Chi, vedendo le cose incredibili che riescono a fare alcuni programmi "commerciali" di giochi, non ha esclamato "ma queste cose il mio Apple non le fa!"?

Altri invece avranno tentato di ottenere risultati simili in Basic, magari compilato, senza riuscire nemmeno a realizzare un efficiente controllo di collisione. Anche le normali routine grafiche Applesoft, sebbene utilizzate da un programma in linguaggio macchina, non sono sufficienti ad ottenere gli effetti grafici e dinamici dei giochi "veri". E allora?

Il trucco c'è! Occorre innanzitutto sapere esattamente che tipo di grafica e quale effetto si vuole ottenere, quindi si devono scrivere delle routine grafiche del tutto indipendenti dal normale software Apple e fatte apposta per ottimizzare le qualità richieste dal nostro tipo di gioco: ovvero la velocità o colore o controlli di collisione e calcoli di spostamenti molto efficienti.

Il programma che presentiamo è in realtà una scusa per capire come sia possibile ottenere un certo tipo di grafica rifacendo le routine in linguaggio macchina che ci interessano.

Avete presente la corsa delle moto di TRON, il film della Walt Disney Production? Ciascuna delle due moto, nella sua corsa, si lascia dietro un muro di energia che i concorrenti non devono urtare.

motomuro

di Valter Di Dio

Come si disegna in HGR

Ci sono vari modi per disegnare qualcosa sullo schermo in alta risoluzione e l'uso di uno o l'altro di questi metodi deriva direttamente dall'effetto che si vuole ottenere. Un primo metodo, abbastanza classico è quello della stampa a blocchetti: ovvero la figura, per esempio l'alieno di certi giochi, è composta da un pacchetto di byte che vengono depositati nella mappa di memoria video. Questo tipo di disegno è particolarmente veloce, pensate un po' alla poke del Basic, ma non consente spostamenti inferiori al byte; dato che sull'Apple ogni byte equivale a sette punti, e ogni pacchetto è di otto byte, il minimo passo che il nostro alieno può effettuare è di sette punti in orizzontale e di otto in verticale, è possibile far spostare l'alieno in verticale anche di un solo punto ricalcolando le posizioni di tutti e otto i byte verticali, ma in questo caso si perde molta velocità. Un altro difetto della shape a blocchi è che essendo la dimensione della shape superiore a quella della figura che contiene, l'alieno si porta dietro un bordo nero che arriva fino alla fine del byte. Se il fondo su cui si muove l'alieno è nero non si nota nulla finché non si sovrappongono due figure, ma se esiste un fondale, magari caricato da disco, questo viene via via distrutto dal passaggio delle shape. I programmi di gioco che usano questo tipo di shape, in genere salvano il fondale che la shape distrugge in una zona si memoria e, da qui, lo rimettono a posto appena tolta la shape. Questo naturalmente comporta una perdita di tempo; ma visto che normalmente le shape sono di pochi byte, il rallentamento è abbastanza contenuto e sicuramente inferiore alla ricostruzione di tutto il fondale ad ogni movimento. Tecnica comunque usata da altri programmi. Ultimo problema per l'uso delle shape a blocchetti è il colore. Nell'Apple per questioni di risparmio di memoria la gestione del colore è alquanto "incasinata". Per motivi hardware i punti sono di un certo colore se sono in posizione dispari, di un altro se sono in posizione pari all'interno del byte; non basta! Passando da un byte di ordine pari a uno di ordine dispari (a fianco) i due colori precedenti vengono scambiati. Inoltre, se il byte che contiene detti punti ha il bit 7 settato allora i punti (tutti e sette) passano al colore complementare! E il BIANCO? Semplice: due punti vicini (quindi uno pari e uno dispari) sono sempre bianchi. Se a questo punto avete rinunciato all'uso del colore non possiamo darvi torto. Tornando alle shape a blocchetti è ormai chiaro che spostando una figura di un byte in orizzontale tutti i suoi punti isolati cambieranno colore; per cui per poter usare delle shape a blocchi colorate ci sono solo due soluzioni: o si usano due shape differenti per i byte pari e dispari oppure si definisce un'unica shape bianca e tramite una opportuna maschera, diversa per le posizioni pari e dispari, si spengono i punti non necessari. È il metodo usato normalmente dalla HPLOT dell'Applesoft; infatti se provate a tracciare una riga colorata in una colonna "sbagliata" il byte di maschera non permette la comparsa della riga stessa.

Nonostante tutte queste controindicazioni il metodo delle shape a blocchetti, seppure con qualche variante, resta comunque il più usato.

Altri metodi utilizzano o delle routine di DRAW per punti o l'uso delle SHAPE TABLE (quindi a vettori) e perciò delle routine dell'Applesoft. In genere questi metodi sono particolarmente veloci ma non consentono la creazione di figure particolarmente complesse o rifinite e vengono quindi usati solo per i programmi "spaziali" dove c'è molto nero e poche semplici figure in movimento.

Le shape a blocchi pre-shiftate

Poco fa abbiamo detto che non è possibile far fare ad una shape a blocchi un passo inferiore a sette punti. Pur rimanendo vero ciò niente ci impedisce di costruirci sette shape del nostro alieno, ciascuna disegnata un punto a destra della precedente e, per eseguire lo spostamento di un passo, cambiare la shape 0 con la 1, poi la 1 con la 2 e così via

fino alla settima, dopodiché si riparte con la zero spostata di 1 byte. È proprio questo il metodo che abbiamo scelto per realizzare il nostro programma.

Per sapere quale shape usare ci basta vedere le locazioni orizzontali (X) non come 256 punti (perché non sono 279 lo vedremo in seguito) ma come un certo numero di byte interi più un resto da zero a sette bit. In pratica dobbiamo calcolare X MODULO 7 e plottare la shape RESTO nel punto X DIVISO 7 (divisione intera). Per comodità di programmazione la routine X MOD 7 è stata realizzata a otto bit e quindi il massimo punto orizzontale è 255. In verticale il discorso è più semplice in quanto si usa la stessa shape spostandola alla locazione di memoria sottostante. Se vi sembra facile andate a vedere come l'Apple gestisce la memoria video! Se immaginiamo lo schermo diviso in strisce orizzontali larghe otto righe, ciascun punto della riga dista da quello sottostante esattamente 1024 byte se non usciamo dalla striscia. In caso di sconfinamento dista 128 byte dal corrispondente punto che si trova otto righe più su. Questo è vero per tre fasce di otto delle succitate strisce, dopodiché in caso di sconfinamento lo spostamento è di solo 40 byte dal punto che occupa una posizione simile nella fascia precedente. Se vi è venuto lo stesso desiderio che avete avuto prima per il colore vi capiamo benissimo, ma mentre è possibile rinunciare al colore non si può rinunciare all'asse Y. E allora?

Anche qui ci sono due possibilità. O si usa un programma che calcola l'indirizzo finale in base al valore della Y o si precalcolano tutti i valori iniziali delle righe di schermo e si mettono in una tabella in modo che leggendo l'Y-esimo valore questo corrisponda alla effettiva locazione iniziale (la prima a sinistra) della rispettiva riga Y. È proprio così che abbiamo fatto. Lo svantaggio di questo metodo è che oltre a occupare non poca memoria per le 192 coppie di puntatori della YTABLE non si può disegnare contemporaneamente sulle due pagine grafiche con la stessa tabella sicché se si vuole usare il Flipping di pagina le tabelle devono essere due e due devono essere pure i programmi di PLOT.

Come sempre accade in questi casi una maggiore occupazione di memoria si risolve però in una velocità decisamente elevata: proprio quello che ci serviva per rendere il nostro programma abbastanza reale.

Ultima routine necessaria ad un programma di giochi è il controllo di collisione: si può agevolmente far fare alla routine di plot che prima di plottare un byte effettua l'AND con il contenuto dello schermo e setta eventualmente un flag di collisione.

Per disegnare i bordi del campo e per pulire lo schermo sono state usate le routine di HGR, HCOLOR, HPLOT e HPLOT TO dell'Applesoft soprattutto la HPLOT TO che è decisamente più comoda della nostra per plottare linee continue. Come si passano i valori a queste routine e dove si trovano è indicato nella tabella 1 (a pagina 64).

Altre due routine usate sono la PDL (X) del Monitor che ritorna un valore tra zero e 255 in relazione alla posizione della manopola X, e la WAIT che effettua un ritardo proporzionale al valore dell'Accumulatore e che ci serve per rallentare il gioco altrimenti troppo veloce (sembra incredibile ma abbiamo accelerato tanto le routi-

```
10  IF  PEEK (24576) <  > 32 THEN 100
20  POKE  - 16297,0: REM  HGR
30  POKE  - 16299,0: REM  PAGE 2
40  POKE  - 16304,0: REM  GRAPHIC
50  FOR I = 1 TO 3000: NEXT
60  GOTO 140
100 D$ =  CHR$ (13) +  CHR$ (4)
110  TEXT : HOME : HGR2
120  PRINT D$"BLOAD PIC.LOGO,A$4000"
130  PRINT D$"BLOAD XO.CODE"
140  TEXT : HOME : PRINT TR$:START = 24576
150  INPUT "Vuoi le istruzioni ?";RT$
160  IF RT$ <  > "S" THEN 1000
170  HOME : PRINT
180  PRINT "  Il gioco consiste nel guidare
     due      moto elettroniche che lasciano
      dietro   di se' un muro di energia."
190  PRINT "  Se urtate il bordo del campo o
      uno      dei muri lasciati dalle moto,
     avete     perso."
200  PRINT "  Le moto si guidano con la tast
     iera:    due tasti per ciascun giocator
     e         permettono di voltare a destr
     a o a      sinistra ":
210  PRINT "rispetto alla direzione di     m
     arcia."
220  PRINT "  La velocita' delle moto si con
     trolla   con le paddle: tutte ruotate i
     n senso   orario e' la velocita' minima
     ("
230  PRINT : INPUT " PREMI RETURN ";RT$
240  HOME : PRINT
250  PRINT "  I tasti del giocatore di sinis
     tra      sono -A- per andare a sinistra
     "
260  PRINT " e -S- per andare a destra, piu'
      la      paddle 0 per la velocita'"
270  PRINT
280  PRINT "  Per il giocatore di destra i t
     asti     sono <- per la direzione sinis
     tra e     -> per la destra; la velocit
     a' e'     controllata dalla paddle 1."

290  PRINT
300  PRINT "  E' possibile anche giocare da
     soli     o contro il Computer. Si posso
     no anche  eliminare alcuni o tutti i ru
     mori e     modificare a piacere i limit
     i di       velocita' delle moto."
310  PRINT : INPUT " Sei pronto ? ";RT$
320  IF RT$ <  > "S" THEN 140
1000  REM  MENU' PRINCIPALE
1005 TR$ = "********************************
     ************
          ********        MC MICROCOMPUTER
           ********
            ********************************
     ************
1020  TEXT : HOME : PRINT : PRINT TR$
1030  PRINT "USA PDL(0) E <RETURN> PER SCEGL
     IERE : ": PRINT
1040  PRINT "      GIOCO NORMALE (DUE CONCOR
     RENTI)": PRINT
1050  PRINT "      ALLENAMENTO (UNA SOLA MOT
     O)": PRINT
1060  PRINT "      GIOCO CONTRO IL COMPUTER"
     : PRINT
1065  PRINT "      FINE": PRINT
1070 X% =  PDL (0) / 36 + 10: VTAB X%: HTAB
     3
1080  INVERSE : PRINT "-->";
1090  IF  PEEK ( - 16384) = 141 THEN  GET RT
     $: GOTO 1120
1100  FOR D = 1 TO 300: NEXT : VTAB X%: HTAB
     3
1110  NORMAL : PRINT "   ": GOTO 1070
1120 X% = (X% - 8) / 2: NORMAL
1130  ON X% GOTO 2000,2400,3000
1140  TEXT : NORMAL : HOME : END
1500  HOME : PRINT : PRINT TR$
1510 SP = 0
1520  PRINT : INPUT " CON OSTACOLI ? (S/N) "
     ;RT$
1530  IF RT$ = "N" THEN 1600
1540  HGR : HCOLOR= 3: POKE  - 16302,0:SP =
     3
1550  FOR X = 8 TO 235 STEP 20
1560  FOR Y = 4 TO 170 STEP 20
1570  IF  RND (1) > RH THEN  HPLOT X,Y TO X +
     20,Y
1580  IF  RND (1) > RV THEN  HPLOT X,Y TO X,
     Y + 20
1590  NEXT : NEXT
1600  RETURN
2000  REM  GIOCO A DUE
2010  POKE 24770,234: POKE 24771,234: POKE 2
     4772,234: POKE 24886,5
2020  GOSUB 2600
2030  HOME : PRINT : PRINT TR$
2050  INPUT " GIOCATORE DI SINISTRA SCRIVI I
     L TUO      NOME: ";GS$
2060  PRINT
2070  INPUT " GIOCATORE DI DESTRA SCRIVI IL
     TUO        NOME: ";GD$
2080  PRINT :GD = 0:GS = 0:RH = .7:RV = .8
2090  INPUT " QUANTE SFIDE VOLETE FARE PER O
     GNI        TURNO DI GIOCO? ";NN
2100  GOSUB 1500
2200  HOME
2210  CALL START + SP
2220 X =  PEEK (8)
2230  IF X = 1 THEN GS = GS + 1
2240  IF X = 2 THEN GD = GD + 1
2250 NN = NN - 1
2260  HOME : INVERSE : VTAB 10
2270  HTAB 2: PRINT GS$;: NORMAL : PRINT " P
     UNTI "GS;
2280  INVERSE : VTAB 10
2290  HTAB 22: PRINT GD$;: NORMAL : PRINT "
     PUNTI "GD
2300  IF NN > 0 THEN 2320
2310  VTAB 15: HTAB 1: PRINT " FINE DELLA PA
     RTITA"
2320  VTAB 22: HTAB 1: INPUT "PREMI RETURN";
     RT$
2330  IF NN > 0 THEN 2100
2340  PRINT : INPUT "ANCORA ? (RET=SI) ";RT$

2350  IF RT$ = "N" THEN 1000
2360  HOME : PRINT TR$: GOTO 2080
2400  REM  ALLENAMENTO
2410  POKE 24770,76: POKE 24771,85: POKE 247
     72,96: POKE 24886,8
2420  GOSUB 2600
2430 RH = .5:RV = .7
2440  GOSUB 1500
2480  REM  GIOCO
2500  CALL START + SP
2510  HOME : VTAB 22: HTAB 1: INPUT " ANCORA
      (RETURN=SI) ";RT$
2520  IF RT$ = "" OR RT$ = "S" THEN 2430
2530  GOTO 1000
2600  REM  MODIFICHE
2610  TEXT : HOME : PRINT : PRINT TR$
2620  PRINT " MODIFICHE POSIBILI:": PRINT
2630  PRINT "1)  TOGLIE IL RUMORE DELLE TURB
     INE": PRINT
2640  PRINT "2)  TOGLIE IL RUMORE DEI MOTORI
     ": PRINT
2650  PRINT "3)  TOGLIE IL RUMORE DEL CRASH"
     : PRINT
2660  PRINT "4)  MODIFICA LA VELOCITA' DELLE
      MOTO": PRINT
2665  PRINT "5)  RESET (TUTTO NORMALE)": PRINT

2670  PRINT : INPUT "  QUALE ? ";RT$
2680 X =  VAL (RT$): IF X = 0 THEN  RETURN
2690  ON X GOTO 2750,2760,2770,2800,2850
2700  GOTO 2600
2750  POKE 24891,32: GOTO 2600
2760  POKE 24766,32: POKE 24878,32: GOTO 260
     0
2770  POKE 25401,32: GOTO 2600
2800  PRINT : INPUT "VELOCITA'= (1 - 16 / NO
     RM=12) ";RR
2810  IF RR < 1 OR RR > 16 THEN 2600
2820  POKE 24886,17 - RR: GOTO 2600
2850  REM  RESET DEFAULT
2860  POKE 24886,5: REM  SPEED
2870  POKE 24766,48: REM  MOTOR L
2880  POKE 24878,48: REM  MOTOR R
2890  POKE 24891,48: REM  TURBO
2900  POKE 25401,48: REM  CRASH
2910  GOTO 2600
3000  REM  AUTOGIOCO
3010  POKE 24770,76: POKE 24771,80: POKE 247
     72,99: POKE 24886,5
3020  GOSUB 2600
3040  TEXT : HOME : PRINT : PRINT TR$
3050  PRINT : PRINT " La paddle 1 regola la
     velocita' della   moto guidata dal comp
     uter. Il computer  muove a caso; ma non
      troppo!"
3060  PRINT : PRINT "Premi RETURN per inizia
     re.";: GET RT$
3080 GS$ = "GIOCATORE":GD$ = "COMPUTER"
3100  GOTO 2360
```

Figura 2 - Listato Basic

```
100  REM  Y TABLE HI.LO
150 L = 24912:L1 = 25104
160  FOR D = 0 TO 2
170  FOR K = 0 TO 3
180  FOR H = 0 TO 7
190  POKE L + H + (K * 16) + (D *
     64),K + 32 + H * 4
200  POKE L + 8 + H + (K * 16) +
     (D * 64),K + 32 + H * 4
210  POKE L1 + H + K * 16 + D * 6
     4,D * 40
220  POKE L1 + 8 + H + D * 64 + K
      * 16,128 + D * 40
240  NEXT
250  NEXT
260  NEXT
```

Figura 1 - Programma che genera la YTABH e la YTABL necessarie alle routine di Hplot.

```
6000- 20 E2 F3   JSR  $F3E2
6003- A2 07      LDX  ##07
6005- 20 EC F6   JSR  $F6EC
6008- A9 00      LDA  ##00
600A- AA         TAX
600B- A8         TAY
600C- 20 57 F4   JSR  $F457
600F- A9 FF      LDA  ##FF
6011- A0 00      LDY  ##00
6013- A2 00      LDX  ##00
6015- 20 3A F5   JSR  $F53A
6018- A9 FF      LDA  ##FF
601A- A2 00      LDX  ##00
601C- A0 BF      LDY  ##BF
601E- 20 3A F5   JSR  $F53A
6021- A9 00      LDA  ##00
6023- AA         TAX
6024- A0 BF      LDY  ##BF
6026- 20 3A F5   JSR  $F53A
6029- A9 00      LDA  ##00
602B- AA         TAX
602C- A8         TAY
602D- 20 3A F5   JSR  $F53A
6030- A9 10      LDA  ##10
6032- 85 FA      STA  $FA
6034- A9 A0      LDA  ##A0
6036- 85 F9      STA  $F9
6038- 85 FB      STA  $FB
603A- A9 F0      LDA  ##F0
603C- 85 F8      STA  $F8
603E- 2C 52 C0   BIT  $C052
6041- A9 00      LDA  ##00
6043- 85 EA      STA  $EA
6045- 2C 10 C0   BIT  $C010
6048- A9 01      LDA  ##01
604A- 85 FE      STA  $FE
604C- A9 03      LDA  ##03
604E- 85 FD      STA  $FD
6050- EA         NOP
6051- EA         NOP
6052- EA         NOP
6053- EA         NOP
6054- EA         NOP
6055- 20 35 61   JSR  $6135
6058- A5 08      LDA  $08
605A- D0 64      BNE  $60C0
605C- A5 1D      LDA  $1D
605E- A2 00      LDX  ##00
6060- 86 1D      STX  $1D
6062- C9 D3      CMP  ##D3
6064- F0 0D      BEQ  $6073
6066- C9 C1      CMP  ##C1
6068- D0 16      BNE  $6080
606A- C6 FE      DEC  $FE
606C- 10 12      BPL  $6080
606E- A9 03      LDA  ##03
6070- 4C 82 60   JMP  $6082
6073- E6 FE      INC  $FE
6075- A9 04      LDA  ##04
6077- C5 FE      CMP  $FE
6079- D0 05      BNE  $6080
607B- A9 00      LDA  ##00
607D- 4C 82 60   JMP  $6082
6080- A5 FE      LDA  $FE
6082- F0 0D      BEQ  $6091
6084- C9 02      CMP  ##02
6086- F0 0E      BEQ  $6096
6088- C9 03      CMP  ##03
608A- F0 0F      BEQ  $609B
608C- E6 FA      INC  $FA
608E- 4C 9D 60   JMP  $609D
6091- C6 FB      DEC  $FB
6093- 4C 9D 60   JMP  $609D
6096- E6 FB      INC  $FB
6098- 4C 9D 60   JMP  $609D
609B- C6 FA      DEC  $FA
609D- 85 FE      STA  $FE
609F- A5 FA      LDA  $FA
60A1- 20 D0 62   JSR  $62D0
60A4- 85 1B      STA  $1B
60A6- 86 1A      STX  $1A
60A8- A5 FB      LDA  $FB
60AA- 85 1C      STA  $1C
60AC- 20 E0 62   JSR  $62E0
60AF- A5 EA      LDA  $EA
60B1- F0 03      BEQ  $60B6
60B3- 4C 20 63   JMP  $6320
60B6- A2 00      LDX  ##00
60B8- 20 1E FB   JSR  $FB1E
60BB- B4 08      STY  $08
60BD- 2C 30 C0   BIT  $C030
60C0- C6 08      DEC  $08
60C2- EA         NOP
60C3- EA         NOP
60C4- EA         NOP
60C5- 20 35 61   JSR  $6135
60C8- A5 09      LDA  $09
60CA- D0 64      BNE  $6130
60CC- A5 1F      LDA  $1F
60CE- A2 00      LDX  ##00
60D0- 86 1F      STX  $1F
60D2- C9 95      CMP  ##95
60D4- F0 0D      BEQ  $60E3
60D6- C9 88      CMP  ##88
60D8- D0 16      BNE  $60F0
60DA- C6 FD      DEC  $FD
60DC- 10 12      BPL  $60F0
60DE- A9 03      LDA  ##03
60E0- 4C F2 60   JMP  $60F2
60E3- E6 FD      INC  $FD
60E5- A9 04      LDA  ##04
60E7- C5 FD      CMP  $FD
60E9- D0 05      BNE  $60F0
60EB- A9 00      LDA  ##00
60ED- 4C F2 60   JMP  $60F2
60F0- A5 FD      LDA  $FD
60F2- F0 0D      BEQ  $6101
60F4- C9 02      CMP  ##02
60F6- F0 0E      BEQ  $6106
60F8- C9 03      CMP  ##03
60FA- F0 0F      BEQ  $610B
60FC- E6 FB      INC  $FB
60FE- 4C 0D 61   JMP  $610D
6101- C6 F9      DEC  $F9
6103- 4C 0D 61   JMP  $610D
6106- E6 F9      INC  $F9
6108- 4C 0D 61   JMP  $610D
610B- C6 FB      DEC  $FB
610D- 85 FD      STA  $FD
610F- A5 FB      LDA  $FB
6111- 20 D0 62   JSR  $62D0
6114- 85 1B      STA  $1B
6116- 86 1A      STX  $1A
6118- A5 F9      LDA  $F9
611A- 85 1C      STA  $1C
611C- 20 E0 62   JSR  $62E0
611F- A5 EA      LDA  $EA
6121- F0 03      BEQ  $6126
6123- 4C 26 63   JMP  $6326
6126- A2 01      LDX  ##01
6128- 20 1E FB   JSR  $FB1E
612B- 84 09      STY  $09
612D- 2C 30 C0   BIT  $C030
6130- C6 09      DEC  $09
6132- 4C 55 60   JMP  $6055
6135- A9 05      LDA  ##05
6137- 20 A8 FC   JSR  $FCA8
613A- 2C 30 C0   BIT  $C030
613D- AD 00 C0   LDA  $C000
6140- 10 09      BPL  $614B
6142- 2C 10 C0   BIT  $C010
6145- C9 C0      CMP  ##C0
6147- B0 03      BCS  $614C
6149- 85 1F      STA  $1F
614B- 60         RTS
614C- 85 1D      STA  $1D
614E- 60         RTS
614F- EA         NOP
```

Figura 3 - Disassemblato delle routine di Motomuro.

ne di grafica da essere poi costretti a inserire due cicli di ritardo per poter controllare le moto prima che uscissero dal bordo come pallottole).

Il programma sembra un po' lungo ma gran parte è costituito dalla tabella delle Y, che si può generare col programma Basic di figura 1.

Proprio per come è stata realizzata la routine di plottaggio lo schermo è "CIRCOLARE" ovvero tutto ciò che esce da una parte rientra da quella opposta (se non incontra ostacoli); si può quindi realizzare anche una specie di tunnel tra i lati opposti del campo di gioco semplicemente non disegnando parte del bordo. In figura 3 trovate il disassemblato delle routine e in figura 4 (pag. 64) il dump di memoria della YTABLE e della shape di un punto preshiftata. I comandi delle moto sono le paddle di velocità (sono al contrario come il gas delle moto!) e i tasti A ed S per il giocatore di sinistra e ← ed → per quello di destra. Le direzioni si intendono sempre relative al senso di marcia delle moto. Occorre quindi un po' di pratica soprattutto quando la moto si muove verso il basso, ma si risparmia l'uso di due dita. Una precedente versione utilizzava infatti quattro tasti di direzione per ciascun concorrente e spesso si perdeva per aver sbagliato dito: con solo due tasti le probabilità di errore passano da 3 su 4 a 1 su 2.

Il programma in Basic

Sebbene il programma sia totalmente in linguaggio macchina e quindi in grado di girare autonomamente, si è preferito comunque gestire da Basic sia i punteggi che le istruzioni ed alcune altre utility che avrebbero troppo appesantito la routine assembler si da impedirne praticamente la pubblicazione per motivi di spazio sulla rivista. Realizzando inoltre tutte queste

Segue figura 3

```
62D0- A2 00      LDX  ##00
62D2- C9 07      CMP  ##07
62D4- 90 06      BCC  $62DC
62D6- 38         SEC
62D7- E9 07      SBC  ##07
62D9- E8         INX
62DA- D0 F6      BNE  $62D2
62DC- 60         RTS
62DD- EA         NOP
62DE- EA         NOP
62DF- EA         NOP
62E0- A9 00      LDA  ##00
62E2- 85 EA      STA  $EA
62E4- A6 1B      LDX  $1B
62E6- A4 1C      LDY  $1C
62E8- C0 C0      CPY  ##C0
62EA- 90 04      BCC  $62F0
62EC- A0 80      LDY  ##80
62EE- B0 0C      BCS  $62FC
62F0- B9 10 62   LDA  $6210,Y
62F3- 85 06      STA  $06
62F5- B9 50 61   LDA  $6150,Y
62F8- 85 07      STA  $07
62FA- A4 1A      LDY  $1A
62FC- C0 28      CPY  ##28
62FE- B0 13      BCS  $6313
6300- BD 14 63   LDA  $6314,X
6303- 85 1E      STA  $1E
6305- 31 06      AND  ($06),Y
6307- 29 7F      AND  ##7F
6309- F0 02      BEQ  $630D
630B- 85 EA      STA  $EA
630D- A5 1E      LDA  $1E
630F- 51 06      EOR  ($06),Y
6311- 91 06      STA  ($06),Y
6313- 60         RTS
6314- 01 02 04 08
6318- 10 20 40 00
631C- EA         NOP
631D- EA         NOP
631E- EA         NOP
631F- EA         NOP
6320- A9 02      LDA  ##02
6322- 85 08      STA  $08
6324- D0 04      BNE  $632A
6326- A9 01      LDA  ##01
6328- 85 08      STA  $08
632A- A9 08      LDA  ##08
632C- 85 09      STA  $09
632E- C6 09      DEC  $09
6330- F0 17      BEQ  $6349
6332- A9 E0      LDA  ##E0
6334- 85 EA      STA  $EA
6336- A6 EA      LDX  $EA
6338- AD 30 C0   LDA  $C030
633B- CA         DEX
633C- D0 FD      BNE  $633B
633E- C6 EA      DEC  $EA
6340- A6 EA      LDX  $EA
6342- E0 20      CPX  ##20
6344- D0 F2      BNE  $6338
6346- 4C 2E 63   JMP  $632E
6349- 2C 51 C0   BIT  $C051
634C- 2C 10 C0   BIT  $C010
634F- 60         RTS
6350- A5 09      LDA  $09
6352- F0 03      BEQ  $6357
6354- 4C 30 61   JMP  $6130
6357- A5 FD      LDA  $FD
6359- 85 1E      STA  $1E
635B- A5 F9      LDA  $F9
635D- 85 EC      STA  $EC
635F- A5 F8      LDA  $F8
6361- 85 FC      STA  $FC
6363- EA         NOP
6364- EA         NOP
6365- A5 1E      LDA  $1E
6367- F0 0D      BEQ  $6376
6369- C9 01      CMP  ##01
636B- F0 0E      BEQ  $637B
636D- C9 02      CMP  ##02
636F- F0 0F      BEQ  $6380
6371- C6 FC      DEC  $FC
6373- 4C 82 63   JMP  $6382
6376- C6 EC      DEC  $EC
6378- 4C 82 63   JMP  $6382
637B- E6 FC      INC  $FC
637D- 4C 82 63   JMP  $6382
6380- E6 EC      INC  $EC
6382- A5 FC      LDA  $FC
6384- 20 D0 62   JSR  $62D0
6387- 86 1A      STX  $1A
6389- 85 1B      STA  $1B
638B- A4 EC      LDY  $EC
638D- B9 10 62   LDA  $6210,Y
6390- 85 06      STA  $06
6392- B9 50 61   LDA  $6150,Y
6395- 85 07      STA  $07
6397- A6 1B      LDX  $1B
6399- BD 14 63   LDA  $6314,X
639C- A4 1A      LDY  $1A
639E- 31 06      AND  ($06),Y
63A0- 29 7F      AND  ##7F
63A2- D0 03      BNE  $63A7
63A4- 4C CC 60   JMP  $60CC
63A7- A5 EA      LDA  $EA
63A9- F0 14      BEQ  $63BF
63AB- A5 1F      LDA  $1F
63AD- C9 88      CMP  ##88
63AF- D0 07      BNE  $63B8
63B1- A9 95      LDA  ##95
63B3- 85 1F      STA  $1F
63B5- 4C CC 60   JMP  $60CC
63B8- A9 88      LDA  ##88
63BA- 85 1F      STA  $1F
63BC- 4C CC 60   JMP  $60CC
63BF- A5 08      LDA  $08
63C1- 30 15      BMI  $63D8
63C3- A5 1E      LDA  $1E
63C5- C9 03      CMP  ##03
63C7- D0 04      BNE  $63CD
63C9- A9 FF      LDA  ##FF
63CB- 85 1E      STA  $1E
63CD- E6 1E      INC  $1E
63CF- A9 95      LDA  ##95
63D1- 85 1F      STA  $1F
63D3- E6 EA      INC  $EA
63D5- 4C 5B 63   JMP  $635B
63D8- C6 1E      DEC  $1E
63DA- 10 04      BPL  $63E0
63DC- A9 03      LDA  ##03
63DE- 85 1E      STA  $1E
63E0- A9 88      LDA  ##88
63E2- 85 1F      STA  $1F
63E4- E6 EA      INC  $EA
63E6- 4C 5B 63   JMP  $635B
```

```
6150- 20 24 28 2C 30 34 38 3C | 6210- 00 00 00 00 00 00 00 00
6158- 20 24 28 2C 30 34 38 3C | 6218- 80 80 80 80 80 80 80 80
6160- 21 25 29 2D 31 35 39 3D | 6220- 00 00 00 00 00 00 00 00
6168- 21 25 29 2D 31 35 39 3D | 6228- 80 80 80 80 80 80 80 80
6170- 22 26 2A 2E 32 36 3A 3E | 6230- 00 00 00 00 00 00 00 00
6178- 22 26 2A 2E 32 36 3A 3E | 6238- 80 80 80 80 80 80 80 80
6180- 23 27 2B 2F 33 37 3B 3F | 6240- 00 00 00 00 00 00 00 00
6188- 23 27 2B 2F 33 37 3B 3F | 6248- 80 80 80 80 80 80 80 80
6190- 20 24 28 2C 30 34 38 3C | 6250- 28 28 28 28 28 28 28 28
6198- 20 24 28 2C 30 34 38 3C | 6258- A8 A8 A8 A8 A8 A8 A8 A8
61A0- 21 25 29 2D 31 35 39 3D | 6260- 28 28 28 28 28 28 28 28
61A8- 21 25 29 2D 31 35 39 3D | 6268- A8 A8 A8 A8 A8 A8 A8 A8
61B0- 22 26 2A 2E 32 36 3A 3E | 6270- 28 28 28 28 28 28 28 28
61B8- 22 26 2A 2E 32 36 3A 3E | 6278- A8 A8 A8 A8 A8 A8 A8 A8
61C0- 23 27 2B 2F 33 37 3B 3F | 6280- 28 28 28 28 28 28 28 28
61C8- 23 27 2B 2F 33 37 3B 3F | 6288- A8 A8 A8 A8 A8 A8 A8 A8
61D0- 20 24 28 2C 30 34 38 3C | 6290- 50 50 50 50 50 50 50 50
61D8- 20 24 28 2C 30 34 38 3C | 6298- D0 D0 D0 D0 D0 D0 D0 D0
61E0- 21 25 29 2D 31 35 39 3D | 62A0- 50 50 50 50 50 50 50 50
61E8- 21 25 29 2D 31 35 39 3D | 62A8- D0 D0 D0 D0 D0 D0 D0 D0
61F0- 22 26 2A 2E 32 36 3A 3E | 62B0- 50 50 50 50 50 50 50 50
61F8- 22 26 2A 2E 32 36 3A 3E | 62B8- D0 D0 D0 D0 D0 D0 D0 D0
6200- 23 27 2B 2F 33 37 3B 3F | 62C0- 50 50 50 50 50 50 50 50
6208- 23 27 2B 2F 33 37 3B 3F | 62C8- D0 D0 D0 D0 D0 D0 D0 D0
```

Figura 4 - Dump di memoria della YTABLE e della shape di un punto preshiftata.

1A, 1B	Punta alla SHAPE usata da DRAW
1C	Ultimo colore usato
26, 27	Indirizzo del BYTE che contiene il punto x, y
30	Maschera del punto in detto BYTE
E0, E1	x-coord. (0,279)
E2	y-coord. (0,191)
E4	Colore
E6	Pagina (32 = pag.1,64 = pag.2)
E7	SCALE =
E8, E9	Indirizzo delle SHAPE TABLE
EA	Collision counter (usato dalla DRAW)
F9	ROT =

Tabella 2 - Locazioni in pagina Zero usate dalle routine grafiche dell'Applesoft e loro significato.

Locazioni in pagina zero usate dal programma

1A	- x BYTE
1B	- x BIT
1C	- y
1D	- ultimo tasto premuto da A
1E	- ultima Shape usata
1F	- ultimo tasto premuto da B
6,7	- indirizzo dell'ultimo punto plottato
8	- velocità di A
9	- velocità di B
EA	- Flag di collisione (e uso generale)
FD	- direzione attuale di A (vedi tab. 4)
FE	- direzione attuale di B (vedi tab. 4)
FA	- x di B
FB	- y di B
F8	- x di A
F9	- y di A

Tabella 3 - Locazioni di memoria usate dal programma in linguaggio macchina. A corrisponde al giocatore di sinistra, B a quello di destra.

funzioni, puramente estetiche, con un programma in Basic si è data l'opportunità a chiunque lo desideri di modificare a piacere buona parte del gioco senza dover lavorare sulle routine in linguaggio macchina che sono sempre scomode da modificare anche da parte dei più smaliziati.

Oltre alle istruzioni e ad alcune inizializzazioni generali il programma in Basic presenta un menu che consente alcune scelte sul tipo di gioco desiderato. Le principali opzioni sono: il gioco contro il computer, l'allenamento e il gioco normale. In tutti e tre i casi è possibile richiedere la presenza in campo di un numero casuale di ostacoli, si possono togliere parte o tutti i suoni ed è possibile modificare leggermente i limiti di velocità delle moto. Il trucco usato per poter effettuare tutte queste modifiche è molto semplice: il programma in linguaggio macchina è completo ma alcune sue parti vengono saltate o attivate costruendo da Basic, con delle semplici POKE, gli opportuni JSR (GOSUB) alle varie routine. I suoni per semplicità non sono eliminati realmente, ma vengono deviati sull'uscita per il registratore, sicché se qualcuno preferisce può collegare questa uscita al proprio impianto stereo da 1200 watt per canale...

HGR2	F3D8	Seleziona e pulisce la pagina 2 in alta risoluzione
HGR	F3E2	Seleziona e pulisce la pagina 1 in alta risoluzione
CLEAR	F3F2	Pulisce la pagina corrente (nero)
BKGND	F3F6	Colora la pagina corrente con l'ultimo colore utilizzato che si trova in 1C
HPOSN	F411	Posiziona il cursore dell'alta risoluzione senza disegnare nulla; l'Acculumatore contiene la coordinata y e i registri X e Y la parte alta e bassa della coordinata x. Deposita nelle locazioni 26, 27, 30 l'indirizzo di memoria del Bit corrispondente.
HPLOT	F457	Richiama HPOSN e tenta di plottare un punto alle coordinate del cursore. Se le coordinate sono fuori dello schermo torna con un errore, se il punto è di un colore non compatibile con la posizione sullo schermo non plotta niente.
HPLOTTO	F53A	Disegna una linea dall'ultimo punto plottato alle coordinate: X,A = x e Y = y.
FIND	F5CB	Converte la posizione del cursore (locazioni 26, 27 e 30) nelle coordinate x e y; deposita il risultato nelle locazioni E0 - E1 ed E2.
DRAW	F601	Disegna una SHAPE: cerca in X e Y l'indirizzo della Shapetable, usa A come ROT = ed E7 come SCALE = . Il colore è quello corrente.
XDRAW	F65D	Identica alla DRAW tranne che usa il colore complementare a quello del punto da plottare.
SETCOL	F6EC	Setta il colore al valore del registro X. Se questo contiene un numero minore di 0 o maggiore di 7 viene ignorato.

Tabella 1 - Principali routine di grafica in alta risoluzione del Basic Applesoft. Si usano caricando nei registri i valori desiderati ed effettuando un JSR all'indirizzo della routine. Attenzione alla HPLOT che in caso di errore stampa ILLEGAL QUANTITY e salta al BASIC.

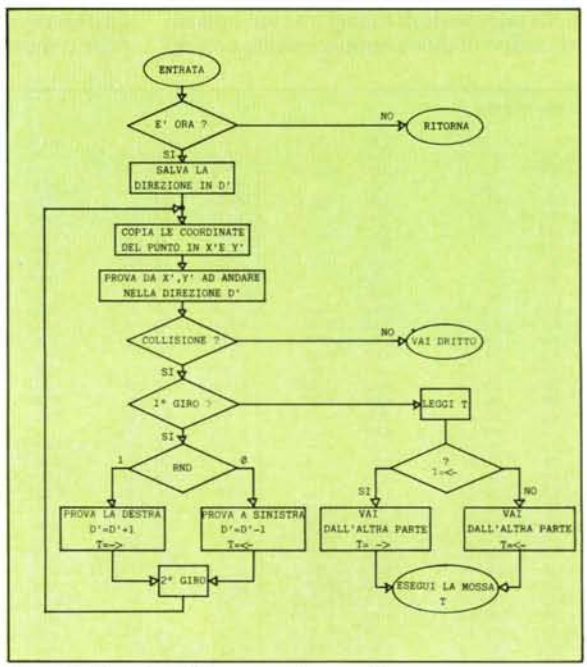


Figura 5 - Flow-chart della routine di mossa del computer.

task 3®

Professione: Elaboratore

STUDIO VENTIDUE - PISTOIA

LA NUOVA GENERAZIONE È GIÀ INIZIATA

Tre posti di lavoro. Tre stampanti. Memoria residente: 192K ram.
Versioni da 5, 10, 20 megabytes su hard disk. Backup su floppy o su disco rimovibile.
Prezzo (unità da 5 Mb + 400K) L. 8.900.000 + IVA.

KYBER® CALCOLATORI

S.R.L. 51100 PISTOIA (ITALY) TEL. 0573/368113 (2 LINEE)
Uffici: VIA ARIOSTO 16-22 Produzione: VIA BELLARIA 54-58

Il gioco contro il computer

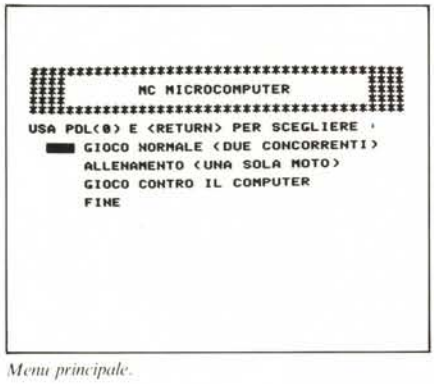
Nonostante il fatto che il computer non abbia alcuna strategia ma si limiti a muovere a caso badando solo a non andare a sbattere contro gli ostacoli è stato più difficile realizzare questa routinetta che tutto il resto del programma; tant'è che, caso rarissimo, si è reso indispensabile l'uso del FLOW-CHART, che trovate in figura 5. Se lo spulciate un attimo notate che tutta la routine si comporta cone se esistesse un giocatore di destra che decide la direzione e poi preme il relativo tasto; da ciò consegue che se qualcuno preme un tasto tra i due del giocatore/computer la mossa viene accettata regolarmente. Questo rende possibile al giocatore di destra una specie di gioco assistito da computer, nel senso che se per caso si trova incastrato o l'avversario gli taglia bruscamente la strada il computer si preoccuperà di non permettere che vada a sbattere e quindi perdere la partita; ma cio non toglie che un giocatore molto abile riesca tuttavia a vincere diverse partite contro un "principiante-computerizzato". MC

0	= Su
1	= Destra
2	= Giù
3	= Sinistra

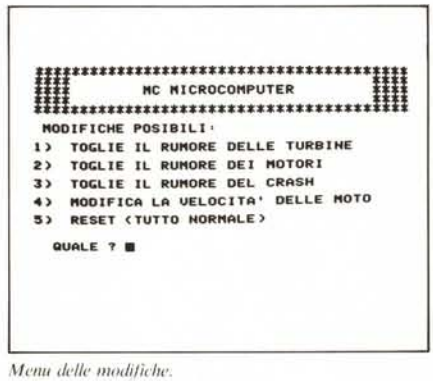
Tabella 4 - Indicatori di direzione, il valore delle locazioni FD e FE può essere uno solo di questi numeri. Ogni incremento corrisponde ad una svolta a destra e ogni decremento ad una svolta a sinistra naturalmente dopo 3 viene zero e prima di zero viene tre.

6000	START	Inizio del programma
6030	GAME	Entrata dopo il disegno del campo di gioco
6050	ooo	Spazio libero per il salto a una eventuale routine di inizio personalizzata (5 BYTE)
6055	LOOP	Inizio del gioco vero e proprio con la lettura della tastiera (6135)
605C-609D		Recupera il tasto premuto ed aggiorna le coordinate del punto in base alla direzione desiderata.
60A1-60B3		Preparazione e salto alla PLOT
60B8	PDL (0)	Legge la velocità dalla paddle 0
60BD	SOUND	Fischio!
60C0	ENDA	Fine mossa per il giocatore di sinistra
60C2	ooo	Tre byte liberi per eventuale JSR
60C5	BMOVE	Inizio mossa per il giocatore di destra del tutto simile alla parte precedente tranne che per le locazioni in pagina zero e la Paddle che diventa (1).
6130	ENDB	Fine della mossa per B
6132	ENDLOOP	Fine del MAIN LOOP
6135	DELAY	Effettua un ritardo di circa 150 µs
613A	RDKEY	Legge un tasto e lo deposita nella locazione KEY del relativo concorrente
614F	ooo	Fine prima parte
6150	Y-TABH	Tabella dei valori precalcolati delle locazioni relative all'asse y. Parte Alta.
6210	Y-TABL	Parte Bassa.
62CF	ooo	Fine della tabella
62D0	XMOD7	Subroutine che calcola il modulo di x/7
62DD	ooo	Tre byte liberi per eventuali JSR
62E0	PLOT	Subroutine che plotta un punto sullo schermo facendo l'XOR con il contenuto precedente. Usa la SHAPE di un punto preshiftata e controlla la presenza di un altro punto coincidente (collisione!).
6314	PSHAPE	Shape di un punto pre-shiftata. Occupa sette byte.
631B	ooo	Cinque byte liberi
6320	BWIN	Entrata routine di vittoria per B
6326	AWIN	Lo stesso per A
6328	WIN	Routine di vittoria. Memorizza il vincitore in 8 (da qui verrà preso dal Basic!) e genera il rumore di collisione
6349	TEXT	Torna in pagina testo
634C	CLRSTRB	Resetta la tastiera
634F	RTS	Torna al chiamante

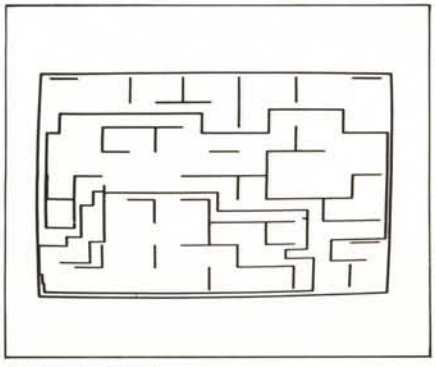
Tabella 5 - Principali punti di entrata del programma.

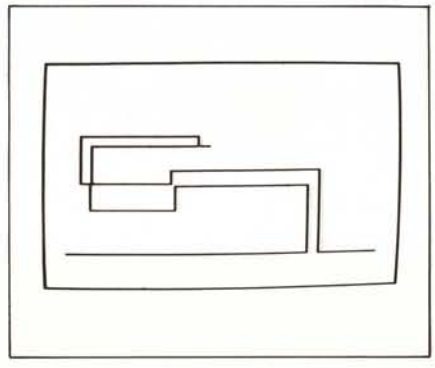


Menu principale.



Menu delle modifiche.





Schermate del gioco.