-1)+1
          K2=K1+LRO-1
          READ(10,REC=N) (X(I3),I3=K1,K2)
          NN=N+1
          READ(10,REC=NN) (Y(I3),I3=K1,K2)
          J=J+1
900   CONTINUE
      N9=ITAB1(1,K)
      IF(X(N9).EQ.X(1).AND.Y(N9).EQ.Y(1)) GOTO 960
      N9=N9+1
      X(N9)=X(1)
      Y(N9)=Y(1)
960   CALL ARPE(X,Y,N9,A,P)
      TAB3(K,1)=A
      TAB3(K,2)=P
1000  CONTINUE
      CLOSE (10)
30    WRITE(*,40) ICC,ICC
40    FORMAT(/' #### Sono presenti ',I4,' curve ####',/,
     -' #### Quante ne tratto ? (da,a) [1,',I4,'] (9999=Fine) :_ '$)
      READ(*,444) ICI,ICF
444   FORMAT(2I4)
      IF(ICI.EQ.0) ICI=1
      IF(ICF.EQ.0) ICF=ICC
      IF(ICI.EQ.9999.OR.ICF.EQ.9999) GOTO 4
      IF(ICI.LT.0.OR.ICF.LT.0.OR.ICI.GT.ICF.OR.ICF.GT.ICC) THEN
          CALL BEL
          GOTO 30
      END IF
2001  WRITE(*,2000)
2000  FORMAT(/' ###  Uscita (S=Stampante  V=Video) [V] ? :_ '$)
      DEV='CON'
      LU=7
      READ(*,2003) TS
2003  FORMAT(A1)
      IF(TS.EQ.' ') TS='V'
      IF(TS.EQ.'S'.OR.TS.EQ.'s') THEN
          DEV='PRN'
      ELSE
          IF(TS.NE.'V'.AND.TS.NE.'v') THEN
              CALL BEL
              GOTO 2001
          END IF
      END IF
```

```
      OPEN(LU,FILE=DEV)
      IF(DEV.EQ.'CON')  THEN
          CALL CLS
      ELSE
          WRITE(LU,2004)
2004      FORMAT(1H1)
      ENDIF
      IF(DEV.EQ.'PRN') THEN
          WRITE(LU,2101)
2101      FORMAT(19X,'* TABELLA DI CLASSIFICAZIONE DELLE CURVE *',//)
      ENDIF
      WRITE(LU,2005)
2005  FORMAT(19X,42('-'))
      WRITE(LU,2006)
2006  FORMAT(19X,'I',10X,'I',2(14X,'I'))
      WRITE(LU,2007)
2007  FORMAT(19X,'I  CURVA   I',2X,'SUPERFICIE',2X,'I  PERIMETRO   I')
      WRITE(LU,2006)
      WRITE(LU,2005)
      WRITE(LU,2006)
      NRIG=0
3114  DO 3000 K=ICI,ICF
      IF(TAB3(K,1).LT.0.AND.TAB3(K,2).LT.0) GOTO 3000
      NOK=K
      WRITE(LU,3001) NOK,TAB3(K,1),TAB3(K,2)

3001  FORMAT(19X,'I    ',I4,'    I ',F12.4,' I ',F12.4,' I')
      IF(DEV.EQ.'PRN') WRITE(LU,2006)
      NRIG=NRIG+1
      IF(DEV.EQ.'CON'.AND.NRIG.GT.9) THEN
          WRITE(LU,2005)
          CALL BEL
          WRITE(LU,1111)
1111      FORMAT(//' Premere un tasto per continuare ')
          CALL LETCAR(RIS)
          CALL CLS
          WRITE(LU,2005)
          WRITE(LU,2006)
          WRITE(LU,2007)
          WRITE(LU,2006)
          WRITE(LU,2005)
          WRITE(LU,2006)
          NRIG=0
      END IF
3000  CONTINUE
3113  WRITE(LU,2005)
      CLOSE(LU)
      NOK=0
      GOTO 30
7777  END
C
C
C     ######  C A L C O L O   A R E A   E   P E R I M E T R O
C
C
```

```
      SUBROUTINE ARPE(X,Y,M,A,P)
C**** X=VETTORE DELLE -X-
C**** Y=VETTORE DELLE -Y-
C**** M=NUMERO DI PUNTI DELLA CURVA
C**** A=AREA
C**** P=PERIMETRO
      DIMENSION X(1),Y(1)
      INTEGER*2 M
      SA=Ø.
      SP=Ø.
22    DO 3 I=1,M-1
      YY=Y(I+1)-Y(I)
      XX=X(I+1)-X(I)
      SA=SA+(Y(I+1)+Y(I))*XX*Ø.5
      SP=SP+SQRT(YY*YY+XX*XX)
3     CONTINUE
      A=ABS(SA)
      P=SP
      RETURN
      END
```