UN APPROCCIO APL ALLA REALIZZAZIONE DI UN SISTEMA GRAFICO DI BASE

G. Faconti

P. Palamidese

Rapporto interno C82-3

Istituto CNUCE

PISA

1. INTRODUZIONE

Questa relazione si propone di illustrare le esperienze maturate durante l'implementazione, in linguaggio APL, di un software grafico 'CORE-like'.

Il termine CORE e' usato per tutti quei sistemi grafici che aderiscono ai principi enunciati nel 1976 a Seillac in Francia nel corso dello 'Workshop on Graphics Standards Methodology'.

Le idee ed i concetti presentati a Seillac sono stati successivamente oggetto di attenzione da parte dello Special Interest Group on Graphics (SIGGRAPH) in ambito ACM che, attraverso la costituzione del Graphics Standard Planning Commetee (GSPC), ha formulato una proposta (CORE Graphics System) per la standardizzazione dei sistemi grafici. La proposta e' stata pubblicata su di un numero speciale della rivista 'Computer Graphics' nel 1979.

I concetti fondamentali su cui si basa la proposta CORE possono essere cosi' riassunti:

- la portabilita' dei programmi e' lo scopo fondamentale nella definizione di uno standard,
- occorre considerare con la massima attenzione tutte quelle risoluzioni che influiscono sulla portabilita' e sulla struttura dei programmi applicativi,
- la metodologia seguita sia per lo sviluppo che per l'utilizzo di un sistema standard e' di per se tanto importante quanto la definizione delle sue caratteristiche funzionali,
- le funzioni di definizione e manipolazione di oggetti ('modelling system') e quelle di visualizzazione degli oggetti stessi ('viewing system') devono essere separate.

In questo senso il CORE Graphics System e' un sistema per la visualizzazione e l'interazione con oggetti grafici definiti in un qualche modello astratto.

I seguenti capitoli trattano alcuni aspetti della implementazione di un sistema aderente alla proposta di standard del 1979. Nel capitolo 2 si prende in esame il processo di generazione dell'immagine e si discutono gli algoritmi utilizzati per risolvere sia problemi bidimensionali che tridimensionali. Il capitolo 3 contiene la descrizione del sistema APLCORE che implementa funzioni per disegni in due dimensioni, con possibilita' di segmentazione,definizione e modifica di attributi, input e output sulla Stazione Grafica IBM 3277. Infine il capitolo 4 presenta le caratteristiche peculiari dell'hardware grafico utilizzato.

2. IL SISTEMA GRAFICO DI BASE

2.1 Portabilita'

Un programma applicativo 'portabile' deve essere indipendente dal particolare sistema hardware/software su cui e' implementato in modo da evitare onerose modifiche quando si passa da un sistema ad un altro. Per raggiungere questo risultato il package grafico deve presentare al programmatore un insieme di funzioni per la creazione dell'immagine che non dipendono dal dispositivo di output, sia questo un plotter, uno schermo di tipo 'storage' o a rinfresco.

Per descrivere questo tipo di sistema, schematizzato in FIGURA 1, si usa il termine 'device independent'. La sola parte 'device dependent' e' il modulo incaricato della gestione del dispositivo grafico utilizzato.

Il sistema CORE si presenta come una interfaccia standard tra l'applicazione e i 'devices' ed e' disegnata in modo tale che l'indipendenza dall'hardware sia ottenuta a vari livelli:

indipendenza per il programmatore cioe' al livello di interfaccia tra applicazione e sistema grafico.

indipendenza all'interno del sistema grafico per permettere al sistema di supportare diversi 'devices' con il minimo di modifiche.

indipendenza dal calcolatore perche' il sistema possa funzionare su calcolatori diversi.

2.2 Modularita'

Poiche' le necessita' dei programmi applicativi e le caratteristiche dell'hardware sono molto diversificate, non e' opportuno realizzare un package grafico monolitico che soddisfi tutte le possibili richieste. Il SIGGRAPH propone la realizzazione di un sistema costituito da diversi moduli in modo tale che solo al piu' alto livello tutte le funzioni siano supportate, mentre ai livelli inferiori siano in parte omesse.

Il problema piu' importante a questo punto e' quello di stabilire quali moduli sono necessari e quali funzioni essi debbano contenere.

La prima fondamentale distinzione e' quella tra le funzioni per il modello e le funzioni grafiche. Le prime servono per costruire e manipolare gli oggetti che si vogliono disegnare, mentre le seconde sono strettamente legate al processo di generazione dell'immagine. Il vantaggio principale di questa distinzione e' che il modulo per il modello non e' necessario ad una applicazione che

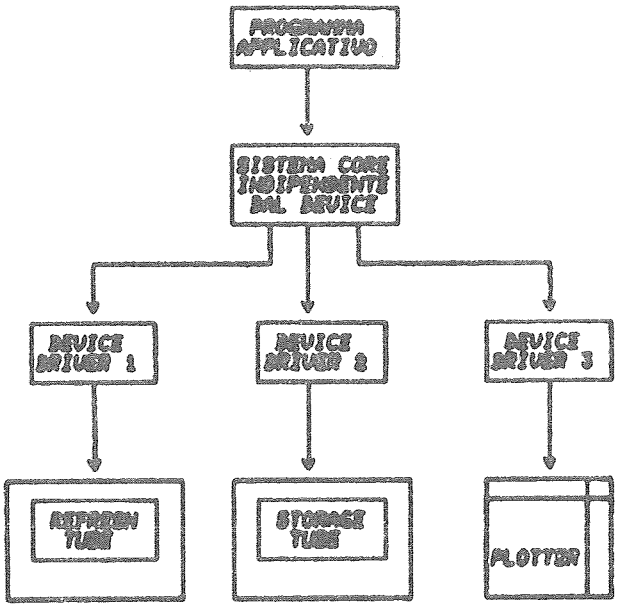


FIGURA 1. SISTEMA GRAFICO INDIPENDENTE DAL DEVICE.

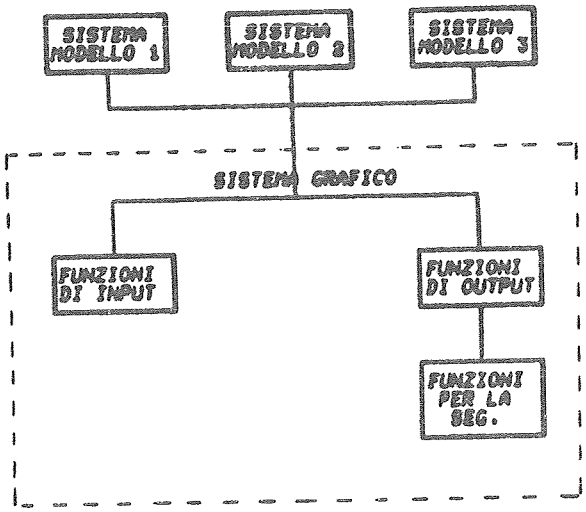


FIGURA 2. SISTEMA MODELLO E SISTEMA GRAFICO

vuole solo generare immagini.

Naturalmente e' difficile definire un 'sistema modello' di base in quanto e' fortemente dipendente dall'applicazione, mentre e' possibile disegnare un solo 'sistema grafico' standard.

Anche il 'sistema grafico' e' strutturato come un insieme di moduli realizzati tenendo conto del fatto che esiste una gran varieta' di 'devices' grafici e che in molti casi l'applicazione non ha bisogno di avere a disposizione il sistema CORE completo.

La struttura modulare del CORE e illustrata in FIGURA 2.

All'interno del sistema grafico sono messi in evidenza tre livelli fondamentali: quello delle primitive di output utilizzate da tutte le applicazioni, quello delle funzioni per la segmentazione non necessarie per semplici presentazioni grafiche su plotter e le funzioni di input utilizzate solo da applicazioni interattive.

Il nucleo del livello di output e' costituito dalle trasformazioni di viewing di cui trattiamo nei prossimi paragrafi.

2.3 Trasformazioni di Viewing

In questo paragrafo prendiamo in considerazione solo quelle trasformazioni che si riferiscono ai problemi in due dimensioni lasciando al paragrafo 2.4 una breve descrizione delle trasformazioni da applicare ad oggetti definiti nello spazio tridimensionale.

Per descrivere le fasi logiche attraverso cui il sistema grafico rappresenta una immagine su un dispositivo di output, e' diffusa in letteratura l'analogia con la rappresentazione fotografica degli oggetti.

Noi abbiamo l'impressione di guardare attraverso ad una finestra (Window) una porzione limitata del mondo. Gli oggetti che vediamo possono essere descritti in un sistema di coordinate qualsiasi (World Coordinate System), la macchina fotografica puo' essere ruotata rispetto alla Window (View-up) e si possono effettuare anche operazioni di scalatura e traslazione (World to World Transformations). Per ogni oggetto occorre separare la parte visibile da quella invisibile e scartare quindi le linee che cadono al di fuori della Window (Clipping). Infine bisogna definire la porzione di superficie del dispositivo grafico (Viewport) su cui viene proiettato il contenuto della Window (vedi FIGURA 3).

2.3.1 Trasformazioni Geometriche

L'utente puo' pensare il proprio problema nel suo universo che e' lo spazio reale bi o tridimensionale e

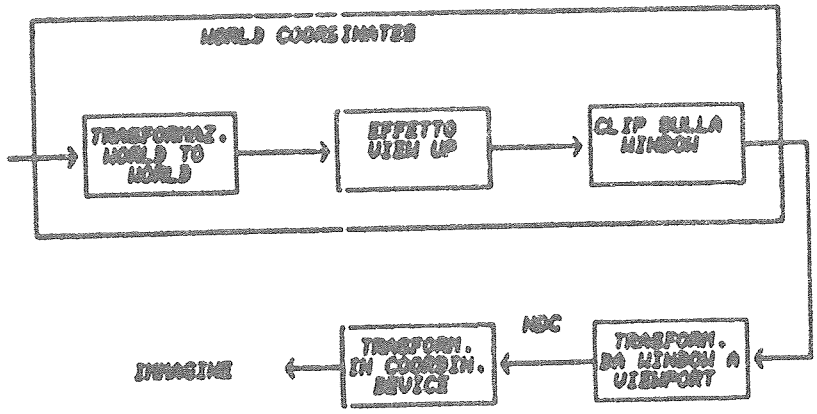


FIGURA 3. PROCESSO DI GENERAZIONE DELL'IMMAGINE.

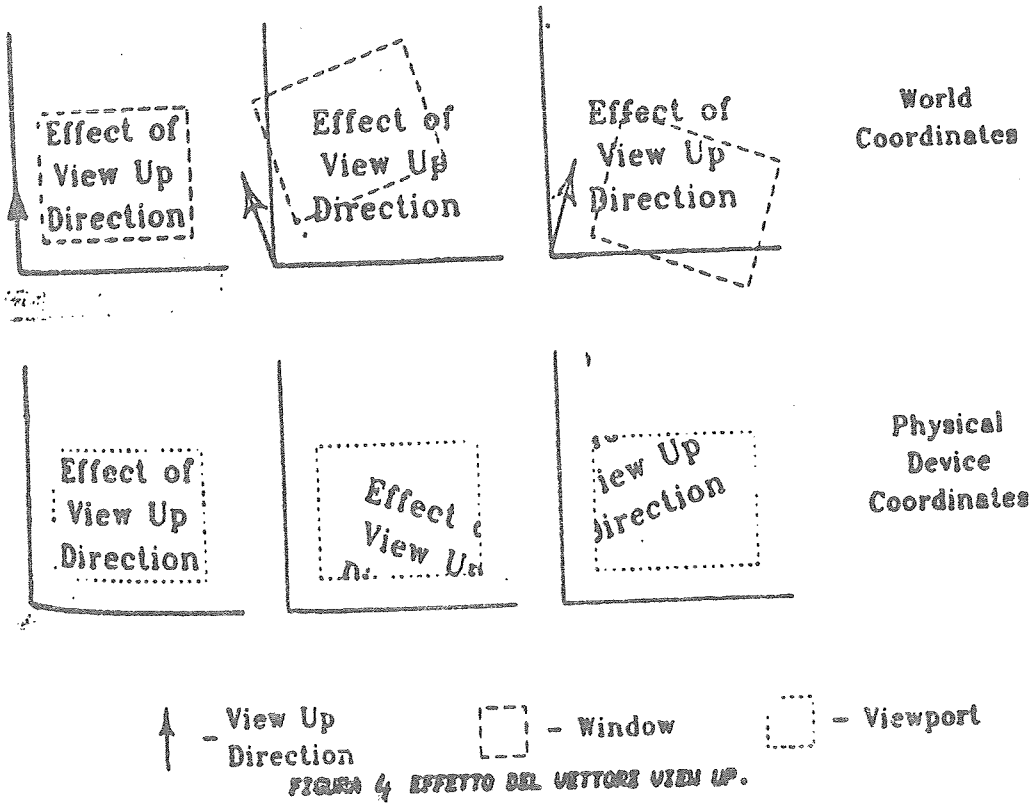


FIGURA 4 EFFETTO DEL VETTORE VIEW UP.

l'elaboratore traccia l'immagine su un supporto di dimensioni limitate (schermo o carta del plotter).

Il problema della creazione dell'immagine e' visto dall'elaboratore come un problema geometrico di trasformazione di coordinate. Ogni figura dello spazio reale e' percio' vista come un insieme di punti dello spazio R^2 (R^3) e ogni trasformazione applicata ai punti della figura F produce la figura trasformata F'. Le trasformazioni che usiamo nel seguito sono trasformazioni lineari e pertanto soddisfano alle seguenti: sono invertibili e possono essere descritte da matrici in R^3 permettendo una facile composizione di trasformazioni come semplice prodotto di matrici.

2.3.2 Coordinate Omogenee

Per poter rappresentare con una sola matrice trasformazioni diverse come rotazioni e traslazioni e' necessario trattare il problema nello spazio proiettivo con coordinate omogenee.

Ricordiamo che un punto P di coordinate (x,y) in R^2 viene rappresentato in coordinate omogenee dall'insieme di valori reali (X,Y,H). Se vogliamo che anche il punto trasformato P* sia espresso in coordinate omogenee la matrice T di trasformazione deve essere di tre righe e tre colonne. Questo ci permette di calcolare anche la sua inversa dato che T e' quadrata.

Si osserva quindi che in coordinate omogenee la trasformazione di vettori n-dimensionali e' effettuata in uno spazio n+1-dimensionale e i risultati n-dimensionali sono ottenuti proiettando nel particolare spazio n-dimensionale che interessa. Cioe' la relazione per passare da un punto in coordinate omogenee (X,Y,H) al corrispondente punto (x,y) in R^2 e' la seguente : (x,y) = (X/H,Y/H).

Ricordiamo che la rotazione attorno all'origine di un angolo A e' data dalla trasformazione

$$\begin{vmatrix} \cos A & -\text{sen } A & 0 \\ \text{sen } A & \cos A & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Una trasformazione di scala secondo i valori (Sx,Sy) e' rappresentata dalla matrice

$$\begin{vmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

La traslazione di Tx e Ty rispettivamente nelle direzioni x e y e' data dalla matrice

0	0	0
0	0	0
Tx	Ty	1

Si puo' quindi concludere che la matrice di trasformazione puo' essere considerata divisa nelle seguenti zone

rotazione scala	0 0
traslazione	1

2.3.3 Trasformazioni World to World

L'utente del sistema grafico puo' trasformare i propri oggetti prima di rappresentarli sul dispositivo di output. Per esempio puo' ingrandire l'immagine in modo da rendere piu' chiari i dettagli o viceversa puo' rimpicciolirla per avere visibili un maggior numero di oggetti.

Ogni trasformazione e' tale da portare il punto (x,y) del disegno originale nel punto (x',y') dello stesso sistema di coordinate. Per esempio queste trasformazioni sono utili quando vengono applicate a simboli definiti secondo una unita' di misura di base e trasformati di volta in volta secondo le esigenze dell'applicazione. Questo metodo viene comunemente utilizzato per la creazione di menu di simboli.

La trasformazione voluta puo' essere complessa e data dalla composizione di piu' trasformazioni elementari (rotazione,traslazione,scala). Naturalmente il risultato dipende dall'ordine con cui vengono eseguite le trasformazioni. La funzione

SET_WORLD_COORDINATE_MATRIX_2 (Sx,Sy,A,Tx,Ty)

costruisce la matrice di trasformazione del tipo

$$MATRIX\_2 = \begin{vmatrix} a & b & 0 \\ d & e & 0 \\ g & h & 1 \end{vmatrix}$$

con i seguenti valori per i parametri

$$\begin{aligned} a &= (Sx)\cos A \\ b &= -\text{sen } A \\ d &= \text{sen } A \\ e &= (Sy)\cos A \\ g &= Tx \\ h &= Ty \end{aligned}$$

La matrice MATRIX_2 esegue la sequenza delle trasformazioni con l seguente ordine: scalatura,rotazione attorno all'origine di un angolo A e traslazione.

Sempre restando nel mondo reale abbiamo visto nel primo paragrafo che si puo' ruotare la macchina fotografica per ottenere degli effetti come quello di FIGURA 4. La freccia indica la direzione in cui viene ruotata la macchina e quindi la Window. Dal punto di vista geometrico indicando con B l'angolo di rotazione applichiamo alle coordinate degli oggetti una rotazione -B mediante la trasformazione contenuta nella matrice MATRIX_VIEW_UP_2

$$MATRIX\_VIEW\_UP\_2 = \begin{vmatrix} \cos(-B) & -\text{sen}(-B) & 0 \\ \text{sen}(-B) & \cos B & 0 \\ 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} \cos B & \text{sen } B & 0 \\ -\text{sen } B & \cos B & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

2.3.4 Clipping

Per rappresentare un disegno bisogna definire quali fra gli oggetti visibili devono essere inclusi nell'immagine. Puo' accadere che alcune linee giacciano in parte nell'area visibile ed in parte in quella invisibile. L'unico modo corretto per rappresentare l'immagine e' quello di eseguire una operazione di clipping cioe' un processo mediante cui ogni elemento del disegno e' diviso nelle sue linee visibili e invisibili e le parti invisibili vengono scartate.

Il clipping puo' essere applicato a vari elementi della figura: punti,vettori,curve di vario tipo,poligoni. In ogni caso si tratta di risolvere delle disuguaglianze per stabilire se il punto (x,y) e' visibile

xmin < x < xmax ymin < y < ymax

dove xmin,xmax,ymin,ymax sono gli estremi della Window.
Tra i diversi algoritmi sviluppati a tale scopo esaminiamo l'algoritmo di clipping per segmenti di retta ideato da Dan Cohen e Ivan Sutherland. L'algoritmo e' diviso in due parti. Nella prima si stabilisce se la linea giace interamente entro la Window o se puo' essere completamente eliminata. Se non soddisfa nessuna di queste condizioni allora viene divisa in due parti e i due tests precedenti vengono applicati a queste due parti. Allora l'algoritmo si basa sul fatto che una linea o e' interamente visibile o puo' essere divisa in modo che una parte si possa eliminare. Consideriamo la FIGURA 5

1001	1000	1010
0001	0000	0010
0101	0100	0110

FIGURA 5

dove il piano e' diviso in nove regioni ciascuna identificata da un codice binario costruito com le seguenti regole

bit	posizione del punto rispetto alla Window
1	a sinistra
2	a destra
3	sotto
4	sopra

Chiaramente se i codici per gli estremi del segmento sono 0,la linea e' interamente all'interno della Window ,se l'intersezione logica dei due codici non e' 0 la linea giace interamente fuori Window.
L'algoritmo e' stato implementato in APL come in FIGURA 6.

2.3.5 Trasformazione World to NDC

Abbiamo visto che usando le trasformazioni si puo' definire un disegno nel sistema di coordinate piu' opportuno e isolare la parte di scena visibile all'interno della

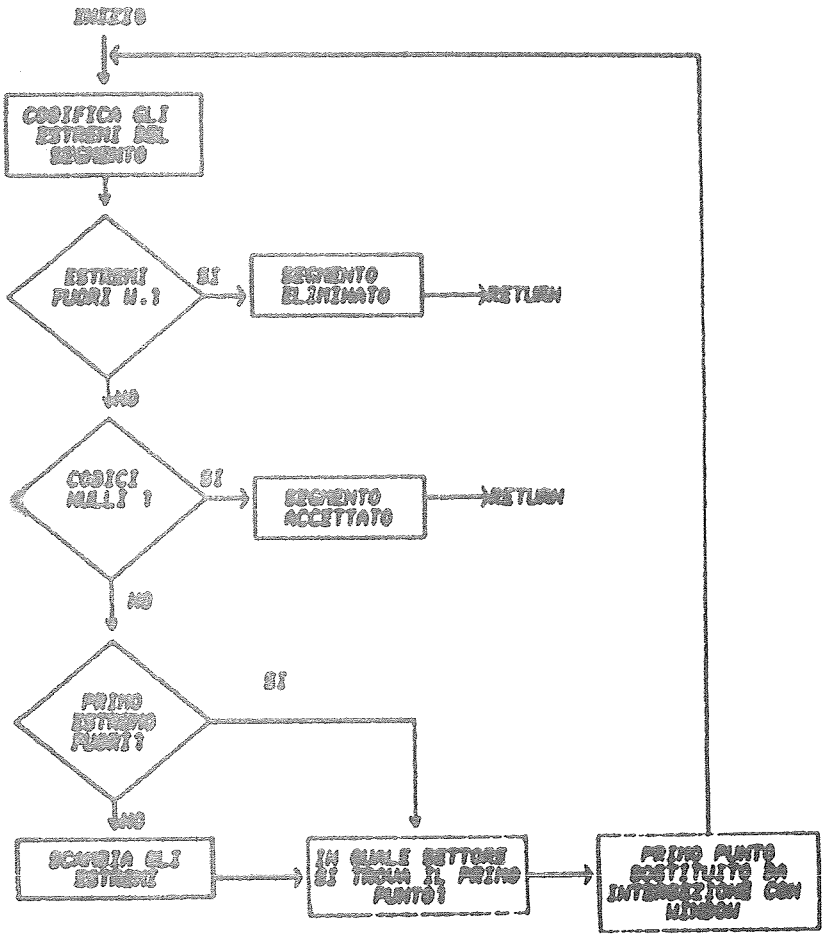


FIGURA 6. ALGORITMO DI CLIPPING

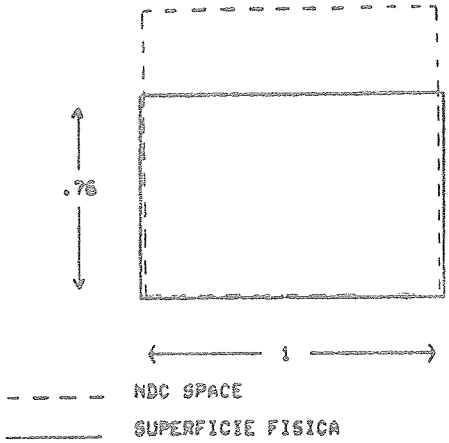


FIGURA 7. MAPPING DELLO SPAZIO NDC SULLA SUPERFICIE DI OUTPUT

Window. Si introduce a questo punto il Sistema di Coordinate Normalizzate NDC system) per definire una superficie virtuale su cui rappresentare l'immagine. Le coordinate normalizzate possono assumere valori da 0 a 1 nelle direzioni x e y e servono per descrivere le dimensioni dello schermo.
In questo ultimo sistema di coordinate e' stato definito lo spazio schermo (NDC space) per la stazione grafica IBM 3277 come rappresentato in FIGURA 7. Si e' preferito scegliere uguale ad 1 la dimensione lungo l'asse x e calcolare di conseguenza la dimensione lungo y tenendo presente che il rapporto tra le dimensioni dello schermo e' 4/3.

La scena descritta all'interno della Window viene riprodotta su una porzione della 'view surface' chiamata Viewport E' spesso utile specificare un Viewport di dimensioni inferiori a quelle dello schermo per lasciare lo spazio a menu di comandi o messaggi, o definire piu' Viewport sullo stesso schermo.

Si usa quindi la Window per definire cosa si vuole disegnare e il Viewport per dire dove lo si vuole rappresentare. Generalmente per evitare distorsioni si ha cura di scegliere Window e Viewport con le stesse dimensioni.

I vertici della Window sono contenuti nella variabile W e quelli del Viewport nella variabile V. Allora il punto (x,y)[w] espresso in coordinate reali viene trasformato nel punto (x,y)[v] in coordinate normalizzate dalla trasformazione

$$(x,y,1)[v] = (x,y,1)[w] \begin{vmatrix} a & 0 & 0 \\ 0 & c & 0 \\ b & d & 1 \end{vmatrix} \quad \text{MATRIX\_WORLD\_TO\_NDC\_2}$$

con i seguenti valori per i parametri della matrice di trasformazione MATRIX_WORLD_TO_NDC_2

```
a = (-/V[2 1]) :-/W[2 1]
c = (-/V[4 3]) :-/W[4 3]
b = V[1]
d = V[3]
```

Le coordinate di schermo si ottengono moltiplicando per 4096 cioe' per il numero di punti lungo la direzione x dello schermo.

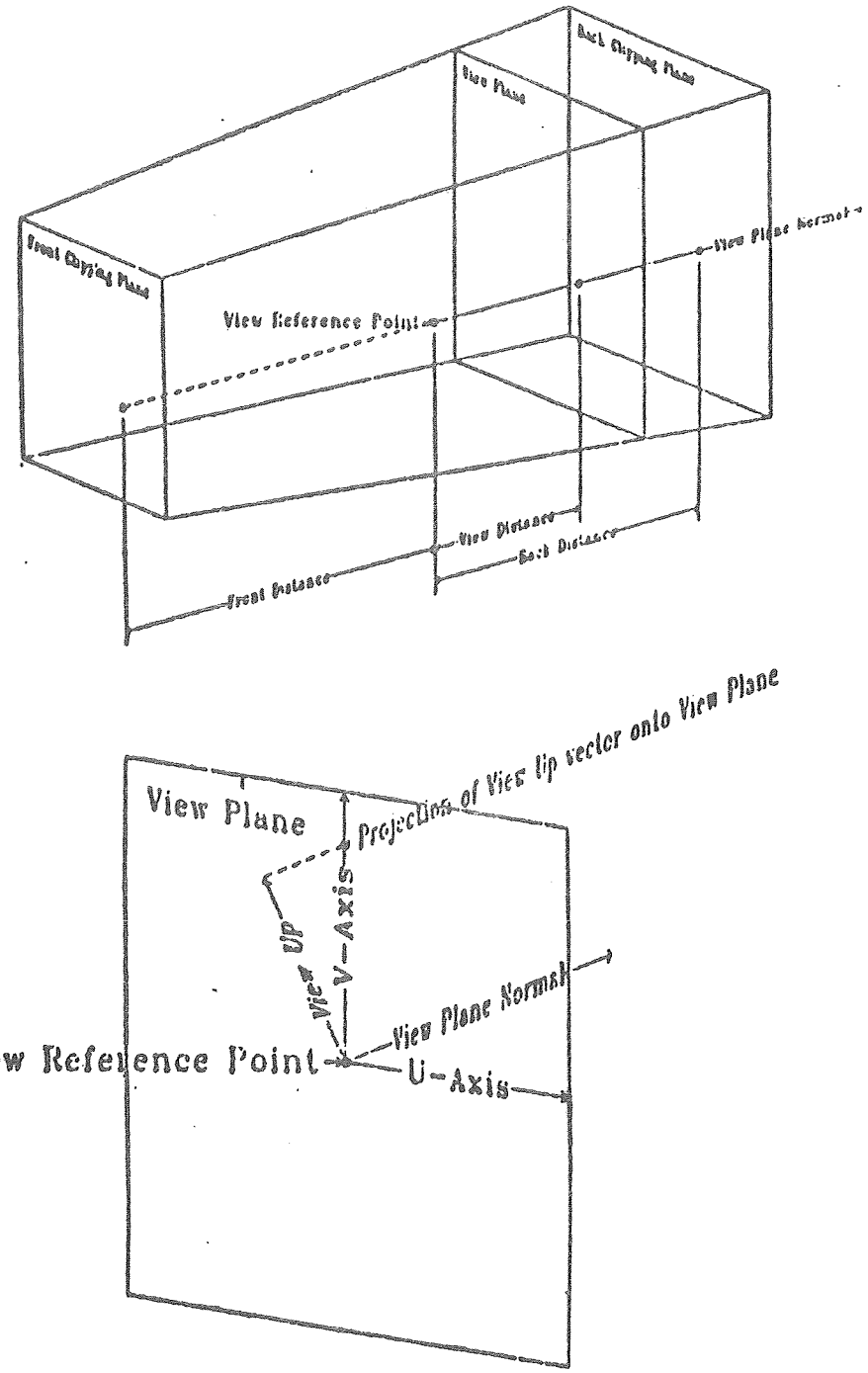


FIGURA 8. PARAMETRI PER LE PROIEZIONI

2.4 Proiezioni Geometriche Planari

Anche per la rappresentazione di oggetti in tre dimensioni vale l'analogia con macchina fotografica che proietta nelle due dimensioni gli oggetti rappresentati nello spazio tridimensionale.

Una proiezione e' definita dai seguenti parametri illustrati in FIGURA 8 :

Projection Type

Center of Projection o Direction of Projection (CPR)

View Plane Normal (VPN), View Plane Distance (VPD), View Reference Point (VRP) che definiscono il View Plane

Front Distance (FD), Back Distance (BD) per il clipping di profondita'

Window sul View Plane

Viewport sulla View Surface

View Up

Tutti i parametri sono definiti relativi al View Reference Point. Il piano di proiezione e' perpedicolare al View Plane Normal, passa alla distanza View Distance dal View Reference Point. View Distance e' positiva se misurata nella direzione del View Plane Normal.

La direzione verticale del View Plane e' definita dal vettore View Up. Questo vettore proiettato sul piano determina un nuovo sistema di riferimento indicato come UV. L'asse verticale V e' la proiezione ortogonale del vettore View Up, l'asse orizzontale U e' il prodotto vettoriale VPN x VUP.

Usando diversi tipi di prospettiva si possono ottenere diversi effetti prospettici. In particolare quando il centro di proiezione e' all'infinito si ha una proiezione parallela, quando e' al finito si ha una proiezione prospettica. Questi due tipi di proiezioni hanno ulteriori sottoclassificazioni come illustrato in FIGURA 9.

2.4.1 Le Matrici di Proiezione

La proiezione puo' essere rappresentata da una matrice 4 x 4. Nella descrizione teorica si discute sulla scelta e la definizione del piano di proiezione che puo' avere una orientazione qualsisi nello spazio. Dal punto di vista matematico e' piu' semplice eseguire la proiezione su uno

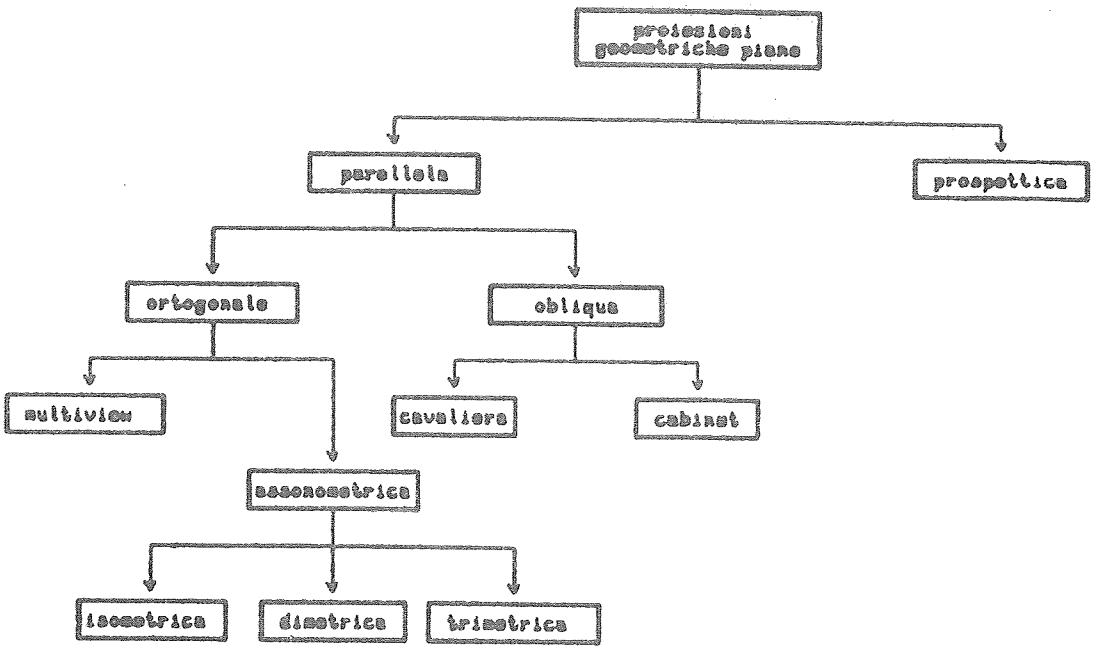


FIGURA 9. PROIEZIONI GEOMETRICHE PIANE

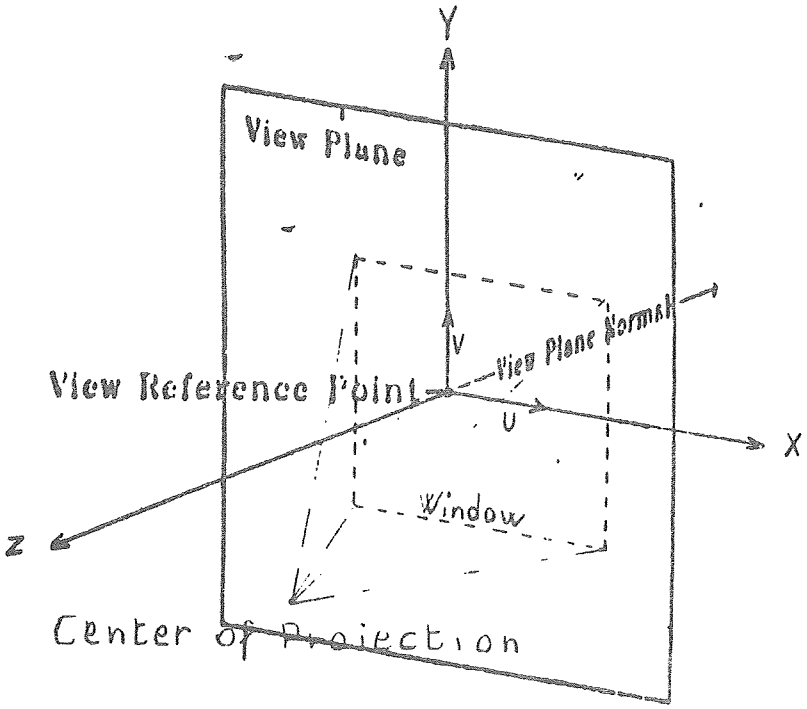


FIGURA 10. VALORI DI DEFAULT

dei piani ortogonali e pertanto e' consuetudine applicare delle trasformazioni all'oggetto in modo da riportare la scena nella situazione piu' favorevole. Nel seguito ci riporteremo sempre alle condizioni di FIGURA 10 dove il piano di proiezione e' il piano XY e la direzione di proiezione e' quella delle Z negative.

Prima di procedere ricordiamo che per eseguire delle rotazioni in senso positivo attorno agli assi principali si applicano le seguenti matrici:

rotazione positiva attorno all'asse X

$$\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos A & \text{sen } A & 0 \\ 0 & -\text{sen } A & \cos A & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

rotazione positiva attorno all'asse Y

$$\begin{vmatrix} \cos A & 0 & -\text{sen } A & 0 \\ 0 & 1 & 0 & 0 \\ \text{sen } A & 0 & \cos A & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

rotazione positiva attorno all'asse Z

$$\begin{vmatrix} \cos A & \text{sen } A & 0 & 0 \\ -\text{sen } A & \cos A & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

La proiezione di un oggetto puo' essere vista come la successione dei seguenti passi:

1. trasla l'origine nel VRP[Vx,Vy,Vz]

$$\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -Vx & -Vy & -Vz & 1 \end{vmatrix}$$

2. ruota il VPN attorno all'asse Y in modo che coincida col piano YZ dalla parte delle Z negative (A = 180 - a)

$$\begin{vmatrix} \cos A & 0 & -\text{sen } A & 0 \\ 0 & 1 & 0 & 0 \\ \text{sen } A & 0 & \cos A & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} -\cos a & 0 & -\text{sen } a & 0 \\ 0 & 1 & 0 & 0 \\ \text{sen } a & 0 & -\cos a & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

3. ruota il VPN attorno all'asse X in modo che giaccia lungo

l'asse Z (B = -b)

$$\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos B & \text{sen } B & 0 \\ 0 & -\text{sen } B & \cos B & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos b & -\text{sen } b & 0 \\ 0 & \text{sen } b & \cos b & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

4. ruota il vettore VUP attorno all'asse Z in modo che giaccia sul piano YZ nella direzione delle Y positive (C = 90 - c)

$$\begin{vmatrix} \cos C & \text{sen } C & 0 & 1 \\ -\text{sen } C & \cos C & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} \text{sen } c & \cos c & 0 & 1 \\ -\cos c & \text{sen } c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Indicando con V la matrice prodotto delle trasformazioni precedenti ricaviamo l'oggetto trasformato:

$$(X,Y,Z,W) \leftarrow\!\!- (x,y,z,w) +.\times V$$

5. applica il clipping di profondita' sui piani Front Plane e Back Plane (vedi rif. 4 pag. 360)

6. esegui i passi successivi in funzione del tipo di proiezione

<u>proiezione ortogonale</u>

- moltiplica per la matrice

$$\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

<u>proiezione prospettica</u>

- applica V al centro di proiezione

$$CPR \leftarrow\!\!- CPR +.\times V$$

- trasla l'origine nel centro di proiezione

- costruisci la matrice di proiezione (CPR[c,d,r])

$$\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1/r \\ -c & -d & 0 & 1 \end{vmatrix}$$

Il vettore risultante (X,Y,0,W) deve essere diviso per W

<u>proiezione obliqua</u>

- e' espressa dalla matrice

$$\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ a & b & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

dove a e b sono definiti nel rif. 3 pag 498.

7. clip bidimensionale sulla Window

8. mapping da Window a Viewport

3. APLCORE

3.1 Il linguaggio

Nel seguito per la descrizione di algoritmi e funzioni si fa uso del linguaggio APL. E' stato scelto questo linguaggio per i seguenti motivi:

- l'APL e' indicato per eseguire operazioni su matrici come quelle richieste dalle trasformazioni geometriche.
- il linguaggio APL offre dei vantaggi da un punto di vista didattico in quanto non ponendo restrizioni sui nomi delle funzioni consente di conservare per intero la sintassi proposta dal SIGGRAPH. Per esempio per chiamare la funzione che disegna una linea passante per piu' punti si invoca:

  POLYLINE_ABS_2

- il concetto di workspace con la possibilita' di separare le funzioni in gruppi omogenei porta naturalmente alla realizzazione di un sistema con struttura modulare.
- la caratteristica di linguaggio interpretato e' molto utile sia durante la realizzazione e il 'debugging' delle funzioni del sistema sia per la realizzazione di programmi applicativi. L'utente puo' vedere il sistema come un linguaggio di comandi da utilizzare interattivamente o come un insieme di funzioni da chiamare dall'interno di una routine.

3.2 I livelli

APLCORE e' stato realizzato secondo i concetti base della proposta SIGGRAPH. In particolare c'e' pieno accordo con le seguenti idee:

- il concetto di tre sistemi di coordinate: world coordinate system, normalized coordinate system, device coordinate system.
- separazione delle funzioni di input da quelle di output
- funzioni di output per trasformare world coordinates in device coordinates attraverso operazioni di viewing.
- concetto di segmenti ciascuno dei quali e' una unita' grafica indipendente.
- definizione degli attributi per le primitive di output e

degli attributi dinamici per i segmenti.

- funzioni per inizializzare e terminare il sistema, inizializzare una o piu' 'view surfaces' e selezionarne una alla volta per la generazione della immagine; funzioni per raggruppare in un 'batch' tutte le modifiche all'immagine.
- separazione delle capacita' funzionali di input e di output in livelli.

APLCORE prevede infatti l'utilizzo da parte di applicazioni che richiedono prestazioni a diverso livello di complessita' come elencato di seguito:

- rappresentazione di immagini in due dimensioni.
- disegno di tipo 'batch' come rappresentazione di immagini su un plotter.
- input sincrono attraverso 'keyboard', 'button', 'locator', 'stroke'.
- disegno interattivo con creazione e modifica di segmenti e dei loro attributi dinamici.
- selezione di un dispositivo di output tra quelli supportati dal sistema mediante caricamento del driver relativo.

Tutte queste funzioni costituiscono insiemi separati come illustrato in FIGURA 11. Ogni spazio di lavoro rappresenta un insieme di capacita' funzionali o livelli; una freccia da uno spazio di lavoro ad un altro significa che il primo e' necessario per disporre delle funzionalita' del secondo. L'applicazione puo' scegliere tra i seguenti livelli:

OUTPUT	1 o 2
INPUT	0 o 1
DIMENSION	2

Vediamo piu' in dettaglio il contenuto degli spazi di lavoro:

CORE contiene le funzioni per inizializzare e terminare il sistema grafico e per controllare gli errori.

CORE1 contiene le funzioni di output al livello 1, cioe' tutte le primitive di output, gli attributi primitivi, le operazioni di viewing e la definizione dei segmenti temporanei. Sono suddivise in quattro gruppi: OUTPUT, VIEWING, CONTROL, ATTRIBUTES.

CORE2 contiene le funzioni di output per il livello 2, cioe' per la definizione dei segmenti 'retained' e degli attributi dinamici 'visibility' e 'highlighting'.

COREI contiene le funzioni per il livello 1 di input, cioe' per la gestione dell'input sincrono attraverso tastiera, tasti funzionali, joy-stick e tavoletta digitalizzatrice.

DRIVER1 contiene tre gruppi di funzioni:

BASEGP comprende le funzioni e le variabili di base per l'output e l'input attraverso la stazione grafica.

INTFGP comprende le funzioni dell'interfaccia tra APLCORE e DRIVER.

SEGMTGP comprende le funzioni per la gestione dei segmenti 'retained'.

In qualsiasi momento dopo l'inizializzazione il sistema grafico puo' essere terminato e successivamente inizializzato di nuovo ad uno dei livelli permessi. E' prevista la possibilita' di inizializzare piu' di una 'view surface', ma una sola alla volta puo' essere selezionata per la rappresentazione dell'immagine.

Il sistema supporta un meccanismo di 'batching of updates' mediante il quale l'applicazione comunica al sistema grafico che sta per iniziare una serie di modifiche all'immagine ma che solo alla fine se ne vuole l'aggiornamento. La strategia usata e' quella definita 'defer deletions' per cui solo le modifiche che comportano una azione di 'new frame' vengono raggruppate e ritardate.

La stazione grafica supporta ad hardware gli attributi grafici elencati di seguito:

INTENSITY identifica il 'modo' in cui deve avvenire la rappresentazione dei segmenti sullo schermo. L' intensita' assume il valore 1 in 'modo stored', il valore .5 in 'modo refreshed' e 0 se la linea deve apparire solo momentaneamente ('write throu mode'). Inizialmente INTENSITY ha il valore 1.

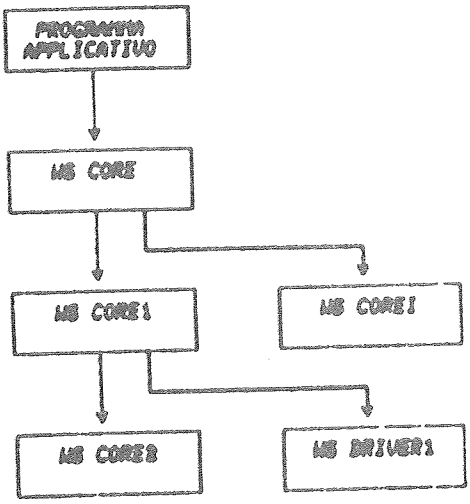


FIGURA 11. APLCORE . STRUTTURA MODULARE

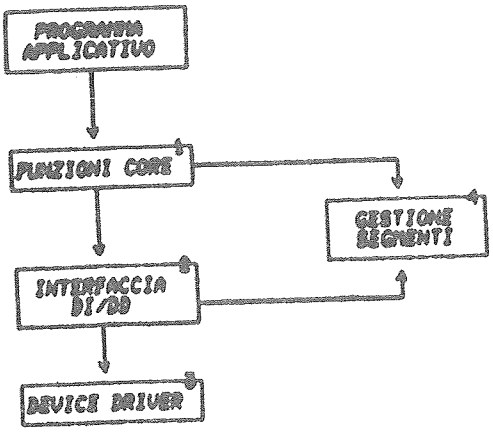
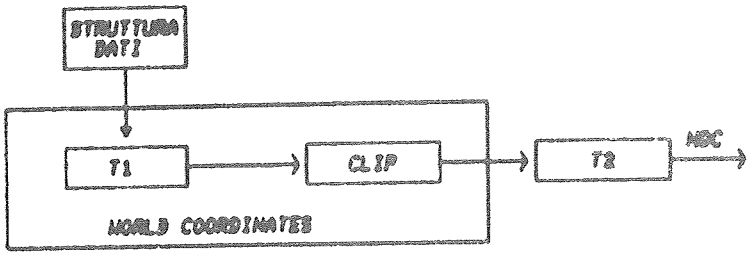


FIGURA 12. APLCORE - STRUTTURA FUNZIONALE



T1 = MATRIX4$ *.= MATRIX4LP4$
T2 = MATRIX4WORLDATO4NDC4$

FIGURA 13. APLCORE . TRASFORMAZIONI DI VIEWING.

LINESTYLE — le linee possono essere di 15 stili diversi identificati da numeri interi che vanno da 1 a 15. Inizialmente LINESTYLE ha valore 1.

LINEWIDTH — ci sono due spessori per le linee in 'stored mode': lo spessore normale o 'focused' (1) e lo spessore doppio o 'defocused' (2). In 'modo refreshed' c'e' solo il primo spessore. Inizialmente LINEWIDTH ha valore 1.

FONT — tutte le fonti i cui nomi sono contenuti nella variabile FONTL sono disponibili e possono venire caricati nel 'pattern generator' assegnando alla variabile FONT un numero intero che funziona da puntatore ad una riga di FONTL. Inizialmente FONT ha valore 1.

CHARSIZE — e' possibile selezionare 64 diverse dimensioni per i caratteri hardware ciascuna delle quali e' multipla di quella iniziale. Il 'charbox' che corrisponde ad 1 ha le seguenti dimensioni (in coordinate device 0 - 4095):

space	linespace	width	height
15	23	12	16

MARKER SYMBOL — sono previsti cinque tipi di simboli. La dimensione del simbolo e' fissa e non e' influenzata da CHARSIZE. Inizialmente MARKER SYMBOL ha valore 1.

3.3 Struttura funzionale

Nella FIGURA 12 sono illustrate le caratteristiche funzionali del sistema. Il modulo 1 contiene le funzioni descritte dal SIGGRAPH cioe' le sole funzioni chiamate dal programma applicativo. Contiene inoltre alcune funzioni di utilita' per trasformazioni di coordinate e valutazione degli attributi dei testi. Il modulo 2 o interfaccia tra le funzioni DI (Device Independent) e le funzioni DD (Device Dependent) contiene solo funzioni che mettono in comunicazione il sistema con le routine che guidano il 'driver'. Il modulo 4 contiene le funzioni che mantengono il 'display file' segmentato.

3.3.1 Le funzioni del CORE

Per schematizzare la descrizione distinguiamo alcuni insiemi di funzioni

primitive grafiche sono usate per disegnare linee e per scrivere testi

funzioni di Windowing permettono al programmatore di scegliere il proprio sistema di coordinate e di definire i contorni del disegno

funzioni di segmentazione per una facile modifica dell'immagine

funzioni di trasformazione per scalare,ruotare,traslare

funzioni di input che permettono all'utente di dare comandi per utilizzare i dispositivi di input

funzioni di utilita' come le funzioni di inizializzazione e quelle per conoscere il valore delle variabili del sistema.

Le primitive grafiche sono state implementate separando la procedura che esegue operazioni che non hanno nulla a che fare con le caratteristiche del dispositivo di output dalla procedura che genera il codice dell'immagine. Per esempio la funzione LINE comprende le seguenti fasi:

controllo dell'input
VIEW
OUTPUT1

View esegue tutte le trasformazioni di viewing illustrate in FIGURA 13 mentre la funzione OUTPUT1 rappresenta l'oggetto descritto dalle coordinate NDC sulla superficie di output.

Le funzioni per la gestione dei segmenti invocano il servizio del 'modulo gestione segmenti' che per comodita' e' contenuto nello spazio di lavoro del 'driver' poiche' il modo in cui il sistema memorizza i segmenti e' strettamente collegato alle caratteristiche fisiche del dispositivo di output. Per esempio la funzione

CREATE_RETAINED_SEGMENT n

chiama la funzione C_R_S che provvede ad inizializzare quelle variabili che sono utilizzate per memorizzare il segmento.

Tutte le funzioni del CORE implementate sono elencate in APPENDICE II.

3.3.2 Gestione dei segmenti

Le applicazioni di computer graphics richiedono spesso che solo una piccola parte di una immagine venga modificata e in generale per una programmazione dinamica e' necessario poter aggiungere nuove parti ad un disegno, muoverle o cancellarle. Il primo passo e' quindi quello di fornire al programmatore la possibilita' di distinguere con dei nomi parti diverse del disegno e di fornirgli poi i comandi per modificare selettivamente l'immagine. Chiameremo segmento l'unita' logica elementare modificabile dell'immagine. Per il programmatore esso non e' altro che una parte del disegno, per il sistema grafico e' un insieme di comandi di output grafico.

Le funzioni di questo modulo mantengono le informazioni relative ai segmenti 'retained'. Quando il segmento n viene creato le funzioni CORE memorizzano il nome del segmento mentre il modulo per la gestione dei segmenti inizializza le seguenti variabili

VISIBILITYn
HIGHLIGHTINGn

REFRESHn
STOREn

Le prime due variabili contengono gli attributi dinamici, mentre nelle altre verranno messe le coordinate in codice e gli attributi primitivi delle funzioni di output. La stazione grafica puo' disegnare in modo stored o rinfrescato e pertanto si prevedono per ogni segmento due tipi di vettori in cui possono essere memorizzati i comandi di output. Se viene impostata a zero la variabile VISIBILITYn l 'immagine del segmento scompare ma il segmento non viene distrutto e puo' essere reso visibile ancora senza che sia necessario ridefinirlo.

Come si e' visto ogni segmento ha per nome un numero intero. Esso costituisce il nome interno del segmento e viene usato come suffisso per le variabili relative al segmento. Le modifiche ai segmenti vengono effettuate modificando il contenuto delle variabili sopra descritte.

Le funzioni del CORE invocamo i servizi di questo modulo per creare, cancellare segmenti o per modificarne gli attributi dinamici. Le funzioni dell'interfaccia DI/DD (FIGURA 12) aggiungono coordinate o testi ai dati segmentati.

Le funzioni di questo modulo sono elencate in APPENDICE III gruppo SEGMTGP.

3.3.3 Interfaccia DI/DD

Questa interfaccia mette in comunicazione la parte 'device independent' (DI) del sistema con quella 'device dependent' (DD). E' costituita da alcune funzioni che ogni 'device driver' deve possedere. Si e' adottatta la convenzione che tutte queste funzioni abbiano la seguente sintassi:

NOMEn

dove NOME e' il nome della funzione ed n e' il numero che identifica il dispositivo fisico (1 per la stazione grafica).

Le funzioni del CORE invocano le funzioni dell'interfaccia per eseguire un output grafico o per richiedere l'accesso ad un dispositivo di input. In quest'ultimo caso c'e' uno scambio di variabili tra il CORE e l'interfaccia. Per esempio in seguito alla richiesta di input da tastiera funzionale mediante la funzione:

AWAIT_ANY_BUTTON

l'interfaccia restituisce la variabile:

BUTTON_NUM

che contiene il numero della PF selezionata.

Tutte le funzioni dell'interfaccia sono elencate nell'APPENDICE III nel gruppo COREGP.

4. LA STAZIONE GRAFICA

La stazione grafica usata per lo sviluppo dell'APLCORE e' stata realizzata utilizzando il dispositivo hardware GA (Graphics Attachment) che consente di connettere ad una IBM 3277 Display Station una grande varieta' di dispositivi grafici.

La configurazione attualmente utilizzata (vedi FIGURA 14) comprende:

IBM 3277 Display Station

TEKTRONIX 619 Storage Display Monitor

tavoletta digitalizzatrice SUMMAGRAPHICS

plotter TEKTRONIX 4662

hardcopy TEKTRONIX

input grafico joystick

Il risultato e' una stazione grafica a due schermi in cui l'operatore puo' comunicare con il computer attraverso la 3277 Display Station mentre l'altro schermo e' utilizzato unicamente per l'input e l'output grafico.

Il GA riceve quelle stringhe di dati che sono delimitate dai caratteri BGS (Begin Graphics Stream) e EGS (End Graphics Stream). Tutto quello che e' contenuto tra questi estremi e' costituito da comandi grafici e da dati. Il GA riconosce ed esegue i comandi e quando opera in 'vector mode' interpreta i dati come coordinate X-Y di punti sulla superficie di output; quando si trova in 'alpha mode' li considera come simboli da rappresentare.

Per esprimere le coordinate si puo' scegliere uno dei seguenti formati: 'absolute' che consente di indirizzare fino a 4095 punti, 'long relative' per indirizzare al massimo 2047 punti dalla posizione corrente e 'short relative' per un indirizzamento simile al precedente ma per soli 127 punti. La dimensione dell'immagine rappresentata per mezzo di vettori relativi puo' essere moltiplicata per un fattore di scala che va da 0 a 63 mediante il comando SCn con n uguale al valore di scala. Il valore 0 riduce tutti gli oggetti ad un punto.

E' possibile scegliere fra 15 tipi di linee che vanno dalla linea solida a diversi tipi di linee tratteggiate. Inoltre ogni linea puo' avere uno spessore normale (normal) o doppio (defocused) per mezzo dei comandi SMZ e SDZ.

Infine i vettori possono essere rappresentati in modo 'stored' o 'refreshed' o 'blinked'. Il modo 'refreshed' puo' essere utilizzato per la cancellazione selettiva dello schermo o per generare una immagine grafica che puo' essere spostata sullo schermo per mezzo del joystick. Questo ultimo effetto e' controllato dal GA senza la necessita' di

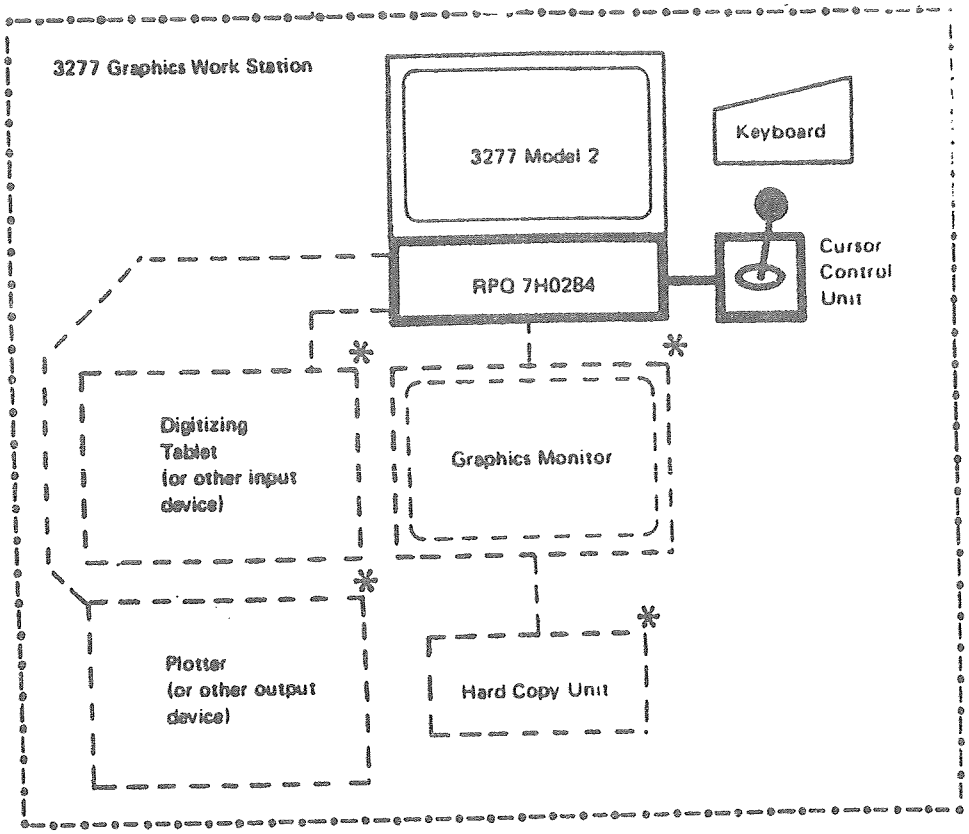

3277 Graphics Work Station
3277 Model 2
Keyboard
RPQ 7H0284
Cursor
Control
Unit
Digitizing
Tablet
(or other input
device)
Graphics Monitor
Plotter
(or other output
device)
Hard Copy Unit

FIGURA 14.

comunicare con il computer.
Il principale elemento di questo processo e' il 'tracking register'. In questo registro si puo' memorizzare una qualsiasi posizione dello schermo e l'utente mediante un joystick puo' variarne il contenuto. Il registro e' influenzato dai seguenti comandi

Initialize Tracking Register (ITR) pone nel registro le coordinate assolute di un punto.

Move To Tracking Register (MTR) il beam si muove nella posizione definita nel registro.

Draw To Tracking Register (DTR) disegna un vettore dalla posizione corrente del beam fino alla posizione contenuta nel registro.

Read Tracking Register With Enter or With No Enter (RTE,RTN) attiva l'input. Premendo un tasto qualsiasi il contenuto del registro viene decodificato e scritto nel buffer alfanumerico insieme ad un byte che contiene il carattere selezionato. Quando si usa RTN bisogna premere il tasto di ENTER dopo aver introdotto l'ultimo punto di una sequenza affinche' l'insieme di punti venga inviato al calacolatore. In pratica RTE attiva l'input di un solo punto,mentre RTN attiva l'input di uno o piu' punti.
Una tipica serie di operazioni per attivare l'input grafico comprende i seguenti comandi

ITR ordina di posizionare un oggetto in un punto stabilito

MTR il contenuto del Tracking Register diventa la posizione corrente

vettori relativi per rappresentare un oggetto

RTE per iniziare l'input

I dati grafici possono essere inviati anche ad un dispositivo di output come un plotter che sia stato connesso all'Input/Output Port. Questo si ottiene inserendo il comando EOP (Enable Output Port) nella stringa grafica. I dati che seguono questo comando sono trasmessi tramite l'Input/Output Port.
E' possibile infine avere una hard copy dell'immagine sullo schermo sia attraverso il controllo diretto dell'hard copy sia inserendo il comando HCY (Hard Copy) nella stringa grafica.
Per quanto riguarda l'input sotto forma di coordinate X-Y,questo puo' avvenire anche utilizzando una tavoletta digitalizzatrice connessa all'Input Port.
Poiche' le applicazioni usano dati alfanumerici la stazione grafica comprende un generatore di caratteri PPG (Programmable Pattern Generator) che puo' essere

inizializzato con un set di simboli opportunamente definiti.
Tutti i simboli estratti dalla memoria del PPG sono influenzati dai seguenti attributi:

scalatura
modo 'refreshed'
modo 'stored'
modo 'blinked'
linewidth

Non sono soggetti all'attributo linestyle.

APPENDICE I
Dizionario dei termini CORE

Device driver la parte del sistema grafico che supporta il dispositivo fisico. Esso genera l'output 'device dependent' e gestisce le interazioni con il device.

World Coordinate System (WC) sistema di coordinate cartesiane usato dall'utente per descrivere gli oggetti del suo disegno.

Normalized Device Coordinate System (NDC) sistema cartesiano indipendente dal 'device' le cui coordinate sono comprese tra 0 e 1. Le coordinate NDC sono usate per descrivere l'immagine degli oggetti.

Device Coordinate System o Screen Coordinate System sistema di coordinate che dipende dal device. Il 'device driver' trasforma le coordinate normalizzate in 'device coordinates'.

Viewing Operation comprende tutte quelle operazioni che servono per definire la parte visibile del disegno e per creare l'immagine mediante trasformazione di coordinate dallo spazio WC a quello NDC.

Modelling System un sistema ad alto livello per la descrizione degli oggetti che costituiscono il disegno. Esso descrive oggetti per il sistema grafico usando coordinate del mondo (WC).

Normalized Device Coordinate Space (NDC-space) e' una regione finita all'interno del sistema di coordinate NDC. Essa definisce la massima regione sul 'device' che il programma applicativo puo' utilizzare.

Current Position (CP) definisce la posizione corrente sul dispositivo di output in coordinate WC. Inizialmente e' nell'origine delle coordinate WC e viene modificato dalle funzioni che creano le primitive di output.

Object l'unita' grafica dell'applicazione. Il sistema CORE mediante le 'viewing operations' crea l'immagine di un oggetto. L'applicazione descrive gli oggetti per il CORE in coordinate WC in termini di primitive di output e attributi.

Image una particolare 'vista' di uno o piu' oggetti ed e' parte di un disegno su una superfivie di output. Essa e' definita dalle primitive di output e dagli attributi del segmento.

View Surface superficie logica di output. Le immagini definite sulla superficie logica sono disegnate sul corrispondente dispositivo fisico di output.

Output Primitive o Primitive un elemento dell'immagine come

linea,testo la cui apparenza e' determinata da attributi grafici.

Primitive Attribute caratteristica grafica di un elemento primitivo del disegno. Il CORE definisce un elenco di attributi primitivi tra cui colore, stile della linea, dimensione del carattere.

Segment e' un insieme di linee o testi che definiscono un oggetto sulla 'view surface'

Temporary Segment segmento senza nome ne' attributi. Rimane visibile fino a che non avviene una azione di new-frame.

Retained Segment e' un segmento identificato da un nome e caratterizzato da attributi dinamici che possono essere cambiati in modo da variare l'immagine. Per variare le primitive contenute nel segmento questo deve essere cancellato e ricreato.

Segment Attribute e' una caratteristica dei segmenti 'retained'. Il CORE definisce per ogni segmento un attributo statico e quattro attributi dinamici. L'attributo statico specifica quale trasformazione dell'immagine (rotazione,scalatura,traslazione) puo' essere attuata. Gli attributi dinamici sono 'visibility,highlighting,detectability e image transformation'. Questi ultimi possono essere modificati durante la vita del segmento per cambiarne le caratteristiche grafiche.

Image Transformation attributo dinamico di un segmento che fornisce i valori di rotazione, scalatura,traslazione dell'immagine.

New Frame Action vengono eliminate tutte le informazioni temporanee e se necessario tutte le informazioni 'retained' vengono ridisegnate. Per un dispositivo di stampa, per esempio, la carta avanza e si posiziona per la stampa su una nuova pagina.

Batch of Updates una sequenza di modifiche all'immagine raggruppate insieme. Per esempio si puo' cambiare l'immagine alla fine del batch su un terminale a 'storage' per evitare successive cancellazioni dallo schermo.

APPENDICE II
Lista delle funzioni APLCORE

funzioni di controllo

INITIALIZE_CORE
TERMINATE_CORE
INITIALIZE_VIEW_SURFACE
TERMINATE_VIEW_SURFACE
SELECT_VIEW_SURFACE
DESELECT_VIEW_SURFACE
BEGIN_BATCH_OF_UPDATES
END_BATCH_OF_UPDATES
NEW_FRAME
LOG_ERROR

operazioni di viewing

SET_WINDOW
SET_VIEW_UP
SET_VIEWPORT_2
MAP_NDC_TO_WORLD_2
MAP_WORLD_TO_NDC_2
SET_WINDOW_CLIPPING
SET_WORLD_COORDINATE_MATRIX_2

primitive di output

MOVE_ABS_2
MOVE_REL_2
LINE_ABS_2
LINE_REL_2
POLYLINE_ABS_2
POLYLINE_REL_2
TEXT
MARKER_ABS_2
MARKER_REL_2
POLYMARKER_ABS_2
POLYMARKER_REL_2

attributi primitivi

SET_INTENSITY
SET_LINESTYLE
SET_LINEWIDTH
SET_FONT
SET_CHARSIZE
SET_CHARUP_2
SET_CHARPATH
SET_CHARSPACE
SET_CHARJUST
SET_CHARPRECISION
SET_MARKER_SYMBOL

segmentazione

CREATE_TEMPORARY_SEGMENT
CLOSE_TEMPORARY_SEGMENT
CREATE_RETAINED_SEGMENT
CLOSE_RETAINED_SEGMENT
DELETE_RETAINED_SEGMENT
RENAME_RETAINED_SEGMENT
DELETE_ALL_RETAINED_SEGMENTS
OPEN_TEMPORARY_SEGMENT
OPEN_RETAINED_SEGMENT
RETAINED_SEGMENTS_NAMES

attributi dinamici

SET_VISIBILITY
SET_HIGHLIGHTING
INQUIRE_VISIBILITY
SET_SEGMENT_VISIBILITY
SET_SEGMENT_HIGHLIGHTING
INQUIRE_SEGMENT_VISIBILITY

INQUIRE_HIGHLIGHTING

INQUIRE_SEGMENT_HIGHLIGHTING

primitive di input

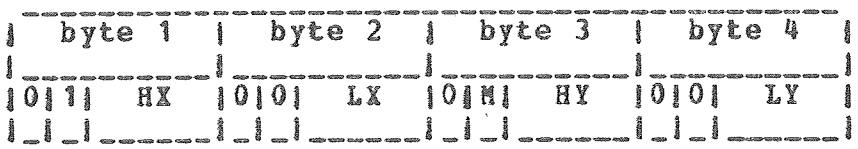
INITIALIZE_DEVICE
INITIALIZE_GROUP
TERMINATE_DEVICE
TERMINATE_GROUP
SET_ECHO
SET_ECHO_SEGMENT
SET_ECHO_SURFACE
SET_ECHO_POSITION
SET_KEYBOARD
SET_BUTTON
SET_LOCATOR_2
AWAIT_ANY_BUTTON_GET_LOCATOR_2

AWAIT_ANY_BUTTON
AWAIT_KEYBOARD
INQUIRE_INPUT_CAPABILITIES
INQUIRE_ECHO
INQUIRE_ECHO_SURFACE
INQUIRE_ECHO_POSITION
INQUIRE_KEYBOARD
INQUIRE_BUTTON
INQUIRE_ECHO_SEGMENTS
INQUIRE_LOCATOR_2

APPENDICE III
Lista delle funzioni del Driver1

Gruppo BASEGP: funzioni

z<-AV IN (Absolut Vector) costruisce le coordinate assolute dalle coordinate schermo contenute nella matrice (vettore) a tre componenti IN secondo il seguente formato



z<-CURSOR XY inizializza il tracking register a partire dalle coordinate schermo XY e gli associa il simbolo contenuto nella variabile TP. Restituisce nella variabile KEY il carattere battuto e in Z le coordinate normalizzate del punto sullo schermo.

DELAYCLEAR funzione per il terminale IBM 3277. Cancella lo schermo prima di un READ o WRITE.

ENCODE IN traduce in indirizzi di schermo le coordinate della matrice a tre colonne IN.

FIXTP M dalle coordinate schermo contenute nella matrice a due colonne M crea il simbolo di tracking e lo assegna alla variabile globale TP.

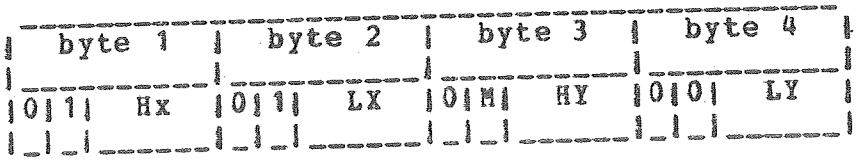
FORMAT F funzione per il terminale IBM 3277. Stabilisce un nuovo formato per lo schermo.

Z<-ITR XY (Initialize Tracking Register) costruisce un vettore che puo' essere caricato nel tracking register. XY e' un vettore a due componenti e contiene 'device coordinates'.

LCG (Load Character Generator) carica il generatore di caratteri esistente nello spazio di lavoro e contenuto nella variabile globale APLCG.

LPG S (Load Pattern Generator) carica il generatore di caratteri di nome S. Il nome deve essere elencato nella variabile FONTL.

Z<-LRV IN (Long Relative Vector) decodifica le coordinate schermo contenute nella matrice (vettore) a tre colonne IN secondo lo schema

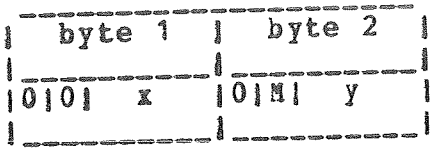


Z<-READSCREEN funzione per il terminale IBM 3277. Legge un input dallo schermo e restituisce un vettore di stato.

SHARE inizializza l'Auxiliary Processor AP231 attraverso cui vengono inviati i comandi grafici.

SCL I costruisce il comando set scale. I assume valori da 0 a 63.

Z<-SRV IN (Short Relative Vector) decodifica le coordinate di schermo contenute nella matrice (vettore) a tre colonne IN secondo lo schema



Gruppo BASEGP: variabili

APLCG Default Tracking Register
DTR Draw to Tracking Register
FONTL Font List
BGS Begin Graphic String
SAM Set Alpha Mode
ITR Initialize Tracking Register
ESC Escape
RTN Read Tracking Register with Enter
SDZ Set Defocused Z
SNZ Set Normal Z
HCY Hard Copy
LTS Set Line Type
SWZ Set Write Thru Z
REF Refresh
MTR Move to Tracking Register

Gruppo COREGP : funzioni

C_SIZE1 calcola le dimensioni del 'charbox' in coordinate NDC in base ai valori correnti degli attributi primitivi. Il 'charbox' e' definito dai seguenti quattro valori: space, linespace, width, height.

ECHO1 POS posiziona il beam nel punto di coordinate NDC indicate in POS.

INDEV1 CN richiede l'input dal device di classe e numero specificati in CN.

INDEV2 N (Input Device 2) accetta l'input da tastiera e forma le variabili LENGTH e INPUT_STRING.

INDEV3 L (Input Device 3) accetta l'input da un tasto funzionale e ne mette il valore in BUTTON_NUM.

INDEV35 L (Input Device 3 e 5) input mediante cursore e tastiera funzionale. La variabile XYNDC alla fine contiene le coordinate NDC del punto selezionato e BUTTON_NUM il valore del tasto.

N-F1 (New Frame) esegue il comando CORE di NEW_FRAME.

OUTPUT1 IN produce output grafico in base alle coordinate NDC contenute nella matrice IN dopo aver costruito la stringa dei comandi comprendente gli attributi definiti dalle funzioni del CORE. Memorizza la stringa in codice nei segmenti.

Z<-PRIM1 I per ogni attributo primitivo restituisce i valori possibili. I e' un puntatore alla seguente lista

COLOR INTENSITY LINESTYLE LINEWIDTH PEN FONT CHARWIDTH CHARHEIGHT DX_CHARUP DY_CHARUP CHARPATH CHARSPACE CHARJUST CHARPRECISION SYMBOL ID

S-F1 (Set Font) carica nel Pattern Generator la fonte definita dalla variabile FONT.

S-V-S1 (Select View Surface). Esegue alcune operazioni necessarie per utilizzare la stazione grafica; in particolare stabilisce il collegamento con il processor 231 mediante spartizione della variabile APL231.

XY TEST scrive una stringa a partire dalle coordinate NDC contenute in XY. XY puo' essere una matrice e S un vettore colonna con lo stesso numero di righe. La funzione memorizza il codice di uscita nel segmento.

Gruppo COREGP: variabili

NDC_SPACE_2 contiene le dimensioni dello spazio NDC indirizzabile sullo schermo (1 .76)

MARKER contiene tutti i simboli definiti.

LOCATOR_CHARACTERISTICS contiene il tipo di implementzione, i tipi di ECHO accettati e il nome della view surface a cui questo 'device' e' fisicamente attaccato.

BUTTON_CHARACTERISTICS come il precedente.

KEYBOARD_CHARACTERISTICS come il precedente.

INPUT_CAPABILITIES_1 contiene il livello di input e i devices disponibili per ogni classe di device.

DEVICE_STATUS contiene l'elenco dei devices disponibili.

Gruppo SEGMTGP

C-R-S N (Create Retained Segment) inizializza le variabili per il segmento N.

D-A-R-S (Delete All Retained Segmets) cancella tutti i segmenti retained.

D-R-S N (Delete Retained Segment) cancella il segmento N.

DRAW-S N (Draw segment) disegna il segment N definito in precedenza.

Z<-I-S-H N (Inquire Segment Highlighting) restituisce il valore di HIGHLIGHTING per il segmento N.

Z<-I-S-V N (Inquire Segment Visibility) restituisce il valore di VISIBILITY per il segmento N.

R-R-S O_N (Rename Retained Segment) cambia nome al segmento indicato nell'argomento.

S-S-H SH (Set Segment Highlighting) assegna un nuovo valore alla variabile HIGHLIGHTINGn e se il segmento e' visibile esegue una NEW_FRAME.

S-S-V SV (Set Segment Visibility) assegna un nuovo valore alla variabile VISIBILITYn e se il segmento e' visibile esegue una NEW_FRAME.

BIBLIOGRAFIA

[1] - ACM/SIGGRAPH GSPC
Status Report of the GSPC
Computer Graphics Vol. 13, N. 3, Aug. 1979

[2] - A. Freiden
A CORE Viewing System for APL
Computer Graphics Vol. 13, N. 1, Mar. 1979

[3] - I. Carlbom, J. Paciorek
Planar Geometric Projections
Computing Surveys Vol. 10, N. 4, Dec. 1978

[4] - W.M.Newmann, R. Sproull
Principles of Interactive Computer Graphics
McGraw-Hill 1979

[5] - Esperienze di Computer Graphics Mediante Implementazione del Sistema CORE
Tesi di Laurea di U.Blancato, A.A. 1980/1981
Relatori: G.Faconti,G.Pacini,P.Palamidese

[6] - IBM 3277 Display Station
Custom Feature Description
GA33-3039-2

STAMPATO PRESSO IL
SERVIZIO TIPOGRAFICO
DEL CNUCE