

IL PROBLEMA DELLE LINEE NASCOSTE

Seconda parte

Nel numero scorso abbiamo iniziato la trattazione del problema delle "hidden lines".

Come ormai noto a chi segue questa rubrica, il problema si presenta nella vista prospettica di una figura solida e consiste nella individuazione delle cosiddette linee nascoste, che sono quelle non visibili dall'osservatore in quanto coperte dagli altri elementi della figura.

Abbiamo introdotto l'argomento cercando di spiegarne le difficoltà, abbiamo proposto un metodo (metodo dei massimi e dei minimi con test punto per punto) per l'individuazione delle linee nascoste nel processo di visualizzazione, abbiamo mostrato alcuni esempi.

Continuando ed approfondendo la trattazione dell'argomento presenteremo due ulteriori metodi: il metodo per segmenti e il metodo per sovrapposizione.

Anche i metodi per la individuazione delle linee nascoste che presentiamo ora, come quelli presentati nello scorso numero lavorano nello spazio immagine. Ovvero il test per la individuazione delle linee nascoste viene fatto direttamente sul piano schermo sui dati bidimensionali che si ottengono alla fine delle routine di traduzione dai dati spaziali tridimensionali.

L'esempio realizzato per illustrare il pri-

mo dei due metodi presentati, che si può utilizzare per la visualizzazione di funzioni del tipo $Y = Y(X,Z)$, lavora con due programmi.

Il primo, nel quale va immessa la funzione, gli intervalli e la definizione di calcolo, i dati della prospettiva, crea un file contenente i dati schermo. Il secondo programma legge il file, e qualsiasi funzione esso rappresenti, visualizza la relativa figura eliminando le linee nascoste.

Il secondo metodo lo abbiamo definito "per sovrapposizione".

È un metodo elementare ed intuitivo, ma avendolo visto recentemente in importanti dimostrazioni di grosse case di computer, abbiamo ritenuto opportuno presentarlo. Consiste nel realizzare il disegno partendo "da dietro" e nel capire la figura via via che si viene avanti, coprendo in tal modo automaticamente le parti nascoste.

Per illustrare questo secondo metodo abbiamo realizzato due programmi: il primo è una implementazione del programma skyline presentato nel numero scorso, il secondo visualizza sul monitor APPLE II a colori un diagramma a barre tridimensionale.

Il metodo dei massimi e dei minimi

Il metodo dei massimi e minimi realizzato per segmenti è la logica evoluzione del

metodo per punti.

Con questo metodo per ogni punto $P(X,Y)$ da disegnare viene eseguito un test di visibilità, ovvero viene verificato se il valore Y è interno alla coppia di valori costituita dal più grande Y e il più piccolo Y fino ad allora disegnati per quello stesso valore di X . Una esemplificazione di tale metodo era costituita dal programma FASCIA pubblicato sul numero scorso.

La visualizzazione di un disegno eseguita punto per punto è estremamente lenta ed è quindi logico il tentativo di realizzare una procedura per segmenti, basata anch'essa sul sistema del confronto con il vettore dei massimi e dei minimi.

Tali due vettori $YMAX(X)$ e $YMIN(X)$ rappresentano sul nostro schermo due spezzate. Il problema di verificare se un segmento è nascosto o meno si presenta praticamente come in figura 1. Il segmento $S1$ è esterno e va sicuramente disegnato, conseguentemente va aggiornato il vettore $YMAX(X)$ tra i valori $X11$ e $X12$; il segmento $S2$ è sicuramente interno e non va disegnato; il segmento $S3$ invece interseca il vettore dei massimi. In questo caso va disegnato solo in parte (tra $X33$ e $X32$).

Per disegnarlo in parte o si ricorre alla routine per punti (che abbiamo presentato nel numero scorso), oppure occorre individuare con qualche metodo il punto di in-

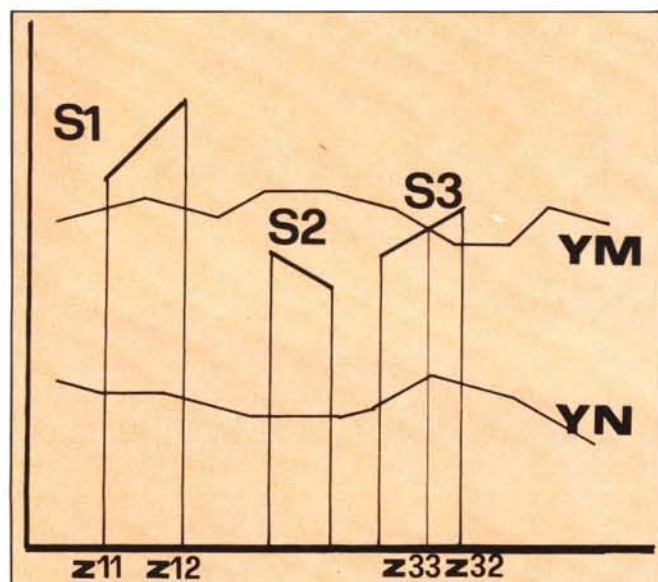


Figura 1 - LINEE NASCOSTE, VISIBILI E SEMINASCOSTE - Nel caso di linee seminascoste va individuato il punto di intersezione tra segmento e vettore dei massimi.

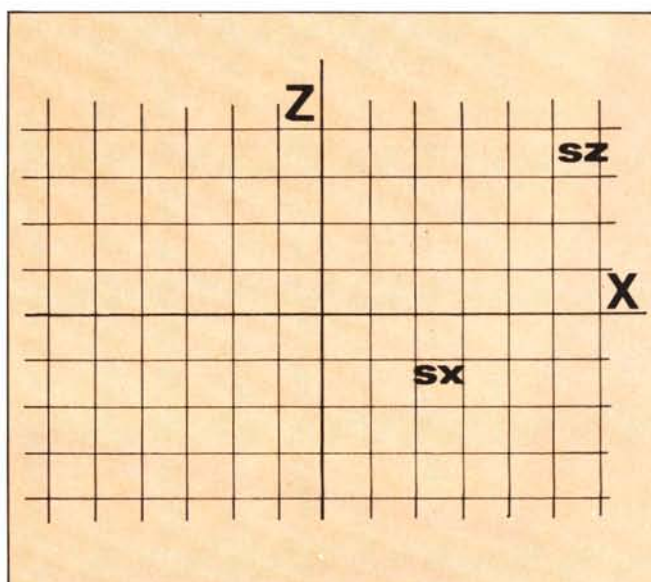


Figura 2 - TRASFORMAZIONE DI PASSO DEI LOOP - Il loop passo 1 è necessario per il caricamento delle matrici, ma può non essere necessario per il calcolo della funzione.

tersezione e considerare solo il segmentino esterno.

Per non appesantire ulteriormente i programmi abbiamo scelto la strada più spiccia. Abbiamo confrontato il punto intermedio del segmento con la fascia, se risulta esterno la disegniamo, se risulta interno non la disegniamo.

Questa approssimazione provoca sicuramente degli errori. Tali errori sono localizzati nelle zone di flesso del disegno (dove comunque le linee si addensano) e sono tanto meno evidenti quanto più è fitto il passo della griglia.

Prima di passare ad illustrare i programmi vogliamo fare una piccola considerazione.

Data la complessità dell'argomento "linee nascoste" abbiamo preferito accostarci poco alla volta, realizzando piccoli passi successivi. Quindi se accettiamo degli errori nei nostri programmi è perché li abbiamo ben localizzati, sappiamo come risolverli, e perché ci ripromettiamo di farlo in seguito.

I programmi

Come al solito prima di vedere i programmi realizzati come esempio dei meto-

di illustrati, descriviamo, soprattutto per i meno esperti, il procedimento di calcolo adottato.

Il cuore del procedimento è costituito dai due loop, I da 1 a NI e J da 1 a NJ. Questi loop sono a passo 1 per poter eseguire il caricamento delle matrici dei valori dei punti schermo: XS(X,Y), YS(X,Z). Per ottenere il passo della X e della Z necessario per il calcolo e gli intervalli di calcolo voluti, dobbiamo trasformare le I e le J in X1 e Z1 (vedi figura 2) con le formule:

$$X1 = I - TX - IX$$

$$Z1 = J - TZ - IZ$$

Ad esempio per tradurre un loop I tra 0 e 100 in loop X1 tra -PIgreco e -PIgreco dovremo porre TX=0.0628832 e IX=-3.14159. Infatti se I=0 allora X1=-3.14, se I=50 allora X1=0, e così via.

Per ogni coppia di valori di calcolo calcoliamo il corrispondente valore Y1, funzione di X1, Z1. Ottenuti così i tre valori spaziali li utilizziamo per la determinazione, mediante i dati caratteristici della nostra prospettiva, in valori schermo, che inseriamo nelle matrici dei valori schermo.

In sostanza utilizziamo una sola coppia di loop per eseguire i calcoli e per caricare le matrici, ed è indispensabile comprendere

il significato di tale coppia di loop per "ambientarci" nello spazio della quale stiamo discutendo.

Per illustrare il metodo dei massimi e dei minimi con test segmento per segmento abbiamo realizzato due programmi.

Il primo elabora e scrive i dati da visualizzare su un file e il secondo legge il file ed esegue sui dati il test di visibilità.

Il programma di scrittura SCRIVIDATI (listato in figura 3) consiste innanzitutto in una serie di istruzioni in cui sono assegnati i valori alle variabili.

Se vogliamo visualizzare altre funzioni, sempre del tipo $Y = Y(X, Z)$, occorrerà modificare quindi tali valori, che sono:

- lunghezza dei loop nelle due direzioni (NI, NJ)
- elementi per la traduzione dei loop (TX, TZ, ecc)
- dati caratteristici dell'output (XS, YS formato, XC, YC centro)
- dati caratteristici della prospettiva (D1, D2)
- costanti varie per la definizione della funzione (C1, C2, ecc.)

La funzione vera e propria va definita all'interno del loop che ne calcola i valori. Tale loop esegue in sequenza:

```

100 REM INIZIALIZZAZIONI
110 REM LUNGHEZZA DEI LOOP NELLE DIREZIONI X,Y
120 NI = 24 : NJ = 24
130 REM CARATTERISTICHE DEL LOOP
140 TX = .1 : TZ = .1 : IX = 0 : IZ = 0
150 D# = CHR# (4)
160 REM DIMENSIONAMENTO MATRICI
170 DIM XS(NI + 1, NJ + 1), YS(NI + 1, NJ + 1)
180 REM DATI SCHERMO
190 XS = 278 : YS = 190 : XC = 140 : YC = 96
200 XM = - 9999 : XN = 9999 : YM = - 9999 : YN = 9999
210 REM DATI PROSPETTIVA
220 D1 = - 20 : D2 = - 35
230 REM COSTANTI VARIE DELLA FUNZIONE
240 C1 = .8 : C2 = 2 : C3 = - 30 : C4 = 25
250 REM LOOP PRINCIPALE
260 TEXT : HOME : PRINT "COORDINATE NELLO SPAZIO"
270 PRINT "I J X1 Z1 Y1" : PRINT : POKE 34,3
280 FOR I = 1 TO NI : FOR J = 1 TO NJ
290 X1 = I * TX + IX : Z1 = J * TZ + IZ
300 Y1 = C1 * COS (2 * X1) - C2 * SIN (3 * Z1) + C3
310 X1 = X1 + C4
320 XS(I, J) = (D1 * X1) / (D2 - Z1)
330 YS(I, J) = (D1 * Y1) / (D2 - Z1)
340 PRINT I, TAB(5)J, TAB(12)X1, TAB(20)Z1, TAB(28)Y1
350 IF XS(I, J) > XM THEN XM = XS(I, J)
360 IF XS(I, J) < XN THEN XN = XS(I, J)
370 IF YS(I, J) > YM THEN YM = YS(I, J)
380 IF YS(I, J) < YN THEN YN = YS(I, J)
390 NEXT J, I
400 REM SCALING
410 DX = XM - XN : DY = YM - YN
420 SX = XS / DX : SY = YS / DY
430 TEXT : HOME : PRINT "COORDINATE VIDEO FORMATTATE"
440 PRINT "I J XS YS" : PRINT : POKE 34,3
450 FOR I = 1 TO NI : FOR J = 1 TO NJ
460 XS(I, J) = (XS(I, J) - XN - DX / 2) * SX + XC
470 YS(I, J) = (YS(I, J) - YN - DY / 2) * SY + YC
480 PRINT I, TAB(5)J, TAB(12)XS(I, J), TAB(26)YS(I, J)
490 NEXT J, I
500 REM VISUALIZZAZIONE DI CONTROLLO
510 HGR2 : HCOLOR= 3 : FOR J = 1 TO NJ : FOR I = 1 TO NI
520 X2 = XS(I, J) : Y2 = YS(I, J)
530 X3 = XS(I + 1, J) : Y3 = YS(I + 1, J)
540 X4 = XS(I, J + 1) : Y4 = YS(I, J + 1)
550 IF I = NI THEN 570
560 HPL0T X2, Y2 TO X3, Y3
570 IF J = NJ THEN 590
580 HPL0T X2, Y2 TO X4, Y4
590 NEXT I, J
600 PRINT D#"OPEN DAT9" : PRINT D#"WRITE DAT9"
610 PRINT NI : PRINT NJ
620 FOR I = 1 TO NI : FOR J = 1 TO NJ
630 X% = XS(I, J) : Y% = YS(I, J)
640 PRINT X% : PRINT Y%
650 NEXT J, I : PRINT D#"CLOSE" : END

```

Figura 3 - PROGRAMMA SCRIVIDATI - Listato - Le variabili necessarie per cambio di passo dei loop sono in riga 140

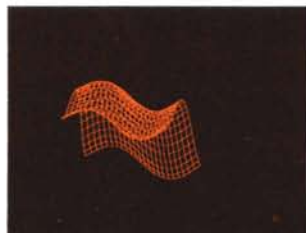


Figura 4 - PROGRAMMA SCRIVIDATI - Output n. 1 - L'output (ancora senza le eliminazioni delle linee nascoste) serve per verificare che i dati che riversiamo nel file siano corretti.

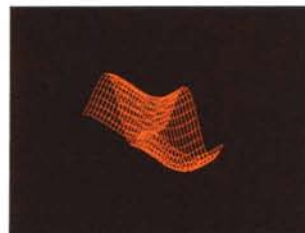


Figura 5 - PROGRAMMA SCRIVIDATI - Output n. 2 - Nelle figure 4 e 5 sono visualizzate due funzioni differenti. Per visualizzarne altre basta cambiare qualche riga del programma.

```

100 REM DIMENS. E INIZIALIZZ. VETTORI M/MAX, M/MIN
110 DIM YM(280), YN(280)
120 FOR I = 0 TO 279 : YM(I) = 0 : YN(I) = 191 : NEXT
130 REM LETTURA DATI GRAFICI
140 D# = CHR# (4) : PRINT D#"OPEN DAT0" : PRINT D#"READ DAT0"
150 INPUT NI : INPUT NJ
160 DIM XS%(NI + 1, NJ + 1), YS%(NI + 1, NJ + 1)
170 FOR I = 1 TO NI : FOR J = 1 TO NJ
180 INPUT XS%(I, J) : INPUT YS%(I, J)
190 NEXT J, I : PRINT D#"CLOSE"
200 REM DISEGNO
210 HGR2 : HCOLOR= 3
220 FOR J = 1 TO NJ : FOR I = 1 TO NI
230 X7 = XS%(I, J) : Y7 = YS%(I, J)
240 X8 = XS%(I + 1, J) : Y8 = YS%(I + 1, J)
250 X9 = XS%(I, J + 1) : Y9 = YS%(I, J + 1)
260 IF I = NI THEN 280
270 X1 = X7 : Y1 = Y7 : X2 = X8 : Y2 = Y8 : GOSUB 310
280 IF J = NJ THEN 300
290 X1 = X7 : Y1 = Y7 : X2 = X9 : Y2 = Y9 : GOSUB 310
300 NEXT I, J : END
310 REM CALCOLO PENDENZA E TEST DENTRO/FUORI FASCIA
320 T = (Y2 - Y1) / (X2 - X1 + .00001)
330 Y3 = (Y1 + Y2) / 2 : X3 = (X1 + X2) / 2
340 IF Y3 > YM(X3) THEN GOSUB 370
350 IF Y3 < YN(X3) THEN GOSUB 420
360 RETURN
370 REM AGGIORNAMENTO YM
380 IF X2 < X1 THEN 400
390 FOR X = X1 TO X2 : YM(X) = (X - X1) * T + Y1 : NEXT X : GOTO 410
400 FOR X = X2 TO X1 : YM(X) = (X - X1) * T + Y1 : NEXT X
410 HPL0T X1, Y1 TO X2, Y2 : RETURN
420 REM AGGIORNAMENTO YN
430 IF X2 < X1 THEN 450
440 FOR X = X1 TO X2 : YN(X) = (X - X1) * T + Y1 : NEXT X : GOTO 460
450 FOR X = X2 TO X1 : YN(X) = (X - X1) * T + Y1 : NEXT X
460 HPL0T X1, Y1 TO X2, Y2 : RETURN

```

Figura 6 - PROGRAMMA LETTURA E VISUALIZZAZIONE - Listato - Si notano le routine di test dentro/fuori fascia e di aggiornamento dei vettori YMAX(X) e YMIN(X) a seconda che il segmento sia superiore od inferiore al disegno fino ad allora tracciato.

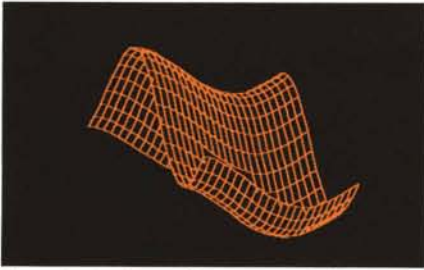


Figura 7 e 8 - Output del programma di lettura e visualizzazione - Sono evidenti le imperfezioni, citate nel testo, relativamente alle linee seminascoste. Queste si verificano nelle zone di margine e di flesso della curva spaziale.

- il calcolo dei punti spaziali della funzione
- il calcolo dei punti schermo secondo la prospettiva voluta
- la stampa dei dati per il loro controllo
- il calcolo dei valori necessari alle operazioni di scaling.

Terminato il loop principale vengono determinati i fattori di scala e successivamente vengono calcolati e stampati i valori dei punti schermo formattati e poi vengono inseriti nelle matrici XS (I,J) e YS (I,J).

Viene poi eseguita una visualizzazione di controllo della curva con il metodo di collegare ciascun punto con i quattro punti contigui. Tale visualizzazione consente di vedere la nostra curva "prima della cura" cioè con tutte le linee nascoste; per "dopo la cura" intendiamo, e lo vedremo, la figura senza le linee nascoste.

A questo punto se tutto è andato bene, viene caricato il file in cui sono inseriti:

- NI,NJ lunghezza dei loop
- XS(I,J) e YX(I,J) matrici contenenti i valori schermo.

Tali valori sono ridotti in numeri interi in quanto la precisione richiesta per l'output è compatibile con la precisione possibile con i numeri interi. In tal modo si risparmierà parecchia memoria e tempo.

Il programma è molto ordinato e segue lo sviluppo logico ora descritto senza ricorrere a subroutine. Nel listato è inserita una funzione del tipo

$$Y = C1 * \sin(C2 * X) + C3 * \cos(C4 * Z) + C5$$

L'output è presentato in figura 4. In figura 5 è presentata un'altra funzione ottenuta con lo stesso programma, modificando, nel modo descritto, alcune istruzioni.

Il programma di lettura e visualizzazione (listato in figura 6) è generalizzato, ovvero funziona con qualunque file realizzato con il programma precedente.

Dopo aver inizializzato il vettore YMAX e YMIN, viene letto il file e i dati relativi sono caricati sulle matrici XS₀(I,J) e YS₀(I,J). Il loro dimensionamento può essere fatto solo dopo aver conosciuto i valori NI e NJ.

Si comincia subito a disegnare.

Occorre partire dalle linee più vicine all'osservatore e per ogni punto vengono realizzati due segmenti utilizzando i valori X7, Y7 per il punto in esame, X8, Y8 per il punto corrispondente sulla colonna contigua e X9, Y9 per il punto corrispondente sulla riga contigua.

Per ciascuno dei due segmenti da visualizzare P7, P8 e P9 viene eseguito il test di visibilità, prendendo in esame il punto intermedio del segmento (righe 330 ÷ 350).

Se il segmento è visibile si aggiorna, nell'intervallo tra X7 e X8 (oppure X7 e X9), il valore del vettore YMAX(X) (oppure YMIN(X)), attribuendogli il valore corrispondente delle Y del segmento. La routine relativa al caso YMAX è in riga 370 e successive. Per fare ciò occorre eseguire il loop in senso crescente e occorre disporre del valore T, pendenza del segmento.

Alcune considerazioni finali:

- il metodo di testare il punto medio del segmento è un po' grossolano. Cercheremo successivamente di realizzare un metodo più affinato;

- per risolvere i problemi ai margini, dove ogni punto è collegato con meno di quattro punti contigui, si sono poste delle uscite dai due loop principali (righe 260 e 280);

```

100 HOME : VTRAB (21) : HGR : HCOLOR= 3
110 H PLOT 0,159 TO 279,159
120 FOR K = 0 TO 80 STEP 4
130 C = INT ( RND (1) * 6 + 1)
140 HCOLOR= C: IF C = 4 THEN 130
150 X1 = RND (1) * 250
160 X2 = X1 + RND (1) * 20 + 10
170 Y = RND (1) * K + K
180 FOR X = X1 TO X2
190 PRINT "GRATTACIELI";
200 H PLOT X, Y TO X, 158: NEXT X, K

```

Figura 9 - PROGRAMMA GRATTACIELI - Listato - È un programma cortissimo, le righe potevano essere ridotte a tre.

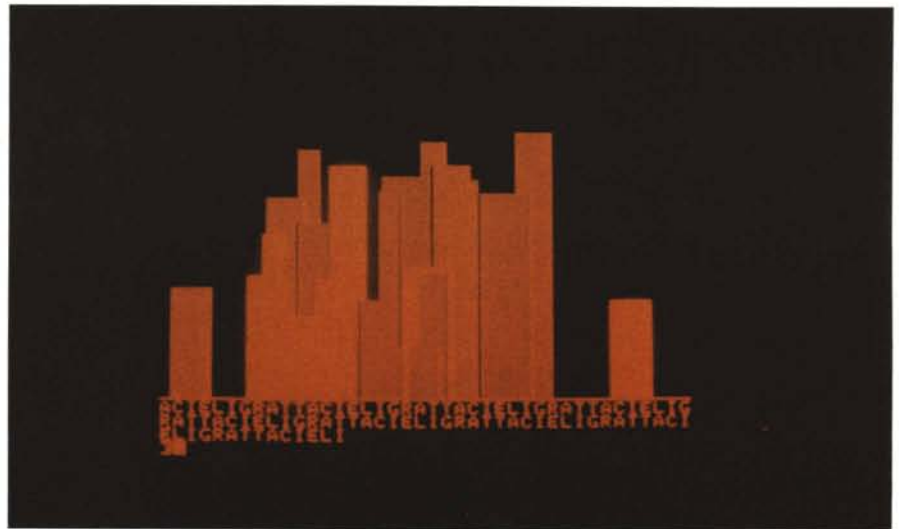


Figura 10 - PROGRAMMA GRATTACIELI - Output - Altezza, larghezza, colori sono random. I grattacieli si sovrappongono via via.

- per permettere, anche nel caso dei punti di confine, il calcolo dei punti di lavoro P7, P8, P9, è stato necessario sopradimensionare di una unità la matrice dei dati. Abbiamo in sostanza lavorato con punti fuori margine fittizi, che tanto poi non abbiamo visualizzato;

- per "far venire meglio" il disegno, abbiamo utilizzato due fattori di scala differenti SX, SY nelle due direzioni. Questo da un punto di vista formale non è corretto, ma ci ha permesso di "riempire" meglio il monitor.

Gli output delle due funzioni senza le linee nascoste sono in figura 7 e figura 8.

Il metodo a sovrapposizione

Tale metodo consiste nel disegnare le figure una sopra all'altra partendo dal fondo, ovvero dalla più lontana dall'osservatore fino alla più vicina.

Può essere utilizzato solo quando si possono compiere le figure e cioè solo su monitor, possibilmente a colori.

Abbiamo realizzato due programmi.

Il primo è semplicissimo e non è altro che una versione modificata del programma SKYLINE pubblicato sul numero scorso di MCmicrocomputer, e si chiama GRATTACIELI.

Il listato è in figura 9. Si nota subito l'assenza delle routine di test. I grattacieli, le cui dimensioni e i cui colori sono casuali, vengono disegnati a partire da quello in fondo e via via sovrapposti.

Viene usata la pagina HGR e sulle quattro righe di testo viene fatta scorrere la parola "GRATTACIELI" via via che il disegno si va formando.

L'output è in figura 10.

Il programma diagramma a barre tridimensionale

Con un diagramma a barre tridimensionale si possono visualizzare grandezze fun-

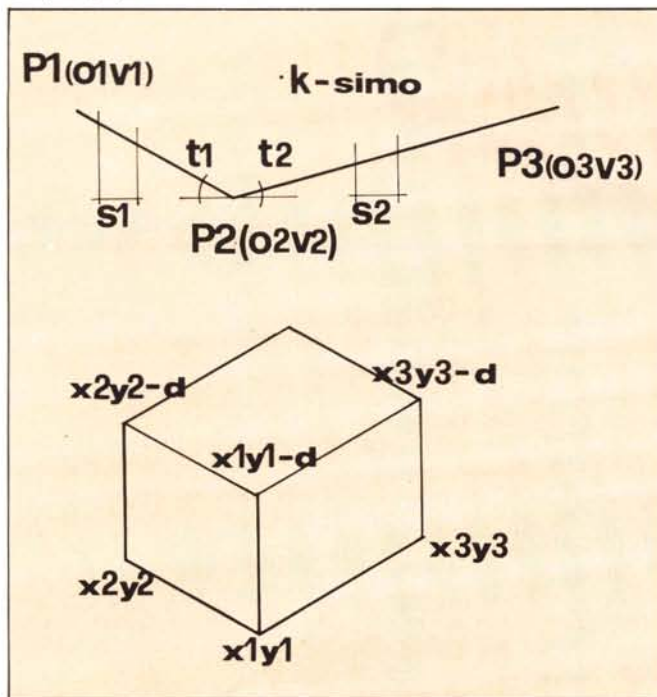
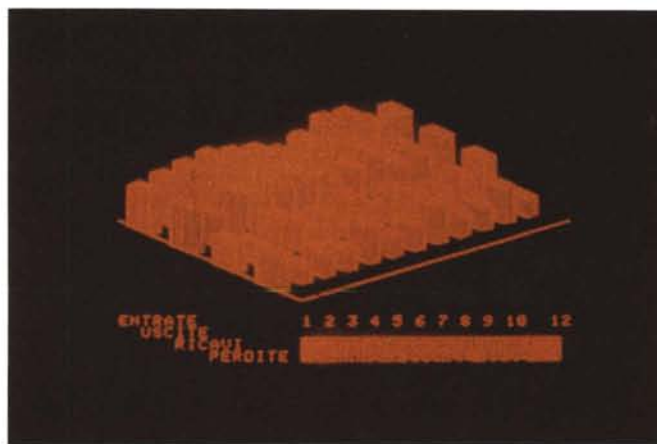


Figura 11 - SCHIZZO PROGRAMMA DIAGRAMMA A BARRE TRIDIMENSIONALE - La posizione sullo schermo dell'elemento K-simo dipende dai due loop I, J e dai valori caratteristici t1, t2, S1, S2. Per ogni elemento K-simo si costruiscono le tre facce del cubetto.



zioni di due variabili. Per rendere più leggibile il diagramma abbiamo utilizzato i colori del monitor APPLE II. Come noto, però, tale monitor non permette la gestione dei pixel a colori indipendentemente uno dagli altri (per far questo occorrerebbe una memoria video tripla), e quindi i risultati "cromatici", a meno di tenere conto delle limitazioni in sede di programma, sono imprevedibili.

Il programma non presenta incognite se utilizzato su un computer con gestione del colore completa.

Esaminiamo contemporaneamente lo schizzo di figura 11 e il listato di figura 12 per comprendere come è stato realizzato il diagramma.

Vanno impostati i valori delle coordinate dei punti P1, P2, P3 (riga 110) e quindi calcolati i valori t1 e t2 angoli sulla orizzontale necessari per determinare la porzione di piano sulla quale riportare le barre.

Vengono poi individuate le grandezze S1 e S2, formato del singolo cubetto, come funzioni del numero di dati da visualizzare.

Nel nostro caso i dati sono 48, 12 lungo P2P3 e 4 (corrispondenti al valore G1 di riga 140, valore che può esser cambiato) lungo P2P1.

I dati vanno caricati con istruzioni di READ e DATA o con istruzioni di INPUT. Noi li abbiamo messi RANDOM.

Va realizzata la scritta sulle quattro righe di testo permesse dalla pagina HGR dell'APPLE II (righe 250-360) e poi si passa al disegno delle barre.

L'elemento K-simo interno ai due loop

```

100 REM INIZIALIZZAZIONI VARIE
110 O1 = 0:V1 = 95:O2 = 110:V2 = 150:O3 = 278:V3 = 94
120 T1 = (V2 - V1) / (O2 - O1)
130 T2 = (V2 - V3) / (O3 - O2)
140 G1 = 4
150 P1 = INT ((O2 - O1) / (2 * G1))
160 P2 = INT ((O3 - O2) / 24)
170 REM CARICAMENTO DEI DATI
180 DIM DD%(G1,12)
190 FOR I = 1 TO G1: FOR J = 1 TO 12:
200 DD%(I,J) = RND (1) * 15 + I ^ 2 + 10
210 NEXT J,I
220 REM CARICAMENTO DEI COLORI
230 FOR I = 1 TO G1: READ CL%(I): NEXT
240 DATA 1,2,5,6
250 REM SQUADRATURA E SCRITTE
260 HGR : HCOLOR= 3: HOME
270 VTAB (21): PRINT "ENTRATE";
280 VTAB (21): HTAB (16): PRINT " 1 2 3 4 5 6 7 8 9 10 12";
290 VTAB (22): HTAB (3): PRINT "USCITE";
300 VTAB (23): HTAB (6): PRINT "RICAVI";
310 VTAB (23): HTAB (17): INVERSE
320 PRINT " BILANCIO AZIENDALE ": NORMAL
330 VTAB (24): HTAB (9): PRINT "PERDITE";
340 VTAB (24): HTAB (17): INVERSE
350 PRINT " ANNO FISCALE 1982 ": NORMAL
360 HPLOT O1,V1 TO O2,V2 TO O3,V3
370 REM LOOP PRINCIPALI
380 FOR I = G1 TO 1 STEP - 1
390 FOR J = 12 TO 1 STEP - 1
400 D = DD%(I,J): HCOLOR= CL%(I)
410 REM ELEMENTO K-SIMO
420 I1 = 2 * P1 * (I - 1 / 2)
430 J1 = 2 * P2 * (J - 1)
440 X1 = O2 - I1 + J1
450 Y1 = V2 - I1 * T1 - J1 * T2
460 GOSUB 470: NEXT J,I: FOR K = 1 TO 9999: NEXT : END
470 REM SUB DISEGNO CUBETTO
480 X2 = X1 - P1:Y2 = Y1 - P1 * T1
490 X3 = X1 + P2:Y3 = Y1 - P2 * T2
500 REM PARETE SINISTRA
510 FOR X = X2 TO X1:Y1 = Y2 + (X - X2) * T1:Y5 = Y1 - D
520 HPLOT X,Y1 TO X,Y5: NEXT X
530 REM PARETE DESTRA
540 FOR X = X1 TO X3:Y1 = Y1 - (X - X1) * T2:Y5 = Y1 - D
550 HPLOT X,Y1 TO X,Y5: NEXT X
560 REM SPIGOLI
570 HCOLOR= 3
580 HPLOT X2,Y2 TO X2,Y2 - D
590 HPLOT X1,Y1 TO X1,Y1 - D
600 HPLOT X3,Y3 TO X3,Y3 - D
610 HPLOT X1,Y1 TO X2,Y2
620 HPLOT X1,Y1 TO X3,Y3
630 FOR X = X2 TO X1:X1 = X:X5 = X + P2
640 Y1 = Y1 - (X1 - X) * T1 - D:Y5 = Y1 - P2 * T2
650 HPLOT X1,Y1 TO X5,Y5: NEXT X
660 RETURN
    
```

Figura 12 - In alto PROGRAMMA DIAGRAMMA A BARRE TRIDIMENSIONALE - Listato - Il programma è facilmente adattabile ad altre quantità e tipi di dati. Andrà in tal caso composta la parte testo che occupa le quattro righe sotto il disegno.

Figura 13 - A sinistra PROGRAMMA DIAGRAMMA A BARRE TRIDIMENSIONALE - Output - L'output risulta molto condizionato dalla gestione del monitor a colori dell'APPLE II, che non permette determinati accostamenti di colori e provoca sbavature e cambi di colore (pagina 19 del manuale HARDWARE).

principali (I da 1 a G1 e J da 1 a 12) è individuato geometricamente sul monitor dai valori X1, Y1, funzione di I, J e delle caratteristiche t1,t2,S1,S2 (righe 410-450).

Attorno ai valori X1, Y1 si costruisce geometricamente il cubetto, che ovviamente è alto DD%(I,J) (riga 480-490).

Trovati i punti caratteristici del cubetto (righe 470 e seguenti) si disegnano sul video le tre facce visibili.

Occorre notare:

- che i loop sono rovesciati (passo -1), perché come abbiamo detto, occorre partire dal fondo;

- che abbiamo inserito anche i disegni degli spigoli in un altro colore. Questi spigoli servono per dare forma al cubetto, ma sono quelli che sul monitor a colori dell'APPLE II provocano le sbavature.

L'output è in figura 13.