ISTITUTO NAZIONALE DI FISICA NUCLEARE
Laboratori Nazionali di Frascati

LNF-87/4(R)
29 Gennaio 1987

G. Ciapetti, R. Colini, M. Pistoni e L. Zanello:

ROTO-TRASLAZIONE DI FIGURE BI-TRIDIMENSIONALI CON LA STAZIONE GRAFICA WHIZZARD 7200 DELLA MEGATEX. APPLICAZIONE: PROGRAMMA PER IL DISPLAY E L'ANALISI INTERATTIVA DEI DATI DEL MICROVERTEX DI UA1.

Servizio Documentazione
dei Laboratori Nazionali di Frascati

INDICE

INFN - Laboratori Nazionali di Frascati
Servizio Documentazione

LNF-87/4(R)
29 Gennaio 1987

ROTO-TRASLAZIONE DI FIGURE BI-TRIDIMENSIONALI CON LA STAZIONE GRAFICA WHIZZARD 7200 DELLA MEGATEK. APPLICAZIONE: PROGRAMMA PER IL DISPLAY E L' ANALISI INTERATTIVA DEI DATI DEL MICROVERTEX DI UA1.

G.Ciapetti(*), R.Colini(*), M.Pistoni(o) e L.Zanello(*)

(*) sezione INFN, Istituto di Fisica dell' Universitá, P.le A. Moro 7, 00100 ROMA

(o) LNF, via E. Fermi 40, 00044 Frascati (ROMA)

1. - Il MEGATEK

Il Sistema WHIZZARD 7200 della MEGATEK (che d' ora in poi chiameremo impropriamente MEGATEK per semplicitá) é una stazione grafica a colori pseudo-tridimensionale dotata di una CPU per l'elaborazione dei dati provenienti dall' host computer a cui é interfacciato.

Il MEGATEK puó rappresentare a discrezione dell' utente sia figure piane (bidimensionali) come comunemente operano i piú diffusi terminali grafici, sia figure nello spazio tridimensionale proiettate. La tridimensionalitá é ottenuta con la rappresentazione assonometrica dei dati graficati.

La principale caratteristica del MEGATEK e' quella di avere elevata velocitá di elaborazione; ció é nato dall' esigenza di rappresentare sullo schermo, figure in movimento, ossia figure soggette a trasformazioni particolari come ad esempio zoom, traslazioni e rotazioni: per ogni trasformazione, qualunque device grafico convenzionale deve necessariamente cancellare la figura e ricrearla nella nuova posizione e con i nuovi attributi. Questo implica il ricalcolo di tutte le coordinate dei punti costituenti tale figura (ricalcolo tanto piú lungo e laborioso, tanto piú é complessa la figura medesima soggetta a trasformazione). Questo problema nel MEGATEK

é risolto a livello hardware. Infatti esso contiene un microprocessore adibito esclusivamente alle operazioni di trasformazione delle coordinate: HCRST (Hardware Clip, Rotate, Scale, and Translate).

L' HCRST presenta un triplice vantaggio: essendo esclusivamente dedicato a tale compito, effettua l' intero ricalcolo di tutte le coordinate in tempi estremamente brevi, inoltre l' host non interviene in tale operazione e quindi non consuma CPU per essa, infine essendo l' HCRST interno al MEGATEK, si risparmiano i tempi di trasferimento delle coordinate dei punti appena ricalcolate, dall' host computer al MEGATEK stesso.

Tutta la struttura hardware del MEGATEK é improntata alla realizzazione di un device grafico molto veloce nella elaborazione e visualizzazione dei dati provenienti dall' host. Esso infatti é dotato di due buses su cui viaggiano le informazioni: uno a 32 bits (Graphic Data Bus) é quello su cui viaggiano i dati provenienti dall' host, che vengono visualizzati sul monitor dopo essere stati opportunamente trasformati, l' altro a 16 bits (Peripheral Bus) trasferisce i dati provenienti dai devices periferici del MEGATEK verso l' host computer.

Direttamente sul Graphic Data Bus sono connesse varie interfaccie:

- l' interfaccia con l' host che permette lo scambio dei dati tra MEGATEK e host computer; é un' interfaccia parallela ad alta velocitá dotata di accesso diretto alla memoria dell' host, DMA;

- una scheda di memoria (Display-List Memory) nella quale vanno a risiedere tutte le istruzioni di grafica provenienti dall' host e le informazioni dei vettori da visualizzare sul MEGATEK nonché gli attributi con i quali questi devono essere rappresentati;

- il Graphics-Processor che si occupa di interpretare le istruzioni di grafica residenti nella Display-List Memory e di interpretare i dati riguardanti i vettori e i relativi attributi in essa contenuti. Il Graphic-Processor inoltre contiene un generatore di caratteri hardware e controlla il flusso dei dati dalla Display-List Memory al Vector-Generator;

- l' HCRST che si occupa di trasformare i vettori prima di passarne le coordinate al Vector-Generator; le trasformazioni sono funzione di alcuni parametri di rotazione, traslazione, zoom e clip calcolati dall' host secondo ben determinate matrici di trasformazione. Tali parametri sono residenti nella Display-List Memory come attributi del dato segmento da trasformare in una zona di memoria riservata a tale segmento chiamata Segment-Header (dei segmenti si parlerá in seguito);

- il Vector-Generator che si occupa di convertire le coordinate dei vettori contenute nella Display-List Memory e di scrivere nella Memoria di Schermo i bit che lo rappresentano fisicamente sul monitor;

- la Memoria di Schermo (Bit-Map Memory) che é in grado di contenere 1024^2 punti con la relativa informazione di colore e di intensitá luminosa. Ad essa é interfacciato direttamente il monitor avente appunto una risoluzione di 1024*1024 pixels. Il monitor del MEGATEK é raster scan a colori e quindi l' immagine da esso rappresentata viene rinfrescata ad intervalli di tempo regolari; l' informazione di quelli che sono i pixels accesi risiede appunto nella Memoria di Schermo.

Sul Peripheral Bus sono invece interfacciati i seguenti devices:

- la Tastiera (Keyboard), particolarmente utile per l' input di dati di tipo alfanumerico;

- la Tablet , particolarmente utile a selezionare finestre sul display o per individuare le coordinate di determinati punti visualizzati;

- il Joystick, particolarmente utile per effettuare trasformazioni sulle figure rappresentate sullo schermo.

Tutti e tre questi devices sono in diretta comunicazione anche col Graphic-Data Bus in modo tale da effettuare l' eco sul display in modo immediato senza l' intervento dell' host. Inoltre l' interfaccia con l' host é in comunicazione anche con il Peripheral Bus e quindi i dati provenienti da questi devices sono direttamente spediti all' host ad ogni sua richiesta.

Il diagramma di flusso di come lavora il MEGATEK, ad esempio durante le operazioni di trasformazione comandate interattivamente, é quindi il seguente: dapprima l' host invia le coordinate dei vettori che il programmatore vuole visualizzare (organizzate in un segmento), queste vanno a risiedere nella Display-List Memory, il Graphics-Processor interpreta tali coordinate e le invia all' HCRST per eventuali trasformazioni, le trasformazioni vengono effettuate secondo parametri contenuti nella Display-List Memory (con esattezza nel Segment-Header), inizialmente aventi i valori di default (per i quali, in veritá, la figura non é soggetta ad alcuna trasformazione), quindi il Graphics-Processor passa l' informazione al Vector-Generator che scrive i vettori sulla Memoria di Schermo e quindi sul monitor. A questo punto l' host interroga il joystick, il quale invia una word contenente l' informazione di una sua eventuale deflessione in x e/o in y, l' host elabora tale informazione e in base ad essa (quindi in funzione delle deflessioni del joystick) calcola i nuovi elementi della matrice di trasformazione e li rispedisce

al MEGATEK; sará quindi il Graphics-Processor a trasferire tali dati nel Segment-Header nella Display-List Memory; e ricomincia il ciclo.

L' host quindi ricalcola ad ogni ciclo al piú 8 elementi della matrice di trasformazione (le prime due colonne), mentre le nuove posizioni di tutti i vettori sono ricalcolate dall' HCRST in hardware internamente al MEGATEK.

Si noti che ogni ciclo viene effettuato in tempi veramente brevi, grazie alle elevate capacitá e velocitá di elaborazione del MEGATEK e questo permette di far compiere al MEGATEK, in brevissimo tempo, molte piccole trasformazioni (in funzione di piccole variazioni degli elementi della matrice di trasformazione effettuate dall' host), in modo da avere l' illusione di trasformazioni continue cioé di figure in movimento: il MEGATEK é in grado quindi di rappresentare movimenti in tempo reale delle figure da esso graficate sul display.

La differenza sostanziale tra i devices grafici tradizionali e passivi ed il MEGATEK é che esso é anche in grado di elaborare i dati provenienti dall' host computer.

Nella sezione di Roma il MEGATEK é interfacciato al VAX 8600 della Digital.

2. - La libreria WAND

2.1 - Introduzione

WAND é il package software realizzato per la gestione interattiva della grafica del MEGATEK organizzato in una libreria.

Tale libreria consente il migliore e semplificato utilizzo di tutte le capacitá del MEGATEK; infatti essa permette di rappresentare sul display qualsiasi tipo di figura bidimensionale o tridimensionale con la creazione di un insieme di vettori avente i propri particolari attributi e di caricare queste informazioni nella Display-List Memory; inoltre essa permette di manipolare con estrema semplicitá i dati contenuti nella Display-List Memory e permette inoltre di sfruttare una particolaritá del MEGATEK: la "segmentation" cioé la creazione di ben 768 subroutines grafiche, e altrettanti segmenti ognuno gestibile indipendentemente e contemporaneamente agli altri e avente ognuno i propri particolari attributi (l' argomento verrá approfondito nel seguito). La libreria permette poi la trasformazione (roto-traslazione-zoom) dei segmenti giá creati; infine la gestione completa di tutti gli organi periferici del MEGATEK quali joystick, tablet e keyboard.

2.2 - System Level Routines

Alle fondamenta di WAND c'é il System level. Esso comprende il driver per il device, le routines strettamente legate al tipo di host computer, le routines di inizializzazione del MEGATEK, le routines per l' allocazione del display, dei vari devices periferici, le routines che riportano i messaggi d' errore e cosí via.

Tali routines sono riservate ed in generale l' utente non ha necessitá di utilizzarle.

2.3 - Workstation Level Routines

Sono le routines disponibili all' utente per la creazione di immagini e la loro manipolazione.

Tutte contenute nella libreria WAND, tra esse oltre alle classiche routines per inizializzare la grafica, maneggiare il buffer, muovere il pennino, tracciare vettori, scrivere testi, dividere il display in piú finestre, porne i limiti fisici, controllare la generazione degli errori, cambiare tutti gli attributi di una data figura rappresentata entro un dato segmento (cioé cambiare colore, cambiare le dimensioni dei caratteri e dei markers, produrre distinte combinazioni di line patterns, blink rates, tipi di markers, tipi di coordinate, angoli di orientazione delle scritte, colori e intensitá luminosa, etc.), troviamo le routines di interazione con le periferiche per la grafica, cioé tutte le routines necessarie per il selezionamento, l' interrogazione, la lettura e la campionatura dei devices grafici, quali l' hardcopy, il joystick, la tablet, la light pen, la keyboard, etc.

2.4 - Utility Level Routines

Tra le varie facility di WAND, una delle piú importanti é che essa permette all'utente di creare un proprio insieme di utilities per effettuare frequenti e ripetute funzioni grafiche sempre nell' ambito dello stesso programma.

Queste utilities costruite nell'interno del programma principale sono in pratica costituite da un set di istruzioni risiedenti nella memoria del MEGATEK e pronte ad essere immediatamente eseguite ogni qual volta vengano richiamate dal programma principale (subroutines) oppure

atte ad essere trasformate ogni volta lo richieda il programma principale (segmenti).

Appare giá evidente una importante differenza tra queste due classi di utility level routines, ma ulteriori differenze saranno evidenziate qui appresso.

2.4.1 - Subroutines

Le subroutines possono essere utili qualora si intenda creare piú volte, magari in posizioni diverse sullo schermo, oppure in tempi diversi nello stesso punto, la stessa figura.

Quando si apre una subroutine il MEGATEK é pronto ad immagazzinare in una parte della sua memoria di lavoro, le istruzioni appresso elencate nel programma fino all'istruzione di chiusura della subroutine stessa (la parola "istruzioni" é da intendersi come istruzioni per il MEGATEK ovvero le chiamate alle routines di WAND).

A questo punto con una particolare routine di WAND si puó richiamare la subroutine ogni qual volta si vuole, e sará l' hardware del MEGATEK ad occuparsi di rappresentarne la figura sullo schermo alle coordinate specificate nella chiamata stessa.

É logico quindi che le istruzioni contenute in una subroutine non vengono rappresentate sullo schermo al momento in cui vengono eseguite dal programma chiamante bensí quando vengono richiamate da esso.

I vantaggi che esse rappresentano sono: - a) maggiore velocitá di esecuzione - b) snellimento del lavoro della CPU del VAX in quanto la loro esecuzione é ad esclusivo carico del MEGATEK.

2.4.2 - Segments

Al contrario delle subroutines, le istruzioni contenute tra le chiamate alle routines di apertura e di chiusura di un segmento, vengono rappresentate sullo schermo al momento in cui vengono eseguite dal programma chiamante.

Tutto ció che individua le caratteristiche della figura, rappresentata dalle istruzioni in esso contenute, quali colore, luminositá, line pattern, blink rate, etc., viene memorizzato in una area di memoria del MEGATEK a questo scopo destinata (Segment-Header), contenuta nella Display-List Memory. Questo affinché tali caratteristiche o attributi possano essere

cambiati a piacere e volontá del programmatore; cosicché é possibile cambiare ad esempio il colore di una figura sul display con una semplice chiamata ad una routine di WAND senza dover necessariamente cancellare tale figura e ricrearla con diverso colore (come avveniva nei devices grafici tradizionali), ma semplicemente cambiando l' attributo del colore nel display list relativo al segmento che rappresenta tale figura; la chiamata in proposito interviene a modificare l'attributo nel Segment-Header relativo alla figura di cui si intende cambiare tale attributo.

Non é tutto: tramite l' uso di opportune routines di WAND é possibile far operare delle trasformazioni (rotazioni, traslazioni e zoom) a figure costruite opportunamente con istruzioni che siano contenute entro un segmento.

A tali routines occorre passare come argomento gli elementi delle prime due colonne di una matrice di trasformazione (3*3 per figure bidimensionali e 4*4 per figure tridimensionali) ricavata dal prodotto di piú matrici: una relativa al fattore di scala, una relativa all' orientazione (tre nel caso tridimensionale) e una relativa alla traslazione.

É chiaro che variando di poco e con continuitá i vari elementi di tali matrici, ricalcolando ad ogni step la matrice prodotto e chiamando l' opportuna routine di trasformazione, si ottiene su schermo una trasformazione continua della desiderata figura.

3. - Subroutines di calcolo delle matrici di trasformazione

Allo scopo di far effettuare a figure bi-tridimensionali, rappresentate entro un dato segmento, movimenti di roto-traslazioni e zoom, sono state appositamente create delle subroutines di utilizzo generale per il calcolo degli elementi della matrice di trasformazione.

Esse si preoccupano di elaborare i dati in ingresso, quali il punto di vista (cioé l'angolo o gli angoli che individuano la posizione della figura rispetto agli assi cartesiani), il fattore di scala, e il fattore di traslazione nonché, nel caso piú generale, le coordinate dei punti, non necessariamente coincidenti, rispetto ai quali si effettua la rotazione o lo zoom.

L' elaborazione consiste nel calcolo degli elementi delle prime due colonne di una matrice di trasformazione. Nel caso delle trasformazioni di figure bidimensionali, tale matrice é 3*3 ed é ottenuta come prodotto di 3 matrici di uguali dimensioni, ognuna rappresentante una

trasformazione elementare: una rappresenta lo zoom, una la rotazione rispetto all' asse Z e una la traslazione. Invece nel caso delle trasformazioni di figure tridimensionali, tale matrice é 4*4 ed é ottenuta come prodotto di 5 matrici di uguali dimensioni, ognuna rappresentante una trasformazione elementare: una rappresenta lo zoom, una la rotazione rispetto all' asse X, una la rotazione rispetto all' asse Y, una la rotazione rispetto all' asse Z, e una la traslazione. Quindi nel primo caso gli elementi da calcolare sono 6 e nel secondo 8.

Trasferendo gli elementi di tale matrice risultante al MEGATEK, il Graphics-Processor li inserisce nel Segment-Header del segmento rappresentante la figura da trasformare.

Le matrici rappresentanti le trasformazioni elementari sono riportate nell' appendice A.

Cosí calcolati tali elementi, non permettono peró trasformazioni intorno a punti generici diversi dall' origine degli assi (di coordinate (0,0,0)), rappresentato nel centro del monitor. Il problema é stato risolto costruendo delle opportune subroutines che non si limitano a calcolare tali elementi secondo il metodo descritto. Queste infatti precedentemente elaborano le coordinate dei punti rispetto ai quali si intende effettuare la trasformazione e dopo effettuano, sulle nuove coordinate appena elaborate, le trasformazioni volute.

L' elaborazione dei centri di trasformazione consiste nell' applicare una trasformazione inversa a tali coordinate, in modo tale da far risultare sul display il centro di trasformazione sempre visualizzato nello stesso punto: ció equivale a veder trasformare la figura proprio intorno a quel punto.

Le subroutines realizzate si chiamano:

MTX3X3

MTX3X3G

MTX4X4

MTX4X4G

Le prime due utili per trasformazione di figure bidimensionali e le altre per trasformazione di figure tridimensionali.

Quelle con la lettera G finale si differenziano dalle altre perché sono piú generali in quanto permettono di effettuare trasformazioni rispetto a qualsivoglia punto dello schermo, mentre le altre, pur avendo il vantaggio di essere di esecuzione piú rapida, permettono di effettuare trasformazioni esclusivamente rispetto all' origine degli assi.

La sintassi per richiamarle é rispettivamente:

CALL MTX3X3 (sclx, scly, rotz, trx, try, mt2)

CALL MTX3X3G (sclx, scly, csclx, cscly, rotz, crotx, croty, trx, try, mt2)

CALL MTX4X4 (sclx, scly, sclz, rotx, roty, rotz, trx, try, trz, mt3)

CALL MTX4X4G

(sclx, scly, sclz, csclx, cscly, csclz, rotx, roty, rotz, crotx, croty, crotz, trx, try, trz, mt3)

ove:

sclx fattore di scala in X

scly fattore di scala in Y

sclz fattore di scala in Z

csclx coordinata X del centro di scala

cscly coordinata Y del centro di scala

csclz coordinata Z del centro di scala

rotx angolo di rotazione rispetto all' asse X

roty angolo di rotazione rispetto all' asse Y

rotz angolo di rotazione rispetto all' asse Z

crotx coordinata X del centro di rotazione

croty coordinata Y del centro di rotazione

crotz coordinata Z del centro di rotazione

trx traslazione in X

try traslazione in Y

trz traslazione in Z

mt2 matrice (3*3), argomento di ritorno

mt3 matrice (4*4), argomento di ritorno

Sono state realizzate inoltre altre 2 subroutines, utili per l'interrogazione della tablet e del joystick.

Le subroutines si chiamano:

READJOY

READTAB

La prima acquisisce l'informazione relativa alla posizione del joystick, ed entrambe comunicano lo stato degli switches presenti sui due devices periferici joystick e tablet (1 sul joystick e 4

sulla tablet).

La sintassi per richiamarle é rispettivamente:

CALL READTAB (*locsts*, *ibutZ*, *ibut1*, *ibut2*, *ibut3*, *ibuton*)

CALL READJOY (*locsts*, *factn*, *dx*, *dy*, *ibuton*)

ove i parametri di ingresso sono:

locsts locator device status (ottenuto con la CALL MWSMLC di WAND)

factn fattore di normalizzazione di *dx* e *dy*

e quelli di uscita sono:

dx delta X di spostamento del joystick

dy delta Y di spostamento del joystick

ibuton indica se é premuto almeno uno switch

ibutz vale 1 se é premuto il bottone Z sulla tablet (altrimenti vale 0)

ibut1 vale 1 se é premuto il bottone 1 sulla tablet (altrimenti vale 0)

ibut2 vale 1 se é premuto il bottone 2 sulla tablet (altrimenti vale 0)

ibut3 vale 1 se é premuto il bottone 3 sulla tablet (altrimenti vale 0)

Si noti che *dx* e *dy* valgono al massimo in valore assoluto *factn* (nel caso di deflessione massima in X o in Y del joystick), vi sono peró due steps intermedi per i quali essi assumono rispettivamente valore $\frac{factn}{2}$ (nel caso di deflessione media) e $\frac{factn}{4}$ (nel caso di deflessione minima), valgono zero se il joystick é nella posizione di riposo.

Infine e stata realizzata un' ulteriore subroutine:

WAIT

che permette di mettere il proprio processo attivo in uno stato di wait, ovvero in uno stato di ibernazione durante un certo intervallo di tempo desiserato; quando il processo si trova in questo stato, esso non consuma CPU. L' utilitá di questa subroutine risulta evidente qualora si voglia ottimizzare il tempo di CPU usato dal programma durante i loops di lettura dei devices periferici (quali la tablet e il joystick) quando questi non vengano manovrati dall' utilizzatore. In questi casi, tra un ciclo di lettura e l' altro si puó interporre un wait di qualche decimo di secondo (o al limite di qualche secondo).

La sintassi per richiamarla é:

CALL WAIT (*deltatime*)

ove l' unico parametro d' ingresso é:

deltatime intervallo di tempo di wait (stringa di 20 caratteri precedentemente definita nel programma).

Il listato di tutte queste subroutines descritte é riportato in appendice B. I moduli oggetto ed i sorgenti, inoltre, possono essere trovati rispettivamente nelle librerie residenti su:

VAXLNF::DISK$USER1:[MASSIMO.MEGATEK]MAXROB.OLB e

VAXLNF::DISK$USER1:[MASSIMO.MEGATEK]MAXROB.TLB .

Infine nell' appendice C sono riportati due esempi di semplici programmini che utilizzano alcune di queste subroutines:

MENU : gestisce un menu' tramite l' uso della tablet;

CUBO : effettua la rotazione di un cubo interrogando il joystick.

4. - Programma MICROVERTEX

Il programma consente la visualizzazione dell' apparato MICROVERTEX dell' esperimento UA1 e delle tracce da esso rivelate sul device grafico MEGATEK che ne dá in ogni istante un'immagine tridimensionale.

Il detector, nel suo insieme, é costituito da un cilindro cavo lungo 80 cm, le cui basi hanno un raggio esterno di 8.5 cm, un raggio interno di 2.5 cm ed é suddiviso in 16 camere a drift identiche.

Ogni camera é costituita da 16 fili equispaziati, il cui piano é al centro dello spicchio, che permettono di misurare un certo numero di coordinate lungo la traiettoria della particella:

- la coordinata Y, lungo la linea dei fili, é individuata dalla posizione del filo che ha raccolto il segnale;
- la coordinata Z, ortogonale alla linea dei fili, é ottenuta dalla misura del tempo di drift, supponendo nota la velocitá;
- la coordinata X, ortogonale al piano della camera, é ottenuta dalla misura della divisione di carica. La precisione sulla coordinata di drift é dell' ordine della decina di micron (50-100

micron), mentre lungo l' asse X é circa il 2% della lunghezza del filo (1.6 mm).

Lo scopo principale del programma é quello di analizzare visivamente i risultati della ricostruzione delle tracce (pattern recognition).

A tal fine le funzioni fondamentali del programma sono:

1) visualizzare tridimensionalmente l' appparato nel suo insieme, le coordinate lette dal sistema di acquisizione dati e i risultati del pattern recognition (elementi di traccia ottenuti); questa ultima operazione non é stata ancora implementata;

2) visualizzare l' apparato sul piano Y-Z, ove si hanno le migliori risoluzioni spaziali;

3) zoomare, ruotare e traslare l' apparato per poter cogliere tutti i dettagli dell' evento rivelato;

4) visualizzare una singola camera oppure una coppia di camere contigue per poter studiare in dettaglio gli elementi di traccia.

4.1 - Utilizzazione del programma

L' esecuzione del programma puó essere semplicemente richiamata tramite il comando:

```
$ RUN  DSK2:[VCEUA1.GRAPH.MAX]VERTEX1          (sul nodo VAXROM).
```

A questo punto sullo schermo del MEGATEK viene visualizzato:

a) L' apparato MICROVERTEX sul piano X-Y.

b) Il sistema di riferimento solidale con l' apparato.

c) Il menú delle operazioni possibili suddiviso in:

1) SYSTEM MENÚ , relativo alle operazioni di rotazione, traslazione, zoom e in generale alle trasformazioni di coordinate, posto in alto e a destra.

2) USER MENÚ , relativo all' inserimento dei punti e delle tracce di ogni evento e all' osservazione della singola camera o della coppia di camere posto in basso e a destra.

Per poter interagire con le periferiche del MEGATEK, é necessario fornire da terminale il file su cui si leggeranno gli eventi.

Una volta eseguita tale operazione termina l' interazione col terminale e tutte le interazioni col programma sono effettuate tramite le periferiche del MEGATEK.

Infatti ogni operazione viene selezionata posizionando il cursore del mouse della tablet sulla corrispondente finestra di menú (che cambierá colore) e premendo uno dei bottoni del mouse stesso.

4.2 - Descrizione delle operazioni possibili

4.2.1 - Trasformazioni continue di coordinate

- ZOOM -

Ingrandisce o rimpiccolisce l' apparato dello stesso fattore nelle tre direzioni X, Y, Z.

- TRANSLATE -

Trasla l' apparato lungo le direzioni Y e Z.

- ROTATE X-Y -

Ruota l' apparato intorno agli assi X e Y.

- ROTATE Y-Z -

Ruota l' apparato intorno agli assi Y e Z.

- ROTATE X-Z -

Ruota l' apparato intorno agli assi X e Z.

Tali operazioni sono caratterizzate dall' uso del joystick che permette di effettuare in modo continuo le trasformazioni desiderate.

La logica di utilizzo é del tipo :

1) La posizione dell' eco del mouse della tablet al momento della selezione determina la velocitá della trasformazione, crescente nella direzione delle X crescenti.

2) La finestra di menú selezionata lampeggia quando il joystick é abilitato per la trasformazione richiesta.

3) La trasformazione viene effettuata quando il bottone del joystick viene mantenuto premuto. Quando sia il joystick che la tablet sono fermi il programma si mette in WAIT per un secondo e ció permette di ottimizzare il consumo di CPU.

4) Il joystick continua ad essere abilitato per la operazione di menú che lampeggia, anche se

l'eco della tablet viene spostato; infatti esso ne viene disabilitato solo nel momento in cui con la tablet si seleziona un' altra operazione.

5) L' utilizzazione del mouse come quella del joystick sono facilitate da un frame di istruzioni posto nella parte piú bassa del display.

4.2.2 - Operazioni immediate

- RESET -

Annulla tutte le trasformazioni precedentemente effettuate, mostrando l' apparato sul piano X-Y.

- VIEW X-Y -

Mostra l' apparato sul piano X-Y (visto dall' asse Z) annullando le precedenti rotazioni e traslazioni, ma lasciando inalterati i fattori di scala.

- VIEW Y-Z -

Mostra l' apparato sul piano Y-Z (visto dall' asse X) annullando le precedenti rotazioni e traslazioni, ma lasciando inalterati i fattori di scala.

- VIEW X-Z -

Mostra l' apparato sul piano X-Z (visto dall' asse Y) annullando le precedenti rotazioni e traslazioni, ma lasciando inalterati i fattori di scala.

- EXIT -

Uscita dal programma e clear dello schermo.

- NEXT -

Permette di visualizzare i punti relativi ad un evento. La prima selezione di tale operazione visualizza il primo evento nel file di dati. Successive selezioni visualizzano successivi eventi nel file. Il messaggio:

No more data in file < filename >

indica che bisogna fornire un altro file di dati su cui leggere altri eventi.

Per comoditá dell' utente, il nome del file e il numero di evento attualmente visualizzato

sono indicati nel frame piú alto del display.

- TRACKS -

Ricostruisce le traiettorie delle particelle dai punti rivelati, ma non é ancora implementata.

- BACK -

Permette di passare dalla visualizzazione di una o due camere a drift (selezionata con le operazioni SECTOR e COUPLE OF SECTORS) a quella dell' intero MICROVERTEX.

Se il settore o i settori visualizzati erano stati trasformati, l' opzione BACK fornisce l'intero apparato a cui sia stata applicata la stessa trasformazione.

La finestra BACK viene resa invisibile quando sia giá visualizzato l' intero MICROVERTEX.

- SECTOR -

Permette di visualizzare una singola camera del MICROVERTEX.

Selezionando l' operazione SECTOR e mantenendo premuto uno dei bottoni del mouse si ha:

- la numerazione delle camere attorno ad una base dell' apparato (qualunque sia la sua posizione).

- la comparsa di un menú orizzontale contenente il numero d' ordine delle 16 camere.

La scelta puó essere effettuata rilasciando il bottone della tablet sul numero di camera che si vuole ottenere.

Per comoditá dell' utente il numero di settore attualmente visualizzato viene scritto in basso a destra del display.

- COUPLE OF SECTORS -

Permette di visualizzare due camere contigue. La loro scelta é identica a quella del singolo settore e dunque avviene mediante numerazione delle camere e un menú contenente le coppie che possono essere scelte.

Anche in questo caso, il numero dei settori attualmente visualizzati viene scritto in basso e a destra del display.

Queste operazioni hanno la caratteristica di far uso del solo mouse della tablet.

L' operazione viene eseguita premendo uno dei bottoni del mouse della tablet sulla finestra corrispondente.

Il sorgente del programma puó essere trovato nel file:

VAXLNF::DISK$USER1:[MASSIMO.MEGATEK]VERTEX1.FOR .

APPENDICE A

Matrici di trasformazione elementare utilizzate dal MEGATEK

Se configurato con l' HCRST appropriato, il Sistema Whizzard 7200 é in grado di compiere trasformazioni bi- e tridimensionali di una immagine, tagliando l' immagine risultante secondo una data finestra con minimo intervento dell' host computer.

L' host fornisce i parametri di traslazione, rotazione e ingrandimento sotto forma di elementi della matrice di trasformazione. L' HCRST applica tali parametri a ciascun vettore e poi taglia il vettore risultante in base alla finestra data prima di farne il display.

I parametri sono memorizzati nell' header di trasformazione e in istruzioni di display list; é possibile pertanto applicare parametri diversi a porzioni diverse della display list senza intervento dell' host.

Il sistema di coordinate usato dal Whizzard 7200 é sinistrorso. Una rotazione positiva intorno ad un certo asse avviene in senso antiorario guardando da un punto situato sul semiasse positivo di tale asse verso l' origine. L' asse Z di default é perpendicolare allo schermo: valori negativi sono verso l' interno dello schermo e valori positivi verso l' esterno di esso. Il semiasse positivo dell' asse X é verso destra e il semiasse positivo dell' asse Y é verso l' alto guardando lo schermo.

Rotazioni e ingrandimenti sono fatti rispetto all' origine. Le traslazioni sono riferite all' origine definita per lo schermo. Per ingrandire o ruotare rispetto ad un punto arbitrario, prima si trasla il punto all' origine, poi si compie l' operazione, poi si ritrasla indietro. Trasformazioni complesse sono il prodotto di trasformazioni semplici. L' ordine in cui si effettua l' ingrandimento e la rotazione influenza l' immagine risultante.

Matrici relative a diverse operazioni possono essere combinate in un' unica matrice che rappresenta la sequenza completa.

Nelle formule seguenti, useremo queste abbreviazioni:

SCLX :	fattore di scala - X
SCLY :	fattore di scala - Y
SCLZ :	fattore di scala - Z
ROTX :	angolo di rotazione attorno all' asse X (radianti)
ROTY :	angolo di rotazione attorno all' asse Y (radianti)
ROTZ :	angolo di rotazione attorno all' asse Z (radianti)
TRX :	traslazione - X
TRY :	traslazione - Y
TRZ :	traslazione - Z

In trasformazioni bidimensionali, ingrandimento, rotazione e traslazione possono essere rappresentati dalle tre semplici matrici:

$$SCL = \begin{bmatrix} SCLX & 0 & 0 \\ 0 & SCLY & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$ROT = \begin{bmatrix} cos(ROTZ) & sin(ROTZ) & 0 \\ -sin(ROTZ) & cos(ROTZ) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$TR = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ TRX & TRY & 1 \end{bmatrix}$$

Per trasformazioni bidimensionali, queste matrici possono essere combinate in qualsiasi ordine per produrre una matrice 3x3 chiamata D. I parametri della trasformazione sono quindi:

$$D = \begin{bmatrix} M_0 & M_3 & 0 \\ M_1 & M_4 & 0 \\ M_2 & M_5 & 1 \end{bmatrix}$$

L' HCRST produce il vettore trasformato $X'\ Y'$ dal vettore della display list $X\ Y\ 1$ compiendo la seguente moltiplicazione di matrice su ciascun vettore prima che esso venga tagliato e rappresentato sullo schermo:

$$[X'\ Y'] = [X\ Y\ 1]\, D$$

La rappresentazione algebrica di trasformazioni tridimensionali é piú complicata poiché rotazioni rispetto ai tre assi possono essere fatte in qualsiasi ordine. Dato un ordine per le rotazioni e visto che solo le componenti X e Y dei vettori trasformati sono tagliate, la trasformazione puó essere ridotta ad una matrice 4x2 chiamata T. La matrice T é il prodotto di quattro matrici semplici in un qualunque ordine dato.

$$SCL = \begin{bmatrix} SCLX & 0 & 0 & 0 \\ 0 & SCLY & 0 & 0 \\ 0 & 0 & SCLZ & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$TR = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ TRX & TRY & TRZ & 1 \end{bmatrix}$$

$$RX = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(ROTX) & sin(ROTX) & 0 \\ 0 & -sin(ROTX) & cos(ROTX) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$RY = \begin{bmatrix} cos(ROTY) & 0 & sin(ROTY) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(ROTY) & 0 & cos(ROTY) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$RZ = \begin{bmatrix} cos(ROTZ) & sin(ROTZ) & 0 & 0 \\ -sin(ROTZ) & cos(ROTZ) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

I parametri di trasformazione per la matrice T sono quindi:

$$T = \begin{bmatrix} M_0 & M_3 & X_{13} & 0 \\ M_1 & M_4 & X_{23} & 0 \\ M_6 & M_7 & X_{33} & 0 \\ M_2 & M_5 & X_{43} & 0 \end{bmatrix}$$

La terza colonna, che é necessaria per la moltiplicazione di matrici, non influenza i parametri della trasformazione; l' ultima colonna é sempre la stessa. Come per la matrice di trasformazione bidimensionale, solo le prime due colonne sono significative.

Quando l' HCRST riceve questi elementi, compie su ciascun vettore, a partire da quell' istante, una moltiplicazione di matrice cosicché dal vettore della display list $X\ Y\ Z\ 1$ risulti il vettore trasformato $X'\ Y'$:

$$[X'\ Y'] = [X\ Y\ Z\ 1]$$

Questo vettore trasformato é poi tagliato secondo la finestra data e ne viene fatto il display.

Gli elementi della matrice di trasformazione sono numeri a 16 bit, con segno. Gli elementi 2 e 5 sono interi. Gli elementi 0, 1, 3, 4, 6, 7 sono numeri reali in virgola fissa con il punto binario tra i bit quattro e cinque (il bit piu' significativo é il bit zero). I primi sei elementi sono usati in trasformazioni bidimensionali; tutti e otto sono usati in trasformazioni tridimensionali.

APPENDICE B

Listati delle subroutines di calcolo delle matrici di trasformazione

```
C=======================================================================
      subroutine  MTX3X3 (sclx, scly, rotz, trx, try, mt2)

C-----------------------------------------------------------------------
C
C     subr. per il calcolo della matrice di trasformazione per la roto-
C     traslazione dei segmenti bidimensionali intorno all' origine del
C     Megatek.
C
C     PARAMETRI :
C             sclx = fattore di scala in x
C             scly = fattore di scala in y
C             rotz = angolo di rotazione attorno all'asse z (in radianti)
C             trx  = traslazione in x
C             try  = traslazione in y
C
C-----------------------------------------------------------------------

      real mt2(3,3)

      cosz=cos(rotz)
      sinz=sin(rotz)

      mt2(1,1) =  sclx * cosz
      mt2(2,1) = -scly * sinz
      mt2(3,1) =  trx
      mt2(1,2) =  sclx * sinz
      mt2(2,2) =  scly * cosz
      mt2(3,2) =  try

      return
      end
```

```
C=====================================================================
      subroutine  MTX3X3G (sclx, scly, csclx, csc1y, rotz,
     *                     crotx, croty, trx, try, mt2)
C---------------------------------------------------------------------
C
C     subr. per il calcolo della matrice di trasformazione per la roto-
C     traslazione dei segmenti bidimensionali generale (intorno a punti
C     generici).
C
C     PARAMETRI :
C             sclx = fattore di scala in x
C             scly = fattore di scala in y
C             csx  = centro di scala in x
C             csy  = centro di scala in y
C             rotz = angolo di rotazione attorno all'asse z (in radianti)
C             crotx= centro di rotazione in x
C             croty= centro di rotazione in y
C             trx  = traslazione in x
C             try  = traslazione in y
C
C---------------------------------------------------------------------
      real mt2(3,3)

      cosz=cos(rotz)
      sinz=sin(rotz)

      mt2(1,1) =  sclx * cosz
      mt2(2,1) = -scly * sinz
      mt2(3,1) =  trx + crotx * cosz + croty * sinz -
     *            sclx * (crotx - csclx) - csclx
      mt2(1,2) =  sclx * sinz
      mt2(2,2) =  scly * cosz
      mt2(3,2) =  try  + csy - csy * scly
      mt2(3,2) =  try + crotx * (-sinz) + croty * cosz -
     *            scly * (croty - csc1y) - csc1y

      return
      end
```

```
C=======================================================================
      subroutine  MTX4X4
     *  (sclx, scly, sclz, rotx, roty, rotz, trx, try, trz, mt3)

C-----------------------------------------------------------------------
C
C     subr. per il calcolo della matrice di trasformazione per la roto-
C     traslazione dei segmenti tridimensionali intorno all'origine del
C     Megatek.
C
C     PARAMETRI :
C             sclx = fattore di scala in x
C             scly = fattore di scala in y
C             sclz = fattore di scala in z
C             rotx = angolo di rotazione attorno all'asse x (in radianti)
C             roty = angolo di rotazione attorno all'asse y (in radianti)
C             rotz = angolo di rotazione attorno all'asse z (in radianti)
C             trx  = traslazione in x
C             try  = traslazione in y
C             trz  = traslazione in z
C
C-----------------------------------------------------------------------

      real  mt3(4,4)

      cosx=cos(rotx)
      sinx=sin(rotx)

      cosy=cos(roty)
      siny=sin(roty)

      cosz=cos(rotz)
      sinz=sin(rotz)

      mt3(1,1) =  sclx * cosy * cosz
      mt3(2,1) = -scly * sinx * siny * cosz - scly * cosx * sinz
      mt3(3,1) = -sclz * cosx * siny * cosz + sclz * sinx * sinz
      mt3(4,1) =  (trx * cosy - (try * sinx + trz * cosx) * siny) * cosz
     $            - (try * cosx - trz * sinx) * sinz
      mt3(1,2) =  sclx * cosy * sinz
      mt3(2,2) = -scly * sinx * siny * sinz + scly * cosx * cosz
      mt3(3,2) = -sclz * cosx * siny * sinz - sclz * sinx * cosz
      mt3(4,2) =  (trx * cosy - (try * sinx + trz * cosx) * siny) * sinz
     $            + (try * cosx - trz * sinx) * cosz

      return
      end
```

```
C=====================================================================
      subroutine MTX4X4G (sclx, scly, sclz, csclx, cscly, csclz,
     * rotx, roty, rotz, crotx, croty, crotz, trx, try, trz, mt3)
C---------------------------------------------------------------------
C
C     Subr. per il calcolo della matrice di trasformazione per la roto-
C     traslazione dei segmenti tridimensionali generale (intorno a punti
C     generici.
C
C     PARAMETRI :
C             sclx  = fattore di scala in x
C             scly  = fattore di scala in y
C             sclz  = fattore di scala in z
C             csclx = centro di scala in x
C             cscly = centro di scala in y
C             csclz = centro di scala in z
C             rotx  = angolo di rotazione attorno all'asse x (in radianti)
C             roty  = angolo di rotazione attorno all'asse y (in radianti)
C             rotz  = angolo di rotazione attorno all'asse z (in radianti)
C             crotx = centro di rotazione in x
C             croty = centro di rotazione in y
C             crotz = centro di rotazione in z
C             trx   = traslazione in x
C             try   = traslazione in y
C             trz   = traslazione in z
C
C---------------------------------------------------------------------

      implicit real*4 (A-H,O-Z)
      real*4  mt3(4,4)

      cosx=cos(rotx)
      sinx=sin(rotx)

      cosy=cos(roty)
      siny=sin(roty)

      cosz=cos(rotz)
      sinz=sin(rotz)

C---------------------------------------------------------------------
C     Trasformazioni di trx, try, trz in funzione dei centri di scala
C     e dei centri di rotazione
C

      tx = trx - sclx * (crotx - csclx) - csclx !rel. al centro di scala in x
      ty = try - scly * (croty - cscly) - cscly !rel. al centro di scala in y
```

```
      tz = trz - sclz * (crotz - csclz) - csclz !rel. al centro di scala in z

c     relativamente al centro di rotazione

      x = cosy * cosz * crotx + cosy * sinz * croty + siny * crotz
      y = (-sinx * siny * cosz - cosx * sinz) * crotx + (-sinx *
     *  siny * sinz + cosx * cosz ) * croty + sinx * cosy * crotz
      z = (-cosx * siny * cosz + sinx * sinz) * crotx + (-cosx *
     *  siny * sinz- sinx * cosz) * croty + cosx * cosy * crotz

      tx = tx + x      ! relativamente al centro di rotazione in x
      ty = ty + y      ! relativamente al centro di rotazione in y
      tz = tz + z      ! relativamente al centro di rotazione in z
c-----------------------------------------------------------------------

      mt3(1,1) =  sclx * cosy * cosz
      mt3(2,1) = -scly * sinx * siny * cosz - scly * cosx * sinz
      mt3(3,1) = -sclz * cosx * siny * cosz + sclz * sinx * sinz
      mt3(4,1) =  (tx  * cosy - (ty  * sinx + tz  * cosx) * siny) * cosz
     $           - (ty * cosx - tz  * sinx) * sinz
      mt3(1,2) =  sclx * cosy * sinz
      mt3(2,2) = -scly * sinx * siny * sinz + scly * cosx * cosz
      mt3(3,2) = -sclz * cosx * siny * sinz - sclz * sinx * cosz
      mt3(4,2) =  (tx  * cosy - (ty  * sinx + tz  * cosx) * siny) * sinz
     $           + (ty * cosx - tz  * sinx) * cosz


      return
      end
```

```
c=============================================================================
      subroutine  readjoy (locsts, factn, dx, dy ,ibuton)
      integer*2  locsts, ibx, iby, ibuton
c     posizione del joystick rispetto ad x
      ibx = iibits ( locsts, 4, 3)+1  !estrae 3 bit a partire dal 4 da locsts
      go to (1,2,3,4,5,6,7,8), ibx
1     continue          ! negativo massimo
      dx=-factn
      go to 9
2     continue          ! negativo medio
      dx=-factn/2.
      go to 9
3     continue          ! negativo minimo
      dx=-factn/4.
      go to 9
4     continue          ! centro
      dx=0.
      go to 9
5     continue          ! centro
      dx=0.
      go to 9
6     continue          ! positivo minimo
      dx=factn/4.
      go to 9
7     continue          ! positivo medio
      dx=factn/2.
      go to 9
8     continue          ! positivo massimo
      dx=factn
c     posizione del joystick rispetto ad y
9     iby = iibits ( locsts, 1, 3)+1  !estrae 3 bit a partire dal 1 da locsts
      go to (11,12,13,14,15,16,17,18), iby
```

```
11      continue          ! negativo massimo
        dy=-factn
        go to 19

12      continue          ! negativo medio
        dy=-factn/2.
        go to 19

13      continue          ! negativo minimo
        dy=-factn/4.
        go to 19

14      continue          ! centro
        dy=0.
        go to 19

15      continue          ! centro
        dy=0.
        go to 19

16      continue          ! positivo minimo
        dy=factn/4.
        go to 19

17      continue          ! positivo medio
        dy=factn/2.
        go to 19

18      continue          ! positivo massimo
        dy=factn

19      ibuton = iibits (locsts, 0, 1)  ! estrae il bit 0 da locsts per verifi-
c                                       ! care se il bottone e' premuto.

        return
        end
```

```
c=============================================================================
      subroutine  readtab (locsts, ibut2, ibut1, ibut2 , ibut3, ibuton)

      implicit integer*2 (i-n)

      ibuton = iibits (locsts, 0, 1)  !estrae il bit 0 da locsts per
c                                     !verificare se qualche bottone e'
c                                     !premuto.

      if (ibuton.eq.0) return

      ibut1 = iibits (locsts, 6, 1)
      ibut2 = iibits (locsts, 5, 1)
      ibut3 = iibits (locsts, 4, 1)
      ibutz = iibits (locsts, 7, 1)

      return
      end
```

```
Comment=====================================================================
Comment
                SUBROUTINE WAIT (TIME)
Comment         ======================
Comment
Comment
Comment=====================================================================

        CHARACTER*20 TIME
        INTEGER ITIME(2)
        INTEGER SYS$HIBER, SYS$SCHDWK ,SYS$BINTIM


Comment         trasforma la stringa inserita in tempo binario (64 bits)

        I = SYS$BINTIM (TIME,ITIME)
        IF (I.NE.1) GO TO 10

        I = SYS$SCHDWK(, , ITIME, )
        I = SYS$HIBER()

        RETURN

10      STOP 'Il tempo inserito e'' sintatticamente errato'

        END
```

APPENDICE C

Esempi di utlizzo dei periferici MEGATEK tramite l' uso della libreria MAXROB

```
C*************************************************************************
      Program  MENU
C     =======  ====
C
C     Descrizione:
C     Questo programma disegna un rettangolo di base 500  e altezza 300 pixels
C     nel primo quadrante del sistema di assi XY del MEGATEK; quindi controlla
C     se l' eco della Tablet (definito come un cerchietto di raggio  30 pixels
C     dalla subroutine 2) e' compreso in tale rettangolo. Se no, va a rilegge-
C     la tablet dopo un tempo di attesa di un secondo; se si, controlla se  e'
C     premuto uno dei 4 pulsanti del mouse. Nel caso che non sia premuto  nes-
C     suno dei 4 pulsanti, torna a leggere la Tablet; nel caso sia premuto  il
C     pulsante 3 termina l' esecuzione del programma;  negli  altri 3 casi,  il
C     programma effettuerebbe operazioni distinte.
C
C*************************************************************************

      implicit integer*2 ( i-n )
      character*20  deltatime

      real*4  mt3(4,4)

      data  pig / 3.14159 /
      deltatime = '0 00:00:01.00'      ! 1 secondo

      call  mwbufr (2)                 ! flush buffer
      call  mwinit ('WAND$CONFIG')     ! inizializza la grafica
      call  mwslct (1)                 ! seleziona il display 1
      call  mwalta (1, loctab )        ! alloca la tablet

      call  mwopen ('SEGMENT', 1)      ! apre un segmento
      call  mwcvty (1)                 ! coordinate assolute
      call  mw2mov (1000, 1000)        ! muove a 1000, 1000
      call  mw2drw (1500, 1000)        ! disegna un rettangolo
      call  mw2drw (1500, 1300)        ! di base 500 e
      call  mw2drw (1000, 1300)        ! altezza 300:      1000<= X <=1500
      call  mw2drw (1000, 1000)        !                   1000<= Y <=1300
      call  mwclos                     ! chiude il segmento
```

```
C-------------------------------------------------------------------------
C       subroutine che disegna un cerchietto utilizzato come cursore
C       grafico
C
        call  mwopen ('SUBROUTINE', 2)          ! apre una subroutine
        call  mwcapp ( 9 )                      ! set del colore a 9
        call  mwcvty ( 8 )                      ! coordinate relative
        ir = 30
        call  mw2mov (ir,0)
        ixo = ir
        iyo = 0
        do  i=1,360
         angle = i*pig/180.
         ix = ir*cos(angle)
         iy = ir*sin(angle)
         idx= ix - ixo
         idy= iy - iyo
         call  mw2drw ( idx,idy )               ! traccia un vettore da
         ixo = ix                               ! ixo, iyo  a  ix, iy
         iyo = iy
        end do
        call  mwclos                            ! chiude la subroutine
C-------------------------------------------------------------------------
C-------------------------------------------------------------------------
C       parte del programma che definisce l'eco della tablet
C
        call  mwcvty (1)                    ! x,y,z assoluti
        call  mweccr (loctab,2)             ! set subroutine 2 come eco della tab.
        call  mwecho (loctab,2)             ! tipo di eco user-defined.
C       call  mwecps (loctab,0,0)           ! posizione iniziale dell'eco
        call  mwlcwd (loctab,-2000,2000,-2000,2000) ! window di eco
        call  mwenlc (loctab, 0, 1)         ! abilita il device per l'input.

C-------------------------------------------------------------------------
C       loop di lettura della tablet

1       call  mwsmlc (loctab, ix, iy, ilcsts)    ! legge la tablet

C       controlla che l'eco della tablet sia compreso nel rettangolo
C       disegnato (col segmento 1)

        if (ix.lt.1000.or.ix.gt.1500.or.iy.lt.1000.or.iy.gt.1300) then
C         non e' nel rettangolo
          call  wait (deltatime)                ! aspetta un secondo
        else
C         e' nel rettangolo; controlla se e' premuto qualche bottone sul mouse
          call  readtab (ilcsts, ibut2, ibut1, ibut2, ibut3, ibuton)
          if (ibuton.ne.1) then             ! nessun bottone del mouse e' premuto
```

```
            call  wait (deltatime)      ! aspetta un secondo
          else if (ibutZ.eq.1) then     ! e' premuto il bottone Z
c               ...................
c               ...................
          else if (ibut1.eq.1) then     ! e' premuto il bottone 1
c               ...................
c               ...................
          else if (ibut2.eq.1) then     ! e' premuto il bottone 2
c               ...................
c               ...................
          else if (ibut3.eq.1) then     ! e' premuto il bottone 3
            go to 2
          end if
        end if
        go to 1

2       continue          ! EXIT
        call  mwdall ('SEGMENT')        ! cancella il display (tutti i segm.)
        call  mwterm                    ! termina la grafica
        call exit                       ! termina esecuzione programma.
        end
```

```
C*******************************************************************************
      Program  CUBO
C     =======  ====
C
C
C     Descrizione:
C     Questo programma disegna un  cubo di lato 2000 pixels,  al centro  dello
C     schermo del MEGATEK nel segmento 100.  Quindi interroga il joystick fa-
C     cendo effettuare, qualora il pulsante del joystick venga tenuto premuto,
C     una rotazione al cubo intorno all' asse X ed intorno all' asse Y con ve-
C     locita' proporzionale alla deflessione in X e in Y del joystick  stesso.
C     L' esecuzione del programma  termina dopo 30 secondi circa  dal rilascio
C     del pulsante del joystick.
C
C*******************************************************************************

      implicit integer*2 ( i-n )
      character*20  deltatime

      real*4  mt3(4,4)

      data  mt3 / 16*0. /
      data  pig / 3.14159 /
      deltatime = '0 00:00:01.00'      ! 1 secondo

      nloop=0

      call  mwbufr (2)                 ! flush buffer
      call  mwinit ('WAND$CONFIG')     ! inizializza la grafica
      call  mwslct (1)                 ! seleziona il display 1

      call  mwaljo (1, locjoy)         ! alloca il joystick
      call  mwchdr (1)                 ! abilita la trasformabilita' dei segm.

C-------------------------------------------------------------------------------
C     Disegna un cubo nel segm 100

      call  mwopen ('SEGMENT', 100)    ! apre il segmento 100
      call  mwcvty ( 1 )               ! coordinate assolute
      call  mw3mov( 1000, 1000, 1000) ! muove il cursore grafico

      call  mw3drw(-1000, 1000, 1000) !
      call  mw3drw(-1000,-1000, 1000) ! Disegna la base quadrata anteriore
      call  mw3drw( 1000,-1000, 1000) !
      call  mw3drw( 1000, 1000, 1000) !
```

```
      call  mw3mov( 1000, 1000,-1000) ! muove il cursore grafico

      call  mw3drw(-1000, 1000,-1000) !
      call  mw3drw(-1000,-1000,-1000) ! Disegna la base quadrata posteriore
      call  mw3drw( 1000,-1000,-1000) !
      call  mw3drw( 1000, 1000,-1000) !

      call  mw3mov( 1000, 1000, 1000) !
      call  mw3drw( 1000, 1000,-1000) !
      call  mw3mov(-1000, 1000, 1000) ! Unisce i vertici delle due
      call  mw3drw(-1000, 1000,-1000) ! basi quadrate
      call  mw3mov(-1000,-1000, 1000) !
      call  mw3drw(-1000,-1000,-1000) !
      call  mw3mov( 1000,-1000, 1000) !
      call  mw3drw( 1000,-1000,-1000) !

      call  mwclos                    ! chiude il segmento
c-----------------------------------------------------------------------

      call  mwenlc (locjoy, 0, 0)     ! abilita il joystick per l' input

5     call  mwsmlc (locjoy, ix, iy, ilcsts)          ! legge il joy
      factn=pig/180.                                 ! un grado
      call  readjoy (ilcsts, factn, dx, dy, ibuton)  ! ne cod la posiz.

c     controlla se e' premuto il bottone
      if (ibuton.eq.0) then
c           Il bottone non 'e premuto: aspetta un secondo e cicla
            call  wait (deltatime)
            nloop=nloop+1                    !dopo 30 secondi di inattivita'
            if (nloop.eq.30) go to 2         !termina l'esecuzione del prog.
      else
c       Il bottone e' premuto: esegue la rotazione del cubo
        nloop=0
        angz=0.
        angx=angx+dy
        angy=angy+dx

        call  mtx4x4g (1.,1.,1.,0.,0.,0.,angx,angy,angz,0.,0.,0.,
     *                 0.,0.,0.,mt3)
        call  mw3smt(100,mt3)          ! trasforma il segmento TSEGM

      end if

      go to 5

2     continue          ! EXIT
      call  mwdall ('SEGMENT')         ! cancella il display (tutti i segm.)


      call  mwterm                     ! termina la grafica
      call exit                        ! termina esecuzione programma.
      end
```