

A. TANASESCU  
I. D. MARINESCU

R. CONSTANTINESCU  
L. BUSUIOC

# GRAFICĂ ASISTATĂ

## Programe FORTRAN pentru reprezentări geometrice

2

AUTOMATICA

ELECTRONICA

INFORMATICA

MANAGEMENT

SERIA PRACTICA





BAIEM

AUTOMATICA

I  
N  
F  
O  
R  
M  
A  
T  
I  
C  
A



E  
L  
E  
C  
T  
R  
O  
N  
I  
C  
A

M A N A G E M E N T

# BIBLIOTECA DE AUTOMATICĂ, INFORMATICĂ, ELECTRONICĂ, MANAGEMENT

**BAIEM**  
**SERIA**  
**„PRACTICĂ“**

- Șt. Lozneau ș.a. Casetofoane. Depanare, Funcționare  
T. Rădulescu ș.a. Centrale telefonice automate.  
S. Călin, I. Dumitrache ș.a. Reglarea numerică a proceselor tehnologice  
G. Ionescu ș.a. Traductoare pentru automatizări industriale, vol. I  
L. Zamfirescu, I. Oprescu. Automatizarea cuptoarelor industriale  
I. Papadache. Automatica aplicată, ediția I și a II-a  
Șt. Alexandru. Automatizarea proceselor tehnologice în industria lemnului  
G. Raymond. Tehnica televiziunii în culori  
J. J. Samuelly, J. Pignaret, A. Sarazin. Instrumentația electronică în fizica nucleară  
T. Homoș. Capacitatea de producție în construcții de mașini  
S. Radu, D. Filoti. Centrale telefonice automate. Sisteme de comutație  
M. Bodea ș.a. Tranzistoare cu efect de câmp  
D. N. Shapiro. Proiectarea radioreceptoarelor  
V. Antonescu, M. Popovici. Ghid pentru controlul statistic al calității producției  
N. Stanciu ș.a. Tehnica imaginii în cinematografie și televiziune  
P. Vezeanu, Șt. Pătrașcu. Măsurarea temperaturii în tehnică  
T. Penescu, V. Petrescu. Măsurarea presiunii în tehnică  
P. Popescu, P. Mihordea. Măsurarea debitului în tehnică  
P. Vezeanu. Măsurarea nivelului în tehnică  
C. Hidoș, P. Isac (coordonatori). Studiul muncii, vol. I—VIII  
V. Baltac ș.a. Calculatorul FELIX C-256. Structură și programare  
G. Sonea, M. Silețchi. Creșterea planificată a productivității muncii  
R. L. Morris. Proiectarea cu circuite integrate TTL  
I. Stăncioiu. Eficiența economică a asimilării de utilaje noi  
Ishikawa Kaoru. Controlul de calitate pentru maștrii  
Magnus Radke. 222 măsuri pentru reducerea costurilor  
A. M. Buhtiarov ș.a. Culegere de probleme de programare  
P. Constantinescu ș.a. Sistemele informatice, modele ale conducerii și sistemelor conduse  
E. S. Buffa. Conducerea modernă a producției, vol. I și II  
A. Vătășescu ș.a. Dispozitive semiconductoare. Manual de utilizare  
A. Nadolo. Măsurarea volumului și cantității lichidelor în industrie  
Ch. Jones. Design. Metode și aplicații  
Gh. Pisău ș.a. Elaborarea și introducerea sistemelor informatice  
C. Hidoș. Analiza și proiectarea circuitelor informaționale în unitățile economice  
A. Vătășescu ș.a. Circuite integrate liniare. Manual de utilizare  
M. Silișteanu ș.a. Scheme de televizoare, magnetofone, picupuri, vol. 1 și 2, ed. a II-a  
D. W. Dawis. Rețele de interconectarea calculatoarelor  
V. Pescaru ș.a. Fișiere, baze și bănci de date  
D. Patriche. Marketing Industrial  
Gh. Baștiurea ș.a. Comanda numerică a mașinilor-unelte  
N. Sprinceană, R. Dobrescu, Th. Borangiu. Automatizări discrete în industrie. Culegere de probleme  
M. Florescu ș.a. Cibernetică, automatică, informatică în industria chimică  
S. Călin, ș.a. Optimizări în automatizări industriale  
S. Maican. Sisteme numerice cu circuite integrate  
M. Simionescu. Proiectarea unitară a circuitelor electronice  
R. Răpeanu ș.a. Echipamente periferice. Catalog  
T. Geber ș.a. Microprocesoare, microcalculatoare și roboți în automatizări industriale  
M. Guran, F. G. Filip. Sisteme terarhizate, în timp real, cu prelucrare distribuită a datelor  
A. Davidoviciu, B. Bărbat. Limbaje de programare pentru sisteme în timp real

Mat. dr .arh. **AURELIAN TĂNĂSESCU**  
ing. **RADU CONSTANTINESCU**      ing. **IOAN D. MARINESCU**  
mat. **LILIANA BUSUIOC**

# GRAFICĂ ASISTATĂ.

Programe FORTRAN  
pentru reprezentări  
geometrice

2



EDITURA TEHNICĂ  
București, 1989

Cuvînt înainte: ing. **NICOLAE VAIDESCU**  
Recenzii: ing. **VLADIMIR FIRȚA**  
dr. ing. **LIVIU DUMITRAȘCU**

Redactor: ing. **PAUL ZAMFIRESCU**

● Toate drepturile pentru această ediție (inclusiv pachetul de programe REPGEO pentru reprezentări geometrice inclus în BNP, avînd drept elaboratori autorii cărții și redacția de informatică și tehnică de calcul) rezervate Editurii Tehnice, România, București, Piața Științei 1

ISBN 973-31-0018-8  
ISBN 973-31-0016-1  
C.Z.: 76:681,3

Desene: **DANA GEGO**  
Coperta: **SIMONA DUMITRESCU**  
Tehnoredactor: **V. E. UNGUREANU**

---

Coli de tipar: 13,5 Bun de tipar: 1.08.1989

---



Tiparul executat sub comanda  
nr. 1412 la  
Intreprinderea poligrafică  
„13 Decembrie 1918”,  
str. Grigore Alexandrescu nr. 89-97  
București,  
Republica Socialistă România

# CUPRINS GENERAL

## VOLUMUL 1

Cuvint înainte .....	5
<i>Prefața autorilor</i> .....	7
Despre carte și autori.....	10
Cuprins general .....	11
<i>Cuprinsul volumului I</i> .....	12
<b>Capitolul I</b>	
<b>Biblioteca SOFTWARE pentru funcții grafice DIGIGRAF 1712 în 2D și 3D.....</b>	15
<b>Capitolul II</b>	
<b>Reprezentări automate în geometria descriptivă. Punctul. Dreapta. Planul. Probleme de paralelism, incidență și perpendicularitate. Subprograme geometrice<sup>1</sup> și de desen .....</b>	69
<b>Capitolul III</b>	
<b>Secțiuni plane în poliedre și intersecția dintre două poliedre .....</b>	97
<b>Capitolul IV</b>	
<b>Conice. Cuadrice. Suprafețe de rotație. Suprafețe de translație. Reprezentare. Secțiuni plane. Intersecții mixte de suprafețe .....</b>	159
<b>Capitolul V</b>	
<b>Generalizarea reprezentării spațiului tridimensional (<math>S^3</math>) pe un spațiu bidimensional (<math>S^2</math>)</b>	213
<b>Capitolul VI</b>	
<b>Reprezentarea obiectelor spațiale .....</b>	263
<b>ANEXE (I)</b> .....	282
<b>ANEXA A – Formule fundamentale necesare înțelegerii reprezentărilor geometrice în grafica pe calculator .....</b>	282
<b>ANEXA B – Subprogramul HAȘURA utilizat pe mesele de desen de tip ARISTO....</b>	307
– Programul pentru produsul dintre două matrice .....	310

## VOLUMUL 2

Cuprins general.....	5
<i>Cuprinsul volumului 2</i> .....	6
<b>Capitolul VII</b>	
<b>Algoritmi pentru rezolvarea problemelor liniilor ascunse (ALPLA) și a suprafețelor invizibile (APIS).....</b>	9
<b>Capitolul VIII</b>	
<b>Reprezentarea spațială grafică și vizuală a curbilor și suprafețelor .....</b>	91
<b>Capitolul IX</b>	
<b>Curbele în grafica pe calculator.....</b>	145
<b>Capitolul X</b>	
<b>Suprafețele în grafica pe calculator .....</b>	187
<b>ANEXE (II)</b>	
<b>ANEXA C – Programul PLANAR pentru desenarea modulară automată a planurilor de arhitectură și construcții .....</b>	259
– Programe pentru trasarea conturilor care îmbracă cu grosimi variabile un graf dat prin noduri și legăturile dintre ele .....	267
<b>ANEXA D – Programul PIESA pentru desenarea automată a unei piese parametrizate din domeniul construcțiilor de mașini .....</b>	282
<b>ANEXA E – Program pentru simularea automată a unor curbe și suprafețe în grafica artistică .....</b>	286
<b>BIBLIOGRAFIE.....</b>	294

<i>Cuprins general</i> .....	5	7.7.14. Subrutina PLOT .....	39
<i>Cuprinsul volumului 2</i> .....	6	7.8. Aplicații rezolvate prin progra-	
<i>Capitolul VII</i>		mul ALPLA .....	52
<b>ALGORITMI PENTRU REZOLVAREA</b>		7.8.1. Realizarea proiecțiilor triplu	
<b>PROBLEMELOR LINIILOR ASCUNSE</b>		ortogonale și ale perspecti-	
<b>(ALPLA) ȘI A SUPRAFETELOR</b>		velor centrale și paralele ale	
<b>INVIZIBILE (APIS)</b>		unui ansamblu de volume de	
A. Algoritmul ALPLA .....	9	arhitectură cu trasarea sau	
7.1. Prezentarea generală a algorit-		eliminarea liniilor invizibile	
mului ALPLA .....	9	(fig. 7.9—fig. 7.12) .....	52
7.1.1. Noțiuni definitorii .....	9	7.8.2. Intersecția dintre un cub și	
7.2. Formularea problemei. Poliedre		un octaedru care au o <i>dia-</i>	
convexe .....	10	gonală verticală comună (fig.	
7.2.1. Sistemele de referință și al-		7.13) .....	58
goritmul de calcul .....	12	7.8.3. Intersecția dintre doi tetra-	
7.3. Algoritmul ALPLA .....	13	edri reguțați în proiecție cen-	
7.3.1. Codificarea algoritmului		trală (fig. 7.14) .....	62
ALPLA .....	14	7.8.4. Vederea perspectivă: combi-	
7.3.2. Subprogramul DIORTO ..	15	nația dintre 10 corpuri cu go-	
7.3.3. Determinarea poziției relative		luri repetate într-o ordine ar-	
a planului de proiecție (ta-		bitrară (fig. 7.15 și fig. 7.16)	
bloul de perspectivă) .....	16	B. Algoritmul APIS .....	65
7.3.4. Subprogramul DATEIN ..	16	7.9 Algoritm și program pentru su-	
7.3.5. Determinarea proiecțiilor no-		prafețe invizibile — APIS ..	65
durilor $N_i$ pe planul de pro-		7.9.1. Descrierea generală .....	65
iecție .....	16	7.9.2. Fazele algoritmului .....	66
7.3.6. Subprogramul DMIMA ....	17	7.9.3. Structurarea memoriei ....	72
7.3.7. Selectarea fețelor pseudo-		7.9.4. Clipping .....	73
frontale .....	17	7.9.5. Ierarhizarea corpurilor ....	75
7.3.8. Crearea vectorului liniilor		7.9.6. Concluzii finale .....	87
fețelor pseudofrontale și se-		7.9.7. Aplicații rezolvate prin pro-	
lectarea liniilor nerepetitive		gramul APIS (fig. 7.17—	
7.3.9. Subprogramul LINFV .....	17	fig. 7.34) .....	88
7.3.10. Subprogramul LSELET ..	18	<i>Capitolul VIII</i>	
7.4. Plotarea configurațiilor convexe		<b>REPREZENTAREA SPAȚIALĂ GRA-</b>	
7.4.1. Subprogramul TRASA .....	18	<b>FIGICĂ ȘI VIZUALĂ A CURBELOR</b>	
7.4.2. Subprogramul LIR .....	18	<b>ȘI SUPRAFETELOR</b>	
7.5. Plotarea configurațiilor concave		8.1. Vizualizarea funcțiilor de două va-	
7.5.1. Subprogramul VIZLIN .....	22	riabile .....	91
7.6. Subprogramele de adaptare ale al-		8.1.1. Algoritmul lui Williamson ..	91
goritmilor în intersecția dintre		8.1.2. Algoritmul lui Wright .....	92
două poliedre oarecare convexe,		8.2. Algoritm și program pentru vizu-	
concave, cu goluri sau deschise		alizarea sau reprezentarea grafi-	
7.6.1. Subprogramul CINT .....	23	că a curbelor și suprafețelor în per-	
7.6.2. Subprogramul FATASG .....	24	sectivă cu studii porțiunilor de	
7.6.3. Subprogramul ESPI .....	24	suprafață ascunse — procedura	
7.7. Programul principal ALPLA		HIDE .....	93
(listare) .....	25	8.2.1. Procedura HIDE. Descri-	
7.7.1. Subrutina DATEIN .....	30	erea generală .....	93
7.7.2. Subrutina DMIMA .....	31	8.2.2. Descrierea algoritmului ..	94
7.7.3. Subrutina LINFV .....	31	8.2.3. Parametrii de apel ai pro-	
7.7.4. Subrutina LSELET .....	32	cedurii HIDE .....	98
7.7.5. Subrutina CINT .....	32	8.2.4. Listarea programului	
7.7.6. Subrutina ESPI .....	33	CURBA 1 .....	98
7.7.7. Subrutina FATASG .....	34	8.2.5. Programul CURBA 2....	99
7.7.8. Subrutina LIR .....	35	8.2.6. Subprogramul HIDE .....	102
7.7.9. Subrutina DIORTO .....	35	8.2.7. Subprogramul PDATA ..	105
7.7.10. Subrutina VIZLIN .....	36	8.2.8. Subprogramul PLOT .....	105
7.7.11. Subrutina DILP .....	38	8.2.9. Subprogramul DATEIN ..	106
7.7.12. Subrutina ORD .....	38	8.2.10. Subprogramul PROIEC ..	106
7.7.13. Subrutina TRASA .....	38		



8.3. Programele necesare utilizării imprimantei grafice pentru reprezentarea curbelor și suprafețelor....	107	9.6.1. Aproximarea polinomială Bernstein .....	164
8.3.1. Programul PLOTARE....	107	9.6.2. Curba Bézier .....	166
8.3.2. Subprogramul COMPACT ..	108	9.6.3. Construcția geometrică a unei curbe Bézier .....	167
8.3.3. Subprogramul CITIRE .....	109	9.6.4. Generalizarea la suprafețe a curbelor Bézier .....	167
8.3.4. Programul ORDONARE ..	109	9.6.5. Proprietăți ale curbei Bézier și aprecieri comparative ..	168
8.4. Aplicațiile procedurii HIDE (figurile 8.6—8.12) .....	110	9.6.6. Aproximarea Bernstein pentru funcțiile de două variabile .....	170
8.5. Interpolarea bivariată și montarea suprafețelor netede bazată pe proceduri locale .....	115	9.6.7. Aplicație .....	171
8.5.1. Introducere .....	115	9.7. Studiul curbelor determinate vectorial prin polinoame .....	172
8.5.2. Descrierea metodei de interpolare bivariată .....	115	9.7.1. Aplicație .....	172
8.5.3. Subrutina SFCFIT .....	116	9.7.2. Aplicație .....	173
8.5.4. Listarea subrutinei SFCFIT ..	122	9.8. Cubica Ferguson .....	173
8.5.5. Listarea subrutinei ITPLBV .....	127	9.8.1. Aplicație .....	174
8.5.6. Programul de interpolare bivariată și montarea suprafețelor netede .....	136	9.8.2. Aplicație .....	174
8.5.7. Subrutina MOV .....	139	9.8.3. Program pentru desenarea plană a curbei Ferguson .....	176
8.6. Trasarea curbelor de nivel ...	139	9.8.4. Forma programabilă a suprafeței de tip Ferguson....	176
8.6.1. Formele tablourilor pentru datele de intrare și pentru datele de ieșire .....	140	9.9. Cubica Bézier .....	177
8.6.2. Legătura posibilă dintre interpolarea bivariată și vizualizarea (perspectivă) a funcțiilor de două variabile ..	141	9.9.1. Aplicație .....	178
8.7. Aplicații ale interpolării bivariete și montării suprafețelor netede bazate pe proceduri locale (figurile 8.14—8.19) .....	141	9.9.2. Curba Bézier determinată de un poligon cu $n$ vîrfuri ..	178
<b>Capitolul IX</b>		9.10. Cubica COONS .....	179
<b>CURBELE ÎN GRAFICA PE CALCULATOR</b>		9.10.1. Observație comparativă asupra cubicelor .....	180
9.1. Introducere. Generalități .....	145	9.20.1. Aplicație .....	181
9.2. Determinarea curbei de interpolare care trece prin $n + 1$ puncte date .....	147	9.11. Curba COONS generală .....	181
9.2.1. Aplicație .....	148	9.11.1. Aplicație .....	182
9.3. Curba cubică parametrică spațială ..	149	9.12. Programul principal pentru interpolarea funcțiilor Hermite, Bézier, B-spline .....	183
9.3.1. Aplicație .....	149	<b>Capitolul X</b>	
9.4. Metode clasice de interpolare. Interpolarea Lagrange și interpolarea Hermite .....	150	<b>SUPRAFEȚELE ÎN GRAFICA PE CALCULATOR</b>	
9.4.1. Interpolarea Lagrange .....	150	10.1. Principii generale de construcție a suprafețelor în grafica pe calculator.....	187
9.4.2. Interpolarea Hermite .....	151	10.1.1. Suprafețele COONS. Modelarea suprafețelor .....	187
9.4.3. Aproximarea COONS .....	153	10.1.2. Suprafața de corecție a tangentelor.....	189
9.4.4. Interpolarea AKIMOV .....	155	10.1.3. Notarea matricială .....	190
9.5. Funcția B-spline și interpolarea cu B-spline .....	155	10.1.4. Suprafețele B-spline....	195
9.5.1. Funcția spline de gradul $m$ ..	156	10.1.5. Suprafețele riglate .....	196
9.5.2. Funcția și curba B-spline ..	157	10.1.6. Aplicația 1 .....	198
9.5.3. Aproximarea curbelor cu ajutorul funcției B-spline....	160	10.2. Suprafețe determinate prin polinoame .....	198
9.5.4. Aprecieri generale și comparative asupra aproximării cu B-spline .....	162	10.2.1. Cazuri speciale .....	199
9.5.5. Aplicație .....	163	10.2.2. Curbele principale ale suprafeței $P_{m,n}$ .....	200
9.6. Curbe Bézier și aproximarea Bézier a curbelor .....	164	10.2.3. Curbele marginale .....	200
		10.2.4. Punctele de colț ale suprafeței .....	201
		10.2.5. Aplicație .....	201
		10.3. Construcția suprafețelor definite prin puncte și prin curbe marginale .....	202

10.3.1. Aplicație. Suprafața $P_{1,2}$	203	10.13. Cubice Bézier marginale direc-	
10.3.2. Aplicație. Suprafața $P_{2,2}$	204	toare .....	238
10.3.3. Aplicație. Suprafața $P_{3,3}$	205	10.14. Curbe COONS marginale direc-	
10.4. Suprafețe determinate vectorial		toare .....	239
prin polinoame .....	206	10.15. Suprafețele definite prin rețele	
10.4.1. Aplicație .....	208	de puncte .....	240
10.4.1. Aplicație .....	208	10.15.1. Placă bicubică Bézier ..	240
10.5. Suprafețe plăci .....	209	10.15.2. Calitățile plăcii bicubice	
10.5.1. Cuplarea (alipirea, racor-		Bézier .....	241
darea) suprafețelor .....	209	10.15.3. Aplicație .....	242
10.5.2. Cuplarea netedă .....	210	10.15.4. Placă bicubică Coons de-	
10.5.3. Aplicație .....	211	terminată printr-o rețea	242
10.6. Cuplarea netedă între trei petice		10.15.5. Suprafața Bézier rețea	
de suprafață alăturate .....	212	$5 \times 3$ . Aplicații .....	244
10.6.1. Cuplarea în serie .....	213	10.16. Suprafețele Bézier. Generalități	245
10.6.2. Aplicație .....	213	10.16.1. Suprafața Bézier generală	246
10.6.3. Cuplarea de colț a petice-		10.16.2. Aplicație .....	247
lor de suprafață .....	214	10.16.3. Aplicație .....	247
10.6.4. Cuplarea netedă de colț		10.16.4. Suprafața placă Bézier	248
a peticelor de suprafață..	215	10.16.5. Aplicație .....	248
10.6.5. Aplicație .....	216	10.17. Programe pentru construcția	
10.7. Suprafețe bicubice .....	218	perspectivă a suprafețelor Bé-	
10.7.1. Cuplarea suprafeței placă		zior și COONS .....	249
$P_{3,3}$ .....	219	10.17.1. Programul pentru con-	
10.7.2. Aplicație .....	219	strucția perspectivă a su-	
10.7.3. Aplicație .....	220	praftețelor Bézier .....	249
10.7. Suprafețe de interpolare definite		10.17.2. Programul pentru con-	
prin curbe marginale (frontiere)	220	strucția perspectivă a su-	
10.8.1. Cuadrice riglate .....	220	praftețelor COONS .....	251
10.8.2. Aplicație .....	221	<b>ANEXE (II)</b>	
10.8.3. Placă bilineară COONS		ANEXA C1 – Programul PLANAR	
definită prin curbe mar-		pentru desenarea modulară auto-	
ginale .....	222	mată a planurilor de arhitectură	
10.8.4. Aplicație .....	223	și construcții .....	259
10.8.5. Aplicație .....	224	ANEXA C2 – Programe pentru tra-	
10.8.6. Aplicație .....	225	sarea conturilor care îmbracă	
10.8.7. Placă bicubică COONS		cu grosimi variabile un graf dat	
definită prin curbe mar-		prin noduri și legăturile dintre	
ginale .....	226	ele .....	267
10.8.8. Aplicație .....	227	– Programul ARBORE .....	270
10.9. Suprafața de interpolare defini-		→ Procedura pentru calcu-	
tă prin 12 vectori în matricea		larea unghiului dintre 2	
restricțiilor (suprafața Ferguson)	228	segmente (Subprogramul	
10.9.1. Aplicație .....	229	ANGLE) .....	274
10.9.2. Aplicație .....	230	→ Procedura de interschim-	
10.10. Suprafața de interpolare		bare a doi întregi (Subpro-	
COONS definită prin 16		gramul INTERS) .....	274
vectori în matricea restricțiilor	230	→ Procedura de calculare a	
10.10.1. Aplicație .....	231	intersecției a două drepte	
10.11. Suprafețe de interpolare		(Subprogramul DRDR) ..	274
COONS definite prin curbe		– Programul CONTUR .....	275
marginale și prin tangentele în		→ Subprogramul ROT .....	278
lungul acestor curbe .....	232	→ Subprogramul PLOTS ..	278
10.11.1. Aplicație .....	234	→ Subprogramul PLOT .....	278
10.11.2. Aplicație .....	235	Aplicații .....	278
10.11.3. Aplicație .....	235	ANEXA D Programul PIESA pen-	
10.12. Cubice Ferguson marginale di-		tru desenarea automată a unei	
rectoare .....	236	piese parametrizate din dome-	
10.12.1. Aplicație .....	236	niul construcțiilor de mașini ..	284
10.12.2. Suprafețe speciale deri-		ANEXA E Program pentru simu-	
vate din cubicele Fer-		larea automată a unor curbe și su-	
guson .....	237	praftețe în grafica artistică	286
		BIBLIOGRAFIE vol 1 și vol. 2....	294

## **A, ALGORITMUL ALPLA**

### **7.1. Prezentarea generală a algoritmului ALPLA**

Algoritmul și programul **FORTRAN**, prezentate în acest capitol, reprezintă o soluție a autorilor pentru Problema Eliminării Liniilor Invizibile (**PELI**).

Algoritmul creat este din categoria celor ce lucrează în „*spațiul obiect*”.

Într-o formă inițială, elaborată deja în anul 1975 algoritmul dădea posibilitatea utilizatorilor să vizualizeze structuri de corpuri topologic închise cu contururi de fețe concave sau chiar cu goluri.

În forma prezentată în lucrare este permisă, în plus, și existența a 2 poliedre intersectate în cadrul descrierii. Pentru diferențierea celor două variante, vom numi **ALPLA** ultima variantă.

Descrierea poliedrelor se face în modul prezentat în capitolul VI, adică prin coordonatele vîrfurilor în 3-D, urmate de descrierea poligoanelor care alcătuiesc fețele poliedrelor.

Definirea fețelor trebuie făcută în sens trigonometric, acest lucru fiind impus de necesitățile de calcul. O astfel de definire permite o eliminare radicală a fețelor total invizibile încă dintr-o fază incipientă a algoritmului, reducînd foarte mult timpul de calcul.

Dacă totuși, o asemenea definire nu este posibilă (în cazul în care sînt introduse în structură elemente care nu sînt topologic închise în spațiu), acest lucru trebuie specificat.

De asemenea, se poate specifica inițial în algoritm dacă întreaga structură introdusă este convexă, caz în care întreg algoritmul se rezumă la eliminarea fețelor invizibile și trasarea celorlalte fețe.

#### **7.1.1. Noțiuni definitorii**

Obiectele sau volumele de arhitectură sînt limitate de fețe, care sînt porțiuni de plan. Fiecare față este limitată de muchii, iar cele două puncte extreme ale muchiei sînt noduri. Fiecare muchie aparține la două fețe. O muchie este convexă (concavă), dacă corespunde unui diedru convex (concav).

Un nod este concav dacă aparține cel puțin unei muchii concave, contrar el este convex. Pe un plan de proiecție sau pe un tablou de perspectivă fețele  $F_i$  sînt proiectate prin poligoane  $P_i$ , muchiile  $M_i$  prin segmentele  $S_i$  și nodurile  $N_i$  prin vîrfurile  $V_i$ .

## 7.2. Formularea problemei. Poliedre convexe

Fiind dat obiectul sau volumul definit prin nodurile  $N_i(XN_i, YN_i, ZN_i)$  și fețele  $F_i$ , tabloul de perspectivă  $P$  și punctul optim de vedere  $\Omega(XV_i, YV_i, ZV_i)$  sau direcția optimă de vedere  $\Delta(\alpha, \beta, \gamma)$ , se cere să se stabilească algoritmul codificabil pentru a reprezenta automat proiecția volumului astfel definit pe tabloul de perspectivă  $P$  (fig. 7.1.a. și fig. 7.1.b.).

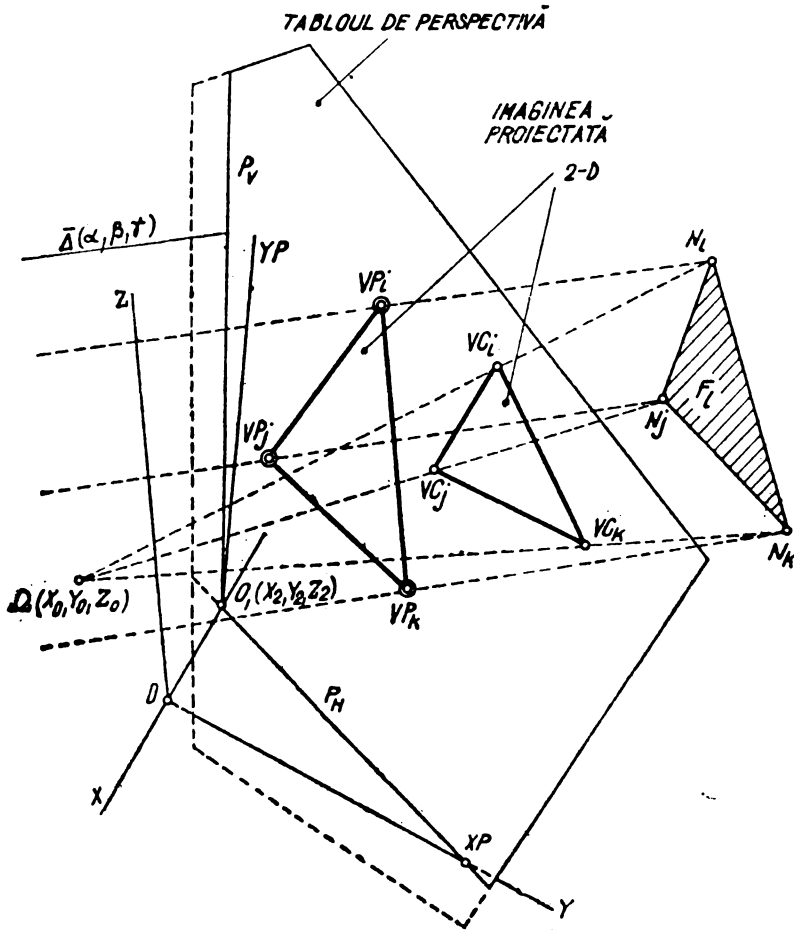


Fig. 7.1. a

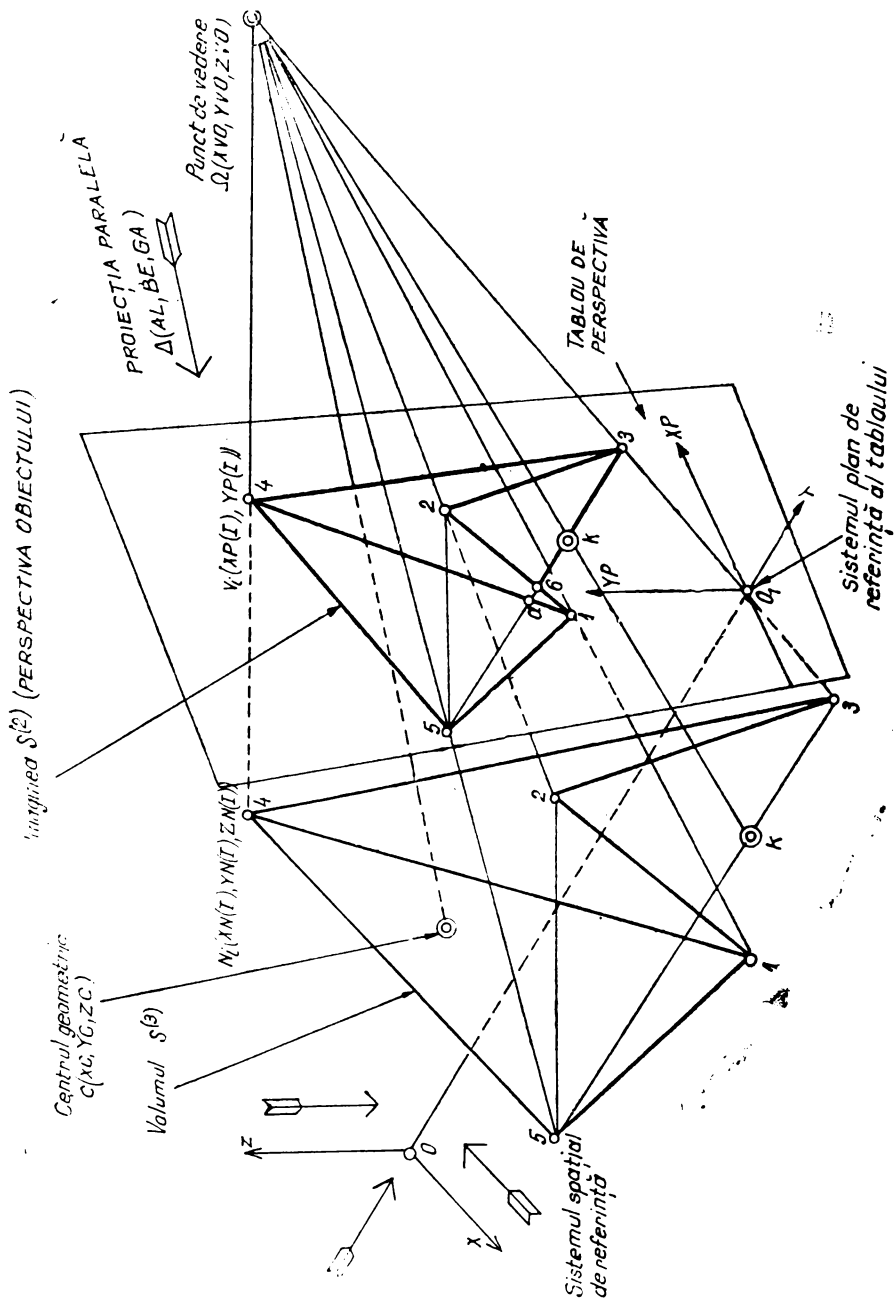


Fig. 7.1. b

## 7.2.1. Sistemele de referință și algoritmul de calcul

Sint folosite două sisteme de referință (fig. 7.2.):

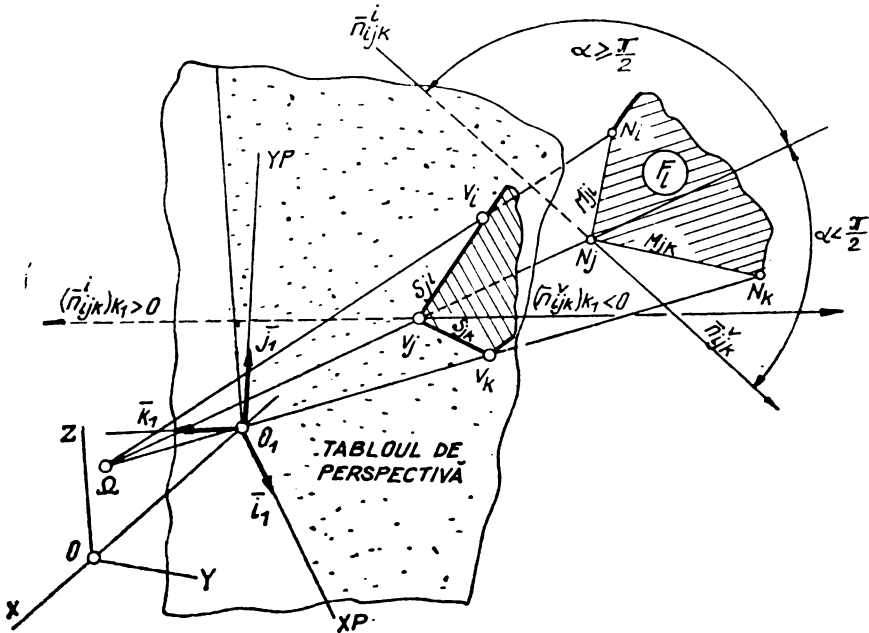


Fig. 7.2

— Sistemul fix  $(X, Y, Z)$  față de care sint definite coordonatele nodurilor, ecuația tabloului de perspectivă și coordonatele punctului de vedere  $\Omega$  sau direcția de proiectare  $\Delta(\alpha, \beta, \gamma)$ .

— Sistemul  $(XP, YP)$  față de care sint definite vîrfurile poligoanelor  $(VP_i$  sau  $VC_i)$  obținute prin proiectarea fețelor suprafeței poliedrale. Acest sistem este astfel ales, încît *direcția pozitivă a axei ZP să fie orientată către observator*.

Notînd prin  $A_i, B_i, C_i, (i = 1, 2)$  cosinșii directori ai axelor asociate tabloului de perspectivă, prin  $XV_i, YV_i, ZV_i$  sau prin  $AL, BE, GA$  coordonatele punctului de vedere ales  $\Omega$  sau cosinșii directori ai direcției de proiectare  $\Delta$ , prin  $X_2, Y_2, Z_2$  coordonatele originii axelor  $(XP, YP)$ , prin  $X, Y, Z$  coordonatele vîrfurilor  $VP_i$  sau  $VC_i$  în raport cu sistemul fix și prin  $A, B, C$  coeficienții ecuației tabloului de perspectivă  $P$ , legătura dintre spațiul  $S^{(3)}$  și spațiul  $S^{(2)}$  se exprimă prin următoarea expresie scrisă matricial:

$$\begin{bmatrix} XP \\ YP \end{bmatrix} = \begin{bmatrix} A_1 & B_1 & C_1 \\ A_2 & B_2 & C_2 \end{bmatrix} \cdot \begin{bmatrix} X - X_2 \\ Y - Y_2 \\ Z - Z_2 \end{bmatrix}$$

În funcție de sistemul de proiecție considerat, centrală sau paralelă, coordonatele vîrfurilor  $V$  au față de sistemul  $XYZ$  următoarele expresii:

— proiecția paralelă:

$$X, Y, Z = XN(I), YN(I), ZN(I) - \frac{F(XN(I), YN(I), ZN(I))}{A \cdot AL + B \cdot BE + C \cdot GA} AL, BE, GA,$$

— proiecția centrală:

$$X, Y, Z = XN(I), YN(I), ZN(I) - \frac{F(XN(I), YN(I), ZN(I))}{A(XV - XN(I)) + B(YV - YN(I)) + C(ZV - ZN(I))} \times \times (XV - XN(I)), (YV - YN(I)), (ZV - ZN(I))$$

unde funcția  $F$  are următoarea definiție:

$$F(X, Y, Z) = AX + BY + CZ - 1$$

Codificînd aceste expresii în acest stadiu, proiecția volumului considerat poate fi desenată cu toate liniile vizibile

### 7.3. Algoritmul ALPLA

Fie  $N_{i,j,k}$  trei puncte succesive feței  $F_i$  (fig. 7.2). Atunci produsul vectorial:

$$\overline{N_j N_i} \times \overline{N_j N_k} = \overline{M_{ji}} \times \overline{M_{jk}}$$

definește un vector normal dirijat fie spre interiorul volumului dacă ordinea punctelor (pentru un observator plasat în exteriorul feței) se stabilește astfel încît  $F_i$  să rămînă în stînga, spre exteriorul volumului în cazul contrar. În continuare, se va considera numai vectorul normal interior feței  $F_i$ , notat prin  $\bar{n}_i$

$$\bar{n}_{ijk} = \bar{n}_i = (\overline{N_j N_i} \times \overline{N_j N_k})$$

Testul algebric în raport cu zero al produsului scalar ( $PS$ ) dintre vectorul normal și vectorul asociat razei vizuale  $\overline{\Omega V_j}$  sau direcției  $\Delta(AL, BE, GA)$ .

$$\overline{P S} = \overline{\Omega V_j} \cdot \bar{n}_i \text{ sau } PS = \overline{\Delta} \cdot \bar{n}_i$$

poate fi pozitiv și nul sau negativ după cum fața vizată este către sau dinspre observator.

Trecînd pe planul de proiecție produsul scalar ( $PS$ ) poate fi înlocuit prin produsul mixt  $PV$ .

$$PV = \bar{k}_1 \cdot (\overline{V_j V_i} \times \overline{V_j V_k}) = \overline{K_1} \cdot (\overline{S_{ji}} \times \overline{S_{jk}})$$

sau

$$PV = \begin{bmatrix} 0 & 0 & 1 \\ XP_i - XP_j & YP_i - YP_j & 0 \\ XP_i - XP_j & YP_k - XP_j & 0 \end{bmatrix} = \begin{bmatrix} XP_i - XP_j & YP_i - YP_j \\ XP_k - XP_j & YP_k - YP_j \end{bmatrix}$$

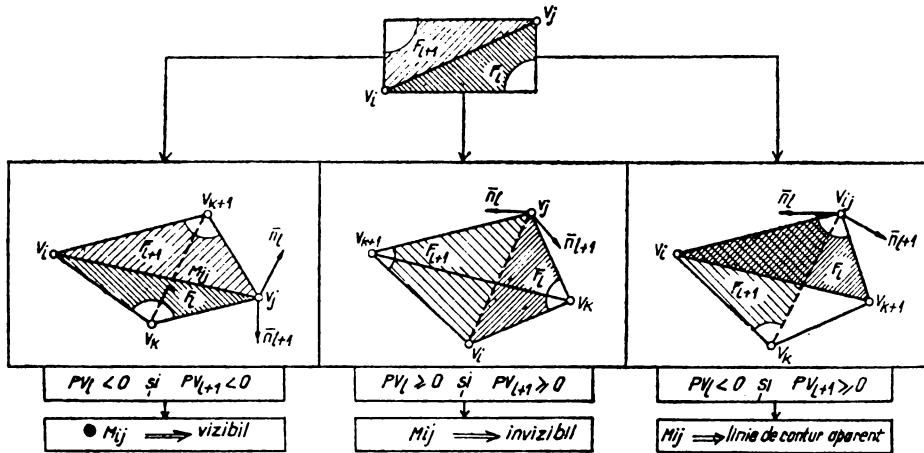
TESTUL PV PENTRU DOUĂ FEȚE ADIACENTE  $F_i$  ȘI  $F_{i+1}$ 

Fig. 7.3.

Conservînd convenția alegerii sistemului  $(XP, YP)$ , testul algebric al produsului scalar poate fi negativ și, nul sau pozitiv după cum fața vizată este vizibilă sau invizibilă (ascunsă).

Testul  $PV$  pentru două fețe adiacente  $F_i$  și  $F_{i+1}$  poate conduce la următoarele situații posibile, concentrate în figura 7.3.

În cazul volumelor poliedrale convexe fețele și deci și muchiile aparținînd acestor fețe sînt fie total vizibile fie total invizibile. Prin urmare muchiile aparținînd unei fețe vizibile sînt vizibile și deci vor fi trasate prin linii continui. În cazul contrar pot fi omise sau trasate prin linii discontinui sau cu altă culoare.

## 7.3.1. Codificarea algoritmului ALPLA

Algoritm cuprinde, în general, trei faze: proiectarea volumului convex pe tabloul de perspectivă, testarea vizibilității și trasarea segmentelor vizibile. Dacă se specifică mai multe puncte de vedere, perspectiva obiectului se trasează pentru fiecare în parte prin reluare automată.

*Semnificația unor variabile din program:*

- $PROP = 1$   $ORTOG = 0$  — proiecție paralelă;  
 $ORTOG = 1$  — proiecție ortogonală paralelă  
 $PROP = 0$   $ORTOG = 0$  — proiecție centrală  
 $TRASI = 1$  — se trasează și segmentele invizibile (cu altă culoare sau punctat);  
 $CONVEX = 0$  — structura conține poliedre închise concave (există fețe parțial vizibile);  
 $CONVEX = 1$  — structura este în întregime convexă (toate fețele parțial vizibile sînt complet vizibile);  
 $CONVEX = 2$  — structura conține corpuri neînchise topologic pentru care nu se poate stabili interiorul);  
 $LISTD = 1$  — se listează datele și rezultatele parțiale



<i>NV</i>	— numărul de vîrfuri ale structurii
<i>NF</i>	— numărul de fețe ale structurii
<i>NPV</i>	— numărul de puncte de vedere
<i>XVO, YVO, ZVO</i>	— coordonatele unui punct de vedere
<i>XC, YC, ZC</i>	— coordonatele centrului geometric al structurii
<i>XN, YN, ZN</i>	— vectori cu coordonatele spațiale corespunzătoare ale vîrfurilor structurii
<i>NVF(I)</i>	— vector ce menține numărul de puncte pentru fiecare față a structurii
<i>MVF(i, j)</i>	— matrice ce menține pentru fiecare față <i>i</i> ordinea topologiei de legare a vîrfurilor acesteia
<i>NVFI(I), NVFV</i>	— vector ce menține numărul de puncte pentru fiecare față invizibilă respectiv vizibilă;
<i>MVFI(i, j), MVFV(i, j)</i>	— matrici ce conțin descrierile pentru fețele invizibile respectiv vizibile
<i>XP(I), YP(I)</i>	— vectori ce mențin coordonatele corespunzătoare ale vîrfurilor în planul de proiecție;
<i>INTERS = 1</i>	— există 2 poliedre intersectabile în cadrul structurii
<i>NV1</i>	— delimitator (în cazul <i>INTERS = 1</i> ) pentru numărul de vîrfuri al primului corp.
<i>NF1</i>	— același lucru pentru fețe
<i>NS-k</i>	— numărul de segmente dublate în cadrul definirii fețelor cu găuri
<i>NSG(I)</i>	— vector ce conține o descriere a segmentelor specificate de NS.

Conservînd noțiunile definitorii cunoscute (7.1.1), pentru configurația geometrică (eventual un ansamblu de obiecte sau de volume de arhitectură) definită prin nodurile  $N_i$  ( $XN(I)$ ,  $YN(I)$ ,  $ZN(I)$ ) și fețele  $F_j$ , centrul geometric  $C(XC, YC, ZC)$  și punctul inițial de vedere  $\Omega(XVO, YVO, ZVO)$  sau direcția de vedere  $\Delta(AL, BE, GA)$  vom stabili algoritmul general codificabil pentru reprezentarea automată a succesiunii proiecțiilor configurației geometrice, astfel definite, pe un plan de proiecție (tablou de perspectivă).

### 7.3.2. Subprogramul DIORTO

*Datele inițiale pentru proiecția ortogonală*

Are drept scop determinarea componentelor vectorilor cosinuşilor directori ai direcțiilor de vedere în cazul proiecției ortogonale pe trei plane de proiecție și se utilizează astfel:

**CALL DIORTO (XV, YV, ZV, XC, YC, ZC, XMAX, YMAX, ZMAX)**

unde  $XV, YV, ZV$  sînt vectori cu dimensiunea 3 conținînd valorile cosinuşilor directori ai direcțiilor de proiectare pe cele trei plane de proiecție.

Activarea subprogramului este realizată de valoarea opțională atribuită variabilei întregi *ORTOG*.

### 7.3.3. Determinarea poziției relative a planului de proiecție (tabloul de perspectivă)

Schimbarea punctului sau direcției de vedere modifică poziția planului de proiecție, admis perpendicular pe raza sau pe direcția de vedere. Calculul parametrilor ce definesc poziția planului de proiecție în raport cu triedrul dreptunghiular  $OXYZ$  se realizează cu subprogramul **DATEIN** (**DATE** Inițiale).

### 7.3.4. Subprogramul DATEIN

*Poziționarea tabloului de perspectivă*

Are drept scop poziționarea planului de proiecție prin determinarea coeficienților ecuației sale, originii și cosinuşilor directori ai sistemului  $O1XPYP$  pentru fiecare din punctele sau direcțiile de vedere, și se utilizează astfel:

**CALL DATEIN (XO, YO, ZO, XC, YC, ZC, A, B, C, P, Y2, A1, B1, A2, B2, G2)**

unde:  $XO, YO, ZO$  — coordonatele punctului de vedere curent;  
 $XC, YC, ZC$  — coordonatele centrului geometric al volumului  
 $A, B, C, P$  — coeficienții planului de proiecție;  
 $Y2$  — depărtarea originii sistemului  $O1XPYP$   
 $A1, B1, A2, B2, G2$  — cosinuşii directori ai axelor  $O1XP$  și  $O1YP$

Observație:

Este luată în considerare numai perspectiva la nivelul observatorului sau descendentă ca fiind în general cea mai utilizată.

Activarea subrutinii se efectuează la fiecare ciclu al punctelor sau direcțiilor de vedere.

### 7.3.5. Determinarea proiecțiilor nodurilor $N_i$ pe planul de proiecție

Comoditatea studiului vizibilității prin utilizarea coordonatelor vîrfurilor  $V_i$ , obținute prin proiectarea pe planul de proiecție a vîrfurilor de definiție  $N$ , impune mai întii determinarea perechilor de valori  $(XP(I), YP(I))$ .

Problema se reduce la stabilirea legăturii dintre spațiul  $S^{(3)}$ , în care este definită configurația și spațiul  $S^{(2)}$  reprezentat prin planul de proiecție. Această legătură se exprimă prin următoarea expresie scrisă matriceal:

$$\begin{bmatrix} XP \\ YP \\ ZP \end{bmatrix} = \begin{bmatrix} A1 & B1 & 0 \\ A2 & B2 & C2 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y - Y2 \\ Z \end{bmatrix}$$

În funcție de sistemul de proiecție considerat, centrală sau paralelă, coordonatele vîrfurilor proiecțiilor au următoarele expresii:

$$X, Y, Z = \frac{F(N_i)}{A \cdot P_{ix} + B \cdot P_{iy} + C \cdot P_{iz}} \cdot P_{ix}, P_{iy}, P_{iz}$$

$$X, Y, Z = \frac{F(N_i)}{A \cdot AL + B \cdot BE + C \cdot GA} \cdot AL, BE, GA$$

unde s-a notat prin  $P_{ix}$ ,  $P_{iy}$  și  $P_{iz}$  proiecțiile segmentului variabil  $\Omega N_i$  pe axele sistemului de referință  $OXYZ$  iar prin  $F(N_i)$ , când nu sînt luate în considerare perspectivele pe plane sferice sau cilindrice, ecuația planului de proiecție  $F(X, Y, Z) = AX + BY + CZ + P$  în care  $X, Y$  și  $Z$  sînt înlocuiți prin  $XN(I)$ ,  $YN(I)$  și  $ZN(I)$ .

### 7.3.6. Subprogramul DMIMA

Are drept scop determinarea componentelor minime și maxime ale vectorilor  $XP$  și  $YP$  și, se utilizează astfel:

<b>CALL DMIMA (NV, XP, XM, XMA)</b>	și	<b>CALL DMIMA (NV, YP, YM, YMA)</b>
-------------------------------------	----	-------------------------------------

unde  $XM(IN)$ ,  $XMA(X)$ ,  $YM(IN)$ ,  $YMA(X)$  sînt valorile minime și maxime ale componentelor vectorilor  $XP$  și  $YP$  și servesc la încadrarea corespunzătoare a reprezentării pe formatul folosit.

### 7.3.7. Selectarea fețelor pseudofrontale

Numim *fețe pseudofrontale* acele fețe care privesc de către observator lasă în spate volumul geometric considerat.

Eliminarea fețelor invizibile conduce la reducerea substanțială a timpului computer întrucît trasarea liniilor ce le definesc poate fi omisă sau executată diferențiat dar fără a mai fi considerate independent la studiul vizibilității.

O față este pseudofrontală dacă scalarul  $PV$ , definit astfel:

$$PV = \begin{vmatrix} XP_i - XP_j & YP_i - YP_j \\ XP_k - XP_j & YP_k - YP_j \end{vmatrix} = \text{negativ}$$

unde  $i, j$  și  $k$  sînt trei vîrfuri consecutive ale feței testate, este negativ.

### 7.3.8. Creerea vectorului liniilor fețelor pseudofrontale și selectarea liniilor nerepetitive

Pentru a evita parcurgerea dublă la trasarea segmentelor ce definesc fețele pseudofrontale adiacente, în cazul configurațiilor convexe și a ușura explorarea liniilor de definiție, în cazul configurațiilor concave, se impune creerea vectorului liniilor fețelor pseudofrontale și apoi selectarea segmentelor ce nu se repetă. Această operație este realizată cu subprogramele **LINFV** (**LINI**ile **Fețelor Vizibile**) și **LSELET** (**Linii SELECT**ate).

### 7.3.9. Subprogramul LINFV

Are drept scop liniarizarea descrierii fețelor și crearea vectorului de segmente. Pentru cazul cu  $INTERS = 1$  completează pentru fiecare segment în matricea **MATF** o intrare cu numărul feței din care face parte acest segment.

Apelul este de tipul următor:

<b>CALL LINFV (MVF, NVF, IFI, LF, MATLIN, MATF, NRSEG, INTERS)</b>
--

**MATLIN** este vectorul de segmente și **NRSEG** un indicator cu numărul de segmente din **MATLIN**. Se liniarizează segmentele fețelor din **MVF** între indicii **IFI** și **LF** și se actualizează dacă este cazul **MATF**.

### 7.3.10. Subprogramul LSELET

Are drept scop eliminarea segmentelor cu dublă apariție din vectorul de segmente pentru a se evita testarea vizibilității lor de 2 ori. Se completează în cazul  $INTERS = 1$  și informația din  $MATF$  cu noua față din care face parte un segment cu dublă apariție. Tot acum se introduc în vectorul de segmente și segmentele dublate în cadrul descrierii fețelor cu găuri.

Apelul este de tipul următor:

**CALL LSELET (MATLIN, MATF, NRSEG, NRLIN, INTERS, NS, NSG)**

$NRLIN$  = numărul de segmente total după eliminarea aparițiilor duble) din vectorul de segmente  $MATLIN$ .

### 7.4. Plotarea configurațiilor convexe

Valoarea variabilei întregi  $CONVEX$  decide activarea fie a subprogramului  $VIZLIN$  fie a subprogramului  $TRASA$ . Pentru configurațiile convexe toate segmentele fețelor pseudofrontale sînt vizibile și deci se activează cel de al doilea subprogram.

#### 7.4.1. Subprogramul TRASA

*Trasarea segmentelor grupate în vectorul liniilor vizibile*

Are drept scop trasarea segmentelor grupate în vectorul  $LINV$  și se utilizează astfel:

**CALL TRASA (LINV, LVS, XP, YP)**

Efectul activității acestei subrutine este trasarea tuturor fețelor ale căror segmente de definiție sînt în vectorul  $LINV$ . Independent de tipul configurației convexe sau concave, trasarea și a segmentelor aparținînd fețelor invizibile decise de valoarea atribuită variabilei întregi  $TRASI$ , antrenează reactivarea subprogramelor  $LINFV$  și  $LSELET$  dar de această dată cu referire la vectorul bidimensional  $MVFI$  al fețelor invizibile.

Evitarea parcurgerii și a segmentelor de intersecție a două fețe adiacente dintre care una este invizibilă impune selectarea segmentelor dintre două fețe adiacente, ambele invizibile, operație efectuată de subprogramul  $LIR$  (Linii Invizibile Rămase).

#### 7.4.2. Subprogramul LIR

Are drept scop compararea elementelor a doi vectori unidimensionali reținînd din al doilea numai pe acelea negăsite în primul.

Se utilizează astfel:

**CALL LIR (LINV, LINI, LVS, LIS, LR)**

unde  $LINI$  — vectorul unidimensional ale cărui elemente reprezintă numerele asociate segmentelor de definire aparținînd fețelor invizibile. Are dimensiunea  $LR$ .

$LR$  — reprezintă contorul. elementelor negăsite în vectorul  $LINV$ .

Efectul reactivării corespunzătoare a subprogramului **TRASA** este trăsarea diferențiată și a segmentelor invizibile.

În acest punct configurația convexă este trasată în întregime; liniile vizibile cu negru și segmentele invizibile cu roșu de exemplu.

## 7.5. Plotarea configurațiilor concave

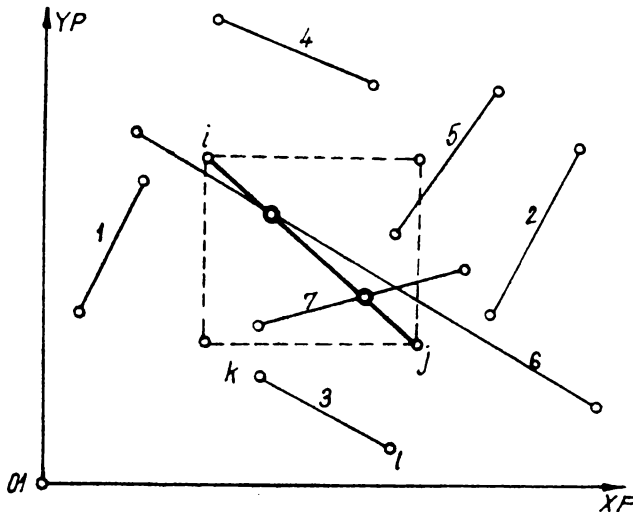
Deoarece pentru aceste configurații unele din fețele pseudofrontale pot fi și parțial invizibile se impune efectuarea unui studiu al vizibilității segmentelor de definiție a fețelor pseudofrontale.

Întrucât unul din scopurile urmărite de algoritm este acela de a reduce pe cât posibil timpul computer, el are la bază următoarele observații:

Numai segmentele de definiție a fețelor pseudofrontale (vectorul  $LINV$ ) fac obiectul studiului vizibilității în raport cu aceste fețe (vectorul  $MVFV$ ).

Între punctele de intersecție ale unui segment cu celelalte segmente proiectate vizibilitatea nu se schimbă și deci pentru a studia vizibilitatea segmentului explorat se studiază vizibilitatea unui punct arbitrar situat între două puncte de intersecție succesive. În acest sens se crează, pentru segmentul explorat, vectorul unidimensional  $U$  ale cărui elemente reprezintă valorile unui parametru  $UP$ , obținut prin intersecția acestui segment cu restul segmentelor, cuprinse între 0 și 1. Sînt parcurse următoarele etape:

Se rețin pentru intersecție, cu segmentul explorat celelalte segmente ce se suprapun parțial sau total cu suprafața delimitată a dreptunghiului din figura 7.4 avînd laturile paralele cu sistemul  $O1XPYP$ .



**INTERSECȚIA SEGMENTULUI EXPLORAT  $ij$  CU  
SEGMENTELE REȚINUTE PENTRU INTERSECȚIE (5, 6 ȘI 7)**

Fig. 7.4

Se intersectează segmentul explorat cu segmentele reținute mai sus prin activarea subprogramului **DILP** (Determinarea Intersecțiilor Liniiilor Proiectate) reținându-se numai valorile pentru care parametrii  $UP$  și  $V$  îndeplinesc condiția 1 din figura 7.5., obținându-se vectorul  $U$ .

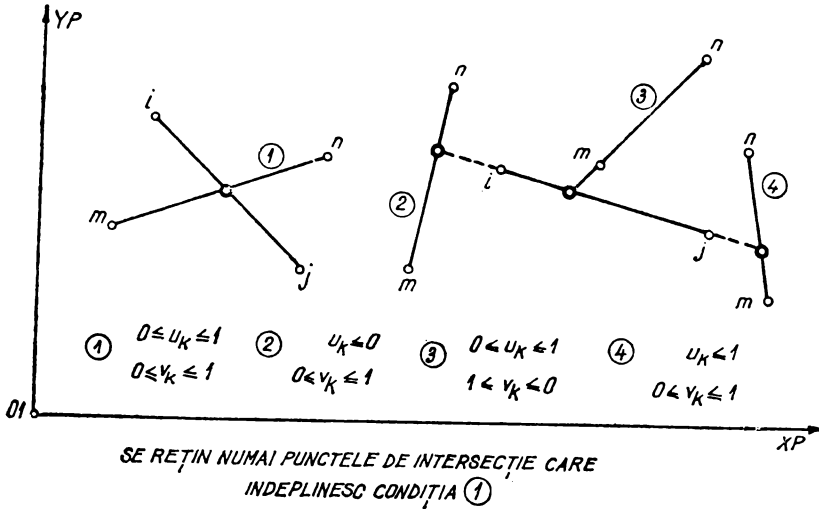


Fig. 7.5

Elementele vectorului  $U$  (fig. 7.6) sînt ordonate crescător prin activarea subprogramului **ORD** (ORDOnare).

Observînd că dacă testul vizibilității unui punct, situat între două puncte de intersecție succesive  $U(I)$  și  $U(I + 1)$  indică vizibilitatea punctului, seg-

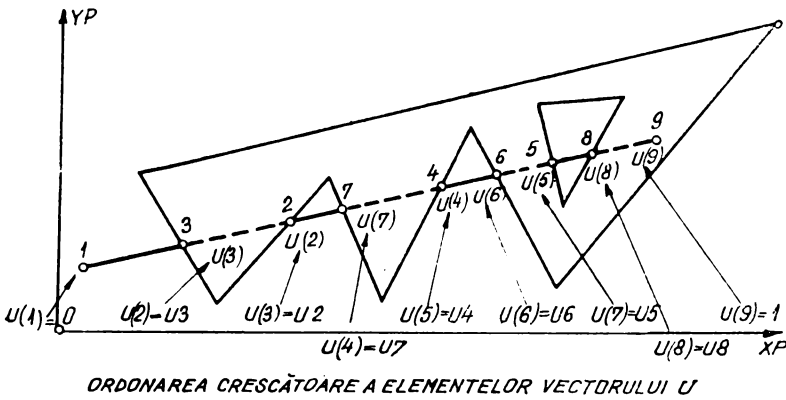


Fig. 7.6

mentul  $U(I) U(I + 1)$  este vizibil; se cercetează vizibilitatea acestui punct în raport cu toate fețele pseudofrontale astfel:

Un punct situat între două puncte succesive de intersecție se testează cu o față pseudofrontală dacă și numai dacă proiecția sa aparține feței respective. Condiția de apartenență este testată prin paritatea numărului intersecției unei drepte, unind punctul vizat cu un punct exterior conturului aparent al configurației, cu segmentele ce definesc fața respectivă. Un punct a cărui proiecție aparține proiecției unei fețe conduce la un număr impar de intersecții (vezi 6.3).

Pentru punctul satisfăcând condițiile de mai sus se determină poziția sa pe muchia corespunzătoare din spațiu.

Punctul este vizibil în raport cu fața testată dacă este satisfăcută condiția  $S \geq 1$  pentru proiecția centrală și  $S \leq 0$  pentru proiecția paralelă,  $S$  fiind parametrul razei de vedere  $\vec{R}_c = \vec{R}_i + s\vec{R}_{ji}$  sau parametrul direcției de vedere  $\vec{R}_p = \vec{R}_j + s\vec{\Delta}$  (fig. 7.7.).

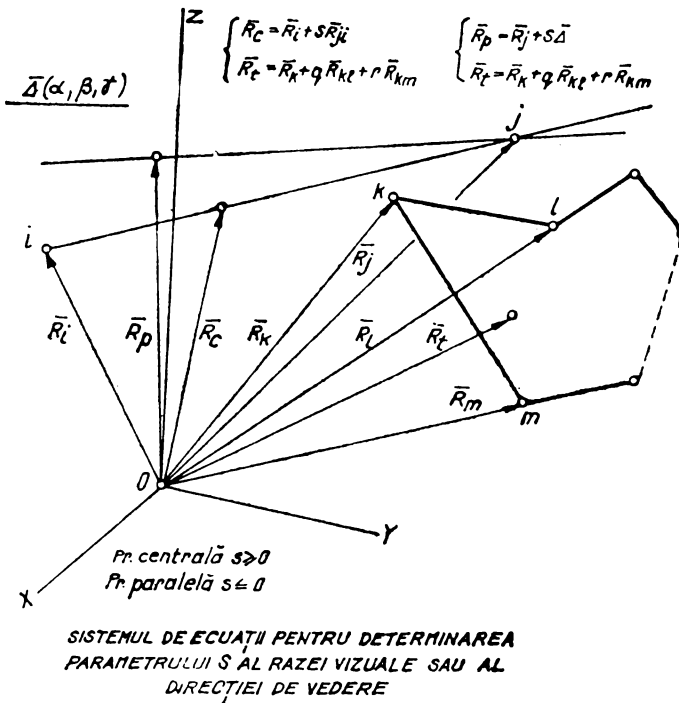


Fig. 7.7

Parametrul  $s$  se determină din sistemele de ecuații, care pentru simplitate sînt scrise vectorial:

$$\begin{aligned} \vec{R}_c &= \vec{R}_i + s\vec{R}_{ji} & \vec{R}_p &= \vec{R}_j + s\vec{\Delta} \\ \vec{R}_i &= \vec{R}_k + q\vec{R}_{kl} + r\vec{R}_{km} & \vec{R}_l &= \vec{R}_k + q\vec{R}_{kl} + r\vec{R}_{km} \end{aligned}$$

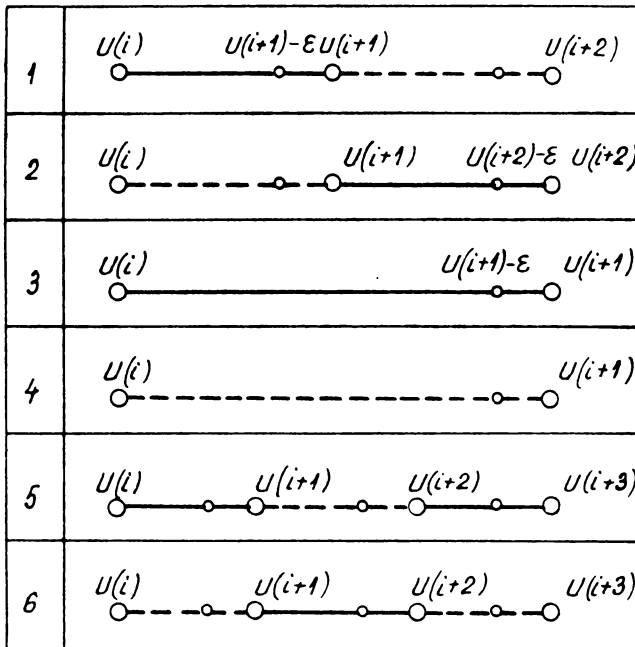
unde indicii  $k, l, m$  se referă la trei vîrfuri oarecare ale feței pseudofrontale vizate. Celelalte notații au semnificația din figură.

Toate observațiile de mai sus sînt incluse în subprogramul **VIZibilitatea LINiilor (VIZLIN)**.

### 7.5.1. Subprogramul VIZLIN

*Realizarea vizibilității segmentului explorat*

Are drept scop studierea vizibilității unor segmente și le trasează corespunzător; dacă punctul median a două puncte de intersecție succesive este vizibil segmentul dintre cele două puncte este vizibil și se trasează cu liniile continuă grosă; în cazul contrar se omite sau se trasează diferit (fig. 7.8). Subprogramul se utilizează astfel:



DACĂ PUNCTUL MEDIAN DINTRE DOUĂ PUNCTE DE INTERSECȚIE SUCCESIVE ESTE VIZIBIL ATUNCI SEGMENTUL ESTE VIZIBIL ÎNTRE CELE DOUĂ PUNCTE

Fig. 7.8

**CALL VIZLIN (XP, YP, LINV, XM, NVFV, MVFV, XN, YN, ZN, LVS, KV, A1, A2, B1, B2, G2, XO, YO, ZO, Y2, YM, SC, MPINT, INVSEG, COEFIN, NPINT, NRS, G, INTSG).**



Prin activarea subprogramului **VIZLIN** se obține reprezentarea configurației considerate exclusiv a segmentelor de intersecție dintre două fețe adiacente invizibile. Trasarea și a acestora din urmă este decisă de valoarea opțională a variabilei *TRASI*.

## 7.6. Subprogramele de adaptare ale algoritmului în intersecția dintre două poliedre oarecare convexe, concave, cu goluri sau deschise

### 7.6.1. Subprogramul CINT

Cu ajutorul acestui subprogram obținem matricea punctelor de intersecție dintre fețele unui poliedru (corp geometric) și muchiile celui alt poliedru (corp geometric).

Instrucțiunea de apel a subprogramului este următoarea:

**CALL CINT (NFI, NFF, MVF, NVF, P, Q, R, NRPTI, MPINT, COEFIN, NPINT, MATLIN, MATF, NRLIN)**

unde semnificația parametrilor este:

- NFI, NFF* — Indici inițial și final în matricea fețelor care delimitează fețele unui corp
- MVF* — Matricea fețelor
- NVF* — Vectorul care menține numărul de puncte ale fiecărei fețe.
- P, Q, R* — Vectorii de coordonate pentru punctele tuturor fețelor.
- NRPTI* — Indice de la care se introduc în *P, Q, R* noile puncte rezultate prin intersecția față-segment.
- MPINT* — Matricea punctelor de intersecție față-segment.
- COEFIN* — Matricea coeficienților de intersecție
- NPINT* — Indice de la care se introduc punctele de intersecție în *MPINT*
- MATLIN* — Vectorul liniarizat al segmentelor unice din care sînt alcătuite fețele.
- MATF* — Matricea care reține pentru fiecare segment, din ce fețe face parte.
- NRLIN* — Numărul de elemente din *MATLIN*.

Așadar acest subprogram are ca scop creerea matricei *MPINT* ale cărei intrări sînt de forma, intrarea *K*:

*Fața 1 Fața 2 Fața 3 Ind. inițial Ind. final* în care se specifică faptul că punctul de intersecție *K* face parte din fețele: fața 1, fața 2, fața 3 și se găsește pe segmentul cu capetele (ind. inițial, indice final).

Subprogramul completează vectorii *P, Q, R* cu coordonatele spațiale ale punctelor de intersecție. Calculul punctelor de intersecție dintre fețele unui poliedru și muchiile celui alt poliedru se face prin apelul ciclic al subrutinei **FATASG**.

Listarea subprogramului **CINT** poate fi urmărită în programul **ALPLA**.

### 7.6.2. Subprogramul FATASG

Cu ajutorul acestui subprogram calculăm coordonatele punctelor de intersecție dintre o față a unui poliedru și o muchie a altui poliedru în cazul în care această intersecție există. Instrucțiunea de apel a subprogramului este următoarea:

**CALL FATASG (J, K, MVF, XF, YF, ZF, XI, YI, ZI, P, Q, R, X, Y, Z, T4, CK)**

unde semnificația parametrilor este:

$J$	— numărul intrării din matricea fețelor a feței ce trebuie intersectată cu un segment;
$K$	— numărul de puncte al poligonului ce definește fața $J$ .
$\begin{cases} XF, YF, ZF \\ XI, YI, ZI \end{cases}$	— coordonatele inițiale și finale ale segmentului de intersecție
$P, Q, R, MVF$	— Identice cu semnificațiile din subprogramul <b>CINT</b>
$X, Y, Z$	— Coordonatele spațiale ale punctului de intersecție
$T4 = 1$	— Există intersecție între față și segment
$T4 = 0$	— Nu există intersecție între față și segment
$CK$	— Coeficientul de intersecție

Cu ajutorul acestui subprogram putem calcula punctul de intersecție al unui segment de dreaptă cu o față a poliedrului utilizând proprietatea legată de suma unghiurilor orientate ale unui punct interior unui contur poligonal plan închis, sumă care este egală cu  $2\pi$  dacă punctul testat este în interiorul conturului (feței poliedrului) (vezi 6.3)

Subprogramul testează de asemenea și apartenența punctului pe frontiera conturului poligonal, caz în care suma unghiurilor orientate este egală cu  $\pi$ . În toate celelalte cazuri suma unghiurilor tinde spre zero.

### 7.6.3. Subprogramul ESPI

Cu ajutorul acestui subprogram extragem segmentele care alcătuiesc poligonul strâmb de intersecție dintre corpurile poliedrale. Instrucțiunea de apel a subprogramului este:

**CALL ESPI (MPINT, NPI, INSEG, NRSRG, NPDEF, MATF, P, Q, R)**

unde semnificația parametrilor este:

$MPINT$	— Matricea punctelor de intersecție
$NPI$	— Numărul de puncte din $MPINT$
$INSEG$	— Vectorul linearizat al segmentelor poligonului strâmb de intersecție
$NRSRG$	— Număr de intrări din $INSEG$
$NPDEF$	— Numărul de puncte inițial al vectorilor $P, Q, R$ .
$MATF$	— Matricea fețelor.

Așadar acest subprogram are rolul de a crea vectorul *INSEG* care conține segmentele poligonului de intersecție dintre corpuri (eventual ale poligoanelor) folosind un algoritm exhaustiv diferit de cel din programul *INTPOL*. Astfel se consideră că un segment de dreaptă face parte din poligonul de intersecție, dacă dintre cele trei fețe a căror intersecție o reprezintă fiecare din capetele lui, două sînt comune dar cele două puncte nu se găsesc pe un segment deja existent.

## 7.7. Program principal ALPLA (Listare)

### DATE DE INTRARE

PROP, ORTOG, TRASI, NV, NF, NPV, CONVEX, LISTD, IORT

XO, YO, ZO, XC, YC, ZC

NV	Nr. total virfuri	NF	Nr. total fețe	NPV	Nr. total puncte de vedere
Prop	0 Proiecție centrală 1 Proiecție paralelă	00 – Centrală 10 – Paralelă	Convex	0 – Concav 1 – Convex	} Corpuri închise
NS	0 – Corpul nu are goluri nr – Număr de segmente pentru care nu se testează vizibilitatea în cazul corpurilor cu ferestre (goluri)				

XO, YO, ZO – Coordonatele punctului de vedere

LISTD – listează sau nu datele

HC, YC, ZC – Coordonatele centrului geometric

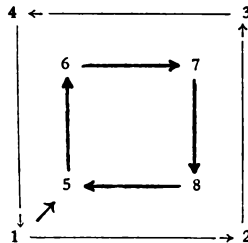
TRASI – trasează sau nu liniile ascunse

INTERS, NV1, NF1, NS

INTERS	1 – intersecție de corpuri	NV1 – număr de virfuri al primului corp
	0 – nu este intersecție	NF1 – număr de fețe al primului corp

Se mai introduc dacă este cazul segmentele pentru NS

(1, 5, ..., )



```

LOGICAL*1 LLL(80),KRT(80)
INTEGER PROP,ORTOG,TRASI,CONVEX
COMMON PROP,ORTOG,TRASI,XX,YY,XM,YM
DIMENSION XN(200),YN(200),ZN(200),XV(10),YV(10),ZV(10),MVF(40,15)
1,MVF(100),XP(200),YP(200),MVFV(40,15),MVF1(40,15),MVFV(100),MVF1(1
200),LINV(200),LINI(200),U(50),MATF(2,100),MPINT(5,50),INSEG(50),
3COEFIN(50),MATLIN(200),NSG(100)
XX=0.
ND=9
TYPE 5000
5000 FORMAT(/' ','FISIERUL DE DATE : ','*)
ACCEPT 5001,NL,LLL
5001 FORMAT(Q,R0A1)
CALL ASSIGN(4,LLL,NL)
CALL FORSET(4,'R')
CALL ASSIGN(2,'LP:')
ITIP=7

C
C   ...CITESTE SI SCRIE TABLOUL CU DATELE PROBLEMEI...
C
95 READ(4,100,END=90) PROP,ORTOG,TRASI,MV,NF,NPV,CONVEX,LISTD
READ(4,101) XV0,YV0,ZV0,XC,YC,ZC
READ(4,103) (XN(I),YN(I),ZN(I),I=1,NV)
READ(4,104) (MVF(I),I=1,NF)
IF (LISTD.EQ.0) GOTO 66
WRITE(2,200)
WRITE(2,100) PROP,ORTOG,TRASI,MV,NF,NPV,CONVEX,LISTD
WRITE(2,101) XV0,YV0,ZV0,XC,YC,ZC
WRITE(2,103) (XN(I),YN(I),ZN(I),I=1,NV)
WRITE(2,104) (MVF(I),I=1,NF)
200 FORMAT(1H0,18X,27HTABLOUL CU DATELE PROBLEMEI/17X,27(1H*)//)
100 FORMAT(8I10)
101 FORMAT(6F10,4)
102 FORMAT (10X,6F10,1)
103 FORMAT(12F6,1)
104 FORMAT(10X,30I2)
66 DO 10 I=1,NF
IN=MVF(I)
READ(4,105) (MVF(I,J),J=1,IN)
IF (LISTD.EQ.0) GOTO 10
WRITE(2,105) (MVF(I,J),J=1,IN)
105 FORMAT(10X,20I3)
10 CONTINUE
DV=6,2H/NPV
NPINT=0
READ(4,100) INTERS,NV1,NF1,NS
IF (NS.EQ.0) GOTO 2345
READ(4,104) (NSG(2*I-1),NSG(2*I),I=1,NS)
2345 IF (INTERSE.NE.1) GOTO 93
NV2=N1-NV1
NF2=NF1-NF1
CALL LINV(MVF,NVF,NF1+1,NF,MATLIN,MATF,NRSEG,1)
81 FORMAT(' ',I3,5X,I3,5X,I3)
CALL LSELET(MATLIN,MATF,NRSEG,NRLIN,1,NS,NSG)
WRITE(2,81) (I6,MATLIN(2*I6-1),MATLIN(2*I6),I6=1,NRLIN)
NRPTI=NV

```

```

CALL CINT(1,NF1,MVF,NVF,XN,YN,ZN,NRPTI,MPINT,COEFIN,NPINT,MATLIN,
2ATF,NRLIN)
WRITE(2,83)(II,(MPINT(JJ,II),JJ=1,5),II=1,NPINT)
83  FORMAT(' ',I2,5I5)
CALL LINFV(MVF,NVF,1,NF1,MATLIN,MATF,NRSEG,1)
CALL LSELET(MATLIN,MATF,NRSEG,NRLIN,1,NS,NSG)
WRITE(2,81)(I6,MATLIN(2*I6-1),MATLIN(2*I6),I6=1,NRLIN)
CALL CINT(NF1+1,NF,MVF,NVF,XN,YN,ZN,NRPTI,MPINT,COEFIN,NPINT,MATLI
2N,MATF,NRLIN)
WRITE(2,83)(II,(MPINT(JJ,II),JJ=1,5),II=1,NPINT)
CALL ESPI(MPINT,NPINT,INSEG,NRSEG,NV,MATF,XN,YN,ZN)
WRITE(2,82)(II,(MPINT(JJ,II),JJ=1,5),XN(NV+II),YN(NV+II),
1ZN(NV+II),II=1,NPINT)
82  FOMHAT(' ',I2,5I5,6X,3F8.3)
93  IF(ORTOG.NE.1) GOTO 11
CALL DMIMA(NV,XN,XNMIN,XNMAX)
CALL DMIMA(NV,YN,YNMIN,YNMAX)
CALL DMIMA(NV,ZN,ZNMIN,ZNMAX)
CALL DIORTO(XV,YV,ZV,XC,YC,ZC,XNMAX,YNMAX,ZNMAX)
GO TO 14
11 CONTINUE
R1=XV0-XC
R2=YV0-YC
DO 1 I=1,NPV
F=(I-1)*DV
XV(I)=XC+R1*COS(F)-R2*SIN(F)
YV(I)=YC+R2*COS(F)+R1*SIN(F)
1 ZV(I)=ZV0
14 CONTINUE
C
C   ...INTER SECTII FETE
C
C   ...INCEPE CICLAREA PUNCTELOR DE VEDERE...
IF(LISTD.EQ.0) GOTO 67
WRITE(2,102)(XV(I),YV(I),ZV(I),I=1,NPV)
67  DO 60 J=1,NPV
X0=XV(J)
Y0=YV(J)
Z0=ZV(J)
CALL DATEIN(X0,Y0,Z0,XC,YC,ZC,A,B,C,P,Y2,A1,B1,A2,B2,G2)
IF(LISTD.EQ.0) GOTO 68
WRITE(2,206)
206 FORMAT(1H0,14X,32HTABLOUL CU REZULTATELE PROBLEMEI/15X,32(1H*)//2X
1,1HI,17X,5HXP(I),17X,5HYP(I)//)
WRITE(2,101) X0,Y0,Z0,A,B,C,P,Y2,A1,B1,A2,B2,G2
C
C   ...INCEPE CICLAREA PENTRU CALCULUL PROIECTIEI...
C
NVT=NV+NPINT
68  DO 25 I=1,NVT
DNR=A*XN(I)+B*YN(I)+C*ZN(I)+P
IF(ORTOG.EQ.1) GOTO 77
IF(PROP.NE.1) GO TO 15
77  AL=A
BE=B
GA=C

```

```

LAM=DNR/(A*AL+B*BE+C*GA)
X=XN(I)-AL*LAM
Y=YN(I)-BE*LAM
Z=ZN(I)-GA*LAM
GO TO 20
15 T1=X0-XN(I)
T2=Y0-YN(I)
T3=Z0-ZN(I)
DVN=SQRT(T1*T1+T2*T2+T3*T3)
CL=T1/DVN
CM=T2/DVN
CN=T3/DVN
L2=DNR/(A*CL+B*CM+C*CN)
X=XN(I)-CL*LAM
Y=YN(I)-CM*LAM
Z=ZN(I)-CN*LAM
20 P1=X
P2=Y-Y2
P3=Z
XP(I)=A1*P1+B1*P2
YP(I)=A2*P1+H2*P2+G2*P3
25 CONTINUE
CALL DMIMA(NV,XP,XM,XMA)
CALL DMIMA(NV,YP,YM,YMA)
DO 27 I=1,NV
YP(I)=YP(I)-YM
27 XP(I)=XP(I)-XM
SX=25./(XMA-XM)
SY=18./(YMA-YM)
SC=AMIN1(SX,SY)
TYPE 5002,SC
5002 FORMAT(' ','INTRODUCETI SCARA(',F6.2,',)NEW F6.2)',',S)
ACCEPT 5003,SCAR
5003 FORMAT(F6.2)
IF(SCAR,NE.0) SC=SCAR
C
C ...SCRIE COORDONATELE PROIECTIEI...
C
IF(LISTD.EQ.0) GOTO 69
DO 30 I=1,NVT
30 WRITE(2,201) I,XP(I),YP(I)
201 FORMAT(1H ,I4,2(11X,F10.2))
WRITE(2,202)
202 FORMAT(1H0,17X,25HMATRICEA FETELOR VIZIBILE/17X,25(1H*)//1X,3(1H.)
1,1HI,3(1H.),27X,15H...MVFV(I,J)...//)
C
C ...SE TESTEAZA VIZIBILITATEA FETELOR...
C
69 TYPE 89
89 FORMAT(' ','DEPLASAMENT PE X SI Y [2F5.1]: ',S)
ACCEPT 63,XX,YY
63 FORMAT(2F5.1)
IF(CONVEX.EQ.2) GOTO 177
KV=0.
KI=0.
DO 55 L=1,NF

```

```

I1=MVF(L,1)
I2=MVF(L,2)
I3=MVF(L,3)
D1=XP(I1)-XP(I2)
D2=YP(I3)-YP(I2)
D3=XP(I3)-XP(I2)
D4=YP(I1)-YP(I2)
PV=D1*D2-D3*D4
N1=NVF(L)
IF(PV) 35,45,45
35 KV=KV+1
   NVFV(KV)=NVF(L)
   DO 40 M=1,N1
40 MVFV(KV,M)=MVF(L,M)
   GO TO 55
45 KI=KI+1
   NVFI(KI)=NVF(L)
   DO 50 M=1,N1
50 MVFI(KI,M)=MVF(L,M)
55 CONTINUE
   GOTO 148
177 KV=NF
   DO 179 KP=1,NF
      N3=NVF(KP)
      NVFV(KP)=N3
      DO 179 NT1=1,N3
179 MVFV(KP,NT1)=MVF(KP,NT1)
148 IF(LISTD.EQ.0) GOTO 71
   DO 60 I=1,KV
      NVI=NVFV(I)
60 WRITE(2,203) I,(MVFV(I,M),M=1,NVI)
   203 FORMAT(I6,4X,10I5)
      IF(CONVEX.EQ.2) GOTO 71
      WRITE(2,204)
   204 FORMAT(1H0,16X,27HMATRICEA FETELOR INVIZIBILE/16X,27(1M*)//1X,7H..
1,I...20X,15H...MVF(I,J)...//)
      DO 65 I=1,KI
         NI=NVI(I)
65 WRITE(2,205) I,(MVFI(I,M),M=1,NI)
   205 FORMAT(I6,4X,10I5)
71 CALL LINFV(MVFV,NVFV,1,KV,LINV,MATF,LV,0)
   CALL LSELET(LINV,MATF,LV,LVS,0,NS,NS6)
   IF(INTER.S.NE.0) GOTO 993
   IF(CONVEX.EQ.1) GOTO 991
993 CALL VIZLIN(XP,YP,LINV,XM,NVFV,MVFV,XN,YN,ZN,LVS,KV,A1,A2,B1,B2,62
1,X0,Y0,Z0,Y2,YM,SC,MPINT,INSE0,COEFIN,NPINT,NRS0,INTER.S)
   GOTO 992
991 CALL TRASA(LINV,LVS,XP,YP,SC)
992 IF(TRASI.EQ.0.OR.CONVEX.EQ.2) GOTO 80
   CALL LINFV(MVFI,NVFI,1,KI,LINI,MATF,LI,0)
   CALL LSELET(LINI,MATF,LI,LIS,0,NS,NS0)
   CALL LIR(LINV,LINI,LVS,LIS,LR)
   LIS=LR-1
   CALL TRASA(LINI,LIS,XP,YP,SC)
80 CONTINUE
   GOTO 95
90 NBLOC=999
   CALL PLOT(0.,0.,999)
   CALL CLOSE(4)
   CALL CLOSE(1)
   STOP
   END

```

## 7.7.1. Subrutina DATEIN

```

SUBROUTINE DATEIN(X,Y,Z,U,V,W,A,B,C,P,E,F,G,H,O,R.
COMMON NP,NO,NN,XX,YY
D1=X-U
D2=Y-V
D3=Z-W
J=SQRT(D1*D1+D2*D2+D3*D3)
DP=SQRT(D1*D1+D2*D2)
A=D1/D
B=D2/D
C=D3/D
DC=U*U+V*V+W*W
DV=X*X+Y*Y+Z*Z
P=(DC-DV)/(2.*D)
IF(NO.NE.1) GOTO 60
P=.0
E=.0
F=0.
G=0.
H=0.
O=0.
R=0.
IF(A.EQ.1..AND.B.EQ.0..AND.C.EQ.0.) GO TO 30
IF(A.EQ.0..AND.B.EQ.1..AND.C.EQ.0.) GO TO 40
IF(A.EQ.0..AND.B.EQ.0..AND.C.EQ.1.) GO TO 50
30 G=1.
R=1.
RETURN
40 F=-1.
R=1.
RETURN
50 F=-1.
O=-1.
RETURN
60 CONTINUE
E=-P/B
F=ABS(D2)/DP
G=ABS(D1)/DP
H=ABS(G*C)
O=ABS(F*C)
R=SQRT(1.-C*C)
IF(C.LT.0.) GO TO 1
IF(A.GT.0..AND.B.GT.0.) GO TO 2
IF(A.LT.0..AND.B.GT.0.) GO TO 3
IF(A.LT.0..AND.B.LT.0.) GO TO 4
IF(A.GT.0..AND.B.LT.0.) GO TO 5
1 IF(A.GT.0..AND.B.GT.0.) GO TO 6
IF(A.LT.0..AND.B.GT.0.) GO TO 7
IF(A.LT.0..AND.B.LT.0.) GO TO 8
IF(A.GT.0..AND.B.LT.0.) GO TO 9
2 F=-F
H=-H
O=-O
RETURN
3 F=-F
G=-G
O=-O

```

continuă la pag. 31



```

RETURN
4 G=-G
RETURN
5 H=-H
RETURN
6 F=-F
RETURN
7 F=-F
G=-G
H=-H
RETURN
8 G=-G
H=-H
O=-O
RETURN
9 O=-O
RETURN
END

```

### 7.7.2. Subrutina DMIMA

```

SUBROUTINE DMIMA(N,T,DMI,DMA)
DIMENSION T(N)
DMI=T(1)
DMA=DMI
DO 3 I=2,N
IF(T(I).LT.DMI) GO TO 1
IF(T(I)-DMA) 3,3,2
1 DMI=T(I)
GO TO 3
2 DMA=T(I)
3 CONTINUE
RETURN
END

```

### 7.7.3. Subrutina LINFV

```

C SUBROUTINA LINFV PENTRU OBTINEREA VECTORULUI DE SEGMENTE DIN
C CARE SINT ALCATUITE FETELE CORPURILOR
SUBROUTINE LINFV(MVF,NVF,IF1,LF,MATLIN,MATF,NRSEG,INTERS)
DIMENSION MVF(40,15),NVF(1),MATLIN(1),MATF(2,100)
NRSEG=0
DO 2 I=IF1,LF
NR=NVF(I)-1
DO 1 J=1,2
LL=J+NR-1
DO 1 K=J,LL
IF(J.EQ.1.AND.INTERS.EQ.1)MATF(1,NRSEG+K)=I
1 MATLIN(2*NRSEG+2*K-J)=MVF(I,K)
2 NRSEG=NRSEG+NR
RETURN
END

```

## 7.7.4. Subrutina LSELET

```

C   SUBRUTINA CARE ELIMINA SEGMENTELE CU DURLA APARITIE
C   DIN VECTORUL OBTINUT DIN LINFV
SUBROUTINE LSELET(MATLIN,MATF,NRSEG,NRLIN,INTFPS,NS,NSG)
DIMENSION MATLIN(1),MATF(2,100),NSG(1)
NFIN=2*NRSEG-1
DO 3 I=1,NFIN,2
8   IF(I.GT.NFIN) GOTO 7
   I1=MATLIN(I)
   I2=MATLIN(I+1)
   IN=I+2
   DO 1 J=IN,NFIN,2
     J1=MATLIN(J)
     J2=MATLIN(J+1)
     IF(.NOT.((I1.EQ.J1.AND.I2.EQ.J2).OR.(I1.EQ.J2.AND.I2.EQ.J1)))
1    GOTO 1
     MATLIN(J)=MATLIN(NFIN)
     MATLIN(J+1)=MATLIN(NFIN+1)
     IF(INTFPS.NE.1) GOTO 4
     MATF(2,(I+1)/2)=MATF(1,(J+1)/2)
     MATF(1,(J+1)/2)=MATF(1,(NFIN+1)/2)
4    NFIN=NFIN-2
     IF(NS.EQ.0) GOTO 3
     DO 5 K=1,NS
       K1=NSG(2*K-1)
       K2=NSG(2*K)
       IF(.NOT.((I1.EQ.K1.AND.I2.EQ.K2).OR.(I1.EQ.K2.AND.I2.EQ.K1)))
1    GOTO 5
       MATLIN(I)=MATLIN(NFIN)
       MATLIN(I+1)=MATLIN(NFIN+1)
       IF(INTFPS.NE.1) GOTO 6
       MATF(1,(I+1)/2)=MATF(1,(NFIN+1)/2)
6    NFIN=NFIN-2
     GOTO 8
5    CONTINUE
1  CONTINUE
3  CONTINUE
2  NRLIN=I/2
   RETURN
   END

```

## 7.7.5. Subrutina CINT

```

C   SUBRUTINA CINT PENTRU OBTINEREA MATRICII PUNCTELOR DE INTERSECIE
C   INTRE FELELE UNUI CORP SI MUCHIILE CELUIALTEI CORP
SUBROUTINE CINT(NFI,NFF,MVF,NVF,P,Q,R,NKPTI,MPINT,COEFIN,MPINT,
2MATLIN,MATF,NRLIN)
DIMENSION MVF(40,15),NVF(1),MPINT(5,50),COEFIN(1),MATF(2,100)
2,MATLIN(1),P(1),Q(1),R(1)
DO 1 I=NFI,NFF
  N1=NVF(I)
DO 2 J=1,NRLIN

```

```

II=MATLIN(2*J-1)
IIF=MATLIN(2*J)
XI=P(II)
YI=Q(II)
ZI=R(II)
XF=P(IIF)
YF=Q(IIF)
ZF=R(IIF)
CALL FATASG(I,N1,MVF,XF,YF,ZF,XI,YI,ZI,P,Q,R,X,Y,Z,T4,CK)
IF(T4,FQ,0) GOTO 2
NRPTI=NRPTI+1
P(NRPTI)=X
Q(NRPTI)=Y
R(NRPTI)=Z
NPINT=NPINT+1
MPINT(1,NPINT)=I
MPINT(2,NPINT)=MATF(1,J)
MPINT(3,NPINT)=MATF(2,J)
MPINT(4,NPINT)=II
MPINT(5,NPINT)=IIF
COEFIN(NPINT)=CK
2 CONTINUE
1 CONTINUE
RETURN
END

```

### 7.7.6. Subrutina ESPI

```

C SUBROUTINA ESPI PENTRU EXTRAGEREA SEGMENTELOR CE ALCATUIESC
C POLIGONUL STRIMB DE INTERSECȚIE DINTRE CORPURI
SUBROUTINE ESPI(MPINT,NPI,INSEG,NRSG,NPDEF,MATF,P,Q,R)
DIMENSION MPINT(5,50),MATF(2,100),INSEG(1),P(1),Q(1),R(1)
EPS=0.0001
NRSG=0
NF=NPI-1
DO 1 I=1,NF
N1=MPINT(1,I)
N2=MPINT(2,I)
N3=MPINT(3,I)
NF1=I+1
DO 2 J=NF1,NPI
IF(ABS(P(NPDEF+I)-P(NPDEF+J)).LT.EPS.AND.ABS(Q(NPDEF+I)-Q(NPDEF+J))
2).LT.EPS.AND.ABS(R(NPDEF+I)-R(NPDEF+J)).LT.EPS) GOTO 2
M1=MPINT(1,J)
M2=MPINT(2,J)
M3=MPINT(3,J)
K1=1
IF(MPINT(4,I).EQ.MPINT(4,J).AND.MPINT(5,I).EQ.MPINT(5,J)) GOTO 2
IF(N1.NE.M1.AND.N1.NE.M2.AND.N1.NE.M3) GOTO 3
MATF(K1,NRSG+1)=N1
K1=K1+1
3 IF(N2.NE.M1.AND.N2.NE.M2.AND.N2.NE.M3) GOTO 4
MATF(K1,NRSG+1)=N2
K1=K1+1
4 IF(N3.NE.M1.AND.N3.NE.M2.AND.N3.NE.M3) GOTO 5
MATF(K1,NRSG+1)=N3
K1=K1+1
5 IF(K1.LT.3) GOTO 2
NRSG=NRSG+1
INSEG(2*NRSG-1)=I+NPDEF
INSEG(2*NRSG)=J+NPDEF
2 CONTINUE
1 CONTINUE
RETURN
END

```

## 7.7.7. Subrutina FATASG

```

C   SUBROUTINA FATASG CARE CALCULEAZA COORDONATELE PUNCTULUI
C   DE INTERSECȚIE ÎNTRĂ O FATA A UNUI CORP ȘI O MUCHEIE A ALTUI
C   CORP ÎN CAZ CA ACEASTA INTERSECȚIE EXISTĂ
SUBROUTINE FATASG(J,K,MVF,XF,YF,ZF,XI,YI,ZI,P,Q,R,X,Y,Z,T4,CK)
REAL*8 ALFA,ALFA1,PI
DIMENSION MVF(40,15),P(1),Q(1),R(1)
EPS=0.0001
PI=3.14159265358979328
F1=XF-XI
F2=YF-YI
F3=ZF-ZI
M1=MVF(J,1)
M2=MVF(J,2)
M3=MVF(J,3)
X23=P(M2)-P(M3)
Y23=Q(M2)-Q(M3)
Z23=R(M2)-R(M3)
T1=Q(M2)*R(M3)-R(M2)*Q(M3)
T2=P(M2)*R(M3)-R(M2)*P(M3)
T3=P(M2)*Q(M3)-Q(M2)*P(M3)
A=Q(M1)*Z23-R(M1)*Y23+T1
B=-P(M1)*Z23+R(M1)*X23-T2
C=P(M1)*Y23-Q(M1)*X23+T3
PL=-P(M1)*T1+Q(M1)*T2-R(M1)*T3
T4=A*F1+B*F2+C*F3
IF(T4.EQ.0)RETURN
CK=- (A*XI+B*YI+C*ZI+PL)/T4
T4=0
IF(CK.LT.-EPS.OR.CK.GT.1+EPS)RETURN
X=CK*F1+XI
Y=CK*F2+YI
Z=CK*F3+ZI
ALFA=0
NF=K-1
DO 1 I=1,NF
L=MVF(J,I)
L1=MVF(J,I+1)
SG1=(Q(L)-Y)*(R(L1)-Z)-(R(L)-Z)*(Q(L1)-Y)
IF(ABS(SG1).LE.EPS)SG1=(P(L)-X)*(Q(L1)-Y)-(Q(L)-Y)*(P(L1)-X)
IF(ABS(SG1).LE.EPS)SG1=(R(L)-Z)*(P(L1)-X)-(P(L)-X)*(R(L1)-Z)
IF(ABS(SG1).LE.EPS)GOTO 1
A2=(P(L)-X)**2+(Q(L)-Y)**2+(R(L)-Z)**2
B2=(P(L+1)-X)**2+(Q(L+1)-Y)**2+(R(L+1)-Z)**2
C2=(P(L+1)-P(L))**2+(Q(L+1)-Q(L))**2+(R(L+1)-R(L))**2
ALFA1=(A2+B2-C2)/(2*SQRT(A2*B2))
ALFA1=DATAN2(DSQRT(1.-ALFA1**2),ALFA1)
IF(ALFA1.LT.0.)ALFA1=ALFA1+PI
ALFA=ALFA1*SIGN(1.,SG1)
CONTINUE
IF(ABS(ALFA).GE.0.1)T4=1
RETURN
END

```

## 7.7.8. Subrutina LIR

```

SUBROUTINE LIR(LV,LI,KV1,KI1,N)
DIMENSION LV(50),LI(50)
N=1
KI=2*KI1-1
KV=2*KV1-1
DO 2 I=1,KI,2
  I1=LI(I)
  I2=LI(I+1)
  DO 1 J=1,KV,2
    J1=LV(J)
    J2=LV(J+1)
    IF (I1.EQ.J1.AND.I2.EQ.J2) GOTO 2
    IF (I1.EQ.J2.AND.I2.EQ.J1) GOTO 2
1 CONTINUE
  LI(N)=I1
  LI(N+1)=I2
  N=N+2
2 CONTINUE
RETURN
END

```

## 7.7.9. Subrutina DIORTO

```

SUBROUTINE DIORTO(X,Y,Z,XC,YC,ZC,XMAX,YMAX,ZMAX)
DIMENSION X(3),Y(3),Z(3)
X(1)=XMAX+1.
Y(1)=YC
Z(1)=ZC
X(2)=XC
Y(2)=YMAX+1.
Z(2)=ZC
X(3)=XC
Y(3)=YC
Z(3)=ZMAX+1.
RETURN
END

```

## 7.7.10. Subrutina VIZLIN

```

SUBROUTINE VIZLIN(X,Y,L,XM,N,M,P,Q,R,NL1,KV,A,R,C,E,F,X0,Y0,Z0,Y2,
2YM,SC,MPINT,INSEG,COEFIN,NPINT,NRSG,INTERS,NV)
COMMON NP,NO,NRI,XX,YY
DIMENSION X(1),Y(1),L(1),U(100),N(1),M(40,15),P(1),Q(1),R(1),MPINT
2(5,50),INSEG(50),COEFIN(50)
E0=.0001
E1=.9999
IF(NRSG.EQ.0)GOTO 35
DO 34 I=1,NRSG
L(2*NL1+2*I-1)=INSEG(2*I-1)
34 L(2*NL1+2*I)=INSEG(2*I)
35 NL=NL1*2+NKSG*2-1
NLT=NRI+NRSG
WRITE(2,31)(I6,L(2*I6-1),L(2*I6),I6=1,NLT)
31 FORMAT(' ',3I5)
DO 30 J=1,NL,2
INT=1
U(INT)=0.
I1=L(I)
I2=L(I+1)
DO 1 J=1,NL,2
J1=L(J)
J2=L(J+1)
IF(I1.EQ.J1.AND.I2.EQ.J2.OR.I1.EQ.J2.AND.I2.EQ.J1) GO TO 1
IF(X(J1).EQ.X(I2).AND.Y(J1).EQ.Y(I2)) GO TO 1
IF(X(J2).EQ.X(I1).AND.Y(J2).EQ.Y(I1)) GO TO 1
IF(X(J2).EQ.X(I2).AND.Y(J2).EQ.Y(I2)) GO TO 1
IF(X(J1).EQ.X(I1).AND.Y(J1).EQ.Y(I1)) GO TO 1
IF(X(J1).LE.AMIN1(X(I1),X(I2)).AND.X(J2).LE.AMIN1(X(I1),X(I2)))
160 TO 1
IF(X(J1).GE.AMAX1(X(I1),X(I2)).AND.X(J2).GE.AMAX1(X(I1),X(I2)))
160 TO 1
IF(Y(J1).LE.AMIN1(Y(I1),Y(I2)).AND.Y(J2).LE.AMIN1(Y(I1),Y(I2)))
160 TO 1
IF(Y(J1).GE.AMAX1(Y(I1),Y(I2)).AND.Y(J2).GE.AMAX1(Y(I1),Y(I2)))
160 TO 1
CALL DILP(X(I1),X(I2),X(J1),X(J2),Y(I1),Y(I2),Y(J1),Y(J2),D,UP,V)
IF(D.EQ.0.) GOTO 1
IF(UP.GE.1..OR.UP.LE.0..OR.V.GT.1..OR.V.LT.0.) GOTO 1
INT=INT+1
U(INT)=UP
1 CONTINUE
IF(INTRS.EQ.0.OR.I.GT.NL1*2)GOTO 7
DO 6 I3=1,MPINT
IF(I1.EQ.MPINT(4,I3).AND.I2.EQ.MPINT(5,I3)) GOTO 92
IF(I1.EQ.MPINT(5,I3).AND.I2.EQ.MPINT(4,I3)) GOTO 92
GOTO 6
92 CK1=SQRT((X(NV+I3)-X(I1))**2+(Y(NV+I3)-Y(I1))**2)
IF(CK1.EQ.0.) GOTO 6
INT=INT+1
U(INT)=CK1/SQRT((X(I2)-X(I1))**2+(Y(I2)-Y(I1))**2)
6 CONTINUE
7 INT=INT+1
U(INT)=1.
CALL ORD(INT,1,U)
CALL PLOT(SC*X(I1),SC*Y(I1),2)

```

```

XT=X(I1)
YT=Y(I1)
DO 30 K=2,INT
IF(U(K).EQ.U(K-1)) GO TO 30
XRE=XT
YRE=YT
XT=X(I1)+U(K)*(X(I2)-X(I1))
YT=Y(I1)+U(K)*(Y(I2)-Y(I1))
UI=U(K-1)+(U(K)-U(K-1))/2.
XF1=X(I1)+UI*(X(I2)-X(I1))
YF1=Y(I1)+UI*(Y(I2)-Y(I1))
X1=(XF1+XM)*A+(YF1+YM)*R
Y1=(XF1+XM)*C+(YF1+YM)*E+Y2
Z1=(YF1+YM)*F
IF(NO.EQ.1) GOTO 55
IF(NP.NF.1) GO TO 11
55 U2=UI
GOTO 12
11 CONTINUE
CALL DILP(P(I1),P(I2),X0,X1,Q(I1),Q(I2),Y0,Y1,D,U2,V2)
IF(D.EQ.0)CALL DILP(P(I1),P(I2),X0,X1,R(I1),R(I2),Z0,Z1,D,U2,V2)
12 CONTINUE
XN1=P(I1)+U2*(P(I2)-P(I1))
YN1=Q(I1)+U2*(Q(I2)-Q(I1))
ZN1=R(I1)+U2*(R(I2)-R(I1))
XU=X0
YU=Y0
ZU=Z0
IF(NO.FQ.1)GOTO 12356
IF(NP.FQ.0) GOTO 36
XU=X1
YU=Y1
ZU=Z1
GOTO 36
12356 IF(X1.NF.0.)XU=X1
IF(Y1.NF.0.)YU=Y1
IF(ZU.NF.0.)ZU=Z1
36 CONTINUE
DO 20 IV=1,KV
NKV=N(IV)
DO 29 IK=2,NKV
IK1=M(IV,IK-1)
IK2=M(IV,IK)
IF((I1.EQ.IK1.AND.I2.EQ.IK2).OR.(I1.EQ.IK2.AND.I2.EQ.IK1))GOTO 20
29 CONTINUE
CALL FATAS6(IV,NKV,M,XN1,YN1,ZN1,X0,Y0,Z0,P,Q,R,VALX,VALY,VALZ,T4,
1CF)
IF(CF.LT.1.001.AND.CF.GE.0.9999)GOTO 20
IF(T4.NF.1) GOTO 20
CALL PLOT(SC*XRE,SC*YRE,2)
IF(NRI) 21,21,22
22 CALL PLOT(SC*XT,SC*YT,1)
GO TO 30
21 CALL PLOT(SC*XT,SC*YT,2)
GOTO 30
20 CONTINUE
CALL PLOT(SC*XRE,SC*YRE,2)
CALL PLOT(SC*XT,SC*YT,1)
30 CONTINUE
RETURN
END

```

## 7.7.11. Subrutina DILP

```

SUBROUTINE DILP(XI1,XF1,XI2,XF2,YI1,YF1,YI2,YF2,D,P1,P2)
A1=XF1-XI1
A2=XF2-XI2
A3=XI2-XI1
B1=YF1-YI1
B2=YF2-YI2
B3=YI2-YI1
D=A1*B2-A2*B1
IF(D.EQ.0.) GO TO 1
DU=A3*B2-A2*B3
DV=A3*B1-B3*A1
P1=DU/D
P2=DV/D
1 RETURN
END

```

## 7.7.12. Subrutina ORD

```

SUBROUTINE ORD(N,M,T)
DIMENSION T(1)
DO 3 I=1,N
DO 2 J=1,N
IF(T(I).GE.T(J)) GO TO 2
L=I-N
LL=J-N
DO 1 K=1,M
L=L+N
LL=LL+N
F=T(L)
T(L)=T(LL)
1 T(LL)=F
2 CONTINUE
3 CONTINUE
RETURN
END

```

## 7.7.13. Subrutina TRASA

```

SUBROUTINE TRASA(L,N,X,Y,SC)
COMMON NP,NO,NI,XX,YY
DIMENSION L(1),X(1),Y(1)
N=N-1
DO 1 I=1,N,2
I1=L(I)
CALL PLOT(SC*X(I1),SC*Y(I1),2)
I2=L(I+1)
1 CALL PLOT(SC*X(I2),SC*Y(I2),1)
RETURN
END

```



## 7.7.14. Subrutina PLOT

```

SUBROUTINE PLOT(X,Y,IPEN)
COMMON NP,NO,NI,XX,YY
X1=X+XX
Y1=Y+YY
IF (IPEN.EQ.999) RETURN
WRITE(2,1)X1,Y1,IPEN
FORMAT(1,'2F15.8,I6)
RETURN
END

```

## 7.7.a. Program ALPLA (variantă)

```

C PROGRAM ALPLA
C ALGORITM PENTRU LINII ASCUNSE SI INTERSECTII DE POLIEDRE
C
      LOGICAL*1 LLL(20)
      INTEGER PROP,ORTOG,TRASI,CONVEX
      COMMON PROP,ORTOG,TRASI,XX,YY
      DIMENSION XN(200),YN(200),ZN(200),MVF(40,15)
1     1,NVF(100),XP(200),YP(200),MVFV(40,15),MVFI(40,15),NVFV(100),NVFI(
      200),LINV(200),LINI(200),U(50),MATF(2,100),MPINT(5,100),INSEG(100),
      3COEFIN(100),MATLIN(200),NSG(100)
      XX=0.
      CALL ASSIGN(2,'LP:')
      TYPE 5000
5000  FORMAT('/',',','FISIÉRUL DE DATE : ',')
      ACCEPT 5001,NL,LLL
5001  FORMAT(0,20A1)
      CALL ASSIGN(4,LLL,NL)
      CALL FDBSET(4,'R')
      TYPE 5010
5010  FORMAT('/',',','FISIER DE DESEN : ',')
      ACCEPT 5001,NL,LLL
      CALL ASSIGN(3,LLL,NL)
      CALL FDBSET(3,'NEW')
      CALL INI(3)
C
C ...CITESTE SI SCRIE TABLOUL CU DATELE PROBLEMEI...
      READ(4,100) PROP,ORTOG,TRASI,NV,NF,CONVEX,LISTD,IORT
      READ(4,101) XV0,YV0,ZV0,XC,YC,ZC
      READ(4,103) (XN(I),YN(I),ZN(I),I=1,NV)
      READ(4,104) (NVF(I),I=1,NF)
      IF (LISTD.EQ.0) GOTO 66
      WRITE(2,200)
      WRITE(2,100) PROP,ORTOG,TRASI,NV,NF,CONVEX,LISTD,NS
      WRITE(2,101) XV0,YV0,ZV0,XC,YC,ZC
      WRITE(2,103) (XN(I),YN(I),ZN(I),I=1,NV)
      WRITE(2,104) (NVF(I),I=1,NF)
200  FORMAT(1H0,18X,27HTABLOUL CU DATELE PROBLEMEI/17X,27(1H*)//)
      100 FORMAT(8I10)
      101 FORMAT(6F10.4)
      102 FORMAT(10X,6F10.1)
      103 FORMAT(12F6.2)
      104 FORMAT(10X,30I2)
66   DO 10 I=1,NF
      IN=NVF(I)
      READ(4,105) (MVF(I,J),J=1,IN)
      IF (LISTD.EQ.0) GOTO 10
      WRITE(2,101) (MVF(I,J),J=1,IN)
105  FORMAT(10X,20I3)
10   CONTINUE
      READ(4,100) I,ITERS,NV1,NF1,NS
      IF (NS.EQ.0) GOTO 2345
      READ(4,104) (NSG(2*I-1),NSG(2*I),I=1,NS)
2345  IF (ITERS.NE.1) GOTO 93
      NV2=NV-NV1

```

```

NF2=NF-NF1
CALL LINFV(MVF,NVF,NF1+1,NF,MATLIN,MATF,NRSEG,1)
CALL LSELET(MATLIN,MATF,NRSEG,NRLIN,1,NS,NSG)
NRPTI=NV
NPINT=0
CALL CINT(1,NF1,MVF,NVF,XN,YN,ZN,NRPTI,MPINT,COEFIN,NPINT,MATLIN,M
*ATF,NRLIN)
CALL LINFV(MVF,NVF,1,NF1,MATLIN,MATF,NRSEG,1)
CALL LSELET(MATLIN,MATF,NRSEG,NRLIN,1,NS,NSG)
CALL CINT(NF1+1,NF,MVF,NVF,XN,YN,ZN,NRPTI,MPINT,COEFIN,NPINT,MATLI
2N,MATF,NRLIN)
CALL ESPI(MPINT,NPINT,INSEG,NRSG,NV,MATF,XN,YN,ZN)
93 IF(ORTOG.NE.1) GOTO 11
CALL DMIMA(NV,XN,XNMIN,XNMAX)
CALL DMIMA(NV,YN,YNMIN,YNMAX)
CALL DMIMA(NV,ZN,ZNMIN,ZNMAX)
11 CALL DIORTO(XVO,YVO,ZVO,XC,YC,ZC,XNMAX,YNMAX,ZNMAX,IORT)
CONTINUE
NVT=NV+NPINT
X0=XVO
Y0=YVO
Z0=ZVO
CALL DATEIN(X0,Y0,Z0,XC,YC,ZC,A,B,C,P,Y2,A1,B1,A2,B2,G2)
IF(LISTD.EQ.0) GOTO 68
WRITE(2,206)
206 FORMAT(1H0,14X,32HTABLOUL CU REZULTATELE PROBLEMEI/15X,32(1H*)//2X
1,1HI,17X,5HXP(I),17X,5HYPI)//)
WRITE(2,101) X0,Y0,Z0,A,B,C,P,Y2,A1,B1,A2,B2,G2
...INCEPE CICLAREA PENTRU CALCULUL PROIECTIEI...
68 DO 25 I=1,NVT
DNR=A*XN(I)+B*YN(I)+C*ZN(I)+P
IF(ORTOG.EQ.1) GOTO 77
IF(PROP.NE.1) GO TO 15
77 AL=A
BE=B
GA=C
LAM=DNR/(A*AL+B*BE+C*GA)
X=XN(I)-AL*LAM
Y=YN(I)-BE*LAM
Z=ZN(I)-GA*LAM
GO TO 20
15 T1=X0-XN(I)
T2=Y0-YN(I)
T3=Z0-ZN(I)
DVN=SQRT(T1*T1+T2*T2+T3*T3)
CL=T1/DVN
CM=T2/DVN
CN=T3/DVN
LAM=DNR/(A*CL+B*CM+C*CN)
X=XN(I)-CL*LAM
Y=YN(I)-CM*LAM
Z=ZN(I)-CN*LAM
20 P1=X
P2=Y-Y2
P3=Z
XP(I)=A1*P1+B1*P2
YP(I)=A2*P1+B2*P2+G2*P3
25 CONTINUE
CALL DMIMA(NV,XP,XM,XMA)
CALL DMIMA(NV,YP,YM,YMA)
DO 27 I=1,NVT
YP(I)=YP(I)-YM
27 XP(I)=XP(I)-XM
SX=250./(XMA-XM)
SY=150./(YMA-YM)
SC=AMINI(SX,SY)
TYPE 5002,SC
5002 FORMAT(' ',INTRODUCETI SCARA('F6.2','NEW F6.2):',%)
ACCEPT 5003,SCAR
5003 FORMAT(F6.2)
IF(SCAR.NE.0) SC=SCAR
C

```

```

C      ...SCRIE COORDONATELE PROIECTIEI...
C      IF(LISTD.EQ.0) GOTO 69
      DO 30 I=1,NVT
30     WRITE(2,201) I,XP(I),YP(I)
201    FORMAT(1H.,14,2(11X,F10.2))
      WRITE(2,202)
202    FORMAT(1H0,17X,25HMATRICEA FETELOR VIZIBILE/17X,25(1H*)//1X,3(1H)
      1,1HI,3(1H.),27X,15H...MVFV(I,J)...//)
C
C      ...SE TESTEAZA VIZIBILITATEA FETELOR...
C
69     TYPE 89
89     FORMAT(' DEPLASAMENT PE X SI Y [2F5.1]: ',*)
      ACCEPT 63,XX,YY
63     FORMAT(2F5.1)
      CALL BEG(XX,YY)
      CALL SSCA(SC,SC)
      IF(CONVEX.EQ.2)GOTO 177
      KV=0.
      KI=0.
      DO 55 L=1,NF
      I1=MVF(L,1)
      I2=MVF(L,2)
      I3=MVF(L,3)
      D1=XP(I1)-XP(I2)
      D2=YP(I3)-YP(I2)
      D3=XP(I3)-XP(I2)
      D4=YP(I1)-YP(I2)
      FV=D1*D2-D3*D4
      NV=NVF(L)
      IF(PV) 35,45,45
35     KV=KV+1
      NVFV(KV)=NVF(L)
      DO 40 M=1,N1
40     MVFV(KV,M)=MVF(L,M)
      GO TO 55
45     KI=KI+1
      NVFI(KI)=NVF(L)
      DO 50 M=1,N1
50     MVFI(KI,M)=MVF(L,M)
55     CONTINUE
177    GOTO 148
      KV=NF
      DO 179 KP=1,NF
      N3=NVF(KP)
      NVFV(KP)=N3
      DO 179 NT1=1,N3
179    MVFV(KP,NT1)=MVF(KP,NT1)
148    IF(LISTD.EQ.0) GOTO 71
      DO 60 I=1,KV
      NVI=NVFV(I)
60     WRITE(2,203) I,(MVFV(I,M),M=1,NVI)
203    FORMAT(16,4X,10IS)
      WRITE(2,204)
204    FORMAT(1H0,16X,27HMATRICEA FETELOR INVIZIBILE/16X,27(1H*)//1X,7H..
      1,I...20X,15H...MVFI(I,J)...//)
      DO 65 I=1,KI
      NI=NVFI(I)
65     WRITE(2,205) I,(MVFI(I,M),M=1,NI)
205    FORMAT(16,4X,10IS)
71     CALL LINFV(MVFV,NVFV,1,KV,LINV,MATF,LV,0)
      CALL LSELET(LINV,MATF,LV,LVS,0)
      IF(INTERS.NE.0)GOTO 993
      IF(CONVEX.EQ.1) GOTO 991
993    CALL VIZLIN(XP,YP,LINV,XM,NVFV,MVFV,XN,YN,ZN,LVS,KV,A1,A2,B1,B2,C2
G2     1,X0,Y0,Z0,YM,MPINT,INSEG,COEFIN,NPINT,NRSG,INTERS,NV)
      GOTO 992
991    CALL TRASA(LINV,LVS,XP,YP)
992    IF(TRASI.EQ.0.OR.CONVEX.EQ.2) GOTO 80
      CALL LINFV(MVFI,NVFI,1,KI,LINI,MATF,LI,0,NS,NSG)
      CALL LSELET(LINI,MATF,LI,LIS,0)
      CALL LIR(LINV,LINI,LVS,LIS,LR)
      LIS=LR-1
      CALL TRASA(LINI,LIS,XP,YP)
80     CONTINUE
      CALL EOF
      CALL CLOSE(2)
      CALL CLOSE(4)
      STOP
      END

```

```

SUBROUTINE DATEIN(X,Y,Z,U,V,W,A,B,C,P,E,F,G,H,O,R)
COMMON NP,NO,NN,XX,YMX
D1=X-U
D2=Y-V
D3=Z-W
D=SQRT(D1*D1+D2*D2+D3*D3)
DP=SQRT(D1*D1+D2*D2)
A=D1/D
B=D2/D
C=D3/D
DC=U*U+V*V+W*W
DV=X*X+Y*Y+Z*Z
P=(DC-DV)/(2.*D)
IF(NO.NE.1) GOTO 60
P=.0
E=.0
F=0.
G=0.
H=0.
O=0.
P=0.
IF(A.EQ.1..AND.B.EQ.0..AND.C.EQ.0.) GO TO 30
IF(A.EQ.0..AND.B.EQ.1..AND.C.EQ.0.) GO TO 40
IF(A.EQ.0..AND.B.EQ.0..AND.C.EQ.1.) GO TO 50
30 G=1.
R=1.
RETURN
40 F=-1.
R=1.
RETURN
50 F=-1.
O=-1.
RETURN
60 CONTINUE
E=-P/B
F=ABS(D2)/DP
G=ABS(D1)/DP
H=ABS(G*C)
O=ABS(F*C)
R=SQRT(1.-C*C)
IF(C.LT.0.) GO TO 1
IF(A.GT.0..AND.B.GT.0.) GO TO 2
IF(A.LT.0..AND.B.GT.0.) GO TO 3
IF(A.LT.0..AND.B.LT.0.) GO TO 4
IF(A.GT.0..AND.B.LT.0.) GO TO 5
1 IF(A.GT.0..AND.B.GT.0.) GO TO 6
IF(A.LT.0..AND.B.GT.0.) GO TO 7
IF(A.LT.0..AND.B.LT.0.) GO TO 8
IF(A.GT.0..AND.B.LT.0.) GO TO 9
2 F=-F
H=-H
O=-O
RETURN
3 F=-F
G=-G
O=-O
RETURN
4 G=-G
RETURN
5 H=-H
RETURN
6 F=-F
RETURN
7 F=-F
G=-G
H=-H
RETURN
8 G=-G
H=-H
O=-O
RETURN
9 O=-O
RETURN
END

```

```

SUBROUTINE DMIMA(N, T, DMI, DMA)
  DIMENSION T(N)
  DMI=T(1)
  DMA=DMI
  DO 3 I=2, N
  IF(T(I).LT.DMI) GO TO 1
  IF(T(I)-DMA) 3,3,2
1 DMI=T(I)
  GO TO 3
2 DMA=T(I)
3 CONTINUE
  RETURN
  END

C   SUBROUTINA LINFV PENTRU OBTINEREA VECTORULUI DE SEGMENTE DI
C   CARE SINT ALCATUITE FETELE CORPURILOR
SUBROUTINE LINFV(MVF, NVF, IF1, LF, MATLIN, MATF, NRSEG, INTERS)
  DIMENSION MVF(40, 15), NVF(1), MATLIN(1), MATF(2, 100)
  NRSEG=0
  DO 2 I=IF1, LF
  NR=NVF(I)-1
  DO 1 J=1, 2
  LL=J+NR-1
  DO 1 K=J, LL
  IF(J.EQ.1.AND.INTERSE.EQ.1)MATF(1, NRSEG+K)=I
1 MATLIN(2*NRSEG+2*K-J)=MVF(I, K)
2 NRSEG=NRSEG+NR
  RETURN
  END

C   SUBROUTINA CARE ELIMINA SEGMENTELE CU DUBLA APARITIE
C   DIN VECTORUL OBTINUT DIN LINFV
SUBROUTINE LSELET(MATLIN, MATF, NRSEG, NFIN, INTERS, NS, NSG)
  DIMENSION MATLIN(1), MATF(2, 100), NSG(1)
  NFIN=2*NRSEG-1
  DO 3 I=1, NFIN, 2
  IF (I.GT.NFIN) GOTO 2
  I1=MATLIN(I)
  I2=MATLIN(I+1)
  IN=I+2
  DO 1 J=IN, NFIN, 2
  J1=MATLIN(J)
  J2=MATLIN(J+1)
  IF(.NOT.((I1.EQ.J1.AND.I2.EQ.J2).OR.(I1.EQ.J2.AND.I2.EQ.J1)))
    *GOTO 1
  MATLIN(J)=MATLIN(NFIN)
  MATLIN(J+1)=MATLIN(NFIN+1)
  IF (INTERSE.NE.1)GOTO 4
  MATF(2, (I+1)/2)=MATF(1, (J+1)/2)
  MATF(1, (J+1)/2)=MATF(1, (NFIN+1)/2)
  NFIN=NFIN-2
  IF(NS.EQ.0) GOTO 3
  DO 5 K=1, NS
  K1=NSG(2*K-1)
  K2=NSG(2*K)
  IF(.NOT.((I1.EQ.K1.AND.I2.EQ.K2).OR.(I1.EQ.K2.AND.I2.EQ.K1)))
    *GOTO 5
  MATLIN(I)=MATLIN(NFIN)
  MATLIN(I+1)=MATLIN(NFIN+1)
  IF (INTERSE.NE.1) GOTO 6
  MATF(1, (I+1)/2)=MATF(1, (NFIN+1)/2)
  NFIN=NFIN-2
  GOTO 8
5 CONTINUE
1 CONTINUE
3 CONTINUE
2 NFIN=I/2
  RETURN
  END

```

```

C   SUBROUTINA CINT PENTRU OBTINEREA MATRICII PUNCTELOR DE INTERSECȚIE
C   INTRE FETELE UNUI CORP SI MUCHIILE CELUIALT CORP
SUBROUTINE CINT(NFI,NFF,MVF,NVF,P,Q,R,NRPTI,MPINT,COEFIN,NPINT,
2MATLIN,MATF,NRLIN)
  DIMENSION MVF(40,15),NVF(1),MPINT(5,100),COEFIN(1),MATF(2,100)
2  MATLIN(1),P(1),Q(1),R(1)
  DO 1 I=NFI,NFF
    N1=NVF(I)
    DO 2 J=1,NRLIN
      I1=MATLIN(2*J-1)
      IIF=MATLIN(2*J)
      XI=P(I1)
      YI=Q(I1)
      ZI=R(I1)
      XF=P(IIF)
      YF=Q(IIF)
      ZF=R(IIF)
      CALL FATASG(I,N1,MVF,XF,YF,ZF,XI,YI,ZI,P,Q,R,X,Y,Z,T4,CK)
      IF(T4.EQ.0) GOTO 2
      NRPTI=NRPTI+1
      P(NRPTI)=X
      Q(NRPTI)=Y
      R(NRPTI)=Z
      NPINT=NPINT+1
      MPINT(1,NPINT)=I
      MPINT(2,NPINT)=MATF(1,J)
      MPINT(3,NPINT)=MATF(2,J)
      MPINT(4,NPINT)=I1
      MPINT(5,NPINT)=IIF
      COEFIN(NPINT)=CK
1  CONTINUE
1  CONTINUE
  RETURN
END

```

```

C   SUBROUTINA ESPI PENTRU EXTRAGEREA SEGMENTELOR CE ALCATUIESC
C   POLIGONUL STRIMB DE INTERSECȚIE DINTRE CORPURI
SUBROUTINE ESPI(MPINT,NPI,INSEG,NRSG,NPDEF,MATF,P,Q,R)
  DIMENSION MPINT(5,100),MATF(2,100),INSEG(1),P(1),Q(1),R(1)
  EPS=0.001
  NRSG=0
  NF=NPI-1
  DO 1 I=1,NF
    N1=MPINT(1,I)
    N2=MPINT(2,I)
    N3=MPINT(3,I)
    NF1=I+1
    DO 2 J=NF1,NPI
      IF(ABS(P(NPDEF+I)-P(NPDEF+J)).LT.EPS.AND.ABS(Q(NPDEF+I)-Q(NPDEF+J))
2) .LT.EPS.AND.ABS(R(NPDEF+I)-R(NPDEF+J)).LT.EPS), GOTO 2
      M1=MPINT(1,J)
      M2=MPINT(2,J)
      M3=MPINT(3,J)
      K1=1
      IF(MPINT(4,I).EQ.MPINT(4,J).AND.MPINT(5,I).EQ.MPINT(5,J))GOTO2
      IF(N1.NE.M1.AND.N1.NE.M2.AND.N1.NE.M3) GOTO 3
      MATF(K1,NRSG+1)=N1
      K1=K1+1
3     IF(N2.NE.M1.AND.N2.NE.M2.AND.N2.NE.M3) GOTO 4
      MATF(K1,NRSG+1)=N2
      K1=K1+1
4     IF(N3.NE.M1.AND.N3.NE.M2.AND.N3.NE.M3) GOTO 5
      MATF(K1,NRSG+1)=N3
      K1=K1+1
5     IF(K1.LT.3) GOTO 2
      IF(NRSG.EQ.0)GOTO 7
      I1=NPDEF+I
      J1=NPDEF+J
      DO 6 K=1,NRSG
        I2=INSEG(2*K-1)
        J2=INSEG(2*K)
        IF(ABS(P(I1)-P(I2)).LT.EPS.AND.ABS(Q(I1)-Q(I2)).LT.EPS.AND

```

```

*ABS(R(I1)-R(I2)).LT.EPS.AND.ABS(P(J1)-P(J2)).LT.EPS.AND.
*ABS(Q(J1)-Q(J2)).LT.EPS.AND.ABS(R(J1)-R(J2)).LT.EPS)GOTO 2
IF (ABS(P(I1)-P(J2)).LT.EPS.AND.ABS(Q(I1)-Q(J2)).LT.EPS.AND.

*ABS(R(I1)-R(J2)).LT.EPS.AND.ABS(P(J1)-P(I2)).LT.EPS.AND.
*ABS(Q(J1)-Q(I2)).LT.EPS.AND.ABS(R(J1)-R(I2)).LT.EPS)GOTO 2
CONTINUE
NRSG=NRSG+1
INSEG(2*NRSG-1)=I+NPDEF
INSEG(2*NRSG)=J+NPDEF
2 CONTINUE
1 CONTINUE
RETURN
END

C SUBROUTINA FATASG CARE CALCULEAZA COORDONATELE PUNCTULUI
C DE INTERSECȚIE ÎNTRE O FATĂ A UNUI CORP ȘI O MUCHIE A ALTUI
C CORP ÎN CAZ CA ACEASTA INTERSECȚIE EXISTA
SUBROUTINE FATASG(J,K,MVF,XF,YF,ZF,XI,YI,ZI,P,Q,R,X,Y,Z,T4,CK)
REAL*8 ALFA,ALFA1,PI
DIMENSION MVF(40,15),P(1),Q(1),R(1)
EPS=0.0001
PI=3.14159265358979328
F1=XF-XI
F2=YF-YI
F3=ZF-ZI
M1=MVF(J,1)
M2=MVF(J,2)
M3=MVF(J,3)
X23=P(M2)-P(M3)
Y23=Q(M2)-Q(M3)
Z23=R(M2)-R(M3)
T1=Q(M2)*R(M3)-R(M2)*Q(M3)
T2=P(M2)*R(M3)-R(M2)*P(M3)
T3=P(M2)*Q(M3)-Q(M2)*P(M3)
A=Q(M1)*Z23-R(M1)*Y23+T1
B=-P(M1)*Z23+R(M1)*X23-T2
C=P(M1)*Y23-Q(M1)*X23+T3
PL=-P(M1)*T1+Q(M1)*T2-R(M1)*T3
T4=A*F1+B*F2+C*F3
IF (T4.EQ.0)RETURN
CK=- (A*XI+B*YI+C*ZI+PL)/T4
T4=0
IF (CK.LT.-EPS.OR.CK.GT.1+EPS)RETURN
X=CK*F1+XI
Y=CK*F2+YI
Z=CK*F3+ZI
ALFA=0
NF=K-1
NRPR=1
DO 1 I=1,NF
L=MVF(J,I)
L1=MVF(J,I+1)
GOTO (200,300,400),NRPR
200 SG1=(Q(L)-Y)*(R(L1)-Z)-(R(L)-Z)*(Q(L1)-Y)
GOTO 500
300 SG1=(P(L)-X)*(Q(L1)-Y)-(Q(L)-Y)*(P(L1)-X)
GOTO 500
400 SG1=(R(L)-Z)*(P(L1)-X)-(P(L)-X)*(R(L1)-Z)
500 IF (ABS(SG1).LE.EPS)GOTO 600
A2=(P(L)-X)**2+(Q(L)-Y)**2+(R(L)-Z)**2
B2=(P(L1)-X)**2+(Q(L1)-Y)**2+(R(L1)-Z)**2
C2=(P(L1)-P(L))**2+(Q(L1)-Q(L))**2+(R(L1)-R(L))**2
ALFA1=(A2+B2-C2)/(2.*SQRT(A2*B2))
ALFA1=DATAN2(DSQRT(1.-ALFA1**2),ALFA1)
IF (ALFA1.LT.0.)ALFA1=PI+ALFA1
ALFA=ALFA+ALFA1*SIGN(1.,SG1)
1 CONTINUE
IF (ABS(ALFA).GE.0.1)T4=1
RETURN
600 IF (NRPR.EQ.3)GOTO 1

NRPR=NRPR+1
ALFA=0.
GOTO 100
END

```

```

SUBROUTINE LIR(LV,LI,KV1,KI1,N)
DIMENSION LV(50),LI(50)
N=1
KI=2*KI1-1
KV=2*KV1-1
DO 2 I=1,KI,2
I1=LI(I)
I2=LI(I+1)
DO 1 J=1,KV,2
J1=LV(J)
J2=LV(J+1)
IF(I1.EQ.J1.AND.I2.EQ.J2) GOTO 2
IF(I1.EQ.J2.AND.I2.EQ.J1) GOTO 2
1 CONTINUE
LI(N)=I1
LI(N+1)=I2
N=N+2
2 CONTINUE
RETURN
END

SUBROUTINE DIORTO(X,Y,Z,XC,YC,ZC,XMAX,YMAX,ZMAX,IORT)
GOTO (1,2,3),IORT
1 X=XMAX+1.
Y=YC
Z=ZC
GOTO 4
2 X=XC
Y=YMAX+1.
Z=ZC
GOTO 4
3 X=XC
Y=YC
Z=ZMAX+1.
RETURN
END

SUBROUTINE VIZLIN(X,Y,L,XM,N,M,P,Q,R,NL1,KV,A,B,C,E,F,XO,YO,ZO),MPINT,
2YM,MPINT,INSEG,COEFIN,NPINT,NRSG,INTERS,NV)
COMMON NP,NO,NRI,XMX,VMX
DIMENSION X(1),Y(1),L(1),U(100),N(1),M(40,15),P(1),Q(1),R(1),MPINT
2(5,100),INSEG(100),COEFIN(50)
EO=.0001
E1=.9999
IF(NRSG.EQ.0)GOTO 35
DO 34 I=1,NRSG
L(2*NLI+2*I-1)=INSEG(2*I-1)
34 L(2*NLI+2*I)=INSEG(2*I)
35 NL=NLI+2+NRSG*2-1
NL=NL+NRSG
DO 30 I=1,NL,2
INT=1
U(INT)=0.
I1=L(I)
I2=L(I+1)
DO 1 J=1,NL,2
J1=L(J)
J2=L(J+1)
IF(I1.EQ.J1.AND.I2.EQ.J2.OR.I1.EQ.J2.AND.I2.EQ.J1) GO TO 1
IF(X(J1).EQ.X(I2).AND.Y(J1).EQ.Y(I2)) GOTO 1
IF(X(J2).EQ.X(I1).AND.Y(J2).EQ.Y(I1)) GOTO 1
IF(X(J2).EQ.X(I2).AND.Y(J2).EQ.Y(I2)) GOTO 1
IF(X(J1).EQ.X(I1).AND.Y(J1).EQ.Y(I1)) GOTO 1
IF(X(J1).LE.AMIN1(X(I1),X(I2)).AND.X(J2).LE.AMIN1(X(I1),X(I2)))
1GO TO 1
IF(X(J1).GE.AMAX1(X(I1),X(I2)).AND.X(J2).GE.AMAX1(X(I1),X(I2)))
1GO TO 1
IF(Y(J1).LE.AMIN1(Y(I1),Y(I2)).AND.Y(J2).LE.AMIN1(Y(I1),Y(I2)))
1GO TO 1
IF(Y(J1).GE.AMAX1(Y(I1),Y(I2)).AND.Y(J2).GE.AMAX1(Y(I1),Y(I2)))
1GO TO 1
CALL DILP(X(I1),X(I2),X(J1),X(J2),Y(I1),Y(I2),Y(J1),Y(J2),D,UP,V

```



```

IF(D.EQ.0.) GOTO 1
IF(UP.GE.1..OR.UP.LE.0..OR.V.GT.1..OR.V.LT.0.) GOTO 1
INT=INT+1
U(INT)=UP
1 CONTINUE
IF(INTERSE.EQ.0.OR.I.GT.NL1*2)GOTO 7
DO 6 I3=1,NPINT
IF(I1.EQ.MPINT(4,I3).AND.I2.EQ.MPINT(5,I3)) GOTO 92
IF(I1.EQ.MPINT(5,I3).AND.I2.EQ.MPINT(4,I3)) GOTO 92
GOTO 6
92 CK1=SQRT((X(NV+I3)-X(I1))**2+(Y(NV+I3)-Y(I1))**2)
IF(CK1.EQ.0.) GOTO 6
INT=INT+1
U(INT)=CK1/SQRT((X(I2)-X(I1))**2+(Y(I2)-Y(I1))**2)
6 CONTINUE
7 INT=INT+1
U(INT)=1.
CALL ORD(INT,1,U)
XT=X(I1)
YT=Y(I1)
INDPEN=0

DO 30 K=2,INT
IF((U(K)-U(K-1)).LT.E0)GOTO 30
XRE=XT
YRE=YT
XT=X(I1)+U(K)*(X(I2)-X(I1))
YT=Y(I1)+U(K)*(Y(I2)-Y(I1))
UI=U(K-1)+(U(K)-U(K-1))/2.
XF1=X(I1)+UI*(X(I2)-X(I1))
YF1=Y(I1)+UI*(Y(I2)-Y(I1))
X1=(XF1+XM)*A+(YF1+YM)*B
Y1=(XF1+XM)*C+(YF1+YM)*E+Y2
Z1=(YF1+YM)*F
IF(NO.EQ.1) GOTO 55
IF(NP.NE.1) GO TO 11
55 U2=UI
GOTO 12
11 CONTINUE
CALL DILP(P(I1),P(I2),X0,X1,Q(I1),Q(I2),Y0,Y1,D,U2,V2)
IF(D.EQ.0)CALL DILP(P(I1),P(I2),X0,X1,R(I1),R(I2),Z0,Z1,D,U2,V2)

12 CONTINUE
XN1=P(I1)+U2*(P(I2)-P(I1))
YN1=Q(I1)+U2*(Q(I2)-Q(I1))
ZN1=R(I1)+U2*(R(I2)-R(I1))
XU=X0
YU=Y0
ZU=Z0
IF(NO.EQ.1)GOTO 12356
IF(NP.EQ.0) GOTO 36
XU=X1
YU=Y1
ZU=Z1
GOTO 36
12356 IF(X1.NE.0.) XU=X1
IF(Y1.NE.0.) YU=Y1
IF(Z1.NE.0.) ZU=Z1
36 CONTINUE
DO 20 IV=1,KV
NKV=N(IV)
DO 29 IK=3,NKV
IK1=M(IV,IK-1)
IK2=M(IV,IK)
IF((I1.EQ.IK1.AND.I2.EQ.IK2).OR.(I1.EQ.IK2.AND.I2.EQ.IK1))
*GOTO 20
29 CONTINUE
CALL FATASG(IV,NKV,M,XN1,YN1,ZN1,XU,YU,ZU,P,Q,R,VALX,VALY,VALZ,T4,
1CF)
IF(CF.LT.1.001.AND.CF.GE.0.999) GOTO 20
IF(T4.NE.1) GOTO 20
IF(NRI) 21,21,22
22 CALL PLOT(XT,YT,1)
GO TO 30
21 INDPEN=0
GOTO 30

```

```

20 CONTINUE
   IF (INDPEN.EQ.0) CALL PLOT(XRE,YRE,0)
   CALL PLOT(XT,YT,1)
   INDPEN=1
30 CONTINUE
RETURN
END

```

```

SUBROUTINE DILP(XI1, XF1, XI2, XF2, YI1, YF1, YI2, YF2, D, P1, P2)
A1=XF1-XI1
A2=XF2-XI2
A3=XI2-XI1
B1=YF1-YI1
B2=YF2-YI2
B3=YI2-YI1
D=A1*B2-A2*B1
IF (D.EQ.0.) GO TO 1
DU=A3*B2-A2*B3
DV=A3*B1-B3*A1
P1=DU/D
P2=DV/D
1 RETURN
END

```

```

SUBROUTINE ORD(N,M,T)
DIMENSION T(1)
DO 3 I=1,N
DO 2 J=1,N
IF (T(I).GE.T(J)) GO TO 2
L=I-N
LL=J-N
DO 1 K=1,M
L=L+N
LL=LL+N
F=T(L)
T(L)=T(LL)
1 T(LL)=F
2 CONTINUE
3 CONTINUE
RETURN
END

```

```

SUBROUTINE TRASA(L,N,X,Y)
DIMENSION L(1),X(1),Y(1)
N=N-1
DO 1 I=1,N,2
I1=L(I)
CALL PLOT(X(I1),Y(I1),0)
I2=L(I+1)
1 CALL PLOT(X(I2),Y(I2),1)
RETURN
END

```



TAB.UL CU REZULTATELE PROBLEMEI  
 \*\*\*\*\*

I	XP(I)		YP(I)		
100.0000	10.0000	200.0000	0.5528	-0.7518	0.3593
-33.2499	50.8750	0.8057	0.5924	-0.2129	0.2495
0.9332					
1		15.41		9.79	
2		12.52		30.43	
3		0.00		36.97	
4		3.38		17.29	
5		24.94		15.81	
6		22.56		35.69	
7		52.89		0.00	
8		51.63		21.57	
9		81.99		24.34	
10		82.28		43.24	
11		71.00		29.02	
12		70.84		47.25	
13		53.41		36.37	
14		47.37		38.87	
15		40.56		34.91	
16		24.85		42.05	
17		39.01		49.12	
18		44.13		47.04	
19		48.39		49.81	
20		37.28		54.34	
21		38.74		37.22	
22		50.37		31.98	
23		45.48		28.93	
24		40.40		31.19	
25		27.07		23.18	
26		42.12		14.91	
27		48.60		19.47	
28		38.68		24.62	
29		32.33		98.34	
30		50.54		95.07	
31		85.40		104.15	
32		69.57		106.31	
33		69.13		60.19	
34		65.09		58.07	
35		60.09		59.65	
36		77.72		63.46	
37		86.66		66.12	
38		73.54		58.90	
39		73.58		42.50	
40		60.66		43.48	
41		65.47		41.64	
42		84.05		38.64	
43		86.36		50.99	
44		77.64		53.68	
45		70.73		55.88	
46		78.73		60.38	
47		79.03		96.37	
48		70.39		93.76	
49		109.02		87.06	
50		116.26		90.48	
51		115.02		73.67	
52		107.84		69.19	
53		97.31		71.63	
54		93.69		69.61	
55		122.60		62.59	
56		127.14		66.23	
57		118.08		68.29	
58		125.85		73.81	
59		107.84		77.47	
60		104.79		75.79	
61		103.43		53.20	
62		106.31		55.41	
63		123.26		50.65	
64		115.72		43.52	
65		124.30		40.83	
66		119.78		36.24	
67		92.76		45.13	
68		96.20		47.88	





TABLOUL CU REZULTATELE PROBLEMEI  
\*\*\*\*\*

Y<sub>102</sub>

I	XP(I)		YP(I)		Y <sub>102</sub>
	XP(I)	XP(I)	YP(I)	YP(I)	
14	9.0000	4.5000	6.0000	1.0000	0.0000
0	0.0000	0.0000	0.0000	1.0000	0.0000
1	0.0000				
2			4.50		4.00
3			9.00		4.00
4			0.00		4.00
5			4.50	11.40	0.00
6			9.00		7.80
7			0.00		7.80
8			4.50		3.80
9			6.11		7.80
10			7.13		4.55
11			2.69		7.80
12			1.87		4.55
13			6.81		4.00
14			2.19		4.00
15			7.13		7.07
16			1.87		7.07
17			7.61		6.29
18			5.95		4.00
19			3.05		4.00
20			1.39		6.29

MATRICEA FETELOR VIZIBILE  
\*\*\*\*\*

I	...I...					...MVIV(I,J)...
	1	4	1	2	4	
1						
2						
3						
4						
5						

MATRICEA FETELOR INVIZIBILE  
\*\*\*\*\*

I	...I...					...MVFI(I,J)...
	1	4	2	3	4	
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

TARLOUL CU REZULTATELE PROBLEMEI

ZOK

I	XP(I)	YP(I)
1	6.0000	12.7000
2	0.0000	6.0000
3	1.0000	-1.0000
4		0.0000
5		1.0000
6		0.0000
7		0.0000
8		4.00
9		4.00
10		4.00
11		4.00
12		11.40
13		0.00
14		7.80
15		7.80
16		7.80
17		7.80
18		3.00
19		7.80
20		4.55
		7.80
		4.55
		4.00
		4.00
		7.07
		7.07
		6.29
		4.00
		4.00
		6.29

MATRICEA FETELOR VIZIBILE

\*\*\*\*\*

I...	...	MVFV(I,J)...
1	4	1
2	8	2
	6	4
	5	4

MATRICEA FETELOR INVIZIBILE

\*\*\*\*\*

I...	...	MVI(I,J)...
1	4	2
2	4	3
3	1	1
4	6	2
5	8	6
6	5	7
7	1	7
8	4	
9	1	
10	2	
11	4	
12	6	
13	8	
14	5	
15	1	
16	2	
17	4	
18	6	
19	8	
	10	
	11	
	12	
	13	
	14	
	15	
	16	
	17	
	18	
	19	
	20	



TABLOUL CU REZULTATELE PROBLEMEI

IOX

	XP(I)	YP(I)
0	4.5000	12.4000
1	0.0000	-1.0000
2	0.0000	0.0000
3	0.0000	0.0000
4	0.00	4.50
5	7.90	0.00
6	7.90	9.00
7	5.40	4.50
8	4.20	4.50
9	5.60	0.00
10	5.60	9.00
11	12.50	4.50
12	5.60	2.89
13	5.02	1.87
14	5.60	6.11
15	5.02	7.13
16	4.92	2.19
17	4.92	6.81
18	6.86	1.87
19	6.86	7.13
20	7.13	1.39
21	7.90	3.05
22	7.90	5.95
23	7.13	7.61

MATRICEA FETELOR VIZIBILE

...MVFV(I,J)...

1	4	1	2	4
2	4	2	3	4
3	4	3	1	4
4	5	7	6	3
5	5	6	7	5

MATRICEA FETELOR INVIZIBILE

...MVFI(I,J)...

1	1	3	2	1
2	8	5	5	8
3	1			
4	2			
5	4			
6	7			
7	3			
8	4			
9	8			
10	5			
11	7			
12	6			
13	2			
14	6			
15	4			
16	8			
17	8			
18	6			
19	9			
20	8			

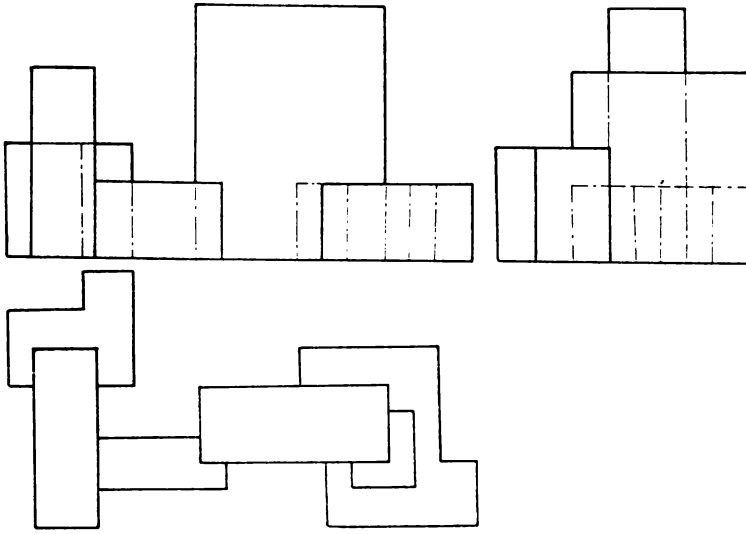


Fig. 7.9

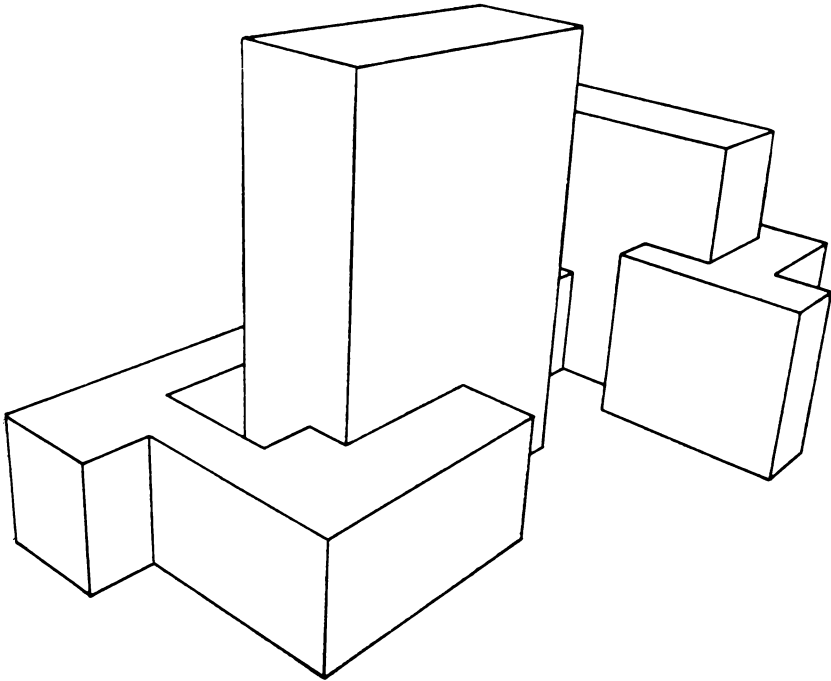


Fig. 7.10

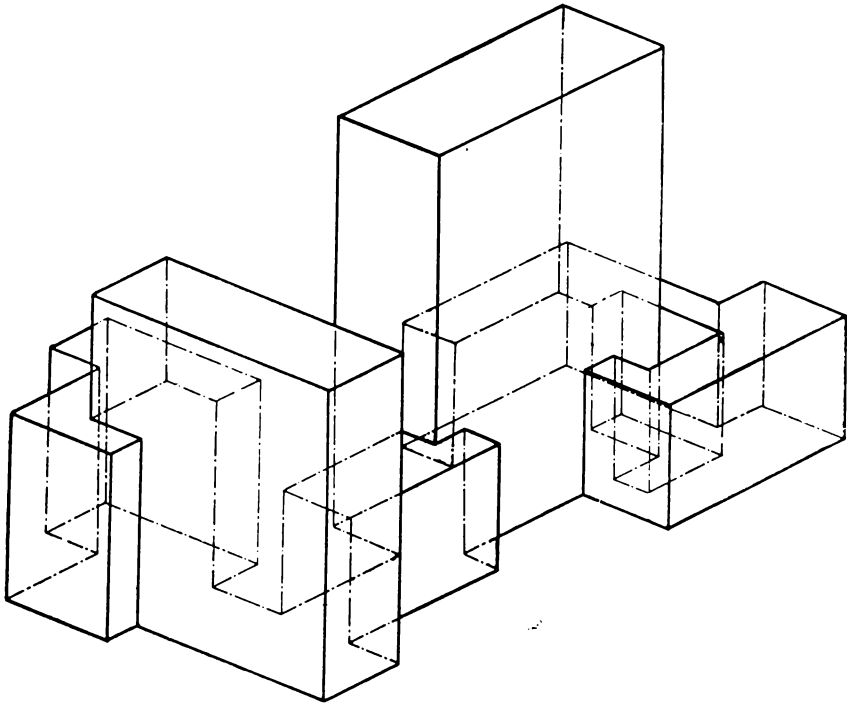


Fig. 7.11

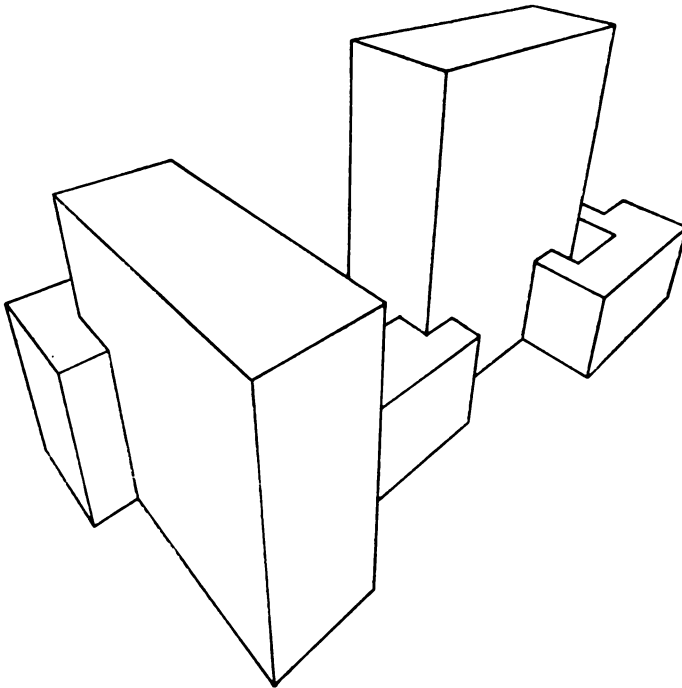


Fig. 7.12

## 7.8.2. Intersecția CUB — OCTAEDRU avînd o diagonală comună

TARLOUL CU DATELE PROBLEMEI

\*\*\*\*\*

	0	0	0	12	14	1	0
	45.0000	40.0000	20.0000	7.2000	7.5000	5.7000	
13.20	8.00	5.70	7.50	13.70	5.70	1.80	8.00
7.50	8.00	11.40	7.50	8.00	0.00	10.15	12.65
10.15	3.35	3.80	2.20	8.00	3.80	4.85	12.60
	4	4	4	4	4	4	4
	5	1	2	5			
	5	2	3	5			
	5	3	4	5			
	5	4	1	5			
	6	2	1	6			
	6	3	2	6			
	6	4	3	6			
	6	1	4	6			
	8	9	6	7	8		
	9	12	10	6	9		
	12	5	11	10	12		
	5	8	7	11	5		
	8	5	12	9	8		
	6	10	11	7	6		
		45.0	40.0	20.0			

TARLOUL CU REZULTATELE PROBLEMEI

\*\*\*\*\*

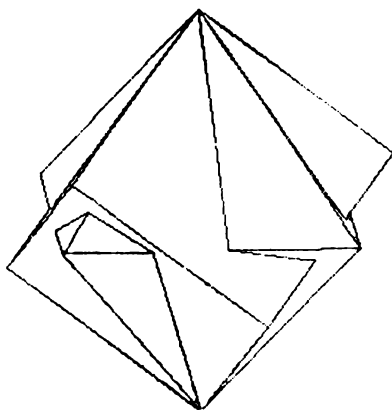
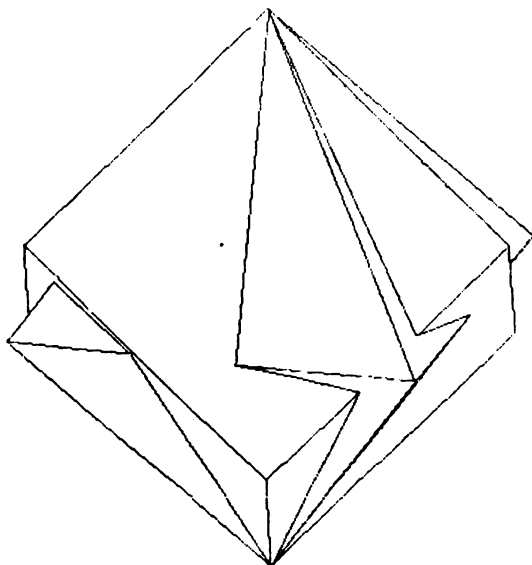
I	XP(I)	YP(I)
45.0000	40.0000	20.0000
-37.4502	59.7603	-0.6519
0.9612		
1	0.53	2.03
2	5.03	2.11
3	4.36	3.27
4	0.53	3.19
5	2.59	5.65
6	2.58	0.00
7	5.59	0.90
8	0.68	3.09
9	0.00	1.94
10	4.24	2.35
11	5.31	3.50
12	1.72	4.24
13	2.59	5.65
14	2.59	5.65
15	2.59	5.65
16	4.75	2.54
17	2.59	5.65
18	2.59	5.65
19	2.59	5.65
20	1.06	3.37
21	2.59	5.65
22	2.59	5.65
23	2.59	5.65
24	2.59	5.65
25	2.59	5.65
26	2.59	5.65
27	2.58	0.00
28	2.58	0.00
29	2.58	0.00
30	4.28	1.93
31	2.58	0.00
32	2.58	0.00

33	2.58	-0.00
34	2.58	0.00
35	2.58	0.00
36	2.58	0.00
37	2.58	0.00
38	2.58	0.00
39	0.69	2.84
40	2.58	-0.00
41	0.89	2.69
42	1.88	2.05
43	2.58	0.00
44	2.58	-0.00
45	2.58	-0.00
46	0.50	2.41
47	2.58	-0.00
48	2.58	0.00
49	2.58	0.00
50	2.22	3.22
51	2.58	0.00
52	2.58	-0.00
53	2.54	5.65
54	2.59	5.65
55	4.51	2.96
56	2.59	5.65
57	3.19	3.24
58	2.59	5.65
59	4.03	2.70
60	2.59	5.65
61	3.02	2.08
62	2.59	5.65
63	2.59	5.65
64	2.59	5.65
65	2.59	5.65
66	2.59	5.65
67	2.59	5.65
68	2.59	5.65
69	0.50	2.72
70	2.58	0.00
71	4.67	2.69
72	2.58	0.00
73	2.58	0.00
74	2.58	-0.00

MATRICEA FETELOR VIZIBILE  
 \*\*\*\*\*

0 ...I... 0 0						...MVFV(I,J)...
1	5	1	2	5		
2	5	2	3	5		
3	5	4	1	5		
4	6	2	1	6		
5	8	9	6	7	8	
6	5	8	7	11	5	





Intersecția dintre un cub și un octaedru care au  
o diagonală verticală comună

Fig. 7.13





TABULCU CU REZULTATELE PROBLEMEI  
 \*\*\*\*\*

I	XP(I)			YP(I)		
	26.0000	26.0000	26.0000	0.5629	0.6052	0.5629
-27.2425	25.0174	-0.7322	0.6411	-0.3334	-0.4122	
0.8665						
1		0.00		0.56		
2		5.13		1.29		
3		1.70		3.50		
4		2.32		5.45		
5		1.85		0.00		
6		4.31		2.48		
7		0.75		4.67		
8		4.78		3.40		
9		3.07		3.24		
10		3.21		1.29		
11		1.77		4.04		
12		1.20		2.64		
13		3.01		1.20		
14		1.33		2.43		
15		3.99		2.65		
16		1.93		4.20		
17		4.31		2.50		
18		3.82		2.15		
19		2.74		2.87		
20		1.87		4.01		

MATRICA FETELOR VIZIBILE  
 \*\*\*\*\*

...I...					...MVIV(I,J)...				
1	4	1	2	3					
2	5	6	7	8					
3									

MATRICA FETELOR INVIZIBILE  
 \*\*\*\*\*

...I...					...MVIV(I,J)...				
1	4	3	3	4					
2	4	3	1	4					
3	1	3	2	5					
4	8	5	7	8					
5									
6	4	1	2	4					
7	2	2	7	6					
8	5	5	7	8					
9	6	6	7	8					
10	6	6	7	8					
11	10	10	11	12					
12	11	11	12	13					
13	11	12	13	14					
14	12	13	14	15					
15	13	13	14	15					
16	13	13	14	15					
17	14	14	15	16					
18	15	15	16	17					
19	17	17	18	19					
20	19	19	20	20					

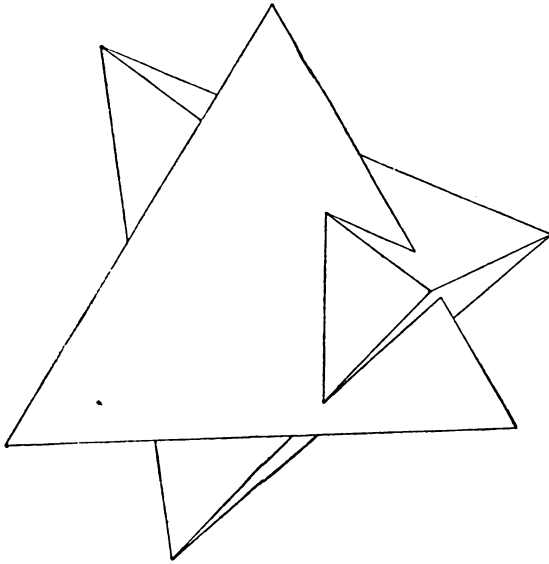


Fig. 7.14

7.8.4. Vedere perspectivă: combinația dintre 10 corpuri cu goluri repetate într-o ordine arbitrară (fig. 7.15 și fig. 7.16)

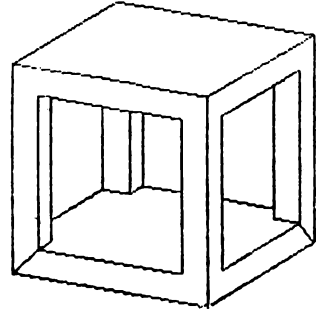


Fig. 7.15

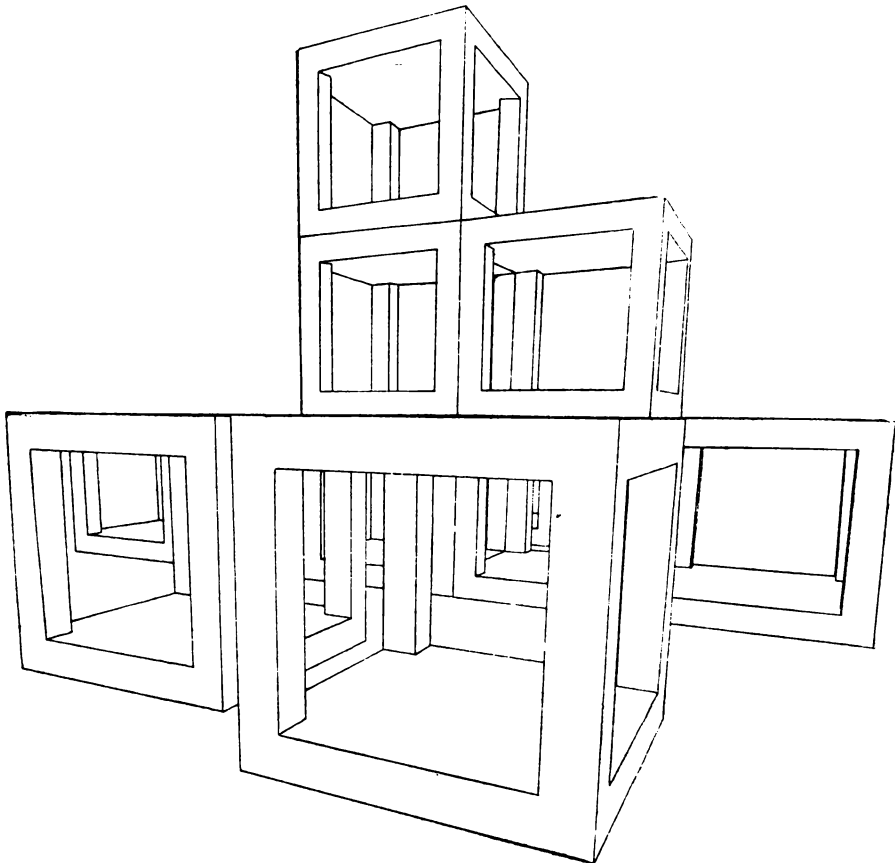


Fig. 7.16

## B. ALGORITMUL APIS

### 7.9. APIS. Algoritm și Program pentru Suprafețe Invizibile

#### 7.9.1. Descrierea generală

Acest al doilea algoritm pentru rezolvarea problemei suprafețelor ascunse face parte din categoria algoritmilor hibridi, calculele realizându-se în „spațiul obiect” iar reprezentarea fiind în „spațiul imagine”.

Ideea de bază este de a realiza un algoritm performant, în cadrul unor restricții bine determinate ale problemei. Restricțiile impuse sînt următoarele:

- 1 — se admit ca intrări pentru algoritm numai corpuri convexe, topologic închise.
- 2 — nu sînt admise intersecții între corpurile din scenă.
- 3 — reprezentarea se face numai într-o fereastră dreptunghiulară arbitrar aleasă de către utilizator.
- 4 — în faza actuală nu este rezolvată problema acoperirilor ciclice dintre corpuri.

Implicațiile pe care le au aceste restricții asupra problemei generale sînt destul de importante ducînd la economii majore de spațiu de memorie cît și de timp de prelucrare.

Algoritmul folosește multe din proprietățile corpurilor convexe care duc la simplificări de fond pentru problemă:

1. Conturul proiecției unui corp convex pe un plan oarecare de proiecție este întotdeauna un poligon convex.
2. Restricția unui poligon convex la o fereastră dreptunghiulară este tot un poligon convex. Mai general chiar poligonul de intersecție dintre 2 poligoane convexe este tot convex.
3. Testul de acoperire între 2 poligoane convexe se poate realiza analizînd un singur punct ce aparține ambelor suprafețe delimitate de poligoane.
4. Procedura de umplere a unui contur convex este foarte simplă. Se poate observa că prima restricție este numai formală, deoarece faptul că sînt admise numai poliedre convexe nu restrînge cadrul problemei. Ținînd seama că orice corp se poate descompune în poliedre convexe problema este aceea de a realiza un preprocesor care să efectueze acțiunea de descompunere.

Cea de a 3-a restricție aduce în multe cazuri scăderi importante ale spațiului de lucru necesar algoritmului. Pentru scenele cu multe corpuri existența unei ferestre de vizualizare duce la restrîngerea numărului de corpuri vizibile. De altfel această restricție nu aduce impedimente majore, deoarece pentru o proiecție centrală spre exemplu dacă fereastra de vizualizare este prea mare, corpurile din margini apar foarte deformate.

De remarcat în plus este că existența ferestrelor de vizualizare a impune realizarea unor proceduri de clipping spațial care vor fi descrise în unul din paragrafele următoare.

Trăsătura cea mai interesantă a algoritmului este modul de vizualizare a structurii de corpuri. Vizualizarea este specifică dispozitivelor de tip raster-scan. Corpurile sînt organizate arborescent de la cele mai „îndepărtate”

(care nu acoperă alte corpuri) spre cele mai apropiate. Fiecare suprafață de corp este acoperită cu un baleiaj negru care șterge ce se găsea dedesubt fiind apoi desenat conturul suprafeței respective. În felul acesta complexitatea scenei crește din spate în față fiecare corp desenat acoperind tot ceea ce se găsește sub proiecția suprafeței sale.

### 7.9.2. Fazele algoritmului

Datele de intrare în algoritmul APIS sînt conținute în 2 fișiere. Un prim fișier „CORP. DAT” conține descrierile secvențiale ale corpurilor din scenă. Descrierile corpurilor sînt asemănătoare celor din algoritmul ALPLA cu deosebirea că fiecare corp în parte au propria definire a virfurilor. Din nou este impusă cerința ca fețele să fie definite în sens trigonometric. Motivele unei asemenea descrieri au fost explicate în paragrafele anterioare.

Al doilea fișier „DATEGEN DAT” conține date despre sistemul de proiecție. Sistemul de proiecție utilizat este cel descris și definit în cap. 1 (Rutine grafice în 3-D).

O primă fază a algoritmului o constituie citirea secvențială a corpurilor, trecerea coordonatelor în sistemul de referință legat de observator și realizarea clipping-ului spațial față de fereastra de vizualizare. Acele corpuri pentru care există proiecție în interiorul ferestrei de vizualizare sînt sortate în ordinea coordonatei  $X_{\min}$  din planul de proiecție iar descrierea rezultată este reținută în memorie.

A doua fază a algoritmului este legată de construirea matricei de acoperire dintre corpuri. Fiecare corp din structură este testat din punct de vedere al adîncimii față de observator față de celelalte corpuri. Între 2 corpuri testate ( $A$  și  $B$ ) nu pot exista decît 3 relații:

$A$  este acoperit de  $B$

$A$  acoperă pe  $B$

$A$  disjunct față de  $B$ .

O matrice binară este completată cu aceste informații. În plus pentru fiecare corp este calculat un coeficient numit „grad de incidență” care specifică numărul de corpuri pe care acesta le acoperă.

A treia fază a algoritmului constă în căutarea unei ordini convenabile în vizualizarea corpurilor. În cazul în care nu există acoperiri ciclice între corpuri se poate găsi întotdeauna cel puțin o ordine de desenare a corpurilor astfel încît scana finală rezultată să fie corectă.

În linii mari se procedează astfel:

a) — Se alege pentru desen primul corp pentru care „gradul de incidență” este 0 (acest corp nu acoperă pe nimeni deci poate fi desenat). Dacă nu există nici un corp cu grad de incidență 0  $\rightarrow$  f.

b) — Se desenează acest corp.

c) — Se selectează din matricea de acoperire toate corpurile care îl acoperă și acestora li se scade gradul de incidență cu 1.

d) — Se setează gradul de incidență al corpului la -- 1.

e) — Se trece la a.

f) — Sfîrșit.

20., 10., 0.	5., 50.,	35., 28.,
20., 20.,	5., 30.,	35., 32.,
10., 20.,	12., 5., 40., 15.	32., 35.,
10., 10.,	5	25., 35.,
20., 10., 20.	4	25., 32.,
20., 20., 20.	4	25., 28.,
10., 20., 20.	4	38., 35.,
10., 10., 30.	2	32., 25., 20.
	4	35., 38., 20.
	3	35., 32., 20.
	4	32., 35., 20.
	4	28., 35., 20.
2, 6, 5, 1	4	25., 32., 20.
	5	25., 28., 20.
3, 7, 6, 2	3	28., 25., 20.
	5	5
4, 8, 7, 3	0., 40.	1, 2, 10, 9, 1
	0., 30.	
4, 1, 5, 8, 4	-10., 50.	3, 11, 10, 2
	-10., 40., 15.	
6, 7, 8, 5	0., 40., 15.	3, 4, 12, 11, 3
	0., 50., 12.	
1, 4, 3, 2, 1	-10., 50., 12.	4, 5, 13, 12, 4
	-10., 40., 12.	5
30., 15.,	6	5, 6, 14, 13, 5
20., 20.,	5	6, 7, 15, 14, 6
15., 20.,	1, 2, 6, 5, 1	7, 8, 16, 15, 7
15., 15., 30.	2	5
20., 20., 30.	2	8, 1, 9, 16, 8
25., 20., 30.	3, 7, 6, 2	9, 10, 11, 12, 13, 14, 15, 16, 9
25., 15., 30.	4, 8, 7, 3	
	4, 1, 5, 8, 4	1, 8, 7, 6, 5, 4, 3, 2, 1
	6, 7, 8, 5	
1, 2, 6, 5, 1	1, 4, 3, 2, 1	
	5	
3, 7, 6, 2	0., 20.,	
	0., 30.,	
4, 8, 7, 3	-15., 30.,	
	-15., 20.,	
4, 1, 5, 8, 4	0., 20., 8.	
	0., 30., 8.	
6, 7, 8, 5	-15., 30., 8.	
	-15., 20., 8.	
1, 4, 3, 2, 1	-7, 5, 25., 16.	
	10.,	
	15.,	
	5., 15.,	
	5., 10.,	
	10., 15.,	
	15., 15.,	
	5., 10., 15.	
1, 2, 6, 5, 1	1, 2, 6, 5, 1	
	3, 7, 6, 2	
3, 7, 6, 2	4, 8, 7, 3	
	1, 5, 8, 4	
4, 8, 7, 3	1, 4, 3, 2, 1	
	5, 6, 9, 5	
4, 1, 5, 8, 4	6, 7, 9, 6	
	7, 8, 9, 7	
6, 7, 8, 5	8, 5, 9, 8	
	5	
1, 4, 3, 2, 1	40., 40.,	
	40., 30.,	
20., 23.,	30., 60.,	
	30., 40.,	
	30., 45., 14.	
	35., 35., 14.	
1, 2, 4, 1	1, 2, 6, 5, 1	
	3, 6, 2	
2, 3, 4, 2	3, 4, 5, 6, 3	
	4, 1, 5, 4	
3, 1, 4, 3	1, 4, 3, 2, 1	
	16	
1, 3, 2, 1	32., 20.	
20., 30.,		
30., 50.,		

Ordinea desenării corpurilor se poate vedea în figurile 7.17—7.25 pentru exemplul considerat.

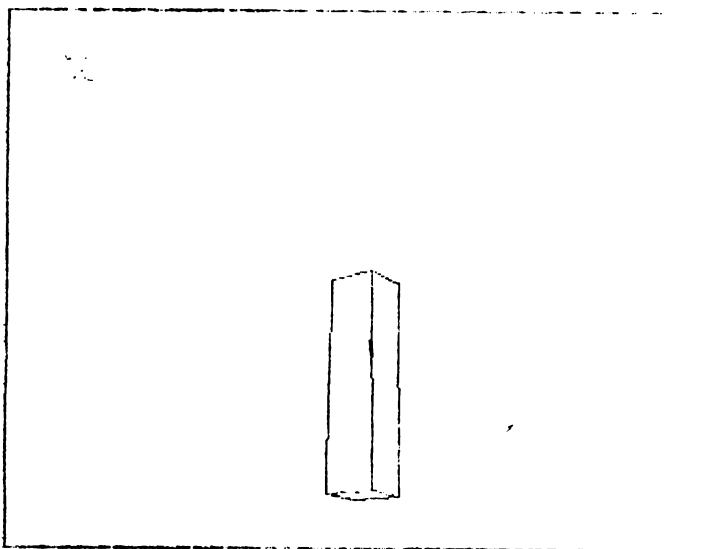


Fig. 7.17

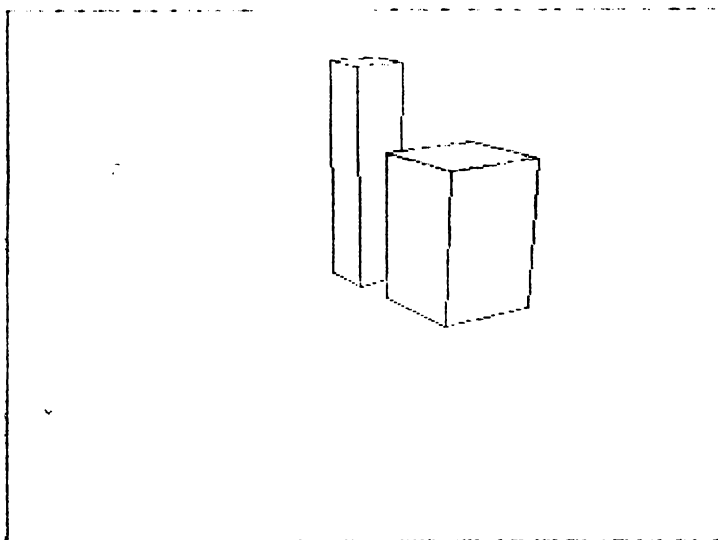


Fig. 7.18

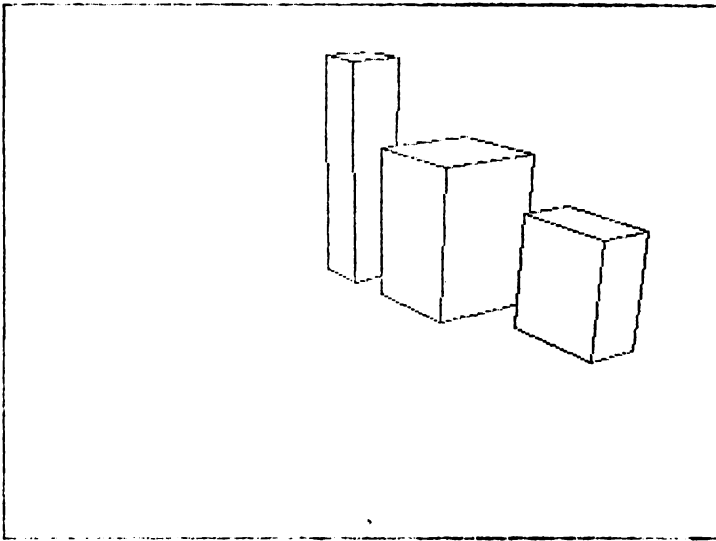


Fig. 7.19

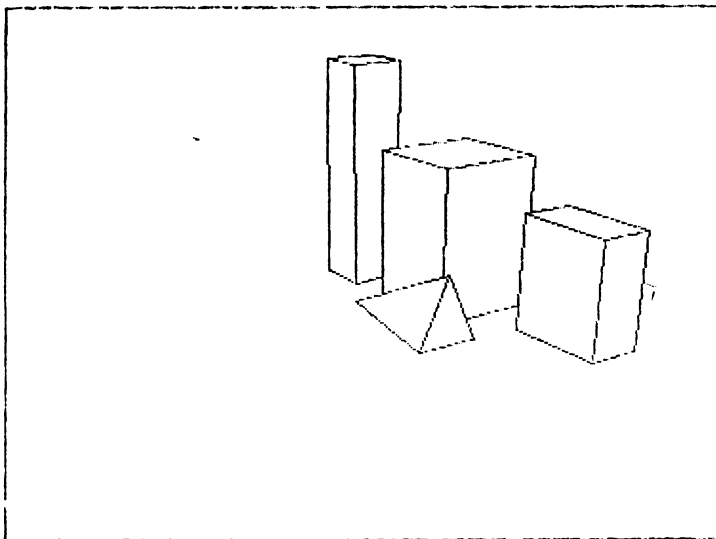


Fig. 7.20

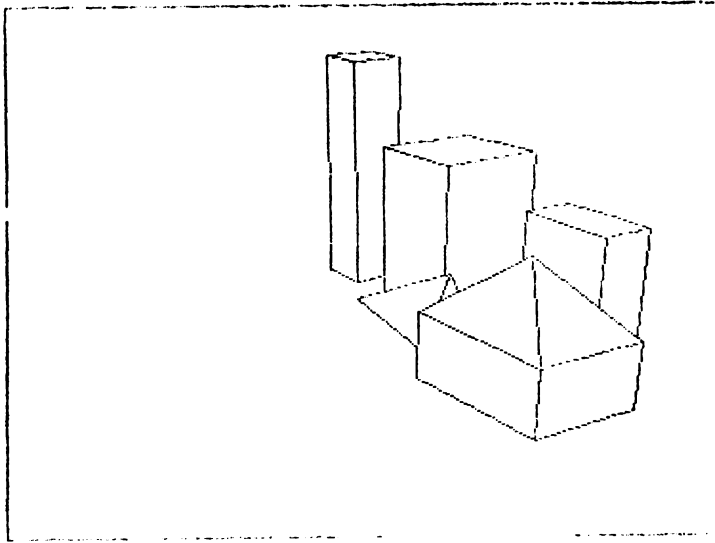


Fig. 7.21

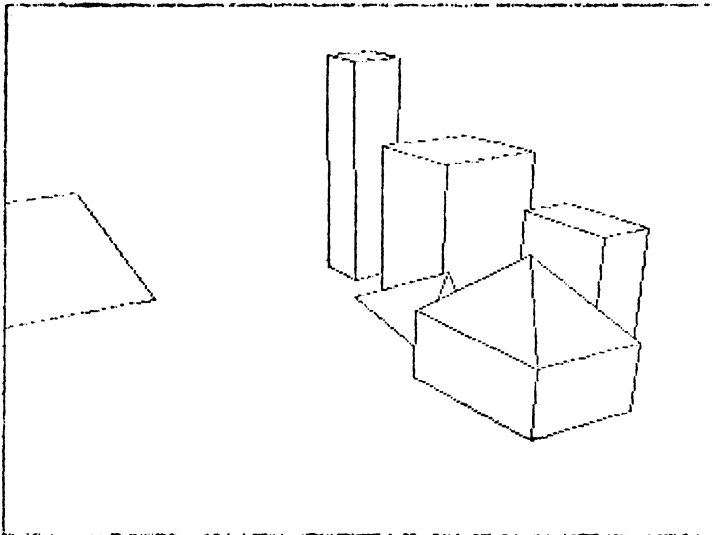


Fig. 7.22



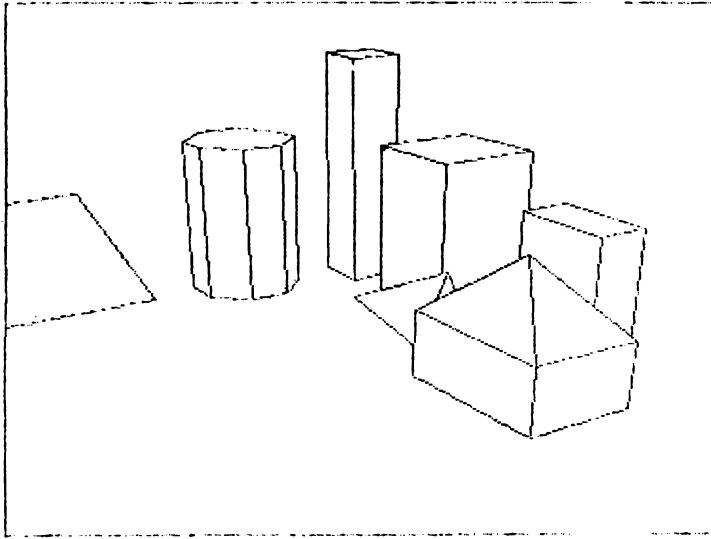


Fig. 7.23

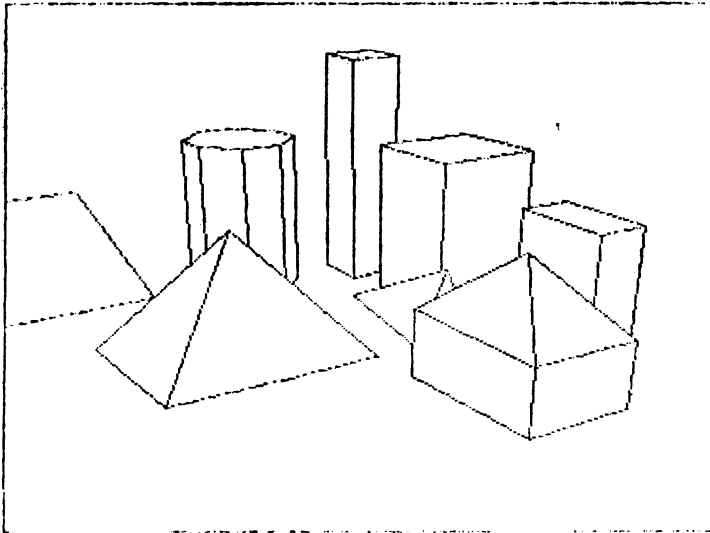


Fig. 7.24

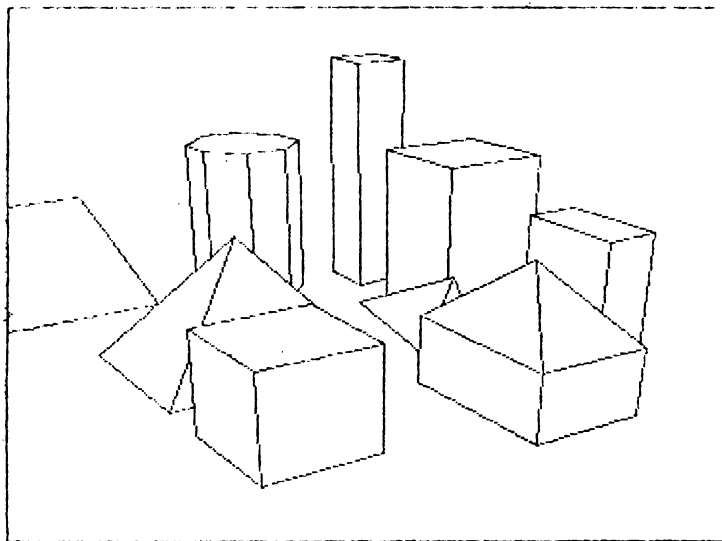


Fig. 7.25

Se poate observa că această procedură nu funcționează în cazul în care există acoperiri ciclice între corpuri. În acest caz la un moment dat nu se poate găsi nici un corp care să nu acopere o parte nedesenată încă.

Trasarea corpurilor se realizează printr-o procedură de umplere a unui contur convex. Prin această metodă tot ce se găsește în spatele corpului curent este acoperit. De remarcat este că în cazul existenței unor display-uri color se pot genera suprafețe în diverse nuanțe și, chiar imagini luminate, cu umbre etc. în funcție de calculele impuse. Pentru display-uri alb negru, procedura de umplere trebuie să fie urmată și de trasarea conturilor fețelor din proiecție.

### 7.9.3. Structurarea memoriei

Versiunea actuală a programului **APIS** este realizată astfel încît admite cel mult 256 de corpuri vizibile în interiorul ferestrei de vizualizare. Acest număr este suficient de mare ținînd seama de rezoluția pe care o au display-urile din dotare. Această restricție este impusă de structurile fixe din memorie. Cele 256 de corpuri convexe pot fi însă oricît de complexe.

Memoria de lucru a fost împărțită în 3 zone.

**ZONA 1** — 8 K — 256 intrări de 32 octeți care conțin date generale despre fiecare corp care are proiecție în fereastra de vizualizare. Aceste date sînt:

a — Dimensiunile  $XMIN$ ,  $XMAX$ ,  $YMIN$ ,  $YMAX$ ,  $ZMIN$ ,  $ZMAX$  ale unui cub din spațiul real care conține corpul respectiv (24 octeți).

b — Referința LEG în zona 1 către înțirarea corespunzătoare următorului corp în ordinea listei *XMIN*. Această listă este actualizată în timpul citirii secvențiale a descrierilor de corpuri (2 octeți).

c — Referința *IREF* în ZONA 2 către intrarea corespunzătoare descrierii contururilor fețelor vizibile ale corpului (2 octeți).

d — *NRF* — indicator cu numărul de fețe vizibile ale corpurilor pentru care există descrieri în ZONA 2 (2 octeți).

e — *IGRI* — gradul de incidență al corpului (2 octeți).

ZONA 2 — zonă dinamică de lucru conținând descrierile contururilor fețelor vizibile pentru fiecare corp. Pentru gestionarea acestei zone se folosesc tehnicile puse la dispoziție de sistemul de operare *RSX*.

Pentru fiecare față vizibilă a unui corp există în ZONA 2 următoarele informații:

a. *NRPF* — numărul de puncte din descrierea conturului feței.

b. *INDC* — un indicator care anunță că fața curentă este ultima din pagina curentă alocată pentru ZONA 2. O față nu își poate întinde descrierea pe două pagini ale ZONEI 2.

c. *A, B, C, D* — coeficienții ecuației planului care conține fața curentă.

d. (*XP(I), YP(I)*),  $I = 1, NRF$  — coordonate în planul de proiecție ale punctelor din descrierea conturului feței (în ordine).

ZONA 3 — 8 K — În prima fază a algoritmului este utilizată ca memorie tampon pentru citirea datelor și stocarea rezultatelor parțiale obținute din clipping-ul fețelor.

În faza a doua reprezintă o matrice binară  $256 \times 256$  care conține matricea de acoperire dintre corpuri.

Valorile din matrice corespund celor 3 cazuri de acoperire

●  $MAT(I, J) = 0$  dacă corpul *I*, acoperă pe *J* sau *I* disjunct față de *J*.

●  $MAT(I, J) = 1$  dacă corpul *I* este acoperit de *J*.

● Informațiile *I* acoperă pe *J* și *I* disjunct față de *J* și se tratează la fel deoarece informația de acoperire este cuprinsă implicit în coeficientul *IGRI*.

Într-o fază viitoare a implementării este posibilă modificarea regiunilor ZONA 1 și ZONA 3 în regiuni dinamice astfel încât să nu mai existe nici o restricție în legătură cu numărul de corpuri admise de algoritm. Este totuși de presupus că gestionarea acestor zone va duce la scăderea performanțelor algoritmului în ce privește timpul de rulare.

#### 7.9.4. CLIPPING

Una din problemele majore puse de algoritm a fost realizarea unei proceduri complete de **clipping spațial** care să reîntoarcă toate informațiile necesare prelucrărilor ulterioare.

Problema clipping-ului a fost structurată pe 3 nivele care relevă cele 3 nivele ascendente din descrierea structurii: *segment-față-corp*.

Clipping-ul la nivel de segment este rezolvat în cap. I (rutine grafice în 3D) de rutina **CLIPS**. Totuși informațiile reîntoarse de rutina **CLIPS** nu erau suficiente pentru realizarea clipping-ului la nivel de față, așa încât s-a realizat rutina **CLIPS 1** care îmbogățește puțin numărul rezultatelor.

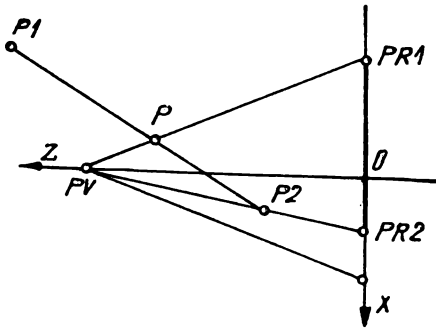


Fig. 7.26

$P1-P2$ , poziția  $P1$  pe plan nu este corectă ( $P1$  se găsește în așa numitul spațiu virtual).

În acest caz trebuie găsit un punct pe segmentul  $P1P2$  pentru care proiecția există. Se alege acest punct cel în care  $P1-P2$  intersectează prima dată planurile piramidei de vedere (punctul  $P$ ). Proiecția acestui punct se va găsi întotdeauna pe una din laturile ferestrei de vizualizare din planul de proiecție.

A doua procedură de clipping — **CLIPF** — realizează decuparea unei fețe poligonale convexe după fereastra de vizualizare. Important este faptul că se modifică definirea feței introducându-se punctele de intersecție cu fereastra de vizualizare (fig. 7.27).

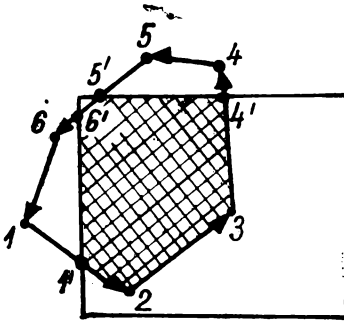


Fig. 7.27

Se observă că dacă descrierea inițială a feței era  $1-2-3-4-5-6-1$  după realizarea clippingului la nivel de față descrierea devine  $1'-2'-3'-4'-5'-6'-1'$ .

Decuparea se face numai pentru fețele efectiv vizibile pentru care descrierea rămâne în sens trigonometric.

În forma actuală procedura este particulară funcționând numai pentru poligoane convexe intersectate cu o fereastră de vizualizare dreptunghiulară. Pentru generalizare se preconizează realizarea unei proceduri rapide de calcul a intersecției dintre două poligoane oarecare. O

astfel de procedură este foarte utilă în rezolvarea cazurilor de acoperiri ciclice și a poliedrelor neconvexe.

**CLIPF** reîntoarce și un indicator de invizibilitate a feței respective (fața nu este cuprinsă în fereastră sau este declarată invizibilă prin calculul produsului vectorial după cum s-a specificat în capitolul anterior).

Informațiile reîntoarse de **CLIPF** sînt centralizate de procedura **CLIPC** (clipping la nivel de corp) care are și menirea de a completa ZONA 1 respectiv ZONA 2 cu noile date despre corp.

Rutina este poate complicată la prima vedere dar trebuie ținut seama că ea rezolvă toate cazurile de clipping, inclusiv cele destul de delicate care apar în cazul proiecției centrale, cînd există vîrfuri aflate în spatele punctului de vedere (vezi fig. 7.26). Tratarea acestui caz foarte interesant face posibilă plasarea punctului de vedere în orice poziții în raport cu structura de corpuri (bineînțeleas însă nu în interiorul corpurilor).

În figură se poate observa faptul că dacă segmentul spațial real este

## 7.9.5. Ierarhizarea corpurilor

Procedura **IERARH** este componenta programului care realizează efectiv acoperirile între corpurile structurii. Conform cu mențiunile anterioare, în cazul poliedrelor convexe (ale căror proiecții sînt contururi convexe) problema acoperirii se rezumă la testarea adîncimii unui singur punct comun suprafețelor proiectate.

Există două aspecte esențiale:

1. Stabilirea existenței suprapunerii a 2 poligoane în planul de proiecție.
2. Testul de adîncime în cazul existenței suprapunerii.

În paragraful 6.2.2 s-au dat cîteva indicații despre modul posibil de stabilire a suprapunerii, care pot fi regăsite și în procedură.

Se poate indica însă succesiunea de calcule care duc la stabilirea suprapunerii:

1. Test *minmax* pe coordonata  $x$ .

Din cauza existenței listei ordonate a corpurilor după  $X_{min}$  testul este deosebit de puternic. În momentul cînd pentru corpul  $I$  se găsește un alt corp  $J$  din listă a.î.

$X_{minJ} \geq X_{minI}$  nici un alt corp de la  $J$  încolo nu mai poate intersecta pe  $I$  și deci se poate trece la testarea corpului următor în listă

2. Test *minmax* pe  $y$
3. Test *minmax* pe  $z$ .

Testul *minmax* pe  $z$  (în cazul cînd testele *minmax* pe  $x$  respectiv  $y$  au arătat posibilitatea suprapunerii) poate ușura foarte mult calculele. Se poate observa că dacă

$Z_{minI} > Z_{maxJ}$  atunci corpul  $I$  este în fața corpului  $J$ .

$Z_{maxI} < Z_{minJ}$  atunci corpul  $J$  este în fața corpului  $I$ .

Nu există nici un impediment în a considera relațiile ca atare chiar dacă nu apare suprapunere efectivă între contururi.

4. Căutarea unei intersecții între contururi în planul de proiecție. La găsirea primei intersecții căutarea se încheie.

5. Testul de interioritate al unui contur față de celălalt.

Dacă se constată că un punct al conturului  $I$  este interior conturului  $J$  sau invers, atunci se poate considera interioritatea întregului contur  $I$  în  $J$  respectiv  $J$  în  $I$ .

Pentru calculul de interioritate în cazul poligoanelor convexe s-a recurs la un calcul special foarte simplu.

Ținînd seama de ordinea fixă de descriere a poligonului, se calculează semnul valorii pe care o dă un vîrf  $P$  al unui poligon introdus în ecuațiile segmentelor ce descriu conturul celuilalt poligon. Dacă semnul se menține constant  $P$  este interior poligonului, altfel este exterior. (Vezi fig. 7.28).

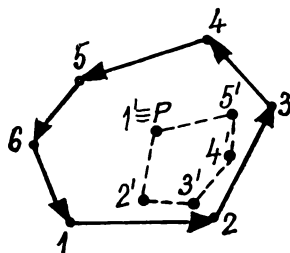


Fig. 7.28

Se poate observa că cele 5 teste se fac în ordinea crescătoare a complexității. Numai puține dintre cazurile de suprapunere ajung pînă la testul 5, majoritatea oprindu-se la foarte simplele teste 1, 2, 3.

După stabilirea existenței suprapunerii se trece la testul de adâncime pentru un punct comun (intersecția calculată în cazul 4 sau punctul  $P$  în cazul 5). Adâncimea față de observator a acestui punct în cele 2 plane reale din spațiu stabilește relația de acoperire dintre corpurile considerate. Informația de acoperire este reținută în ZONA 3 și în coeficientul IGRI al fiecărui corp.

```

PROGRAM PRINCIPAL
  INTEGER IRRB(9), IWRB(11), DSW, IVECT1(4096)
  COMMON /ZONA1/XMIN(256), XMAX(256), YMIN(256), YMAX(256), ZMIN(256),
  1 ZMAX(256), LEG(256), IREF(256), NRF(256), IGRI(256)
  COMMON /ZONA3/MAT(256,16)
  COMMON /ZONA2/VECT1(2048)
  COMMON /PAGE/IWTR
  EQUIVALENCE (VECT1, IVECT1)
  CALL ASSIGN(1, 'DATEGEN.DAT')
  READ(1,1) XC, YC, ZC, XS, YS, ZS, D, DX, DY, ITIP
  CALL INISF
  CALL SETC(XC, YC, ZC)
  CALL SETS(XS, YS, ZS)
  CALL SEID(0)
  CALL SETDIM(DX, DY)
  READ(1,1) XV, YV, ZV
  CALL SETIV(XV, YV, ZV)
  1 FORMAT(9F8.2, 2, 15)
  CALL CLOSE(1)
  CALL ASSIGN(1, 'CORP.DAT')
  CALL COMPLE(ITIP, 1, INDF, NRI)
  CALL CLOSE(1)
  CALL IERARH(INDF, ITIP)
  CALL ORDRES(NRI)
  ENI

C *****
C
C Procedura - COMPLE - pentru completarea legaturilor in
C lista XMIN pentru extensiunile corpurilor din structura
C *****
C
C SUBROUTINE COMPLE(ITIP, NRLOG, INDF, NRI)
C *****
C
C Parametri de apel :
C ITIP - tipul proiectiei
C NRLOG - numarul logic al fisierului de date
C INDF - indicele de inceput in lista XMIN reinters
C NRI - numarul real de corpuri din structura reinters
C *****
C
  1 COMMON /ZONA1/XMIN(256), XMAX(256), YMIN(256), YMAX(256), ZMIN(256),
  2 ZMAX(256), LEG(256), IREF(256), NRF(256), IGRI(256)
  NRI=1
  1 CALL CLIPC(NRI, IIF, ISF, ITIP, NRLOG)
  IF(ISF.EQ.1) GOTO 7
  IF(IIF.EQ.0) GOTO 1
  INDF=1
  LEG(1)=0
  2 NRI=NRI+1
  3 CALL CLIPC(NRI, IIF, ISF, ITIP, NRLOG)
  IF(ISF.EQ.1) GOTO 7
  IF(IIF.EQ.0) GOTO 3
  I=INDF
  IF(XMIN(NRI)-XMIN(I)) 4,4,5
  4 LEG(INDF)=INDF
  INDF=NRI
  GOTO 2
  5 K=1
  I=LEG(I)
  IF(I.EQ.0) GOTO 6
  IF(XMIN(NRI)-XMIN(I)) 6,6,5
  6 LEG(INDF)=I
  LEG(NRI)=I
  GOTO 2
  7 NRI=NRI+1
  JO=(NRI-1)/16 + 1
  DO 1000 I=1, NRI
  DO 1000 J=1, JO
  1000 MAT(I, J)=0
  RETURN
  END

```

```

C *****C
C PROCEURA -IERARH- PENTRU COMPLETAREA MATRICII DE ACOPERIRE
C SI A GRADULUI DE INCIDENTA PENTRU FIECARE CORP DIN STRUCTURA
C *****C
C SUBROUTINE IERARH(INDP,ITIP)
C *****
C PAFAMETRII DE APEL SINT :
C INDP - INCEPUTUL LISTEI XMIN A CORPURILOR DIN STRUCTURA
C ITIP -TIPUL DE PROIECTIE
C *****C
C INTEGER GRI,IVECT(4095)
C COMMON /ZONA1/XMIN(256),XMAX(256),YMIN(256),YMAX(256),ZMIN(256),
1 ZMAX(256),LEG(256),IREF(256),NRF(256),GRI(256)
2 /ZONA2/VECT(2048)
3 /SPD /XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,DIST,DX,DY
EQUIVALENCE(VECT,IVECT)
I=INDP
GOTO 10
100 I=LEG(I)
10 J=LEG(I)
3000 IF(J.EQ.0) RETURN
IF(YMIN(J).GE.XMAX(I)) THEN
GOTO 100
ENDIF
IF((YMIN(J).GE.YMAX(I)).OR.(YMAX(J).LE.YMIN(I))) THEN
GOTO 150
ENDIF
IF(ZMIN(J).GE.ZMAX(I)) THEN
CORPUL I ACOPERA PE J
CALL UMPLC(J,I)
GOTO 150
ENDIF
IF(ZMAX(J).LE.ZMIN(I)) THEN
CORPUL J ACOPERA PE I
CALL UMPLC(I,J)
GOTO 150
ENDIF
CORPURILE SINT POSIBIL INTERSECTABILE IN PLANUL DE PROIECTIE
PENTRU TESTUL DE ADINCIME SE CAUTA O INTERSECIE INTRE CONTURURI
ASSIGN 150 TO IET2
INC1=IREF(I)
CALL PAG(INC1,ICRT1,INC11)
DO 50 K=1,NRF(I)
NRPF1=IVECT(2*INC11-1)
INC21=INC11+3
DO 40 L=1,NRPF1-1
INC21=INC21+2
X1=VECT(INC21)
Y1=VECT(INC21+1)
X2=VECT(INC21+2)
Y2=VECT(INC21+3)
INCJ=IREF(J)
CALL PAG(INCJ,ICRTJ,INC1J)
DO 30 M=1,NRF(J)
NRPFJ=IVECT(2*INC1J-1)
INC2J=INC1J+3
DO 20 N=1,NRPFJ-1
INC2J=INC2J+2
X3=VECT(INC2J)
Y3=VECT(INC2J+1)
X4=VECT(INC2J+2)
Y4=VECT(INC2J+3)
IF(MIN(X3,X4).GE.MAX(X1,X2).OR.MAX(X3,X4).LE.MIN(X1,X2))
1 GOTO 20
IF(MIN(Y3,Y4).GE.MAX(Y1,Y2).OR.MAX(Y3,Y4).LE.MIN(Y1,Y2))
1 GOTO 20
PROD=(X4-X3)*(Y2-Y1)-(Y4-Y3)*(X2-X1)
IF(PROD.EQ.0) GOTO 20
C CALCULUL COEFICIENTULUI C2
COEF=((Y3-Y1)*(X2-X1)-(X3-X1)*(Y2-Y1))/PROD
IF(COEF.LT.0..OR.COEF.GT.1.) GOTO 20
C CALCULUL COEFICIENTULUI C1
COEF=((X1-X3)*(Y4-Y3)-(Y1-Y3)*(X4-X3))/PROD
IF(COEF.LT.0..OR.COEF.GT.1.) GOTO 20
C CALCULUL PUNCTULUI DE INTERSECIE
XINT=X1+COEF*(X2-X1)
YINT=Y1+COEF*(Y2-Y1)
ASSIGN 20 TO IET1
GOTO 1000
C SECVENTA DE CALCUL A ADINCIMII PUNCTULUI ( XINT , YINT )
C IN FUNCTIE DE TIPUL DE PROIECTIE
C INITIAL PAGINA CURENTA ESTE PAGINA CE CONTINE FETELE CORPULUI J

```

```

1000 IF(ITIP.EQ.0) GOTO 21
      DFV=DIST*VECT(INC1J+3)+VECT(INC1J+4)
      DJ=XINT*VECT(INC1J+1)+YINT*VECT(INC1J+2)+VECT(INC1J+4)
      DFVJ=SQRT(XINT*XINT+YINT*YINT+DIST*DIST)
      ADJ=DFV*DFVJ/(DFV-DJ)
      GOTO 22
21 1 ADI=(XINT*VECT(INC1J+1)+YINT*(INC1J+2)+VECT(INC1J+4)
22 1 /VECT(INC1J+3)
      IF(ICRTI.EQ.ICRTJ) GOTO 23
      CALL UMAP
      CALL MAPS(ICRTI)
23 IF(11IP.EQ.0) GOTO 24
      DPV=DIST*VECT(INC1I+3)+VECT(INC1I+4)
      DJ=XINT*VECT(INC1I+1)+YINT*VECT(INC1I+2)+VECT(INC1I+4)
      ADI=DPV*DFVJ/(DFV-DJ)
      GOTO 25
24 1 ADI=(XINT*VECT(INC1I+1)+YINT*VECT(INC1I+2)+VECT(INC1I+4)
25 1 /VECT(INC1I+3)
      IF(ICRTI.EQ.ICRTJ) GOTO 28
      CALL UMAP
      CALL MAPS(ICRTJ)
28 CONTINUE
      IF(ABS(ADI-ADJ).LT.0.1) GOTO 29
      IF(ADI-ADJ)28,29,27
      CORPUL I ACOPERA CORPUL J
26 CALL UMPLC(J,I)
      GOTO IET2
      CORPUL J ACOPERA CORPUL I
27 CALL UMPLC(I,J)
      GOTO IET2
29 GOTO IET1
      C TERMINAREA SECVENTEI
20 CONTINUE
      IF(IVECT(2*INC1J).EQ.0) THEN
          INC1J=INC1J+5+2*NRPFJ
          ELSE
              ICRTJ=ICRTJ+1
              CALL UMAP
              CALL MAPS(ICRTJ)
              INC1J=1
          ENDIF
30 CONTINUE
      IF(ICRTI.EQ.ICRTJ) GOTO 40
      CALL UMAP
      CALL MAPS(ICRTI)
40 CONTINUE
      IF(IVECT(2*INC1I).EQ.0) THEN
          INC1I=INC1I+5+2*NRPFI
          ELSE
              ICRTI=ICRTI+1
              CALL UMAP
              CALL MAPS(ICRTI)
              INC1I=1
          ENDIF
50 CONTINUE
      C SE TESTEZA INTERIORITATEA UNUI CORP FATA DE ALTUL IN PLANUL DE
      C PROIECTIE
      ITR=0
      INCI=IREF(I)
      CALL PAG(INCI,ICRTI,INC1I)
      DO 60 K=1,NRF(I)
          NRPFI=IVECT(2*INC1I-1)
          INC2I=INC1I+3
          DO 70 L=1,NRPFI-1
              INC2I=INC2I+2
              XINT=VECT(INC2I)
              YINT=VECT(INC2I+1)
              INCJ=IREF(J)
              CALL PAG(INCJ,ICRTJ,INC1J)
              DO 80 M=1,NRF(J)
                  NRPFJ=IVECT(2*INC1J-1)
                  INC2J=INC1J+3
                  DO 90 N=1,NRPFJ-1
                      INC2J=INC2J+2
                      X2=VECT(INC2J)
                      Y2=VECT(INC2J+1)
                      X3=VECT(INC2J+2)
                      Y3=VECT(INC2J+3)
                      IF((XINT-X2)*(Y3-Y2)+(YINT-Y2)*(X2-X3).GT.0) GOTO 81
90 CONTINUE
      C UN PUNCT AL CORPULUI I INTERIOR UNEI FETE A CORPULUI J
      C ASSIGN 81 TO IET1
      GOTO 1000
81 IF(IVECT(2*INC1J).EQ.0) THEN
          INC1J=INC1J+5+2*NRPFJ

```



```

                                ELSE
                                ICRTJ=ICRTJ+1
                                CALL UMAP
                                CALL MAPS(ICRTJ)
                                INCIJ=1
                                ENDIF
30      CONTINUE
        IF(ICRTI.EQ.ICRTJ) GOTO 70
        CALL UMAP
        CALL MAPS(ICRTI)
70      CONTINUE
        IF(IVECT(2*INCI1).EQ.0) THEN
            INCI1=INCI1+5+2*NKRFI
            ELSE
            ICRTI=ICRTI+1
            CALL UMAP
            CALL MAPS(ICRTI)
            INCI1=1
            ENDIF
40      CONTINUE
        IF(ITR.EQ.1) GOTO 1500
        ITR=ITR+1
        I1=1
        I=J
        J=I1
1500     ASSIGN 1500 TO IET2
        I1=1
        I=J
        J=I1
180      J=EG(J)
        IF(J.NE.0) GOTO 3000
        GOTO 100
        END

C *****
C
C PROCEDURA - ORDIDES - CARE STABILESTE ORDINEA DE DESENARE
C A CORPURIOR DIN STRUCTURA SI APELEAZA RUTINA DE DESEN
C *****
C
C SUBROUTINE ORDIDES(NRI)
C *****
C PARAMETRI DE APEL :
C NRI - NUMARUL DE CORPURI DIN STRUCTURA
C *****
C
C INTEGER GRI
C COMMON /ZONA1/XMIN(256), XMAX(256), YMIN(256), YMAX(256), ZMIN(256),
1  ZMAX(256), LEG(256), IREF(256), NRF(256), GRI(256)
2  /SPD/XV, YV, ZV, XC, YC, ZC, XS, YS, ZS, DIST, DX, DY
C CALL ASSIGN(1, I1:7)
C CALL INI(1)
C CALL WNDW(-DX, -DY, DX, DY, 5, 5, 750)
C CALL CLEAR(2)
C CALL PLOT1(-DX, -DY, 0)
C CALL PLOT1(DX, -DY, 1)
C CALL PLOT1(DX, DY, 1)
C CALL PLOT1(-DX, DY, 1)
C CALL PLOT1(-DX, -DY, 1)
5  DO 1 I=1, NRI
    IF(GRI(I).NE.0) GOTO 1
    CALL DEGEN(I)
    GRI(I)=-1
    DO 3 J=1, NRI
        CALL TEST(I, J, ITEST)
        IF(ITEST.EQ.0) GOTO 3
        GRI(J)=GRI(J)-1
3  CONTINUE
    GOTO 5
1  CONTINUE
RETURN
END

```

```

*****
PROCEDURA DE DESEN A UNUI CORP.
SE UMPLI E CONTURUL FIECAREI FETE CU FOND INCHIS DUPA
CARE SE TRASEAZA CONTURUL CU LINIE DESCHISA
*****

SUBROUTINE DESEN(NRI)
*****

PARAMETRI DE APEL :
NRI - INDICELE CORPULUI DIN STRUCTURA
*****

LOGICAL*1 A1,A2
INTEGER IVECT(4096)
COMMON /ZONA1/XMIN(256),XMAX(256),YMIN(256),YMAX(256),ZMIN(256),
1 ZMAX(256),LEG(256),IREF(256),NRF(256),IGRI(256)
2 /ZONA2/VECT(2048)
EQUIVALENCE (VECT, IVECT)
DATA A1,A2/'@','?'
NRF=NRF(NRI)
IREF=IREF(NRI)
CALL PAG(IREFC,NRFAG,IND)
DO 1 I=1,NRF
NRF=IVECT(2*IND-1)
CALL TIPVEC(A1,A1)
CALL FILL(VECT(IND+5),NRF)
CALL TIPVEC(A2,A2)
CALL FLG(VECT(IND+5),NRF)
IF(IVECT(2*IND).EQ.1) THEN
CALL UMAP
NRFAG=NRFAG+1
CALL MAPS(NRFAG)
IND=1
GOTO 1
ENDIF
IND=IND+5+2*NRF
CONTINUE
RETURN
END

```

\*\*\*\*\*
PROCEDURA DE CLIPPING PENTRU UN CORP CONVEX
\*\*\*\*\*

SUBROUTINE CLIPC ( NRI, IIF, ISF, ITIP, NRLOG )
\*\*\*\*\*

Parametri de apel :
NRI = numarul intrarii din ZONA1
IIF = indicator de invizibilitate a corpului
ISF = indicator de sfirsit de fisier
ITIP = tipul proiectiei
NRLOG = numarul logic al fisierului de date
\*\*\*\*\*

INTEGER GRI, IVECT(4096), IVECT1(4096)
COMMON /ZONA1/XMIN(256),XMAX(256),YMIN(256),YMAX(256),ZMIN(256),
1 ZMAX(256),LEG(256),IREF(256),NRF(256),GRI(256)
2 /ZONA2/VECT(2048)
3 /ZONA3/VECT1(2048)
4 /SPD /XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,DIST,DX,DY
EQUIVALENCE (VECT, IVECT), (VECT1, IVECT1)
C citirea definitiei corpului din fisierul de intrare
1 READ (NRLOG,1,END=1000) NRFC
2 FORMAT (I4)
C citirea coordonatelor virfurilor si transformarea lor in coordonate
3 ale sistemului de proiectie
DO 3 I=1,NRFC\*3,3

```

      READ (NRLOG,2)X,Y,Z
2     FORMAT(3F10.4)
      CALL NEWC (X,Y,Z,VECT1(I),VECT1(I+1),VECT1(I+2))
3     CONTINUE
C citirea numarului de fete ale corpului
      READ (NRLOG,1)NRFT
C initializari
      VM=10.**10
      XMIN(NRI)=VM
      XMAX(NRI)=-VM
      YMIN(NRI)=VM
      YMAX(NRI)=-VM
      ZMIN(NRI)=VM
      ZMAX(NRI)=-VM
      NRC=6*NRPC+1
      IF (NRI.EQ.1) THEN
          INTR=1
          NRPAG=0
      ENDIF
      ASSIGN 5 TO IET1
      IIF=0
C ciclu pentru toate fetele corpului
      DO 9 I=1,NRFT
C citirea numarului de puncte ale fetei
      READ (NRLOG,1) NRPF
C citirea descrierii fetei curente
      READ (NRLOG,4) (IVECT1(NRC+J),J=0,NRPF-1)
4     FORMAT(20I4)
C apelul rutinei de clipping pentru fata
      NR1=3*NRPC+(NRPF+1)/2+1
      CALL CLIPF(VECT1,NRPC,IVECT1(NRC),NRPF,VECT1(NR1),ITIP,
1         IVE,A,B,C,D,XMIN(NRI),XMAX(NRI),YMIN(NRI),YMAX(NRI)
2         ZMIN(NRI),ZMAX(NRI))
C testul de vizibilitate
      IF (IVE.EQ.0) GOTO 9
      GOTO IET1
C initializari in cazul primei fete vizibile a corpului
5     IIF=1
      NR1(NRI)=1
      ASSIGN 6 TO IET1
C calcule si remapari ale zonei ZONA2
      IF (INTR+5+2*NRPF.GT.2048) THEN
          NRPAG=NRPAG+1
          CALL UMAP
          CALL MAPS(NRPAG)
          INTR=1
      ENDIF
      IREF(NRI)=2048*NRPAG+INTR
      INTR1=INTR
      GOTO 7
C cazul general pentru o fata vizibila
6     IF (INTR+5+2*NRPF.GT.2048) THEN
          IVECT(2*INTR1)=1
          NRPAG=NRPAG+1
          CALL UMAP
          CALL MAPS(NRPAG)
          INTR=1
      ELSE
          IVECT(2*INTR1)=0
      ENDIF
      INTR1=INTR
      NR1(NRI)=NR1(NRI)+1
C copierea rezultatelor in ZONA2
7     IVECT(2*INTR-1)=NRPF
      VECT(INTR+1)=A
      VECT(INTR+2)=B
      VECT(INTR+3)=C
      VECT(INTR+4)=D
      INTR=INTR+4
      DO 8 J=1,2*NRPF
8     VECT(INTR+J)=VECT1(NR1+J-1)
      INTR=INTR+2*NRPF+1
9     CONTINUE
      RETURN
1000 ISF=1
      RETURN
      END

```

```

C *****
C Procedura - CLIPF - pentru realizarea clipping-ului unei
C fete poligonale convexe. Descrierea fetei este realizata in sens
C trigonometric. Clipping-ul se face fata de fereastra din planul
C de proiectie de dimensiuni DX , DY. Procedura reinvoarca noua
C descriere a fetei poligonale cuprinsa efectiv in fereastra de
C vizualizare. Se reinvoarca si un indicator de vizibilitate - IVF
C Se reinvoarca in plus coeficientii planului care contine fata si
C dimensiunile extreme ale fetei functie de tipul de proiectie.
C *****
C
C SUBROUTINE CLIPF (V1, NRPC, IV1, NRPF, VL, ITIP, IVF, A, B, C, D
C , XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX)
C
C *****
C Parametrii procedurii :
C V1 - matrice de dimensiune 3xNRPC cu coordonate ale punctelor
C NRPC - numarul de puncte ale corpului
C IV1 - vector de dimensiune NRPF cu punctele fetei curente
C NRPF - numarul de puncte din descrierea fetei. In final contine
C numarul real de puncte din descrierea fetei dupa clipping.
C VL - vector de dimensiune variabila cu coordonatele proiectiei.
C ITIP - tipul proiectiei
C IVF - indicator de vizibilitate a fetei 0=fata invizibila
C A, B, C, D
C - coeficientii planului care contine fata
C XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX
C - dimensiunile extreme ale fetei ( functie de ITIP )
C *****
C DIMENSION V1(3, NRPC), IV1(NRPF), VL(2, NRPF)
C COMMON /SPD /XV, YV, ZV, XC, YC, ZC, XS, YS, ZS, DIST, DX, DY
C /COR T/XCOLT(4), YCOLT(4)
C calculul coeficientilor planului fetei
C TYPE 1200, ((V1(I1, J1), I1=1, 3), J1=1, NRPC)
C1200 FORMAT(3F10.2)
C TYPE 1201, (IV1(J), J=1, NRPF)
C1201 FORMAT(20I4)
C I=1
C12 I=1
C IP1=IV1(I)
C IP2=IV1(I+1)
C IP3=IV1(I+2)
C DX1=V1(1, IP1)-V1(1, IP2)
C DY1=V1(2, IP1)-V1(2, IP2)
C DZ1=V1(3, IP1)-V1(3, IP2)
C X2=V1(1, IP2)-V1(1, IP3)
C Y2=V1(2, IP2)-V1(2, IP3)
C DZ2=V1(3, IP2)-V1(3, IP3)
C A= DY1*DZ2 - DZ1*DY2
C B= DZ1*DX2 - DX1*DZ2
C C= DX1*DY2 - DY1*DX2
C PROD=SQRT(A*A+B*B+C*C)
C IF (PROD.EQ.0.) THEN
C I=I+1
C GOTO 12
C ENDIF
C A=A/PROD
C B=B/PROD
C C=C/PROD
C D= A*V1(1, IP2)-B*V1(2, IP2)-C*V1(3, IP2)
C calculul vizibilitatii fetei functie de tipul de proiectie
C IVF=1
C IF (ITIP.EQ.0) GOTO 1
C D1= DIST*C+D
C IF (D1.GT.0) GOTO 2
C IVF=0
C RETURN
C 1
C IF (C.GT.0.) GOTO 2
C IVF=0
C RETURN
C 2
C CONTINUE
C fata este vizibila ; se testeaza daca intersecteaza fereastra de
C vizualizare . Daca da , se calculeaza noul contur.
C IVF=0
C IC=0
C IVF=0
C ASSIGN 4 TO IET1
C J=1
C DO 3 I=1, NRPF-1
C IP1=IV1(I)
C IP2=IV1(I+1)
C CALL CLIPF(V1(1, IP1), V1(2, IP1), V1(3, IP1), V1(1, IP2), V1(2, IP2),

```

```

1  V1(3,IP2),I1IP,VL(1,J),VL(2,J),VL(1,J+1),VL(2,J+1),I1,I1,NP)
   IF(NP.EQ.1) GOTO 3
   GOTO 1ET1
4  IVF=1
   IPP=I1
   ILC=I2
   J=J+2
   ASSIGN 5 TO IET1
   GOTO 3
5  IF(I1+2.EQ.0) THEN
6  VL(1,J)=VL(1,J+1)
   VL(2,J)=VL(2,J+1)
   J=J+1
   GOTO 3
   ENDIF
   IF(L1.NE.0)
   IF (L1.NE.ILC) GOTO 7
   GOTO 8
7  VL(1,J+2)=VL(1,J+1)
   VL(2,J+2)=VL(2,J+1)
   VL(1,J+1)=VL(1,J)
   VL(2,J+1)=VL(2,J)
   VL(2,J)=YCOLT(MOD(ILC,4)+1)
   VL(1,J)=XCOLT(MOD(ILC,4)+1)
   ILC=ILC+1
   J=J+1
   IF(MOD(ILC-1,4)+1.NE.L1) GOTO 7
   IF(L2.NE.0) ILC=I2
   J=J+2
   GOTO 3
   ENDIF
   ILC=I2
   GOTO 6
3  CONTINUE
   IF(J.EQ.1) RETURN
   IF(IPP.EQ.0) GOTO 9
   IF(ILC.EQ.IPP) GOTO 10
11  VL(1,J)=XCOLT(MOD(ILC,4)+1)
   VL(2,J)=YCOLT(MOD(ILC,4)+1)
   ILC=ILC+1
   J=J+1
   IF(MOD(ILC-1,4)+1.NE.IPP) GOTO 11
10  VL(1,J)=VL(1,1)
   VL(2,J)=VL(2,1)
   J=J+1
9  CONTINUE
   NRPF=J-1
C s-a modificat conturul curent al fetei in functie de clipping-ul segmentelor
C NRPF contine numarul de puncte ale conturului
C se calculeaza valorile extreme ale fetei
   D1=DIST*C+D
   DO 13 I=1,NRPF 1
   IF(VL(1,I).LT.XMIN)XMIN=VL(1,I)
   IF(VL(2,I).LT.YMIN)YMIN=VL(2,I)
   IF(VL(1,I).GT.XMAX)XMAX=VL(1,I)
   IF(VL(2,I).GT.YMAX)YMAX=VL(2,I)
   IF(I1P.#0.0) GOTO 14
   D01=SGN1(VL(1,I)*VL(1,I)+VL(2,I)*VL(2,I)+DIST*DIST)
   D2=A*VL(1,I)+B*VL(2,I)+D
   Z=D01*D1/(D1-D2)
   GOTO 15
14  Z=(4D+A*VL(1,I)+B*VL(2,I))/C
15  IF(Z.LT.ZMIN)ZMIN=Z
   IF(Z.GT.ZMAX)ZMAX=Z
13  CONTINUE
C TYPE 1202,NRPF,IVF,A,B,C,D,((VL(I,J),I=1,2),J=1,NRPF)
C1202 FORMAT(2I5,4F8.2,/,2F10.3)
C TYPE 1203,XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
C1203 FORMAT(&F10.3)
RETURN
END

```

```

*****
Proceduri pentru maparea memoriei din ZONA2
*****

SUBROUTINE MAPS(NRFAG)
INTEGER IWDB(11), DSW
COMMON /PAG1/IWDB
IWDB(5)=128*NRFAG
CALL MAP(IWDB, DSW)
IF(DSW.EQ.1) RETURN
STOP
END

SUBROUTINE UMAP
INTEGER IWDB(11), DSW
COMMON /PAG1/IWDB
CALL UNMAP(IWDB, DSW)
IF(DSW.EQ.1) RETURN
STOP
END

SUBROUTINE PAG(IND, IPAG, IDEP)
IPAG=0
IDEP=IND-2048*IPAG
RETURN
END

*****
PROCEDURA DE COMPLETARE A MATRICII MAT PENTRU CAZUL IN CARE
CORPUL INDICE M ACOPEREA CORPUL INDICE N
*****

SUBROUTINE UMLE (N, M)
*****

PARAMETRI DE APEL :
N      - indicele corpului acoperit
M      - indicele corpului acoperitor
*****

DIMENSION MASK(16)
COMMON /ZONA1/XMIN(256), XMAX(256), YMIN(256), YMAX(256), ZMIN(256),
      ZMAX(256), LEG(256), IREF(256), NRF(256), IGRI(256)
      /ZONA3/MAT(256,16)
      /MASTI/MASK
DATA MASK/1,2,4,8,16,32,64,128,256,512,1024,2048,4096,8192,
      16384,32768/X/
1  JO=(M-1)/16+1
1  MO=MUD(M-1,16)+1
  MAT(N,JO)=MAT(N,JO).OR.MASK(MO)
  IGRI(M)=IGRI/M+1
  RETURN
  END

*****
PROCEDURA - TEST - DE TESTARE A VALORII UNUI BIT DIN
MATRICEA MAT
*****

SUBROUTINE TEST(N,M,ITEST)
*****

PARAMETRII DE APEL :
N, M - indicii elementului de matrice
ITEST - valoarea elementului
*****

COMMON /ZONA3/MAT(256,16)
1 /MASTI/MASK(16)
  JO=(M-1)/16+1
  MO=MUD(M-1,16)+1
  ITEST=MASK(MO).AND.MAT(N,JO)
  IF(ITEST.EQ.MASK(MO)) THEN
    ITEST=1
  ELSE
    ITEST=0
  ENDIF
  RETURN
  END

```

```

C      PROCEDURA PENTRU REALIZAREA CLIPPING-ULUI SPATIAL
C      PENTRU UN SEGMENT DEFINIT PRIN COORDONATELE CAPETELOR
C      SEGMENTULUI.
C      CLIPPING-UL SE REALIZEAZA FATA DE PIRAMIDA DE VEDERE
C      CU VIRFUL IN PUNCTUL DE VEDERE PENTRU PROIECTIA CENTRALA
C      SAU FATA DE DE SUPRAFATA PRISMATICA IN CAZUL PROIECTIEI
C      PARALELE.
C      PARAMETRI DE APEL :
C      X1,Y1,Z1,X2,Y2,Z2 - COORDONATELE CAPETELOR SEGMENTULUI FATA
C      DE TRIUNghiUL ATASAT PLANULUI DE PROIECTIE
C      I - TIPUL DE PROIECTIE (0=PARALELA)
C      PARAMETRI REINTORSI DE PROCEDURA :
C      NP - INDICATOR DE LIPSA A PROIECTIEI
C      1=NU EXISTA PROIECTIE 0=EXISTA PROIECTIE
C      XP1,YP1,XP2,YP2 - COORDONATELE (DACA EXISTA) IN PLANUL DE
C      PROIECTIE.
C      ZP1,ZP2 - "ADINCIMEA" PUNCTELOR REALE PROIECTATE
C      MASURATA FATA DE PUNCTUL DE VEDERE
C      CK1,CK2 - COEFICIENTI PE SEGMENTUL INICIAL CARE
C      DEFINESC POZITIA PUNCTELOR REALE DIN SPATIUL
C      CARE AU FOST PROIECTATE [ FATA DE (X1,Y1,Z1) ]
C
C      SUBROUTINE CLIPSI(X1,Y1,Z1,X2,Y2,Z2,I,
*      NP,XP1,YP1,XP2,YP2,ZP1,ZP2,CK1,CK2)
C      DIMENSION XI(4),YI(4),ZI(4),XK(4)
C      COMMON /SPD/XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,D,DX,DY
*      /CL1/XT1,YT1,ZT1,XI,YI,ZI,XK,COEF,DEL X,DEL Y,DEL Z,IT,IAPEL
C      IT=I
C      DO 1000 IND=1,4
1000  XK(IND)=0.
C      ZP1=D-Z1
C      ZP2=D-Z2
C      NP=1.
C      IF(IT.EQ.0) GOTO 1
C      IF(Z1.GE.D.AND.Z2.GE.D) GOTO 99999
C      ZP1=SQRT(X1*X1+Y1*Y1+(D-Z1)*(D-Z1))
C      ZP2=SQRT(X2*X2+Y2*Y2+(D-Z2)*(D-Z2))
C      TRANSFORMARE DE SCARA
1  CALL PRO(X1,Y1,Z1,XP1,YP1,IT)
  CALL PRO(X2,Y2,Z2,XP2,YP2,IT)
  XT1=X1/DX
  XT2=X2/DX
  YT1=Y1/DY
  YT2=Y2/DY
  ZT1=Z1/D
  ZT2=Z2/D
C      CALCUL DE EXTERIORITATE
C      COEF=ZT1-1.
  IF(IT.EQ.0) COEF=-1.
  I1=1
  IF(IT.EQ.0) GOTO 2
  IF(ZT1.GE.1.) GOTO 3
2  IF(XT1+COEF.GT.0.) GOTO 3
  IF(YT1+COEF.GT.0.) GOTO 3
  IF(-XT1+COEF.GT.0.) GOTO 3
  IF(-YT1+COEF.GT.0.) GOTO 3
  I1=0
  I2=2
3  COEF=ZT2-1.
  IF(IT.EQ.0) COEF=-1.
  IF(IT.EQ.0) GOTO 4
  IF(ZT2.GE.1.) GOTO 5
4  IF(XT2+COEF.GT.0.) GOTO 5
  IF(YT2+COEF.GT.0.) GOTO 5
  IF(-XT2+COEF.GT.0.) GOTO 5
  IF(-YT2+COEF.GT.0.) GOTO 5
  I2=0
C      CALCULUL DE INTERSECTII CU PLANELE SPATIULUI DE VEDERE
5  IF(I1+I2.EQ.0) GOTO 26
  NINT=1
  IF(I1+I2.EQ.3) NINT=2
  COEF=1.-ZT1
  DELX=XT2-XT1
  DELY=YT2-YT1
  DELZ=ZT2-ZT1
  IF(IT.NE.0) GOTO 6
  COEF=1.
  DELZ=0.
  IAPEL=0
6  CALL INTPL(XT1,DELY,IBUN)
  IF( IBUN.EQ.0) GOTO 7
  NINT=NINT-1

```

```

7      IF(NINT.EQ.0) GOTO 10
      CALL INTPL(YT1,DELY,IBUN)
      IF(IBUN.EQ.0)GOTO 8
      NINT=NINT-1
8      IF(NINT.EQ.0)GOTO 10
      CALL INTPL(-XT1,-DEL X,IBUN)
      IF(IBUN.EQ.0)GOTO 9
      NINT=NINT-1
9      IF(NINT.EQ.0)GOTO 10
      CALL INTPL(-YT1,-DELY,IBUN)
      IF(IBUN.EQ.0)GOTO 10
      NINT=NINT-1
10     IF(NINT.NE.0)GOTO 99999
      NINT=I1+I2
      ASSIGN 26 TO IET1
      GOTO (11,16,21),NINT
11     CK1=XK(IAPEL)
      CALL PRO(XI(IAPEL)*DX,YI(IAPEL)*DY,ZI(IAPEL)*D,XP1,YP1,IT)
      ZP1=D*(1.-ZI(IAPEL))
      IF(IT.NE.0)
      * ZP1=SQRT(DX*DX*XI(IAPEL)**2+DY*DY*YI(IAPEL)**2+
      * D*D*(1.-ZI(IAPEL))**2)
      GOTO (12,13,14,15),IAPEL
12     XP1=1.
      GOTO IET1
13     YP1=1.
      GOTO IET1
14     XP1=-1.
      GOTO IET1
15     YP1=-1.
      GOTO IET1
16     CK2=XK(IAPEL)
      CALL PRO(XI(IAPEL)*DX,YI(IAPEL)*DY,ZI(IAPEL)*D,XP2,YP2,IT)
      ZP2=D*(1.-ZI(IAPEL))
      IF(IT.NE.0)
      * ZP2=SQRT(DX*DX*XI(IAPEL)**2+DY*DY*YI(IAPEL)**2+
      * D*D*(1.-ZI(IAPEL))**2)
      GOTO (17,18,19,20),IAPEL
17     XP2=1.
      GOTO IET1
18     YP2=1.
      GOTO IET1
19     XP2=-1.
      GOTO IET1
20     YP2=-1.
      GOTO IET1
21     DO 22 IND=1,IAPEL-1
      IF(XK(IND).EQ.0) GOTO 22
      IF(XK(IND).LT.XK(IAPEL)) GOTO 23
      GOTO 20
      CONTINUE
      CK1=XK(IND)
      CK2=XK(IAPEL)
      CALL PRO(XI(IND),YI(IND),ZI(IND),XP1,YP1,IT)
      CALL PRO(XI(IAPEL),YI(IAPEL),ZI(IAPEL),XP2,YP2,IT)
      IF(II.EQ.0) THEN
      ZP1=D*(1.-ZI(IND))
      ZP2=D*(1.-ZI(IAPEL))
      ELSE
      ZP1=SQRT(DX*DX*XI(IND)**2+DY*DY*YI(IND)**2+D*D*(1.-ZI(IND))**2)
      ZP2=SQRT(DX*DX*XI(IAPEL)**2+DY*DY*YI(IAPEL)**2+
      * D*D*(1.-ZI(IAPEL))**2)
      * ENDIF
      ASSIGN 24 TO IET1
      GOTO (12,13,14,15),IND
24     ASSIGN 24 TO IET1
      GOTO (17,18,19,20),IAPEL
25     IN=IND
      IND=IAPEL
      IAPL=IM
      GOTO 23
26     CONTINUE
      NP=0
99999  RETURN
      END
C      PROCEDURA APILATA DE CLIPS. CALCULEATA INTENSIVIA DINTRE
C      SEGMENTUL NORMALIZAT SI PLANELE SPATIULUI VIZUAL
C
      SUBROUTINE INTPL(X1,DELT,IBUN)
      DIMENSION XI(4),YI(4),ZI(4),XK(4)
      COMMON /CL1/XT1,YT1,ZI1,XI,YI,ZI,XK,COEF,DEL X,DELY,DEL Z,IT,IAPEL
      IBUN=0
      IAPEL=IAPEL+1
      IF(DEL Z+DEL T.EQ.0)GOTO 9999
      T=(DEL Z-X1)/(DEL Z+DEL T)

```



```

IF (T.EQ.0..OR.T.GE.1.)GOTO 9999
IF (IAPEL.EQ.1)GOTO 1
IF (T.EQ.XK(IAPEL-1))GOTO 9999
IF (IAPEL.EQ.4.AND.T.EQ.XK(1))GOTO 9999
1 ZINT=Z(1)+T*DELZ
IF (IT.EQ.0)GOTO 2
IF (ZINT.GE.1.)GOTO 9999
2 XI(IAPEL)=XI(1)+T*DELX
YI(IAPEL)=Y(1)+T*DELY
IF (IT.EQ.0)GOTO (7,8,7,8),IAPEL
GOTO (3,4,5,6),IAPEL
3 IF (XI(IAPEL).GE.ABS(YI(IAPEL)))GOTO 9
GOTO 9999
4 IF (YI(IAPEL).GE.ABS(XI(IAPEL)))GOTO 9
GOTO 9999
5 IF (-XI(IAPEL).GE.ABS(YI(IAPEL)))GOTO 9
GOTO 9999
6 IF (-YI(IAPEL).GE.ABS(XI(IAPEL)))GOTO 9
GOTO 9999
7 IF (ABS(YI(IAPEL)).LE.1.)GOTO 9
GOTO 9999
8 IF (ABS(XI(IAPEL)).LE.1.)GOTO 9
GOTO 9999
9 ZI(IAPEL)=ZINT
XK(IAPEL)=T
IBLN=1
9999 RETURN
END

```

### 7.9.6. Concluzii finale

Algoritmul APIS prezintă câteva particularități și elemente noi care îl fac să fie un exemplu interesant în domeniul considerat.

Se pot enumera în acest sens:

- 1 — o procedură generală de clipping foarte puternică.
- 2 — utilizarea eficientă a proprietăților particulare ale poliedrelor și poligoanelor convexe.
- 3 — utilizarea memoriei virtuale pentru mărirea spațiului de lucru.
- 4 — utilizarea unor structuri de date rapide și eficiente pentru algoritm (lista  $X_{min}$  referințe înlănțuite)
- 5 — o metodă originală de trasare a corpurilor.

Pentru versiunile ulterioare sînt posibile numeroase îmbunătățiri, care sînt lăsate și în seama celor ce doresc să ducă mai departe acest algoritm. Dintre aceste îmbunătățiri cele mai importante ar putea fi

1. Rezolvarea acoperirilor ciclice
2. Introducerea de poliedre concave, cu găuri, etc.
3. Generalizarea virtualizării memoriei pentru toate structurile de date.
4. Utilizarea unor echipamente de afișaj cu calitate superioare (display-uri color, proceduri hard de umplere a conturilor, etc.)

7.9.7. Aplicații. Alte aplicații ale algoritmului APIS sînt ilustrate în figurile 7.29, . . . 7.34.

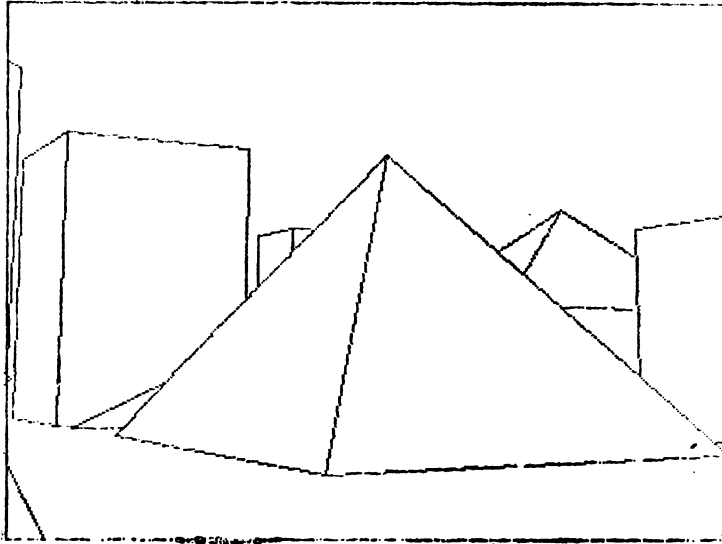


Fig. 7.29

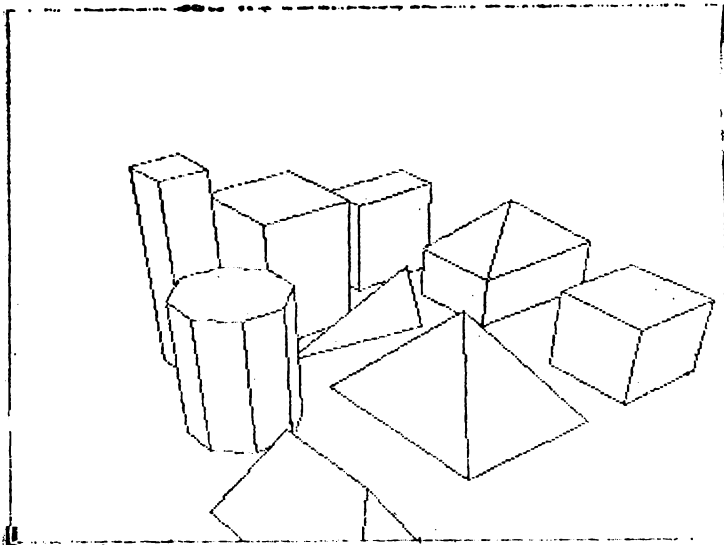


Fig. 7.30

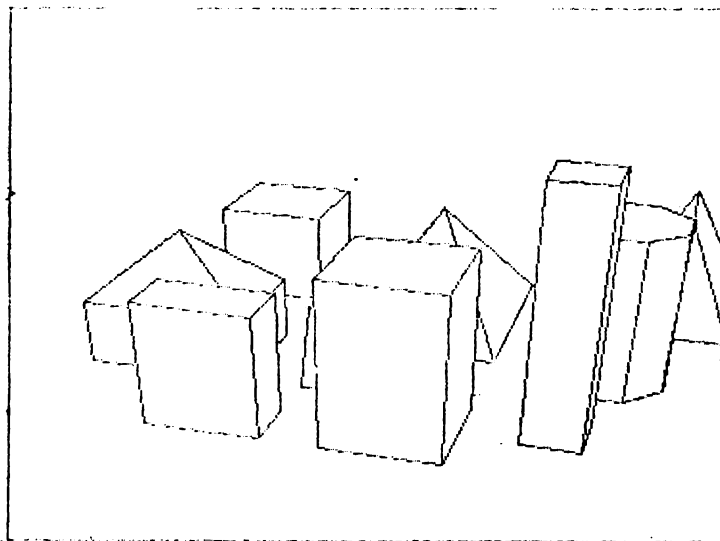


Fig. 7.31

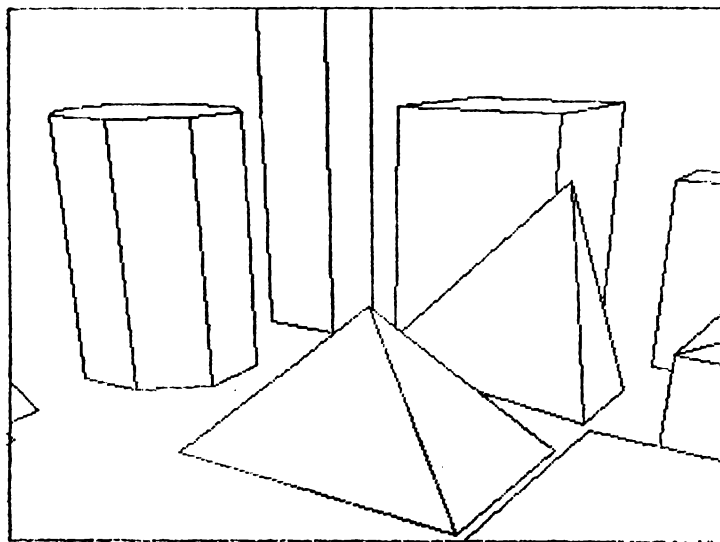


Fig. 7.32

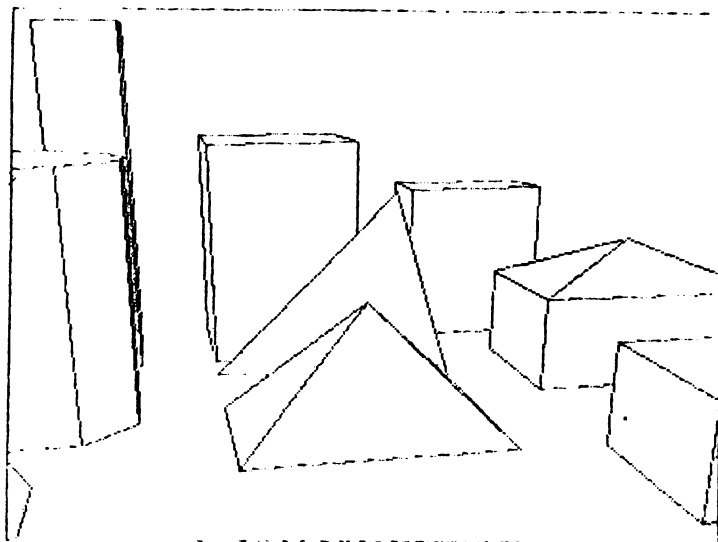


Fig. 7.33

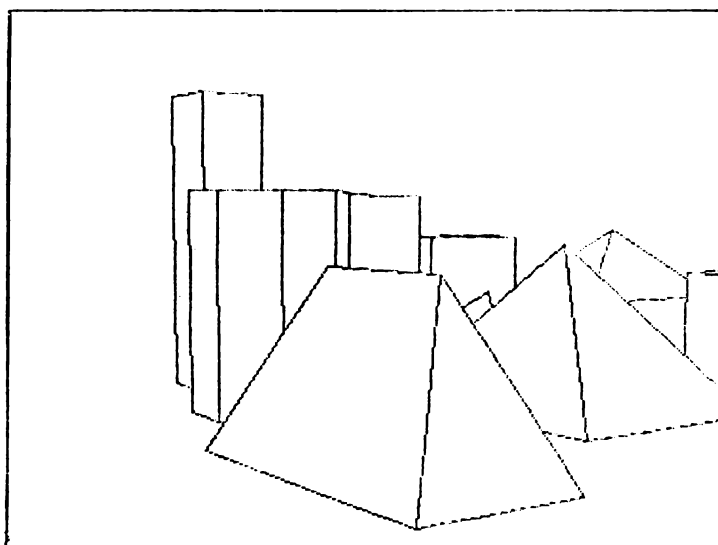


Fig. 7.34

# Reprezentarea spațială grafică și vizuală a curbelor și suprafețelor

## 8.1. Vizualizarea funcțiilor de două variabile

În foarte multe discipline, specialiștii utilizează ecuațiile funcțiilor cu două variabile, fără a-și putea imagina alura suprafețelor corespunzătoare. Totuși, specialiștii pot deduce indicii importante privind și interpretând anumite proprietăți fizico-chimice. Așa se explică faptul că au fost imaginate mai multe metode care permit o vizualizare în spațiul cu trei dimensiuni a suprafețelor reprezentând diferite valori ale funcției atunci când sînt făcute să varieze cele două variabile în interiorul unui domeniu dat. Metodele propuse în acest scop nu trebuie să fie greoaie și nu trebuie să conducă la calcule foarte lungi.

În cele ce urmează vom studia doi algoritmi care sînt cel mai mult utilizați la ora actuală, adică cel al lui *Williamson* și cel al lui *Wright*.

### 8.1.1. Algoritmul lui Williamson

Pornind de la infinitatea de puncte care constituie suprafața reprezentînd funcția de două variabile  $z = f(x, y)$ , se creează un subansamblu permițînd vizualizarea acesteia. Se constituie un tabel de valori secționînd suprafața prin niște plane paralele cu planul  $y$  o  $z$ , adică planele  $x = \text{constant}$ . Curbele astfel obținute sînt reportate într-un plan după ce le-am făcut să aibă un decalaj, în scopul simulării profunzimii (adîncimii). Acest artificiu permite obținerea unei impresii pregnante de adîncime (profunzime), în timp ce se utilizează o simplă imagine de perspectivă cavalieră frontală. Principiul este următorul:

— se începe prin afișarea curbei care este cea mai apropiată de observator. Această curbă servește la inițializarea a ceea ce vom numi funcția vizuală maximă sau linia de creastă;

— se consideră apoi, curba care se găsește imediat în spatele primei, în raport cu observatorul. Se va compara fiecare punct de pe noua curbă cu valorile funcției vizuale maxime. De fiecare dată cînd pentru un  $x$  dat se găsește un  $y$  inferior lui  $y$  corespunzător de pe linia de creastă, aceasta înseamnă că punctul este ascuns. De fiecare dată cînd se va găsi un punct astfel ca  $y$  să fie superior lui  $y$  corespunzător de pe linia de creastă, aceasta va însemna că trebuie afișat punctul aflat în discuție și actualizată (evidențiată) funcția vizuală maximă din acest punct;

— se continuă procesul de comparare, apoi de afișare și de evidențiere (actualizare), pînă cînd toate planele de secțiune au fost explorate. Ultima curbă afișată va fi cea mai îndepărtată în raport cu observatorul.

Pentru a realiza acest algoritm, se folosește un tabel care cuprinde toate coordonatele punctelor formînd linia de creastă. Pentru fiecare punct al curbei de afișat, se caută dacă există un  $x$  corespunzător în tabelul „*linia de creastă*”. Acesta este triat în ordinea crescătoare a  $x$ -urilor pentru a accelera cercetarea. Dacă nu se găsește un  $x$  corespunzător punctului dorit, se vor găsi, totuși, două puncte  $x_i$  și  $x_{i+1}$  ale liniei de creastă astfel că  $x_i \leq x \leq x_{i+1}$ . Se calculează, apoi, prin interpolare liniară între  $j_i$  și  $j_{i+1}$  valoarea funcției vizuale maxime în punctul  $x$ . Dacă această valoare este mai mare decît  $y$ -ul de intrare, nu se face nimic (punctul este ascuns), dacă nu, se actualizează lista punctelor care formează linia de creastă prin inserția punctului  $(x, y)$ . Notăm în trecere că, în general este necesar să se mențină în paralel o funcție vizuală minimă, care permite reprezentarea și dedesubtului suprafeței. Principiul este exact același ca pentru funcția vizuală maximă. Putem face următoarele precizări:

— acest algoritm duce calculul pînă la precizia mașinii. Aceasta înseamnă că în lista „*linia de creastă*” pot figura numeroase puncte care vor fi confundate la afișaj, datorită rezoluției limitate a ecranului;

— Williamson cere utilizatorului să evalueze mărimea tabelelor care conțin funcția vizuală maximă și funcția vizuală minimă, ceea ce este destul de dificil;

— operațiunile de cercetare a unui element și de inserție sînt costisitoare, mai ales cînd numărul de puncte crește (pot fi mai mult de o mie!) chiar dacă folosirea unei metode de cercetare dicotomică permite ca aceasta să fie relativ rapidă, inserția va fi costisitoare în orice caz.

Metoda lui Williamson a reprezentat un progres foarte mare, în raport cu primele metode apărute. Ea este, totuși, relativ mai dificil adaptată la folosirea unui terminal grafic, deoarece nu ține cont de caracteristicile sale.

### 8.1.2. Algoritmul lui Wright

Dacă punctul de plecare este comun și anume secționarea suprafețelor prin planele  $x = \text{const.}$  în folosirea liniilor de creastă maxime și minime, în schimb apar două diferențe importante:

— folosirea unei proiecții perspective pentru prezentarea în spațiu permițînd, în special, o tratare simplă a curbelor cu  $y = \text{const.}$  pe aceeași figură;

— folosirea slăbiciunii rezoluției ecranului, pentru a nu considera decît ceea ce se petrece în punctele de coordonate întregi. Funcția vizuală este conținută atunci într-un tabel de mărime fixă, cuprinzînd tot atîtea elemente cîte puncte sînt pe o linie a ecranului (512 sau 1024 pentru cazurile curente). Cele două tabele „*linie de creastă*” conțin în fiecare element ordonata punctului celui mai înalt din toate punctele de pe aceeași abscisă. Problema cercetării (căutării) unui element este, deci, evitată (ne mulțurim cu o indexare) și nu există niciodată vreo inserție în listă. Singurele operații consistă, eventual în actualizarea funcției vizuale între două puncte  $x_{i+1}, \dots, x_{j-1}$ . Algoritmul este atunci suficient de performant pentru ca să se aibă în vedere prezentarea în același timp a curbelor cu  $x = \text{const.}$  și a curbelor cu  $y = \text{const.}$  Singura

problemă ar putea fi aceea că nu ne putem mulțumi să suprapunem traseul cu  $y = \text{const.}$  cu traseul curbelor cu  $x = \text{const.}$  Atunci trebuie să se opereze în felul următor:

- se trasează a  $j^{\text{a}}$  curbă cu  $x = \text{Const.}$
- se trasează apoi în fiecare punct porțiunea de curbă cu  $y = \text{Const.}$ , cuprinsă între a  $j^{\text{a}}$  și a  $(j + 1)^{\text{a}}$  curbă cu  $x = \text{Const.}$ , actualizând liniile de creastă;
- se trasează apoi a  $(j + 1)^{\text{a}}$  curbă cu  $x = \text{Const.}$

Un studiu comparativ al timpilor de execuție al celor doi algoritmi a arătat că dacă cel al lui Williamson era puțin mai rapid cînd nu existau prea multe plane de secțiune (pînă la 20), cel al lui Wright devenea în mod rapid cu mult mai bun, deși nu prezintă rezultate mai complete. Trebuie notat, de asemenea, faptul că a secționa o suprafață doar prin douăzeci de plane, presupune că suprafața este relativ regulată și că, în general, se folosește un număr mai mare de plane de secțiune (între 30 și 50).

În concluzie putem spune că a pune la dispoziția utilizatorilor a unor instrumente interactive care permit vizualizarea unei funcții de două variabile și care permit modificarea valorilor parametrilor, ca și a punctelor de vedere, se dovedește a fi un efort de programare relativ puțin costisitor, dar, care aduce un serviciu apreciabil.

În cele ce urmează vom folosi pentru vizualizarea suprafețelor, într-un fel, algoritmul Williamson.

## 8.2. Algoritm și program pentru vizualizarea sau reprezentarea grafică a curbelor și suprafețelor în perspectivă, cu studiul porțiunilor ascunse — procedura HIDE

### 8.2.1. Procedura HIDE. Descrierea generală

Procedura HIDE este oarecum o implementare a algoritmului „**Hidden-Line Plotting Program**“: prezentat de *Hugh Williamson* (Communication of the ACM 1970).

Procedura produce o reprezentare bidimensională a unei suprafețe sau figuri, prin plotarea segmentelor unei succesiuni de curbe. Fiecare curbă este plotată pe porțiunile în care nu este ascunsă de nici o curbă plotată anterior.

Procedura permite următoarele opțiuni:

- Traducerea șirurilor înainte de plotare pentru simularea pașilor în adîncime.
- Trasarea părții de dedesubt a suprafeței. În acest caz liniile sînt considerate vizibile acolo unde cad sub orice curbă plotată anterior. Această opțiune împreună cu posibilitatea programului de a plota maximul vizual asigură plotarea părții vizibilă a întregii suprafețe.
- Alegerea unghiului reprezintă valoarea în radiani a unghiului dintre axa ce simulează adîncimea și axa  $x$  (fig. 8.1).
- Alegerea scărilor de reprezentare pentru axa  $X$  și  $Y$ .

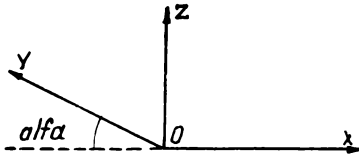


Fig. 8.1

În algoritm nu sînt incluse în această fază secvențe pentru realizarea reprezentărilor în perspectivă sau a rotațiilor.

Totuși, dacă șirurile ce trebuie plotate sînt transformate corespunzător înainte de apelul procedurii **HIDE**, pot fi obținute rezultatele dorite. Acest fapt este, credem, util să fie lăsat, însă, în seama cititorului.

### 8.2.2. Descrierea algoritmului

Ideea care stă la baza algoritmului este o secvență de calcul pentru obținerea maximumului vizual a două curbe.

Fie  $XG, G$  și  $X, Y$  vectorii corespunzători absciselor și ordonatelor celor 2 linii frînte obținute prin discretizarea a 2 curbe (fig. 8.2 a).

Una din curbe este maximumul vizual curent, iar cealaltă reprezintă curba următoare în secvență.

Ceea ce interesează este ceea ce introduce nou față de maximumul vizual curent ( $XG, G$ ), curba discretizată ( $X, Y$ ) (figura 8.2 b).

După cum se poate observa, numărul de puncte conținute în vectorii ( $XG, G$ ) după ce  $N$  curbe au fost plotate este o sumă dintre:

- numărul inițial de puncte al oricăreia dintre cele  $N$  curbe plotate
- numărul de puncte de intersecție ale diferitelor curbe care fac parte din ( $XG, G$ ) de la pasul  $N-1$
- numărul de puncte necesare pentru simularea discontinuităților în curba de maxim.

Punctele de tipul  $c$  apar în cazurile prezentate în continuare.

Curba de maxim conține un salt, și cum abscisele a 2 puncte consecutive trebuie să fie strict crescătoare, se introduce în curba de maxim un nou punct avînd abscisa ( $X-EPS$ ) (fig. 8.3 a și fig. 8.3 b).

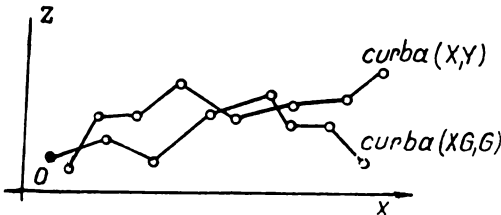


Fig. 8.2 a



Fig. 8.2 b

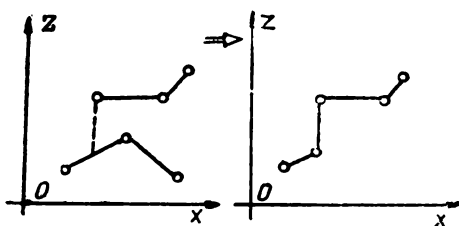


Fig. 8.3 a

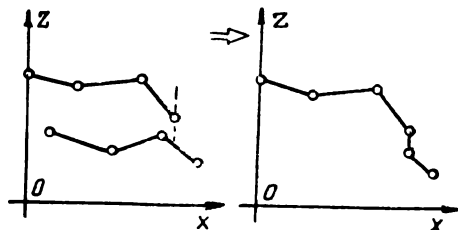


Fig. 8.3 b



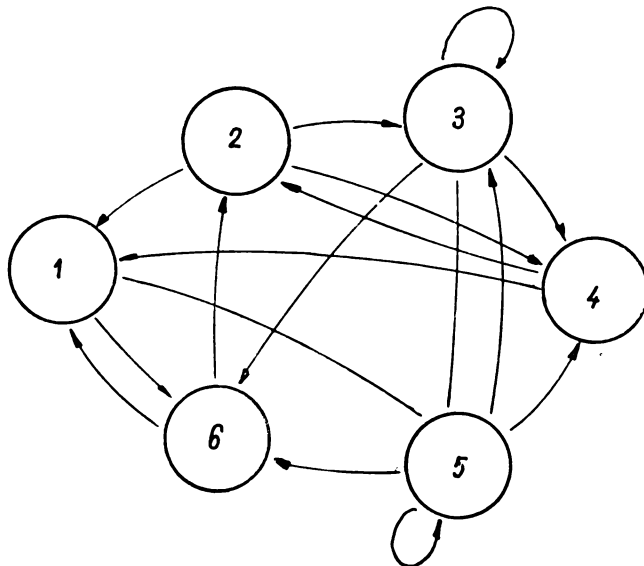


Fig. 8.4

Cum saltul trebuie să apară în grafic ca o linie verticală valoarea lui  $EPS$  trebuie luată astfel încît pentru plotter  $X-EPS = X$  deci  $EPS < precizia\ plotterului$  (în program  $EPS = 10^{-5}$ ).

Problemele care trebuie rezolvate în cadrul implementării sînt următoarele:

1. Calculul unei noi curbe de maxim
2. Plotarea porțiunilor de curbă ce alcătuiesc maximul curent dar nu au fost plotate anterior

Schema algoritmului este aceea a unui automat cu 6 stări care sînt prezentate în schema alăturată, diagrama de stări următoare fiind cea din figura 8.4.

În fiecare din stări se execută cîteva operații, pe care le vom prezenta în cele ce urmează.

Cele 6 stări au fost considerate cu privire la parcurgerea curbei a  $J$ -a din cele  $n$  considerate, iar situarea în una din cele 6 stări se face pe baza următoarelor considerente:

STAREA 1	$XG(I) < X(J)$ ;	$XG(I + 1) < X(J + 1)$
STAREA 2	$XG(I) < X(J)$ ;	$XG(I + 1) \geq X(J + 1)$
STAREA 3	$XG(I) = X(J)$ ;	$XG(I + 1) \leq X(J + 1)$
STAREA 4	$XG(I) = X(J)$ ;	$XG(I + 1) > X(J + 1)$
STAREA 5	$XG(I) > X(J)$ ;	$XG(I + 1) \leq X(J + 1)$
STAREA 6	$XG(I) > X(J)$ ;	$XG(I + 1) > X(J + 1)$

Din schema grafică a fiecărei stări (fig. 8.5) se observă că pentru o stare oarecare există 9 poziții selective ale celor două segmente considerate, poziții care se deosebesc prin valorile variabilelor  $F_1$  și  $F_2$ .

Acțiunile care se realizează în fiecare din cazuri sînt următoarele:

1. Secvența de calcul  $F_1, F_2$ .

$F_1$  și  $F_2$  sînt valorile cu semn ale diferențelor valorilor extreme ale funcțiilor  $X$  și  $XG$  aflate în intervalul  $[X(J), X(J + 1)]$ .

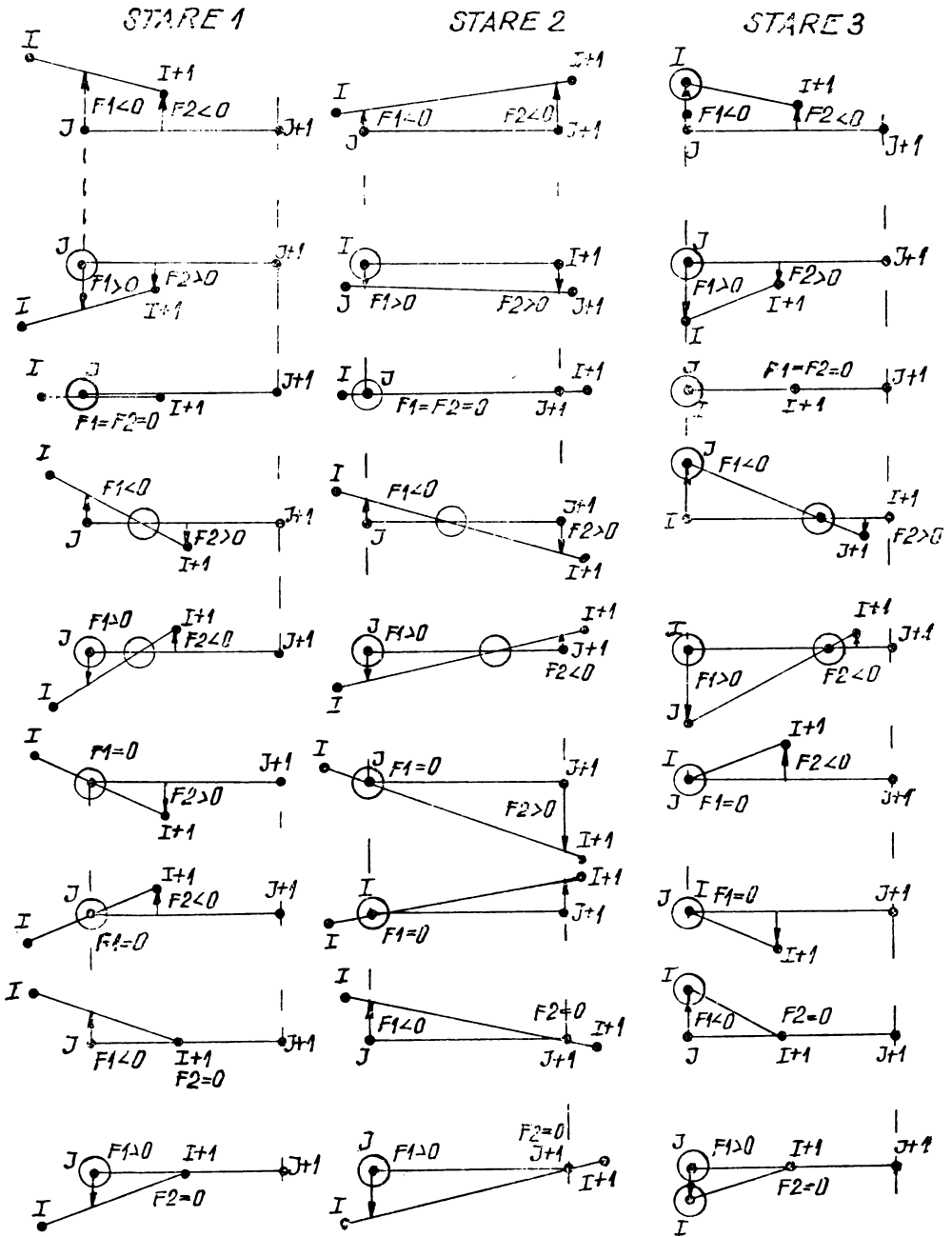


Fig. 8.5

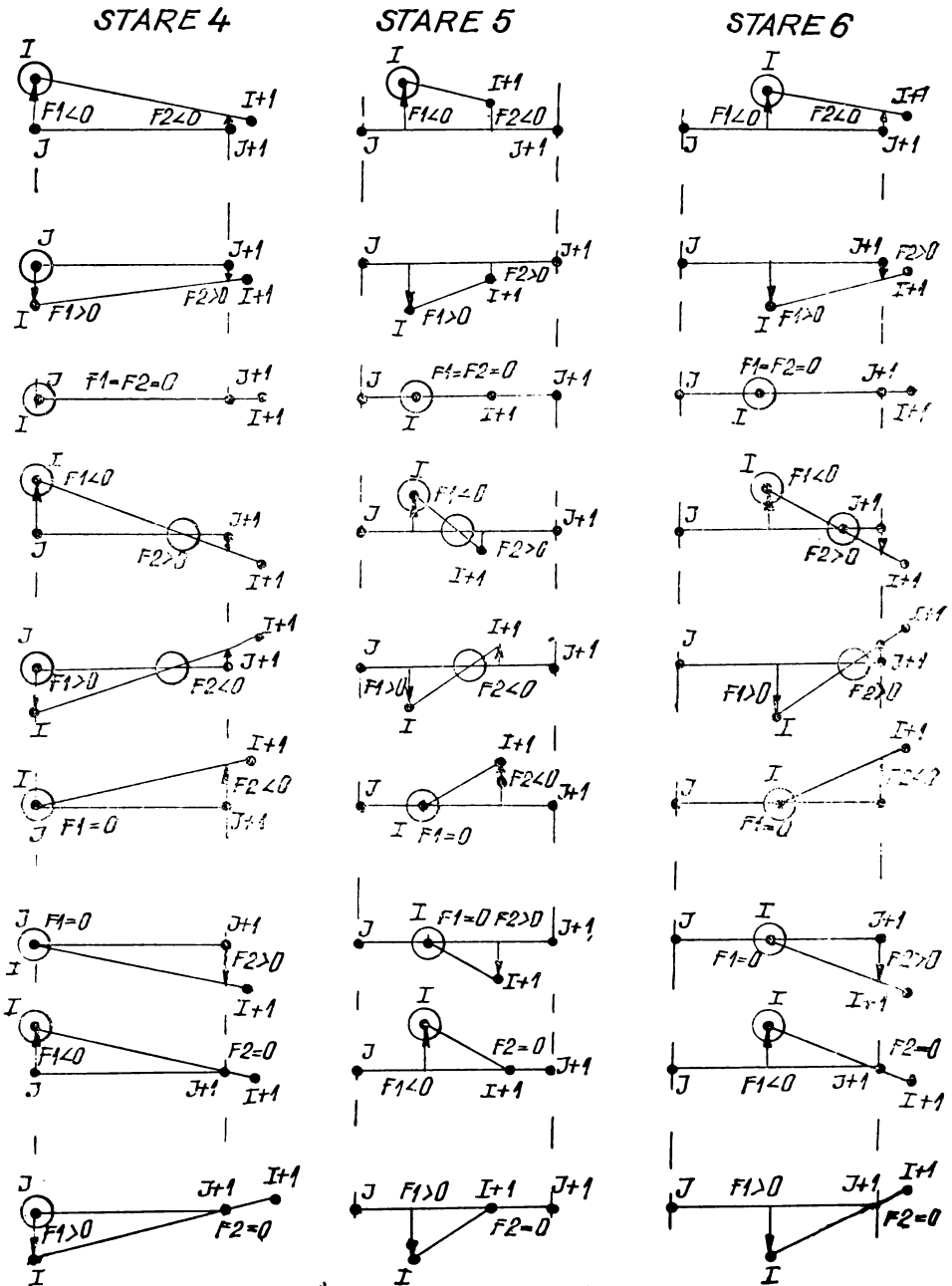


Fig. 8.5

2. Secvența de introducere în  $(XH, H)$  (curba curentă de maxim) a punctelor inițiale.  
Punctele introduse în fiecare caz sînt specificate în schema stărilor prin cerculețe negre (fig. 8.5).
8. Introducerea în  $(XH, H)$  a punctelor de intersecție dintre cele 2 segmente considerate (dacă există).
4. Secvența de plotare a unei porțiuni vizibile dacă este cazul și recalcularea variabilei *SEMN*.  
Variabila *SEMN* nu spune dacă la terminarea tratării cazului curent curba se află deasupra sau dedesubtul maximumului curent.
5. Calculul cazului următor (ieșirea) dacă s-a detectat sfîrșitul uneia dintre curbe.

### 8.2.3. Parametrii de apel ai procedurii HIDE

Apelul subrutinei **HIDE** este următorul:

**CALL HIDE (X, Y, XG, G, XH, H, NT, NG, STEPZ, ALFA, SCARAX, SCARAY, MAXDIM)**

Semnificația parametrilor de apel este următoarea:

*X* — vectorul ordonat strict crescător al absciselor; *Y* — vectorul ordonatelor corespunzătoare lui *X*; *XG* — vectorul de abscise ale punctelor ce mențin curba curentă de maxim; *G* — vectorul curent al ordonatelor pentru *XG*; *XH, H* — vectori de lucru; *NT* — numărul de puncte al lui *X(Y)*; *NG* — numărul de puncte pentru *XG(G)*; *STEPZ* — lungimea pasului pentru simularea adîncimii; *ALFA* — unghiul dintre axa *Z* și axa *X*; *SCARAX* — scara de reprezentare pe axa *X*; *SCARAY* — scara de reprezentare pe axa *Y*; *MAXDIM* — lungimea maximă admisă pentru vectorii *XG, G, XH, H*. La primul apel al funcției **HIDE**, *NG* avînd valoarea 0, deoarece încă nu există nici un vector de maxim, are următoarea semnificație: *NG* = 0 plotează maximumul funcției *NG* = 1 plotează minimumul funcției

Se poate, de asemenea, specifica dacă pentru un anumit apel nu se mai dorește translatarea vectorilor *X* și *Y*. În acest fel vectorii rămîn translați la valorile de la apelul anterior dînd lui *NT* valoarea — *NT*.

### 8.2.4. Program principal CURBA 1

```

DIMENSION X(100),Y(100),XG(300),G(300),XH(300),H(300),
IXG(300),IG(300)
DATA NPT,NG,SCARAX,SCARAY,PI,MAXDIM/
150,21,2.,2.,3.141592654,300/
CALL ASSIGN(2,'LP:')
NG1=0
NG2=-1
ALFA=3*PI/4.
STEPX=10./49.
STEPZ=10./20.
XIF1=-5.
C XIF1=1
YIF1=-5.
C YIF1=1
Y1=YINT
DO 2 I=1,NCH
DO 1 J=1,NPT
X(J)=XIF1+(J-1)*STEPX
1 Y(J)=(X(J)*Y1*(X(J)**2-Y1**2))/24.
WRITE(2,100)I
100 FORMAT(///'I',CURBA DE MAXIM NR: 'I2,///)
CALL HIDE(X,Y,XG,G,XH,H,NPT,IG1,STEPZ,ALFA,SCARAX,SCARAY,MAXDIM)
WRITE(2,200)I
200 FORMAT(///'I',CURBA DE MINIM NR: 'I2,///)
CALL HIDE(X,Y,XG,G),XIF1,YIF1,NG2,STEPZ,ALFA,SCARAX,SCARAY,
1*MAXDIM)
2 Y1=Y1+STEPZ
STOP
END

```

## 8.2.5. Programul CURBA 2

```

      DIMENSION X(100),Y(100),XG(300),G(300),XH(300),H(300),
      IG1(300),Y1(100),XP(30),YP(30),ZP(30,30),XCOLT(4),YCOLT(4),
      ZXG1(300),XIC(2),YIC(2)
      DATA PI,MAXDIM/3.141592654,300/
      DIST(A,B,C,X,Y)=A*X+B*Y+C
      CALL ASSIGN(3,'LP1')
      CALL ASSIGN(1,'ITPLHV.DAT')
      CALL FDBSET(1,'R')
      HEAD(1,15)LX,LY
15     FORMAT(2I4)
      HEAD(1,16)(XP(I),I=1,LX)
      HEAD(1,16)(YP(I),I=1,LY)
      HEAD(1,16)((ZP(I,J),I=1,LX),J=1,LY)
16     FORMAT(11F7.3)
      CALL ASSIGN(2,'PLOT.DAT')
      CALL FDBSET(2,'N')
      CALL PLOTS(0,0,2)
17     TYPE 1000
1000    FORMAT(' 1.VEDERE GENERALA  2.SECTIONE  0.EXIT : ',S)
      ACCEPT 100,IPROP
      IF(IPROP.EQ.0)GOTO 1022
      IF(IPROP.EQ.2)GOTO 1010
      TYPE 99
99     FORMAT(' ',TIPUL PROIECTIEI [0=CENTRALA-1=FAHALELA-2=CAVALIERA
      1 OBLICA]:',S)
      ACCEPT 100,IPROP
100    FORMAT(I1)
      IF(IPROP.EQ.2.)GOTO 105
      TYPE 101
101    FORMAT(' ',DATE INITIALE [NPT,NCR,XINI,XFIN,YINI,YFIN,
      1SCARAX,SCARAY,X0,Y0,Z0] '')
      ACCEPT 102,NPT,NCR,XINI,XFIN,YINI,YFIN,SCARAX,SCARAY,X0,Y0,Z0
102    FORMAT(2I3,4F6.2,2F4.1,3F6.2)
      GOTO 5
105    TYPE 103
103    FORMAT(' ',DATE INITIALE [NPT,NCR,XINI,XFIN,YINI,YFIN,
      1SCARAX,SCARAY,ALFA1,ALFA2] '')
      ACCEPT 104,NPT,NCR,XINI,XFIN,YINI,YFIN,SCARAX,SCARAY,ALFA1,ALFA2
104    FORMAT(2I3,4F6.2,2F4.1,2F4.1)
      ALFA=PI*ALFA1/ALFA2
5     NG1=0
      NG2=-1
      STEPX=ABS(XFIN-XINI)/(NPT-1)
      STEPY=ABS(YFIN-YINI)/(NCR-1)
      Y1(1)=YINI
      IF(IPROP.EQ.2) GOTO 2
      XC=0.
      YC=0.
      ZC=0.
      IF(IPROP.GT.0)GOTO 1
      XC=(XINI+XFIN)/2.
      YC=(YINI+YFIN)/2.
      CALL ITPLEV(3,LX,LY,XP,YP,ZP,1,XC,YC,ZC)
1     CALL DATEEN(X0,Y0,Z0,XC,YC,ZC,A,B,C,P,F1,G0,H0,0,R)
      IF(X.LT.0.)GOTO 2
      XINI=XFIN

```

```

STEPX=-STEPX
Y1(1)=YFIN
STEPZ=-STEPZ
2  DO 3 I=1,NCR
   DO * J=1,NPT
   X(J)=XINI+(J-1)*STEPX
4  Y1(J)=Y1(1)
   CALL ITPLBY(3,LX,LY,XP,YP,ZP,NPT,X,Y1,J)
   IF(IPROP.EQ.2) GOTO 6
   CALL PROIEC(X,Y1,Y,NPT,A,B,C,P,F1,GG,HH,U,R,IPROP,X0,Y0,Z0)
   WHITE(4,222)(X(K1),K1=1,NPT)
222  FORMAT(' ',12F10.3)
6  IF(IPROP.NE.2) NPT=-NPT
   CALL HIDE(X,Y,XG,G,XH,H,NPT,NG1,STEPZ,ALFA,SCARAX,
1SCARAY,MAXDIM)
   CALL HIDE(X,Y,XG1,G1,XH1,H1,-NPT,NG2,STEPZ,ALFA,SCARAX,SCARAY,
1MAXDIM)
J  Y1(1)=Y1(1)+STEPZ
   GOTO 17
1010  TYPE=101
   ACCEPT 102,NPT,NCR,XCOLT(1),XCOLT(J),YCOLT(1),YCOLT(2),
1 SCARAX,SCARAY,X0,Y0,Z0
   TYPE=1011
1011  FORMAT(' DEFINITI SECTIUNEA PRIN DOUA PUNCTE (XD1,YD1,XD2,YD2)')
   ACCEPT 16,XD1,YD1,XD2,YD2
   DIAG=SQRT((XCOLT(3)-XCOLT(1))**2+(YCOLT(2)-YCOLT(1))**2)
   XCOLT(2)=XCOLT(1)
   XCOLT(4)=XCOLT(3)
   YCOLT(3)=YCOLT(2)
   YCOLT(4)=YCOLT(1)
   XC=(XCOLT(1)+XCOLT(3))/2.
   YC=(YCOLT(1)+YCOLT(2))/2.
   CALL ITPLBY(3,LX,LY,XP,YP,ZP,1,XC,YC,ZC)
   CALL DATEIN(X0,Y0,Z0,XC,YC,ZC,A,B,C,P,F1,GG,HH,U,H)
   AD=YD2-YD1
   BD=XD1-XD2
   CD=-YD1*BD-XD1*AD
   DMAX=0.
   K=0
   DP=DIST(AD,BD,CD,X0,Y0)
   DO 1012 I=1,4
   DX=DIST(AD,BD,CD,XCOLT(I),YCOLT(I))
   IF(DP*DX.GT.0.)GOTO 1012
   IF(ABS(DX).LE.ABS(DMAX))GOTO 1012
   DMAX=DX
   K=I
1012  CONTINUE
   IF(DMAX.NE.0.) GOTO 1023
   TYPE=1024
1024  FORMAT(' SECTIUNE INCORECT ALEASA')
   GOTO 17
1023  ST=DMAX/(NCR-1)
   NG1=0
   NG2=-1
L  TYPE=2000,ST,DMAX,CD
02000  FORMAT(' ',3F8.2/)

```

```

DO 1013 I=1,NCK
K=0
CDK=CD-ST*(I-1)
IF (BD.EQ.0.)GOTO 1014
YI=(-CDK-AD*XCOLT(1))/BD
IF (YI.LT.YCOLT(1).OR.YI.GT.YCOLT(2))GOTO 1015
K=K+1
XIC(K)=XCOLT(1)
YIC(K)=YI
1015 YI=(-CDK-AD*XCOLT(3))/BD
IF (YI.LT.YCOLT(1).OR.YI.GT.YCOLT(2))GOTO 1014
K=K+1
XIC(K)=XCOLT(3)
YIC(K)=YI
IF (K.EQ.2)GOTO 1018
1014 IF (AD.EQ.0.)GOTO 1016
XI=(-CDK-BD*YCOLT(1))/AD
IF (XI.LT.XCOLT(1).OR.XI.GT.XCOLT(3))GOTO 1017
K=K+1
XIC(K)=XI
YIC(K)=YCOLT(1)
IF (K.EQ.2)GOTO 1018
1017 XI=(-CDK-BD*YCOLT(2))/AD
IF (XI.LT.XCOLT(1).OR.XI.GT.XCOLT(3))GOTO 1018
K=K+1
XIC(K)=XI
YIC(K)=YCOLT(2)
1018 IF (K.LT.2)GOTO 1013
NPTK=NPT*SQRT((XIC(2)-XIC(1))**2+(YIC(2)-YIC(1))**2)/DIAG
IF (NPTK.LE.1)GOTO 1013
STEPX=(XIC(2)-XIC(1))/(NPTK-1)
STEPLY=(YIC(2)-YIC(1))/(NPTK-1)
D TYPE 2001,XIC(1),YIC(1),XIC(2),YIC(2),NPTK,STEPX,STEPLY
D2001 FORMAT(' ',2F8.2,'/ ',2F8.2,'/ ',I3,2F8.2)
DO 1019 K=1,NPTK
X(K)=XIC(1)+STEPX*(K-1)
1019 Y1(K)=YIC(1)+STEPLY*(K-1)
CALL ITPLBV(3,LX,LY,XP,YP,ZP,NPTK,X,Y1,Y)
CALL PROIEC(X,Y1,Y,NPTK,A,B,C,P,F1,GG,HH,O,R,1,XU,Y0,Z0)
IF (X(2).GT.X(1))GOTO 1020
DO 1021 J=1,NPTK/2
XX=X(J)
X(J)=X(NPTK+1-J)
X(NPTK+1-J)=XX
YY=Y(J)
Y(J)=Y(NPTK+1-J)
1021 Y(NPTK+1-J)=YY
1020 NPTK=-NPTK
D TYPE 2002,(X(JJ),JJ=1,-NPTK)
D2002 FORMAT(' ',8F9.2)
CALL HIDE(X,Y,XG,G1,XH,H,-NPTK,NG2,STEPZ,ALFA,SCARAX,SCARAY,
1 MAXDIM)
CALL HIDE(X,Y,XG1,G1,XH,H,-NPTK,NG2,STEPZ,ALFA,SCARAX,SCARAY,
1 MAXDIM)
1013 CONTINUE
GOTO 17

1022 CALL CLOSE(2)
STOP
END

```

## 8.2.6. Subprogramul HIDE

,LP1:=HYDE

```

SUBROUTINE HIDE (X,Y,XG,G,XH,M,NT,NG,STEPZ,ALFA,SCARAX
1,SCARAY,MAXDIM)
DIMENSION X(1),Y(1),XG(1),G(1),XH(1),H(1)
DATA EPS/.00001/
F (XX,XI,YI,XF,YF)=YI+(XX-XI)*(YF-YI)/(XF-XI)
IF (MAXDIM.LE.0) RETURN
N.CALC=0
IF (NT.GT.0) GOTO 74
NT=-NT
N.CALC=1
74 DO 71 I=2,NT
IF (X(I-1).LT.X(I)) GOTO 74
MAXDIM=0
RETURN
71 CONTINUE
IF (NG.GT.0) GOTO 5000
SIGNUM=1
IF (NG.EQ.-1) SIGNUM=-1
XG(MAXDIM)=SIGNUM
NRC=0
XINI=X(1)
YINI=Y(1)
STEPX=STEPZ*COS(ALFA)
STEPSY=STEPZ*SIN(ALFA)
DO 200 I=1,NT
XG(I)=X(I)-XINI
200 G(I)=(Y(I)-YINI)*SIGNUM
CALL PDATA(XG,G,1,NT,SCARAX,SCARAY*SIGNUM)
NG=NT
RETURN
5000 IF (N.CALC.EQ.1) GOTO 5001
NRC=NRC+1
5001 STX=STEPX*NRC
STY=STEPSY*NRC
SIGNUM=XG(MAXDIM)
DO 201 I=1,NT
X(I)=X(I)-XINI-STX
201 Y(I)=(Y(I)-YINI+STY)*SIGNUM
C WRITE(2,72)((X(I1+J1-1),J1=1,10),(Y(I1+J2-1),J2=1,10),
C I1=1,NT,10)
C WRITE(2,72)((XG(I1+J1-1),J1=1,10),(G(I1+J2-1),J2=1,10),
C I1=1,NG,10)
C72 FORMAT(' ',10F12.4)
C
C
C CALCULAREA CAZULUI INITIAL
NH=0
ICAZ=3
I=1
J=1
IIND=1
IF (X(1)-XG(1)) 100,108,102
10 DO 107 J=1,NT
IF (X(J)-XG(1)) 103,104,105
103 NH=NH+1
XH(NH)=X(J)

```



```

107 H(NH)=Y(J)
    GOTU 112
105 J=J-1
    ICAZ=6
    IF (X(J+1).GE.XG(2)) ICAZ=5
104 SEMN=1.
    F2=F(XG(I),X(J),Y(J),X(J+1),Y(J+1))-G(I)
    IF (F2.GE.0.) GOTU 26
    NH=NH+1
    XH(NH)=XG(I)-EPS
    H(NH)=F(XH(NH),X(J),Y(J),X(J+1),Y(J+1))
    CALL PRATA(XH,H,IIND,NH,SCARAX,SCARAY*SIGNUM)
    SEMN=-1.
    GOTU 26
102 GO 101 I=1,NG
    IF (XG(I)-X(J)) 101,108,109
101 CONTINUE
    GOTU 112
109 ICAZ=1
    I=I-1
108 SEMN=-1
    F2=Y(J)-F(X(J),XG(I),G(I),XG(I+1),G(I+1))
    IF (F2.LE.0.) GOTU 26
    IIND=NH+1
    SEMN=1
26 IF (NH.LT.MAXDIM-1) GOTU 2H
112 MAXDIM=-MAXDIM
    RETURN
C   SECVENTA DE CALCUL F1,F2
2B F1=F2
    IF (X(J+1)-XG(I+1)) 1,2,3
1   F2=Y(J+1)-F(X(J+1),XG(I),G(I),XG(I+1),G(I+1))
    GOTU 4
2   F2=Y(J+1)-G(I+1)
    GOTU 4
3   F2=F(XG(I+1),X(J),Y(J),X(J+1),Y(J+1))-G(I+1)
C
C   SECVENTA DE INTRODUCERE PUNCTE INITIALE
C
C
4   GOTU (8,8,5,5,6,6),ICAZ
5   IF (F1.GE.0)GOTO 9
    GOTU 7
6   IF (F1.GT.0)GOTO 10
7   NH=NH+1
    XH(NH)=XG(I)
    H(NH)=G(I)
    GOTU 10
8   IF (F1.LT.0) GOTU 10
9   NH=NH+1
    XH(NH)=X(J)
    H(NH)=Y(J)
C
C
C   PUNEY IN XH PUNCTELE DE INTERSECTIE DACA EXISTA
C

```

```

10  IF(F1*F2.GE.0) GOTO 16
    GOTO (11,14,12,14,12,13),IC-Z
11  P=XG(I+1)-X(J)
    R=X(J)
    GOTO 15
12  P=XG(I+1)-XG(I)
    R=XG(I)
    GOTO 15
13  P=X(J+1)-XG(I)
    R=XG(I)
    GOTO 15
14  P=X(J+1)-X(J)
    R=X(J)
15  NH=NH+1
    XH(NH)=R+P*ABS(F1)/(ABS(F1)+ABS(F2))
    H(NH)=F(XH(NH),X(J),Y(J),X(J+1),Y(J+1))
C
C
C   SECVECTA DE PLOTARE A UNEI PORTIUNI VIZIBILE DACA E CAZUL
C
16  IF(F1*F2.GE.0.AND.F1.NE.0) GOTO 19
    IF(F1.LT.0.OR.SEMN.LT.0) GOTO 17
C   WRITE(2,73) ICAZ,NH,I,J,IIND,SEMN
C73  FORMAT(' ',5I12,F12.1)
    CALL PDATA(XH,H,IIND,NH,SCARAX,SCARAY*SIGNUM)
    IF(F2.NE.0) GOTO 18
    SEMN=-1
    GO TO 18
17  SEMN=SIGN(1.,F2)
18  IIND=NH
C
C
C   CALCULUL CAZULUI URMATOR
C
19  ICAZ=0
    IF(XG(I+1)-X(J+1))21,20,22
20  J=J+1
    IF(J.EQ.NT) GOTO 27
    ICAZ=-2
21  I=I+1
    IF(I.EQ.NG) GOTO 37
    ICAZ=ICAZ+5
    GOTO 23
22  J=J+1
    IF(J.EQ.NT) GOTO 27
    ICAZ=1
23  IF(XG(I+1).GT.X(J+1)) ICAZ=ICAZ+1
    GOTO 26
27  IF(F2.GT.0.) GOTO 29
    NH=NH+1
    XH(NH)=X(J)
    H(NH)=F(X(J),XG(I),G(I),XG(I+1),G(I+1))
    GOTO 30
29  NH=NH+1
    XH(NH)=X(J)

```

```

,LP:=HYD

H(NH)=Y(J)
CALL PDATA(XH,H,INDI,NH,SCARAX,SCARAY*SIGNUM)
GO TO 30
37 IF (F) 33,32,31
33 NH=NH+1
XH(NH)=XG(I)
H(NH)=G(I)
32 NH=NH+1
XH(NH)=XG(1)+EPS
H(NH)=F(XH(NH),X(J),Y(J),X(J+1),Y(J+1))
INDI=NH
31 DO 34 K=J+1,NT
NH=NH+1
XH(NH)=X(K)
34 H(NH)=Y(K)
CALL PDATA(XH,H,INDI,NH,SCARAX,SCARAY*SIGNUM)
30 NG=NH
DO 35 I=1,NG
XG(I)=XH(I)
35 G(I)=H(I)
DO 36 I=1,NT
X(I)=X(I)+X*IGI+STX
36 Y(I)=Y(I)*SIGNUM+Y*II-I-STY
RETURN
END

```

### 8.2.7. Subprogramul PDATA

```

,LP:=HYDE

SUBROUTINE PDATA(X,Y,INDI,INDF,SCX,SCY)
DIMENSION X(1),Y(1)
CALL PLOT(X(INDI)*SCX,Y(INDI)*SCY,2)
DO 1 I=INDI+1,INDF
CALL PLOT(X(I)*SCX,Y(I)*SCY,1)
1 CONTINUE
RETURN
END

```

### 8.2.8. Subprogramul PLOT

```

0001 SUBROUTINE PLOT(X,Y,JPEN)
0002 WRITE(2,1)X,Y,JPEN
0003 1 FORMAT(' ',2F10.4,'IR)
0004 RETURN
0005 END

```

## 8.2.9. Subprogramul DATEIN

```

                                *LP:=HYDE
SUBROUTINE DATEIN(X,Y,Z,U,V,W,A,B,C,P,F,G,H,O,R)
D1=X-U
D2=Y-V
U3=Z-W
D=SQRT(D1*D1+D2*D2+D3*D3)
DP=SQRT(D1*D1+D2*U2)
A=D1/D
B=U2/D
C=U3/D
DC=U*U+V*V+W*W
DV=X*X+Y*Y+Z*Z
F=(DC-DV)/(2.*D)
IF(A.EQ.0..AND.B.EQ.0.)GOTO 31
F=ABS(D2)/DP
G=ABS(D1)/DP
GOTO 32
31  G=0.
    F=1.
32  H=ABS(G*C)
    O=ABS(F*C)
    R=SQRT(1.-C*C)
    IF(C.LT.0.) GO TO 1
    IF(A.GE.0..AND.B.GE.0.) GO TO 2
    IF(A.LT.0..AND.B.GE.0.) GO TO 3
    IF(A.LT.0..AND.B.LT.0.) GO TO 4
    IF(A.GE.0..AND.B.LT.0.) GO TO 5
1   IF(A.GE.0..AND.B.GE.0.) GO TO 6
    IF(A.LT.0..AND.B.GE.0.) GO TO 7
    IF(A.LT.0..AND.B.LT.0.) GO TO 8
    IF(A.GE.0..AND.B.LT.0.) GO TO 9
2   F=-F
    H=-H
    O=-O
    RETURN
3   F=-F
    G=-G
    O=-O
    RETURN
4   G=-G
    RETURN
5   H=-H
    RETURN
6   F=-F
    RETURN
7   F=-F
    G=-G
    H=-H
    RETURN
8   G=-G
    H=-H
    O=-O
    RETURN
9   O=-O
    RETURN
END

```

## 8.2.10 Subprogramul PROIEC

```

                                *LP:=HYDE
SUBROUTINE PROIEC(XN,YN,ZN,NVT,A,B,C,P,F,G,H,O,R,PROP,X0,Y0,Z0)
DIMENSION XN(100),ZN(100),YN(100)
INTEGER PROP
C
C   ...INCEPE CICLARĂ PENTRU CALCULUL PROIECTIEI...
C
68  DO 25 I=1,NVT
    DNR=A*XN(I)+B*YN(I)+C*ZN(I)+P
    IF(PROP.NE.1) GO TO 15
    AL=A
    BE=B
    GA=C
    LAM=DNR/(A*AL+B*BE+C*GA)
    X=XN(I)-AL*LAM
    Y=YN(I)-BE*LAM
    Z=ZN(I)-GA*LAM
    GO TO 20
15  T1=X0-XN(I)
    T2=Y0-YN(I)
    T3=Z0-ZN(I)
    DVN=SQRT(T1*T1+T2*T2+T3*T3)
    CL=T1/DVN
    CM=T2/DVN
    CN=T3/DVN
    LAM=DNR/(A*CL+B*CM+C*CN)
    X=XN(I)-CL*LAM
    Y=YN(I)-CM*LAM
    Z=ZN(I)-CN*LAM
25  XN(I)=F*X+G*Y
    ZN(I)=H*X+O*Y+R*Z
    RETURN
END

```

## 8.3. Programe necesare utilizării imprimantei grafice pentru reprezentarea curbelor și suprafețelor

### 8.3.1. Programul PLOTARE

\*LF=PEL,ANUL=FIN

```

LOGIC4=1, BUF(1000), VECT(6), ESC, GE, EL
INTEG(7), TAB(=00,57), YL, XC, XCM, NY, XMIN, XMAX, XCMIN
DATA VECT, ESC, GE, EL / 1, 2, 4, 8, 16, 32, "03", "07", "05 /
COMMON //ZONA//BUF
CALL ASSIGN(2, 'GRAF.DAT')
CALL FDBSET(2, 'R')
HEAD(2) XMIN, X, XMAX, NY
INTABMAX=X
CALL CITIRE(TAB, INTABMAX, INUSF)
XCM=1000
YL=NY+1
TYPE 1
1  FORMAT(' CIND IMPRIMANTA ESTE PREGATITA TASTATI CR!')
   ACCEPT 2, BUF(1)
2  FORMAT(A1)
   DO 20 NRI=1, NY, 6
   DO 16 J=1, XCM+1
16  BUF(J)=64
   XCM=2
   XCMIN=1000
   DO 11 I=1, 6
   YL=YL-1
   IF (YL.EQ.0) GOTO 17
   IND=1
15  IF (TAB(IND,1).EQ.1000) GOTO 10
   IF (YL.GT.TAB(IND,2)) GOTO 11
   IF (YL.LT.TAB(IND,4)) GOTO 12
   IF (TAB(IND,2).EQ.TAB(IND,4)) GOTO 13
   XC=INT((FLOAT(TAB(IND,2))-YL)/FLOAT(TAB(IND,2)-TAB(IND,4)))
1  *FLOAT(TAB(IND,3)-TAB(IND,1)))+TAB(IND,1)
   IF (XC.LT.XMIN) XC=XMIN-1
   IF (XC.GT.XMAX) XC=XMAX+1
   IF (TAB(IND,5).EQ.1000) GOTO 30
   IF (XC.EQ.TAB(IND,5).AND.(XC.EQ.XMIN-1.OR.XC.EQ.XMAX+1)) GOTO 10
   K=MINO(XC, TAB(IND,5))
   DO 31 JJ=K, MAXU(XC, TAB(IND,5))-1
   IF (BUF(JJ)=64.LT.VECT(I)) BUF(JJ)=BUF(JJ)+VECT(I)
31  CONTINUE
   IF (JJ.GT.XCM) XCM=JJ
   IF (K.LT.XCMIN) XCMIN=K
30  TAB(IND,5)=XC
   GOTO 10
13  IF (MAXU(TAB(IND,1), TAB(IND,3)).LT.XMIN.OR.
1  MINO(TAB(IND,1), TAB(IND,3)).GT.XMAX) GOTO 12
   K=MAXU(XMIN, MINO(TAB(IND,1), TAB(IND,3)))
   DO 33 JJ=K,
1  MINO(XMAX, MAXU(TAB(IND,1), TAB(IND,3)))
   IF (BUF(JJ)=64.LT.VECT(I)) BUF(JJ)=BUF(JJ)+VECT(I)
33  CONTINUE
   IF (JJ=1.GT.XCM) XCM=JJ-1
   IF (K.LT.XCMIN) XCMIN=K
12  TAB(IND,1)=1000
10  IND=IND+1
   IF (IND.LE.INTABMAX) GOTO 15
   IF (INUSF.EQ.1) GOTO 11
   CALL COMPACT(TAB, INTABMAX)

```

,LP:=PLOTARE.FTN

```

      IND=INTABMAX+1
      CALL CITIRE(TAB,INTABMAX,INDSF)
      GOTO 15
11    CONTINUE
17    NBLK=(XCMIN-1)/9
      IF(NBLK.EQ.0)GOTO 40
      DO 41 J=1,NBLK
41    BUF(J)=32
      BUF(NBLK+1)=ESC
      BUF(NBLK+2)=GE
      I=NBLK*9+1
      DO 42 J=NBLK+3,XCMAX-NBLK*8+3
      BUF(J)=BUF(1)
      I=I+1
42    CONTINUE
      BUF(J)=EL
      CALL TRANS(J)
      GOTO 43
40    BUF(1)=ESC
      BUF(2)=GE
      BUF(XCM+1)=EL
      CALL TRANS(XCM+1)
43    CALL COMPACT(TAB,INTABMAX)
      CALL WAITF
20    CONTINUE
      WRITE(3,25)19
25    FORMAT(A1)
      STOP
      END

```

### 8.3.2. Subprogramul COMPACT

,LP:=PLOTARE.FTN

```

SUBROUTINE COMPACT(TAB,INTABMAX)
INTEGER TAB(500,5)
I=INTABMAX
J=1
4    IF(TAB(J,1).NE.1000)GOTO 1
      IF(J.EQ.1)GOTO 2
      DO 3 I1=J,I-1
        TAB(I1,1)=TAB(I1+1,1)
        TAB(I1,2)=TAB(I1+1,2)
        TAB(I1,3)=TAB(I1+1,3)
        TAB(I1,4)=TAB(I1+1,4)
        TAB(I1,5)=TAB(I1+1,5)
3    CONTINUE
      I=I-1
      GOTO 4
2    I=I-1
      J=J-1
      IF(J.EQ.0)GOTO 5
      GOTO 4
1    J=J-1
      IF(J.NE.0)GOTO 4
      INTABMAX=I
      RETURN
5    INTABMAX=1
      RETURN
      END

```

## 8.3.3. Subprogramul CITIRE

```

SUBROUTINE CITIRE(TAB,INTABMAX,INDSF)
  INTEGER TAB(500,5)
  INDSF=0
  DO 1 I=INTABMAX+1,500
    READ(2,END=3)(TAB(I,J),J=1,4)
    TAB(I,5)=1000
1   CONTINUE
    GOTO 4
3   INDSF=1
4   INTABMAX=I-1
    RETURN
  END

```

## 8.3.4. Programul ORDONARE

```

,LP1=ORDONARE.FTN

```

```

LOGICAL*1 FISI(25)
  INTEGER XMIN,XMAX
  TYPE 1
3   FORMAT(' NUME FISIER DE DATE : ',S)
  ACCEPT 2,NRCAR,FISI
4   FORMAT(0,25A1)
  CALL ASSIGN(2,FISI,NRCAR)
  CALL FDBSET(2,'R')
  TYPE 3
3   FORMAT(' INTRODUCETI COORDONATELE FERESTREI DE DESEN'/
1   ' XMIN,XMAX,YMIN,YMAX(*F10.5) : ',S)
  ACCEPT 4,XMINU,XMAXD,YMIND,YMAXD
4   FORMAT(4F10.5)
  TYPE 5
5   FORMAT(' INTRODUCETI COORDONATELE IN SPATIU PAGINA'/
1   ' XMINP,XMAXP (2I5) : ',S)
  ACCEPT 5,XMIN,XMAX
  IF(XMIN.LT.3)GOTO 11
6   FORMAT(2I5)
12  CALL ASSIGN(1,'GRAF.DAT')
  CALL FDBSET(1,'N')
  DEL=FLOAT(XMAX-XMIN)/(XMAXD-XMIND)
  NY=INT((YMAXD-YMIND)*DEL)
  WRITE(1)XMIN,32767,XMAX,NY
10  READ(2,END=7)XI,YI,XF,YF
  IXI=INT((XI-XMIND)*DEL)+XMIN
  IXF=INT((XF-XMIND)*DEL)+XMIN
  IYI=INT((YI-YMIND)*DEL)
  IYF=INT((YF-YMIND)*DEL)
  IF(IYI.GT.IYF)GOTO 9
  WRITE(1)IXF,IYF,IXI,IYI
  GOTO 10
11  XMAX=XMAX+3-XMIN
  XMIN=3
  GOTO 12
9   WRITE(1)IXI,IYI,IXF,IYF
  GOTO 10
7   CALL CLOSE(1)
  CALL CLOSE(2)
  STOP
END

```

## 8.4. Aplicațiile procedurii HIDE [Figurile 8.6 – 8.12]

suprafata :

$$z(x,y) = 0,2 \sin x \cos y - e^{[-(x-\bar{x})^2 - (y-\bar{y})^2]} \cos [1,75((x-\bar{x})^2 + (y-\bar{y})^2)] 1,5$$

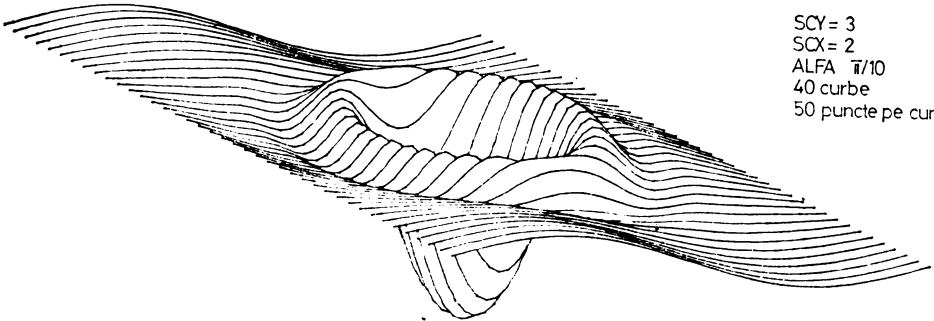
$$x \in [0, 2\bar{x}]$$

$$y \in [0, 2\bar{y}]$$

$\alpha = \bar{\alpha}/4$  unghi de plotare

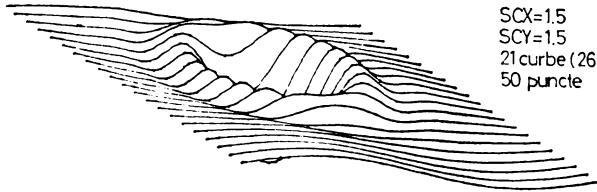
scara 10:1

26 de curbe cu 50 puncte pe fiecare curba



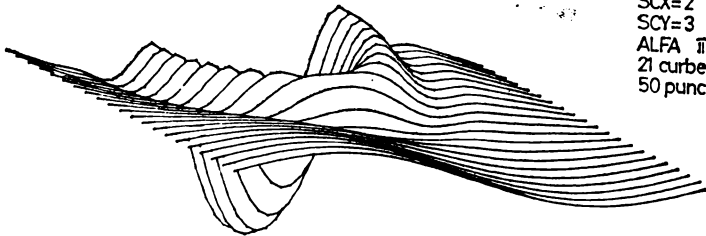
SCY= 3  
SCX= 2  
ALFA  $\bar{\alpha}/10$   
40 curbe  
50 puncte pe curba

Fig. 8.6 a



SCX=1.5  
SCY=1.5  
21 curbe (26-5)  
50 puncte

Fig. 8.6 b



SCX=2  
SCY=3  
ALFA  $\bar{\alpha}/10$   
21 curbe  
50 puncte

Fig. 8.6 c

**DESENE EXECUTATE LA IMPRIMANTA GRAFICĂ**

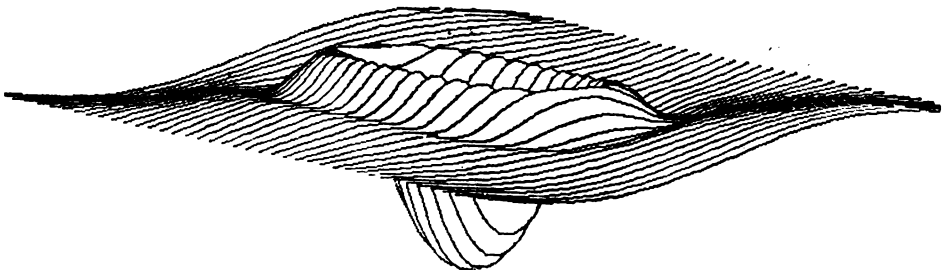


Fig. 8.6 d



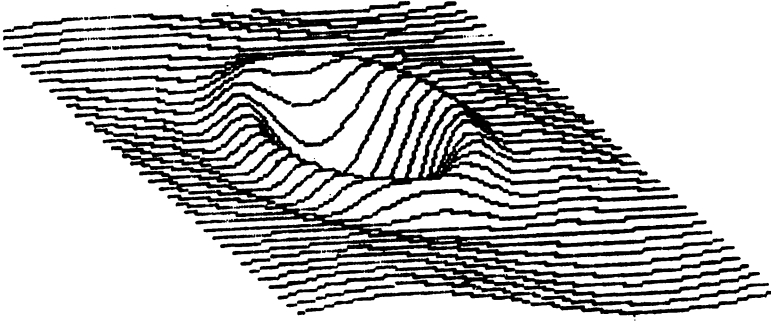


Fig. 8.6 e

suprafata :  
 $z = T(x^3 - 2x^2 + x)(y^3 - 2y^2 + y)$   
 suprafata COONS  
 $x \in [0, 1]$   
 $y \in [0, 1]$   
 $\alpha = 3\pi/4$  (unghi de plotare)  
 scara 1/1  
 40 de curbe cu 50 puncte pe fiecare curba

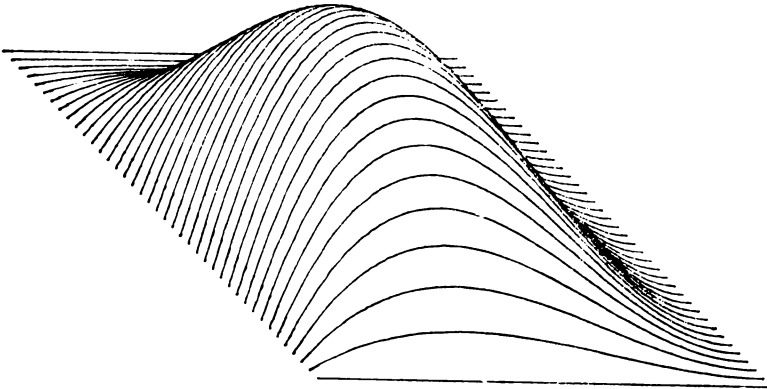


Fig. 8.7

suprafata :  $z = y^2(2y - 3) - y(2x^3 - 3x^2 + 1)(y - 1)^2$   
 suprafata FERGUSON  
 $x \in [0, 1]$   $y \in [0, 1]$   
 $\alpha = 3\pi/4$  unghi de plotare  
 40 de curbe cu 50 de puncte pe fiecare curba

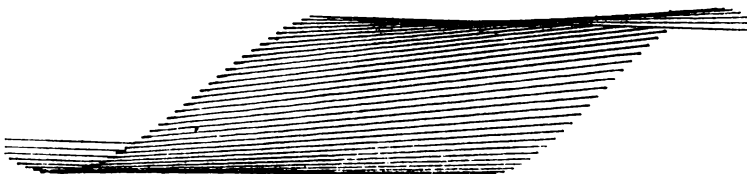


Fig. 8.8

suprafata :  $z = x^2 - 4x + y^2 + 4$

scu

$$x = 2s + 1$$

$$y = 2t + 1$$

$$z = 4(s^2 - s - t^2 + t)$$

$$x \in [-1, 1] \quad y \in [-1, 1]$$

$\alpha = 3\pi/4$  unghi de plotare

40 de curbe cu 50 de puncte pe fiecare curva

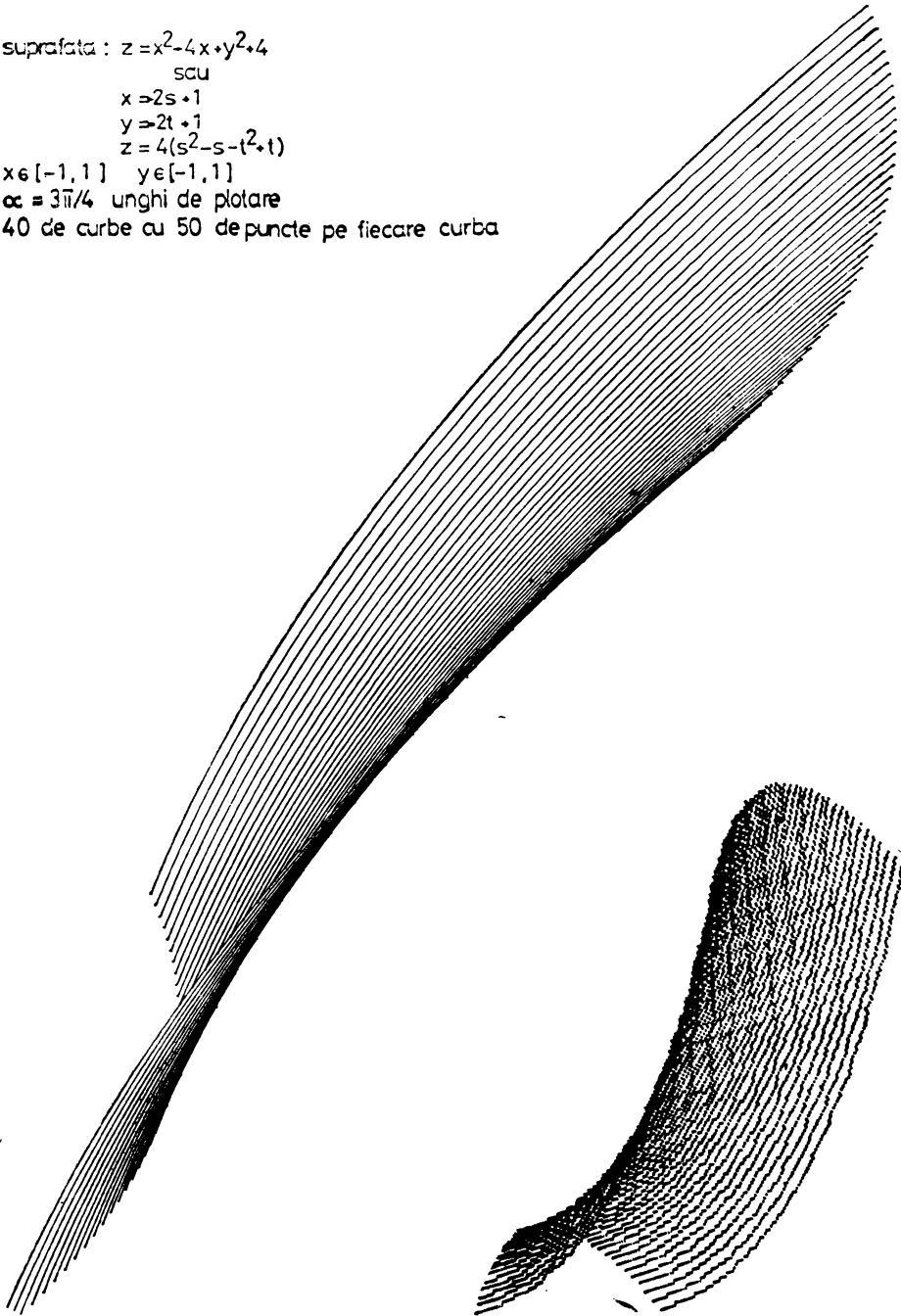


Fig. 8.9 a

Fig. 8.9 b

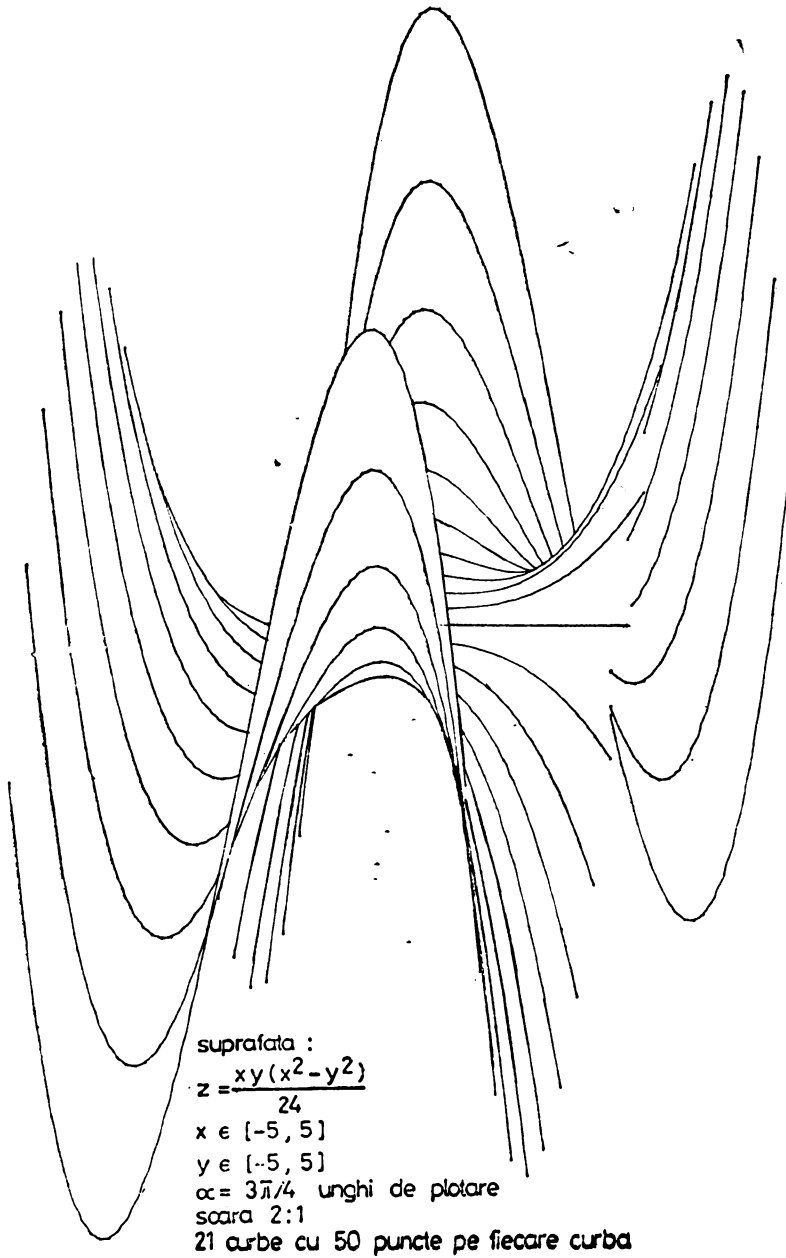


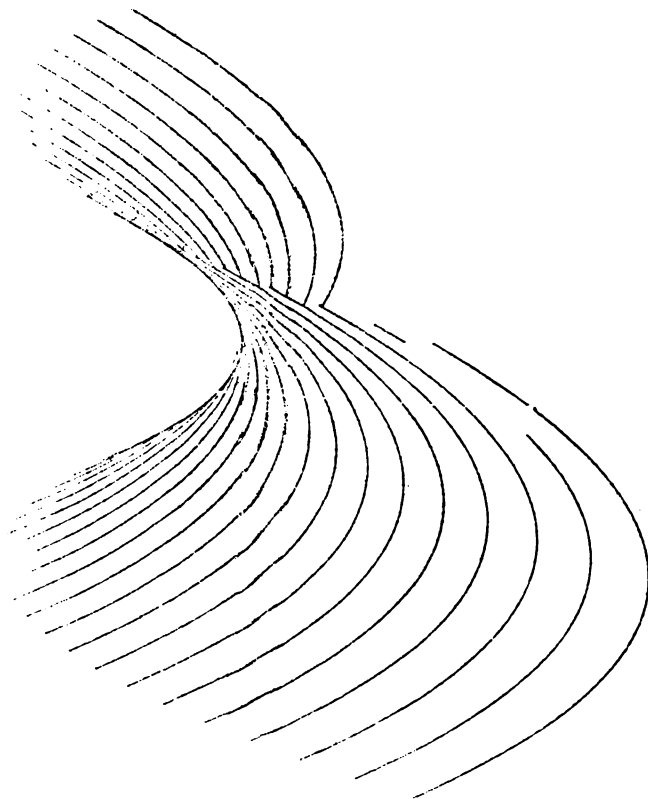
Fig. 8.10

SUPRAFAȚA

$$Z = \frac{1}{8} y^3 \sin x + \sin x$$

$$x \in [0, 4\pi]$$

$$y \in [-5, 5]$$



suprafața :

$$z = \frac{x^2 - y^2}{5}$$

(paraboloid hiperbolic)

$$x \in [-5, 5]$$

$$y \in [-5, 5]$$

$\alpha = 3\pi/4$  unghi - de plotare

scara 1:1

21 de curbe cu 50 puncte pe fiecare curba

Fig. 8.12

Fig. 8.11

## 8.5. Interpolarea bivariată și montarea suprafețelor netede bazată pe proceduri locale

### 8.5.1. Introducere

Prin interpolarea bivariată și montarea suprafețelor netede bazată pe proceduri locale se realizează *filarea curbelor de niveu* în mod automat precum și desenarea automată a acestora, fie că se pornește de la o rețea de puncte inițial definită, fie că se pornește de la o mulțime de puncte dispersate. Studiul este foarte util nu numai prin aplicațiile sale în construcții ci și în alte domenii cum ar fi de exemplu căutarea deformării formei unei din circuitele electronice sau din undele sonice, etc.

Programele au fost scrise în limbajul **FORTRAN** iar execuția la ploter a fost făcută pe masa de desen **ARISTO**. Testarea programelor elaborate în dialog interactiv a fost făcută pe calculatorul Independent I 100.

Astfel fie o funcție de două variabile  $z = z(x, y)$  pentru care se presupune că există valori date în vîrfurile unui caroiaj dreptunghiular  $x = x_i$  ( $i = 1, 2, \dots$ ) și  $y = y_j$  ( $j = 1, 2, \dots$ ). Atît  $x_i$  cît și  $y_j$  pot fi inegal distanțate. Funcția de interpolare  $z = z(x, y)$  este netedă, adică funcția și derivatele ei de ordinul 1 sînt continue.

Schema de interpolare, pe care o prezentăm, este o extensie a metodei pentru interpolarea bivariată dezvoltată de *Hiroshi Akima* și este bazată pe proceduri locale.

Schema este proiectată pentru a elimina ondulațiile excesive între punctele de caroiaj date.

În ceea ce privește calculele necesare, în general, dacă funcția cu o singură variabilă, de interpolare, este liniară nu mai este necesară dezvoltarea unei interpolări cu 2 variabile, interpolarea realizîndu-se întîi pe direcția  $x$  apoi pe direcția  $y$  (sau invers). Rezultatul este întotdeauna neted și identic în cele 2 cazuri.

Cînd s-a aplicat metoda anterioară folosind funcția de o singură variabilă propusă de *H. Akima* s-a observat că rezultatul nu este întotdeauna neted și chiar mai mult, rezultatele diferă de cele obținute cînd se aplică funcția în ordine inversă.

Acest lucru sugerează necesitatea dezvoltării unei metode integrate de interpolare cu 2 variabile.

### 8.5.2. Descrierea metodei de interpolare bivariată

Metoda este bazată pe o funcție pe porțiuni, compusă dintr-un set de polinoame bicubice în  $x$  și  $y$ . Un polinom bicubic în  $x$  și  $y$  este un polinom care are termeni de forma  $x^\alpha y^\beta$ , unde  $\alpha = 0, 1, 2, 3$  și  $\beta = 0, 1, 2, 3$ . Fiecare polinom este aplicabil unei suprafețe dreptunghiulare mărginite de dreptele  $x = x_i$ ,  $x = x_{i+1}$ ,  $y = y_j$  și  $y = y_{j+1}$ . Fiecare polinom este determinat de valorile date ale funcției  $z(x, y)$ , și de valorile derivatele parțiale  $z_x = (\partial_z / \partial_x)$ ,  $z_y = (\partial_z / \partial_y)$ , și  $z_{xy} = (\partial_x^2 / \partial_x \partial_y)$  în cele 4 puncte de colț ale dreptunghiului. Aceste derivate parțiale sînt determinate local utilizînd anumite presupuneri care vor fi discutate mai jos.

Trebuie specificate că presupunerile pentru determinarea derivatelor parțiale nu sînt unice.

Presupunem că  $z_x$  și  $z_y$  în punctul  $(x_3, y_3)$  sînt date de

$$(z_x)_{33} = (w_{x2} * c_{23} + w_{x3} * c_{33}) / (w_{x2} + w_{x3}),$$

$$(z_y)_{33} = (w_{y2} * d_{32} + w_{y3} * d_{33}) / (w_{y2} + w_{y3})$$

$$w_{x2} = |c_{43} - c_{33}|, \quad w_{x3} = |c_{23} - c_{13}|, \quad w_{y2} = |d_{34} - d_{33}|, \quad w_{y3} = |d_{32} - d_{31}|,$$

iar diferențele divizate de ordinul întii sînt date de:

$$c_{ij} = [(z)_{i+1,j} - (z)_{ij}] / (x_{i+1} - x_i) \quad d_{ij} = [(z)_{i,j+1} - (z)_{ij}] / (y_{j+1} - y_j),$$

Extinzînd aceste presupuneri la cazul bidimensional presupunem că derivata parțială de ordinul doi  $z_{xy}$  în punctul  $(x_3, y_3)$  este determinată de:

$$(z_{xy})_{33} = [w_{x2}(w_{y2}e_{22} + w_{y3}e_{23}) + w_{x3}(w_{y2}e_{32} + w_{y3}e_{33})] / [(w_{x2} + w_{x3})(w_{y2} + w_{y3})]$$

unde coeficienții de ponderare sînt cei anteriori, la care se adaugă:

$$e_{ij} = (c_{i,j+1} - c_{ij}) / (y_{j+1} - y_j) - (d_{i+1,j} - d_{ij}) / (x_{i+1} - x_i)$$

În cazul în care  $w_{x2} = w_{x3} = 0$  și (sau)  $w_{y2} = w_{y3} = 0$  derivatele parțiale sînt nedefinite. În scopul de a da un rezultat unic definit în aceste cazuri, punem  $w_{x2} = w_{x3} = 1$  și (sau)  $w_{y2} = w_{y3} = 1$  în mod convențional.

Rezultatul obținut este o suprafață netedă nu numai în punctele caroiajului dar și în lungul liniilor care formează dreptunghiurile. De exemplu considerînd valorile lui  $z$  în lungul segmentului ce unește punctele  $(x_i, y_j)$  și  $(x_{i+1}, y_j)$  se observă că valorile lui  $z$  și  $z_x$  în cele două puncte, determină în mod unic un polinom cubic în  $x$  pe segment.

Polinomul bicubic în  $x$  și  $y$  reprezentînd valorile lui  $z$  în oricare dintre dreptunghiurile ce au drept limită segmentul considerat se reduce la un polinom cubic în  $x$ . Astfel, cele două polinoame bicubice vor coincide unul cu celălalt în lungul segmentului.

Acest fapt probează continuitatea rezultatului interpolării în lungul segmentului.

Deoarece  $(z_x) = z_{xy}$ , putem proba pe baza acelorași considerente continuitatea lui  $z_y$  și astfel netezirea lui  $z$  în lungul segmentului.

Vom prezenta în continuare procedura de bază a programului, care în fază finală urmărește trasarea curbelor de nivel pentru o suprafață dată și pentru orice nivel al suprafeței.

### 8.5.3. Subrutina SFCFIT

Această procedură potrivește o suprafață netedă a unei funcții de 2 variabile  $z = z(x, y)$  unui set de date de intrare reprezentînd punctele unui caroiaj într-un plan  $x - y$ . Ea generează un set de rezultate pentru un caroiaj îndeșit divizînd în mod egal coordonatele  $x$  și  $y$  în fiecare interval între o pereche de puncte de intrare și interpolînd valoarea  $z$  generează un set de puncte de ieșire format din punctele inițiale și punctele interpolate.

Apelul procedurii este următorul:

**CALL SFCFIT (IU, LX, LY, X, Y, Z, MX, MY, NU, NV, U, V)**

unde parametrii de apel au următoarea semnificație:

$IU$  — numărul logic al unității standard de ieșire;  $LX$  — numărul de coordonate pe  $X$  ale caroiajului (trebuie să fie mai mare decât 2);  $LY$  — numărul de coordonate pe  $Y$  ale caroiajului (trebuie să fie mai mare decât 2);  $X$  — vector de dimensiunea  $LX$  ce conține coordonatele pe  $X$  ale punctelor de intrare) în ordine crescătoare sau descrescătoare);  $Y$  — vector de dimensiunea  $LY$  ce conține coordonatele pe  $Y$  ale punctelor de intrare (în ordine crescătoare sau descrescătoare);  $Z$  — matrice ( $LX$ ,  $LY$ ) ce conține valorile funcției în punctele de intrare  $MX$  — numărul de subintervale între fiecare pereche de puncte de intrare pe axa  $X$  ( $\geq 2$ );  $MY$  — numărul de subintervale între fiecare pereche de puncte de intrare pe axa  $Y$  ( $\geq 2$ );  $NU$  — numărul de puncte de ieșire pe axa  $X$ ;  $-NU = (LX - 1 * MX + 1)$ ;  $NV$  — numărul de puncte de ieșire pe axa  $Y$ ;  $-NV = (LY - 1) * MY + 1$ ;

Parametrii de ieșire ai subrutinei sînt:

$U$  — vector de dimensiune  $NU$  unde se vor găsi valorile coordonatelor  $X$  ale punctelor de ieșire;  $V$  — vector de dimensiune  $NV$  unde se vor găsi valorile coordonatelor  $Y$  ale punctelor de ieșire.

## PROGRAM PRINCIPAL

### CURBE DE NIVEL

```

LOGICAL*1 FISI(15),FISE(15),ZERU(40,40)
DIMENSION X(40),Y(40),Z(40,40),U(500),V(500),w1(15),w2(15)
100
1  FORMAT(' MOD DE LUCRU =CAROIAJ Z=NIVEL 3=EXIT : ',3)
   ACCEPT 2,ITIP
   IF (ITIP.EQ.3) GOTO 3000
2  FORMAT(5I5)
   TYPE 3
3  FORMAT(' NOME FISIER DE INTRARE : ',5)
   ACCEPT 4,IRRI,FISI
4  FORMAT(0,15A1)
   TYPE 5
5  FORMAT(' NOME FISIER DE IESIRE : ',5)
   ACCEPT 6,IRE,FTCF
   IF (ITIP.EQ.2) GOTO 1000
   CALL ASSIGN(4,FISI,NRI)
   CALL FDSSET(4,1,1)
1000 READ (4,2)IX,LX,LY,MY,IU
   READ (4,6)(X(IX),IX=1,LX)
   READ (4,6)(Y(IY),IY=1,LY)
   DO 7 IY=1,LY
7  READ (4,6)(Z(IX,IY),IX=1,LX)
6  FORMAT(10F6,2)
   NRZERO=0
C  CALCUL SCURURI PENTRU ZEROURI (PUNCTE NEDEFINITE)
   DO 406 I=1,LX
   DO 406 J=1,LY
   IF (Z(I,J).NE.0.)GOTO 408
   ZERU(I,J)=1
408  CONTINUE
   DO 400 I=1,LX
   DO 400 J=1,LY
   IF (Z(I,J).NE.0.)GOTO 400
   Z(I,J)=0.
   NRZERO=NRZERO+1
   DO 401 K1=1,3
   IX=1+K1-2
   IF (IX.EQ.0)GOTO 401
   IF (IX.GT.LX)GOTO 401
   DO 409 K2=1,3
   IF (K1.EQ.2.AND.K2.EQ.2)GOTO 409
   JX=J+K2-2
   IF (JX.EQ.0)GOTO 409
   IF (JX.GT.LY)GOTO 409
   Z(ZERU(IX,JX).EQ.0)Z(I,J)=Z(I,J)+1.
409  CONTINUE
401  CONTINUE
400  CONTINUE
   IF (NRZERO.EQ.0)GOTO 407
C  CALCUL SCOR MAXIM PE MATRICE
   DO 402 I=1,NRZERO
   MAXSC=0
   DO 403 K1=1,LX
   DO 403 K2=1,LY
   IF (ZERU(K1,K2).NE.1)GOTO 403
   IF (Z(K1,K2).LE.MAXSC)GOTO 403

```

```

      IMAX=K1
      JMAX=K2
      MAXSC=Z(K1,K2)
403  CONTINUE
      SUMA=0.
C    CALCUL COTE NENFINITE
      DU 404 K1=1.3
      IX=IMAX+K1-2
      IF (IX.EQ.0) GOTO 404
      IF (IX.GT.LX) GOTO 404
      DU 410 K2=1.3
      IF (K1.EQ.2.AND.K2.EQ.2) GOTO 410
      JX=JMAX+K2-2
      IF (JX.EQ.0) GOTO 410
      IF (JX.GT.LY) GOTO 410
      IF (ZERO (IX,JX).EQ.1) GOTO 405
      SUMA=SUMA+Z (IX,JX)
      GOTO 410
405  Z (IX,JX)=Z (IX,JX)+1
410  CONTINUE
404  CONTINUE
      Z (IMAX,JMAX)=SUMA/MAXSC
      ZERO (IMAX,JMAX)=2
402  CONTINUE
      DU 406 I=LX-2,-1
      DU 406 J=LY-2,-1
      IF (ZERO (I,J)+ZERO (I-1,J)+ZERO (I,J-1)+ZERO (I-1,J-1).EQ.0) GOTO 406
      ZERO (I,J)=1
406  CONTINUE
407  CALL CLOSE (4)
      CALL ASSIGN (4,FTSE,NRE)
      CALL FDBSET (4,INFW)
      WRITE (4) LX,LY,MX,MY,1
      CALL SFCFIT (4,IX,LY,X,Y,Z,MX,MY,U,V,ZERO,IER)
      IF (IER.NE.0) GOTO 3000
      CALL CLOSE (4)
      GOTO 100
1000 CALL ASSIGN (4,FTSI,NRI)
      CALL FDBSET (4,INFI)
      CALL ASSIGN (3,FTSE,NRE)
      CALL FDBSET (3,INFW)
      CALL NIVEL (4,3,M1,W2,U,V,ZERO)
      CALL CLOSE (3)
1001 CALL CLOSE (4)
      GOTO 100
3000 STOP
      END

```

### Subprogramul NIVEL

```

      SUBROUTINE NIVEL (NFI,NFE,w1,w2,U,V,ZERO)
      DIMENSION U (1),V (1),w1 (1),w2 (1)
      VIRTUAL PLOT (2000,4)
      LOGICAL *1 ZERO (40,40)
      F (Z1,Z2,*)=(w-Z1)/(Z2-Z1)
      DATA EPS/0.001/
      CALL INI (NFE)
      TYPE 1191
1191  FORMAT (' DORIT CURBE CU PAS CONSTANT ? (I=DA U=NU) : ',5)
      ACCEPT 2,ICC
      IF (ICC.NE.1) GOTO 8000
      TYPE 1193
1193  FORMAT (' INTRODUCETI NIVEL BAZA , INCREMENT [ZPB,Z] : ',5)
      ACCEPT 3,WMIN,WPAS
      FORMAT (2F8.2)
      WMIN=WPAS
      REWIND NFI
      READ (NFI) LX,LY,MX,MY,IU
      IF INX=(LX-1)*MX+1
      IF INY=(LY-1)*MY+1
      READ (NFI) (U (I),I=1,IFINX)
      READ (NFI) (V (I),I=1,IFINY)
      IF (ITH.EQ.2) GOTO 1122
      ITH=2
      TYPE 1120

```



```

1120 FOR A1(1) DIMITY TRASARE CONTOR : (U=NU I=DA) : 1,5)
ACCEPT 1121,111
1121 FORMAT(12)
IF (111.NE.1)GOTO 1122
DO 300 I=2,LX
DO 300 J=2,LY
XS=U((I-2)*MX+1)
XD=U((I-1)*MX+1)
YJ=V((J-2)*MY+1)
YS=V((J-1)*MY+1)
IF (ZERO(I,J).EQ.0)GOTO 301
IF (J.EQ.2)CALL LIN(XS,YJ,XS,YJ)
IF (J.EQ.LY)CALL LIN(XS,YS,XD,YS)
IF (I.EQ.2)CALL LIN(XS,YJ,XS,YS)
IF (I.EQ.LX)CALL LIN(XD,YJ,XD,YS)
GOTO 300
301 IF (I.GT.2.AND.7FRO(I-1,J).EQ.0)CALL LIN(XS,YJ,XS,YS)
IF (I.LT.LX.AND.7FRO(I+1,J).EQ.0)CALL LIN(XD,YJ,XD,YS)
IF (J.GT.2.AND.7FRO(I,J-1).EQ.0)CALL LIN(XS,YJ,XD,YJ)
IF (J.LT.LY.AND.7FRO(I,J+1).EQ.0)CALL LIN(XS,YS,XD,YS)
300 CONTINUE
1122 IF (ICC.NE.1)GOTO 1123
W=WPAS
GOTO 1124
1123 TYPE 1004
1004 FORMAT(' INTRUDUCETI VALOARE NIVEL : 1,5)
ACCEPT 1005,*
1005 FORMAT (F6.2)
1124 LAM1=LX-1
LYM1=LY-1
MXP1=MX+1
MYP1=MY+1
LM=1
DO 101 J=1,1600
REAL (NF1,ENU=999)NRX,NRY,ZMIN,ZMAX
IF (ZMIN.LT.W.AND.ZMAX.GT.W)GOTO 25
DO 26 K=1,MYP1
HEAD (NF1)
GOTO 101
26 NRY=NMY-1
25 NRX=NX-1
JY=(NRY-1)*MYP1
IX=(NRX-1)*MX+1
HEAD (NF1)(WZ(K),K=1,MXP1)
YLU=V(JY)
DO 8 JJ=1,MY
YL=YLU
YLU=V(JY+JJ)
STY=YLU-YL
DO 9 K1=1,MXP1
W1(K1)=WZ(K1)
HEAD (NF1)(WZ(K),K=1,MXP1)
F1=W1(1)
F2=W2(1)
XCU=U(IX)
DO 8 I1=1,MX
XC=XCU
XCU=U(IX+I1)
STA=XCU-XC
F4=F1
F3=F2
F1=W1(I1+1)
F2=W2(I1+1)
ICUD=U
IF (W.GE.F1) ICUD=ICOD+1
IF (W.GE.F2) ICUD=ICOD+2
IF (W.GE.F3) ICUD=ICOD+4
IF (W.GE.F4) ICUD=7-ICOD
IF (ICUD)11,8,11
GOTO (10,20,30,40,50,60,70),ICOD
11 PLOTP(LM,1)=XC+F(F4,F1,W)*STX
10 PLOTP(LM,4)=YL+F(F1,F2,W)*STY
PLOTP(LM,2)=YL
PLOTP(LM,3)=XC
GOTO 80

```

```

20      PLOTP(LM,1)=XC+F(F3,F2,w)*STX
      PLOTP(LM,4)=YL+F(F1,F2,w)*STY
      PLOTP(LM,2)=YL
      PLOTP(LM,3)=XCU
      GOTO 80
30      PLOTP(LM,3)=XC+F(F3,F2,w)*STX
      PLOTP(LM,1)=XC+F(F4,F1,w)*STA
      PLOTP(LM,2)=YL
      PLOTP(LM,4)=YLU
      GOTO 80
40      PLOTP(LM,3)=XC+F(F3,F2,w)*STX
      PLOTP(LM,2)=YL+F(F4,F3,w)*STY
      PLOTP(LM,1)=XC
      PLOTP(LM,4)=YLU
      GOTO 80
50      PLOTP(LM+1,1)=XC+F(F4,F1,w)*STX
      PLOTP(LM,3)=XC+F(F3,F2,w)*STX
      PLOTP(LM,2)=YL+F(F4,F3,w)*STY
      PLOTP(LM+1,4)=YL+F(F1,F2,w)*STY
      PLOTP(LM,1)=XC
      PLOTP(LM,4)=YLU
      PLOTP(LM+1,2)=YL
      PLOTP(LM+1,3)=XCU
      LM=LM+1
      GOTO 80
60      PLOTP(LM,2)=YL+F(F4,F3,w)*STY
      PLOTP(LM,4)=YL+F(F1,F2,w)*STY
      PLOTP(LM,1)=XC
      PLOTP(LM,3)=XCU
      GOTO 80
70      PLOTP(LM,2)=YL+F(F4,F3,w)*STY
      PLOTP(LM,1)=XC
      PLOTP(LM,4)=YL
      PLOTP(LM,3)=XC+F(F4,F1,w)*STX
      LM=LM+1
80      CONTINUE
901     CONTINUE
999     LM=LM+1
      TYPE 12,w,LM,M,0
      FORMAT(F7.2,F2IM)
12      IF(LM.EQ.0)GOTO 93
1125     INDI=1
      INDU=2
      INDI=1
      INDI=1
      JTI=1
      ASSIGN 3004 TO TET2
3005     IF(INDI.GE.LM)GOTO 93
3004     DO 3005 I=INDU,LM
      IF(ABS(PLOTP(INDI,3)-PLOTP(I,1)).GT.EPS)GOTO 3000
      IF(ABS(PLOTP(INDI,4)-PLOTP(I,2)).GT.EPS)GOTO 3000
      SECVENTE DE MUTRE
      DO 3001 J=1,4
      V1=PLOTP(I,J)
      PLOTP(I,J)=PLOTP(INDU,J)
      PLOTP(INDU,J)=V1
      GOTO 3003
3001     IF(ABS(PLOTP(INDI,3)-PLOTP(I,3)).GT.EPS)GOTO 3005
      IF(ABS(PLOTP(INDI,4)-PLOTP(I,4)).GT.EPS)GOTO 3005
3003     DO 3002 J=1,4
      V1=PLOTP(I,J)
      PLOTP(I,J)=PLOTP(INDU,1+MOD(J+1,4))
      PLOTP(INDU,1+MOD(J+1,4))=V1
3002     INDI=INDU
3003     INDU=INDU+1
      GOTO 1ET2
3005     CONTINUE
      TEST DE DETERMINARE A CURBELOR INCHISE
      IF(ABS(PLOTP(INDI,1)-PLOTP(INDU,3)).LT.EPS.AND

```

```

      *ABS(PLUTP(INDI,2)-PLUTP(IND,4)).LT.EPS)JTIP=3
C     DESPARTIRE DUPA VALORILE LUI JTIP
C     JTIP=1  PRIMA PARTE A UNEI CURBE
C     JTIP=2  A DOUA PARTE A UNEI CURBE
C     JTIP=3  CURBA INCHISA
      GOTO (5000,6000,7000),JTIP
C     SECVENTA PENTRU PLOTAREA UNEI PORTIUNI DE CURBA
4001  CALL PLOT(PLUTP(INDI,1),PLOT(INDI,2),0)
      DU 4000 JJI=INDI+1,IND
      IF(JJI.NE.IM+1)GOTO 4000
      JJI=IS
ERROR 55-W ASSIGNMENT TO DU VARIABLE WITHIN LOOP
      (JJI=1 IN MODULE NIVEL AT LINE 171)

4000  CALL PLOT(PLUTP(JJI,1),PLOT(JJI,2),0)
      CALL PLOT(PLUTP(JJI,1),PLOT(JJI,2),1)
      CALL PLOT(PLUTP(IND,3),PLOT(IND,4),1)
      GOTO TET1
2003  IM=(INDI+IND)/2
      XS=PLUTP(IM,1)
      YS=PLUTP(IM,2)
      DU 2000 IS=IM+1,IND
      DIST=SURT((PLUTP(IS,1)-XS)**2+(PLUTP(IS,2)-YS)**2)
      IF(DIST.GE.S.)GOTO 2001
2000  CONTINUE
      ISCK=0
      GOTO TET3
C     ISCK=0 NU S-A REUSIT SCRIEREA PE CURBA
2001  UN=ATAN2(PLUTP(IS,2)-YS,PLOT(IS,1)-XS)
      IF(XS.GT.PLOT(IS,1))UN=UN+3.14152
      XTR=XS
      YTR=YS
      IF(XS.LT.PLOT(IS,1))GOTO 2002
      XTR=PLOT(IS,1)
      YTR=PLOT(IS,2)
2002  CALL BEG(XTR,YTR)
      CALL ANG(UN)
      CALL TRXY(DIST/2,.-.75,XPL,YPL)
      CALL BEG(0.,0.)
      CALL ANG(0.)
      CALL NUM(XPL,YPL,UN,1.5,1,*.2)
      ISCK=1
C     ISCK=1 PORTIUNEA DE CURBA A FOST SCRISA
      GOTO TET3
C     TRATAREA CAZULUI CU JTIP=1
5000  IF(INDI.NE.IND) GOTO 5001
      ISCK=0
      JTIP=2
      VI=PLUTP(IND,1)
      PLOT(IND,3)=PLOT(IND,3)
      PLOT(IND,3)=VI
      VI=PLUTP(IND,2)
      PLOT(IND,4)=PLOT(IND,4)
      PLOT(IND,4)=VI
      GOTO 3004
5001  ASSIGN 5002 TO TET3

      GOTO 2003
5002  IF(ISCK.EQ.0) IM=0
      ASSIGN 5003 TO TET1
      GOTO 4001
5003  ASSIGN 5004 TO TET2
      INDI=IND
      I=INDI
      GOTO 3007
5004  ASSIGN 5004 TO TET2
      INDI=IND
      JTIP=2
      GOTO 3006
C     TRATAREA CAZULUI CU JTIP=2
6000  IF(INDI.NE.IND)GOTO 6001
6004  INDI=IND+1
      INDI=INDI
      INDI=IND+1
      JTIP=1
      GOTO 3005
6001  IF(ISCK.EQ.1)GOTO 6002
      ASSIGN 6003 TO TET3
      GOTO 2003

```

```

6003  ASSIGN 6004 TO IET1
      IF (ISCH.EQ.0)IK=0
      GOTO 4001
6002  ISCH=0
      GOTO 6003
C   TRATAREA CAZULUI CU JTIP=3
7000  ISCH=0
      GOTO 6001
93    IF (ICC.EQ.1)GOTO 6000
      TYPE 1010
1010  FORMAT(' MAI STMT NIVELE DE CALCULAT ? (U=NO I=DA) : ',3)
      ACCEPT 2,III
      IF (III)6000,1010,6000
1012  CONTINUE
2     FORMAT(I1)
      CALL EUP
      STOP
      END

```

### 8.5.4 Subprogramul SFCFIT

```

0001  SUBROUTINE SFCFIT(IU,LX,LY,X,Y,Z,MX,MY,U,V,ZERO,IER)
C   CALCULAREA SUPRAFETELOR NETEDE
C   ACEASTA SUBMUTINA GASESTE O SUPRAFATA NETEDA PENTRU
C   O FUNCTIE DE DOUA VARTABILE, Z=Z(X,Y) CORESPUNZATOARE
C   UNUI SET DE PUNCTE DE INTRARE DATE IN CAROIAJ IN PLANUL X-Y
C   GENEAREAZA UN SET DE PUNCTE DE IESIRE IN CAROIAJ DIVIZINU
C   COORDONATELE PE X SI Y IN FIECARE INTERVAL CUPRINS INTRE
C   O PERECHE DE PUNCTE ALE CAROIAJULUI INITIAL,INTERPOLEAZA
C   VALOAREA Z CORESPUNZATOARE FIECARUI PUNCT AL CAROIAJULUI
C   INDESIT SI GENEAREAZA UN SET DE PUNCTE DE IESIRE FORMAT
C   DIN PUNCTELE INITIALE SI DIN PUNCTELE OBTINUTE PRIN
C   INDESIREA CAROIAJULUI
C   METODA FOLOSESTE O FUNCTIE PE PORTIUNI COMPUSA DINTR-UN
C   SET DE POLINOAME BICUBICE IN X SI Y,FIECARE POLINOM
C   FIIND APLICABIL PENTRU UN DREPTUNGHI AL CAROIAJULUI DE
C   INTRARE IN PLANUL X-Y.FIECARE POLINOM ESTE DETERMINAT LOCAL
C   PARAMETRI DE INTRARE
C   IU=NUMARUL LOGIC AL INITATII STANDARD DE IESIRE
C   LX=NUMARUL DE PUNCTE AL CAROIAJULUI DAT PE AXA X
C   (VALOAREA MINIMA 2)
C   LY=NUMARUL DE PUNCTE AL CAROIAJULUI DAT PE AXA Y
C   (VALOAREA MINIMA 2)
C   X=VECTOR DE DIMENSIUNEA LX CONTININD COORDONATELE PE X
C   ALE PUNCTELOR CAROIAJULUI DAT(IN ORDINE CRESCATOARE SAU
C   DESCHEASCATOARE)
C   Y=VECTOR DE DIMENSIUNEA LY CONTININD COORDONATELE PE Y
C   ALE PUNCTELOR CAROIAJULUI DAT(IN ORDINE CRESCATOARE SAU
C   DESCHEASCATOARE)
C   Z=VECTOR DUBLU DIMENSIUNAT DE DIMENSIUNE (LX,LY)
C   CONTININD VALORILE FUNCTIEI IN PUNCTELE
C   CAROIAJULUI DAT
C   MX=NUMARUL DE SUBINTERVALE INTRE FIECARE PERECHE
C   DE PUNCTE DE PE AXA X
C   (VALOAREA MINIMA 2)
C   MY=NUMARUL DE SUBINTERVALE INTRE FIECARE PERECHE
C   DE PUNCTE DE PE AXA Y
C   (VALOAREA MINIMA 2)
C   NUMAR DE PUNCTE AL CAROIAJULUI INDESIT PE AXA X
C   = (LX-1)*MX+1
C   NV=NUMAR DE PUNCTE AL CAROIAJULUI INDESIT PE AXA Y
C   = (LY-1)*MY+1
C   PARAMETRI DE IESIRE
C   U=VECTOR DE DIMENSIUNEA NU CONTININD COORDONATELE X
C   ALE CAROIAJULUI INDESIT
C   V=VECTOR DE DIMENSIUNEA NV CONTININD COORDONATELE Y
C   ALE CAROIAJULUI INDESIT
C   IER=INDICATOR DE EROR
C   CITEVA DIN VARIABILELE INTERNE
C   ZA=DIFERENTA DIVIZATA A LUI Z IN RAPORT CU X
C   ZB=DIFERENTA DIVIZATA A LUI Z IN RAPORT CU Y
C   ZAB =DIFERENTA DIVIZATA DE ORDINUL DOI A LUI Z
C   IN RAPORT CU X SI Y
C   ZX=DERIVATA PARTIALA A LUI Z IN RAPORT CU X
C   ZY=DERIVATA PARTIALA A LUI Z IN RAPORT CU Y
C   ZXY=DERIVATA PARTIALA DE ORDINUL DOI A LUI Z
C   IN RAPORT CU X SI Y

```

```

C DECLARATII
DIMENSION X(LX), Y(LY), Z(40,40), U(1), V(1), W(15,15)
DIMENSION ZA(4,2), ZB(5), ZAB(2,3), ZX(2), ZY(2), ZXY(2)
LOGICAL I1,ZERU(40,40)
EQUIVALENCE (ZB2,ZA(1)), (ZB3,ZA(2)), (ZB4,ZA(3)),
* (ZB5,ZA(4)), (ZB6,ZA(5)), (ZB7,ZA(6)), (ZB8,ZA(7)),
* (ZB9,ZA(8)), (ZB10,ZB(1)), (ZB11,ZB(2)), (ZB12,ZB(3)),
* (ZB13,ZB(4)), (ZB14,ZB(5)), (ZB15,ZAB(1)),
* (ZB16,ZAB(2)), (ZB17,ZAB(3)), (ZB18,ZAB(4)),
* (ZB19,ZAB(5)), (ZB20,ZAB(6)), (ZB21,ZA(1)),
* (ZX4,ZX(2)), (ZY4,ZY(1)), (ZXY4,ZXY(2)), (PU0,Z33), (PU1,ZY33),
* (P10,ZX33), (P11,ZXY33)
EQUIVALENCE (XMI,JX), (IXML,JY), (DU,UV,OX,OY),
* (FX,FXA,FXB,FXC,FXD,FXE), (W2,WY2,AW,OU), (W3,WY3,OW,UI),
* (Z4B1,P12), (Z4B2,P13), (Z4B3,P20), (Z4B4,P21),
* (WX2,C,U2), (WX3,D,W3), (Z3A2,P02), (Z4A2,P03),
* (Z3B2,P22), (Z3B4,P23)

C CALCULE PRELIMINARE
C SETAREA ANUMITOR PARAMETRI DE INTRARE LA VALORILE
C VARIABILELOR LOCALE
100 IU = IU
100 LX0 = LX
100 LXM1 = LX0 - 1
100 LYM2 = LXM1 - 1
100 LY0 = LY
100 LYM1 = LY0 - 1
100 LYM2 = LYM1 - 1
100 MX0 = MX
100 MXP1 = MX0 + 1
100 MXM1 = MX0 - 1
100 MY0 = MY
100 MYM1 = MY0 + 1
100 MYM2 = MY0 - 1
100 NU0 = LXM1*MX0+1
100 NY0 = LYM1*MY0+1

C VERIFICAREA ERORILOR
IER=0
10 IF (LXM2.LT.0) GO TO 400
10 IF (LYM2.LT.0) GO TO 410
10 IF (MAM1.LE.0) GO TO 420
10 IF (MYM1.LE.0) GO TO 430
10 IX = 2
10 IF (X(1)-X(2)) 10, 460, 30
10 DO 20 IX=3,LX0
10 IF (X(IX-1)-X(IX)) 20, 460, 470
20 CONTINUE
30 GO TO 50
30 DO 40 IX=4,LX0
30 IF (X(IX-1)-X(IX)) 470, 460, 40
40 CONTINUE
50 IY = 2
10 IF (Y(1)-Y(2)) 60, 490, 80
60 DO 70 IY=3,LY0
10 IF (Y(IY-1)-Y(IY)) 70,490, 500
70 CONTINUE

80 GO TO 100
80 DO 90 IY=3,LY0
10 IF (Y(IY-1)-Y(IY)) 500, 490, 90
90 CONTINUE

C CALCULAREA VECTORULUI U
100 FPA = MX0
100 FPA = 1.0/FPA
100 KU = 1
100 X4 = X(1)
100 X3 = X4
100 DO 120 IX=2,LX0
100 X4 = X4
100 X4 = X(IX)
100 DU = (X4-X3)*FPA
100 DO 110 JX=1,MXM1
100 KU = KU + 1
100 U(KU) = U(KU-1) + DU
110 CONTINUE
100 KU = KU + 1
100 U(KU) = X4

```

```

120 CONTINUE
      WRITE (100) (0(I,I),I=1,KU)
C CALCULAREA VECTORULUI V
      MY = MYU
      MY = 1.0/FMY
      KJ = 1
      Y4 = Y(1)
      V(1) = Y4
      DO 140 IY=2,IYU
      Y3 = Y4
      Y4 = Y(IY)
      UV = (Y4-Y3)*MY
      DO 130 IJ=1,MYM1
      KV = KV + 1
      V(KV) = V(KV-1) + UV
130 CONTINUE
      KV = KV + 1
      V(KV) = Y4
140 CONTINUE
C CICLURI DE PRINCIPALIE
      WRITE (100) V(I,I),I=1,KV)
      JYMX = MYU
      KJU = 0
      DO 390 IY=2,IYU
      IYM2 = IY - 2
      IYM3 = IYM2 - 1
      IYML = IY - IYU
      IYML1 = IYML + 1
      JYB = 0
      IF (IYML.EQ.0) JYMX = MYM1
      JYMA = MAXU
      KJU = 0
      DO 380 IX=1,IXU
      IYML1 = IX - 1
      IXML = IX - IXU
      IF (IXML.EQ.0) JYMA = MYM1
C SECVENȚA DE GĂSIRE A VALORILOR X,Y SI Z DE CALCUL AL
C VALORILOR ZA,ZB SI ZM SI DE ESTIMARE A LUK CIND ESTE
C NECESAR
C CALCUL PRELIMINAR CIND IX.EQ.1
      IF (IXML1.EQ.0) GO TO 150
      Y3 = Y(IY-1)
      Y4 = Y(IY)
      F3 = 1.0/(Y4-Y3)
      F350 = 0.5*F3
      IF (IYM2.GT.0) F2 = 1.0/(Y3-Y(IY-2))
      IF (IYM3.GT.0) F1 = 1.0/(Y(IY-2)-Y(IY-3))
      IF (IYML1.LT.0) F4 = 1.0/(Y(IY+1)-Y4)
      IF (IYML1.LT.0) F5 = 1.0/(Y(IY+2)-Y(IY+1))
      GO TO 160
C SALVAREA VECTORILOR VALORI
150 Z3A2 = Z3A3
      Z4A2 = Z4A3
      X3 = X4
      Z33 = Z43
      Z3B3 = Z4B3
      A3 = A4
      A350 = A3*F3
      Z3A3 = Z3A4
      Z4A3 = Z4A4
      Z43B2 = Z44B2
      Z43B3 = Z44B3
      Z43B4 = Z44B4
160 X4 = X5
      Z43 = Z53
      Z4B1 = Z5B1
      Z4B2 = Z5B2
      Z4B3 = Z5B3
      Z4B4 = Z5B4
      Z435 = Z5B5
      A4 = A5
      Z3A4 = Z3A5
      Z4A4 = Z4A5
      Z44B2 = Z45B2
      Z44B3 = Z45B3
      Z44B4 = Z45B4
170 A5 = X6
      Z53 = Z63
      Z54 = Z64
      Z5B1 = Z6B1
      Z5B2 = Z6B2
      Z5B3 = Z6B3
      Z5B4 = Z6B4
      Z5B5 = Z6B5

```

```

CALCULUL VALORILOR ZA, ZB SI ZAB SI
ESTIMAREA VALORILOR ZA
C CIND (IY.LE.3).OR.(IY.LE.(Y-1))
200 1A6 = 1A0 + 1
    IF (IX.0E.LX) GO TO 260
    AB = X(1A6)
    Z63 = Z(1A6,1Y-1)
    Z64 = Z(1A6,1Y)

    Z6B3 = (Z64-Z63)*B3
    IF (LYM2.EU.0) GO TO 200
    IF (LYM2.EU.0) GO TO 190
    Z62 = Z(1A6,1Y-2)
    Z6B2 = (Z63-Z62)*B2
    IF (LYM1.NE.0) GO TO 190
    Z6B4 = Z6B3 + Z6B3 - Z6B2
    GO TO 210
190 Z65 = Z(1A6,1Y+1)
    Z6B4 = (Z65-Z64)*B4
    IF (LYM2.NE.0) GO TO 210
    Z6B2 = Z6B3 + Z6B4 - Z6B4
    GO TO 210
200 Z6B2 = Z6B3
    Z6B4 = Z6B3
210 IF (LYM3.LE.0) GO TO 220
    Z6B1 = (Z62-Z(1A6,1Y-3))*B1
    GO TO 230
220 Z6B1 = Z6B2 + Z6B2 - Z6B3
230 IF (LYM1.GE.0) GO TO 240
    Z6B5 = ((1A6,1Y+2)-Z65)*B5
    GO TO 250
240 Z6B5 = Z6B4 + Z6B4 - Z6B5
250 IF (1A6.EU.1) GO TO 170
    AB = 1.0/(AB-AB)
    Z3A5 = (Z63-Z53)*A5
    Z4A5 = (Z64-Z54)*A5
    ZAB52 = (Z6B2-Z5B2)*A5
    ZAB53 = (Z6B3-Z5B3)*A5
    ZAB54 = (Z6B4-Z5B4)*A5
    IF (1A6.EU.2) GO TO 160
    GO TO 260
C ESTIMAREA VALORILOR ZA SI ZAB
C CIND (1A.GE.LA-1).AND.(1A.GE.LC)
260 IF (LX.M2.EU.0) GO TO 270
    Z3A5 = Z3A4 + Z3A4 - Z3A3
    Z4A5 = Z4A4 + Z4A4 - Z4A3
    IF (1X.M1.EU.0) GO TO 290
    ZAB52 = Z4A52 + Z4A52 - Z4352
    ZAB53 = Z4A53 + Z4A53 - Z4353
    ZAB54 = Z4A54 + Z4A54 - Z4354
    GO TO 290
C ESTIMAREA VALORILOR ZA SI ZAB
C CIND (1A.GE.LX-1).AND.(1A.EU.2)
270 Z3A5 = Z3A4
    Z4A5 = Z4A4
    IF (1A.M1.EU.0) GO TO 290
    ZAB52 = Z4A52
    ZAB53 = Z4A53
    ZAB54 = Z4A54
C ESTIMAREA VALORILOR ZA SI ZAB
C CIND 1A.EU.1
280 IF (1A.M1.NE.0) GO TO 290
    Z3A3 = Z3A4 + Z3A4 - Z3A5
    Z3A2 = Z3A3 + Z3A3 - Z3A4
    Z4A3 = Z4A4 + Z4A4 - Z4A5

    Z4A2 = Z4A3 + Z4A3 - Z4A4
    ZAB52 = Z4A52 + Z4A52 - ZAB52
    ZAB53 = Z4A53 + Z4A53 - ZAB53
    ZAB54 = Z4A54 + Z4A54 - ZAB54
    GO TO 300
C DIFERENTIAREA NUMERICA PENTRU DETERMINAREA
C DERIVATELOR PARTIALE ZX,ZY SI ZAY CA MEDII PONDERATE
C ALE DIFERENTELOR DIVIZATE ZA,ZB SI ZAB RESPECTIV
C SALVAREA MECHELOR VALORI CIND 1A.NE.1
290 ZAY33 = ZAY3
    ZAY34 = ZAY4
    ZY33 = ZY43
    ZY34 = ZY44
    ZAY33 = ZAY43
    ZAY34 = ZAY44

```

## VIII. Reprezentarea grafică a curbelor și suprafețelor

```

C CALCULUL NGU
300 DO 350 JY=1,2
    W2 = ABS(7A(4,JY)-ZA(3,JY))
    W3 = ABS(ZA(2,JY)-ZA(1,JY))
    SW = W2 + W3
    IF (SW.EQ.0.0) GO TO 310
    WX2 = W2/SW
    WX3 = W3/SW
    GO TO 320
310 WX2 = 0.5
    WX3 = 0.5
320 ZA(JY) = WX2*7A(2,JY) + WX3*ZA(3,JY)
    W2 = ABS(ZB(JY+3)-7B(JY+2))
    W3 = ABS(ZB(JY+1)-7B(JY))
    SW = W2 + W3
    IF (SW.EQ.0.0) GO TO 330
    WY2 = W2/SW
    WY3 = W3/SW
    GO TO 340
330 WY2 = 0.5
    WY3 = 0.5
340 ZY(JY) = WY2*ZB(1,JY+1) + WY3*ZB(JY+2)
    ZX(JY) = WY2*(WX2*ZAB(1,JY)+WX3*ZAB(2,JY))
    * WY3*(WX2*ZAB(1,JY+1)+WX3*ZAB(2,JY+1))
350 CONTINUE
    IF (IXM1.EQ.0) GO TO 380
C DETERMINAREA COEFICIENTILOR POLINOMULUI
ZX3B3 = (ZX34-ZX33)*B3
ZX4B3 = (ZX44-ZX43)*B3
ZY3A3 = (ZY43-ZY33)*A3
ZY4A3 = (ZY44-ZY34)*A3
A = ZA3B3 - ZX3B3 - ZY3A3 + ZXY33
B = ZX4B3 - ZX3B3 - ZXY43 + ZXY33
C = ZY4A3 - ZY3A3 - ZXY34 + ZXY33
D = ZX44 - ZXY43 - ZX34 + ZXY33
E = A + A - B - C
P02 = (2.0*(Z3B3-ZY33)+Z3B3-ZY34)*B3
P03 = (-2.0*(Z3B3+ZY34+ZY33)*B3)
P12 = (2.0*(ZX3B3-ZXY33)+ZX3B3-ZXY34)*B3
P13 = (-2.0*(ZX3B3+ZY34+ZY33)*B3)
P20 = (2.0*(Z3A3-ZX33)+Z3A3-ZX43)*A3
P22 = (2.0*(ZY3A3-ZXY33)+ZY3A3-ZXY43)*A3
P23 = (-3.0*(A+E)+D)*A3*B3
P30 = (-2.0*(Z3A3+ZX43+ZX33)*A3)
P31 = (-2.0*(ZY3A3+ZY43+ZY33)*A3)
P32 = (-3.0*(E-C)+D)*B3*A3
P33 = (D+E+E)*A3*B3
C CALCULUL POLINOMULUI
ZMIN=10.*30
ZMAX=-7*MIN
DO 370 JY=1,MYP1
    KV = KVV + JY
    VY = V(KV) - YJ
    Q0 = P00 + VY*(P01+VY*(P02+VY*P03))
    Q1 = P10 + VY*(P11+VY*(P12+VY*P13))
    Q2 = P20 + VY*(P21+VY*(P22+VY*P23))
    Q3 = P30 + VY*(P31+VY*(P32+VY*P33))
    DO 360 JX=1,MXP1
        KU = KVV + JX
        UX = U(KU) - XJ
        W(JX,JY) = Q0 + UX*(Q1+UX*(Q2+UX*Q3))
        IF (W(JX,JY).LT.7*MIN) ZMIN=W(JX,JY)
        IF (W(JX,JY).GT.7*MAX) ZMAX=W(JX,JY)
360 CONTINUE
370 CONTINUE
    IF (ZERO(IX,IY).EQ.1) GO TO 372
    WRITE(IUU) IX,IY,ZMIN,ZMAX
    DO 371 JY=1,MYP1
        WRITE(IUU) (W(JX,JY),JX=1,MXP1)
371 KU = KVV + MXU
372 CONTINUE
    KVV = KVV + MYU
390 CONTINUE
C IESIRE NORMALA
RETURN
C IESIRE IN CAZ DE EROR
400 IER=1
    GO TO 520
410 IER=2
    GO TO 520
420 IER=3
    GO TO 520
430 IER=4
    GO TO 520
460 IER=5
    GO TO 520
470 IER=6
    GO TO 520
490 IER=7
    GO TO 520
500 IER=8
    GO TO 520
520 RETURN
END

```



## 8.5.5 Listarea subrutinei ITPLBV

```

0001      SUBROUTINE ITPLBV(LX,LY,X,Y,Z,N,U,V,W,IER)
C INTERPOLARE BIVARIATA
C ACEASTA SUBROUTINA INTERPOLEAZA VALORILE UNEI FUNCTII
C  $Z=Z(X,Y)$  PORȚIND DE LA VALORILE FUNCȚIEI DATE INTR-UN
C CAROIAJ IN PLANUL X-Y SI DE LA UN SET DE PUNCTE DATE
C IN PLAN
C METODA FOLOSESTE O FUNCTIE DEFINITA PE PORTIUNI COMPUSA
C DINTR-UN SET DE POLINOAME BICUBICE IN X SI Y.FIECARE
C FIECARE POLINOM ESTE APLICABIL PE UN DREPTUNGHI AL
C CAROIAJULUI DAT IN PLANUL X-Y
C PARAMETRI DE INTRARE
C L=NUMARUL DE PUNCTE AL CAROIAJULUI DAT PE AXA X
C (VALOAREA MINIMA 2)
C LY=NUMAR DE PUNCTE AL CAROIAJULUI DAT PE AXA Y
C (VALOARE MINIMA 2)
C X=VECTOR DE DIMENSIUNE LX CONTININD COORDONATELE
C PE AXA X ALE CAROIAJULUI DAT (IN ORDINE CRESCATOARE)
C Y=VECTOR DE DIMENSIUNE LY CONTININD COORDONATELE
C PE AXA Y ALE CAROIAJULUI DAT (IN ORDINE CRESCATOARE)
C Z=VECTOR DUNLU DIMENSIONAT DE DIMENSIUNE (LX,LY)
C CONTININD VALORILE FUNCȚIEI (VALORILE Z)
C IN PUNCTELE CAROIAJULUI DAT
C N=NUMARUL DE PUNCTE T CARE INTERPOLAREA
C VALORII Z ESTE DORITA (VALOARE MINIMA 1)
C U=VECTOR DE DIMENSIUNE N CONTININD COORDONATELE
C PE AXA X ALE PUNCTELOR DORITE
C V=VECTOR DE DIMENSIUNE N CONTININD COORDONATELE
C PE AXA Y ALE PUNCTELOR DORITE
C PARAMETRU DE IESIRE
C W=VECTOR DE DIMENSIUNE N CONTININD VALORILE
C INTERPOLATE 7 IN PUNCTELE DORITE
C IER=INDICATOR DE ERORARE IN DATELE DE INTRARE
C CATEVA VARIABLE INTERNE
C ZA=DIFERENTA DIVIZATA A LUI Z IN RAPORT CU X
C ZB=DIFERENTA DIVIZATA A LUI Z IN RAPORT CU Y
C ZAB=DIFERENTA DIVIZATA DE ORDINUL DOI A LUI
C Z IN RAPORT CU X SI Y
C ZX=DERIVATA PARTIALA A LUI Z IN RAPORT CU X
C DERIVATA PARTIALA A LUI Z IN RAPORT CU Y
C ZXY=DERIVATA PARTIALA DE ORDINUL DOI A LUI Z
C IN RAPORT CU X SI Y
C DECLARATII
      DIMENSION X(1), Y(1), Z(30,30), U(1), V(1), W(1)
0002      DIMENSION ZA(5*2), ZB(2*5), ZAB(3,3), ZX(4,4), ZY(4,4),
0003      * ZXY(4,4)
0004      EQUIVALENCE (ZBA1,ZA(1)), (ZBA2,ZA(2)), (ZBA3,ZA(3)),
      * (ZBA4,ZA(4)), (ZBA5,ZA(5)), (ZBA1,ZA(6)), (ZBA2,ZA(7)),
      * (ZBA3,ZA(8)), (ZBA4,ZA(9)), (ZBA5,ZA(10)), (ZB1,ZB(1)),
      * (ZB2,ZB(2)), (ZB3,ZB(3)), (ZB4,ZB(4)), (ZB5,ZB(5)),
      * (ZB1,ZB(6)), (ZB2,ZB(7)), (ZB3,ZB(8)), (ZB4,ZB(9)),
      * (ZB5,ZB(10)), (ZAB2,ZAB(1)), (ZAB3,ZAB(2)),
      * (ZAB2,ZAB(3)), (ZAB3,ZAB(4)), (ZAB3,ZAB(5)),
      * (ZAB3,ZAB(6)), (ZAB4,ZAB(7)), (ZAB4,ZAB(8)),
      * (ZAB4,ZAB(9)), (ZAB3,ZX(6)), (ZAB3,ZX(7)),
      * (ZAB4,ZX(10)), (ZAB4,ZX(11)), (ZY33,ZY(6)),
      * (ZY33,ZY(7)), (ZY34,ZY(10)), (ZY44,ZY(11))

```

```

* (ZXY33,ZXY(6)), (ZY43,ZXY(7)), (ZXY34,ZXY(10)),
* (ZXY44,ZXY(11)), (P00,Z33), (P01,ZY33), (P10,ZX33),
* (P11,ZXY33)
EQUIVALENCE (LX0,ZX(1)), (LXM1,ZX(4)), (LXM2,ZX(13)),
* (LXP1,ZX(16)), (LY0,ZY(1)), (LYM1,ZY(4)), (LYM2,ZY(13)),
* (LYP1,ZY(16)), (IX,ZXY(1)), (IY,ZXY(4)), (IXPV,ZXY(13)),
* (IYPV,ZXY(16)), (IMN,JX), (IMX,JY), (JXM2,JX1),
* (JYM2,JY1), (UK,DX), (VK,DY), (A1,A5,B1,B5,ZX(2),A,W0),
* (A2,ZX(5),B,W1), (A4,ZX(8),C,W2), (B2,ZY(2),D,W3),
* (Y2,ZX(14)), (Y4,ZY(3),B3S0), (Y5,ZX(15),P02),
* (B4,ZY(14),E), (X2,ZX(3),A3S0), (X4,ZX(9)), (X5,ZX(12)),
* (Z23,ZY(5),P03), (Z24,ZY(8),P12), (Z32,ZY(9),P13),
* (Z34,ZY(12),P20), (Z35,ZY(15),P21), (Z42,ZXY(2),P22),
* (Z43,ZXY(5),P23), (Z44,ZXY(3),P30), (Z45,ZXY(8),P31),
* (Z53,ZXY(9),P32), (Z54,ZXY(12),P33), (W2,WY2,W4),
* (W3,WY3,W1,W5), (WX2,ZXY(14)), (WX3,ZXY(15))

```

C CALCULE PRELIMINARE

C SETAREA ANUMITOR PARAMETRI DE INTRARE LA VALOAREA

C VARIABILELOR LOCALE

```

LX0 = LX
LXM1 = LX0 - 1
LXM2 = LXM1 - 1
LXP1 = LX0 + 1
LY0 = LY
LYM1 = LY0 - 1
LYM2 = LYM1 - 1
LYP1 = LY0 + 1
N0 = N

```

C VERIFICAREA ERORILOR

```

IF (LXM2.LT.0) GO TO 710
IF (LYM2.LT.0) GO TO 720
IF (N0.LT.1) GO TO 730
DO 10 IX=2,LX0
IF (X(IX-1)-X(IX)) 10, 740, 750

```

10 CONTINUE

```

DO 20 IY=2,IY0
IF (Y(IY-1)-Y(IY)) 20, 770, 780

```

20 CONTINUE

C SETAREA INITIALA A VALORILOR ANTERIOARE ALE LUI

C IX SI IY

```

IXPV = 0
IYPV = 0

```

C CICLUL DO PRINCIPAL

```

DO 700 K=1,N0
UK = U(K)
VK = V(K)

```

C SECVENTA DE LOCALIZARE A PUNCTULUI DORIT

C PENTRU GASIREA VALORII IX PENTRU CARE

C (U(K).GE.X(IX-1)).AND.(U(K).LT.X(IX))

```

IF (LXM2.EQ.0) GO TO 80
IF (UK.GE.X(LX0)) GO TO 70
IF (UK.LT.X(1)) GO TO 60

```

```

IMN = 2
IMX = LX0

```

30 IX = (IMN+IMX)/2

```

IF (UK.GE.X(IX)) GO TO 40

```

```

      IX = IX
      GO TO 50
40  IMN = IX + 1
50  IF (IMX.GT.IMN) GO TO 30
      IX = IMX
      GO TO 90
60  IX = 1
      GO TO 90
70  IX = LXP1
      GO TO 90
80  IX = 2
C GASINEA VALORII IY PENTRU CARE
C (V(K).GE.Y(IY-1)).AND.(V(K).LT.Y(IY))
90  IF (LYM2.EQ.0) GO TO 150
      IF (VK.GE.Y(LY0)) GO TO 140
      IF (VK.LT.Y(1)) GO TO 130
      IMN = 2
      IMX = LY0
100  IY = (IMN+IMX)/2
      IF (VK.GE.Y(IY)) GO TO 110
      IMX = IY
      GO TO 120
110  IMN = IY + 1
120  IF (IMX.GT.IMN) GO TO 100
      IY = IMX
      GO TO 160
130  IY = 1
      GO TO 160
140  IY = LYP1
      GO TO 160
150  IY = 2
C SE VERIFICA DACA PUNCTUL DORIT ESTE IN ACELASI
C DREPTUNGHI CA SI PUNCTUL PRECEDENT, DACA DA SE SARE
C LA CALCULUL POLINOMIILOR
160  IF (IX.EQ.IXPV .AND. IY.EQ.IYPV) GO TO 690
      IXPV = IX
      IYPV = IY
C SECVENTA DE GASIRE A VALORILOR X,Y SI Z NECESARE
C CALCULAREA VALORILOR ZA,ZB SI ZAB SI ESTIMARIILE LOR.
C CIND ESTE NECESAR
      JX = IX
      IF (JX.EQ.1) JX = 2
      IF (JX.EQ.LXP1) JX = LX0
      JY = IY
      IF (JY.EQ.1) JY = 2
      IF (JY.EQ.LYP1) JY = LY0
      JXM2 = JX - 2
      JXNL = JX - LX0
      JYM2 = JY - 2
      JYNL = JY - LY0
C IN ZONA DE INTERES T.E. IN DREPTUNGHIUL CARE CONTINE
C PUNCTUL DORIT
      X3 = X(JX-1)
      X4 = X(JX)
      A3 = 1.0/(X4-X3)
      Y3 = Y(JY-1)

```

```

Y4 = Y(JY)
r3 = 1.0/(Y4-r3)
r33 = Z(JX-1,JY-1)
r43 = Z(JX+JY-1)
r34 = Z(JX-1,JY)
r44 = Z(JX+JY)
r3A3 = (r43-r33)*r3
r4A3 = (r44-r34)*r3
r3A3 = (r34-r33)*r3
r4A3 = (r44-r43)*r3
rA3B3 = (r4A3-r3A3)*A3
C IN DIRECTIONA X
IF (LXM2.EQ.0) GO TO 230
IF (JXM2.EQ.0) GO TO 170
x2 = x(JX-2)
r2 = 1.0/(x3-x2)
r23 = Z(JX-2,JY-1)
r24 = Z(JX-2,JY)
r3A2 = (r33-r23)*r2
r4A2 = (r34-r24)*r2
IF (JXML.EQ.0) GO TO 180
170 x5 = x(JX+1)
r4 = 1.0/(x5-x4)
r53 = Z(JX+1,JY-1)
r54 = Z(JX+1,JY)
r3A4 = (r53-r43)*r4
r4A4 = (r54-r44)*r4
IF (JXM2.NE.0) GO TO 190
r3A2 = r3A3 + r3A4 - r4A4
r4A2 = r4A3 + r4A4 - r4A4
GO TO 190
180 r3A4 = r3A3 + r3A4 - r3A2
r4A4 = r4A3 + r4A4 - r4A2
190 rA2B3 = (r4A2-r3A2)*B3
rA4B3 = (r4A4-r3A4)*B3
IF (JX.LE.3) GO TO 200
r1 = 1.0/(x2-x(JX-3))
r3A1 = (r23-Z(JX-3,JY-1))*r1
r4A1 = (r24-Z(JX-3,JY))*r1
GO TO 210
200 r3A1 = r3A2 + r3A4 - r3A3
r4A1 = r4A2 + r4A4 - r4A3
210 IF (JX.GE.LXM1) GO TO 220
r5 = 1.0/(x(JX+2)-x5)
r3A5 = (r(JX+2,JY-1)-r53)*r5
r4A5 = (r(JX+2,JY)-r54)*r5
GO TO 240
220 r3A5 = r3A4 + r3A4 - r3A3
r4A5 = r4A4 + r4A4 - r4A3
GO TO 240
230 r3A2 = r3A3
r4A2 = r4A3
GO TO 180
C IN DIRECTIONA Y
240 IF (LYM2.EQ.0) GO TO 310
IF (JYM2.EQ.0) GO TO 250

```

```

Y2 = Y(JY-2)
B2 = 1.0/(Y3-Y2)
Z32 = Z(JX-1,JY-2)
Z42 = Z(JX,JY-2)
Z3B2 = (Z33-Z32)*.2
Z4B2 = (Z43-Z42)*.2
IF (JYML.F0.0) GO TO 260
250 Y5 = Y(JY+1)
B4 = 1.0/(Y5-Y4)
Z35 = Z(JX-1,JY+1)
Z45 = Z(JX,JY+1)
Z3B4 = (Z35-Z34)*.4
Z4B4 = (Z45-Z44)*.4
IF (JYM2.NE.0) GO TO 270
Z3B2 = Z3B3 + Z3B3 - Z3B4
Z4B2 = Z4B3 + Z4B3 - Z4B4
GO TO 270
260 Z3B4 = Z3B3 + Z3B3 - Z3B2
Z4B4 = Z4B3 + Z4B3 - Z4B2
270 ZA3B2 = (Z4B2-Z3B2)*.3
ZA3B4 = (Z4B4-Z3B4)*.3
IF (JY.LE.3) GO TO 280
B1 = 1.0/(Y2-Y(JY-3))
Z3B1 = (Z32-Z(JX-1,JY-3))*B1
Z4B1 = (Z42-Z(JX,JY-3))*B1
GO TO 290
280 Z3B1 = Z3B2 + Z3B2 - Z3B3
Z4B1 = Z4B2 + Z4B2 - Z4B3
290 IF (JY.GE.LYMI) GO TO 300
B5 = 1.0/(Y(JY+2)-Y5)
Z3B5 = (Z(JX-1,JY+2)-Z35)*B5
Z4B5 = (Z(JX,JY+2)-Z45)*B5
GO TO 320
300 Z3B5 = Z3B4 + Z3B4 - Z3B3
Z4B5 = Z4B4 + Z4B4 - Z4B3
GO TO 320
310 Z3B2 = Z3B3
Z4B2 = Z4B3
GO TO 260
C IN DIRECTIILE DIAGONALE
320 IF (LXM2.F0.0) GO TO 400
IF (LYM2.F0.0) GO TO 410
IF (JXML.F0.0) GO TO 350
IF (JYM2.F0.0) GO TO 330
ZA4B2 = ((Z53-Z(JX+1,JY-2))*B2-Z4B2)*A4
IF (JYML.F0.0) GO TO 340
330 ZA4B4 = ((Z(JX+1,JY+1)-Z54))*B4-Z4B4)*A4
IF (JYM2.NE.0) GO TO 380
ZA4B2 = ZA4B3 + ZA4B3 - ZA4B4
GO TO 380
340 ZA4B4 = ZA4B3 + ZA4B3 - ZA4B2
GO TO 380
350 IF (JYM2.F0.0) GO TO 360
ZA2B2 = (Z3B2-(Z23-Z(JX-2,JY-2))*B2)*A2
IF (JYML.F0.0) GO TO 370
360 ZA2B4 = (Z3B4-(Z(JX-2,JY+1)-Z24))*B4)*A2

```

```

IF (JYM2.NE.0) GO TO 390
7A2B2 = 7A2B3 + 7A2B3 - ZA2B4
GO TO 390
370 7A2B4 = ZA2B3 + 7A2B3 - ZA2B2
GO TO 390
380 IF (JXM2.NE.0) GO TO 350
ZA2B2 = 7A3B2 + 7A3B2 - ZA4B2
ZA2B4 = 7A3B4 + 7A3B4 - ZA4B4
GO TO 420
390 IF (JXML.NE.0) GO TO 420
ZA4B2 = 7A3B2 + 7A3B2 - ZA2B2
ZA4B4 = 7A3B4 + 7A3B4 - ZA2B4
GO TO 420
400 7A2B2 = ZA3B2
ZA4B2 = ZA3B2
7A2B4 = 7A3B4
7A4B4 = 7A3B4
GO TO 420
410 ZA2B2 = ZA2B3
ZA2B4 = ZA2B3
7A4B2 = ZA4B3
7A4B4 = ZA4B3
C DIFERENTIERE NUMERICĂ PENTRU DETERMINAREA DERIVATELOR
C PARTIALE ZX, ZY SI ZX Y CA MEDII PONDERATE ALE DIFERENTELOR
C DIVIZATE ZA, ZB SI ZAH RESPECTIV
420 DO 480 JY=2,3
DO 470 JX=2,3
W2 = ABS(ZA(JX+2.,JY-1))-ZA(JX+1.,JY-1))
W3 = ABS(ZA(JX.,JY-1))-ZA(JX-1.,JY-1))
SW = W2 + W3
IF (SW.EQ.0.0) GO TO 430
WX2 = W2/SW
WX3 = W3/SW
GO TO 440
430 WX2 = 0.5
WX3 = 0.5
440 ZX(JX.,JY) = WX2*7A(JX.,JY-1) + WX3*ZA(JX+1.,JY-1)
W2 = ABS(ZB(JX-1.,JY+2))-ZB(JX-1.,JY+1))
W3 = ABS(ZB(JX-1.,JY))-ZB(JX-1.,JY-1))
SW = W2 + W3
IF (SW.EQ.0.0) GO TO 450
WY2 = W2/SW
WY3 = W3/SW
GO TO 460
450 WY2 = 0.5
WY3 = 0.5
460 ZY(JX.,JY) = WY2*7B(JX-1.,JY) + WY3*ZB(JX-1.,JY+1)
ZXY(JX.,JY) =
* WY2*(WX2*ZAB(JX-1.,JY-1)+WX3*ZAB(JX.,JY-1)) +
* WY3*(WX2*ZAB(JX-1.,JY)+WX3*ZAB(JX.,JY))
470 CONTINUE
480 CONTINUE
C CIND (U(K).LT.X(1)).OR.(U(K).GT.X(LX))
IF (IX.EQ.LXP1) GO TO 530
IF (IX.NE.1) GO TO 590
W2 = A4*(3.0*A3+A4)

```

```

W1 = 2.0*A3*(A3-A4) + W2
DO 500 JY=2,3
ZX(1,JY) = (W1*Za(1,JY-1)+W2*ZA(2,JY-1))/(W1+W2)
ZXY(1,JY) = ZXY(2,JY) + ZXY(2,JY) - ZXY(3,JY)
ZY(1,JY) = ZY(2,JY) + ZY(2,JY) - ZY(3,JY)
DO 490 JX1=2,3
JX = 5 - JX1
ZA(JX,JY) = ZX(JX-1,JY)
ZY(JX,JY) = ZY(JY-1,JY)
ZXY(JX,JY) = ZXY(JX-1,JY)
490 CONTINUE
500 CONTINUE
X3 = X3 - 1.0/A4
Z33 = Z33 - Z3A0/A4
DO 510 JY=1,5
ZB(2,JY) = ZB(1,JY)
510 CONTINUE
DO 520 JY=2,4
ZB(1,JY) = ZB(1,JY) - ZAB(1,JY-1)/A4
520 CONTINUE
A3 = A4
JX = 1
GO TO 570
530 W4 = A2*(3.0*A3+A2)
W5 = 2.0*A3*(A3-A2) + W4
DO 550 JY=2,3
ZX(4,JY) = (W4*Za(4,JY-1)+W5*ZA(5,JY-1))/(W4+W5)
ZY(4,JY) = ZY(3,JY) + ZY(3,JY) - ZY(2,JY)
ZXY(4,JY) = ZXY(2,JY) + ZXY(3,JY) - ZXY(2,JY)
DO 540 JX=2,3
ZX(JX,JY) = ZX(JX+1,JY)
ZY(JX,JY) = ZY(JX+1,JY)
ZXY(JX,JY) = ZXY(JX+1,JY)
540 CONTINUE
550 CONTINUE
X3 = X4
Z33 = Z43
DO 560 JY=1,5
ZB(1,JY) = ZB(2,JY)
560 CONTINUE
A3 = A2
JX = 3
570 ZA(3,1) = ZA(JX+1,1)
DO 580 JY=1,3
ZAB(2,JY) = ZAB(JX,JY)
580 CONTINUE
C CIND (V(K).LT.Y(1)).OR.(V(K).GT.Y(LY))
590 IF (IY.EQ.LYP1) GO TO 630
IF (IY.NE.1) GO TO 600
W2 = B4*(3.0*B3+B4)
W1 = 2.0*B3*(B3-B4) + W2
DO 620 JX=2,3
IF (JX.EQ.3 .AND. IX.NE.LXP1) GO TO 600
IF (JX.EQ.2 .AND. IX.EQ.1) GO TO 600
ZY(JX,1) = (W1*Zb(JX-1,1)+W2*ZB(JX-1,2))/(W1+W2)
ZX(JX,1) = ZX(JX,2) + ZX(JX,2) - ZX(JX,3)

```

```

      ZXY(JX+1) = ZXY(JX+2) + ZXY(JX+2) - ZXY(JX+3)
600 GO TO 610 JY1=2,3
      JY = 5 - JY1
      ZY(JX,JY) = ZY(JX,JY-1)
      ZX(JX,JY) = ZX(JX,JY-1)
      ZXY(JX,JY) = ZXY(JX,JY-1)
610 CONTINUE
620 CONTINUE
      Y3 = Y3 - 1.0/64
      Z33 = Z33 - 7362/64
      Z3A3 = Z3A3 - 2A3H2/64
      Z3B3 = Z3B2
      Z3B3 = Z3B2
      B3 = B4
      GO TO 670
630 K4 = 62*(J40+J3+H2)
      W5 = 2.0*H3*(H3-H2) + W4
      GO ONO JX=2,3
      IF (JX.EQ.3 .AND. IX.EQ.LXP1) GO TO 640
      IF (JX.EQ.2 .AND. IX.EQ.1) GO TO 640
      ZY(JX+4) = (W4*Z4(JX-1,4)+W5*Z4(JX-1,5))/(W4+W5)
      ZX(JX+4) = ZX(JX+4) + ZX(JX+3) - ZX(JX+2)
      ZXY(JX+4) = ZXY(JX+3) + ZXY(JX+3) - ZXY(JX+2)
640 GO 650 JY=2,3
      ZY(JX,JY) = ZY(JX,JY+1)
      ZX(JX,JY) = ZX(JX,JY+1)
      ZXY(JX,JY) = ZXY(JX,JY+1)
650 CONTINUE
660 CONTINUE
      Y3 = Y4
      Z33 = Z33 + Z3B3/3
      Z3A3 = Z3A3+7A3B3/63
      Z3B3 = Z3B4
      Z3B3 = 7A3B4
      B3 = B2
670 IF (IX.NF.1 .AND. IX.NF.LXP1) GO TO 680
      JX = IX/LXP1 + 2
      JX1 = 5 - JX
      JY = IY/LYP1 + 2
      JY1 = 5 - JY
      ZX(JX,JY) = ZX(JX1,JY) + ZX(JX,JY1) - ZX(JX1,JY1)
      ZY(JX,JY) = ZY(JX1,JY) + ZY(JX,JY1) - ZY(JX1,JY1)
      ZXY(JX,JY) = ZXY(JX1,JY) + ZXY(JX,JY1) - ZXY(JX1,JY1)
C IFTERMINAREA COEFICIENTILOR POLINOMULUI
680 ZA3B3 = (ZX34-ZA33)*B3
      ZX4B3 = (ZX44-ZX43)*B3
      ZY3A3 = (ZY43-ZY33)*A3
      ZY4A3 = (ZY44-ZY34)*A3
      A = ZA3B3 - ZX3B3 - ZY3A3 + ZXY33
      B = ZX4B3 - ZX3B3 - ZXY43 + ZXY33
      C = ZY4A3 - ZY3A3 - ZXY34 + ZXY33
      D = ZXY44 - ZX4B3 - ZXY34 + ZXY33
      E = A + A - B - C
      A350 = A3*A3
      B350 = B4*B3
      P02 = (2.0*(Z3B3-ZY33)+Z3B3-ZY34)*B3

```



```

P03 = (-2.0*Z3B3+7Y34+ZY33)*B3S0
P12 = (2.0*(ZX3A3-ZXY33)+ZX3B3-ZXY34)*B3
P13 = (-2.0*ZX3B3+7XY34+ZY33)*B3S0
P20 = (2.0*(Z3A3-7X33)+Z3A3-ZX43)*A3
P21 = (2.0*(ZY3A3-7XY33)+ZY3A3-ZXY43)*A3
P22 = (3.0*(A+E)+D)*A3*B3
P23 = (-3.0*E-B-D)*A3*B3S0
P30 = (-2.0*Z3A3+7X43+ZX33)*A3S0
P31 = (-2.0*ZY3A3+7XY43+ZY33)*A3S0
P32 = (-3.0*E-C-D)*B3*A3S0
P33 = (D+E+F)*A3S0*B3S0
C CALCULUL POLINOMULUI
690 DY = VK - Y3
    G0 = P00 + DY*(P01+DY*(P02+DY*P03))
    G1 = P10 + DY*(P11+DY*(P12+DY*P13))
    G2 = P20 + DY*(P21+DY*(P22+DY*P23))
    G3 = P30 + DY*(P31+DY*(P32+DY*P33))
    DX = UK - X3
    W(K) = G0 + DX*(G1+DX*(G2+DX*G3))
700 CONTINUE
C IESIRE NORMALA
    RETURN
C IESIRE IN CAZ DE ERORARE
710   IER=1
    GO TO 800
720   IER=2
    GO TO 800
730   IER=3
    GO TO 800
740   IER=4
    GO TO 800
750   IER=5
    GO TO 800
770   IER=6
    GO TO 800
780   IER=7
    RETURN
800   END

```

## 8.5.6. Program interpolare bivariată și montarea suprafețelor netede

```

      LOGICAL*81 F1S1(15),F1S2(15)
      DIMENSION X(91),Y(91),Z(3),X1(9),Y1(9),X2(9),Y2(9),X3(9),Y3(9),X4(9),Y4(9),X5(9),Y5(9)
      *PLOT(256,4),XPL(3)(256),YPL(3)(256)
      *EQU(15)(50),XN(1),YCN(1),Y1(1),Y2(1),Y3(1),Y4(1),Y5(1)
      *OBJECT(1),B1(1),C1(1),YPLUT(1),B1(1),C1(1)
      F(2),Z(2)=(-71+Z(2)-Z1)
      N71A=EPSZ(2),EPSVZ
100  TYPE 1
1   FORMAT(0,80,' BE INTRODUCERE NIVEL : 1.0')
      ACCEPT 2,111F
      IF(110,FO,3) GOTO 2000
2   FORMAT(5I2)
      TYPE 3
3   FORMAT(0,80,' FISIE DE INTRODUCERE : 1.0')
      ACCEPT 4,68I,F1S1
4   FORMAT(0,150I)
      TYPE 5
5   FORMAT(0,80,' FISIE DE IESIRE : 1.8')
      ACCEPT 4,80F,F1S2
      IF(111,FO,2) GOTO 1000
      CALL ASSIGN(4,F1S),*P1)
      CALL FDBSET(4,*P1)
      READ(4,2)IX,IY,*X,*Y,*Z
      IF(10)1001,1002,1003
1002 READ(4,6)(X(I),I)=1,I=1,IY
      READ(4,6)(Y(I),I)=1,I=1,IY
      GO 7 IY=1,IY
7   READ(4,6)(Z(I),I),I=1,IY
6   FORMAT(16F5,2)
      CALL CLOSE(4)
      CALL ASSIGN(4,F1S2,*P2)
      CALL FDBSET(4,*P2)
      WRITE(4)IX,IY,*X,*Y,*Z
      CALL SECSET(4,IX,IY,*X,*Y,*Z,*X,*Y,*Z)
      CALL CLOSE(4)
      GOTO 100
1000 CALL ASSIGN(4,F1S1,*P1)
      CALL FDBSET(4,*P1)
      CALL ASSIGN(3,F1S2,*P2)
      CALL FDBSET(3,*P2)
      CALL PLOTS(0,0,0,3)
2000 REWRITE 4
      READ(4)IX,IY,*X,*Y,*Z
      IF(10)1003,1001,1002
1003 IF(IY=(IY-1)*X+1)
      IF(IY=(IY-1)*X+1)
      READ(4)(U(1),I=1,IF(IY)
      READ(4)(V(1),I=1,IF(IY)
      TYPE 1004
1004 FORMAT(0,16,' INTRODUCERE VALORILE NIVEL : 1.8')
      ACCEPT 1005,0
1005 FORMAT(FA,2)
      IY=)
      IY=IY-1

```

```

LY=1+LY-1
MX=1+MX+1
MY=1+MY+1
L=1
DO 10 J=1+LY-1
  Y=1
  DO 102 I=1+LX-1
    SF=AF(I) (S2(K),K=1,MXP1)
    YL=V(J,Y)
    DO 10 J=1+MY
      YL=YL+V(J,Y+1)
      STY=YL+Y
      DO 10 K=1,MXP1
        W1(K1)=2(K1)
        W2(K) (S2(K),K=1,MXP1)
        F1=VF(1,Y)
        F2=V(1)
        YF=V(1,X)
        F3=VF(1,Y)
        XCU=XCU+1
        STX=XCU-XC
        F4=F1
        F5=F2
        F1=W1(I1+1)
        F2=W2(I1+1)
        ICCI=0
        IF (C,CF,F1) ICCI=ICCI+1
        IF (C,SF,F2) ICCI=ICCI+2
        IF (C,GF,F3) ICCI=ICCI+4
        IF (C,GF,F4) ICCI=7-ICCI
        IF (ICCI) 11,8,11
11      GOTO (13,20,30,40,50,60,70) *ICCI
10      PLOT(I,*)=XC+F(F4,F1,*)*STX
        PLOT(I,*)=YL+F(F1,F2,*)*STY
        PLOT(I,*)=YLU
        GOTO 80
20      PLOT(I,*)=XC+F(F3,F2,*)*STX
        PLOT(I,*)=YL+F(F1,F2,*)*STY
        PLOT(I,*)=YLU
        GOTO 80
30      PLOT(I,*)=XC+F(F3,F2,*)*STX
        PLOT(I,*)=YL+F(F4,F3,*)*STX
        PLOT(I,*)=YLU
        GOTO 80
40      PLOT(I,*)=XC+F(F3,F2,*)*STX
        PLOT(I,*)=YL+F(F4,F3,*)*STY
        PLOT(I,*)=YLU
        GOTO 80
50      PLOT(I,*)=XC
        PLOT(I,*)=YLU
        GOTO 80

```

```

50  PLOTP(LM+1,1)=XC+F(F4,F1,w)*STX
    PLOTP(LM+3)=XC+F(F3,F2,w)*STX
    PLOTP(LM+2)=YL+F(F4,F3,w)*STY
    PLOTP(LM+1,4)=YL+F(F1,F2,w)*STY
    PLOTP(LM,1)=XC
    PLOTP(LM,4)=YLU
    PLOTP(LM+1,2)=YL
    PLOTP(LM+1,3)=XCU
    LM=LM+1
    GOTO 80
60  PLOTP(LM,2)=YL+F(F4,F3,w)*STY
    PLOTP(LM,4)=YL+F(F1,F2,w)*STY
    PLOTP(LM,1)=XC
    PLOTP(LM,3)=XCU
    GOTO 80
70  PLOTP(LM,2)=YL+F(F4,F3,w)*STY
    PLOTP(LM,1)=XC
    PLOTP(LM,4)=YL
    PLOTP(LM,3)=XC+F(F4,F1,w)*STX
80  LM=LM+1
8   CONTINUE
    IX=IX+PX
102  CONTINUE
    JY=JY+PY
101  CONTINUE
    LM=LM-1
554  WRITE(3,554)((PLOTP(I,J),J=1,4),I=1,*)
    FORMAT(4F15,6)
    INDI=1
    INDU=1
    INDCI=0
    XMIN=U(1)
    XMAX=U(1F1NX)
    YMIN=V(1)
    YMAX=V(1F1NY)
150  DO 96 I=INDI,LM
    IF (PLOTP(I,1).EQ.XMIN)GOTO 90
    IF (PLOTP(I,2).EQ.YMIN)GOTO 90
    IF (PLOTP(I,2).EQ.YMAX)GOTO 90
    IF (PLOTP(I,3).EQ.XMAX)GOTO 91
    IF (PLOTP(I,4).EQ.YMIN)GOTO 91
    IF (PLOTP(I,4).EQ.YMAX)GOTO 51
96   CONTINUE
12   INDCI=1
    GOTO 97
90   CALL MOV(PLOTP,INDU,I,0)
    GOTO 97
91   CALL MOV(PLOTP,INDU,I,2)
97   IND=INDU
    INDU=INDU+1
    DO 92 I=INDU,LM
    IF (ABS(PLOTP(IND,3)-PLOTP(I,1)).LT.EPS.AND.ABS(PLOTP(IND,4)-
    PLOTP(I,2)).LT.EPS) GOTO 90
    IF (ABS(PLOTP(IND,3)-PLOTP(I,3)).LT.EPS.AND.

```

```

*ABS(PLOTP(INO,4)-PLOTP(I,4)).LT.FPS) GOTO 91
92  CONTINUE
    CALL FBATA(ZPLOT,YPLOT,INOT,IND)
    CALL PLOT(PLOTP(INO,3),PLOTP(INO,4),I)
    IAI=I*100
    IF (IAI).GE.1M)GOTO 93
    IF (IAI).GE.1)GOTO 97
    GOTO 150
93  TYPE 1010
1010 FORMAT('NOT SURE LEVEL OF CALCULATED ? (I=NO I=AA) ? 0,3)
    ACCEPT 2,311
    IF (111)2000,1012,2000
1012  CALL PLOT(0.,0.,490)
    CALL CLOSE(3)
1001  CALL CLOSE(4)
    GOTO 100
3900  STOP
      END

```

### 8.5.7. Subrutina MOV

```

SUBROUTINE MOV (PLOTP,IAO,IND2,I)
DIMENSION PLOTP(256,4)
X1=PLOTP(IND2,I+1)
X2=PLOTP(IND2,I+2)
X3=PLOTP(IND2,I+3)
X4=PLOTP(IND2,I+4)
PLOTP(IND2,1)=PLOTP(IND1,1)
PLOTP(IND2,2)=PLOTP(IND1,2)
PLOTP(IND2,3)=PLOTP(IND1,3)
PLOTP(IND2,4)=PLOTP(IND1,4)
PLOTP(IND1,1)=X1
PLOTP(IND1,2)=X2
PLOTP(IND1,3)=X3
PLOTP(IND1,4)=X4
RETURN
END

```

## 8.6. Trasarea curbelor de nivel

O primă aplicație pentru programul de interpolare a unei suprafețe a fost realizarea unui program care utilizând drept intrare datele rezultate din interpolare să traseze curbele pentru care  $z = ct$ .

S-a pornit de la ideea că din modelul rezultat din interpolare se pot obține puncte suficient de „dese” astfel încât o interpolare ulterioară de tip liniar pentru nivelul  $z$  dat să dea un rezultat satisfăcător.

În acest caz algoritmul este următorul:

Fie 4 puncte care alcătuiesc un dreptunghi și pentru care se cunosc coordonatele  $x, y, z$ , (fig. 8.13).

Se caută pe fiecare din segmentele (12), (13), (24) și (34) puncte astfel încât nivelul  $z$  cerut să fie situat în interiorul segmentului considerând pentru  $z$  o interpolare liniară.

Cum în interiorul dreptunghiului nu putem afla alte puncte decât rezolvînd ecuația suprafeței atunci considerăm că în interiorul dreptunghiului nivelul  $z$  este reprezentat de segmentul (sau segmentele) ce unesc punctele de pe frontieră aflate anterior. În general pe frontieră se pot găsi numai 2 puncte astfel încît pentru marea majoritate a cazurilor segmentul este unic determinat.

Un dezavantaj al metodei constă în faptul că segmentele din care sînt alcătuite curbele de nivel sînt dispartate (nu apar în continuare din cauza modului de baleiere a suprafeței) ceea ce face ca în primul rînd curbele să nu fie plotate iar în al doilea rînd mișcările mesei de desen să fie mai multe decît necesare și dezordonate.

Pentru evitarea acestui inconvenient s-a introdus în algoritmul general o secvență de „așezare” a segmentelor obținute unul în continuarea celuilalt astfel încît curbele să se deseneze continuu. În schimb această secvență este o mare consumatoare de timp bazîndu-se pe un algoritm de ordonare care îndeobște este inefficient pentru șiruri mari de date de intrare.

Rezultatele obținute cu ajutorul programului curbelor de nivel s-au dovedit a fi suficient de precise mai ales în cazul în care s-au folosit factori mai mari de îndesire a punctelor originale. În ce privește utilizarea programului ea este evidentă programul fiind interactiv.

Singura cerință este existența unui fișier în care să fie memorate rezultatele interpolării anterioare utilizînd metoda bidimensională.

### 8.6.1. Formele tablourilor pentru datele de intrare și pentru datele de ieșire

#### INTERPOLAREA BIVARIATĂ ȘI MONTAREA SUPRAFETELOR NETEDE

##### ● DATE DE INTRARE

IX nr. de pcte.	X(IX) caroiaj	Funcția Z(IX, IY)								
		IY = (nr. de puncte)								
		1	2	3	4	5	6	7	8	9
		Y(IY) = (caroiaj)								
		0,0	5,0	10,0	15,0	20,0	25,0	30,0	35,0	40,0
1	0,0									
2	5,0									
3	10,0									

##### ● DATE DE IEȘIRE

KX	U(KX)	Funcția îndesita W(KX, KY)				
		KY = (nr. de puncte)				
		1	2	3	4	5...
		V(KY) = (caroiaj, pas îndesit)				
		0,0	2,5	5,0	7,5	10,0...
1	0,0					
2	2,5					
3	5,0					

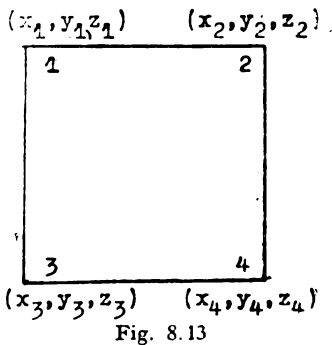


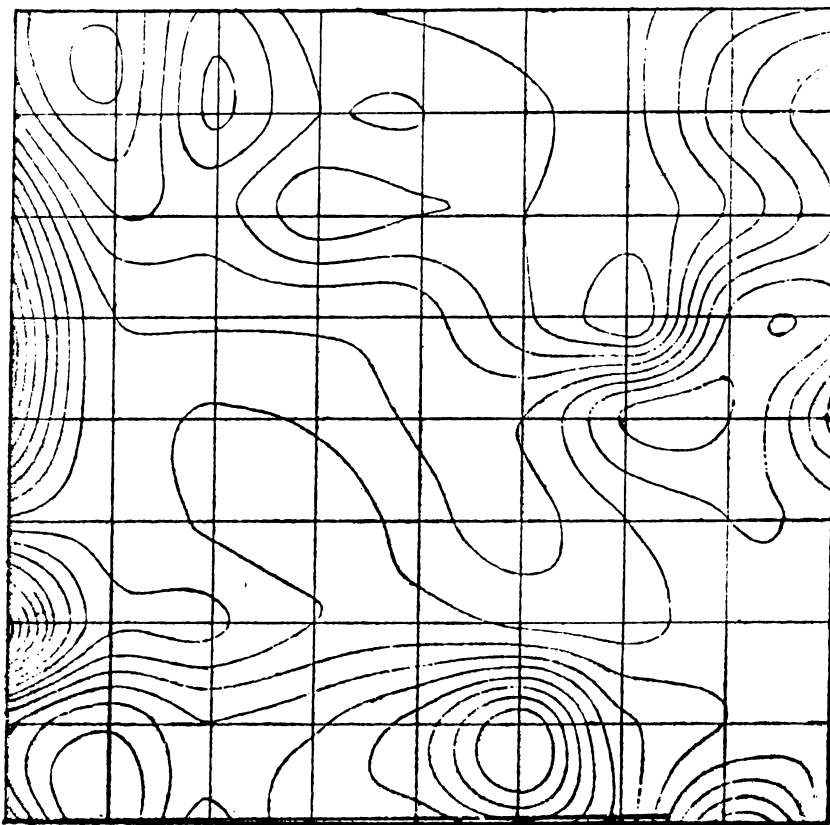
Fig. 8.13

Legătura posibilă dintre interpolarea bivariată și vizualizarea (perspectivă) a funcțiilor de două variabile

Algoritmul și programul pentru vizualizarea (perspectivă) a funcțiilor de două variabile cu studiul porțiunilor de suprafață vizibile este perfect aplicabil și în cazul suprafețelor netede determinate prin interpolarea bivariată.

În felul acesta există posibilitatea completării integrale a imaginii, pe care ne-am putut-o face despre suprafața în cauză. Pe lângă determinarea curbelor de nivel de cote dorite avem și perspectivele reliefului din oricare punct de vedere.

### 8.7. Aplicații ale interpolării bivariante și montării suprafețelor netede bazată pe proceduri locale [figurile 8.14 – 8.19]



Exemplu de determinare și de trasare continuă a curbelor de nivel într-o interpolare bivariată pe un carouaj dat

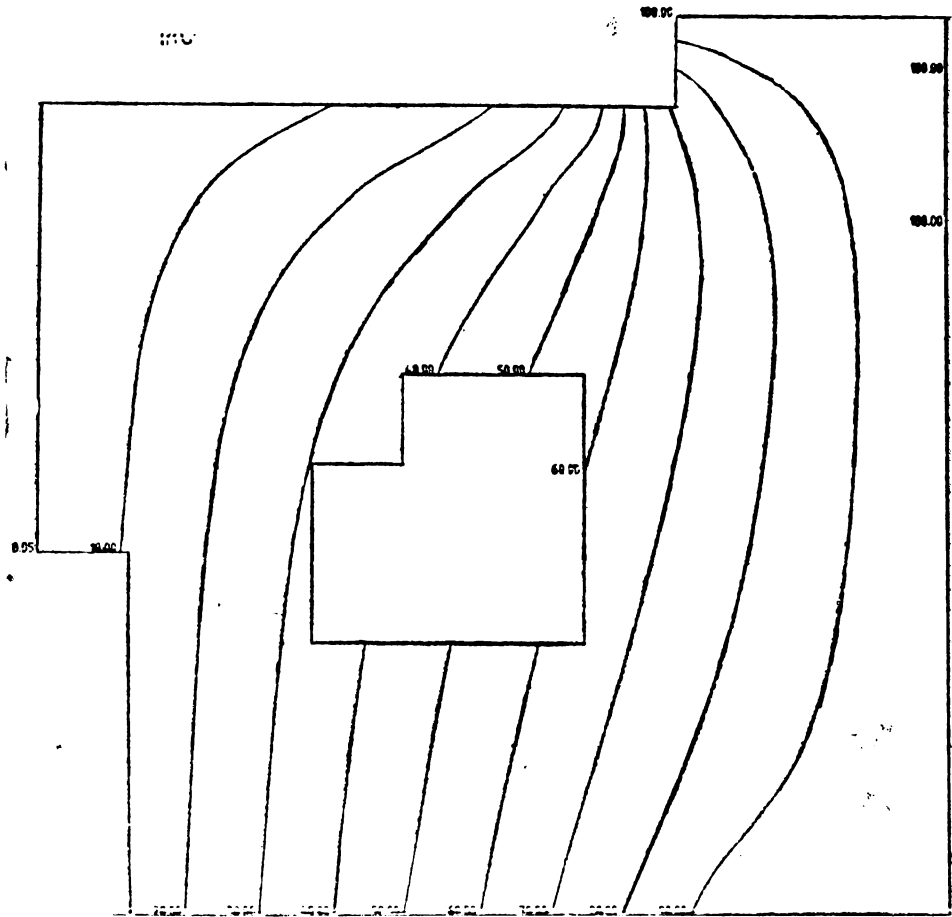


Fig. 8.15

Din acest exemplu rezultă faptul că programul prezintă și facilitatea de a „fila” curbele de nivel în cazul în care interiorul domeniului sau la periferia sa se întâlnesc obstacole mai mici sau mai mari și de diferite forme. Deasemenea există posibilitatea scrierii sub diferite forme a cotelor pe aceste trasări automate ale curbelor de nivel.



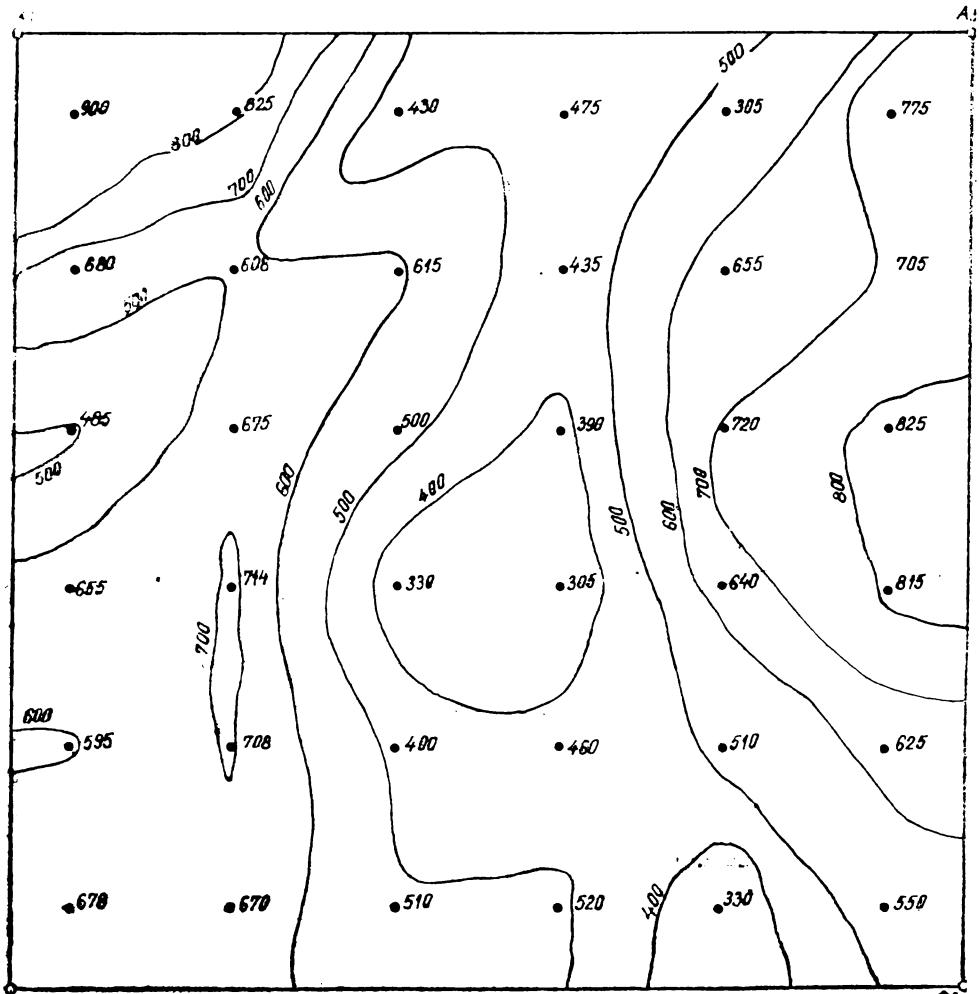


Fig. 8.16

Distribuția apei  $m^3/ha$  pentru aspersoare A1, A2, A3, A4 situate la distanța de  $18 \times 18m$ . Determinarea și trasarea manuală a curbelor (figura 8.16) în comparație cu determinarea și trasarea automată printr-o interpolare bivariată (figura 8.17). Diferența este net în favoarea calculatorului.

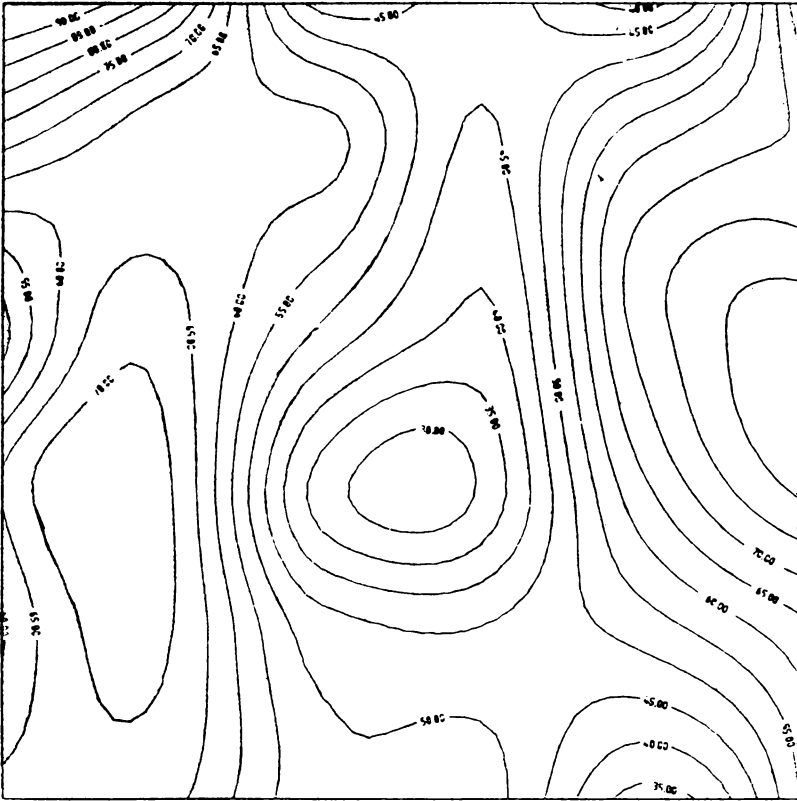


Fig. 8.17

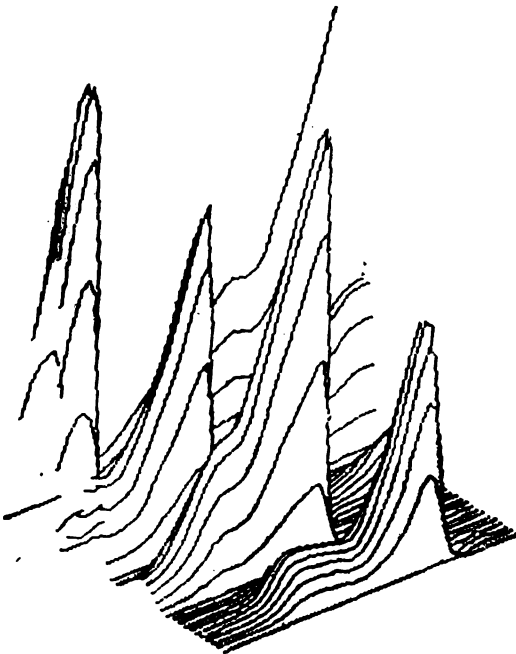


Fig. 8.18

Perspectiva reliefului (fig. 8.18) și o secțiune cu un plan vertical prin relieful respectiv (fig. 8.19)

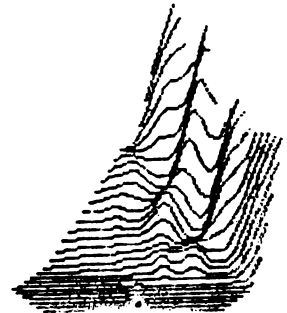


Fig. 8.19

## 9.1. Introducere. Generalități.

Un domeniu important în realizarea reprezentării grafice prin calculator se ocupă cu aproximarea grafică sau vizuală a curbelor și suprafețelor.

În mod frecvent acest fapt duce la o problemă de fixare a datelor: se dă un număr de puncte prin care trebuie să se potrivească o curbă sau o suprafață.

De aici înainte, deci, este de rezolvat o problemă clasică de interpolare.

Interpolarea este un caz special al unei probleme mai generale de aproximare.

Se dă o funcție  $f(t)$  care va fi aproximată de o sumă finită

$$g(t) = \sum_{i=1}^n c_i \psi_i(t)$$

a unei funcții mai simple  $\psi_i(t)$  astfel încît să satisfacă un anumit set de restricții pentru funcția  $g(t)$ . Deoarece trebuie să fie determinate  $n$  constante necunoscute  $c_1, c_2, \dots, c_n$ , este necesar să fie specificate cel puțin condiții restrictive pentru funcția  $g(t)$ . De obicei aceste restricții sînt impuse funcției  $g(t)$  astfel încît ea să fie o „bună” aproximare pentru funcția  $f(t)$ .

Dacă funcția  $g(t)$  este continuă pe intervalul de aproximare  $[a, b]$  atunci desigur se alege de asemenea funcțiile  $\psi_1(t), \dots, \psi_n(t)$  continue pe  $[a, b]$ .

Dacă introducem spațiul liniar  $C[a, b]$  ca set al tuturor funcțiilor continue pe  $[a, b]$ , putem admite că

$$\psi_i(t) \in C[a, b]$$

astfel, o bună aproximare  $g(t)$  poate fi obținută pentru funcția  $f(t)$  dacă:

1.  $\|f - g\| = \min$
2. Se asigură o soluție unică pentru setul de coeficienți  $c_i$ .

Aici  $\|\dots\|$  înseamnă norma lui Tchebyscheff pe  $C[a, b]$ . Se știe că norma Tchebyscheff a funcției  $f(t) \in C[a, b]$  este definită ca funcția corect evaluată,

$$\|f\| = \max_{a < t < b} |f(t)|$$

În general  $C^k[a, b]$  înseamnă setul de funcții continue al primelor  $k$  derivate (funcții diferentiabile) pe intervalul  $[a, b]$ . În teoria aproximării restricțiile ce vor fi impuse funcției  $g(t)$  vor fi foarte des următoarele:

1. Constrîngeri de interpolare:

$$g(t_i) = f(t_i), \quad \forall i \in [1: n]$$

De exemplu, ambele funcții vor avea aceleași valori într-un număr distinct de puncte în  $[a, b]$

2. Amestec de interpolări și restricții:

a)  $g(t_i) = f(t_i), \quad i \in [1: k < n]$

b)  $g'(t_1) = f'(t_1)$  și  $g'(t_k) = f'(t_k)$

c)  $g(t) \in C^2$ , adică  $g(t)$  este **dubiu** diferentiabilă.

3. Restricții ortogonale:

$$(f - g, \psi_i) = \int_a^b [f(t) - g(t)] \psi_i(t) dt = 0,$$

adică constantele  $c_1, c_2, \dots, c_n$  sînt astfel alese încît să satisfacă relația de mai sus, funcțiile  $\psi_1, \psi_2, \dots, \psi_n$  fiind ortogonale.

4. Restricții variaționale:

$$\|f - g\| = \min \{ \|f - h\| \mid h \in \text{distanța } (\psi_1, \dots, \psi_n) \}$$

adică constantele  $c_1, c_2, \dots, c_n$  sînt astfel alese încît să se obțină minimul din toate funcțiile  $\|f - h\|$  din setul tuturor combinațiilor lineare posibile  $h = c_1\psi_1 + c_2\psi_2 + \dots + c_n\psi_n$ .

În grafica pe calculator este necesar să ne ocupăm în mod deosebit de anumite calități care sînt caracteristice în condițiile specifice unei curbe sau suprafețe. Unii numesc această calitate ca fiind „*proprietatea formei*“.

Astfel reprezentarea formei poate necesita anumite modificări ale aproximării obișnuite din teoria aproximării. În general, formele în problemele de proiectare aplicată la calculator, nu pot fi identificate cu funcțiile simple în sensul  $y = f(x)$ . Aceasta se explică prin faptul că forma celor mai multe obiecte de construcții este independentă intrinsec de sistemul de coordonate. De exemplu, dacă trebuie să potrivim o curbă sau o suprafață printr-un set de puncte alese inițial, atunci factorul important în determinarea formei obiectului este relația dintre aceste puncte și nu dintre puncte și un anumit sistem de coordonate ales în mod arbitrar.

De aceea în multe aplicații apare ca o condiție esențială ca alegerea unui sistem de coordonate să nu afecteze forma. Mai mult, formele obiectelor de construcții pot avea tangente verticale. Dacă o astfel de formă a fost reprezentată de o funcție obișnuită de tipul  $y = f(x)$ , tangentele verticale vor exclude o aproximare prin funcțiile analitice (de exemplu polinoamele). Curbele și suprafețele, foarte frecvent, nu sînt plane, sau sînt închise sau deschise și de aceea nu pot fi reprezentate de o funcție obișnuită.

Pentru toate aceste motive, reprezentarea dominantă a formelor în proiectarea asistată de calculator se face în forma parametrică, nu de o funcție  $y = f(x)$ , ci de un set de două funcții  $x = x(t)$ , și  $y = y(t)$ .

Deci putem spune că un punct pe o astfel de curbă este reprezentat prin vectorul

$$\bar{P}_c = [x(t) \ y(t)]$$

De asemenea un punct situat pe o curbă din spațiu este dat de vectorul

$$\bar{P}_c[x(t) \ y(t) \ z(t)]$$

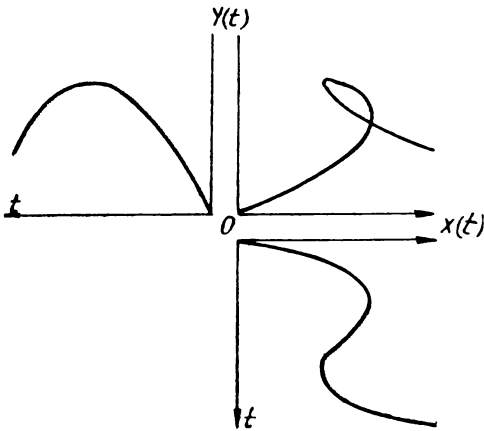


Fig. 9.1

figura 9.1. Curba rezultată nu poate fi reprezentată de o funcție obișnuită  $y = f(x)$ .

Cu ajutorul curbelor și suprafețelor care sînt reprezentate sub forma parametrică, problema interpolării sau aproximării unui astfel de obiect este redusă la problema determinării prin interpolare sau aproximare a componentelor de reprezentare ale vectorului obiectului.

O aproximare generală de producere a unei curbe sau suprafețe prin hardware duce exact la problema clasică de aproximare și anume, la aproximarea curbelor și suprafețelor arbitrare printr-o sumă de funcții mai simple care pot fi produse prin generatori de funcții hardware. O clasă de funcții care sînt foarte potrivite pentru producerea prin hardware, sînt polinoamele.

## 9.2. Determinarea curbei de interpolare care trece prin $n + 1$ puncte date

Polinomul de gradul 2 determină parabola  $y = ax^2 + bx + c$  a căreia axă este paralelă cu axa  $oy$ .

Virful parabolei are coordonatele  $-b/2a$  și  $(4ac - b^2)/(4a)$

Parabola care trece prin două puncte de coordonate  $(0, 0)$  și  $(1, 0)$  și are ca virf punctul  $V\left(\frac{1}{2}, V\right)$  are ca ecuație

$$y = 4v(1 - x)x$$

În grafica pe calculator se utilizează în mod frecvent curbele definite prin polinoame sau curbe care trebuie să treacă printr-un anumit număr de puncte date. O astfel de curbă are ecuația de forma:

$$y = a_0 + a_1x + \dots + a_nx^n = \sum_{i=0}^n a_i x^i$$

Un punct situat pe o suprafață este reprezentat prin vectorul

$$\vec{P}_s = [x(u, v) \ y(u, v) \ z(u, v)]$$

O astfel de reprezentare parametrică a formelor este cea mai potrivită descriere a modului în care curbele sînt trasate de un plotter sau vizualizate pe un ecran.

În acest caz cele două funcții  $x(t)$  și  $y(t)$  se aplică în sensul unei „funcții de dirijare” către sistemul Servo al plotterului sau către sistemul de deflexie al tubului cu raze catodice, trasînd curba respectivă.

Acest mod de producere (trasare) al curbelor poate fi ilustrat în

în care  $a_0, a_1, \dots, a_n$  sînt constante. Dacă  $a_n \neq 0$  relația de mai sus exprimă curba algebrică de grad  $n$ .

Putem considera doi vectori

$$\bar{a} = [a_0, a_1, \dots, a_n] \text{ și}$$

$$\bar{x} = [1, x, \dots, x^n]$$

deci putem scrie:  $y = \bar{a}\bar{x}$

Această ecuație conține  $n + 1$  coeficienți care sînt componentele vectorului  $\bar{a}$ .

Să considerăm punctele  $A_i(x_i, y_i)$  și să găsim condiția ca curba să treacă prin aceste puncte.

Dacă notăm vectorul  $\bar{x}_i = [1, x_i, \dots, x_i^n]$  avem

$$y_i = \bar{a}\bar{x}_i$$

Condiția ca curba să treacă prin cele  $n + 1$  puncte  $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$  conduce la rezolvarea sistemului de  $n + 1$  ecuații liniare cu  $n + 1$  necunoscute care în scriere matriceală este

$$\bar{y} = \bar{a}\bar{X} \text{ unde } \bar{y} = [y_0, y_1, \dots, y_n] \text{ iar}$$

$$\bar{X} = \begin{bmatrix} 1, 1, \dots, 1 \\ x_0, x_1, \dots, x_n \\ \dots \dots \dots \\ x_0^n, x_1^n, \dots, x_n^n \end{bmatrix}$$

Dacă  $x_i \neq x_j$  pentru  $i \neq j, i, j = 0, 1, \dots, n$  există matricea inversă  $\bar{X}^{-1}$  și rezolvarea sistemului conduce la

$$\bar{a} = \bar{y}\bar{X}^{-1}$$

și prin urmare ecuația explicită a curbei de interpolare care trece prin punctele date  $(x_i, y_i) i = 0, 1, \dots, n$  este

$$y = \bar{y}\bar{X}^{-1}\bar{X}$$

### 9.2.1. Aplicație

Să se determine cubica care trece prin următoarele patru puncte

$$(0,1); \left(\frac{1}{3}, 0\right); \left(\frac{2}{3}, 0\right) \text{ și } (1,0) \text{ (fig. 9.2).}$$

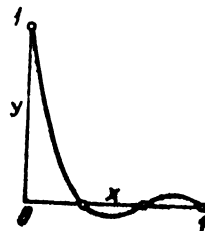


Fig. 9.2

Ecuția explicită a curbei este

$$y = [1,0,0,0] \bar{X}^{-1} [1, x, x^2, x^3]^T \text{ (matricea transpusă)}$$

$$y = 1 - \frac{11}{2}x + 9x^2 - \frac{9}{2}x^3. \text{ Matricele } \bar{X} \text{ și } \bar{X}^{-1} \text{ utilizate}$$

sint

$$\bar{X} = \begin{bmatrix} 1, & 1, & 1, & 1 \\ 0, & \frac{1}{3}, & \frac{2}{3}, & 1 \\ 0, & \frac{1}{9}, & \frac{4}{9}, & 1 \\ 0 & \frac{1}{27}, & \frac{8}{27}, & 1 \end{bmatrix}; \quad \bar{X}^{-1} = \begin{bmatrix} 1, & -\frac{11}{2}, & 9, & -\frac{9}{2} \\ 0, & 9, & -\frac{45}{2}, & \frac{27}{2} \\ 0, & -\frac{9}{2}, & 18, & -\frac{27}{2} \\ 0, & 1, & -\frac{9}{2}, & \frac{9}{2} \end{bmatrix}$$

### 9.3. Curba cubică parametrică spațială

Cubica parametrică spațială are o deosebită utilizare în studiul curbelor și suprafețelor în grafica pe calculator. Ea este exprimată vectorial prin ecuația:

$$\bar{P} = \bar{A} + \bar{B}t + \bar{C}t^2 + \bar{D}t^3 \quad t \in [0, 1]$$

sau poate fi exprimată parametric prin funcțiile cubice (fig. 9.3 a și b):

$$x(t) = a_x + b_x t + c_x t^2 + d_x t^3$$

$$y(t) = a_y + b_y t + c_y t^2 + d_y t^3$$

$$z(t) = a_z + b_z t + c_z t^2 + d_z t^3 \quad \text{unde } t \in [0, 1]$$

Cubica parametrică este definită prin 12 coeficienți.

#### 9.3.1. Aplicație

Să se determine cubica spațială definită prin patru puncte  $A(0, 0, 0)$ ,  $B(1, 0, 1)$ ,  $C(1, 1, 0)$ ,  $D(0, 1, 0)$  știind că valorile parametrului sînt  $0; \frac{1}{3}; \frac{2}{3}$  și  $1$

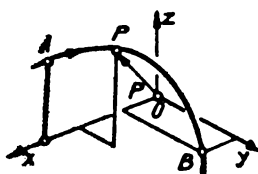


Fig. 9.3 a

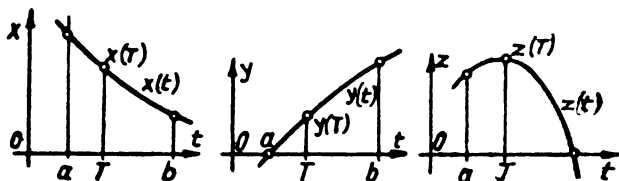


Fig. 9.3 b

Este necesară cunoașterea coordonatelor celor patru puncte  $A_1, A_2, A_3$  și  $A_4$  precum și valorile parametrului în intervalul  $[0,1]$  deci:

$$0 \leq t_1 < t_2 < t_3 < t_4 \leq 1$$

Coefficienții pot fi determinați rezolvînd sistemul de patru ecuații liniare al funcțiilor cubice parametriche

$$x_i = a_x + b_x t_i + c_x t_i^2 + d_x t_i^3 \quad i = 1, 2, 3, 4$$

și analog pentru  $y_i$  și  $z_i$ .

Astfel coeficienții  $a_x, b_x, c_x, d_x$  se obțin rezolvînd sistemul:

$$0 = a_x$$

$$1 = \frac{b_x}{3} + \frac{c_x}{9} + \frac{d_x}{27}$$

$$1 = \frac{2b_x}{3} + \frac{4c_x}{9} + \frac{8d_x}{7}$$

$$0 = b_x + c_x + d_x$$

și similar obținem mai departe opt coeficienți.

Valorile rezultate în final sînt:

$$a_x = 0; \quad b_x = \frac{9}{2}; \quad c_x = -\frac{9}{2}; \quad d_x = 0$$

$$a_y = 0; \quad b_y = -\frac{7}{2}; \quad c_y = \frac{27}{2}; \quad d_y = -9$$

$$a_z = 0; \quad b_z = 9; \quad c_z = -\frac{45}{2}; \quad d_z = \frac{27}{2}$$

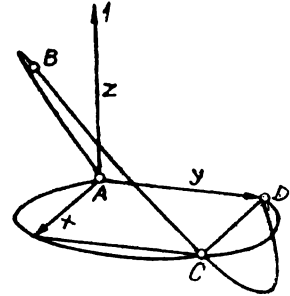


Fig. 9.4

Proiecția axonometrică a curbei și proiecția axonometrică a proiecției sale orizontale sînt redată în figura 9.4.

## 9.4. Metode clasice de interpolare

### Interpolarea Lagrange și interpolarea Hermite

#### 9.4.1. Interpolarea Lagrange

Interpolările *Lagrange* și *Hermite* aparțin ambele clasei de interpolări de polinoame.

Polinoamele Lagrange se situează printre cele mai simple polinoame de interpolare.

Fiind date  $n + 1$  numere reale distincte  $x_0 < x_1 < \dots < x_n$  și un al doilea set de  $n + 1$  numere reale corespondente  $y_i, i \in [0: n]$  se definește polinomul Lagrange de gradul  $n$  asociat celor două seturi  $\{x_i\}$  și  $\{y_i\}$  ca fiind polinomul  $P(x)$  de gradul  $n$  care rezolvă problema de interpolare

$$P(x_i) = y_i, \quad \forall i \in [0: n]$$

Numerele  $x_i$  sînt numite „noduri” iar numerele  $y_i$  sînt numite „restricții” ale funcției de interpolare. Aceste polinoame există întotdeauna și sînt unice. În special, dacă  $f(x)$  este o funcție definită în  $x_0, x_1, x_2, \dots, x_n$  astfel încît

$$f(x_i) = y_i, \quad \forall i \in [0: n]$$

interpolarea Lagrange de gradul  $n$  a funcției  $f(x)$  cu nodurile în  $x_i, i \in [0: n]$  este polinomul

$$P_n(x) = \sum_{i=0}^n f(x_i) L_{i,n}(x)$$

cu

$$L_{i,n} = \frac{(x - x_0) \dots (x - x_{i-1}) (x - x_{i+1}) \dots (x - x_n)}{(x_i - x_0) \dots (x_i - x_{i-1}) (x_i - x_{i+1}) \dots (x_i - x_n)}$$



Polinomul  $P_n(x)$  poate fi scris așa dar și sub forma

$$P_n(x) = \sum_{i=0}^n f(x_i) \frac{\prod_{\substack{j=0 \\ j \neq i}}^n (x - x_j)}{\prod_{\substack{j=0 \\ j \neq i}}^n (x_i - x_j)}$$

Din definiția lui  $L_{i,n+1}(x)$  rezultă

$$L_{i,n}(x_i) = \delta_{ij}, \quad i, j \in [0:n]$$

unde  $\delta_{ij}$  este delta Kronecker

$$\delta_{ij} = \begin{cases} 1 & \text{dacă } i = j \\ 0 & \text{altfel} \end{cases}$$

Ca cel mai simplu exemplu posibil alegem  $n = 1$ . Astfel avem

$$P_1(x) = \frac{x - x_1}{x_0 - x_1} \cdot Y_0 + \frac{x - x_0}{x_1 - x_0} Y_1 = Y_0 + (y_1 - y_0) \frac{x_0 - x}{x_0 - x_1}$$

Rezultatul este ecuația segmentului de linie dreaptă care leagă  $y_0 = f(x_0)$  și  $y_1 = f(x_1)$ . Astfel interpolarea poligonului ce trece printr-un număr de puncte date, atât de obișnuit în grafica pe calculator, este o interpolare de gradul 1 a polinomului Lagrange pe porțiuni.

Desigur polinoamele Lagrange nu sînt singurele polinoame care rezolvă problema de interpolare  $P(x_i) = f(x_i)$ ,  $\forall i \in [0:n]$ .

De fapt, există infinit de multe polinoame care pot fi folosite, dar polinomul Lagrange este polinomul unic de gradul  $n$  de rezolvare a problemei. Oricum, interpolarea Lagrange peste tot intervalul  $[x_0, x_n]$  are un dezavantaj serios: deoarece gradul  $n$  al polinomului Lagrange corespunde numărului de noduri (plus 1) orice încercare de mărire a exactității de aproximare prin sporirea numărului de noduri, produce de asemenea o creștere a gradului polinomului. Totuși, polinoamele de grad mai mare, să zicem 5 sau mai mare vor cauza ondulații în curbă (*fenomenul Runge-Méray*).

O soluție posibilă în problemele instabilității și ondulațiilor, care sînt de obicei neacceptate în scopurile constructive este să se subîmpartă intervalul  $[x_0, x_n]$  într-un număr de subintervale și să se strîngă la un loc un număr de polinoame Lagrange de grad mic, fiecare aproximînd funcția peste unul din subintervale. Orice aproximare arbitrară în sensul noimei Tchebyscheff poate fi realizată.

O soluție de asemenea posibilă este să se utilizeze polinoamele *Hermite* în locul polinoamelor Lagrange.

### 9.4.2. Interpolarea Hermite

Interpolarea Hermite corespunde unui polinom pentru o funcție  $f$ , astfel încît la un număr dat de noduri, nu numai funcția  $f$  este interpolată ci de asemenea și un număr dat de derivate consecutive ale funcției  $f$ . Acesta este exact un polinom de gradul 3

$$P_3(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0$$

care satisface restricțiile:

$$P(a) = f(a), \quad P'(a) = f'(a)$$

$$P(b) = f(b), \quad P'(b) = f'(b)$$

Acest polinom este numit „*polinomul de interpolare cubică Hermite*” al funcției  $f$ . Fiind date o funcție  $f(t)$  și  $n + 1$  noduri  $a = t_0 < t_1 < \dots < t_n = b$  divizând intervalul închis  $[a, b]$  în  $n$  subintervale închise  $[t_{i-1}, t_i]$ ,  $i \in [1 : n]$ , putem constitui polinoamele cubice Hermite  $P_i(t)$  care satisface restricțiile:

$$P_i(t_i - 1) = f(t_i - 1), \quad P'_i(t_i - 1) = f'(t_i - 1)$$

$$P_i(t_i) = f(t_i), \quad P'_i(t_i) = f'(t_i),$$

și formează ulterior polinomul cubic pe porțiuni

$$s(t) = P_i(t), \quad \forall_i: t \in [t_{i-1}, t_i].$$

În fiecare din intervalele  $[t_{i-1}, t_i]$ , funcția  $s(t)$  este un polinom și astfel este infinit diferențiabilă. În toate nodurile interioare  $t_i$ ,  $i \in [1 : n - 1]$  avem

$$f(t_i) = P_i(t_i) = P_{i+1}(t_i) = s(t_i) \quad \text{și} \quad P'_{i+1}(t_i) = s'(t_i) = P'_i(t_i) = f'(t_i)$$

De aceea funcția  $s(t)$  este cel puțin odată diferențiabilă în nodurile interioare, sau

$$s(t) \in C^1[a, b]$$

Dacă se dorește să se asigure o netezire mai mare, trebuie să se schimbe pe porțiuni polinoamele Hermite, la un grad mai mare de 3. Interpolarea unei funcții prin polinoame cubice Hermite pe porțiuni se face direct. Ea este dată de relația

$$s(t) = \sum_{i=0}^n f(t_i) I_{i0}(t) + \sum_{i=0}^n f'(t_i) I_{i1}(t)$$

cu

$$I_{i0}(t) = \begin{cases} \frac{(t - t_{i-1})^2}{(t_i - t_{i-1})^3} [2(t_i - t) + (t_i - t_{i-1})] & \text{dacă } t \in [t_{i-1}, t_i] \\ \frac{(t_{i+1} - t)^2}{(t_{i+1} - t_i)^3} [2(t - t_i) + (t_{i+1} - t_i)] & \text{dacă } t \in [t_i, t_{i+1}] \\ 0 & \text{altfel} \end{cases}$$

$$I_{i1}(t) = \begin{cases} \frac{(t - t_{i-1})^2 (t - t_i)}{(t_i - t_{i-1})^2} & \text{dacă } t \in [t_{i-1}, t_i] \\ \frac{(t - t_{i+1})^2 (t - t_i)}{(t_{i+1} - t_i)^2} & \text{dacă } t \in [t_i, t_{i+1}] \\ 0 & \text{altfel} \end{cases}$$

$$I_{00}(t) = \begin{cases} \frac{(t_1 - t)^2}{(t_1 - t_0)^3} [2(t - t_0) + (t_1 - t_0)] & \text{dacă } t \in [t_0, t_1] \\ 0 & \text{altfel} \end{cases}$$

$$I_{n_0}(t) = \begin{cases} \frac{(t - t_{n-1})^2}{(t_n - t_{n-1})^3} [2(t_n - t) + (t_n - t_{n-1})] & \text{dacă } t \in [t_{n-1}, t_n] \\ 0 & \text{altfel} \end{cases}$$

$$I_{01}(t) = \begin{cases} \frac{(t - t_1)^2 (t - t_0)}{(t_1 - t_0)^3} & \text{dacă } t \in [t_0, t_1] \\ 0 & \text{altfel} \end{cases}$$

$$I_{n1}(t) = \begin{cases} \frac{(t - t_{n-1})^2 (t - t_n)}{(t_n - t_{n-1})^3} & \text{dacă } t \in [t_{n-1}, t_n] \\ 0 & \text{altfel} \end{cases}$$

Aceste funcții au proprietățile

$$\begin{cases} I_{i_0}(t) = \delta_{ij} \\ I'_{i_0}(t) = 0, \quad i, j \in [0: n] \end{cases}$$

și

$$\begin{cases} I_{i1}(t) = 0, \quad i, j \in [0: n] \\ I'_{i1}(t) = \delta_{ij} \end{cases}$$

### 9.4.3. Aproximarea COONS

Problema care s-ar putea pune, este ce se poate face dacă derivatele funcției  $f(t)$  în nodurile interioare sînt necunoscute.

Există mai multe moduri de preîntîmpinare a acestei dificultăți. Să considerăm, de exemplu, următoarea interpolare Hermite, cunoscută ca „*aproximarea lui COONS*”

Astfel un caz deosebit de interpolare cubică este aproximarea prin polinoame cubice, raționale, introdusă de Coons.

Această metodă și-a găsit o anumită notorietate în grafica computer. Vom prezenta această metodă în forma corespunzătoare pentru aplicațiile de grafică pe calculator.

Considerăm un punct pe o curbă din spațiu dat prin coordonate omogene, adică prin vectorul

$$\bar{P}(t) = [wx(t) \quad wy(t) \quad wz(t) \quad w(t)]$$

Funcțiile  $wx(t)$ ,  $wy(t)$ ,  $wz(t)$  și  $w(t)$  vor fi parametrizate prin polinoamele cubice

$$\begin{aligned} wx(t) &= a_3 t^3 + a_2 t^2 + a_1 t + a_0 \\ wy(t) &= b_3 t^3 + b_2 t^2 + b_1 t + b_0 \\ wz(t) &= c_3 t^3 + c_2 t^2 + c_1 t + c_0 \\ w(t) &= d_3 t^3 + d_2 t^2 + d_1 t + d_0 \end{aligned}$$

Coordonatele simple

$$x = \frac{wx}{w}, \quad y = \frac{wy}{w}, \quad z = \frac{wz}{w}$$

vor fi date sub forma polinoamelor cubice raționale. Cu aceste relații putem scrie:

$$\bar{P}(t) = [t^3 t^2 t 1] \cdot \begin{bmatrix} a_3 & b_3 & c_3 & d_3 \\ a_2 & b_2 & c_2 & d_2 \\ a_1 & b_1 & c_1 & d_1 \\ a_0 & b_0 & c_0 & d_0 \end{bmatrix} = [t^3 t^2 t 1] \cdot A$$

Prima derivată a curbei în punctul  $P$  este

$$\bar{P}'(t) = [3t^2 2t 1 0] \cdot A$$

iar a doua derivată este

$$\bar{P}''(t) = [6t 2 0 0] \cdot A$$

Avantajul reprezentării parametrice este acela că putem alege orice distanță arbitrară a valorilor parametrului pe curbă.

De aici, pentru simplitatea calculului se aranjează ca parametrul  $t$  să primească valori de la 0 la 1. Apare astfel problema de a se determina coeficienții matriciei  $A$  astfel încît funcția  $\bar{P}(t)$  să satisfacă următoarele restricții:

$$\begin{aligned} \bar{P}(t)|_{t=0} &= p_0 \quad \text{și} \quad \bar{P}'(t)|_{t=0} = p'_1 \\ \bar{P}(t)|_{t=1} &= p_1 \quad \text{și} \quad \bar{P}'(t)|_{t=1} = p'_0 \end{aligned}$$

Avem

$$\begin{bmatrix} p_0 \\ p_1 \\ p'_0 \\ p'_1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \cdot A$$

Rezolvînd această ecuație matricială pentru  $A$  obținem

$$A = M \cdot \begin{bmatrix} p_0 \\ p_1 \\ p'_0 \\ p'_1 \end{bmatrix}$$

cu

$$M = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Matricea  $M$  este denumită uneori „matricea magică”. Din ecuațiile de mai sus obținem:

$$\bar{P}(t) = [2t^3 - 3t^2 + 1; -2t^3 + 3t^2; t^3 - 2t^2 + t; t^3 - t^2] \cdot \begin{bmatrix} p_0 \\ p_1 \\ p'_0 \\ p'_1 \end{bmatrix}$$

Pe de altă parte, interpolarea cubică Hermite a funcției

$$f(t), t \in [0, 1] \text{ cu restricțiile}$$

$$f(0) = p_0; f'(0) = p'_0; f(1) = p_1 \text{ și } f'(1) = p'_1$$

este

$$S(t) = I_{00}p_0 + I_{10}p_1 + I_{01}p'_0 + I_{11}p'_1$$

Comparația ecuațiilor confirmă faptul că curba  $\bar{P}(t)$  este interpolarea Hermite pentru cazul  $n = 1$ ,  $t_0 = 0$ ,  $t_n = 1$ . Problema potrivirii unei curbe netede prin  $n + 1$  seturi de puncte ordonate date  $(p_0, p_1, \dots, p_n)$  poate fi realizată prin punerea cap la cap a acestor așa numite funcții „spline” cubice așa cum vom vedea mai departe.

Deoarece în punctele interioare, de obicei numai valorile sînt definite și nu și pantele, cele patru elemente date pentru determinarea coeficienților pot fi folosite pentru a satisface restricțiile pentru două funcții spline adiacente  $\bar{P}_i$  și  $\bar{P}_j$  după cum urmează

$$\begin{aligned} \bar{P}_i(1) &= \bar{P}_j(0) = p_{j(0)} \\ \bar{P}'_i(1) &= \bar{P}'_j(0) \\ \bar{P}''_i(1) &= \bar{P}''_j(0) \end{aligned}$$

Se obține astfel o aproximare  $\bar{P}(t) \in C^2$ , cu specificarea valorilor și a vectorilor tangenți în cele două puncte de capăt ale curbei.

#### 9.4.4. Interpolarea AKIMOV

Interpolarea Akimov este caracterizată prin utilizarea *curbelor plane Ferguson*. Vectorii tangenți în punctele de aplicație ale curbei de interpolare se stabilesc din configurația punctelor de aplicație cu ajutorul relației

$$\left| \frac{X_{i-1}C_i}{C_iA_i} \right| = \left| \frac{X_{i+1}D_i}{D_iB_i} \right|, i = 2, n - 2$$

Deci tangenta  $t_i$  în punctul de aplicație  $x_i$  se determină astfel încît  $(x_{i-1}A_iC_i)$  și  $(x_{i+1}B_iD_i)$  să aibă valoarea absolută identică (fig.9.5).

Lungimea vectorului tangent este egală cu aceea a corzii  $x_i x_{i+1}$ . În figura 9.6 este reprezentată curba obținută prin interpolarea Akimov (mai gros) și pentru aceleași puncte de sprijin (aplicație) este desenată (mai slab) curba obținută prin interpolarea Lagrange a funcției

## 9.5. Funcția B-Spline și interpolarea cu B-spline

Din cauza proprietăților lor matematice simple, polinoamele au fost foarte frecvent folosite pentru aproximarea altor funcții. Polinoamele, în general, tind să prezinte ondulații nedorite atunci cînd trec prin mai mult de cîteva puncte, însă aceste ondulații nu pot deranja atît timp cît sînt de amplitudine mică.

Curbele trase cu ajutorul unei curbe *spline* sau *French* sînt mult mai netede.

De aceea, este foarte avantajos să existe funcții care combină simplitatea polinoamelor cu o netezire proprie.

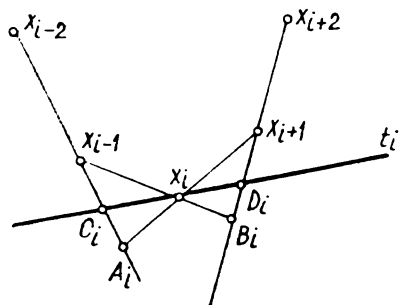


Fig. 9.5

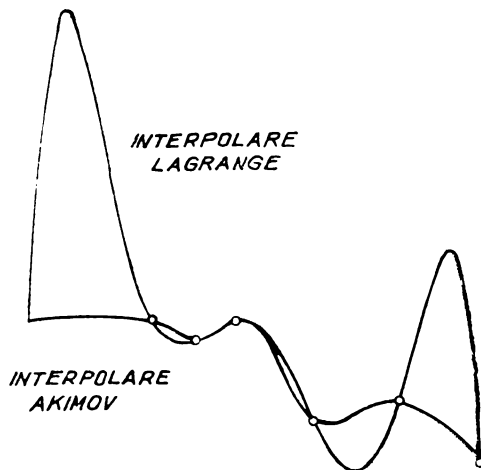


Fig. 9.6

Funcțiile spline au fost studiate intensiv în ultimele două decenii. Restrîngem discuțiile la proprietățile cele mai importante ale funcțiilor polinomiale spline, „funcțiile B-spline”.

### 9.5.1. Funcția spline de gradul $m$

Se consideră un șir de numere reale  $x_0 < x_1 < \dots < x_n$ . O funcție spline  $S(x)$  de gradul  $m$  cu nodurile  $x_0, x_1, \dots, x_n$  este o funcție definită pe întreaga axă reală, avînd următoarele două proprietăți:

- 1) În fiecare interval  $(x_i, x_{i+1})$  pt.  $i = 0, 1, \dots, n$ , cu  $x_0 = -\infty$  și  $x_{n+1} = +\infty$ , funcția  $S(x)$  este dată de un anumit polinom de gradul  $m$  sau mai mic.
- 2) Funcția  $S(x)$  și derivatele sale de ordinul  $1, 2, \dots, m-1$  sînt continue pe întregul interval.

Astfel, o funcție spline este o funcție polinomială pe porțiuni (de exemplu funcția Coons discutată în (9.4.3) este o funcție spline). Pentru  $m = 0$ , condiția (2) nu este valabilă, deoarece o funcție spline de gradul 0 este o funcție pas.

O funcție spline de gradul 1 este un poligon. Pentru  $m > 0$ , o funcție spline de gradul  $m$  poate fi tot atît de bine definită ca o funcție în  $C^{m-1}$ , care se obține ca un întreg nedeterminat de ordinul  $m$  al funcției pas.

În general, funcția  $S(x)$  este dată de polinoame diferite în intervalele alăturate  $(x_{i-1}, x_i)$  și  $(x_i, x_{i+1})$ . Funcția  $S(x)$  poate fi un singur polinom pe întreaga axă reală. Definiția generală a funcțiilor spline include toate polinoamele de grad  $m$  sau mai mic.

Funcțiile spline cubice s-au dovedit a fi foarte folositoare pentru rezolvarea problemelor practice de interpolare. Din păcate, ocazional ele prezintă o discontinuitate sau puncte de inflexiune laterale în curbă. Astfel ar fi de dorit să existe o posibilitate de interpolare în care curba să nu treacă neapărat prin valorile date, ci să răspundă la o „tensiune” efectivă care ar putea fi considerată ca fiind produsă prin tragere în cele două capete. Aplicînd o asemenea tensiune, punctele de inflexiune nedorite pot fi îndepărtate.

Noțiunea de „spline sub tensiune” poate fi aproximată analitic, ducînd la interpolări sau la construirea aproximativă de curbe și suprafețe sub ten-

sune. Există în literatura de specialitate seturi de algoritmi pentru interpolarea curbilor și suprafețelor definite în mod obișnuit sau parametric.

În scopuri practice, restricția la un singur tip particular de spline simplu și deosebit, s-a dovedit a fi folositoare. Acest tip de funcție spline este dat de funcția de putere trunchiată

$$X_+^m = \begin{cases} x^m & \text{dacă } x > 0 \\ 0 & \text{dacă } x \leq 0 \end{cases}$$

O funcție spline  $S(x)$  de grad impar  $2k - 1$  cu nodurile  $x_1, x_2, \dots, x_n$  este denumită „spline naturală”, dacă este definită în fiecare din cele două intervale  $(-\infty, x_1)$  și  $(x_n, +\infty)$  de un polinom de gradul  $< k$ . Se poate demonstra că funcțiile spline naturale asigură o soluție unică de interpolare. Mai mult, o astfel de interpolare  $S(t)$  este o funcție de interpolare netedă pentru  $n$  puncte date, adică este o funcție care minimizează integrala

$$\int_a^b [S^{(k)}(x)]^2 dx \quad \text{pentru } 0 < k < n$$

Să ne întoarcem acum la problema de interpolare

$$\begin{aligned} s'(t_0) &= f'(t_0) \\ s(t_i) &= f(t_i), \quad t_i = i \cdot h, \quad i \in [0: n] \\ s'(t_n) &= f'(t_n) \end{aligned}$$

Din punct de vedere al calculelor, această problemă ar putea fi mult simplificată dacă ar fi posibil să se obțină  $s(t)$  prin îmbinarea funcțiilor spline în care rigurozitatea dorită este construită apriori. De exemplu, dacă folosim funcții spline, cubice, curba rezultantă  $s(t)$  ar avea proprietatea  $s \in C^2[t_0, t_n]$ . O asemenea simplificare poate fi realizată folosind un caz special de funcții spline numite „funcții B-spline”.

### 9.5.2. Funcția și curba B-spline

Să definim mai întâi o funcție B-spline de grad impar  $r = 2k - 1$  cu noduri spațiale distribuite uniform:

$$\beta_r(\tau) = \frac{1}{r!} \sum_{j=-k}^k (-1)^{j+k} \binom{2k}{j+k} (j-\tau)_+^r$$

Funcția  $\beta_r(\tau)$  este simetrică față de  $\tau = 0$ , are forma de clopot și este negativă pe intervalul  $[-k, k]$ . În afara acestui interval funcția este identic zero. Mai exact funcția are proprietățile următoare

$$\beta_r(\tau) \begin{cases} > 0 & \text{dacă } |\tau| < k \\ = 0 & \text{“ } |\tau| = k \\ \equiv 0 & \text{“ } |\tau| > k \end{cases}$$

$$\beta_r(-\tau) = \beta_r(\tau)$$

$$\beta_r(0) > \beta_r(\tau) \quad \text{dacă } \tau \neq 0$$

$$\sum_{i=-\infty}^{\infty} |\beta_r(i)| = \sum_{i=1-k}^{k-1} \beta_r(i) = 1$$

derivatele  $\beta_r^{(p)}(\tau)$ ,  $p = 1, 2, \dots, r$ , ale funcției  $\beta_r(\tau)$  sînt date de

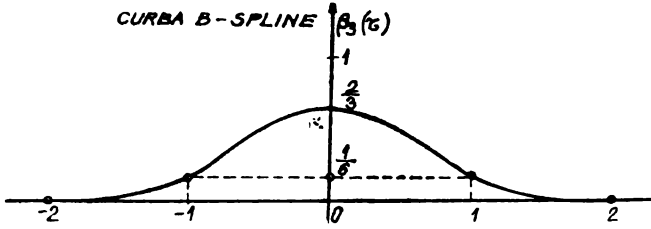


Fig. 9.7

$$\beta_r^{(p)}(\tau) = \frac{1}{(r-p)!} \sum_{j=-k}^k (-1)^{j+k+p} \binom{2k}{j+k} (j-\tau)_+^{k-p}$$

rezultă din această ecuație că

$$\beta_r(\tau) \in C^{r-1}$$

$$\beta_r^{(p)}(\pm k) = 0, \quad \forall p \in [0: r-1]$$

Graficul funcției B-spline  $\beta_3(\tau)$  este reprezentat în figura 9.7. Să considerăm intervalul de aproximare  $[0, 1]$  și un nod uniform cu intervalul  $h = 1/q$ . Se poate vedea că cele  $q+r$  funcții  $\beta_r(qt-i)$ ,  $i \in [1-k: q+k-1]$  sînt liniar independente pe  $[0, 1]$ . De aceea o funcție spline arbitrară de gradul  $r$  pe  $[0, 1]$  cu  $(q+1)$  noduri în  $t_j = j/q$ ,  $j \in [0: q]$  poate fi reprezentată de funcția

$$S_r(t) = \sum_{i=1-k}^{q+k-1} a_i \beta_r(qt-i)$$

pe suportul compact pentru fiecare  $\beta_r(qt-i)$  (intervalul peste care  $\beta_r(qt-i)$  este ne-nul) există cel mult  $(r+1)$  termeni nenuli în această sumare pentru orice  $t \in [0, 1]$  dinainte prevăzut. În plus, avem

$$\sum_{i=1-k}^{q+k-1} \beta_r(qt-i) = 1 \quad \forall t \in [0, 1]$$

și de aceea

$$\|S_r(t)\| \leq \sup_i |a_i|, \quad \forall t \in [0, 1]$$

Derivata lui  $S_r(t)$  este

$$\frac{d^p}{dt^p} S_r(t) = q^p \sum_{i=1-k}^{q+k-1} a_i \beta_r^{(p)}(qt-i)$$

Rezolvarea problemei de interpolare propusă (sau pentru orice problemă similară) este acum imediată. Alegem curba B-spline cubică

$$S_3(t) = \sum_{i=-1}^{n+1} a_i \beta_3(nt-i)$$

Prima derivată este

$$S_3'(t) = n \sum_{i=-1}^{n+1} a_i (-1) \beta_3'(nt-i)$$



avem

$$S_3'(t_0) = a_{-1}n\beta_3'(nt_0 + 1) + a_0n\beta_3'(nt_0) + \dots + a_{n+1}n\beta_3'(nt_0 - n - 1) = f'(t_0)$$

$$S_3(t_i) = a_{-1}\beta_3(nt_i + 1) + a_0\beta_3(nt_i) + \dots + a_{n+1}\beta_3(nt_i - n - 1) = f(t_i)$$

$$S_3'(t_n) = a_{-1}n\beta_3'(nt_n + 1) + a_0n\beta_3'(nt_n) + \dots + a_{n+1}n\beta_3'(nt_n - n - 1) = f'(t_n)$$

$$\forall i \in [0: n]$$

Acestea constituie un set de  $n + 3$  ecuații lineare, adică un sistem

$$\bar{B} \cdot \bar{a} = c$$

unde  $\bar{a} = [a_{-1}, a_0, \dots, a_{n+1}]$  este vectorul coeficienților (încă necunoscuți acum) și  $\bar{C}$

$$\bar{C} = [f'(t_0), f(t_0), f(t_0), \dots, f(t_i), \dots, f(t_n), f'(t_n)]^T$$

este vectorul restricțiilor.

Matricea  $B$  este determinată de valorile funcțiilor  $\beta_3(\tau)$  și  $\beta_3'(\tau)$  în nodurile date în următorul tabel:

$\tau =$	-2	-1	0	+1	+2
$\beta(\tau)$	0	$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$	0
$\beta_3'(\tau)$	0	$\frac{1}{2^n}$	0	$-\frac{1}{2^n}$	0
$\beta_3''(\tau)$	0	$n^2$	$-2n^2$	$n^2$	0

care rezultă din matricea

$$\bar{B} = \begin{bmatrix} -\frac{n}{2} & 0 & \frac{n}{2} & 0 & 0 & 0 \\ \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & 0 & 0 \\ 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & 0 \\ 0 & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ \cdot & & & & & \\ \cdot & & & & & \\ 0 & 0 & 0 & & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ 0 & 0 & 0 & & 0 & -\frac{n}{2} & 0 & \frac{n}{2} \end{bmatrix}$$

Matricea  $B$  trebuie să fie inversată pentru a se obține vectorul coeficienților

$$a = B^{-1} \cdot c$$

Matricea  $B$  este nesingulară. Funcțiile B-spline având suportul foarte mic, duc la matrice bine condiționate, având o structură specială care simplifică calculul. Astfel dacă schimbăm una din valorile  $c_i$ , ea afectează cel mult patru din coeficienții  $a_i$ .

*Schoenberg* care a studiat primul funcțiile B-spline a demonstrat că acestea sînt singurele funcții spline nenule cu suportul cel mai mic.

### 9.5.3. Aproximarea curbelor cu ajutorul funcției B-spline

Fie  $x_i$  nodurile iar  $x$  vectorul nodurilor. Ansamblul de funcții polinomiale spline  $S(K, x)$  de gradul  $k-1$  definite pe  $x$  este un spațiu vectorial de dimensiune finită. El este așa dar generat de un ansamblu minim de funcții spline independente. În general pot fi folosite în comun mai multe baze pentru  $S(k, x)$ . O bază specială (de unde vine și denumirea de B-spline) o constituie extensia *polinoamelor lui Bernstein*.

Funcția B-spline de bază  $N_{i,k}(t)$  de gradul  $k-1$  avînd ca suport  $(x_i, x_{(i+k) \bmod n})$  este dată de procedeul recursiv următor:

$$N_{i,1}(t) = \begin{cases} 1 & \text{pentru } x_i \leq t < x_{i+1} \\ 0 & \text{altfel} \end{cases}$$

iar pentru  $k > 1$

$$N_{i,k}(t) = \frac{t - x_i}{x_{i+k-1} - x_i} N_{i,k-1}(t) + \frac{x_{i+k} - t}{x_{i+k} - x_{i+1}} N_{i+1,k-1}(t)$$

(în această formulă se adoptă convențiile următoare: indicii sînt considerați modulo  $n$  și raportul  $\frac{0}{0}$  are ca valoare 0) —

Fără a pierde din generalitate, putem presupune acum că nodurile sînt întregi  $0, 1, \dots, n$  adică se consideră o bază uniformă a funcției B-spline. Dacă se presupune că vectorul nodurilor poate avea mai multe noduri identice, se va limita la  $k$  multiplicitatea fiecăruia dintre noduri.

Anumite baze ale funcției B-spline sînt interesante pentru vectori particulari. De exemplu baza generată pornind de la  $X = (0, 1, 2, \dots, n)$  este numită *bază periodică* a funcției B-spline. Într-adevăr, fiecare funcție de bază este translatarea unei singure funcții de bază (fig. 9.8a).

Se numește *bază neperiodică* a funcției B-spline pe  $[0, n]$  baza generată de vectorul nodurilor

$$X = (\underbrace{0, 0, \dots, 0}_{\text{de } k \text{ ori}}, 1, 2, \dots, n-1, \underbrace{n, n, \dots, n}_{\text{de } k \text{ ori}})$$

Astfel se obține o pierdere de periodicitate (fig. 9.8 b).

Cazul extrem pentru o bază neperiodică pe  $[0, n]$  este dat cu ajutorul vectorului nodurilor

$$X = (\underbrace{0, 0, \dots, 0}_{\text{de } k \text{ ori}}, \underbrace{1, 1, \dots, 1}_{\text{de } k \text{ ori}})$$

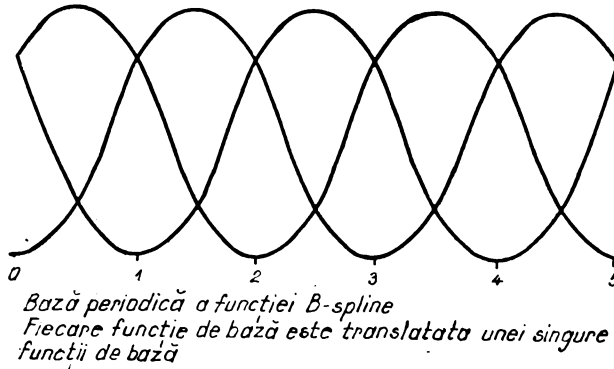


Fig. 9.8 a

adică  $n = 1$ . Baza degenerată a funcției B-spline verifică, atunci, pentru  $t \in (x_0, x_t)$  relațiile

$$\begin{aligned} N_{k-1,1}(t) &\equiv 1 \quad \text{și} \quad N_{i,1}(t) = 0 \quad \text{pentru} \quad i \neq k-1 \\ N_{k-2,2}(t) &= 1-t \quad \text{și} \quad N_{k-1,2}(t) = t, \quad N_{i,2}(t) = 0 \quad \text{pentru} \quad \begin{matrix} i \neq k-2 \\ i \neq k-1 \end{matrix} \end{aligned}$$

Continuînd calculul se poate arăta că baza funcției B-spline nu este alta decît ansamblul polinoamelor lui Bézier. Vom insista mai departe asupra acestei probleme.

În privința aplicațiilor, în majoritatea cazurilor pot fi utilizați cu succes următorii vectori ai nodurilor:

$$[0, 1, 2, 3, 4], \quad [0, 0, 0, 1, 2, 2, 2], \quad [0, 0, 1, 1, 2, 2]$$

Am arătat că vectorul nodal determină domeniul parametrului  $t$ .

Fie  $P = P_0 P_1 \dots P_m$  un poligon deschis sau închis ( $P_{m+1} = P_0$ ) considerat ca linie frîntă directoare. Curba B-spline de gradul  $k-1$  (ordinul  $K$ ) asociată poligonului  $P$  este dată de funcția:

$$S_m(P) = \sum_{i=0}^m P_i N_{i,k}(t), \quad x_0 \leq t < x_t \quad (i = m+1 \text{ sau } i = m+k)$$

Pentru a obține funcția B-spline curbă închisă (sau periodică) asociată lui  $P$  vom lua ca vector  $X = (x_0, x_1, \dots, x_{m+1})$  cu  $x_t = (i - k2) \bmod (m+1)$

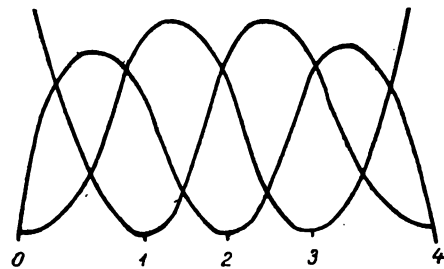
Pentru a obține funcția B-spline curbă deschisă (sau neperiodică) asociată lui  $P$ , se ia

$$x_i = (x_0, x_1, \dots, x_{m+k}) \text{ cu}$$

$$x_0 = x_1 = \dots = x_{k-1} = 0$$

$$x_{k-1+t} = i \text{ cu } i = 1, 2, \dots, m-k+1$$

$$x_m = x_{m+1} = \dots = x_{m+k} = m-k+2$$



Bază neperiodică a funcției B-spline

Fig. 9.8 b

Se poate urmări pe figura 9.8 c, marea eficacitate a funcțiilor B-spline: poligonul este o foarte bună imagine a curbei pe care o definește. În acest caz curba B-spline este de ordinul 4.

### 9.5.4. Precizări generale și comparative asupra aproximării cu B-spline

Aproximarea B-spline este o generalizare a aproximării lui Bernstein. Ea are ca și aceasta din urmă proprietatea de diminuare a variației. Ea are o altă proprietate interesantă pe care nu o posedă aproximația lui Bernstein: Funcțiile B-spline de bază sînt nule peste tot cu excepția suportului lor. O aproximare B-spline este o schemă *locală*. În fiecare punct aproximarea nu ține cont decît de comportamentul funcției în vecinătatea acestui punct. Astfel o perturbare locală a funcției de obținut se traduce printr-o perturbare locală pe funcția B-spline de aproximare. Aceasta este principala diferență față de aproximarea lui Bernstein, care este globală (fig.9.8 d).

Se poate observa că, pentru a se potrivi cel mai bine cu funcția de aproximare, avem de ales între două metode: modificarea gradului  $k$  sau schimbarea nodurilor adăugîndu-le puncte. De fapt, se păstrează un grad foarte mic, ceea ce face ca metoda să convergă extrem de rapid, mult mai mult, decît utilizînd alte metode (de exemplu, metoda lui Bézier).

Proprietatea de diminuare a variației și caracterul local al aproximării prin funcția B-spline explică comportamentul geometric al curbelor B-spline:

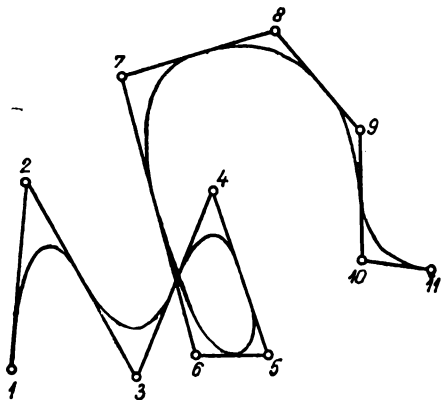
— un punct de pe o astfel de curbă este o combinație convexă de  $K$  vîrfuri vecine ale poligonului: astfel, curba „se potrivește” bine poligonului;

— dacă se deplasează un vîrf al poligonului, curba nu este perturbată decît în vecinătatea vîrfului deplasat. Această proprietate este esențială deoarece permite desenatorului să facă retușuri pe o curbă care nu îl satisface pe deplin, fără a modifica totalitatea traseului (contrariu curbelor lui Coons și Bézier);

— se poate face să treacă o funcție B-spline printr-o suită de puncte aliniate;

— se pot crea puncte de schimbare de sens, dar acestea nu pot apărea accidental (contrariu curbelor lui Bézier).

În concluzie, putem spune că utilizarea funcțiilor B-spline are avantaje enorme în raport cu metoda lui Bézier: mai întîi, un algoritm foarte simplu



Curba B-spline de ordinul 4  
Eficacitate: poligonul asociat dă o foarte  
bună imagine a curbei pe care o definește

Fig. 9.8 c

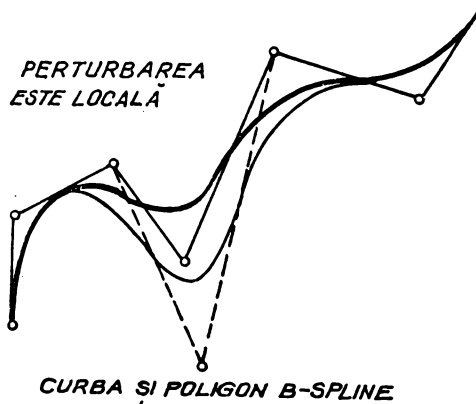


Fig. 9.8 d

permite generarea curbelor deschise sau închise; se poate controla forma curbei mulțumită poligonului de definiție, într-un mod mai eficient decât cu aproximarea lui Bézier; metoda fiind locală, timpii de calcul depind numai de gradul curbei (în general foarte mic, 3 sau 4, contrariu lui Bézier, putând să ajungă pînă la o sută); în sfîrșit, proprietăți geometrice mult mai interesante (controlul punctelor de schimbare de sens, posibilitate de a desena dreptă, etc.). Aproximarea cu ajutorul funcțiilor B-spline pare, deci, a fi — la ora actuală — cea mai interesantă pentru a fi pusă în aplicare în cadrul unui sistem interactiv. Numeroase programe sînt în multe țări în curs de dezvoltare ca și la noi de altfel.

### 9.5.5. Aplicație.

Să se determine curba B-spline definită de virfurile poligonului plan deschis  $P_0P_1P_2P_3$  pentru  $K = 3$

Coordonatele punctelor:  $P_0(0,0)$ ,  $P_1(1,1)$ ,  $P_2(3,-1)$ ,  $P_3(1,-1)$  (fig. 9.9)

Curba B-spline căutată este funcția

$$P(t) = P_0N_{0,3}(t) + P_1N_{1,3}(t) + P_2N_{2,3}(t) + P_3N_{3,3}(t)$$

Ca să obținem funcția  $N_{i,3}(t)$ ,  $i = 1, 2, 3, 4$  este necesar să alegem vectorul nodal (de exemplu al doilea din relația amintită)

$$[x_0, x_1, x_2, x_3, x_4, x_5, x_6] = [0, 0, 0, 1, 2, 2, 2]$$

astfel încît parametrul  $t$  variază între 0 și 2. Am arătat că în general pentru curbele B-spline deschise se obține poligonul de grad  $k-1$  pentru virfurile liniei frînte

$$P_j (j = 0, 1, \dots, n) \text{ de la } 0 \text{ la } n - k + 2$$

Calculăm ușor:

$$\begin{aligned} \text{a) } & \left. \begin{aligned} N_{0,1}(t) &= 1 \\ N_{1,1}(t) &= 1 \end{aligned} \right\} t = 0 \\ & \left. \begin{aligned} N_{2,1}(t) &= 1 \\ N_{3,1}(t) &= 1 \end{aligned} \right\} t \in (0, 1) \\ & \left. \begin{aligned} N_{4,1}(t) &= 1 \\ N_{5,1}(t) &= 1 \end{aligned} \right\} t = 2 \end{aligned}$$

Astfel  $N_{i,1}(t) = 0$ ,  $i = 0, 1, 2, 3, 4, 5$

Graficul funcției  $N_{2,1}(t) = 1$  este redat pe figura 9.10.

$$\begin{aligned} \text{b) } & \left. \begin{aligned} N_{1,2}(t) &= 1 - t \\ N_{2,2}(t) &= \begin{cases} t \\ 2 - t \end{cases} \\ N_{3,2}(t) &= t - 1 \end{aligned} \right\} \begin{aligned} t &\in (0, 1) \\ t &\in (0, 1) \\ t &\in (1, 2) \\ t &\in (1, 2) \end{aligned} \end{aligned}$$

Astfel  $N_{i,2}(t) = 0$ ,  $i = 0, 1, 2, 3, 4$

Graficul funcției  $N_{2,2}(t)$  este redat pe figura 9.11

$$\text{c) } N_{0,3}(t) = (1 - t^2) \quad t \in (0, 1)$$

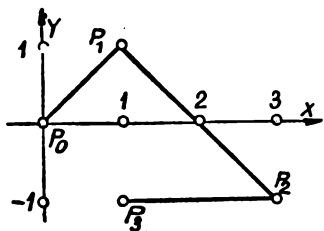


Fig. 9.9

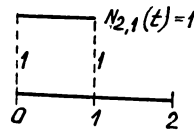


Fig. 9.10

$$N_{1,3}(t) = \begin{cases} \frac{1}{2} (4t - 3t)^2 & t \in (0, 1) \\ \frac{1}{2} (2 - t)^2 & t \in (1, 2) \end{cases} \quad N_{2,3}(t) = \begin{cases} \frac{1}{2} t^2 & t \in (0, 1) \\ \frac{1}{2} (-3t^2 + 8t - 4) & t \in (1, 2) \end{cases}$$

$$N_{3,3}(t) = (t - 1)^2 \quad t \in (1, 2)$$

Astfel,  $N_{i,3}(t) = 0$ ,  $i = 0, 1, 2, 3$ .

Graficul funcției  $N_{i,3}(t)$  este redat pe figura 9.12

Avem

$$\begin{aligned} x(t) &= 0 \cdot N_{0,3}(t) + 1 \cdot N_{1,3}(t) + 3 \cdot N_{2,3}(t) + 1 \cdot N_{3,3}(t) = \\ &= \begin{cases} \frac{1}{2} (4t - 3t^2) + \frac{3}{2} t^2 & t \in (0, 1) \\ \frac{1}{2} (2 - t)^2 + \frac{3}{2} (-3t^2 + 8t - 4) + (t - 1)^2, & t \in (1, 2) \end{cases} \\ y(t) &= \begin{cases} 2t & t \in (0, 1) \\ -3t^2 + 8t - 3 & t \in (1, 2) \end{cases} \\ z(t) &= 0 \cdot N_{0,3}(t) + 1 \cdot N_{1,3}(t) - 1 \cdot N_{2,3}(t) - 1 \cdot N_{3,3}(t) = \\ &= \begin{cases} \frac{1}{2} (4t - 3t^2) - \frac{1}{2} t^2 & t \in (0, 1) \\ \frac{1}{2} (2 - t)^2 - \frac{1}{2} (-3t^2 + 8t - 4) - (t - 1)^2 & t \in (1, 2) \end{cases} \\ y(t) &= \begin{cases} 2(t - t^2) & t \in (0, 1) \\ t^2 - 4t + 3 & t \in (1, 2) \end{cases} \end{aligned}$$

Graficul curbei B-spline căutate este redat în figura 9.13.

Dacă pentru același poligon (linie frântă) din fig. 9.9 se alege  $K = 2$  se obține linia frântă dată.

Pentru  $K = 3$  se obține curba B-spline compusă din două arce de parabolă care se racordează în mijlocul segmentului  $P_1P_2$ .

Pentru  $K = 3$  este reprezentată și curba Bézier (cu linie întreruptă și nevalabilă pentru indicațiile  $t = 1$ ,  $t = 2$  de pe figura 9.13)

Pentru  $K = 4$  curba B-spline este cubică.

Curba B-spline de gradul  $K = n$ , unde  $n$  este numărul vîrfurilor liniei frante, este curba Bézier. Curbele Bézier sînt așa dar, de la caz la caz, curbe B-spline speciale. Acest fapt va fi arătat în detaliu în paragraful 9.6.

## 9.6. Curbe Bézier și aproximarea Bézier a curbelor

### 9.6.1. Aproximarea polinomială Bernstein

În afară de interpolarea unei curbe prin niște valori date, apare necesitatea interpolării unei suprafețe adică a determinării unei suprafețe care să aproximeze cît mai exact un set de informații date. Un caz foarte tipic este

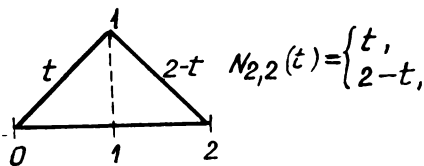


Fig. 9.11

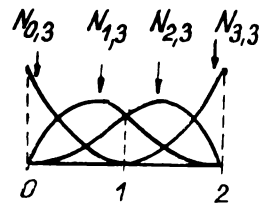


Fig. 9.12

problema de proiectare a caroseriilor autoturismelor sau a fuselajelor de avioane unde o reprezentare matematică va fi dată printr-o schiță sau un model din argilă, furnizat de proiectant.

Bézier și-a bazat metoda lui de aproximare pe o metodă clasică de aproximare, cunoscută ca aproximarea polinomială Bernstein.

Astfel fie  $f(t)$  o funcție arbitrară cu valoarea reală continuă pe intervalul  $[0, 1]$

Aproximarea polinomială Bernstein de gradul  $n$  referitoare la funcția  $f$  este

$$B_n(f(t)) = \sum_{k=0}^n f\left(\frac{k}{n}\right) \Phi_{k,n}(t),$$

cu

$$\Phi_{k,n}(t) = \binom{n}{k} t^k (1-t)^{n-k},$$

$$K \in [0, 1 \dots, n] \text{ iar } \binom{n}{k} = C_n^k$$

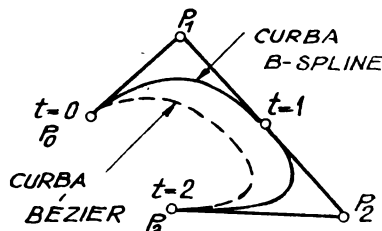


Fig. 9.13

Cititorii familiarizați cu teoria probabilității vor recunșea că funcțiile acestei ecuații sînt identice cu funcțiile binomiale de probabilitate.

Reamintind unele proprietăți elementare ale acestor funcții putem afirma:

1.  $\forall K \in [0 : n]; \Phi_{k,n} \geq 0; \forall t \in [0 : 1]: \sum_{k=0}^n \Phi_{k,n}(t) = 1$
2.  $\forall K \in [1 : n - 1]: \Phi_{0,n}(0) = 1; \Phi_{k,n}^{(k)}(0) = \frac{n!}{(n-k)!}$   
 $\forall p \in [0 : k - 1]; \Phi_{k,n}^{(p)}(0) = 0; \forall q \in [0 : n - k - 1]: \Phi_{k,n}^{(q)}(1) = 0$
3.  $\forall p \in [0 : n - 1]; \Phi_{n,n}^{(p)} = 0; \Phi_{n,n}(1) = 1$
4.  $\Phi_{k,n}^{(n-k)}(1) = \frac{(-1)^{n-k} n!}{(n-k)!}$
5.  $\Phi_{k,n}\left(\frac{k}{n}\right) = \binom{n}{k} K^k (n-k)^{n-k} > \Phi_{k,n}(t)$  dacă  $t \neq \frac{k}{n}$

Condițiile 2 și 3 implică faptul că cele două valori în punctele de capăt  $f(0)$  și  $f(1)$  sînt singurele valori, în general, care sînt interpolate de polinomul Bernstein  $B_n$ .

Din condițiile pentru  $\Phi_{k,n}(t)$  date mai sus, derivatele în punctele de capăt ale funcției  $B_n(t)$  pot fi obținute astfel:

$$\frac{d^p}{dt^p} B_n(f(t)) \Big|_{t=0} = \frac{n!}{(n-p)!} \sum_{k=0}^p (-1)^{p-k} \binom{p}{k} f\left(\frac{k}{n}\right)$$

$$\frac{d^p}{dt^p} B_n(f(t)) \Big|_{t=1} = \frac{n!}{(n-p)!} \sum_{k=0}^p (-1)^k \binom{p}{k} f\left(\frac{n-k}{n}\right)$$

Derivata de ordinul  $p$  în punctele de capăt (extremitățile segmentului)  $t = 0$  și  $t = 1$  este determinată prin valorile funcției  $f(t)$  în punctele respective și în vecinătatea lor. Îndeosebi, prima derivată exprimă înclinarea dreptei care leagă punctul de capăt cu punctul interior adiacent.

Polinoamele Bernstein satisfac teorema de aproximare a lui Weierstrass. Ele converg uniform crescător cu funcția pe care o aproximează iar aproximarea funcției  $B_n(f(t))$  este mai netedă decât însăși funcția  $f$ . Aproximarea Bézier presupune specificarea unui set de  $n + 1$  puncte bine ordonate:

$$P = \left\{ \left( \frac{i}{n}, v_i \right) \mid i \in [0 : n] \wedge v_i \in R \right\}$$

Relația de ordonare este definită prin

$$\left( \frac{i}{n}, v_i \right) \leq \left( \frac{j}{n}, v_j \right) \text{ dacă } i \leq j$$

Poligonul (deschis) format prin unirea punctelor succesive este denumit poligonul Bézier cu  $n$  laturi și este asociat cu polinomul Bernstein de gradul  $n$

$$B_n(P, t) = \sum_{k=0}^n V_k \Phi_{k,n}$$

în care  $\Phi_{k,n}$  sînt funcțiile definite mai sus iar valorile  $v_i$  ale vîrfurilor poligonului Bézier sînt coeficienții în ordinea lor dată. Acest poligon va fi denumit curba Bézier asociată cu poligonul Bézier. Într-un proces de proiectare interactiv, obiectivul nu este să se aproximeze poligonul lui Bézier ci mai de grabă funcțiile poligonului Bézier ca mod prin care forma curbei Bézier asociată să poată fi controlată. Faptul că astfel se asigură un mod de proiectare interactiv mai real, a fost descoperirea ingenioasă a lui Bézier.

În concluzie caracteristica remarcabilă a aproximării lui Bernstein este aceea de a păstra alura funcției primitive. În particular, aproximarea este cel puțin întotdeauna la fel de netedă ca și funcția inițială, o interpolare polinomială a lui Lagrange sau o aproximare prin metoda celor mai mici pătrate converg mai repede decât metoda Bernstein, dar cu numeroase ondulări în jurul funcției. Aproximarea lui Bernstein are, pe de altă parte, proprietatea de a micșora variația, adică ea reproduce exact funcțiile lineare și nu are mai multe zerouri decât primitiva însăși. Această proprietate este importantă pentru că această metodă va fi folosită pentru a netezi o curbă dată. De aici se poate deduce că numărul de intersecții ale graficului  $B_n(f)$  cu toată dreapta nu depășește numărul de intersecții ale funcției inițiale cu dreapta.

### 9.6.2. Curba Bézier

Fie  $P_n (i = 0, 1, \dots, n, n+1)$  puncte ordonate din  $R^2$  și să considerăm poligonul deschis format unind succesiv cele  $n + 1$  puncte. Curba lui Bézier asociată poligonului este funcția vectorială următoare:

$$B_n(P_0, P_1, \dots, P_n) = \sum_{i=0}^n \Phi_i(t) P_i$$

în care  $\Phi_i(t)$  sînt densități binomiale.

Să considerăm o funcție  $F : R \rightarrow R$   $F(t)$  cu  $t \in [0, 1]$ . Se definește aproximarea vectorială de gradul  $n$  a funcției  $F$  prin relația

$$B_n(F) = \sum_{i=0}^n \Phi_i(t) F\left(\frac{i}{n}\right)$$

Dacă se consideră funcția  $F$  lineară  $P_i, P_{i+1} (i = 0, 1, \dots, n-1)$ , atunci  $(Ft)$  admite ca graf poligonul  $P_0, P_1, \dots, P_n$ .



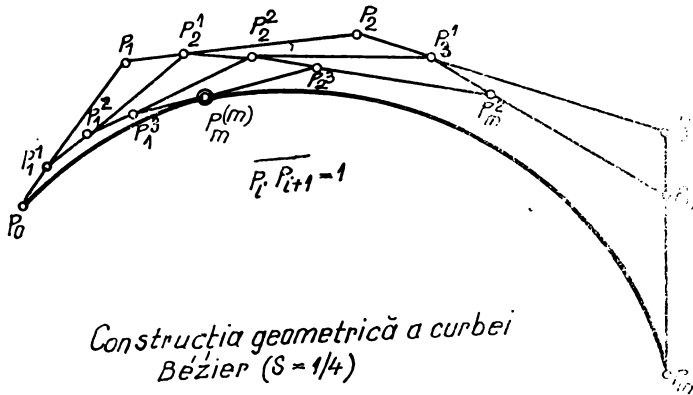


Fig. 9.14

### 9.6.3. Construcția geometrică a unei curbe Bézier ( $t = 1/4$ )

Se poate construi geometric punctul de pe curbă, corespunzător valorii  $t$  a parametrului. De-a lungul a  $(i + 1)$  una laturi a poligonului ( $i = 0, 1, \dots, n - 1$ ), se stabilește un punct la distanța  $t$  de  $P_i$  (luînd pentru  $P_i, P_{i+1}$  valoarea 1). Să unim cele  $n$  puncte, astfel obținute  $P_{i+1}^{(1)}$  și să considerăm poligonul lui Bézier 'cu  $n - 1$  laturi  $P_1^{(1)} P_2^{(1)} P_n^{(1)}$ . Să reîncepem operația precedentă: obținem  $P_1^{(2)} P_2^{(2)} P_n^{(2)}$ . După  $n$  iterații, se obține unicul punct  $P_m^{(n)}$  care este punctul căutat (fig. 9.14). La construirea unei curbe a lui Bézier un instrument foarte prețios este furnizat de hodograf. Să considerăm punctele  $P_i^* = P_{i+1} - P_i$ ,  $i = 0, \dots, n - 1$  construite pornind de la  $P_0 P_1 P_n$ . Se numește hodograf al lui  $B_n(P)$  curba lui Bézier de gradul  $(n-1)B_{n-1}(P^*)$  definită de poligonul  $P^* = P_0^* P_1^* P_2^* P_{n-1}^*$ . Acest hodograf nu este, în fapt, decît curba derivată din  $B_n(P)$ , la aproximativ o constantă multiplicativă. El permite detectarea proprietăților curbei ca, de exemplu, punctele de inflexiune, punctele de schimbare de direcție și punctele de curbură infinită.

### 9.6.4. Generalizarea la suprafețe a curbilor Bézier

Trecerea la suprafețe se face prin simplă generalizare, poligonul fiind înlocuit cu o „rețea” sprijinindu-se pe un ansamblu de puncte  $P$ :

$$\{P_{u,v}(u = 0, \dots, m; v = 0, \dots, n)\}$$

Pornind de la acest ansamblu de puncte, se construiește funcția vectorială:

$$F(s, t) : [0, 1] \times [0, 1] \rightarrow R^3$$

$$F(s, t) = mn \left( \frac{u+1}{m} - s \right) \left[ \left( \frac{v+1}{n} - t \right) P_{u,v} + \left( t - \frac{v}{n} \right) P_{u,v+1} \right] +$$

$$+ mn \left( s - \frac{u}{m} \right) \left[ \left( \frac{v+1}{n} - t \right) P_{u+1,v} + \left( t - \frac{v}{n} \right) P_{u+1,v+1} \right]$$

$$\frac{u}{m} \leq s \leq \frac{u+1}{m}; \quad \frac{v}{n} \leq t \leq \frac{v+1}{n}$$

$$u \in [0, 1, \dots, m-1]; \quad v \in [0, 1, \dots, n-1]$$

Graficul pentru  $s$  și  $t$  care respectă inegalitățile de mai sus este porțiunea de plan  $P_{uv}; P_{u+1,v}; P_{u+1,v+1}; P_{u,v+1}$ .

Înlocuind  $F$  prin valoarea sa în formula de aproximare a lui Bernstein a unei funcții de două variabile, obținem

$$B_{m,n}[F(s, t)] = \sum_{u=0}^m \sum_{v=0}^n \Phi_u(s) \psi_v(t) P_{u,v}$$

în care  $\Phi_u$  și  $\psi_v$  sînt densitățile binomiale.

Construcția unei suprafețe se poate deduce, atunci, din construcția a două ansambluri de curbe ale lui Bézier aparținînd suprafeței, fiecare dintre ele fiind o familie de generatoare.

### 9.6.5. Proprietăți ale curbei Bézier și aprecieri comparative

1. Curba Bézier are punctele de capăt (extremitățile) comune cu poligonul Bézier. Toate celelalte vîrfuri ale poligonului, în general, nu sînt situate pe curba Bézier.

2. Înclinarea vectorilor tangenți în punctele de capăt ale curbei Bézier este egală cu înclinarea primelor și respectiv ultimelor laturi (segmente) ale poligonului Bézier.

3. Curba Bézier se află în întregime în corpul convex al punctelor extreme, din poligonul Bézier.

În figura 9.14 a sînt reprezentate funcțiile de bază Bernstein  $\Phi_{k,3}(t)$   $k \in [0:3]$  pe care de altfel le-am calculat în Aplicația 1 din (9.5.5), iar în figura 9.15 sînt arătate polinomul Bézier, corpul lui convex (linia întreruptă) și curba Bézier asociată.

Curbele lui Bézier oferă avantajul considerabil de a permite desena-torului să vadă mai bine alura curbei pe care caută s-o obțină privind poligonul de aproximare. Modul de a opera este următorul: Desenează mai întii curba pe care dorește s-o obțină. Apoi selecționează pe această curbă un anumit număr de puncte, definind astfel un poligon deschis.

Sistemul calculează atunci curba lui Bézier corespunzătoare acestui poligon. În funcție de rezultat, utilizatorul modifică poziția unuia sau mai multor vîrfuri ale poligonului, astfel ca să deformeze curba răspuns. — Acest proces se repetă pînă cînd curba inițială și curba calculată coincid. Dispunem atunci, de un model matematic pentru curba de plecare.

Vedem, deci, că acest procedeu este cu atît mai interesant cu cît poligonul este mai apropiat de curba de reprezentat. De fapt, aici apar slăbiciunile metodei lui Bézier. Într-adevăr, pentru a obține acest rezultat, trebuie fie să se decupeze poligonul în puncte foarte numeroase, fie să se utilizeze o altă formă de parametrizare care permite să se ia mai bine în considerare curba (în special, făcînd să intervină dis-

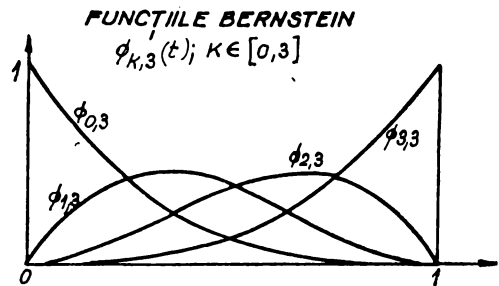


Fig. 14 a

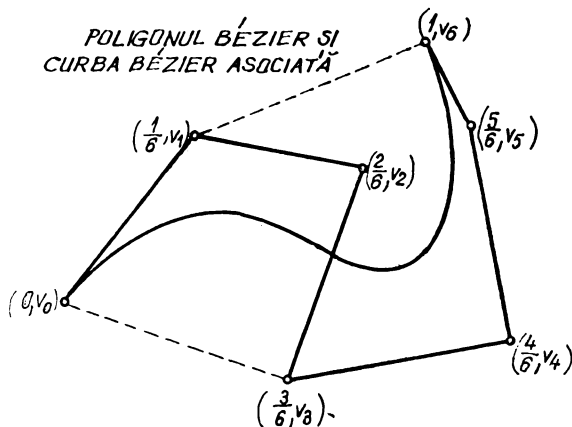


Fig. 9.15 a

influență este maximă asupra mediului apropiat și descrește pe măsură ce te depărtezi de el (fig. 9.15 b). Desigur acuratețea aproximării este limitată prin rezolvarea relativă a prezentării deoarece toate curbele spațiale desfășurate de tabloul de prezentare sînt approximate prin poligoane (linii frînte).

Din acest motiv, în proiectarea unei lucrări de precizie, desfășurarea interactivă trebuie asistată de un plotter.

În figura 9.16 este arătată posibilitatea producerii la plotter nu numai a curbelor Bézier deschise, ci și de asemenea a curbelor Bézier închise.

Pierre Bézier precizează însuși faptul că metoda lui de proiectare nu se limitează la folosirea polinoamelor Bernstein sau a polinoamelor în general. Astfel, cea mai bună alegere a funcțiilor de bază pare să fie funcțiile B-spline. Comparate cu polinoamele Bernstein, funcțiile B-spline oferă următoarele avantaje suplimentare (vezi și aprecierile comparative din 9.5.4.).

1. Aproximarea prin curbe B-spline produce o apropiere mai mare a curbei față de poligonul Bézier, decît aproximarea Bernstein—Bézier.

2. Aproximarea prin curbe B-spline oferă amplasarea dorită a curbei, proprietate care la aproximarea Bernstein lipsește cu desăvîrșire.

3. În aproximarea Bernstein—Bézier, gradul curbei Bézier crește proporțional cu numărul de laturi ale poligonului Bézier. Într-o aproximare prin curbe B-spline, ambii parametrii sînt independenți și astfel pot fi aleși în mod arbitrar.

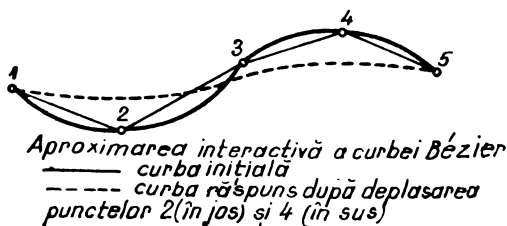


Fig. 9.15 b

tanța dintre punctele de definiție). În cele două cazuri, rezultatul este un timp de calcul destul de lung, repede incompatibil cu un dialog interactiv, și erori la calcul destul de mari. În sfîrșit un ultim inconvenient ține de definiția însăși a procedurii de interpolare, care este o schemă globală, adică aplicîndu-se la ansamblul de puncte ale poligonului.

Pot fi obținute deformări locale, căci, dacă fiecare vîrf are o influență asupra ansamblului, această

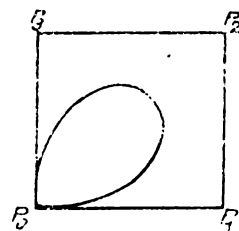


Fig. 9.16

În figura 9.17 este realizată o comparație între o curbă Bernstein — Bézier (linia continuă) și o curbă B-spline (linia întreruptă), ambele curbe fiind produse de același poligon cu 5 laturi (6 vîrfuri).

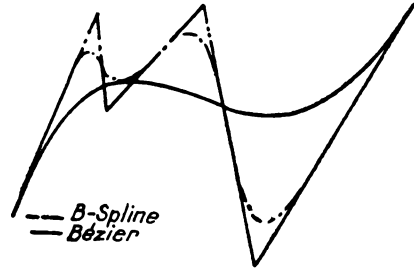


Fig. 9.17

Pot fi arătate alte exemple comparative interesante referitoare la posibilitățile de aproximare prin curbe Bézier și curbe B-spline pentru aceleași poligoane de bază cu  $n$  laturi. În cazul curbelor B-spline cubice prima și a treia latură a poligonului cu trei laturi sînt tangente la curbă.

În figura 9.18 este reprezentat comparativ un exemplu de aproximare prin curbe spațiale Bézier și B-spline pentru o linie frîntă spațială definită de 10 puncte.

#### 9.6.6. Aproximarea Bernstein pentru funcțiile de două variabile.

Aproximarea lui Bernstein poate fi extinsă la funcțiile de două variabile. Fie  $f(s, t) : R^2 \rightarrow R$ . Se definește

$$B_{m,n}[f; s, t] = B_m B_n[f; s, t] = \sum_{i=0}^m \sum_{j=0}^n \Phi_i(s) \psi_j(t) f\left(\frac{i}{m}, \frac{j}{n}\right)$$

în care  $\Phi_i$  și  $\psi_j$  sînt densități binomiale.

Proprietățile lui  $B_{m,n}$  se obțin pornind de la cele văzute înainte în cazul cu o variabilă. De exemplu,  $B_{m,n}(f)$  coincide cu  $f$  numai în cele patru colțuri ale domeniului său de definiție  $(0, 0)$ ,  $(0, 1)$ ,  $(1, 0)$ ,  $(1, 1)$ . Totuși de-a lungul

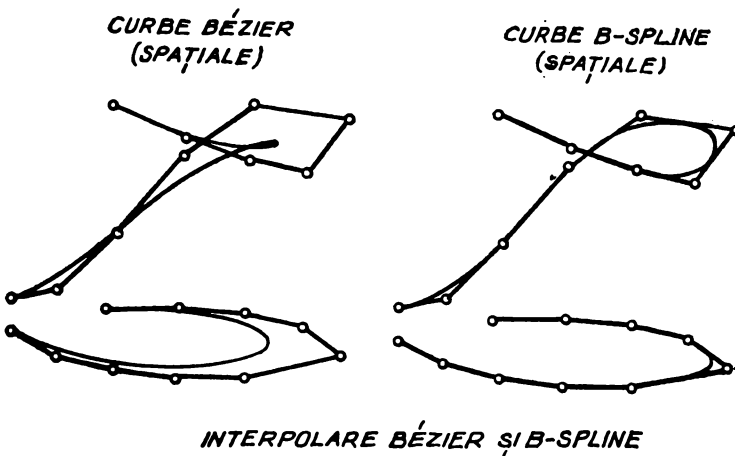


Fig. 9.18

celor patru limite  $B_{m,n}(f)$  se reduce la aproximarea lui Bernstein a valorilor funcțiilor  $f$  pe fiecare frontieră, adică:

$$B_{m,n}[f; s, t]_{t=0} = B_m[f; s, 0] = \sum_{i=0}^m \Phi_i(s) f\left(\frac{i}{m}, 0\right)$$

Valorile lui  $B_{m,n}(f)$  în interiorul domeniului de definiție sînt combinații convexe ale  $(m+1) \cdot (n+1)$  puncte ale ansamblului  $\left(\frac{i}{m}, \frac{j}{n}\right)$ .

### 9.6.7. Aplicație

Să se determine curba spațială Bézier pentru poligonul definit de punctele  $(0, 0, 0)$ ,  $(1, 1, 1)$ ,  $(2, 2, 2)$ ,  $(3, 3, 3)$ . Să se exprime și curbele B-spline cubice pentru poligonul  $(\bar{P}_i)_{i=0}^3$ .

Fie  $\{\bar{P}_i\}_{i=0}^3$  vectorii de poziție ai vîrfurilor poligonului care definește curba. Curba Bézier pentru poligonul dat se exprimă prin ecuația

$$\bar{R}(t) = \sum_{i=0}^3 \bar{P}_i B_{e_{im}}(t)$$

unde parametrul  $t$  este cuprins în intervalul  $[0, 1]$ ,  $\bar{R}(t)$  este vectorul de poziție al punctului curbei iar  $B_{e_{im}}(t)$  sînt polinoamele Bernstein, adică

$$B_{e_{im}}(t) = \binom{m}{i} t^i (1-t)^{m-i}$$

În cazul problemei vîrfurile  $\{P_i\}_{i=0}^3$  ale poligonului sînt în linie dreaptă, deci curba Bézier este segmentul  $(0, 0, 0)$ ,  $(3, 3, 3)$ .

Notăm  $r_1(t)$ ,  $r_2(t)$ ,  $r_3(t)$  componentele vectorului  $\bar{R}(t)$

Atunci

$$\begin{aligned} r_1(t) &= 0 \cdot B_{e_{03}}(t) + 1 \cdot B_{e_{13}}(t) + 2 \cdot B_{e_{23}}(t) + 3 \cdot B_{e_{33}}(t) = \\ &= \binom{3}{1} t(1-t)^2 + 2 \binom{3}{2} t^2(1-t) + 3 \binom{3}{3} t^3 = 3(t - 2t^2 + t^3) + 6(t^2 - t^3) + 3t^3 = 3t \end{aligned}$$

Evident și

$$r_2(t) = 3t$$

$$r_3(t) = 3t$$

Curba Bézier este deci segmentul

$$\bar{R}(t) = (3t, 3t, 3t), \quad [0, 1].$$

Curbele B-spline cubice pentru poligonul  $\{\bar{P}_i\}_{i=0}^3$  se pot exprima vectorial sub forma

$$\bar{R}(t) = [t^3, t^2, t, 1] \cdot \frac{1}{6} \begin{bmatrix} -1 & -3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

unde parametrul  $t$  variază în intervalul  $[0, 1]$ .

## 9.7. Studiul curbelor determinate vectorial prin polinoame

Cubica Ferguson, curba Bézier și cubica Coons sînt curbe exprimate prin polinoame vectoriale speciale.

Ecuția vectorială generală a curbelor exprimate prin polinoame este:

$$\bar{P}(u) = \bar{A}_0 + \bar{A}_1 u + \bar{A}_2 u^2 + \dots + \bar{A}_n u^n = \bar{A} \bar{u}$$

unde  $0 \leq u \leq 1$  iar  $\bar{u} = [1, u, u^2, \dots, u^n]^T$

De asemenea

$\bar{A} = [\bar{A}_0, \bar{A}_1, \bar{A}_2, \dots, \bar{A}_n]$  matricea vectorilor constanți care definește de fapt curba.

Să impunem condiția ca curba să treacă prin punctele  $P(u_i)$ .

Avem

$$\bar{P}(u_i) = \bar{A} \bar{u}_i$$

în care  $\bar{u}_i = [1, u_i, u_i^2, \dots, u_i^n]^T$  și sistemul de  $n + 1$  ecuații liniare pentru  $i = 0, 1, \dots, n$  se scrie matricial sub forma

$$\bar{P} = \bar{A} \bar{U} \quad \text{în care}$$

$$\bar{P} = [\bar{P}(u_0), \bar{P}(u_1), \dots, \bar{P}(u_n)] \quad \text{iar}$$

$$\bar{U} = \begin{bmatrix} 1, & 1, \dots, 1 \\ U_0, & U_1, \dots, U_n \\ U_0^2, & U_1^2, \dots, U_n^2 \\ \dots & \dots \\ U_0^n, & U_1^n, \dots, U_n^n \end{bmatrix}$$

Deoarece,  $u \neq u_k$  pentru  $j \neq k$ ,  $j, k = 0, 1, \dots, n$  există inversa  $\bar{U}^{-1}$  a matricei  $\bar{U}$  și rezolvînd sistemul obținem ecuația vectorială a curbei de interpolare care trece prin punctele  $P(u_i)$  unde  $i = 0, 1, 2, \dots, n$

$$\bar{A} = \bar{P} \bar{U}^{-1} \quad \text{și} \quad \bar{P}(u) = \bar{P} \bar{U}^{-1} \bar{u}$$

### 9.7.1. Aplicație

Să se determine curba plană care trece prin punctele  $(0,0)$ ,  $(0,1)$ ,  $(1,0)$ , și  $(2,0)$ , pentru parametrii  $0; \frac{1}{3}; \frac{2}{3}; 1$ .

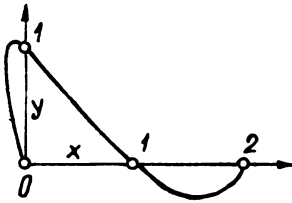


Fig. 9.19

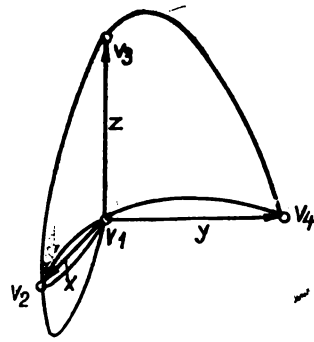


Fig. 9.20

Ecuțiile parametrice ale curbei pot fi exprimate prin cubice. Astfel:

$$x = [0, 0, 1, 2] \bar{U}^{-1}[1, u, u^2, u^3]^T = -\frac{5u}{2} + 9u^2 - \frac{9u^3}{2}$$

$$y = [0, 1, 0, 0] \bar{U}^{-1}[1, u, u^2, u^3]^T = 9u - \frac{45u^2}{2} + \frac{27u^3}{2}$$

graficul curbei este redat în figura 9.19.

### 9.7.2. Aplicație

Să se determine cubica spațială care trece prin punctele

$V_1(0, 0, 0)$ ,  $V_2(1, 0, 0)$ ,  $V_3(0, 0, 1)$  și  $V_4(0, 1, 0)$  pentru parametrii  $0, \frac{1}{3}, \frac{2}{3}, 1$ .

Ecuțiile parametrice ale curbei sînt:

$$x = [0, 0, 1, 0] \bar{U}^{-1}[1, u, u^2, u^3]^T = -4,5u + 18u^2 - 13,5u^3$$

În mod analog obținem

$$y = u - 4,5u^2 + 4,5u^3$$

$$z = 9u - 22,5u^2 + 13,5u^3$$

Imaginea perspectivă a curbei și a proiecției sale orizontale sînt redată în figura 9.20

## 9.8. Cubica Ferguson

Cubica Ferguson este curba determinată de două puncte extreme  $P_0$  și  $P_1$  (de început și de sfîrșit) ale curbiei de tangentele  $p_0$  și  $p_1$  în aceste puncte.

Această curbă a fost introdusă în grafica pe calculator de *J. C. Ferguson* în anul 1964.

Ecuția vectorială a cubicei Ferguson este

$$\bar{P}(t) = \bar{P}_0 F_1(t) + \bar{P}_1 F_2(t) + \bar{P}'_0 F_3(t) + \bar{P}'_1 F_4(t), \quad t \in [0, 1]$$

în care funcțiile  $F_i$ ,  $i = 1, 2, 3, 4$  sînt polinoamele cubice:

$$F_1(t) = 2t^3 - 3t^2 + 1$$

$$F_2(t) = -2t^3 + 3t^2$$

$$F_3(t) = t^3 - 2t^2 + t$$

$$F_4(t) = t^3 - t^2$$

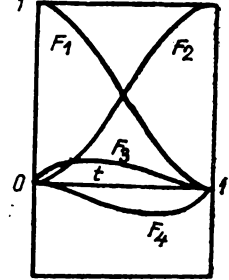


Fig. 9.21

reprezentate grafic în figura 9.21

Din expresia funcției  $P(t)$  rezultă în mod evident faptul că curba este o cubică. Deoarece  $F_1(0) = 1$ ,  $F_2(0) = F_3(0) = F_4(0) = 0$  avem  $P(0) = P_0$  deci punctul de început al curbei este  $\bar{P}_0$ . Analog putem demonstra că punctul final al curbei este  $P_1$ .

Dacă derivăm funcția  $P(t)$  obținem

$$\bar{P}'(t) = \bar{P}_0 F_1'(t) + \bar{P}_1 F_2'(t) + \bar{P}'_0 F_3'(t) + \bar{P}'_1 F_4'(t)$$

Deoarece  $F_1'(0) = F_2'(0) = F_4'(0) = 0$  și  $F_3' = 1$  rezultă  $P'(0) = P'_0$  și deci curba are în punctul de început vectorul tangent  $\bar{P}'_0$ . Analog putem demonstra că vectorul tangent în punctul final al curbei este  $P'_1$ .

Aceste caracteristici ale cubicei Ferguson le notăm în general prin  $\bar{P}_0$ ,  $\bar{P}_1$ ,  $\bar{P}'_0$ ,  $\bar{P}'_1$ .

### 9.8.1. Aplicație

Să se determine cubica Ferguson definită prin punctele terminale  $P_0(0, 0, 0)$ , și  $P_1(1, 1, 0)$  și prin vectorii tangenți curbei în aceste puncte  $P'_0 = [5, 0, 5]$  și  $P'_1 = (10, 0, 0)$ .

Din ecuația cubicei Ferguson obținem ecuațiile sale parametrice:

$$x(t) = F_2(t) + 5F_3(t) + 10F_4(t)$$

$$y(t) = F_2(t)$$

$$z(t) = 5F_3(t)$$

Înlocuind expresiile funcțiilor  $F_i$ ,  $i = 1, 2, 3, 4$  rezultă:

$$x(t) = 13t^3 - 17t^2 + 5t$$

$$y(t) = -2t^3 + 3t^2$$

$$z(t) = 5t^3 - 10t^2 + 5t$$

Imaginea axonometrică a curbei și a proiecției sale orizontale este dată în figura 9.22.

În figura 9.23 a fost schimbată valoarea vectorului  $\bar{P}'_1 = [10, 0, 0]$  cu vectorul  $[-5, 0, 0] = \bar{P}'_1$

### 9.8.2. Aplicație

Să se efectueze calculul vectorilor și să se exprime ecuațiile parametrice ale curbei Ferguson dată printr-o ecuație de forma:

$$\bar{P}(t) = \bar{m}t^3 + \bar{n}t^2 + \bar{p}t + \bar{q}$$

unde  $\bar{P}(t)$  este vectorul punctului spațial al curbei iar  $\bar{m}$ ,  $\bar{n}$ ,  $\bar{p}$ ,  $\bar{q}$ , sint vectorii coeficienților ( $t$  este parametrul).

Se vor considera  $\bar{P}_0 = \bar{A}(0, 0)$ ,  $\bar{P}_1 = \bar{B}(1, 0)$ ,

$$\bar{P}'_0 = \bar{a}(0, 1), \quad \bar{P}'_1 = \bar{b}(0, -1).$$

$$\bar{P}(0) = \bar{A} \quad \bar{P}(1) = \bar{B}$$



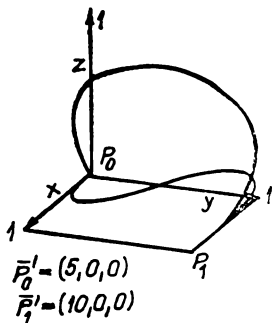


Fig. 9.22

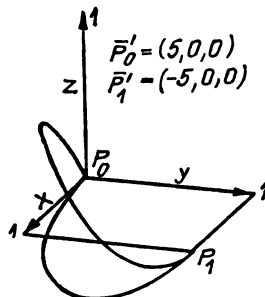


Fig. 9.23

Să efectuăm calculul vectorilor:

Deoarece  $\bar{P}(0) = \bar{q}$ , avem  $\bar{q} = \bar{A}$  și de asemenea:

$$\bar{P}(1) = \bar{m} + \bar{n} + \bar{p} + \bar{q} = \bar{m} + \bar{n} + \bar{p} + \bar{A} = \bar{B}$$

Pentru vectorii tangenți

$$\frac{d\bar{P}}{dt}(0) = \bar{p} = \bar{a}, \quad \frac{d\bar{P}}{dt}(1) = 3\bar{m} + 2\bar{n} + \bar{p} = \bar{b}$$

Vectorii  $\bar{m}$ ,  $\bar{n}$ ,  $\bar{p}$ ,  $\bar{q}$  ai coeficienților se obțin prin rezolvarea sistemului:

$$\begin{cases} \bar{m} + \bar{n} + \bar{p} + \bar{q} = \bar{B} \\ 3\bar{m} + 2\bar{n} + \bar{p} = \bar{b} \\ \bar{p} = \bar{a} \end{cases}$$

Avem:

$$\begin{aligned} \bar{m} + \bar{n} &= \bar{B} - \bar{A} - \bar{a} \\ 3\bar{m} + 2\bar{n} &= \bar{b} - \bar{a} \end{aligned}$$

Deci:

$$\begin{aligned} \bar{m} &= 2\bar{A} - 2\bar{B} + \bar{a} + \bar{b} \\ \bar{n} &= 3\bar{A} + 3\bar{B} - 2\bar{a} - \bar{b} \end{aligned}$$

Pentru valorile indicate avem

$$\begin{aligned} m_1 &= 2 \cdot 0 - 2 \cdot 1 + 0 + 0 = -2 \\ m_2 &= 2 \cdot 0 - 2 \cdot 0 + 0 + 1 - 1 = 0 \\ n_1 &= -3 \cdot 0 + 3 \cdot 1 - 2 \cdot 0 - 0 = 3 \\ n_2 &= -3 \cdot 0 + 3 \cdot 0 - 2 \cdot 1 - (-1) = -1 \\ p_1 &= 0 \\ p_2 &= 0 \\ q_1 &= 0 \\ q_2 &= 1 \end{aligned}$$

Ecuatiile parametrice ale curbei Ferguson sînt

$$\begin{aligned} x &= -2t^3 + 3t^2 \\ y &= -t^2 + 1 \end{aligned}$$

### 9.8.3. Program pentru desenarea plană a curbei Ferguson

Elementele necesare pentru întocmirea programului sînt componentele vectorilor de poziție pentru punctele inițial și final ale curbei precum și pentru tangențele în punctele inițial și final ale curbei căutate (fig. 9.24.).

Forma programabilă a ecuației curbei este

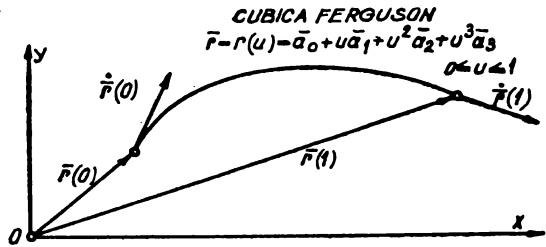


Fig. 9.24

$$\bar{r}(u) = [1, u, u^2, u^3] \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} \bar{r}(0) \\ \bar{r}(1) \\ \dot{\bar{r}}(0) \\ \dot{\bar{r}}(1) \end{bmatrix}$$

formă matricială la care s-a ajuns de la forma

$$\bar{r} = \bar{r}(u) = \bar{r}(0) \cdot (1 - 3u^2 + 2u^3) + \bar{r}(1) (3u^2 - 2u^3) + \dot{\bar{r}}(0) \cdot (u - 2u^2 + u^3) + \dot{\bar{r}}(1) \cdot (-u^2 + u^3)$$

### 9.8.4. Forma programabilă a suprafeței de tip Ferguson

Dacă în ecuația curbei

$$\bar{r}(u) = \bar{a}_0 + u\bar{a}_1 + u^2\bar{a}_2 + u^3\bar{a}_3$$

înlocuim vectorii  $\bar{a}_0, \bar{a}_1, \bar{a}_2, \bar{a}_3$  prin alte funcții care depind de parametrul  $V$  de tipul

$$\bar{a}_i = \bar{a}_{i0} + v\bar{a}_{i1} + v^2\bar{a}_{i2} + v^3\bar{a}_{i3} \quad i = 0, 1, 2, 3,$$

obținem ecuația suprafeței Ferguson:

$$\begin{aligned} \bar{r} = \bar{r}(u, v) &= \bar{a}_{00} + v\bar{a}_{01} + v^2\bar{a}_{02} + v^3\bar{a}_{03} + u(\bar{a}_{10} + v\bar{a}_{11} + v^2\bar{a}_{12} + v^3\bar{a}_{13}) + \\ &+ u^2(\bar{a}_{20} + v\bar{a}_{21} + v^2\bar{a}_{22} + v^3\bar{a}_{23}) + u^3(\bar{a}_{30} + v\bar{a}_{31} + v^2\bar{a}_{32} + v^3\bar{a}_{33}) = \\ &= \sum_{i=0}^3 \sum_{j=0}^3 \bar{a}_{ij} \cdot u^i v^j \end{aligned}$$

sau sub forma matricială

$$\bar{r} = \bar{r}(u, v) = [1, u, u^2, u^3] \cdot \begin{bmatrix} \bar{a}_{00} & \bar{a}_{01} & \bar{a}_{02} & \bar{a}_{03} \\ \bar{a}_{10} & \bar{a}_{11} & \bar{a}_{12} & \bar{a}_{13} \\ \bar{a}_{20} & \bar{a}_{21} & \bar{a}_{22} & \bar{a}_{23} \\ \bar{a}_{30} & \bar{a}_{31} & \bar{a}_{32} & \bar{a}_{33} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

unde  $u, v \in [0, 1]$

Din cei 16 vectori  $\bar{a}_{ij}$  se poate determina segmentul de suprafață Ferguson pentru  $u, v \in [0, 1]$ . După cum se știe vectorii  $\bar{a}_{ij}$  au interpretări geometrice deduse din valorile

$F, \frac{\partial F}{\partial u}, \frac{\partial F}{\partial v}, \frac{\partial^2 F}{\partial u \partial v}$  în punctele de colț ale suprafeței. Vectorii  $\frac{\partial F}{\partial u}, \frac{\partial F}{\partial v}$  sînt colineari cu vectorii  $T_u, T_v$  tangenți curbelor parametrice  $v = \text{const}, u = \text{const}$ .

```

PROGRAM FERF
PROGRAM PENTRU DETERMINAREA CURBEI FERGUSON
PARAMETRII RX, RY, RIX, RIY SÎNT COMPONENTELE VECTORULUI DE POZITIE
PENTRU PUNCTUL INITIAL SI FINAL AL CURBEI
PARAMETRII TX, TY, TIX, TIY SÎNT COMPONENTELE VECTORULUI TANGENT IN
PUNCTUL INITIAL SI FINAL AL CURBEI
PARAMETRUL DD ESTE VARIATIA PARAMETRULUI U
CALL ASSIGN (1, 'RR:')
CALL ASSIGN (2, 'LR:')
CALL ASSIGN (3, 'PP:')
CALL IY (3)
RX= (1, 1)RUX, RUY, RIX, RIY
RY= (4)R10.3, 40X)
TX= (1, 1)TUX, TUY, TIX, TIY
TY= (1, 2)T0U
RZ= (1, 2)E10.3, 70X)
CALL IY (0., -150., 0., 150.)
CALL IY (150., 0., -150., 0.)
CALL IY (5., 100., 0., 1., 0., 'PROGRAM FERF', 12)
RY=RX
TY=RY
TX=TX
TY=TY
RZ=3*(R1X-R2X)-2*T1X-T1X
RY=3*(R1Y-R2Y)-2*T1Y-T1Y
RZ=2*(R2X-R1X)+T1X+T1X
RZ=2*(R2Y-R1Y)+T1Y+T1Y

IY=
CALL PLOT (0., 0., 0)
1 IY (1, 0, 1) GO (1) 11
X=RX+U*(R1X+U*(R2X+U*(R3X)))
Y=RY+U*(R1Y+U*(R2Y+U*(R3Y)))
2 IY (2, 2) X, Y
3 IY (1, 1, X, Y, 2*10.3)
CALL PLOT (X, Y, 1)
U=U+.1
GO TO 11
1 CALL EOF
STOP
END

```

## 9.9. Cubica Bézier

Cubica Bézier este determinată prin polinomul cubic

$$\bar{P}(t) = \bar{V}_1 B_1(t) + \bar{V}_2 B_2(t) + \bar{V}_3 B_3(t) + \bar{V}_4 B_4(t) = \sum_{i=1}^4 \bar{V}_i B_i(t)$$

În care  $\bar{V}_i$  sînt vectorii celor patru vîrfuri  $V_i$  ale poligonului director al cubicei iar funcțiile cubice  $B_i$  sînt:

$$B_1(t) = (1 - t)^2$$

$$B_2(t) = 3t(1 - t)^2$$

$$B_3(t) = 3t^2(1 - t)$$

$$B_4(t) = t^3$$

Aceste funcții cubice  $B_i$  sînt reprezentate în figura 9.25.

Poligonul director are o anumită semnificație pentru cubica Bézier. Astfel  $\bar{P}(0) = \bar{V}_1$  deci  $V_1$  este punctul de început al curbei deoarece  $B_1(0) = 1$  iar  $B_2(0) = B_3(0) = B_4(0) = 0$ .

În mod analog putem demonstra că  $V_4$  este punctul final al curbei. Să derivăm ecuația  $\bar{P}(t)$ . Rezultă

$$\bar{P}'(t) = \sum_{i=1}^4 \bar{V}_i \cdot B_i'(t)$$

Deoarece  $B_1'(0) = -3$ ;  $B_2'(0) = 3$ ;  $B_3'(0) = B_4'(0) = 0$  rezultă  $\bar{P}'(0) = 3(\bar{V}_2 - \bar{V}_1)$ , adică latura  $V_1V_2$  a poligonului este tangentă curbei în punctul  $V_1$  de început al curbei. Analog latura  $V_3V_4$  a poligonului este tangentă curbei în punctul final  $V_4$ .

### 9.9.1. Aplicație

Să se determine cubica Bézier definită de poligonul director de vîrfuri  $V_1(0, 0, 0)$ ,  $V_2(1, 0, 0)$ ,  $V_3(0, 0, 1)$  și  $V_4(0, 1, 0)$

Ecuatiile parametrice ale curbei sînt:

$$\begin{aligned} x(t) &= B_2(t) \\ y(t) &= B_4(t) \\ z(t) &= B_3(t); \quad t \in [0, 1] \end{aligned}$$

Imaginea axonometrică a acestei curbe Bézier și a proiecției sale orizontale este reprezentată în figura 9.26.

### 9.9.2. Curba Bézier determinată de un poligon cu $n$ vîrfuri

Această curbă are ca ecuație polinomul de grad  $n - 1$

$$\bar{P}(t) = \sum_{i=1}^n \bar{V}_i B_i^n(t); \quad t \in [0, 1]$$

Vectorii  $\bar{V}_i$  sînt determinați de vîrfurile poligonului iar  $B_i^n$  sînt funcțiile de gradul  $n - 1$ :

$$B_i^n(t) = \binom{n-1}{i-1} t^{i-1} (1-t)^{n-i}; \quad i = 1, 2, \dots, n$$

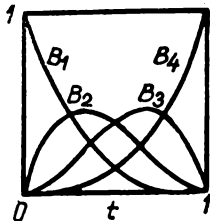


Fig. 9.25

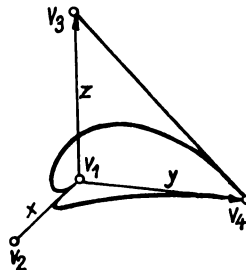


Fig. 9.26

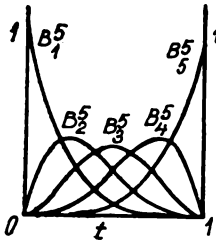


Fig. 9.27

De exemplu 5 vîrfuri definesc curba de gradul 4 a cărei ecuație este pentru  $n = 5$

$$B_1^5(t) = (1 - t)^4$$

$$B_3^5(t) = 6t^2(1 - t)^2$$

$$B_5^5(t) = t^4$$

$$B_2^5(t) = 4t(1 - t)^3$$

$$B_4^5(t) = 4t^3(1 - t)$$

Pentru  $t \in [0, 1]$  (figura 9.27).

Pentru  $n = 4$  putem scrie direct ecuația folosind expresiile de mai sus în care scriem, de exemplu,  $B_2$  în locul lui  $B_2^4$  (din funcțiile cubice).

Valorile funcției  $B_i^n$ ,  $i = 1, 2, 3, \dots, n$  sînt pozitive în intervalul  $[0, 1]$ .

## 9.10. Cubica Coons

Cubica Coons este determinată printr-un polinom cubic și are ca ecuație:

$$\begin{aligned} \bar{P}(t) &= \frac{\bar{V}_1 C_1(t) + \bar{V}_2 C_2(t) + \bar{V}_3 C_3(t) + \bar{V}_4 C_4(t)}{6} = \\ &= \frac{1}{6} \sum_{i=1}^4 \bar{V}_i C_i(t); \quad t \in [0, 1] \end{aligned}$$

În această ecuație  $\bar{V}_i$  sînt vectorii celor 4 vîrfuri  $V_i$  ale poligonului caracteristic al cubicei iar  $C_i$  sînt funcțiile cubice reprezentate în figura 9.28 și care au ca ecuații:

$$C_1(t) = (1 - t)^3$$

$$C_2(t) = 3t^3 - 6t^2 + 4$$

$$C_3(t) = -3t^3 + 3t^2 + 3t + 1$$

$$C_4(t) = t^3; \quad t \in [0, 1]$$

Poligonul caracteristic  $V_1, V_2, V_3, V_4$  are pentru cubică următoarea semnificație geometrică:

Dacă  $C_1(0) = 1$ ;  $C_2(0) = 4$ ;  $C_3(0) = 1$ ;  $C_4(0) = 0$  rezultă

$$\bar{P}(0) = \frac{\bar{V}_1 + 4\bar{V}_2 + \bar{V}_3}{6}$$

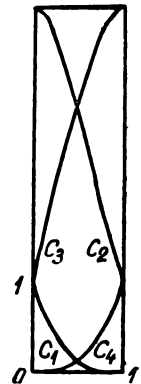


Fig. 9.28

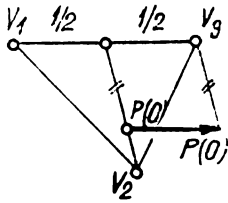


Fig. 9.29

Punctul  $P(0)$  este originea vectorului tangent  $\bar{P}(0)$  care este paralel cu latura  $V_1V_3$  și are ca lungime jumătatea segmentului  $V_1V_3$ . Într-adevăr, să derivăm ecuația  $P(t)$ . Obținem

$$\bar{P}'(t) = \frac{1}{6} \sum_{i=1}^4 \nabla_i C'_i(t)$$

Avem de asemenea  $C'_1(0) = -3$ ;  $C'_2(0) = 0$ ;  $C'_3(0) = 3$  și  $C'_4(0) = 0$  ceea ce probează afirmația făcută.

Analog tangenta în punctul  $P(1)$  este paralelă cu latura  $V_2V_4$  și vectorul tangent  $\bar{P}'(1)$  are ca lungime jumătatea segmentului  $V_2V_4$ . Analizând figurile 9.29 și 9.30 pot fi observate și alte relații care leagă punctele  $v_1v_2v_3v_4$  și vectorii  $\bar{V}_1\bar{V}_2\bar{V}_3\bar{V}_4$ . Astfel  $\bar{V}_1c = \bar{V}_1 + 4\bar{V}_2 + \bar{V}_3$  deoarece vectorul  $V_1c$  are componentele  $\bar{V}_1V_3$  și  $4\bar{V}_1V_2$ . De asemenea  $V_1A = AB = BC$  și  $EV_3 = 3EA$  deci în triunghiul  $0 = \bar{V}_1V_2V_3$  avem  $V_2D = 3V_2P(0)$ .

Mai departe alte calități ale cubicei Coons pot fi urmărite pe figurile 9.31 a, b, c și d în cazul în care punctele  $v_1, v_2, v_3, v_4$  sau toate patru sînt situate pe o dreaptă. De exemplu în cazul b)  $v_2 = \frac{v_1v_3}{2}$ ,  $P(0) = v_2$  și tangenta în  $P(0)$  este dreapta  $V_1V_3$ . Analog dacă  $V_3 = \frac{V_2V_4}{2}$  și  $P(1) = V_3$  atunci tangenta în  $P(1)$  este  $v_2v_4$ . În cazul c) atât  $P(0)$  cît și  $P(1)$  aparțin dreptei.

În cazul d) avem  $v_1 = v_2$  și rezultă  $v_1P(0) = 1/6v_1v_3$ .

Dacă  $v_1 = v_2 = v_3$  atunci  $P(0) = v_1$  și arcul este segmentul  $P(0)P(1) = 1/6v_1v_4$ .

Punctele  $v_1 = v_2$  sau  $v_1 = v_2 = v_3$  pot fi considerate ca puncte duble, respectiv triple pentru cubica Coons.

### 9.10.1. Observație comparativă asupra cubicelor

Analizînd cubicele Bézier și Coons definite de cele patru puncte  $V_1, V_2, V_3$  și  $V_4$  precum și cubica Ferguson definită de punctele  $P_0$  și  $P_1$  și de vectorii tangenți  $P'_0$  și  $P'_1$  în aceste puncte, constatăm:

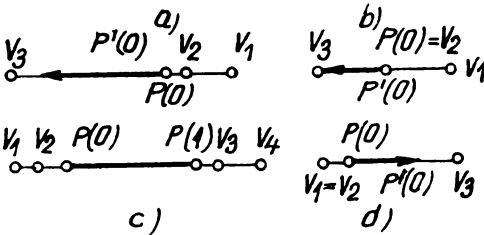


Fig. 9.30

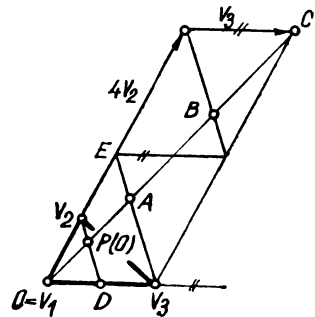


Fig. 9.31

1) Cubica Bézier este cubica Ferguson în care:

$$\bar{P}_0 = \bar{V}_1; \quad \bar{P}_1 = \bar{V}_4; \quad \bar{P}'_0 = 3(\bar{V}_2 - \bar{V}_1); \quad \bar{P}'_1 = 3(\bar{V}_4 - \bar{V}_3)$$

2) Cubica Coons este cubica Ferguson în care:

$$\bar{P}_0 = \frac{\bar{V}_1 + 4\bar{V}_2 + \bar{V}_3}{6}; \quad \bar{P}_1 = \frac{\bar{V}_2 + 4\bar{V}_3 + \bar{V}_4}{6}$$

$$\bar{P}'_0 = \frac{\bar{V}_3 - \bar{V}_1}{2}; \quad \bar{P}'_1 = \frac{\bar{V}_4 - \bar{V}_2}{2}$$

### 9.10.2. Aplicație

Să se determine cubica Coons definită prin punctele

$$V_1(0, 0, 0), V_2(1, 0, 0), V_3(0, 0, 1) \text{ și } V_4(0, 1, 0)$$

Din ecuația cubicei Coons determinăm ecuațiile parametrice ale curbei:

$$6x(t) = 3t^3 - 6t^2 + 4; \quad 6y(t) = t^3; \quad 6z(t) = -3t^3 + 3t^2 + 3t + 1$$

Din relația dintre punctul  $V_2$  și triunghiul  $V_1 V_2 V_3$  rezultă:

$$P(0) = \left( \frac{4}{6}, 0, \frac{1}{6} \right)$$

Pentru punctul final al curbei rezultă  $P(1) = \left( \frac{1}{6}, \frac{1}{6}, \frac{4}{6} \right)$  din relația dintre punctul  $v_3$  și triunghiul  $V_2 V_3, V_4$ .

Prin derivarea ecuațiilor parametrice avem:

$$x'(t) = \frac{3t^2}{2} - 2t; \quad y'(t) = \frac{t^2}{2}; \quad z'(t) = \frac{-3t^2}{2} + t + \frac{1}{2}$$

Astfel vectorul tangent în punctul  $P(0)$  este  $\bar{P}'(0) = \left[ 0, 0, \frac{1}{2} \right]$  are lungimea  $\frac{v_1 v_3}{2}$  și este paralel cu latura  $v_1 v_3$ .

Vectorul tangent în punctul  $P(1)$  este

$\bar{P}'(1) = \left[ -\frac{1}{2}, \frac{1}{2}, 0 \right]$  este paralel cu dreapta  $V_2 V_4$  și are lungimea  $\frac{v_2 v_4}{2}$ . Aceste relații pot fi urmărite pe imaginea axonometrică din figura 9.32.

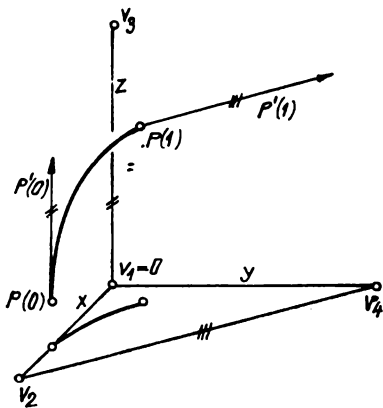


Fig. 9.32

## 9.11. Curba Coons generală

Cubica Coons definită printr-un polinom cubic poate fi generalizată ca o curbă Coons definită de un poligon caracteristic cu  $n > 4$  vîrfuri. Curba  $C$  poate fi compusă din  $(n - 3)$  arce  $c_1, c_2, \dots, c_{n-3}$  corespunzător poligoanelor  $v_1 v_2 v_3 v_4, v_2 v_3 v_4 v_5, \dots, v_{n-3} v_{n-2} v_{n-1} v_n$ .

Alegem pentru curba integrală  $C$  parametrul  $t, 0 \leq t \leq 1$  și pentru orice arc  $C_1, C_2, \dots, C_{n-3}$  parametrul  $s, 0 \leq s \leq 1$

Ecuația curbei  $C_{k+1}$  este (mai bine zis a sumei arcelor):

$$6\bar{P}(t) = \sum_{i=1}^4 \bar{V}_{i+k} C_i(s)$$

în care notăm  $p = t(n - 3)$  pentru  $t \neq 1$  partea zecimală și  $K$  partea întreagă.

Pentru  $t = 1$  avem  $K = n - 4$  și  $S = 1$

Pentru analiza vectorilor tangenți în punctele arcelor de curbă se ține seama de faptul că curba este netedă, adică există identitate între vectorul  $P''(1)$  tangent curbei  $C_k$  și vectorul  $P''(0)$  tangent curbei  $C_{k+1}$  pentru  $K = 1, 2, \dots, n - 3$ .

Din ecuația derivatei a doua pentru curba  $c_k$   $\bar{P}''(t) = \sum_{i=1}^4 \bar{V}_i C_i''(t)$  deducem pentru  $\bar{P}''(1)$

$$C_1''(t) = 6 - 6t; \quad C_2''(t) = 18t - 12; \quad C_3''(t) = -18t + 6; \quad C_4''(t) = 6t$$

deci obținem relația  $\bar{P}''(1) = \bar{V}_{k+1} - 2\bar{V}_{k+2} + \bar{V}_{k+3}$  care este aceeași pentru vectorul  $P''(0)$  al curbei  $c_{k+1}$ .

De aceea curba Coons generală  $C$  aparține curbelor B-spline.

### 9.11.1. Aplicație

Să se determine curba Coons definită prin 10 puncte

$V_1(1, 0, -1); V_2(0, 0, 0), V_3(-1, 0, 1) \quad V_4 = V_5 = V_6(0, 1, -1, 1)$  punct triplu  $V_7(1, -0,5); 0,5; V_8 = V_9 = V_{10}(1,0, 0)$  punct triplu.

Rezolvarea problemei poate fi urmărită pe imaginile perspective din figurile 9.33 și 9.34 ținând seama că în ultima figură punctul  $V_4(-1, -1, 1)$  nu mai este identic cu punctele  $V_4 = V_6$  deci curba își schimbă alura în această regiune. Sînt puse în evidență liniile frînte ale poligoanelor strimbe directe precum și imaginile perspective ale proiecțiilor horizontale ale curbelor Coons generale.

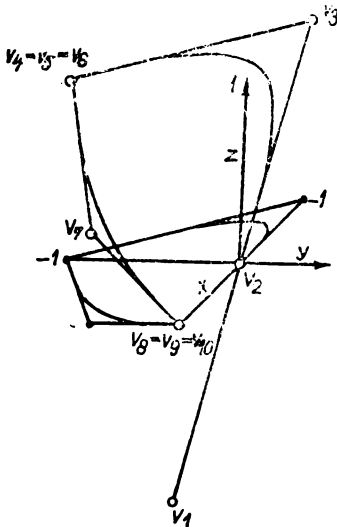


Fig. 9.33

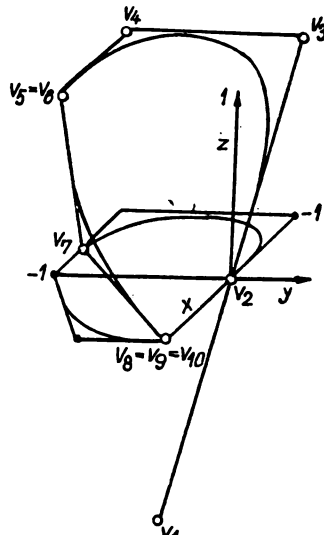


Fig. 9.34



## 9.12. Program principal interpolare (Hermite, Bézier, B-Spline)

```

DIMENSION XF(22),YF(22),XC(50),YC(50),XD(50),YD(50)
LOGICAL*1 NUMFIS(2)
TYPE 222
222 FORMAT(' DISPOZITIV IESIRE: 1--MASA 2--IMPRIMANTA 3--DISC : ',*)
ACCEPT 2,IU
GOTO (223,224,225),IU
223 CALL ASSIGN(1,'PP1:')
GOTO 226
224 CALL ASSIGN(2,'LP:')
GOTO 226
225 TYPE 227
227 FORMAT(' NUME FISIER DISC : ')
ACCEPT 228,NCAR,NUMFIS
228 FORMAT(Q,20A1)
CALL ASSIGN(3,NUMFIS,NCAR)
226 CONTINUE
CALL PLOIIS (0.,0.,IU)
100 TYPE 1
1 FORMAT(' ', 'NR. DE PUNCTE INITIALE : ',*)
ACCEPT 2,NPT
2 FORMAT (I2)
IF(NPT.LE.0)GOTO 100
TYPE 3
3 FORMAT(' ', ' INTRODUCETI COORDONATELE PUNCTELOR CU FORMAT '
*, ' xxx,xx')
DO 7 I=1,NPT
TYPE 4,I
4 FORMAT(' ', ' ABSCISA PUNCT ',I2,' : ',*)
ACCEPT 5,XF(I+1)
5 FORMAT (4F5.2)
TYPE 6,I
6 FORMAT(' ', ' ORDONATA PUNCT ',I2,' : ',*)
7 ACCEPT 5,YF(I+1)
TYPE 8
8 FORMAT(' ', ' NR DE PUNCTE PE SEGMENT : ',*)
ACCEPT 2,NRI
TYPE 9
9 FORMAT(' ', ' INTRODUCETI SCARILE PE X SI Y SI LG-AXE
* [4F5.2] : ',*)
ACCEPT 5,SCX,SCY,XLNTH,YLNTH
CALL FACTOR (SCX,SCY)
CALL AXIS(0.,0.,XLNTH,0.,1.)
CALL AXIS(0.,0.,YLNTH,90.,1.)
38 TYPE 10
10 FORMAT(' ', 'METODA :1. HERMITE 2. BEZIER 3. B-SPLINE 4. EXIT: ',*)
ACCEPT 2,ITIP
GOTO (11,12,13,46),ITIP
C METODA HERMITE
11 TYPE 14
14 FORMAT(' ', 'METODA HERMITE',/,/, ' CONDITII INITIALE :',/,/
*, ' CONDITIA 1:PR.TGX(0)',/,/ ' CONDITIA 2:PR.TGY(0)'
*,/,/ ' CONDITIA 3:PR.TGX(1)',/,/ ' CONDITIA 4:PR.TGY(1)',/,/
23 DO 41 I=1,NPT-1
JJ=0
42 TYPE 15,I
15 FORMAT(' ', ' CONDITIA

```

```

DO 16 J=1,4
TYPE 17,J
17  FORMAT (' ', ' CONDITIA ',I2,' =',,9)
    GOTO (18,19,20,21),J
18  ACCEPT 5,XC(2*I-1)
    GOTO 16
19  ACCEPT 5,YC(2*I-1)
    GOTO 16
20  ACCEPT 5,XC(2*I)
    GOTO 16
21  ACCEPT 5,YC(2*I)
16  CONTINUE
    IF(JJ.EQ.1)GOTO 43
41  CONTINUE
    INDI=1
    INDF=NPT-1
    GOTO 30
C    METODA BEZIER
12  TYPE 22
22  FORMAT(' ', ' METODA BEZIER',/, ' CONDITII INITIALE :',/,
*  ' CONDITIA 1:X(1)',/, ' CONDITIA 2:Y(1)',/, ' CONDITIA
*  3:X(2)',/, ' CONDITIA 4:Y(2)',/)
    GOTO 23
J    METODA B-SPLINE
13  TYPE 24
24  FORMAT(' ', ' METODA B-SPLINE ',/)
    IF (XP(NPT+1).EQ.XP(2).AND.YP(NPT+1).EQ.YP(2)) GOTO 25
28  TYPE 26
26  FORMAT (' DORITI TRECEREA CURBEI PRIN PCT.DE CAPAT ?'
*  , ' [0=NU,1=DA] :',,9)
    ACCEPT 2,IK
    IF (IK.EQ.0) GOTO 29
    IF (IK.NE.1)GOTO 28
    XP(1)=2*XP(2)-XP(3)
    YP(1)=2*YP(2)-YP(3)
    XP(NPT+2)=2*XP(NPT+1)-XP(NPT)
    YP(NPT+2)=2*YP(NPT+1)-YP(NPT)
    GOTO 27
25  XP(1)=XP(NPT)
    YP(1)=YP(NPT)
    XP(NPT+2)=XP(3)
    YP(NPT+2)=YP(3)
27  INDI=1
    INDF=NPT-1
    GOTO 30
29  INDI=2
    INDF=NPT-2
30  DO 31 I=INDI,INDF
    GOTO (32,33,34),ITIP
32  CALL HERM(I+1,XP,XC,CX1,CX2,CX3,CX4)
    CALL HERM(I+1,YP,YC,CY1,CY2,CY3,CY4)
    GOTO 35
33  CALL BEZIER(I+1,XP,XC,CX1,CX2,CX3,CX4)
    CALL BEZIER(I+1,YP,YC,CY1,CY2,CY3,CY4)
    GOTO 35
34  CALL BSPLIN(I+1,XP,CX1,CX2,CX3,CX4)

```

```

      CALL BSPLIN(I+1,YP,CY1,CY2,CY3,CY4)
35  CALL POL3(NRI,XD,CX1,CX2,CX3,CX4)
      CALL POL3(NRI,YD,CY1,CY2,CY3,CY4)
      TYPE 36,I,CX1,CX2,CX3,CX4,CY1,CY2,CY3,CY4
36  FORMAT(' COEFICIENTI AI POLINOAMELOR ',I2,' :',/,
* ' POLINOM X(T) ',5X,4F12.4/
* , ' POLINOM Y(T) ',5X,4F12.4///)
      CALL PDATA(XD,YD,I,NRI)
31  CONTINUE
39  TYPE 37
37  FORMAT(' DORITI MODIFICARI IN COND INITIALE ? (0=NU,1=DA)
* :',%)
      ACCEPT 2,IK
      IF(IK.EQ.0)GOTO 38
      IF (IK.NE.1)GOTO 39
      IF (ITIP.EQ.3) GOTO 50
45  TYPE 40,INDI,INDF
40  FORMAT(' NR POLINOM E ',I2,', ',I2,', I:',%)
      ACCEPT 2,I
      JJ=1
      GOTO 42
43  TYPE 44
44  FORMAT(' DORITI ALTE MODIFICARI ? (0=NU,1=DA) :',%)
      ACCEPT 2,IK
      IF(IK.EQ.0)GOTO 30
      IF(IK.NE.1)GOTO 43
      GOTO 45
46  CALL PLOT(0.,0.,999)
      XSTOP
50  TYPE 51,I,NPT
51  FORMAT(' NUMAR PUNCT [',I2,', ',I2,', ]:',%)
      ACCEPT 2,IK
      IF(IK.EQ.0) GOTO 28
      TYPE 52,IK
52  FORMAT(' ABSCISA SI ORDONATA PUNCT ',I2,' [2F5.2]:',%)
      ACCEPT 5,XP(IK+1),YP(IK+1)
      GOTO 50
      END

      C
0001  SUBROUTINE HERM(I,V,W,C1,C2,C3,C4)
0002  DIMENSION V(1),W(1)
0003  C1=2.*V(I)-2.*V(I+1)+W(2*I-3)+W(2*I-2)
0004  C2=-3.*V(I)+3.*V(I+1)-2.*W(2*I-3)-W(2*I-2)
0005  C3=W(2*I-3)
0006  C4=V(I)
0007  RETURN
0008  END

      C
0001  SUBROUTINE BEZIER(I,V,W,C1,C2,C3,C4)
0002  DIMENSION V(1),W(1)
0003  C1=-V(I)+V(I+1)+3.*W(2*I-3)-W(2*I-2)
0004  C2=3.*V(I)-4.*W(2*I-3)+3.*W(2*I-2)
0005  C3=-3.*V(I)+3.*W(2*I-3)
0006  C4=V(I)
0007  RETURN
0008  END

```

```
      C
0001      SUBROUTINE BSPLIN(I,V,C1,C2,C3,C4)
0002      DIMENSION V(1)
0003      C1=(-V(I-1)+3.*V(I)-3.*V(I+1)+V(I+2))/6.
0004      C2=(3.*V(I-1)-6.*V(I)+3.*V(I+1))/6.
0005      C3=(-3.*V(I-1)+3.*V(I+1))/6.
0006      C4=(V(I-1)+4.*V(I)+V(I+1))/6.
0007      RETURN
0008      END
```

```
      C
0001      SUBROUTINE POL3(NRI,V,C1,C2,C3,C4)
0002      DIMENSION V(1)
0003      DEL=1./(NRI-1)
0004      V0=DEL*DEL
0005      D3V0=6.*C1*DEL*V0
0006      D2VK=D3V0+2.*C2*V0
0007      D1VK=C1*V0*DEL+C2*V0+C3*DEL
0008      V(1)=C4
0009      DO 1 K=2,NRI
0010      V(K)=V(K-1)+D1VK
0011      D1VK=D1VK+D2VK
0012      D2VK=D2VK+D3V0
0013      RETURN
0014      END
```

### 10.1. Principii generale de construcție a suprafețelor în grafica pe calculator.

În general, formele suprafețelor pot fi definite prin ecuații în mod analitic (cilindri, sfere, hiperboloizi, etc.). Însă în cele mai multe cazuri, o astfel de definiție analitică nu este dată iar proiectantul trebuie să creeze suprafețe din elemente mai simple, cum ar fi de exemplu curbele sau punctele.

Problema pe care o vom aborda rezultă din nevoile industriei (de automobile, aeronautică, navală etc.). Când se concepe un prototip este nevoie să se fabrice o machetă pentru a verifica dacă proiectele stilștilor sînt realizabile. Există mașini capabile să fabrice astfel de machete, cu condiția de a putea furniza în prealabil „programul” de comandă (dirijare). Aceasta înseamnă a dispune de ecuații ale diferitelor suprafețe care compun obiectul sau, mai exact de o bună aproximare a acestora. Diferitele tipuri de suprafețe pe care le vom studia se bazează pe un principiu de aproximare prin carouri, fiecare carou fiind definit prin patru curbe limită. Calitățile așteptate de la diferitele modele matematice sînt: întreținerea (asigurarea) unor bune elemente fizice printr-o bună comportare a derivatelor prime (puncte de racordare a carourilor) și a derivatelor secunde (curbura suprafețelor); facilitatea de modificare a alurei suprafețelor pornind de la parametri cu o semnificație fizică ușor de înțeles și de manipulat de către ne-matematicieni; facilitate de calcul, aceste metode fiind destinate unei folosiri interactive. În prezent sînt utilizate trei mari tipuri de suprafețe: *suprafețele lui Coons*, *suprafețele lui Bézier* și *suprafețele B-spline*. Ele au apărut în această ordine cronologică, fiecare aducînd o ameliorare considerabilă în raport cu precedentă.

#### 10.1.1. Suprafețele Coons. Modelarea suprafețelor

Privind interpolarea curbelor, am arătat că există o varietate de metode prin care pot fi construite curbe, prin interpolarea sau aproximarea unui set dat de valori scalare, specificînd punctele sau derivatele în aceste puncte.

În spațiul tridimensional o curbă a fost reprezentată printr-o funcție vector

$$\vec{P}(t) = [x(t), y(t), z(t)]$$

în reprezentarea parametrică iar algoritmi care produc curbe au fost reprezentați de un operator  $\Phi_i$  aplicat funcției vector  $\vec{P}(t)$  care poartă informațiile

$$Q(t) = \Phi_i P(t)$$

În general sînt folosite numai un număr finit de puncte discrete ale funcției  $P(t)$  sau ale derivatelor.

În mod analog, în spațiul tridimensional, un punct arbitrar pe o suprafață poate fi definit sub forma parametrică, de funcția

$$\bar{P}(u, v) = [x(u, v), y(u, v), z(u, v)]$$

iar metoda de producere a suprafeței poate fi notată simbolic prin

$$Q(u, v) = \Phi_{u,v} \cdot P(u, v)$$

unde  $P(u, v)$  reprezintă datele din care va fi construită funcția  $Q(u, v)$ .

Dacă se fixează parametrul  $v$  și este făcut să varieze parametrul  $u$  punctul corespunzător se deplasează pe o curbă trasată pe suprafață. Dacă se fixează  $u$  și este făcut să varieze  $v$ , punctul corespunzător se deplasează pe o altă curbă trasată pe suprafață. Dacă se alege pentru  $v$  un ansamblu de valori ( $v_0, \dots, v_n$ ) și dacă pentru fiecare din ele este făcut să varieze  $u$  de la  $u_0$  la  $u_n$  se obține o familie de curbe definind suprafața dată.

După cum o curbă complexă poate fi reprezentată printr-o succesiune de segmente de curbe mai simple care se racordează suficient de bine, tot astfel se va reprezenta o suprafață complexă printr-un ansamblu de bucăți de suprafețe adiacente (petice) și care se racordează cel mai bine pentru a da o bună aproximare a suprafeței inițiale. Fiecare porțiune de suprafață este delimitată prin patru segmente de curbe limită (marginale, de frontieră) care pot fi notate  $(0v)$ ,  $(1v)$ ,  $(u0)$ ,  $(u1)$ , cu  $u \in [0, 1]$  și  $v \in [0, 1]$  ( $0v$ ) reprezintă curba generată de un punct  $P(uv)$  pentru  $u = 0$ . Să presupunem atunci că aceste patru curbe sînt date. Se introduc două funcții scalare,  $F_0$  și  $F_1$  de o singură variabilă, continue și monotone pe  $[0, 1]$ .

Se definește atunci o suprafață a cărei ecuație este:

$$Q(uv) = (iv) F_i(u) + (uj) F_j(v) - (ij) F_i(u) F_j(v)$$

Se folosește aici o notație condensată pentru a simplifica scrierea. Indicii  $i$  și  $j$  sînt cu valoare pe  $[0, 1]$ . Cînd o expresie comportă unul sau mai mulți indici, trebuie înțeleasă ca o sumă a valorilor posibile ale indicilor. Astfel, primul termen din ecuația suprafeței se scrie  $(iv) F_i(u) = (0v) F_0(u) + (1v) F_1(u)$ . Membrul din dreapta al ecuației cuprinde, deci, opt termeni

Se presupune, în plus, că  $F_0$  și  $F_1$  verifică

$$\begin{array}{ll} F_0(0) = 1 & F_0(1) = 0 \\ F_1(0) = 0 & F_1(1) = 1 \end{array}$$

Se poate de monstra că cele patru curbe limită date aparțin suprafeței.

Pentru a crea o suprafață complexă după Coons, se *juxtapun* porțiuni de suprafață.

Să considerăm cele două porțiuni de suprafață  $A$  și  $B$  (fig. 10.1a) pentru ca să existe continuitatea suprafeței pe  $A$  și  $B$  trebuie să avem  $A(1v) = B(0v)$ . Se poate cere, de asemenea, o continuitate a tangentelor între  $A$  și  $B$ . Pentru a realiza aceasta, căutăm să evităm existența a două semi-plane tangente distincte de-a lungul curbei limită comune (fig. 10.1 b), astfel ca să nu existe discontinuitate.

Se poate arăta că dacă se adaugă constrîngerea pe  $F_0$  și  $F_1$  ca derivatele lor prime să fie nule în punctele  $0$  și  $1$  atunci derivata în direcția lui  $u$ , de-a lungul curbei limită  $(0v)$ , de exemplu, nu depinde decît de derivatele în direcția lui  $u$  în punctele extreme ale curbei limită. Avem deci (fig. 10.1):

$$A(1v)_u = B(0v)_u; A(10)_u = B(00)_u \text{ și } A(11)_u = B(01)_u$$

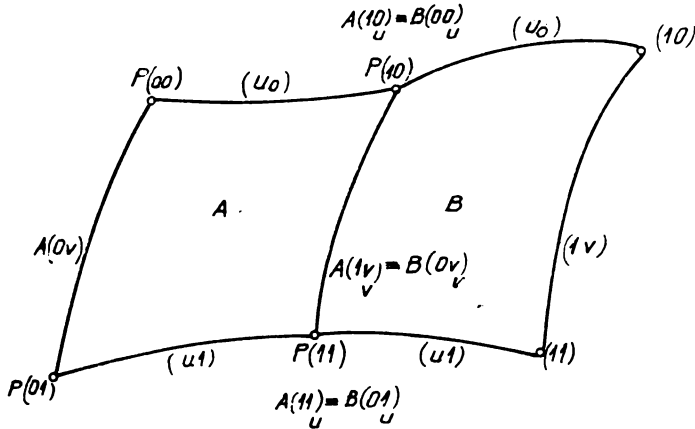


Fig. 10.1-a

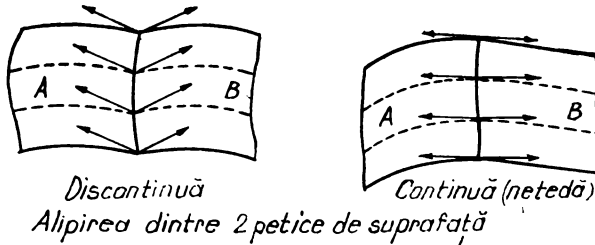


Fig. 10.1-b

cu notația

$$\frac{\partial Q}{\partial u} = (uv)_u$$

### 10.1.2. Suprafața de corecție a tangentelor

Ecuția în discuție descrie o suprafață destul de generală în sensul în care aceasta poate fi definită prin orice fel de curbă limită. Totuși, luând cele patru curbe limită s-a precizat o suprafață ale cărei derivate în cele două direcții pe curbele limită sînt fixate, deoarece ele se exprimă în funcție de cele opt tangente la colțuri. Această proprietate este, desigur, restrictivă și am dori să putem construi suprafețe care au derivate oarecare de-a lungul celor patru curbe inițiale (în afara celor patru colțuri în care ele sînt impuse). Se ia, atunci, o nouă ecuație de suprafață definită pornind de la vectori dați arbitrar și care este notată apriori:

$$\left\{ \frac{\partial Q}{\partial u} \right\}_{u=0} = (0v)_u \left( \frac{\partial Q}{\partial u} \right)_{u=1} = (1v)_u \text{ cu } \left( \frac{\partial Q}{\partial u} \right)_{v=0} = \left( \frac{\partial Q}{\partial u} \right)_{v=1} = \bar{0}.$$

și

$$\left( \frac{\partial Q}{\partial u} \right)_{v=0} = \left( \frac{\partial Q}{\partial u} \right)_{v=1} = \bar{0}; \quad \left( \frac{\partial Q}{\partial v} \right)_{v=0} = (u0)_v \left( \frac{\partial Q}{\partial v} \right)_{v=1} = (u1)_v$$

cu aceleași condiții la limite ca mai sus. Se ia, de asemenea, pentru cele patru colțuri:

$$(ij)_{uv} = \left( \frac{\partial^2 Q}{\partial u \partial v} \right)_{\substack{u=i \\ v=j}}$$

În rezumat, fiecărui punct al perimetrului limită — îi este asociat un vector. Se definește, atunci, o nouă suprafață a cărei ecuație este următoarea:

$$Q(uv) = (iv)_u G_i(u) + (uj)_v G_j(v) - (ij)_{uv} G_i(u) G_j(v)$$

Dacă se impun funcțiilor  $G_0$  și  $G_1$  condițiile următoare

$$G_0(0) = G_0(1) = G_1(0) = G_1(1) = 0$$

$$G'_0(0) = G'_1(0) = 1; \quad G'_0(1) = G'_1(1) = 0$$

arătam că, atunci cînd se suprapun cele două suprafețe (suprafață avînd ca ecuație suma lor se obține o suprafață care se sprijină pe curbele limită inițiale. Cum, în plus, vectorii care au fost luați apriori pentru a defini această suprafață corectivă reprezintă derivatele direcționale ale acesteia, deducem de aici procedeul de calcul următor: se calculează mai întii, mulțumită primei suprafețe o suprafață care se sprijină pe cele patru curbe inițiale. Această suprafață admite, atunci, derivata pe limite. Se face diferența între aceste derivate și cele pe care am dori să le obținem. Aceasta dă, atunci, o familie de vectori care permit să se definească o suprafață care desenată va, avea, atunci, ca ecuație

$$Q(uv) = S(uv) + C(uv)$$

adică compunerea unei suprafețe inițiale și a unei suprafețe corective.

### 10.1.3. Notarea matricială

Unul din avantajele reprezentării parametrice a unei suprafețe prin care utilizatorul posedă un control complet este alegerea corespunzătoare a parametrizării.

Prin subseturile unui domeniu dat

$$[u_{min}, u_{max}] \times [v_{min}, v_{max}]$$

se pot ușor stabili secțiunile unei suprafețe. Aceasta reprezintă o caracteristică indispensabilă pentru toate situațiile în care o suprafață urmează să fie compusă dint-un număr de elemente sau porțiuni de suprafață. Această libertate a parametrizării poate fi folosită pentru a alege

$$[0, 1] \times [0, 1]$$

ca domeniu al unei suprafețe de interpolare sau aproximare. Orice alt domeniu poate fi normalizat în mod corespunzător. Cea mai largă aproximare folosită în această problemă este să se formeze produsul a doi operatori unidimensionali, conform aproximării suprafeței

$$Q(u, v) = \Phi_u \cdot \Phi_v \cdot P(u, v)$$

Rezultatul acestei operații este acela că  $\Phi_v$  operează asupra informațiilor  $P(u, v_j)$ , în timp ce  $\Phi_u$  operează simultan asupra informațiilor  $P(u_i, v)$ . Operatorii  $\Phi_u$  și  $\Phi_v$  sînt comutativi deci



$$\Phi_u \cdot \Phi_v = \Phi_v \cdot \Phi_u$$

dacă  $P(u, v)$  este o funcție continuă.

Dacă sînt folosite un număr finit de valori  $P(u_i, v_j)$  ale funcției  $P(u, v)$  avem **suprafața generală Coons (B-spline)**

$$Q(u, v) = \sum_{j=0}^n \sum_{i=0}^m P(u_i, v_j) \cdot S_{i,m}(u) \cdot T_{j,n}(v)$$

unde  $S_{i,m}(u)$  și  $T_{j,n}(v)$  interpolatează sau aproximează funcțiile unidimensionale. De obicei  $S$  și  $T$  sînt funcții de același tip (de exemplu interpolarea Lagrange).

Deoarece funcțiile de interpolare trebuie să satisfacă condițiile:

$$S_{i,m}(u_k) = \delta_{i,k} \text{ și } T_{j,n}(v_k) = \delta_{j,k}$$

unde  $\delta_{i,k}$  și  $\delta_{j,k}$  sînt *funcțiile delta* ale lui *Kronecker*, este evident că suprafața construită  $Q(u, v)$  se potrivește prin setul de puncte

$$\{(u_i, v_j) \mid i, j \in [0 : m] \times [0 : n]\}$$

Dacă  $P(u, v)$  reprezintă o suprafață de aproximat prin  $Q(u, v)$ , eroarea de aproximare dispare numai în aceste puncte.

În notarea matricială ecuația Coons generală se poate scrie

$$Q(u, v) = S \cdot P \cdot T^T$$

unde  $S$  și  $T$  sînt vectorii funcțiilor de interpolare, adică

$$S = [S_{0,m}(u) \quad S_{1,m}(u) \quad \dots \quad S_{m,m}(u)]$$

$$T = [T_{0,n}(v) \quad T_{1,n}(v) \quad \dots \quad T_{n,n}(v)]$$

$T^T$  este transpusa matricei  $T$ ,

iar  $P$  este matricea restricțiilor, adică

$$P = \begin{vmatrix} P(u_0, v_0) & P(u_0, v_1) & \dots & P(u_0, v_n) \\ P(u_1, v_0) & P(u_1, v_1) & \dots & P(u_1, v_n) \\ \vdots & \vdots & \ddots & \vdots \\ P(u_m, v_0) & P(u_m, v_1) & \dots & P(u_m, v_n) \end{vmatrix}$$

Domeniul de operare

$$\Phi_{u,v} = \Phi_u \cdot \Phi_v$$

este o rețea reticulară de puncte, iar interpolarea se face exact în aceste puncte. Acest fapt este ilustrat în fig. 10.2. O simplă adăugare a celor doi operatori  $\Phi_u(u)$  și  $\Phi_v(v)$  va da în punctele de intersecție ale celor două familii de curbe interpolate, de două ori valoarea aproximată. Pentru a corecta acest fapt trebuie executată următoarea operație:

$$\begin{aligned} Q(u, v) - [\Phi_u \oplus \Phi_v] P(u, v) &= \\ &= [\Phi_u + \Phi_v - \Phi_u \Phi_v] P(u, v) \end{aligned}$$

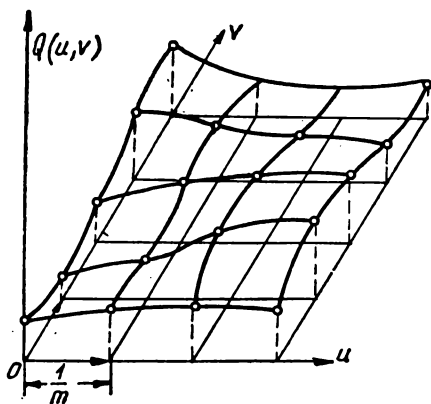


Fig. 10.2

Dacă  $\Phi$  indică setul de transformări lineare dintr-un spațiu liniar într-un subspațiu, se poate arăta că structura algebrică  $(\Phi, \{., \oplus\})$  este o rețea distributivă. Dacă sînt definite o identitate și un complement, structura devine o algebră booleeană. Gordon a demonstrat că aproximarea

$$\Phi_{u,v} = \Phi_u \cdot \Phi_v$$

este minimă deoarece este interpolată într-un număr minim de puncte — punctele reticulare — în timp ce aproximarea

$$\Phi_{u,v} = \Phi_u \oplus \Phi_v$$

este maximă, deoarece este interpolată într-un număr maxim de puncte și anume, toate punctele de pe liniile reticulare

$$\Phi_u P(u,v) \text{ și } \Phi_v P(u,v)$$

Ultimul tip de aproximare este numit „aproximarea sumei boleeene”.

Curbele de-a lungul cărora interpolarea este exactă sînt numite „funcții limită” și respectiv „funcții de asociere”.

Fiecare dintre operatorii  $\Phi_u$  și  $\Phi_v$  interpoleză funcțiile unidimensionale și compară aceste funcții cu funcțiile de asociere corespunzătoare.

O altă posibilitate de aproximare a unei suprafețe este să se folosească simplu unul dintre cei doi operatori, aproximînd astfel suprafața printr-una din cele două operații:

$$Q(u,v) = \Phi_v P(u,v_j)$$

sau

$$Q(u,v) = \Phi_u P(u_i,v)$$

Această tehnică este deseori folosită în aplicațiile în care suprafața ce trebuie să fie construită se întinde mult într-o singură direcție (de exemplu un fuselaj de avion).

Poate fi, de asemenea, amintit faptul că problema interpolării unei suprafețe poate fi rezolvată prin existența unei unice aproximări cubice spline.

Astfel se consideră o funcție  $F(u,v)$  și un polinom bicubic pe porțiuni  $Q(u,v) \in C^2$  astfel încît:

$$\begin{aligned} Q(u_i, v_j) &= F(u_i, v_j), \quad \forall i, j \in [0 : n] \\ \frac{\partial Q}{\partial u}(u_i, v_j) &= \frac{\partial F}{\partial u}(u_i, v_j), \quad \forall j \in [0 : n]; i = 0, n \\ \frac{\partial Q}{\partial v}(u_i, v_j) &= \frac{\partial F}{\partial v}(u_i, v_j), \quad \forall i \in [0 : n]; j = 0, n \\ \frac{\partial^2 Q}{\partial u \partial v}(u_i, v_j) &= \frac{\partial^2 F}{\partial u \partial v}(u_i, v_j), \quad i, j = 0, m \end{aligned}$$

Aproximarea este dată de funcția

$$Q(u,v) = \sum_{i=1}^{n+1} \sum_{j=-1}^{n+1} c_{ij} \beta_i(u) \beta_j(v)$$

unde  $\beta_i(u)$  și  $\beta_j(v)$  sînt cubicele B-spline cunoscute.

Coefficienții  $c_{ij}$  pot fi determinați prin rezolvarea unui sistem de ecuații liniare analog cazului bidimensional.

Ca un exemplu de aproximare am arătat în (10.1.1) care este tehnica de alipire (juxtapunere) a suprafețelor introdusă de Coons.

În forma ei cea mai simplă alipirea de suprafețe Coons era determinată de patru curbe limită (marginale),

$$P(u,0), P(u,1), P(0,v), P(1,v)$$

și o interpolare liniară între aceste curbe (o asociere liniară):

$$Q(u,v) = P(u,0)(1-v) + P(u,1)v + P(0,v)(1-u) + \\ + P(1,v)u - P(0,0)(1-u)(1-v) - P(0,1)(1-u)v - P(1,0)u(1-v) - P(1,1)uv$$

Cele două operații

$$\Phi_v P(u,j) = P(u,0)(1-v) + P(u,1)v$$

$$\Phi_u P(i,v) = P(0,v)(1-u) + P(1,v)u$$

sînt date drept cazuri speciale.

Mai general, funcțiile liniare de asociere pot fi înlocuite cu funcții polinomiale de grad mai mare.

Astfel, fie  $(1-u)$  înlocuit în general prin  $I_{00}(u)$

$(1-v)$  înlocuit în general prin  $I_{00}(v)$

$u$  înlocuit în general prin  $I_{01}(u)$

$v$  înlocuit în general prin  $I_{01}(v)$

În această notație primul indice arată poziția caroului unitar al funcției marginale iar al doilea indice arată că funcția asociată este încărcată de o funcție unidimensională care reprezintă o restricție marginală a derivatei de ordinul indicat.

**Avem**

$$Q(u,v) = \sum_{i=0}^1 P(i,v) I_{i0}(u) + \sum_{j=0}^1 P(u,j) I_{j0}(v) - \\ - \sum_{i=0}^1 \sum_{j=0}^1 P(i,j) \cdot I_{i0}(u) I_{j0}(v)$$

Aceste suprafețe simple Coons pot fi îmbinate așadar prin continuitatea poziției. Pentru continuitatea primelor derivate ale funcțiilor, care este esențială în cele mai multe aplicații, definirea suprafeței de alipire trebuie să fie astfel modificată, încît derivatele parțiale să fie, de asemenea, specificate. Astfel, în afara celor patru curbe  $P(u,j)$  și  $P(i,v)$  utilizatorul va putea să specifice în mod necesar și pantele tangentelor la curbele  $P_v(u,j)$  și  $P_u(i,v)$ ;  $i, j \in [0 : 1]$  în extremitățile corespunzătoare virfurilor patratului (caroului) prin:

$$P_u(u,v) = \frac{\partial P}{\partial u} \quad \text{și} \quad P_v(u,v) = \frac{\partial P}{\partial v}$$

Aplicînd operatorii:

$$\Phi_v P(u,j) = P(u,0) I_{00}(v) + P(u,1) I_{10}(v) + P_v(u,0) I_{01}(v) + \\ + P_v(u,1) I_{11}(v)$$

$$\Phi_u P(i,v) = P(0,v) I_{00}(u) + P(1,v) I_{10}(u) + P_u(0,v) I_{01}(u) + \\ + P_u(1,v) I_{11}(u)$$

$$\text{și ținând seama că } P_{uv}(u, v) = \frac{\delta^2 p}{\delta u \delta v}$$

obținem:

$$\begin{aligned} Q(u, v) = & \sum_{i=0}^1 [P(i, v) I_{v0}(u) + P_u(i, v) I_{v1}(u)] + \\ & + \sum_{j=0}^1 [P(u, j) I_{j0}(v) + P_v(u, j) \cdot I_{j1}(v)] - \\ & - \sum_{i=0}^1 \sum_{j=0}^1 [P(i, j) I_{v0}(u) I_{j0}(v) + P_v(i, j) I_{v0}(u) I_{j1}(v) + \\ & + P_u(i, j) I_{v1}(u) I_{j0}(v) + P_{uv}(i, j) I_{v1}(u) I_{j1}(v)] \end{aligned}$$

Sau în formă matricială

$$\begin{aligned} \bar{Q}(u, v) = & [I_{00}(u) \ I_{10}(u) \ I_{01}(u) \ I_{11}(u)] \cdot \begin{bmatrix} P(0, v) \\ P(1, v) \\ P_u(0, v) \\ P_u(1, v) \end{bmatrix} + \\ & + [P(u, 0) \ P(u, 1) \ P_v(u, 0) \ P_v(u, 1)] \begin{bmatrix} I_{00}(v) \\ I_{10}(v) \\ I_{01}(v) \\ I_{11}(v) \end{bmatrix} - \\ & - [I_{00}(u) \ I_{10}(u) \ I_{01}(u) \ I_{11}(u)] \cdot \\ & \cdot \begin{bmatrix} P(0, 0) & P(0, 1) & P_v(0, 0) & P_v(0, 1) \\ P(1, 0) & P(1, 1) & P_v(1, 0) & P_v(1, 1) \\ P_u(0, 0) & P_u(0, 1) & P_{uv}(0, 0) & P_{uv}(0, 1) \\ P_u(1, 0) & P_u(1, 1) & P_{uv}(1, 0) & P_{uv}(1, 1) \end{bmatrix} \cdot \begin{bmatrix} I_{00}(v) \\ I_{10}(v) \\ I_{01}(v) \\ I_{11}(v) \end{bmatrix} \end{aligned}$$

În contrast cu o astfel de construcție a sumei booleene a suprafeței de alipire, utilizarea construcției Carteziene este mult mai simplă:

$$\begin{aligned} \bar{Q}(u, v) = & \sum_{i=0}^1 \sum_{j=1}^1 [P(i, j) I_{i0}(u) I_{j0}(v) + P_u(i, j) I_{i1}(u) I_{j0}(v) + \\ & + P_v(i, j) I_{i0}(u) I_{j1}(v) + P_{uv}(i, j) I_{i1}(u) I_{j1}(v)] \end{aligned}$$

Dacă alegem ca funcții de asociere polinoamele Hermite cubice prezentate anterior

$$\begin{aligned} I_{00}(x) &= 2x^3 - 3x^2 + 1 \\ I_{10}(x) &= -2x^3 + 3x^2 \\ I_{01}(x) &= x^3 - 2x^2 + x \\ I_{11}(x) &= x^3 - x^2 \end{aligned}$$

obținem pentru forma suprafeței Coons matricea

$$\bar{Q}(u, v) = [u^3 u^2 u 1] \cdot \bar{M} \cdot \bar{P} \cdot \bar{M}^T \cdot [v^3 v^2 v 1]^T$$

unde  $P$  este matricea

$$\bar{P} = \begin{bmatrix} P(0,0) & P(0,1) & P_v(0,0) & P_v(0,1) \\ P(1,0) & P(1,1) & P_v(1,0) & P_v(1,1) \\ P_u(0,0) & P_u(0,1) & P_{uv}(0,0) & P_{uv}(0,1) \\ P_u(1,0) & P_u(1,1) & P_{uv}(1,0) & P_{uv}(1,1) \end{bmatrix}$$

iar  $\bar{M}^T$  este matricea transpusă

$$\bar{M} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

#### 10.1.4. Suprafețele B-spline

Suprafețele B-spline pot fi exprimate așa dar prin relația echivalentă

$$\bar{Q}(u, v) = \sum_{i=0}^m \sum_{j=0}^n P_{i,j} M_{i,k}(u) N_{j,q}(v)$$

unde  $P_{i,j}$  ( $i = 0, 1, \dots, m$ ;  $j = 0, 1, \dots, n$ ) sînt vîrfurile poligonului de  $m, n$  vîrfuri, care determină două sisteme parametrice de curbe B-spline în vederea modelării suprafeței iar

$$M_{i,1}(u) = \begin{cases} 1 & x_i \leq u \leq x_{i+1} \\ 0 & \text{altfel} \end{cases}$$

$$M_{i,k}(u) = \frac{(u - x_i) M_{i,k-1}(u)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - u) M_{i+1, k-1}(u)}{x_{i+k} - x_{i+1}}$$

$$N_{j,1}(v) = \begin{cases} 1 & y_j \leq v \leq y_{j+1} \\ 0 & \text{altfel} \end{cases}$$

$$N_{j,q}(v) = \frac{(v - y_j) N_{j,q-1}(v)}{y_{j+q-1} - y_j} + \frac{(y_{j+q} - v) N_{j+1, q-1}(v)}{y_{j+q} - y_{j+1}}$$

cu convenția  $0/0 = 0$ .

Componentele  $x_i, y_j$  ale vectorilor sînt determinate fie pentru curbele  $u$ , fie pentru curbele  $v$ .

În cazul  $k = m, q = n$  se obține suprafața Bézier așa cum vom vedea mai departe.

Cazuri interesante pentru aplicațiile practice întâlnite mai des pot fi obținute pentru

$$k = q = 3 \quad \text{respectiv} \quad k = q = 4$$

### 10.1.5. Suprafețele riglate

Fie curbele limită  $\bar{P}(u, 0)$ ,  $\bar{P}(u, 1)$  și dreptele  $\bar{P}(0, v)$ ,  $\bar{P}(1, v)$  care mărginesc suprafața.

Ecuția suprafeței riglate de interpolare este:

$$[\bar{P}(u, 0) \quad \bar{Q}(u, v) \quad \bar{P}(u, 1)] \cdot [(1-v) \quad -1v]^T = \bar{0}$$

în care  $\bar{P}(u, 0)$ ,  $\bar{P}(u, 1)$  și  $\bar{Q}(u, v)$  exprimă vectorial curbele limită și suprafața de interpolare,  $[\quad]^T$  este transpusa matricii  $[\quad]$  și  $\bar{0}$  este vectorul nul.

Prin înmulțirea matricelor ecuația capătă forma

$$\bar{Q}(u, v) = (1-v) \cdot \bar{P}(u, 0) + v \cdot \bar{P}(u, 1)$$

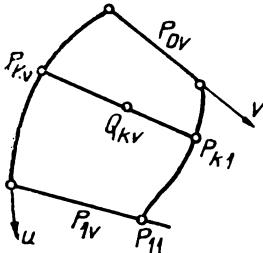


Fig. 10.3

Substituind de exemplu  $v = 0$  obținem  $\bar{Q}(u, 0) = \bar{P}(u, 0)$  și așa mai departe. Curba  $Q_{kv}$  este determinată prin ecuația:

$$\bar{Q}(k, v) = (1-v) \cdot \bar{P}(k, 0) + v \cdot \bar{P}(k, 1)$$

$\bar{P}(k, 0)$  și  $\bar{P}(k, 1)$  sînt vectorii constanți (ai punctelor), iar  $Q_{kv}$  este o dreaptă (fig. 10.3).

Asemănător se poate defini suprafața riglată cu ajutorul curbelor marginale  $P_{0v}$ ,  $P_{1v}$  tot în baza acelorași relații.

Respectiv:

$$[(1-u) \quad -1u] \cdot [\bar{P}(0, v) \quad \bar{Q}(u, v) \quad \bar{P}(1, v)]^T = \bar{0}$$

$$\bar{Q}(u, v) = (1-u) \cdot \bar{P}(0, v) + u \cdot \bar{P}(1, v)$$

Elementele matricii  $[\bar{P}(u, 0) \quad \bar{Q}(u, v) \quad \bar{P}(u, 1)]$  respectiv

$[\bar{P}(0, v) \quad \bar{Q}(u, v) \quad \bar{P}(1, v)]$  sînt vectori și ecuațiile anterioare exprimă condensat cele trei ecuații pentru coordonatele  $x, y$  și  $z$  ale vectorului.

Astfel, de exemplu, pentru coordonata  $z$  matricea este

$$[f(u, 0) \quad z \quad f(u, 1) \quad \text{respectiv} \quad [f(0, v) \quad z \quad f(1, v)]$$

Ecuțiile pot avea forma

$$z = (1-y) \cdot f(x, 0) + y \cdot f(x, 1)$$

$$z = (1-x) \cdot f(0, y) + x \cdot f(1, y)$$

dacă curbele marginale sînt definite explicit,  $z$  fiind cota punctului general  $Q(x, y, z)$  al suprafeței de interpolare. Curbele directe ale suprafeței riglate

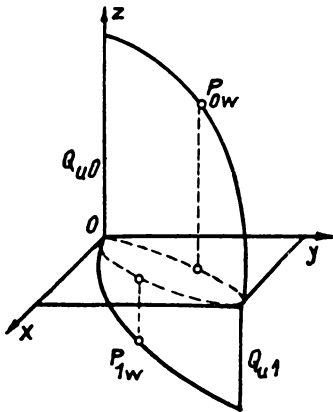


Fig. 10.4

considerate sînt  $P_{0v}$  și  $P_{1v}$  respectiv  $Q_{u0}$  și  $Q_{u1}$  (fig. 10.4).

Astfel fie suprafața riglată determinată de aceste curbe spațiale

$$\bar{P}(0, v) = \left( v, \sin\left(\frac{v\pi}{2}\right), \cos\left(\frac{v\pi}{2}\right) \right)$$

$$\bar{P}(1, v) = \left( v \cdot v^2, \frac{-v}{2} \right),$$

care sînt reprezentate axonometric și prin proiecțiile orizontale ale curbelor  $P_{0,v}$  și  $P_{1,v}$  în figura 10.4 (vezi aplicația din 10.8.6).

Ecuatiile parametriche ale acestei suprafețe sînt

$$x = (1 - u) \cdot v + u \cdot v = v$$

$$y = (1 - u) \cdot \sin\left(\frac{v\pi}{2}\right) + u \cdot v^2$$

$$z = (1 - u) \cdot \cos\left(\frac{v\pi}{2}\right) - u \cdot v/2 \text{ pe domeniul } 0 \leq u, v \leq 1.$$

Punctele de colț au coordonatele

$$P_{00} = (0, 0, 1), P_{10} = (0, 0, 0)$$

$$P_{01} = (1, 1, 0) P_{11} = (1, 1, -1/2)$$

și determină două drepte marginale (limită):

$$\bar{Q}(u, 0) = (1 - u) \cdot \bar{P}(0, 0) + u \cdot \bar{P}(1, 0)$$

$$x = (1 - u) \cdot 0 + u \cdot 0$$

$$y = (1 - u) \cdot 0 + u \cdot 0$$

$$z = (1 - u) \cdot 1 + u \cdot 0$$

și deci

$$\bar{Q}(u, 0) = (0, 0, 1 - u);$$

$$\bar{Q}(u, 1) = (1 - u) \cdot \bar{P}(0, 1) + u \cdot \bar{P}(1, 1)$$

$$x = (1 - u) \cdot 1 + u \cdot 1$$

$$y = (1 - u) \cdot 1 + u \cdot 1$$

$$z = (1 - u) \cdot 0 + u \cdot \left(\frac{-1}{2}\right)$$

Așadar:

$$\bar{Q}(u, 1) = (1, 1, -u/2)$$

### 10.1.6. Aplicație

Să se determine suprafața riglată definită prin parabola

$P_{0y}$ :  $z = uy(1 - y)$  în planul  $x = 0$  și sinusoida

$P_{1y}$ :  $z = \sin(2\pi y)$  în planul  $x = 1$  (fig. 10.5)

Ecuția suprafeței este

$$z = (1 - x) 4y(1 - y) + x \sin(2\pi y)$$

fiind mărginită la placa  $0 \leq x, y \leq 1$

Imaginea perspectivă a suprafeței este redată în figura 10.6

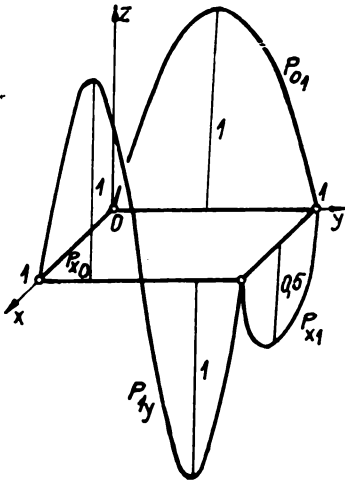


Fig. 10.5

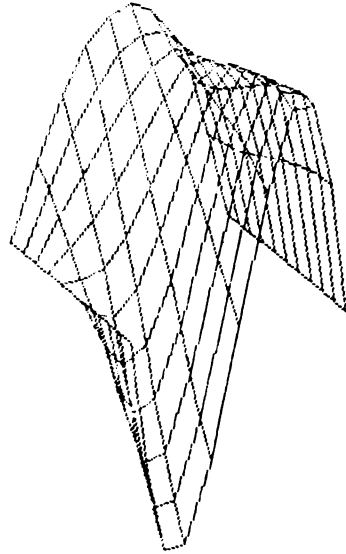


Fig. 10.6

## 10.2. Suprafețe determinate prin polinoame

În grafica pe calculator sînt mult utilizate suprafețele determinate polinomial prin funcția de două variabile  $x$  și  $y$ :

$$z = \sum_{j=0}^n \sum_{i=0}^m a_{ij} x^j y^i$$

Suprafața poate fi notată prin  $P_{m,n}$  sau mai simplu prin  $P$ .  
Funcția  $Z$  poate fi exprimată matricial sub forma

$$Z = \bar{X} \bar{A} \bar{Y}$$



în care vectorii  $\bar{X}$  și  $\bar{Y}$  sînt:

$$\bar{X} = [1, x, x^2, \dots, x^m]$$

$$\bar{Y} = [1, y, y^2, \dots, y^m]^T \text{ (matricea transpusă)}$$

iar matricea  $\bar{A}$  este:

$$\bar{A} = \begin{bmatrix} a_{00}, & a_{01}, & \dots, & a_{0m} \\ a_{10}, & a_{11}, & \dots, & a_{1,m} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ a_{m,0} & a_{m,1}, & \dots, & a_{m,n} \end{bmatrix}$$

Această matrice poate fi considerată ca o „amprentă” a suprafeței.

### 10.2.1. Cazuri speciale

Suprafețele  $P_{0,0}$ ,  $P_{0,1}$  și  $P_{1,0}$  sînt plane.

Suprafața  $P_{1,1}$  este paraboloidul hiperbolic.

$$z = a_{00} + a_{10}x + a_{01}y + a_{11}xy$$

ale cărei generatoare sînt paralele cu planele  $(xz)$  și  $(yz)$ .

Suprafețele  $P_{0,n}$  respectiv  $P_{m,0}$  pentru  $m \geq 2$ ,  $n \geq 2$  sînt suprafețe cilindrice definite prin funcțiile:

$$Z = \sum_{j=0}^n a_{0j}y^j \quad \text{respectiv} \quad Z = \sum_{i=0}^m a_{i0}x^i$$

Generatoarele acestor suprafețe sînt paralele cu axa  $X$  respectiv  $Y$ , iar curbele directoare ale suprafețelor cilindrice sînt conținute în planele  $(YZ)$  respectiv  $(XZ)$ .

Suprafețele  $P_{1,n}$  respectiv  $P_{m,1}$  pentru  $m, n \geq 1$  sînt suprafețele riglate

$$z = \sum_{j=0}^n \sum_{i=0}^1 a_{ij}x^i y^j$$

$$\text{respectiv:} \quad z = \sum_{j=0}^1 \sum_{i=0}^m a_{ij}x^i y^j$$

ale căror generatoare sînt conținute în planele  $y = \text{Konstant}$ , respectiv  $x = \text{Konstant}$ .

### 10.2.2. Curbele principale ale suprafeței $P_{m,n}$

Curbele principale ale suprafeței  $P_{m,n}$  sînt secțiunile plane paralele cu planele  $(YZ)$  și respectiv  $(XZ)$  obținute prin intersecția suprafeței cu planele  $x = X$  respectiv  $y = Y$ .

Curbele principale sînt curbe algebrice de gradul  $n$ , respectiv  $m$ .

Ecuțiile lor sînt următoarele:

$$z = \sum_{j=0}^n a_j y^j; \quad a_j = \sum_{i=0}^m a_{ij} X^i; \quad j = 1, 2, \dots, m$$

respectiv

$$z = \sum_{i=0}^m b_i x^i; \quad b_i = \sum_{j=0}^n a_{ij} Y^j; \quad i = 1, 2, \dots, n$$

Ecuțiile curbelor principale ale suprafeței  $P_{m,n}$  în notație matricială sînt:

$$\boxed{Z = \bar{X}_i \bar{A} \bar{Y}_j} \quad \text{respectiv} \quad \boxed{z = \bar{X} \bar{A} \bar{Y}_j}$$

unde:

$$\bar{x}_i = [1, x_i, \dots, x_i^m]$$

$$\bar{y}_j = [1, y_j, \dots, y_j^n]^T$$

iar matricea  $\bar{A}$  este:

$$\bar{A} = \begin{bmatrix} a_{00} & a_{01} & \dots & a_{0n} \\ a_{10} & a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ a_{m0} & a_{m1} & \dots & a_{mn} \end{bmatrix}$$

### 10.2.3. Curbele marginale

Să alegem  $0 \leq x \leq 1$  și  $0 \leq y \leq 1$ . Scriind  $0 \leq x$  și  $y \leq 1$  obținem pentru curbele din planele;

$$x = 0$$

$$y = 1$$

$$y = 0$$

$$x = 1$$

adică curbele principale marginale sau de frontieră ale suprafeței (placă) corespunzătoare domeniului plan dat.

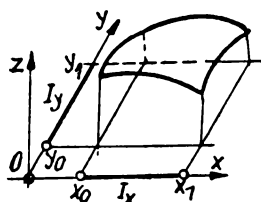


Fig. 10.7

Ecuțiile curbelor marginale ale suprafeței  $P_{m,n}$  sînt următoarele (fig. 10.7):

$$Z = \sum_{i=0}^n a_{0i} y^i,$$

$$Z = \sum_{j=0}^m c_j y^j; \quad c_j = \sum_{i=0}^m a_{ij}$$

$$Z = \sum_{i=0}^m a_{i0} x^i$$

$$Z = \sum_{i=0}^m d_i x^i; \quad d_i = \sum_{j=0}^n a_{ij}$$

#### 10.2.4. Punctele de colț ale suprafeței

Aceste curbe marginale sau de frontieră ale plăcii de suprafață alcătuiesc așa numitele „puncte de colț” ale suprafeței  $P_{m,n}$  și au coordonatele

$$(0, 0, a_{00}); \quad \left(1, 0, \sum_{j=0}^m a_{1j}\right)$$

$$\left(0, 1, \sum_{j=0}^n a_{0j}\right); \quad \left(1, 1, \sum_{j=0}^n \sum_{i=0}^m a_{ij}\right)$$

#### 10.2.5. Aplicație

Ecuția

$$Z = (1 - x^2) y$$

reprezintă determinarea suprafeței  $P_{2,1}$ . Dacă considerăm domeniul  $0 \leq x, y \leq 1$ , obținem curbele marginale:

$Z = y$  deci o dreaptă în planul  $x = 0$ , apoi dreapta

$Z = 0$  în planele  $X = 1$  și  $y = 0$  și în sfîrșit parabola

$$Z = 1 - x^2 \text{ în planul } y = 1.$$

Punctele de colț ale suprafeței placă limitată a domeniului plan dat are coordonatele:

$$(0, 0, 0); (1, 0, 0); (0, 1, 1) \text{ și } (1, 1, 0).$$

Imaginea perspectivă a acestei suprafețe împreună cu cîteva curbe principale este dată în figura 10.8.

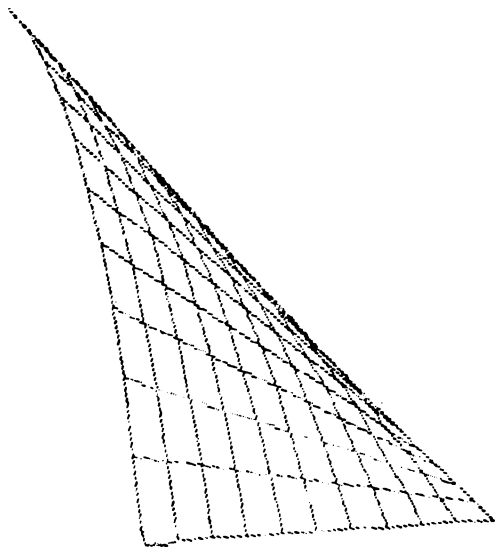


Fig. 10.8

### 10.3. Construcția suprafețelor definite prin puncte și prin curbe marginale

Să considerăm suprafețele determinate prin polinoame sub forma

$$Z_B = \sum_{j=0}^n \sum_{i=0}^m a_{ij} x_B^i y_B^j$$

și punctul  $B(x_B; y_B; z_B)$ . Să căutăm condițiile ca suprafața să treacă prin acest punct s-au printr-o mulțime de puncte inițial date, rezolvînd sistemele de ecuații liniare pentru coeficienții  $a_{ij}$ .

Evident, prin definiție, fiecare punct induce condiții pentru determinarea suprafeței.

Funcția  $Z$  conține  $p = (m + 1)(n + 1)$ , coeficienți care pot fi determinați prin găsirea punctelor de intersecție  $Z_{ij}$  ale curbelor principale rezolvînd sistemul:

$$Z = \bar{X}_i \bar{A} \bar{Y}, \quad i = 0, 1, \dots, m$$

$$\bar{Z} = \bar{X} \bar{A} \bar{Y}_j, \quad j = 0, 1, \dots, n$$

Acest sistem de  $p$  ecuații liniare pentru determinarea coeficienților matricii  $\bar{A}$  poate fi scris matricial

$$\bar{Z} = \bar{X}^T \bar{A} \bar{Y} \quad \text{în care avem}$$

$$\bar{Z} = \begin{bmatrix} z_{00}, z_{01}, \dots, z_{0n} \\ z_{10}, z_{11}, \dots, z_{1n} \\ \dots \\ z_{m,0}, z_{m,1}, \dots, z_{m,n} \end{bmatrix} \quad \bar{X} = \begin{bmatrix} 1, 1, \dots, 1 \\ x_0^1, x_1^1, \dots, x_m^1 \\ \dots \\ x_0^m, x_1^m, \dots, x_m^m \end{bmatrix}$$

$$\bar{Y} = \begin{bmatrix} 1, 1, \dots, 1 \\ y_0^1, y_1^1, \dots, y_n^1 \\ \dots \\ y_0^n, y_1^n, \dots, y_n^n \end{bmatrix}$$

Deoarece există inversele matricelor  $\bar{X}$  și  $\bar{Y}$  putem determina matricea  $\bar{A}$  din ecuația:

$$\bar{A} = (\bar{X}^{-1})^T \bar{Z} \bar{Y}^{-1}$$

În aceste condiții suprafața de interpolare  $P_{m,n}$  care trece prin punctele  $Z_{ij}$  are ecuația:

$$Z = \bar{X}(\bar{X}^{-1})^T \bar{Z} \bar{Y}^{-1} \bar{Y}$$

unde toate matricele au semnificațiile cunoscute.

Fie  $m, n \neq 0$ . Primele două ecuații ale curbelor marginale induc fiecare câte  $m + 1$  condiții, iar ultimele două curbe marginale induc și ele câte  $n + 1$  condiții fiecare.

Curbele marginale care se întîlnesc în punctele de colț induc și ele

$$2(m + 1 + n + 1) - 4 = 2(m + n)$$

condiții pentru ca suprafața să le conțină.

Este astfel posibilă o analiză a numărului de condiții necesare și suficiente pentru suprafața  $P_{m,n}$  ca să conțină (să fie determinată de) curbele marginale.

De exemplu:

$$\frac{P_{1,n}; P_{m,1}}{0} \mid \frac{P_{2,2}}{1} \mid \frac{P_{2,3}; P_{3,2}}{2} \mid \frac{P_{3,3}}{4}$$

Ca aplicație să determinăm suprafețele de interpolare de tipul  $P_{1,2}$ ;  $P_{2,2}$  și  $P_{3,3}$ .

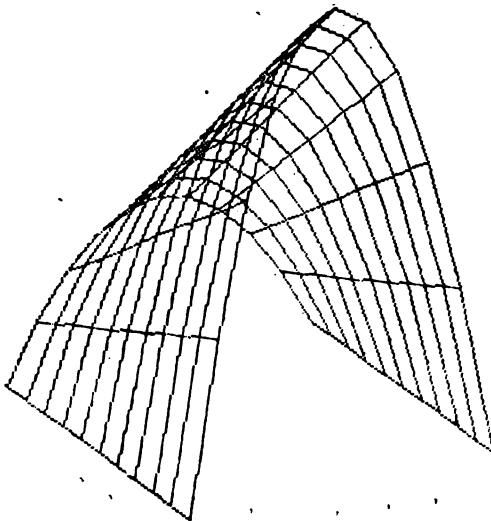


Fig. 10.9

### 10.3.1. Aplicație. Suprafața $P_{1,2}$

Să determinăm suprafața de interpolare de tipul  $P_{1,2}$  care trece prin următoarele 6 puncte:

$$(0; 0; 0); (1; 0; 0); (0; 1; 0)$$

$$(1; 1; 0); (0; 0, 5; a); (0, 5; 0, 5; b)$$

Prin rezolvarea sistemului de șase ecuații liniare cu șase necunoscute obținem coeficienții suprafeței  $P_{1,2}$ , precum și ecuația explicită a suprafeței.

$$Z = y(1 - y) [8(b - a)x + 4a]$$

Pentru  $a = 0,5$  și  $b = 1$  imaginea perspectivă a suprafeței și a citorva curbe principale este reprezentată în figura 10.9.

10.3.2. Aplicație. Suprafața  $P_{2,2}$ 

Să determinăm suprafața de interpolare de tipul  $P_{2,2}$ , definită prin curbele marginale și printr-un punct interior.

Astfel curbele marginale sînt dreptele  $z = 0$

pentru

$$x = 0, x = 1, y = 1$$

și parabola

$z = (1-x)x$  pentru  $y = 0$ . Punctul interior are coordonatele  $(0,5; 0,5; 1)$ .

Din aceste condiții deducem succesiv:

$$0 \equiv a_{00} + a_{01}y + a_{02}y^2 \text{ pentru latura situată pe } oy$$

$$\text{De aceea } a_{00} = a_{01} = a_{02} = 0$$

Analog din ecuația:

$$Z = a_{10}x + a_{20}x^2 \text{ obținem } a_{10} = 4 \text{ și } a_{20} = -4$$

și deci identitatea:

$$[0 \equiv 4 + a_{11}y - 4 + a_{12}y + a_{21}y + a_{22}y^2]$$

Ecuațiile:

$$a_{11} + a_{21} = 0; a_{12} + a_{22} = 0$$

și identitatea

$$0 \equiv 4 + a_{11}x - 4x^2 + [a_{12}x + a_{21}x^2 + a_{22}x^2]$$

Conduc la ecuațiile:

$$a_{11} + a_{12} = -4; a_{21} + a_{22} = 4$$

În sfîrșit, din condiția pentru punctul interior avem:

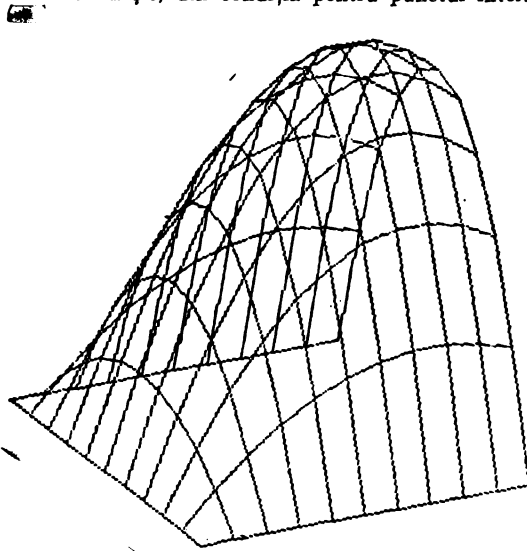


Fig. 10.10

$$1 = \left[ \frac{4}{2} + \frac{a_{11}}{4} - \frac{4}{4} + \frac{a_{12}}{8} + \frac{a_{21}}{8} + \frac{a_{22}}{16} \right]$$

Rezolvînd sistemele obținem valorile:

$$a_{11} = -a_{21} = 4$$

$$a_{12} = -a_{22} = -8$$

Ecuația explicită a suprafeței de interpolare  $P_{2,2}$  este:

$$Z = 4x(1 + y - x - 2y^2 - xy + 2xy^2)$$

Imaginea perspectivă a suprafeței și a citorva curbe principale este reprezentată în fig 10.10

10.3.3. Aplicație. Suprafața  $P_{3,3}$ 

Să determinăm suprafața de interpolare de tipul  $P_{3,3}$  definit prin 16 puncte.

Vom diviza în egală măsură curbele marginale, deci în aceste condiții avem:

$$\bar{X} = \bar{Y} = \begin{bmatrix} 1, & 1, & 1, & 1 \\ 0, & \frac{1}{3}, & \frac{2}{3}, & 1 \\ 0, & \frac{1}{9}, & \frac{4}{9}, & 1 \\ 0, & \frac{1}{27}, & \frac{8}{27}, & 1 \end{bmatrix}; \quad \bar{X}^{-1} = \bar{Y}^{-1} = \begin{bmatrix} 1 - \frac{11}{2} & 9 - \frac{9}{2} \\ 0 & 9 - \frac{45}{2} & \frac{27}{2} \\ 0 & -\frac{9}{2} & 18 - \frac{27}{2} \\ 0 & 1 & -\frac{9}{2} & \frac{9}{2} \end{bmatrix}$$

Alegem mai departe cota pentru  $Z_{33} = K$  și în rest toate cotele  $z_{ij}$  egale cu zero. Deci matricea  $\bar{Z}$  este:

$$\bar{Z} = \begin{bmatrix} 0, & 0, & 0, & 0 \\ 0, & K, & 0, & 0 \\ 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0 \end{bmatrix}$$

În aceste condiții suprafața de interpolare are ecuația:

$$Z = [1, x, x^2, x^3] (X^{-1})^T \bar{X} \bar{Y}^{-1} [1, y, y^2, y^3]^T$$

Obținem în final după înmulțirea matricelor ecuația explicită a suprafeței de interpolare  $P_{3,3}$

$$Z = Kxy(2 - 5x + 3x^2)(2 - 5y + 3y^2)$$

Pentru  $K = 1$  imaginea perspectivei a suprafeței este redată în figura 10.11.

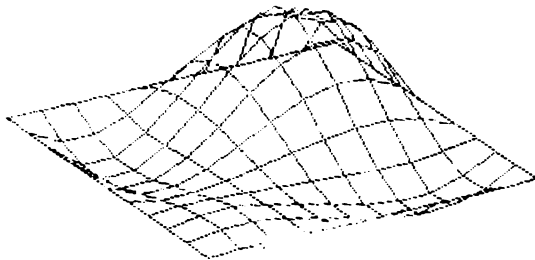


Fig. 10.11

### 10.4. Suprafețe determinate vectorial prin polinoame

Considerăm suprafața determinată polinomial prin funcția dependentă de parametrii  $u$  și  $v$

$$\bar{P}(u, v) = \bar{u}\bar{B}\bar{v}$$

în care vectorii  $\bar{u}$  și  $\bar{v}$  sînt pentru  $0 \leq u \leq 1$  și  $0 \leq v \leq 1$

$$\bar{u} = [1, u, u^2, \dots, u^m]$$

$$\bar{v} = [1, v, v^2, \dots, v^n]^T \quad (\text{matricea transpusă})$$

iar matricea  $\bar{B}$  este:

$$\bar{B} = \begin{bmatrix} \bar{B}_{00} & \bar{B}_{01} & \dots & \bar{B}_{0n} \\ \bar{B}_{10} & \bar{B}_{11} & \dots & \bar{B}_{1n} \\ \dots & \dots & \dots & \dots \\ \bar{B}_{m0} & \bar{B}_{m1} & \dots & \bar{B}_{mn} \end{bmatrix}$$

Elementele acestei matrice care determină suprafața au semnificații geometrice. Curbele parametrice pentru  $u = u_i = \text{Konstant}$ , respectiv  $v = v_j = \text{Konstant}$  sînt polinoamele de gradul  $n$ , respectiv  $m$ .

$$\bar{P}(u_i, v) = \bar{u}_i\bar{B}\bar{v}; \quad 0 \leq u_i \leq 1; \quad u_i = [1, u_i, u_i^2, \dots, u_i^m]$$

$$\bar{P}(u, v_j) = \bar{u}\bar{B}\bar{v}_j; \quad 0 \leq v_j \leq 1; \quad \bar{v}_j = [1, v_j, v_j^2, \dots, v_j^n]^T$$

Punctul asociat acestor curbe parametrice este determinat de ecuația:

$$\bar{P}(u_i, v_j) = \bar{u}_i\bar{B}\bar{v}_j$$

Suprafața căutată dependentă de parametrii  $u_i, v_j$  și care trece prin punctele date  $P_{ij}$  trebuie să îndeplinească condiția

$$\bar{P}_{ij} = \bar{u}_i\bar{B}\bar{v}_j$$

Deoarece în matricea  $\bar{B}$  sînt  $(m+1)(n+1)$  vectori necunoscuți este necesar să rezolvăm sistemul de  $(m+1)(n+1)$  ecuații liniare din care obținem  $(m+1)(n+1)$  puncte  $P_{ij}$  pentru suprafața căutată, respectiv parametrii  $u_0, u_1, \dots, u_m$  și  $v_0, v_1, \dots, v_n$ .



Să notăm:

$$\bar{U} = \begin{bmatrix} 1, & 1, & \dots, & 1 \\ u_0, & u_1, & \dots, & u_m \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ u_0^m, & u_1^m, & \dots, & u_m^m \end{bmatrix}; \quad \bar{V} = \begin{bmatrix} 1, & 1, & \dots, & 1 \\ v_0, & v_1, & \dots, & v_n \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ v_0^n, & v_1^n, & \dots, & v_n^n \end{bmatrix}$$

$$\bar{P} = \begin{bmatrix} \bar{P}_{00}, & \bar{P}_{01}, & \dots, & \bar{P}_{0n} \\ \bar{P}_{10}, & \bar{P}_{11}, & \dots, & \bar{P}_{1n} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \bar{P}_{m0}, & \bar{P}_{m1}, & \dots, & \bar{P}_{mn} \end{bmatrix}$$

Cu această notație ecuația matricială a sistemului se poate scrie sub forma produsului matricelor

$$\bar{P} = \bar{U}^T \bar{B} \bar{V}$$

de unde obținem matricea  $\bar{B}$ , având în vedere existența matricelor  $\bar{U}^{-1}$  și  $\bar{V}^{-1}$ .

Astfel:

$$\bar{B} = (\bar{U}^{-1})^T \bar{P} \bar{V}^{-1}$$

iar ecuația suprafeței de interpolare căutată este:

$$\bar{P}(u, v) = \bar{u} (\bar{U}^{-1})^T \bar{P} \bar{V}^{-1} \bar{v}$$

În această ecuație matricea  $\bar{P}$  are semnificația de „amprentă” a suprafeței.

Nodurile  $P_{ij}$  obișnuite ale rețelei de curbe parametrice sînt similare nodurilor rețelei suprafeței Bézier cu excepția nodurilor de colț ale suprafeței.

Dacă alegem în mod special  $u = x$  și  $v = y$ , ecuația suprafeței de interpolare este exprimată explicit prin:

$$z = [1, x, \dots, x^m] \cdot \bar{A} \cdot [1, y, \dots, y^n]^T$$

în care  $\bar{A}$  este matricea constantelor:

$$\bar{A} = \begin{bmatrix} a_{00}, & a_{01}, & \dots, & a_{0n} \\ a_{10}, & a_{11}, & \dots, & a_{1n} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ a_{m0}, & a_{m1}, & \dots, & a_{mn} \end{bmatrix}$$

și deci ecuația explicitată a suprafeței este:

$$Z = \sum_{j=0}^n \sum_{i=0}^m a_{ij} x^i y^j$$

#### 10.4.1. Aplicație

Să determinăm suprafața de interpolare definită de nouă puncte de intersecție ale curbelor parametrice pentru  $u, v = 0, \frac{1}{2}, 1$  (Suprafață bicuadratică).

Alegem punctele astfel:

$$P(0, 0) = (0, 0, 0); \quad P\left(0, \frac{1}{2}\right) = \left(0, \frac{1}{2}, 0\right)$$

$$P(0, 1) = (0, 1, 0)$$

$$P\left(\frac{1}{2}, 0\right) = \left(\frac{1}{2}, -\frac{1}{2}, 1\right); \quad P\left(\frac{1}{2}, \frac{1}{2}\right) = \left(0, \frac{1}{2}, 1\right)$$

$$\bar{P}\left(\frac{1}{2}, 1\right) = \left(\frac{1}{2}, 1, 0\right); \quad P(1, 0) = (1, 0, 0)$$

$$P\left(1, \frac{1}{2}\right) = \left(1, \frac{1}{2}, 0\right); \quad P(1, 1) = (1, 1, 0)$$

În aceste condiții matriceale  $\bar{U} = \bar{V}$  deci:

$$\bar{U} = \bar{V} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & \frac{1}{2} & 1 \\ 0 & \frac{1}{4} & 1 \end{bmatrix}$$

iar inversele acestor matrice sînt:

$$\bar{U}^{-1} = \bar{V}^{-1} = \begin{bmatrix} 1 & -3 & 2 \\ 0 & 4 & -4 \\ 0 & -1 & 2 \end{bmatrix}$$

Să determinăm ecuațiile parametrice ale suprafeței de interpolare.

Astfel ecuația parametrică pentru coordonata  $x$  este:

$$x = [1, u, u^2] \cdot \begin{bmatrix} 1 & 0 & 0 \\ -3 & +4 & -1 \\ 2 & -4 & 2 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & -3 & 2 \\ 0 & 4 & -4 \\ 0 & -1 & 2 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}$$

și analog se determină  $y$ .

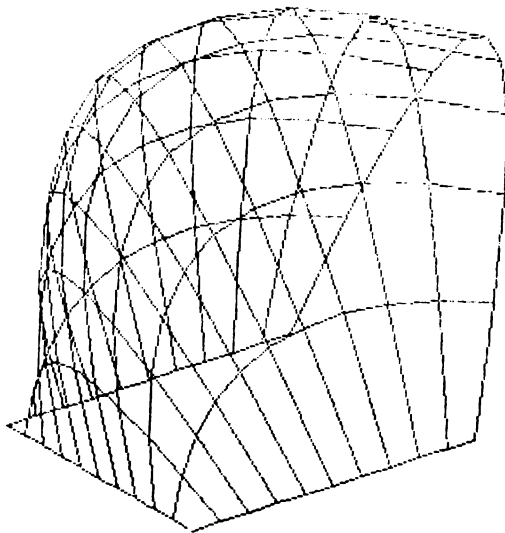


Fig. 10.12

Imaginea perspectivă a suprafeței definite de aceste ecuații parametrice este reprezentată în figura 10.12.

Dacă notăm parametric în funcție de un singur parametru  $t$  funcțiile care definesc suprafața, deci dacă:

$$A_1(t) = 1 - 3t + 2t^2$$

$$A_2(t) = 4t(1 - t)$$

$$A_3(t) = t(2t - 1)$$

Obținem prin înmulțirea matricelor

$$\begin{aligned} x &= \left[ \frac{A_2(u)}{2} + A_3(u) \right] \cdot [A_1(v) + \\ &+ A_3(v)] + A_3(u) \cdot A_3(v) = \\ &= u(1 - 4v + 4v^2) + 4v(2u - 1)(1 - v) \end{aligned}$$

Analog se determină:

$$\begin{aligned} y &= -\frac{1}{2} A_2(u) \cdot A_1(v) + \frac{A_3(v)}{2} + \\ &+ A_3(v) = v - 2u(1 - u)(2v^2 - 3v + 1) \end{aligned}$$

$$\begin{aligned} z &= A_2(u)[A_1(v) + A_3(v)] = \\ &= 4u(1 - u)(1 + v - 2v^2) \end{aligned}$$

## 10.5. Suprafețe plăci

### 10.5.1. Cuplarea (alipirea, racordarea) suprafețelor

Să considerăm alipirea (racordarea, cuplarea, conexiunea) suprafețelor  $A_{m,n}$  și  $B_{m,o}$  definite respectiv pe domeniile:

$A_{m,n}$	$B_{m,o}$	coeficienți
$0 \leq x, y \leq 1$	$0 \leq x \leq 1$ $1 \leq y \leq 2$	$a_{ij}, b_{ij}$

unde:  $i = 0, 1, \dots, m$

$j = 0, 1, \dots, n$

Primul indice comun  $m$  exprimă cerința de alipire.

Să considerăm transformarea:

$$x = u$$

$$y - 1 = v$$

și să definim domeniul suprafeței  $B$  prin  $0 \leq u; v \leq 1$ .

Considerăm, de asemenea, că alipirea peticelor de suprafață se face în lungul curbei marginale (frontieră) din planul  $y = 1$ . Deci curbele marginale ale primei suprafețe  $A$  și ale suprafeței  $B$  trebuie să satisfacă respectiv relațiile:

$$\boxed{Z = \sum_{i=0}^m x^i \sum_{j=0}^n a_{ij}} \quad \text{și} \quad \boxed{Z = \sum_{i=0}^m b_{i0} u^i}$$

De aici decurg  $m + 1$  condiții pentru coeficienții suprafeței  $B$ :

$$b_{i0} = \sum_{j=0}^n a_{ij}; \quad i = 0, 1, \dots, m$$

Cerința de cuplare a peticului de suprafață  $B_{m,q}$  cu cel al suprafeței  $A_{m,n}$  este echivalentă cu  $m + 1$  condiții pentru suprafața  $B$ , deci în final avem  $(m + 1)(q + 1) - (m + 1) = q(m + 1)$  condiții independente. Dar cuplarea nu este netedă.

### 10.5.2. Cuplarea netedă

Fie  $n > 0$  și cele două petice de suprafață  $A$  și  $B$  alăturate și cuplate (neneted) în lungul frontierei comune (curbei marginale).

Condiția de cuplare netedă a celor două petice de suprafață  $A$  și  $B$  în lungul curbei marginale comune se exprimă prin identitatea tangentelor în lungul acestei curbe.

Derivatele parțiale ale funcției care definesc suprafața sînt:

$$\frac{\partial A}{\partial x} = \sum_{j=0}^n \sum_{i=1}^m i a_{ij} x^{i-1} y^j.$$

$$\frac{\partial A}{\partial y} = \sum_{j=1}^n \sum_{i=0}^m j a_{ij} x^i y^{j-1}$$

Avem:

$$\left[ \frac{\partial A}{\partial y} \right]_{y=1} = \sum_{j=1}^n j \sum_{i=0}^m a_{ij} x^i$$

deoarece curba marginală are tangenta perpendiculară și analog suprafața  $B$  are tangenta perpendiculară în lungul curbei marginale asociate; deci:

$$\left[ \frac{\partial B}{\partial v} \right]_{v=0} = \sum_{i=0}^m b_{i1} u^i$$

adică ambele funcții sînt identice în lungul frontierei comune, decurgînd din aceasta  $m + 1$  condiții pentru ecuația suprafeței  $B$ .

$$b_{i1} = \sum_{j=1}^n j a_{ij}, \quad i = 0, 1, \dots, m$$

Cerința de cuplare netedă este echivalentă cu  $2(m+1)$  condiții, deci în final suprafața  $B$  are:

$$(m+1)(q+1) - 2(m+1) = (m+1)(q-1)$$

condiții independente.

Mai departe este necesar ca să fie considerate și derivatele mixte ale celor două funcții care definesc suprafețele  $A$  și  $B$  deoarece suprafețele cuplate neted în lungul frontierei asociate trebuie să aibă și vîrfurile asociate.

Astfel să determinăm derivatele parțiale mixte pentru ambele suprafețe:

$$\frac{\partial^2 A}{\partial x \partial y} = \sum_{j=1}^n j \sum_{i=1}^m i a_{ij} x^{i-1} y^{j-1}$$

$$\frac{\partial^2 B}{\partial u \partial v} = \sum_{j=1}^q j \sum_{i=1}^m i b_{ij} u^{i-1} v^{j-1}$$

Vîrfurile în lungul frontierei asociate sînt:

$$\left[ \frac{\partial^2 A}{\partial x \partial y} \right]_{y=1} = \sum_{j=1}^n j \sum_{i=1}^m i a_{ij} x^{i-1}$$

$$\left[ \frac{\partial^2 B}{\partial u \partial v} \right]_{v=0} = \sum_{i=1}^m i b_{i1} u^{i-1}$$

Din aceste două relații rezultă deci că este necesar ca:

$$\boxed{\left[ \frac{\partial^2 A}{\partial x \partial y} \right]_{y=1} = \left[ \frac{\partial^2 B}{\partial u \partial v} \right]_{v=0}}$$

pentru vîrfurile asociate.

### 10.5.3. Aplicație

Să considerăm peticul de suprafață  $P_{2,1}$ , din figura 10.13 a cărei ecuație este  $z = (1-x^2)y$  și domeniul  $0 \leq x \leq 1$ ;  $y \leq 1$  pe care se proiectează curbele marginale

$z = y$  în planul  $x = 0$  (bisectoarea)

$z = (1-x^2)$  în planul  $y = 1$  (parabolă)

$z = 0$  în planul  $x = 1$  (segment)

$z = 0$  în planul  $y = 0$  (segment)

Să considerăm de asemenea, suprafața  $B_{2,2}$  asociată neted peticului de suprafață  $P_{2,1}$ , pe domeniul

$0 \leq x \leq 1$ ;  $1 \leq y \leq 2$  pe care trebuie să o determinăm fig. (10.13.a).

Astfel avem de ales frontiera, de exemplu,  $y = 2$  în planul  $z = 0$  și curba marginală  $x = 1$  în planul  $z = 0$ .

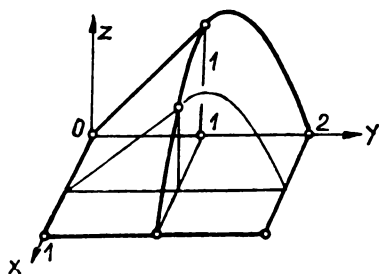


Fig. 10.13

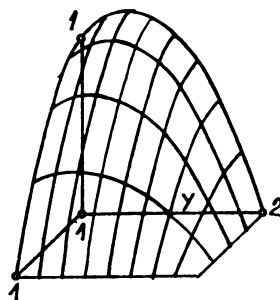


Fig. 10.13.a.

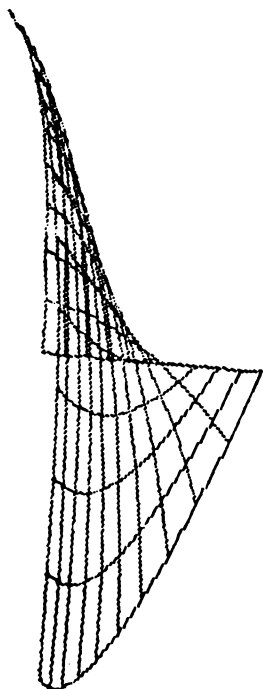


Fig. 10.13. b

Obținem sistemul de ecuații:

$$b_{00} + b_{01} + b_{02} = 1$$

$$b_{10} + b_{11} + b_{12} = 0$$

$$b_{20} + b_{21} + b_{22} = 0$$

Condiția ca frontiera comună să fie identică induce:

$$b_{00} = 1; b_{11} = 0; b_{20} = -1$$

Condiția de cuplare netedă induce valorile:

$$b_{01} = 1; b_{12} = 0; b_{21} = -1$$

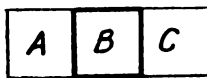
Obținem deci:

$$b_{02} = -2; b_{10} = 0; b_{22} = 2$$

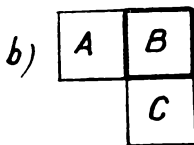
Ecuația suprafeței asociată (cuplată) neted  $B$  este așadar (fig. 10.13 a)

$$Z = (1 - x^2)(5y - 2y^2 - 2)$$

### 10.6. Cuplarea netedă între trei petice de suprafață alăturate



a)



b)

Să considerăm plăcile  $A, B, C$ . Există două posibilități de cuplare a suprafeței  $B$  cu suprafețele alăturate  $A$  și  $C$ : cuplarea în serie și cuplarea de colț (fig. 10.14).

### 10.6.1. Cuplarea în serie

Fie  $A_{m,n}$ ;  $B_{m,q}$ ;  $C_{m,r}$  cele trei petice de suprafață pe care intenționăm să le cuplăm neted, poziția suprafețelor în serie fiind în ordinea dată, adică cu suprafața  $B$  între suprafețele  $A$  și  $C$ .

Necesitatea de asociere induce pentru suprafața  $B$   $2(m+1)$  condiții, iar asocierea netedă induce  $4(m+1)$  condiții.

Pentru asociere numărul parametrilor independenți este  $(m+1)(q-1)$  pentru suprafața  $B$  și  $(m+1)(q-3)$  pentru asocierea netedă.

### 10.6.2. Aplicație

Să se asocieze neted parabolozii hiperbolici  $A_{1,1}$  și  $C_{1,1}$  din figura 10.15 prin intermediul suprafeței riglate  $B_{1,3}$ .

Asocierea netedă a acestei suprafețe  $B_{1,3}$  induce 8 condiții.

Efectuăm transformarea:

$$u = x$$

$$v = y - 2$$

Ecuatiile suprafețelor  $A$  și  $C$  sînt:

$$z = (1-x)(1-y)$$

$$z = (1-u)(1-v)$$

Condițiile de asociere netedă conduce la relațiile:

$$b_{00} = 0; b_{10} = 0; b_{01} = -1; b_{11} = 1$$

$$1 = b_{01} + b_{02} + b_{03}; -1 = -1 + 2b_{02} + 3b_{03}$$

$$-1 = b_{11} + b_{12} + b_{13}; 1 = 1 + 2b_{12} + 3b_{13}$$

Rezolvînd sistemul de ecuații obținem:

$$b_{02} = 6; b_{03} = -4;$$

$$b_{12} = -6; b_{13} = 4;$$

Deci ecuația suprafeței  $B_{1,3}$  asociată neted este:

$$z = y(x-1)(1-6y+4y^2)$$

Imaginea perspectivă a suprafeței  $B_{1,3}$  este redată în figura 10.15.b.

În figura 10.15.a sînt reprezentate curbele marginale și o secțiune paralelă cu planul  $(yz)$ .

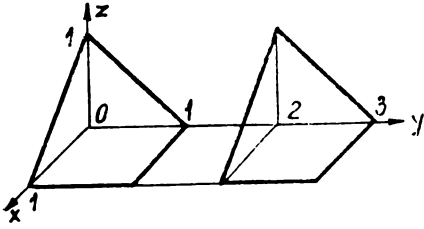


Fig. 10.15

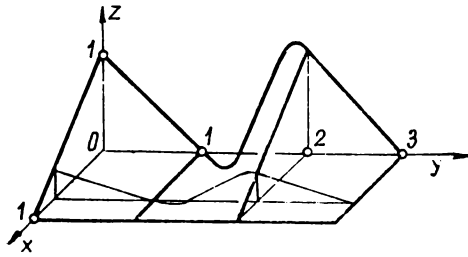


Fig. 10.15 a

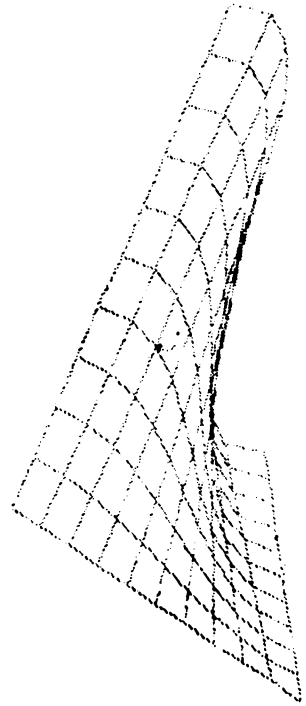


Fig. 10.15 b

### 10.6.3. Cuplarea de colț a peticelor de suprafață

Fie  $A_{m,n}$ ,  $C_{r,q}$  și  $B_{m,q}$  cele trei petice de suprafață pe care intenționăm să le cuplăm neted după schema din figura 10.14.b, adică suprafața  $B$  va ocupa poziția din colț. Suprafețele sînt definite pe domeniile:

$$0 \leq x; \quad y \leq 1 \quad \text{și} \quad 1 \leq x, \quad y \leq 2$$

pentru suprafața  $A$ , respectiv  $C$ , iar domeniul de colț al suprafeței  $B$  este:

$$0 \leq x \leq 1; \quad 1 \leq y \leq 2$$

În aceste condiții efectuăm transformarea:

$$s = x, \quad v = y - 1$$

$$t = y - 1, \quad u = x - 1$$

Cerința ca suprafețele  $A$  și  $B$  să aibă o curbă asociată în planul  $y = 1$  impune  $m + 1$  condiții:

$$\boxed{b_{i0} = \sum_{j=0}^n a_{ij}} \quad i = 0, 1, \dots, m$$



Mai departe trebuie ca suprafețele  $B$  și  $C$  să aibă o curbă asociată în planul  $x = 1$ , deci apar  $q + 1$  condiții:

$$\boxed{C_{0j} = \sum_{i=0}^m b_{ij}} \quad j = 0, 1, \dots, q$$

În final cerința ca suprafețele  $A$ ,  $B$  și  $C$  să aibă pentru  $x = 1$ ,  $y = 1$  un punct asociat este:

$$\boxed{\sum_{j=0}^q C_{0j} = \sum_{i=0}^m a_{i0}}$$

Din aceste trei relații rezultă că poziția de colț a suprafeței  $B_{m,q}$  în raport cu suprafețele  $A_{m,n}$  și  $C_{r,q}$  induce  $m + q + 1$  condiții pentru suprafața  $B$ . Numărul condițiilor independente este în final:

$$(m + 1)(q + 1) - (m + q + 1) = mq$$

#### 10.6.4. Cuplarea netedă de colț a peticelor de suprafață

Cuplarea netedă a suprafețelor  $A$  și  $B$  este exprimată prin relația:

$$\left[ \frac{\partial A}{\partial y} \right]_{y=1} = \left[ \frac{\partial B}{\partial t} \right]_{t=0} \quad \text{care induce } m + 1 \text{ condiții:}$$

$$b_{i1} = \sum_{j=1}^n j a_{ij}, \quad i = 0, 1, \dots, m$$

Cuplarea netedă a suprafețelor  $B$  și  $C$  este exprimată prin relația:

$$\left[ \frac{\partial B}{\partial s} \right]_{s=1} = \left[ \frac{\partial C}{\partial u} \right]_{u=0} \quad \text{care induce } q + 1 \text{ condiții}$$

$$c_{1j} = \sum_{i=1}^m i b_{ij}, \quad j = 0, 1, \dots, q$$

Condițiile care decurg din cerința de asociere netedă a suprafețelor  $A$  și  $C$  în punctele de colț asociate sînt:

$$\left[ \frac{\partial A}{\partial y} \right]_{x=y=1} = \left[ \frac{\partial C}{\partial v} \right]_{u=v=0}; \quad \left[ \frac{\partial A}{\partial x} \right]_{x=y=1} = \left[ \frac{\partial C}{\partial u} \right]_{u=v=0}$$

și induc două ecuații:

$$c_{01} = \sum_{j=1}^n \sum_{i=0}^m j a_{ij} \quad \text{și} \quad c_{10} = \sum_{j=0}^q \sum_{i=0}^m i a_{ij}$$

Din  $c_{1j}$  rezultă:

$$c_{11} = \sum_{i=1}^m i b_{i1}$$

și ținând seama de  $b_{i1}$  avem:

$$c_{11} = \sum_{i=1}^m i \sum_{j=1}^n j a_{ij}$$

Această ultimă relație conduce pentru punctele de colț asociate ale suprafețelor  $A$  și  $C$  la ecuațiile:

$$\left[ \frac{\partial^2 A}{\partial x \partial y} \right]_{x=y=1} = \sum_{j=1}^n j \sum_{i=1}^m i a_{ij}; \quad \left[ \frac{\partial^2 C}{\partial u \partial v} \right]_{u=v=0} = c_{11}!$$

Suprafețele care îndeplinesc relațiile  $c_{01}$ ,  $c_{10}$  și  $c_{11}$  pot fi numite suprafețe plăci asociate neted de colț.

Cerința de cuplare netedă a suprafeței  $B$  cu suprafețele  $A_{m,n}$  și  $C_{r,q}$ , induce:

$$(m + q + 1) + (m + 1 + q + 1) - 3 = 2(m + q)$$

pentru suprafața  $B$ .

Numărul parametrilor independenți pentru cuplarea netedă de colț a suprafeței  $B$  cu suprafețele  $A_{m,n}$  și  $C_{r,q}$ , în punctele de colț este:

$$(m + 1)(q + 1) - 2(m + q) = (m - 1)(q - 1)$$

și mai este necesar ca în suprafețele  $A$  și  $C$  valoarea indicilor  $m, q$  să fie cel puțin 1.

### 10.6.5. Aplicație

Să considerăm două suprafețe  $A$  și  $C$  de tip conoid

$$z = xy(1 - x)^2 \quad \text{și}$$

$$z = c(2 - x)(y - 1)^2$$

definite respectiv pe domeniile:

$$0 \leq x, y \leq 1$$

și

$$1 \leq x, y \leq 2$$

din figura 10.16.

Se cere cuplarea netedă a acestor suprafețe cu suprafața de colț  $B_{2,2}$ .

Numărul condițiilor induse sînt opt, fiecare parametru fiind independent.

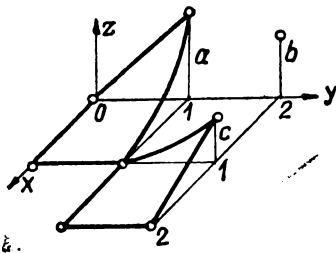


Fig. 101.6

Trebuie să alegem de exemplu, în punctul  $(0,2)$  cota arbitrară  $b$  și să efectuăm transformarea

$$\begin{aligned} u &= x - 1; & s &= x \\ v &= y - 1; & t &= y - 1 \end{aligned}$$

Suprafețele  $A$  și  $B$  au curba asociată în planul  $y = 1$ . Rezultă din aceasta pentru coeficienții suprafeței  $B$  identitatea

$$\begin{aligned} a(1-x)^2 &\equiv b_{00} + b_{10}s + b_{20}s^2 \text{ și deci} \\ b_{00} &= a; \quad b_{10} = -2a; \quad b_{20} = a \end{aligned}$$

Suprafețele  $A$  și  $B$  au tangentele perpendiculare identice în lungul curbei asociate. Deci rezultă identitatea:

$$\begin{aligned} a(1-x)^2 &\equiv b_{01} + b_{11}s + b_{21}s^2 \text{ și de aceea} \\ b_{01} &= a; \quad b_{11} = -2a; \quad b_{21} = a \end{aligned}$$

În mod absolut analog suprafețele  $B$  și  $C$  au tangentele perpendiculare identice în lungul curbei asociate, deci:

$$\begin{aligned} -cv^2 &\equiv b_{10} + 2b_{20} + t(b_{11} + 2b_{21}) + t^2(b_{12} + 2b_{22}) \text{ de unde} \\ (*) \quad & -c = b_{12} + 2b_{22} \end{aligned}$$

Suprafețele  $B$  și  $C$  au curba asociată în planul  $x = 1$ .  
Rezultă deci:

$$cv^2 \equiv t(b_{01} + b_{11} + b_{21}) + t^2(b_{02} + b_{12} + b_{22})$$

și deci ecuația:

$$(**) \quad b_{02} + b_{12} + b_{22} = 0$$

În final, pentru cota  $b$  a suprafeței  $B$  în punctul de colț  $(0,2)$ , deducem relația:

$$(***) \quad b = b_{00} + b_{01} + b_{02}$$

Rezolvând sistemul de ecuații (\*), (\*\*), (\*\*\*) obținem coeficienții:

$$b_{01} = a; \quad b_{02} = b - 2a; \quad b_{22} = b - 2a - 2c$$

Ecuația suprafeței  $B$  asociată (cuplată) neted suprafețelor  $A$  și  $C$  este:

$$Z = a(1-x)^2 + at(1-s)^2 + t^2[b-2a + (4a+3c-2b)s - (b-2a-2c)s^2]$$

sau după transformarea în sistemul inițial:

$$Z = a(1-x)^2 + a(1-x)^2(y-1) + (y-1)^2[b-2a + (4a+3c-2b)x + (b-2a-2c)x^2]$$

Dacă alegem concret:

$$a = 1.; \quad b = -0,5; \quad c = 1.$$

obținem suprafața reprezentată în imaginea perspectivă din figura 10.16. a pe care pot fi urmărite și curbele principale ale suprafeței  $B$ . În figura 10.16b sînt reprezentate cele trei suprafețe legate prin curbele marginale.

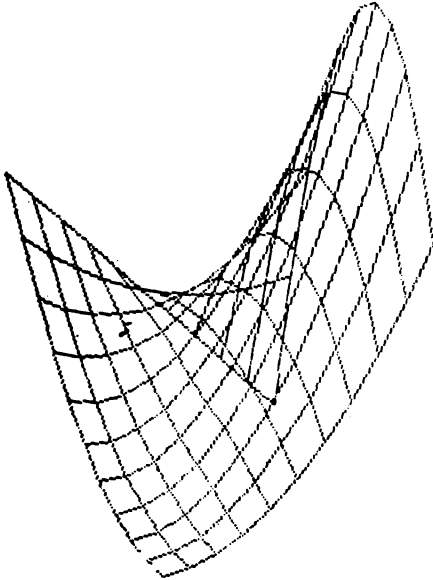


Fig. 10.16 a

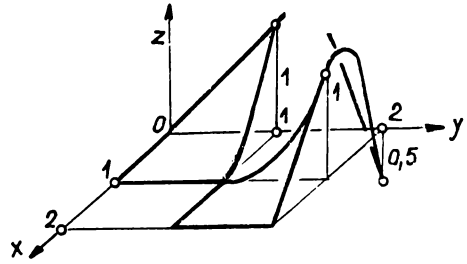


Fig. 10.16 b

### 10.7. Suprafețe bicubice

Suprafețele bicubice sînt utilizate frecvent în grafica pe calculator. Ecuația suprafeței bicubice se exprimă prin relația:

$$Z = \sum_{j=0}^3 \sum_{i=0}^3 a_{ij} x^i y^j, \quad 0 \leq x; \quad y \leq 1$$

Curbele principale ale acestor suprafețe sînt cubice.

Suprafața este definită prin 16 condiții, de exemplu, prin 16 puncte de coordonate

$$x = 0, \frac{1}{3}, \frac{2}{3}, 1$$

și

$$y = 0, \frac{1}{3}, \frac{2}{3}, 1$$

Curbele marginale induc 4 condiții, de exemplu punctele de început și tangentele în aceste puncte.

Suprafața este definită de curbele marginale și de patru puncte interioare. Numai marginea induce 12 condiții.

10.7.1. Cuplarea suprafeței placă  $P_{3,3}$ 

Numărul parametrilor independenți pentru cuplarea (asocierea) dintre două sau trei plăci este cuprins în următorul tabel:

Cuplare	Cuplare netedă	
12	8	Două plăci
8	0	Trei plăci în serie
9	4	A treia placă de colț

Este de reținut în special condiția de cuplare în serie pentru trei plăci.

## 10.7.2. Aplicație

Să considerăm suprafața de tip  $P_{3,3}$  definită prin punctele de colț în care are valoarea nulă sau  $T$  și prin punctul interior  $B$  de coordonate de exemplu,  $\frac{1}{3}, \frac{1}{3}, 1$ .

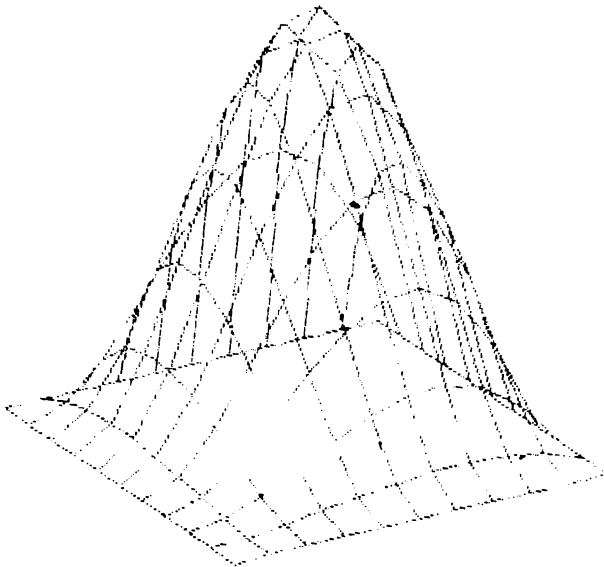


Fig. 10.17

Sistemul de 16 ecuații liniare pentru determinarea coeficienților  $a_{ij}$  are soluția:

$$\begin{aligned} a_{00} = a_{10} = a_{20} = a_{30} = \\ = a_{01} = a_{02} = a_{03} = 0 \end{aligned}$$

$$\begin{aligned} a_{21} = a_{12} = a_{32} = a_{23} = \\ = -2T, a_{22} = 4T \end{aligned}$$

$$a_{11} = a_{13} = a_{31} = a_{33} = T$$

Ecuția suprafeței este:

$$Z = Txy(1-x)^2(1-y)^3$$

Pentru valoarea  $T = 45,5625$  și  $B\left(\frac{1}{3}, \frac{1}{3}, 1\right)$ , obținem imaginea perspectivă a suprafeței și a curbelor principale din figura 10.17.

### 10.7.3. Aplicație

Considerăm suprafața bicubică  $Z = Txy(1-x)^2(1-y)^2$  și o suprafață de tip  $B_{3,3}$  pe care dorim să o cuplăm neted în serie cu prima suprafață pe domeniul  $0 \leq x \leq 1, 1 \leq y \leq 2$ . Asocierea netedă a celor două suprafețe induce opt condiții și anume:

$$b_{00} = b_{10} = b_{20} = b_{30} = b_{01} = b_{11} = b_{21} = b_{31} = 0$$

Alegem ca două curbe marginale ale suprafeței  $B_{3,3}$  segmentele  $x=1$  și  $y=2$  în planul  $(xy)$ . Cubica marginală în planul  $x=0$  este definită de direcția sa tangentei în punctul final.

Alegem mai departe cote nule în virfurile de colț  $(0,2,0)$  și  $(1,2,0)$ , precum și valorile  $T = 64$ ,  $S =$

$$= \frac{27}{4}. \text{ În aceste condiții punctul } B \text{ are coordo-$$

natele  $B \left( \frac{1}{3}; \frac{1}{3}; v = 1,404663 \right)$ , iar punctul  $N$

necunoscut al cudecii marginale are coordonatele  $\left( 0, \frac{5}{3}, -1 \right)$ .

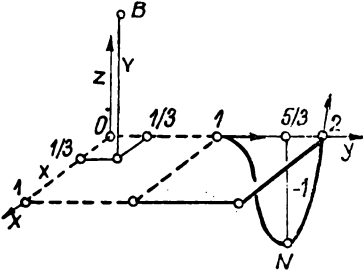


Fig. 10.18

Datele problemei sînt ilustrate în figura 10.18.

## 10.8. Suprafețe de interpolare definite prin curbe marginale (frontiere)

### 10.8.1. Cuadrice riglate

Considerăm punctele de colț  $P_{00}$ ,  $P_{10}$ ,  $P_{11}$  și  $P_{01}$ .

Suprafața de interpolare definită de aceste patru puncte deci de patrulaterul strîmb corespunzător, este o cuadrică riglată.

Ecuția vectorială este:

$$\bar{P}(u, v) = [1 - u, u] \cdot \bar{M}_{2,2} \cdot [1 - v, v]^T, \quad 0 \leq u, v \leq 1$$

în care:

$$\bar{M}_{2,2} = \begin{bmatrix} \bar{P}_{00} & \bar{P}_0 \\ \bar{P}_{10} & \bar{P}_{11} \end{bmatrix}$$

este o „amprentă“ a suprafeței, iar  $1 - t$  sînt funcții liniare.

Prin înmulțirea matricelor obținem ecuația:

$$\bar{P}(u, v) = [\bar{P}_{00}(1 - u) + \bar{P}_{10}u] (1 - v) + [\bar{P}_{01}(1 - u) + \bar{P}_{11}u] v$$

Alegem parametrul constant  $u = U$  și prin substituție rezultă:

$$\bar{P}(U, v) = [\bar{P}_{00}(1 - U) + \bar{P}_{10}U] (1 - v) + [\bar{P}_{01}(1 - U) + \bar{P}_{11}U] v$$

Curbele parametrice ale suprafeței  $\bar{P}(u, v)$  sînt drepte.

Dacă alegem special  $U = 0$  și substituim în ecuația  $\bar{P}(U, v)$ , obținem:

$$\bar{P}(0, v) = \bar{P}_{00}(1 - v) + P_{01}v$$

Aceasta este ecuația vectorială a dreptei care trece prin punctele  $P_{00}$  și  $P_{01}$ , adică este ecuația curbei marginale care în acest caz este o dreaptă.

Analog se poate demonstra că suprafața conține și celelalte curbe (laturi) marginale.

### 10.8.2. Aplicație

Să se determine cuadră riglată definită prin punctele de colț:

$$P_{00} = (0, 0, 0)$$

$$P_{10} = (1, 0, 0)$$

$$P_{11} = (0, 1, 0)$$

$$P_{01} = (0, 0, 1)$$

Substituind aceste valori în ecuația  $\bar{P}(u, v)$ , obținem ecuațiile parametrice ale cuadrice

$$x = u(1 - v)$$

$$y = uv$$

$$z = (1 - u)v$$

Eliminând parametrii  $u$  și  $v$  rezultă ecuația implicită a suprafeței:

$$xy + xz + yz + y^2 - y = 0$$

Imaginea perspectivă a acestei suprafețe și câteva drepte parametrice pot fi văzute pe figura 10.19.

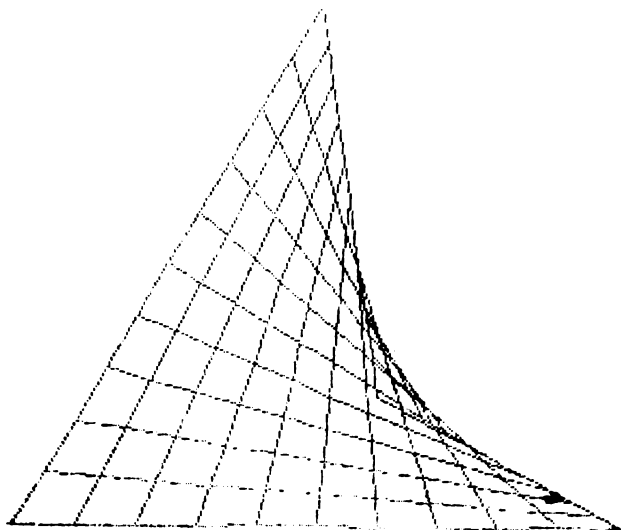


Fig. 10.19

## 10.8.3. Placa biliniară Coons definită prin curbe marginale

Suprafețele Coons sînt suprafețe de interpolare definite prin curbe marginale generale  $P_{u0}$ ;  $P_{u1}$ ,  $P_{0v}$  și  $P_{1v}$ , adică prin curbele care îndeplinesc condițiile de margine (fig. 10.20 a):

$$\begin{bmatrix} [\bar{P}_{u0}]_{u=0}, [\bar{P}_{u1}]_{u=0} \\ [\bar{P}_{u0}]_{u=1}, [\bar{P}_{u1}]_{u=1} \end{bmatrix} = \begin{bmatrix} [\bar{P}_{0v}]_{v=0}, [\bar{P}_{0v}]_{v=1} \\ [\bar{P}_{1v}]_{v=0}, [\bar{P}_{1v}]_{v=1} \end{bmatrix} = \begin{bmatrix} [\bar{P}_{00}, P_0] \\ [\bar{P}_{10}, \bar{P}_1] \end{bmatrix}$$

Un tip special de suprafață Coons este suprafața Coons biliniară:

$$\boxed{[1 - u, -1, u] \cdot \bar{C}_{3.3} \cdot [1 - v, -1, v]^T = 0}, \quad 0 \leq u, v \leq 1$$

„Amprenta” suprafeței este matricea  $\bar{C}_{3.3}$

$$\bar{C}_{3.3} = \begin{bmatrix} \bar{P}_{00} & \bar{P}_{0v} & \bar{P}_{01} \\ \bar{P}_{u0} & \bar{P}_{(u,v)} & \bar{P}_{u1} \\ \bar{P}_{10} & \bar{P}_{1v} & \bar{P}_{11} \end{bmatrix}$$

Elementele acestei matrice ocupă o poziție bine definită așa cum se poate observa pe schița din figura următoare (fig. 10.20 b).

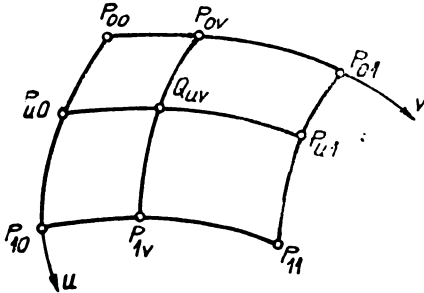


Fig. 10.20 a

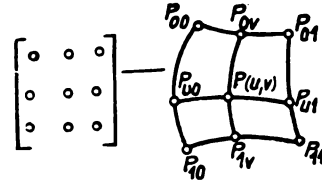


Fig. 10.20 b

Să alegem  $u = 0$  și să substituim în ecuația de mai sus a suprafeței. Obținem ecuația vectorială:

$$[1, -1, 0] \cdot \begin{bmatrix} \bar{P}_{00} & \bar{P}_{0v} & \bar{P}_{01} \\ \bar{P}_{00} & \bar{P}_{(0v)} & \bar{P}_{01} \\ \bar{P}_{10} & \bar{P}_{1v} & \bar{P}_{11} \end{bmatrix} \cdot \begin{bmatrix} 1 - v \\ -1 \\ v \end{bmatrix} = 0$$

de unde prin înmulțirea matricelor rezultă:

$$\bar{P}(0, v) = \bar{P}_{0v}$$

adică marginea  $P(0, v)$  a plăcii de suprafață este tocmai curba dată  $P_{0v}$ .

Similar se demonstrează că curbele  $P_{1v}$ ,  $P_{u0}$  și  $P_{u1}$  sînt celelalte margini ale plăcii de suprafață.



Dacă alegem două curbe marginale opuse ale suprafeței

$$\bar{P}_{u0} = \bar{P}_{00}(1 - u) + \bar{P}_{10}u$$

$$\bar{P}_{u1} = \bar{P}_{01}(1 - u) + \bar{P}_{11}u$$

și le substituim în ecuația  $\bar{P}(u, v)$ , obținem:

$$\bar{P}(u, v) = \bar{P}_{0v}(1 - u) + \bar{P}_{1v}u$$

adică ecuația suprafeței riglate.

Dreptele alcătuiesc un sistem de curbe parametrice.

Dacă alegem mai departe constanta  $v = V$ , atunci ecuația cărei aparțin curbele parametrice este:

$$\bar{P}(u, V) = \bar{P}_{0v}(1 - u) + \bar{P}_{1v}u$$

unde  $\bar{P}_{0v}$  și  $\bar{P}_{1v}$  sînt vectorii constanți (punctele), iar  $\bar{P}(u, v)$  este dreapta care le leagă.

*Așadar suprafața biliniară Coons care are drept curbe marginale opuse două segmente de dreaptă, este o suprafață riglată.*

Această suprafață conține un al doilea sistem de curbe parametrice (drepte) care se sprijină pe curbele (segmentele) marginale opuse:

$$\bar{P}_{0v} = \bar{P}_{00}(1 - v) + \bar{P}_{01}v$$

$$\bar{P}_{1v} = \bar{P}_{10}(1 - v) + \bar{P}_{11}v$$

În concluzie suprafața biliniară Coons este dublu riglată.

#### 10.8.4. Aplicație

Să se determine suprafața biliniară Coons definită prin următoarele curbe marginale:

$$\bar{P}_{u0} = [u, u^2, 1 - u]$$

$$\bar{P}_{u1} = [u + 1, u^2 - 1, 2 - u]$$

$$\bar{P}_{1v} = [1 + v, 1 - v^2, v]$$

$$\bar{P}_{0v} = [v, -v^2, v + 1]$$

pe domeniul  $0 \leq u, v \leq 1$  unde:

$$x = u + v$$

$$y = u^2 - v^2$$

În acest caz ecuația suprafeței biliniară Coons este:

$$[1 - u, -1, u] \cdot \begin{bmatrix} 1, & 1 + v, & 2 \\ 1 - u, & v, & 2 - u \\ 0, & v, & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 - v \\ -1 \\ v \end{bmatrix} = 0$$

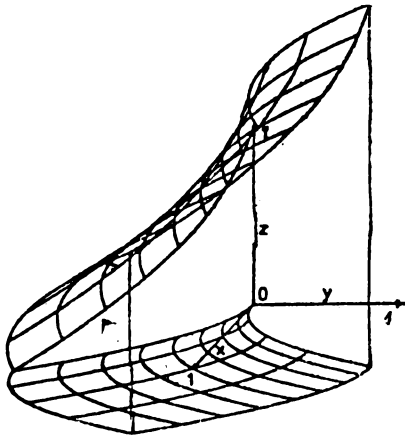


Fig. 10.21

sau după înmulțirea matricelor:

$$z = 1 - u + v$$

Ecuția implicită a acestei cuadrice riglate se obține prin eliminarea parametrilor  $u$  și  $v$ . Avem în final:

$$xz - x - y = 0$$

Imaginea perspectivă a suprafeței și a proiecției sale orizontale este redată pe figura 10.21.

Sînt date, de asemenea, cîteva curbe parametrice.

### 10.8.5. Aplicație

Să se determine suprafața biliniară Coons definită în figura 10.22 prin cele 4 curbe limită astfel:

În planul  $x = 0$  curba limită este parabola  $z = 4y(1 - y)$ ,

În planul  $x = 1$  curba limită este sinusoida  $z = \sin(2\pi y)$ ,

În planul  $y = 0$  curba limită este dreapta  $z = 0$  și

În planul  $y = 1$  curba limită este semicercul  $z = -\sqrt{x(1-x)}$ ,

Ecuția acestei suprafețe placă biliniară Coons este:

$$0 = [(1-x), -1, x] \cdot \begin{bmatrix} 0 & 4y(1-y) & 0 \\ 0 & z & -\sqrt{x(1-x)} \\ 0 & (\sin 2\pi y) & 0 \end{bmatrix} \cdot \begin{bmatrix} 1-y \\ -1 \\ y \end{bmatrix}$$

Să insistăm asupra produsului acestor matrici. Avem:

$$[(1-x) \cdot 0 - 1 \cdot 0 + x \cdot 0, (1-x)4y(1-y) - z + x \cdot \sin(2\pi y),$$

$$(1-x) \cdot 0 + (-1) \cdot (-\sqrt{x(1-x)} + x \cdot 0)] \cdot \begin{bmatrix} 1-y \\ -1 \\ y \end{bmatrix} = 0$$

Sau mai departe:

$$[0 \cdot (1-y), -(1-x)4y(1-y) - z + x \sin(2\pi y), y\sqrt{x(1-x)}] = 0$$

Așadar:

$$z = 4y(1-x)(1-y) + x \sin(2\pi y) - y\sqrt{x(1-x)}$$

Forma de ansamblu a suprafeței este reprezentată în figura 10.23.

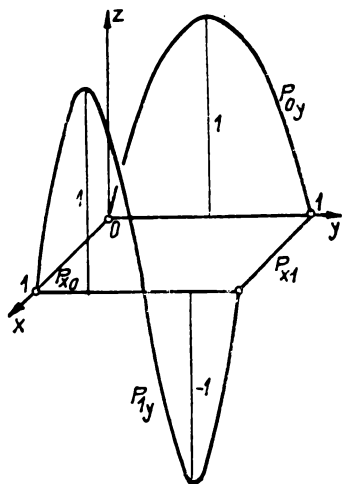


Fig. 10.22

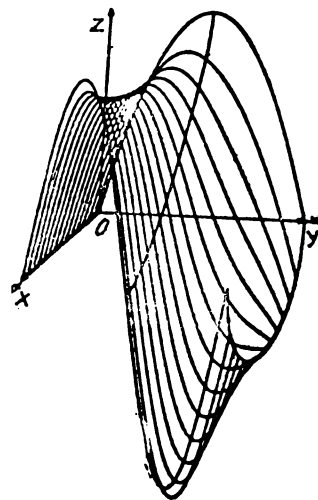


Fig. 10.23

### 10.8.6. Aplicație

Să se determine suprafața riglată definită de următoarele curbe marginale spațiale opuse:

$$\bar{P}_{0v} = \left[ v, \sin\left(v \frac{\pi}{2}\right), \cos\left(v \frac{\pi}{2}\right) \right]$$

$$\bar{P}_{1v} = \left[ v, v^2, -\frac{v}{2} \right]$$

Ecuatiile parametrice ale acestei suprafețe riglate  $P(u, v)$  sînt (vezi paragraful 10.1.5)

$$\begin{aligned} x &= (1-u)v + uv = v \\ y &= (1-u) \sin\left(v \frac{\pi}{2}\right) + uv^2 \\ z &= (1-u) \cos\left(v \frac{\pi}{2}\right) - \frac{uv}{2} \end{aligned}$$

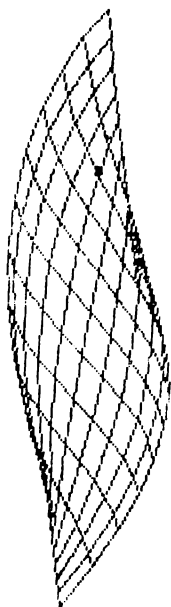


Fig. 10.24

Imaginea perspectivă a acestei suprafețe rețea este redată în figura 10.24.

## 10.8.7. Placa bicubică Coons definită prin curbe marginale

Această suprafață este, de asemenea mult utilizată în grafica pe calculator ca și suprafața biliniară Coons.

Ecuția plăcii bicubică Coons definită prin curbele marginale este:

$$[F_1(u), -1, F_2(u)] \cdot \bar{C}_{3,3} \cdot [F_1(v), -1, F_2(v)]^T = 0$$

unde:

$$0 \leq u, v \leq 1$$

$F_1$  și  $F_2$  sînt funcțiile cubice (polinoamele cubice)

$$F_1(t) = 2t^3 - 3t^2 + 1$$

$$F_2(t) = -2t^3 + 3t^2$$

Curbele marginale date sînt  $\bar{P}_{u0}$ ,  $\bar{P}_{u1}$ ,  $\bar{P}_{0v}$ ,  $\bar{P}_{1v}$ .

Înlocuind  $u = 0$  în ecuația suprafeței avem:

$$[1, -1, 0] \cdot \begin{bmatrix} \bar{P}_{00} & \bar{P}_{0v} & \bar{P}_{01} \\ \bar{P}_{00} & \bar{P}(0'v) & \bar{P}_{01} \\ \bar{P}_{00} & \bar{P}_{1v} & \bar{P}_{01} \end{bmatrix} \cdot \begin{bmatrix} F_1(v) \\ -1 \\ F_2(v) \end{bmatrix} = 0$$

de unde rezultă  $\bar{P}(0, v) = \bar{P}_{0v}$  adică marginea  $P(0, v)$  a suprafeței este curba  $P_{0v}$  marginală. Similar putem demonstra că celelalte curbe marginale ale suprafeței sînt  $P_{1v}$ ,  $P_{u0}$ ,  $P_{u1}$ .

Dacă înlocuim funcțiile  $F_1$  și  $F_2$  prin funcțiile liniare  $1 - t$  și  $t$  obținem ecuația plăcii biliare Coons.

Derivînd ecuația plăcii obținem vectorii tangenți curbelor parametrice. Să notăm derivatele funcțiilor  $F_1$ ,  $F_2$ ,  $P_{u0}$  și  $P_{u1}$  prin  $F_1'$ ,  $F_2'$ ,  $\bar{P}_{u0}^u$  și  $\bar{P}_{u1}^u$ .

Să calculăm derivatele:

$$\frac{\partial \bar{P}(u, v)}{\partial u} = \bar{P}^u(u, v); \quad \frac{\partial \bar{P}(u, v)}{\partial v} = \bar{P}^v(u, v); \quad \frac{\partial^2 \bar{P}(u, v)}{\partial u \partial v} = P^{uv}(u, v)$$

Avem deci:

$$\begin{aligned} \frac{\partial \bar{P}(u, v)}{\partial u} = \bar{P}^u(u, v) = & -F_1(v) [\bar{P}_{00}F_1'(u) - \bar{P}_{u0}^u + \bar{P}_{10}F_2'(u)] + \\ & + \bar{P}_{0v}F_1'(u) + \bar{P}_{1v}F_2'(u) - F_2(v) [\bar{P}_{01}F_1'(u) - \bar{P}_{u1}^u + \bar{P}_{11}F_2'(u)] \end{aligned}$$

Prin analogie putem nota direct:

$$\begin{aligned} \bar{P}^v(u, v) = & -F_1'(v) [\bar{P}_{00}F_1(u) - \bar{P}_{u0} + \bar{P}_{u0}F_2(u)] + \bar{P}_{0v}^vF_1(u) + \\ & + \bar{P}_{1v}^vF_2(u) - F_2'(v) [\bar{P}_{01}F_1(u) - \bar{P}_{u1} + \bar{P}_{11}F_2(u)] \end{aligned}$$

Vectorii tangenți curbelor parametrice marginale sînt determinați prin ecuațiile:

$$\begin{aligned}\bar{P}^u(0, v) &= \bar{P}_{00}^u F_1(v) + \bar{P}_{01}^u F_2(v) \\ \bar{P}^u(1, v) &= \bar{P}_{10}^u F_1(v) + \bar{P}_{11}^u F_2(v) \\ \bar{P}^v(u, 0) &= \bar{P}_{00}^v F_1(u) + \bar{P}_{10}^v F_2(u) \\ \bar{P}^v(u, 1) &= \bar{P}_{01}^v F_1(u) + \bar{P}_{11}^v F_2(u)\end{aligned}$$

Derivata mixtă este:

$$\begin{aligned}\frac{\partial^2 \bar{P}(u, v)}{\partial u \partial v} = P^{uv}(u, v) &= -F_1'(v) [\bar{P}_{00} F_1'(u) - \bar{P}_{u0}^u + \bar{P}_{10} F_2'(u)] + \\ &+ \bar{P}_{00}^v F_1'(u) + \bar{P}_{10}^v F_2'(u) - F_2'(v) [\bar{P}_{01} F_1'(u) - \bar{P}_{u1}^u + \bar{P}_{11} F_2'(u)]\end{aligned}$$

În punctele de colț ale suprafeței avem:

$$F_1'(0) = F_1'(1) = F_2'(0) = F_2'(1) = 0$$

și de asemenea:

$$\bar{P}^{uv}(0, 0) = \bar{P}^{uv}(1, 0) = \bar{P}^{uv}(0, 1) = \bar{P}^{uv}(1, 1) = 0$$

### 10.8.8. Aplicație

Să se determine suprafața plăcă bicubică Coons definită prin curbele marginale

$$\begin{aligned}\bar{P}_{0v} &= \left[ v, \sin\left(v \frac{\pi}{2}\right), \cos\left(v \frac{\pi}{2}\right) \right] \\ \bar{P}_{1v} &= \left[ v, v^2, -\frac{v}{2} \right]\end{aligned}$$

Ecuațiile parametrice ale suprafeței sînt:

$$\begin{aligned}[F_1(u), -1, F_2(u)] \cdot \begin{bmatrix} 0, & v, & 1 \\ 0, & x, & 1 \\ 0, & v, & 1 \end{bmatrix} \cdot \begin{bmatrix} F_1(v) \\ -1 \\ F_2(v) \end{bmatrix} &= 0 \\ [F_1(u), -1, F_2(u)] \cdot \begin{bmatrix} 0, & \sin\left(v \frac{\pi}{2}\right), & 1 \\ 0, & y, & 1 \\ 0, & v^2, & 1 \end{bmatrix} \cdot \begin{bmatrix} F_1(v) \\ -1 \\ F_2(v) \end{bmatrix} &= 0 \\ [F_1(u), -1, F_2(u)] \cdot \begin{bmatrix} 1, & \cos\left(v \frac{\pi}{2}\right), & 0 \\ 1 - u, & z, & -\frac{u}{2} \\ 0, & -\frac{v}{2}, & -\frac{1}{2} \end{bmatrix} \cdot \begin{bmatrix} F_1(v) \\ -1 \\ F_2(v) \end{bmatrix} &= 0\end{aligned}$$

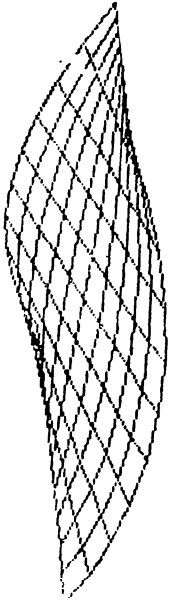


Fig. 10.25

După înmulțirea matricelor și ordonare obținem ecuațiile parametriche ale suprafeței sub forma:

$$\begin{aligned}
 x &= v \\
 y &= (2u^3 - 3u^2 + 1) \sin\left(v \frac{\pi}{2}\right) + u^2(3 - 2u)v^2 \\
 z &= uv^2(3 - 2v) \left( u^2 - \frac{3u}{2} + \frac{1}{2} \right) - u^2v \frac{3 - 2u}{2} - \\
 &\quad - u(2u^2 - 3u + 1) + (2u^3 - 3u^2 + 1) \cos\left(v \frac{\pi}{2}\right)
 \end{aligned}$$

Imaginea perspectivă a suprafeței este dată în figura 10.25.

### 10.9. Suprafața de interpolare definită prin 12 vectori în matricea restricțiilor (suprafața Ferguson)

Această suprafață este un tip special de suprafață bicubică Coons ale cărei curbe marginale sînt cubice Ferguson. Să demonstrăm acest fapt. Astfel curbele marginale sînt definite de ecuațiile:

$$\begin{aligned}
 \bar{P}_{0v} &= \bar{P}_{00}F_1(v) + \bar{P}_{01}F_2(v) + \bar{P}_{00}^vF_3(v) + \bar{P}_{01}^vF_4(v) \\
 \bar{P}_{1v} &= \bar{P}_{10}F_1(v) + \bar{P}_{11}F_2(v) + \bar{P}_{10}^vF_3(v) + \bar{P}_{11}^vF_4(v) \\
 \bar{P}_{u0} &= \bar{P}_{00}F_1(u) + \bar{P}_{10}F_2(u) + \bar{P}_{00}^uF_3(u) + \bar{P}_{10}^uF_4(u) \\
 \bar{P}_{u1} &= \bar{P}_{01}F_1(u) + \bar{P}_{11}F_2(u) + \bar{P}_{01}^uF_3(u) + \bar{P}_{11}^uF_4(u)
 \end{aligned}$$

unde cei 8 vectori tangenți sînt notați prin:

$$\bar{P}_{00}^u, \bar{P}_{01}^u, \bar{P}_{10}^u, \bar{P}_{11}^u, \bar{P}_{00}^v, \bar{P}_{01}^v, \bar{P}_{10}^v, \bar{P}_{11}^v$$

în punctele de colț ale suprafeței placă Ferguson definită prin 12 vectori și a cărei ecuație este:

$$\bar{F}(u,v) = [F_1(u), F_2(u), F_3(u), F_4(u)] \cdot \bar{F} \cdot [F_1(v), F_2(v), F_3(v), F_4(v)]^T$$

Matricea restricțiilor sau „amprenta” suprafeței este:

$$\bar{F} = \begin{bmatrix} \bar{P}_{00} & \bar{P}_{01} & P_{00}^v & P_{01}^v \\ \bar{P}_{10} & \bar{P}_{11} & P_{10}^v & P_{11}^v \\ \hline \bar{P}_{00}^u & \bar{P}_{01}^u & 0 & 0 \\ \bar{P}_{10}^u & \bar{P}_{11}^u & 0 & 0 \end{bmatrix} = \begin{bmatrix} \bar{1} & \bar{2} \\ \bar{3} & \bar{4} \end{bmatrix}$$

Această matrice poate fi împărțită în patru secțiuni.

Prima secțiune conține vectorii punctelor de colț.

A doua secțiune conține vectorii tangenți curbilor marginale  $P_{0v}$  și  $P_{u1}$ .

A treia secțiune conține vectorii tangenți curbilor marginale  $P_{u0}$  și  $P_{1v}$ .

A patra secțiune conține vectorii nuli (în acest caz).

Să alegem, de exemplu,  $u = 0$ . Din ecuația  $P(u, v)$  a suprafeței avem ecuația curbei marginale:

$$\begin{aligned}\bar{P}(0, v) &= [1, 0, 0, 0] \bar{F} [F_1(v), F_2(v), F_3(v), F_4(v)]^T = \\ &= \bar{P}_{00}F_1(v) + \bar{P}_{01}F_2(v) + \bar{P}_{00}^vF_3(v) + \bar{P}_{01}^vF_4(v) = \bar{P}_{0v}\end{aligned}$$

La fel putem demonstra că  $P_{1v}$ ,  $P_{u0}$  și  $P_{u1}$  sînt curbe marginale ale suprafeței.

### 10.9.1. Aplicație

Dacă  $u = x$ ,  $v = y$  și dacă coordonatele din cele trei secțiuni ale matricei  $F$  sînt:

$$\bar{1} = \begin{bmatrix} 1, & 0 \\ 1, & 0 \end{bmatrix}; \quad \bar{2} = \begin{bmatrix} 0, & 0 \\ 0, & 0 \end{bmatrix}; \quad \bar{3} = \begin{bmatrix} -1, & 0 \\ 0 & 0 \end{bmatrix}$$

În aceste condiții ecuația explicită a suprafeței  $\bar{P}(u, v)$  este:

$$Z = [F_1(x), F_2(x), F_3(x), F_4(x)] \times$$

$$\times \begin{bmatrix} 1 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} F_1(y) \\ F_2(y) \\ F_3(y) \\ F_4(y) \end{bmatrix}$$

iar după înmulțirea matricelor obținem ecuația suprafeței:

$$\begin{aligned}Z &= [F_1(x) + F_2(x)] F_1(y) - \\ &- F_1(x) F_3(y) = y^3 - y^2 - y + \\ &+ 1 - x^2y(2x - 3)(y - 1)^2\end{aligned}$$

Ambele sisteme de curbe parametrice ale suprafeței sînt cubice.

Imaginea perspectivă a suprafeței împreună cu rețeaua de curbe parametrice este redată pe epura din figura 10.26.

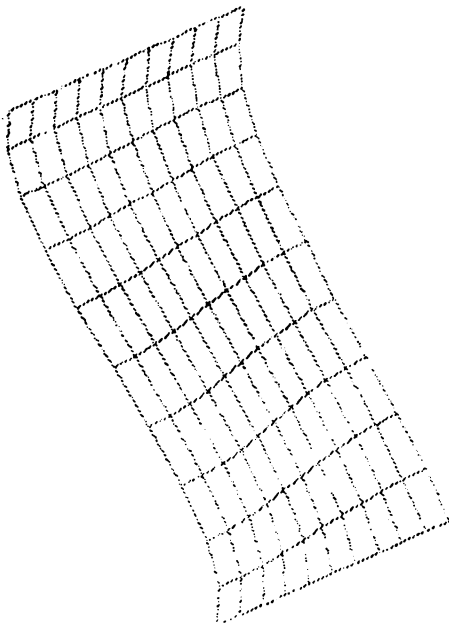


Fig. 10.26

### 10.9.2. Aplicație

Suprafața Ferguson definită în planul  $xy$  de derivatele nule este evident planul  $xy$  (în treaga secțiune a matricei  $\bar{M}$  are elementele nule).

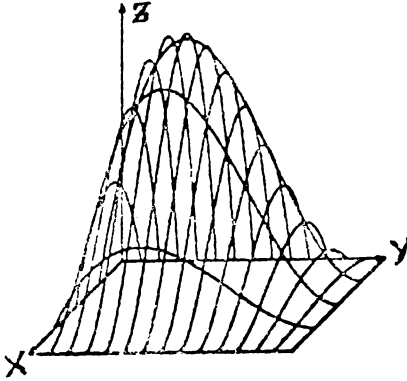


Fig. 10.27

Suprafața Coons definită de secțiunea caracteristică

$$\begin{bmatrix} T & 0 \\ 0 & 0 \end{bmatrix}$$

are ecuația:

$$Z = T \cdot F_3(x) \cdot F_3(y) = T(x^3 - 2x^2 + x)(y^3 - 2y^2 + y)$$

Imaginea suprafeței este redată în figura 10.27 pentru  $T = 27 \cdot 27/16$ .

Suprafața are un maxim egal cu 1 în punctul  $(1/3, 1/3)$ . Pentru diferite valori  $T$  se obține prin afinitate familia tuturor suprafețelor.

### 10.10. Suprafața de interpolare Coons definită prin 16 vectori în matricea restricțiilor

Generalizarea suprafeței de interpolare definită prin 12 vectori în matricea restricțiilor se poate face înlocuind în secțiunea a patra vectorii nuli prin secțiunea:

$$\bar{4} = \begin{bmatrix} \bar{P}_{00}^{uv} & \bar{P}_{01}^{uv} \\ \bar{P}_{10}^{uv} & \bar{P}_{11}^{uv} \end{bmatrix}$$

Astfel matricea  $\bar{F}$  devine matricea  $\bar{C}_{4,4}$  iar ecuația suprafeței de interpolare definită prin 16 vectori în matricea restricțiilor este:

$$\bar{P}(u,v) = [F_1(u), F_2(u), F_3(u), F_4(u)] \cdot \bar{C}_{4,4} \cdot [F_1(v), F_2(v), F_3(v), F_4(v)]^T$$

Deoarece elementele în matricele  $\bar{F}$  și  $\bar{C}_{4,4}$  sînt vectori constanți suprafețele de interpolare sînt de gradul 6 în raport cu funcțiile cubice  $F_1, F_2, F_3$  și  $F_4$ . Deci curbele parametrice sînt cubice.

Se poate demonstra că derivatele mixte sînt:

$$\frac{\partial^2 \bar{P}(u,v)}{\partial u \partial v} = [F_1'(u), F_2'(u), F_3'(u), F_4'(u)] \cdot \bar{C}_{4,4} \cdot F_1'(v),$$

$$F_2'(v), F_3'(v), F_4'(v)]^T$$

și pentru  $u = v = 0$  avem  $F_1'(0) = F_2'(0) = F_4'(0) = 0$ , iar  $F_3'(0) = 1$ .



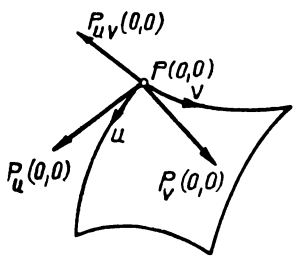


Fig. 10.28

Așadar:

$$\left[ \frac{\partial^2 \bar{P}(u,v)}{\partial u \partial v} \right]_{u=v=0} = [0, 0, 1, 0] \cdot \bar{C}_{4,4} \times [0, 0, 1, 0]^T = P^{uv}_{00}$$

deci în punctul  $P(0,0)$  suprafața are vectorul  $\bar{P}_{10}^{uv}$  și similar în celelalte puncte de colț, avem vectorii:  $\bar{P}_{10}^{uv}$ ,  $\bar{P}_{01}^{uv}$ ,  $\bar{P}_{11}^{uv}$  în  $P_{10}$ ,  $P_{01}$ ,  $P_{11}$  respectiv (fig. 10.28 și fig. 10.29 a, b, c, d).

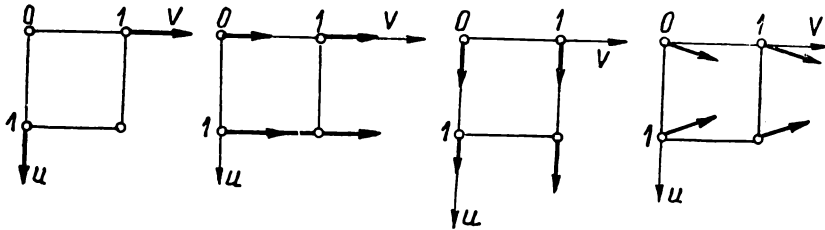


Fig. 10.29

### 10.10.1. Aplicație

Să se determine suprafața de interpolare definită prin 16 vectori în matricea  $C_{4,4}$  știind că secțiunea 4 este:

$$\begin{pmatrix} 1, 0 \\ 1, 0 \end{pmatrix}$$

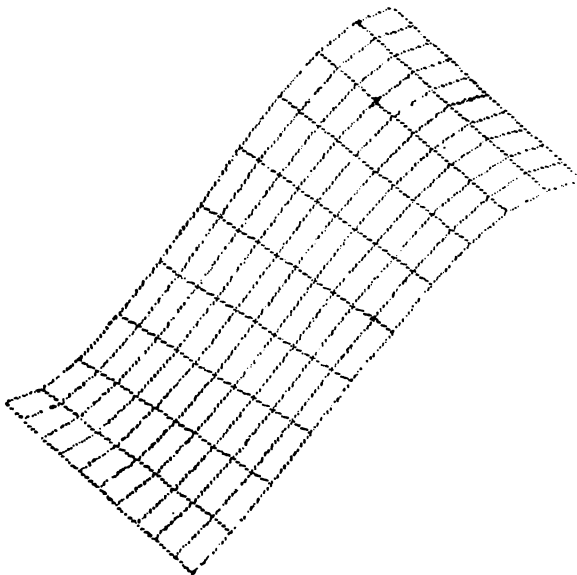


Fig. 10.30

Ecuția suprafeței este:

$$Z = [F_1(x), F_2(x), F_3(x), F_3(x)] \times$$

$$\times \begin{bmatrix} 1, 0, -1, 0 \\ 1, 0, 0, 0 \\ 0, 0, 1, 0 \\ 0, 0, 1, 0 \end{bmatrix} \cdot \begin{bmatrix} F_1(y) \\ F_2(y) \\ F_3(y) \\ F_4(y) \end{bmatrix}$$

$$[F_3(t) = t^3 - 2t^2 + t; F_3(t) = t^3 - t^2]$$

După înmulțirea matricelor și înlocuirea funcțiilor  $F_1, F_2, F_3, F_4$ , se obține ecuația explicită a suprafeței:

$$Z = xy(y - 1)^2 + y^3 - y^2 - y + 1$$

Curbele  $x = \text{constant}$  sînt cubice.  
Curbele  $y = \text{constant}$  sînt drepte.  
Suprafața de interpolare este riglată.

Imaginea perspectivă a suprafeței este redată în figura 10.30.

### 10.11. Suprafețe de interpolare Coons definite prin curbe marginale și prin tangentele în lungul acestor curbe

Ecuția suprafeței Coons generală este:

$$\boxed{\bar{U}\bar{C}_{5,5}\bar{V} = \bar{0}}$$

unde:

$$\bar{U} = [F_1(u), -1, F_2(u), F_3(u), F_3(u)]$$

$$\bar{V} = [F_1(v), -1, F_2(v), F_3(v), F_4(v)]^T$$

iar matricea restricțiilor este:

$$\bar{C}_{5,5} = \begin{array}{ccc|cc} \bar{P}_{00} & \bar{P}_{0v} & \bar{P}_{01} & \bar{P}_{00}^v & \bar{P}_{01}^v \\ \bar{P}_{u0} & \bar{P}(u,v) & \bar{P}_{u1} & \bar{P}_{u0}^v & \bar{P}_{u1}^v \\ \bar{P}_{10} & \bar{P}_{1v} & \bar{P}_{11} & \bar{P}_{10}^v & \bar{P}_{11}^v \\ \hline \bar{P}_{00}^u & \bar{P}_{0v}^u & \bar{P}_{01}^u & \bar{P}_{00}^{uv} & \bar{P}_{01}^{uv} \\ \bar{P}_{10}^u & \bar{P}_{1v}^u & \bar{P}_{11}^u & \bar{P}_{10}^{uv} & \bar{P}_{11}^{uv} \end{array} = \begin{array}{c|c} \bar{1} & \bar{2} \\ \hline \bar{3} & \bar{4} \end{array}$$

Notăția vectorilor tangenți poate fi urmărită pe figura 10.31.

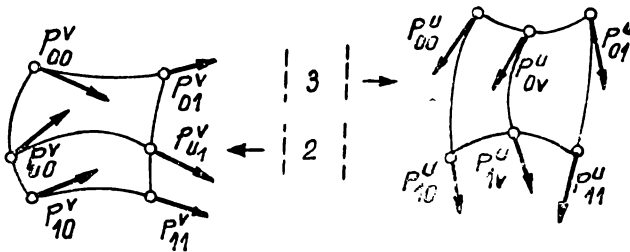


Fig. 10.31

Suprafața  $P(u, v)$  conține curbele marginale. Astfel, de exemplu, să alegem  $u = 0$ . Deoarece:

$$F_1(0) = 1; F_2(0) = F_3(0) = F_4(0) = 0$$

obținem prin înlocuire în ecuația suprafeței Coons generală:

$$[1, -1, 0, 0] \cdot \bar{C}_{5,5} \cdot \bar{V} = 0$$

și după înmulțirea matricelor:

$$(\bar{P}_{00} - \bar{P}_{00})F_1(v) - \bar{P}_{0v} - \bar{P}(0, v) = \bar{0}$$

de unde:

$$\bar{P}(0, v) = \bar{P}_{0v}$$

deci obținem curba marginală  $P_{0v}$ . La fel se poate demonstra că și celelalte curbe date sînt curbe marginale ale suprafeței.

*Suprafața  $P(u, v)$  are în lungul curbelor marginale tangente date. Astfel derivînd ecuația suprafeței Coons generală avem:*

$$\bar{U}' \bar{C}'_{5,5} \bar{V} = \bar{0}$$

În această ecuație notăm:

$$\bar{U}' = [F'_1(u), -1, F'_2(u), F'_3(u), F'_4(u)] \text{ și } \bar{C}'_{5,5}$$

în care avem elementele  $\bar{P}_{u0}$ ,  $\bar{P}^u(u, v)$ ,  $\bar{P}_{u1}$ ,  $\bar{P}_{u0}^{uv}$ ,  $\bar{P}_{u1}^{uv}$ .

Pentru  $u = 0$  obținem din ecuația derivată:

$$[0, -1, 0, 1, 0] \cdot [\bar{C}'_{5,5}]_{u=0} \cdot \bar{V} = \bar{0}$$

iar după înmulțirea matricelor:

$$- [-\bar{P}^u(0, v) + \bar{P}_{0v}^u] = \bar{0}$$

de unde:

$$P^u(0, v) = P_{0v}^u$$

deci suprafața are în lungul curbei  $P(0, v)$  tangenta perpendiculară  $P_{0v}^u$ .

*Suprafața are în punctele de colț, elemente date prin secțiunea 4. Să derivăm mai departe ecuația suprafeței Coons generală în raport cu  $V$ . Avem:*

$$\bar{U}' \bar{C}''_{5,5} \bar{V}' = \bar{0}$$

în care:

$$\bar{V}' = [F'_1(v), -1, F'_2(v), F'_3(v), F'_4(v)] \text{ și } \bar{C}''_{5,5}$$

în care avem elementele:  $\bar{P}_{0v}$ ,  $\bar{P}^{uv}(u, v)$ ,  $\bar{P}_{1v}$ ,  $\bar{F}_{0v}^{ua}$ ,  $\bar{P}_{1v}^{uv}$ .

Dacă alegem  $u = v = 0$  și substituim aceste valori în ultima ecuație derivată obținem:

$$[0, -1, 0, 1, 0] \cdot [\bar{C}''_{5,5}]_{u=v=0} \cdot [0, -1, 0, 1, 0] = \bar{0}$$

deci:

$$- [-\bar{P}^{uv}(0, 0) + \bar{P}_{00}^{uv}] = \bar{0}$$

de unde  $\bar{P}(u, v) = \bar{P}_{00}^{uv}$ , deci suprafața de interpolare are în punctul de colț  $P(0, 0)$  vectorul  $\bar{P}_{00}^{uv}$ . Analog se poate demonstra pentru celelalte puncte de colț ale suprafeței.

### 10.11.1. Aplicație

Se alege  $u = x$ ,  $v = y$  și se consideră suprafața definită prin curbele marginale care sînt:

– parabola  $z = y(1 - y)$  în planul  $x = 0$

– dreptele  $z = 0$  în planul  $x = 1$ ,  $y = 0$  și  $y = 1$ . Se cere suprafața prin ecuația explicită.

Avem deci de determinat ecuația suprafeței Coons generală în care avem:

$$P_{00}^y = 1; P_{01}^y = -1; P_{10}^y = P_{11}^y = 0; P_{00}^z = P_{10}^z = P_{01}^z = P_{11}^z = 0$$

Alegem mai departe tangentele horizontale la curbele marginale principale  $P_{0y}$  și  $P_{1y}$ . Din acest fapt decurge:

$$P_{0y}^z = P_{1y}^z = 0$$

și de aceea:

$$P_{00}^{zy} = P_{01}^{zy} = P_{10}^{zy} = P_{11}^{zy} = 0$$

Funcția  $P_{x0}^y$  trebuie să îndeplinească condiția marginală: pentru  $x = 0$  să ia valoarea 1, pentru  $x = 1$  valoarea 0.

Derivata  $P^{xy}$  trebuie să fie nulă pentru  $x = 0$  și  $x = 1$ .

Această calitate o îndeplinește funcția  $F_1(x)$  pe care o alegem ca funcție  $P_{x0}^y$ . Analog se alege  $P_{x1}^y = F_2(x) - 1$ . Astfel elementele matricii  $\bar{C}_{5,5}$  vor fi:

$$\bar{C}_{3,3} = \begin{bmatrix} 0, & y(1-y), & 0, & 1, & -1 \\ 0, & Z, & 0, & F_1(x) & F_2(x) - 1 \\ 0, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0 \end{bmatrix}$$

Din ecuația  $\bar{U}\bar{C}_{5,5}\bar{V} = 0$  deducem ecuația explicită a acestei suprafețe Coons generale:

$$Z = F_1(x)y(1-y)$$

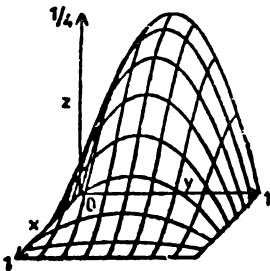


Fig. 10-32

În planele  $y = \text{constant}$  curbele principale sînt cubice, iar în planele  $x = \text{constant}$  curbele principale sînt parabole.

Imaginea perspectivă a suprafeței este redată în figura 10.32.

10.11.2. Aplicație

Să se determine ecuația suprafeței Coons generală definită de matricea restricțiilor  $\bar{C}_{5,5}$  următoare în care secțiunea vectorilor derivatelor mixte este  $\begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix}$ :

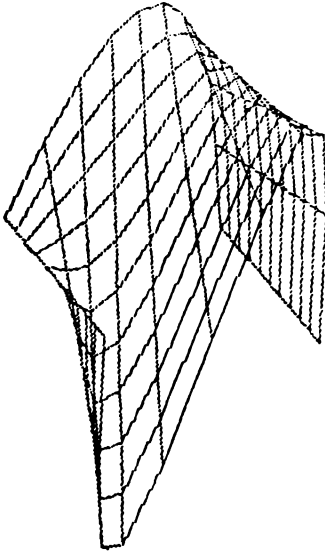


Fig. 10.33

$$\bar{C}_{5,5} = \begin{bmatrix} 0 & 4y(1-y) & 0 & 4 & -4 \\ 0 & Z & 0 & (2\pi - 4)x + 4 & (2\pi + 4)x - 4 \\ 0 & \sin(2\pi y) & 0 & 2\pi & 2\pi \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Prin înmulțirea matricelor și ordonare obținem ecuația suprafeței Coons generală

$$Z = \sin(2\pi y) \cdot (-2x^3 + 3x^2) + y[4 + x(2\pi - 4) + x^2(1 - 6\pi) + x^3(4\pi - 1)] + y^2[-4 + x(5 - 6\pi) + x^2(18\pi - 4) + x^3(3 - 12\pi)] + y^3[x(4\pi - 1) + x^2(3 - 12\pi) + x^3(8\pi - 2)]$$

Imaginea suprafeței este redată în figura 10.33.

10.11.3. Aplicație

Să se construiască suprafața netedă compusă prin cuplarea petecilor de suprafață definite pe cele cinci domenii din figura 10.34. Primul domeniu este mărginit de segmentele care leagă punctul de colț  $P_{00} = (0, 0, 0)$  cu  $P_{10} = (1, 0, 0)$  și  $P_{00}$  cu  $P_{01} = (0, 1, 0)$ . Celelalte două curbe marginale sînt cubicele  $P_{x1}$  respectiv  $P_{y1}$  în planul  $y = 1$  respectiv  $x = 1$  care au punctul final  $P_{11} = (1, 1, -0, 2)$  tangenta orizontală ca și punctul inițial  $P_{01}$  sau  $P_{10}$ .

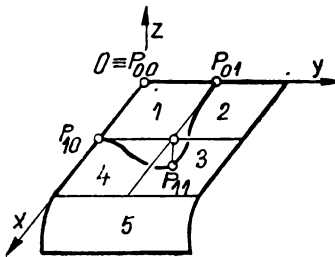


Fig. 10.34

Ecuatiile acestor cubice sînt respectiv:

$$Z = 0,4x^3 - 0,6x^2 \quad \text{și} \quad Z = 0,4y^3 - 0,6y^2$$

Curbele marginale pentru celelalte domenii se obțin prin simetrie față de planele  $x = 1$  și  $y = 1$ . Placa a cincea este generată de un cilindru parabolic, iar parabola, directoare este în planul  $(xz)$  care conține și punctele  $(2, 0, 0)$  și  $(2, 2; 0; -0, 3)$ . În primul punct parabola are ca tangentă axa  $x$ . Matricea restricțiilor este (pentru prima placă):

$$\begin{bmatrix} 0, & 0, & 0, & 0, & 0 \\ 0, & Z(x, y) & (0,4x^3 - 0,6x^2) & 0, & 0 \\ 0, & (0, 4y^3 - 0,6y^2) & -0,2 & 0, & 0 \\ 0, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0 \end{bmatrix}$$

În aceste condiții ecuația explicită a suprafeței este:

$$- [-Z(x, y) + 0,4y^3 - 0,6y^2] \cdot F_2(x) - (0,4x^3 - 0,6x^2 - 0,2)F_3(y) = 0$$

și după aranjarea și ordonarea termenilor:

$$Z(x, y) = 1,6x^3y^3 + 2,4x^3y^2 + 2,4x^2y^3 - 3,6x^2y^2 - 0,4y^3 + 0,6y^2$$

Analog se determină ecuațiile celorlalte trei plăci ale suprafețelor.

Ecuația suprafeței cilindrice este:

$$Z = -7,5(x - 2)^2$$

## 10.12. Cubice Ferguson marginale directe

Cubica Ferguson este definită de punctele inițial și final al curbei și de tangentele în aceste puncte. Fie  $P_1$  și  $P_2$  cele două puncte și respectiv  $P'_1$  și  $P'_2$  cele două tangente în aceste puncte.

Ecuația vectorială a cubicei Ferguson este:

$$\bar{P}(t) = \bar{P}_0F_1(t) + \bar{P}_1F_2(t) + \bar{P}'_0F_3(t) + \bar{P}'_1F_4(t), \quad t \in [0, 1]$$

în care funcțiile  $F_i$ ,  $i = 1, 2, 3, 4$  sînt polinoamele cubice cunoscute.

Putem utiliza aceste cubice Ferguson drept curbe marginale directe ale suprafeței.

### 10.12.1. Aplicație

Să considerăm imaginea din figura 10.35 pe care am reprezentat două curbe Ferguson și două semicercuri. Să se determine cele două variante de suprafețe pentru:

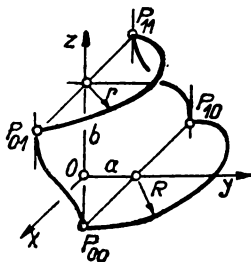


Fig. 10.35

$$P_{00}(R, a, 0); \quad P_{10}(-R, a, 0)$$

$$P_{01}(r, 0, b); \quad P_{11}(-r, 0, b)$$

$$P_{u0}(R \cos \pi u; a + R \sin \pi u; 0)$$

$$P_{u1}(r \cos \pi u; r \sin \pi u; b)$$

Ecuațiile cubicelor Ferguson sînt:

$$\bar{P}_{0v} = \bar{P}_{00}F_1(v) + \bar{P}_{01}F_2(v) + \bar{P}'_{00}F_3(v) + \bar{P}'_{01}F_4(v)$$

$$\bar{P}_{1v} = \bar{P}_{10}F_1(v) + \bar{P}_{11}F_2(v) + \bar{P}'_{10}F_3(v) + \bar{P}'_{11}F_4(v)$$

Să dăm vectorii tangenți:

$$\bar{P}'_1 = [0, 0, p_1] \quad \text{și} \quad \bar{P}'_2 = [0, 0, p_2]$$

unde  $p_1$  și  $p_2$  sînt lungimile lor.

În aceste condiții cubicele marginale Ferguson devin definite prin vectorii:

$$\bar{P}_{0v} = [rF_1(v) + rF_2(v); aF_1(v); bF_2(v) + p_1F_3(v) + p_2F_4(v)]$$

$$\bar{P}_{1v} = [-rF_1(v) - rF_2(v); aF_1(v); bF_2(v) + p_1F_3(v) + p_2F_4(v)]$$

„Amprentele” pentru coordonatele  $x$ ,  $y$  și  $z$  sînt matricele:

$$\begin{bmatrix} R, & rF_1(v) + rF_2(v), & r \\ R \cos \pi u, & x, & r \cos \pi u \\ -R, & -rF_1(v) - rF_2(v), & -r \\ a, & aF_1(v), & 0 \\ a + R \sin \pi u, & y, & r \sin \pi u \\ a, & aF_1(v), & 0 \\ 0, & bF_2(v) + p_1F_3(v) + p_2F_4(v), & b \\ 0, & z, & b \\ 0, & bF_2(v) + p_1F_3(v) + p_2F_4(v), & b \end{bmatrix}$$

Substituind aceste matrice în ecuația suprafeței bicubice Coons definită prin curbele marginale:

$$[F_1(u), -1, F_2(u)] \bar{C}_{3,3} [F_1(v), -1, F_2(v)]^T = \bar{0} \quad 0 \leq u; \quad v \leq 1$$

obținem:

$$\begin{aligned} x &= [RF_1(v) + rF_2(v)] \cos \pi u \\ y &= aF_1(v) + [RF_1(v) + rF_2(v)] \sin \pi u \\ z &= bF_2(v) + p_1F_3(v) + p_2F_4(v) \end{aligned}$$

Alegem  $a = 1$ ,  $b = 2$ ,  $r = 1$ ,  $R = 2$ ,  $p_1 = 2$ ,  $p_2 = 1$  pentru prima variantă și  $p_2 = 10$  pentru a doua variantă.

Ecuatiile parametrice ale celor două suprafețe vor fi:

$$\begin{aligned} x &= (2v^3 - 3v^2 + 2) \cos \pi u \\ y &= 2v^3 - 3v^2 + 1 + (2v^3 - 3v^2 + 2) \sin \pi u \\ z &= -v^3 + v^2 + 2v \quad \text{pentru prima variantă} \\ z &= 8v^3 - 8v^2 + 2v \quad \text{pentru a doua variantă} \end{aligned}$$

Imaginile suprafețelor sînt redată în perspectivă în figurile 10.36 și 10.37.

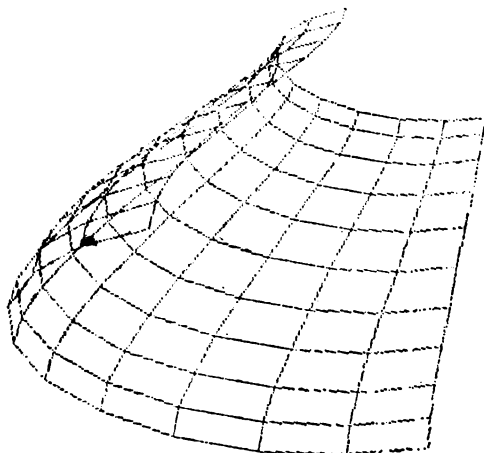


Fig. 10.36

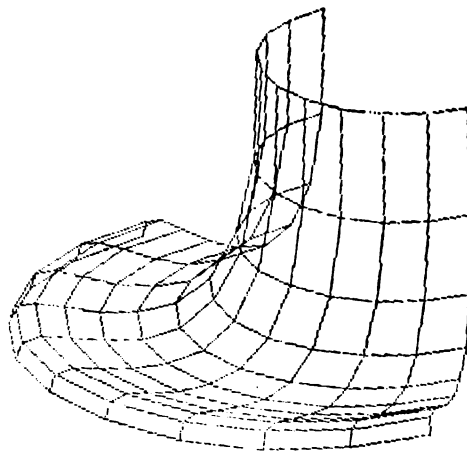


Fig. 10.37

### 10.12.2. Suprafețe speciale derivate din cubicele Ferguson

Să considerăm suprafața definită prin două curbe marginale opuse constante care se aleg drept curbe directe. Putem alege pe aceste curbe directe perechi de puncte care împreună cu tangentele în aceste puncte definesc curbe cubice Ferguson. Cu alte cuvinte prin această definire am înlăturat două din cele patru curbe marginale Ferguson (opuse).

Fie suprafața exprimată parametric prin ecuațiile:

$$\begin{aligned} x &= u(2u^2 - 3u + 2) \\ y &= (2v - 1) \cdot F_1(u)/2 \\ z &= 4v(1 - v) \cdot F_1(u) \end{aligned}$$

unde cubica Ferguson  $F_1(u)$  este:

$$F_1(u) = 2u^3 - 3u^2 + 1$$

Perspectiva axonometrică izometrică a acestei suprafețe este dată în figura 10.38.

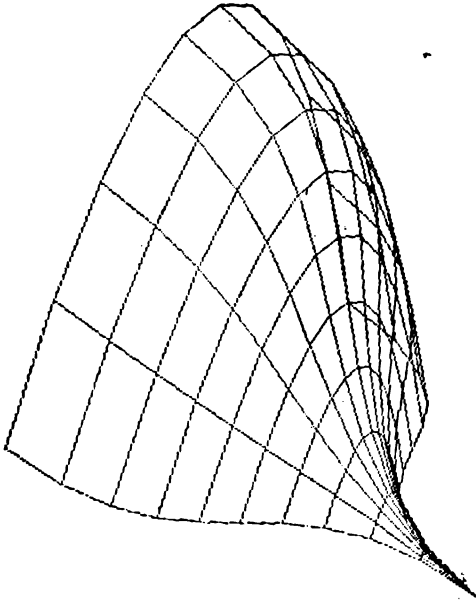


Fig. 10.38

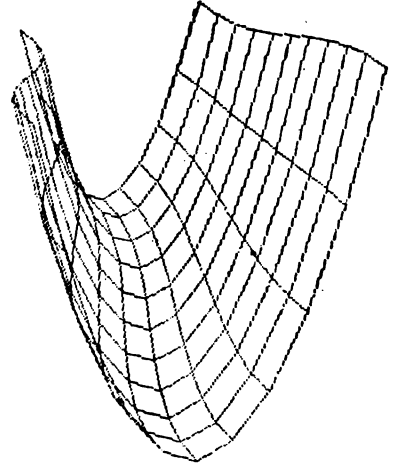


Fig. 10.39

Remarcă:

Dacă în funcția  $F_1(u) = 2u^3 - 3u^2 + 1$  modificăm  $(+1)$  în  $(-1)$  obținem cu aceleași ecuații parametrice suprafața ilustrată prin perspectiva paralelă din figura 10.39 obținută pentru parametrii directori  $AL = 0,650$ ,  $BE = 0,500$  și  $GA = 0,450$  (trimetrie). Așadar putem obține infinit de multe suprafețe nu numai schimbând curbele directoare și tangentele în punctele curbilor Ferguson, ci și prin schimbarea anumitor elemente în funcțiile determinate. Pe această cale pot fi descoperite forme plastice nebanuite pentru grafică în general sau pentru pînzele subțiri în construcții și arhitectură.

### 10.13. Cubice Bézier marginale directoare.

Să considerăm cubica Bézier:

$$\bar{P}(t) = \bar{V}_1 B_1(t) + \bar{V}_2 B_2(t) + \bar{V}_3 B_3(t) + \bar{V}_4 B_4(t) = \sum_{i=1}^4 \bar{V}_i B_i(t)$$

$$t \in [0, 1]$$

pe care o putem alege drept curbă marginală directoare împreună cu trei segmente de dreaptă alese, de asemenea, curbe marginale.

Și în acest caz pot fi definite două variante de suprafețe considerind, de exemplu, cubica Bézier determinată de punctele:

$$V_2(1, 0, 1); V_3(1, 1, 2) \text{ — în prima variantă}$$

$$V_2(-1, 0, 1); V_3(-1, 1, 2) \text{ — în varianta a doua.}$$

Celelalte curbe marginale sînt segmentele în planul:

$$Z = 0; x = 1; y = 0 \text{ și } y = 1, \text{ iar } V_1 = P_{00} \text{ și } V_4 = P_{10}$$



Procedînd la fel ca în cazul cubicelor Ferguson, obținem ecuațiile parametrice ale suprafețelor în cele 2 variante:

$$\begin{aligned} x &= (2u^3 - 3u^2 + 1)(1-v) + 3v + u = u + p \\ y &= v[v(3-2v)(2u^3 - 3u^2 + 1) + 3u^2 - 2u^3] \\ z &= (2u^3 - 3u^2 + 1)3v(1-v^2) \end{aligned}$$

Pentru a doua variantă avem  $x = u - p$ , iar  $y$  și  $z$  sînt la fel ca în prima variantă. Imaginile perspective ale suprafețelor sînt redată pe figurile 10.40.a și 10.40.b, 10.40.c.

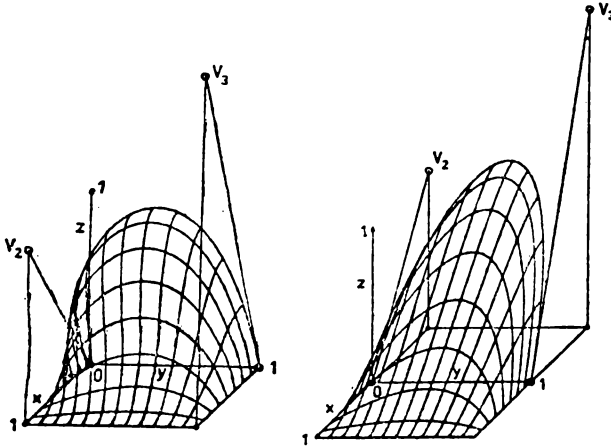


Fig. 10.40 ab

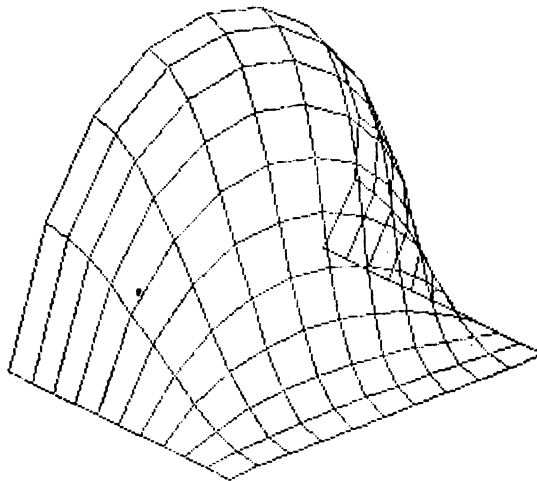


Fig. 10.40 c

### 10.14. Curbe Coons marginale directoare

Curbele Coons (marginale directoare) sînt asemănătoare cu curbele Bézier și sînt definite prin linia frîntă (poligonul director), care are vîrfurile  $V_1$ ,

$V_2, \dots, V_n$  ( $n > 4$ ). Ecuația curbei este studiată în cadrul curbelor Coons generale. Considerînd aplicația în care am determinat curba Coons care trece prin 10 puncte date putem alege această curbă Coons marginală directoare. Celelalte trei curbe marginale ale suprafeței sînt segmente de dreaptă. Vor rezulta două variante de suprafețe din cauza modificării poligonului director.

Astfel curba marginală  $P_{u0}$  este:

$$\bar{P}_{u0} = \bar{C}[c_x, c_y, c_z]$$

Din matricea „amprentă” a suprafeței:

$$\begin{bmatrix} \bar{P}_{00}, & \bar{P}_{0v}, & \bar{P}_{01} \\ \bar{C}, & \bar{P}(u,v), & \bar{P}_{u1} \\ \bar{P}_{10} & \bar{P}_{1v} & P_{11} \end{bmatrix}$$

obținem ecuațiile parametrice explicite ale suprafeței:

$$\begin{bmatrix} 0, & 0, & 0 \\ c_x, & x, & u \\ 1, & 1, & 1 \end{bmatrix}; \begin{bmatrix} 0, & v, & 1 \\ c_y, & y, & 1 \\ 0, & v, & 1 \end{bmatrix}; \begin{bmatrix} 0, & 0, & 0 \\ c_z, & z, & 0 \\ 0, & 0, & 0 \end{bmatrix}$$

din care după efectuarea calculului rezultă:

$$\begin{cases} x = c_x F_1(v) + u F_2(v) \\ y = c_y F_1(v) + v \\ z = c_z F_1(v) \end{cases}$$

Imaginile perspective ale suprafeței în cele două variante sînt redată în figurile 10.41.a și 10.41.b.

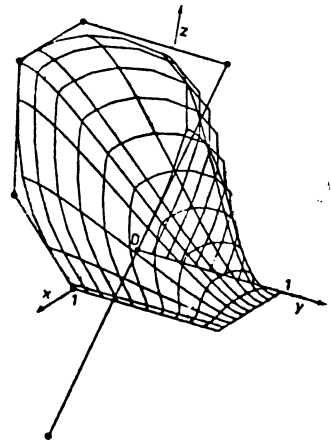
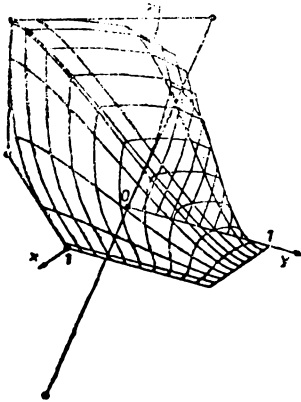


Fig. 10.41 a b

## 10.15. Suprafețe definite prin rețele de puncte

### 10.15.1. Placa bicubică Bézier

Suprafața bicubică Bézier este definită prin 16 noduri  $U_{ij}$  ( $i, j = 1, 2, 3, 4$ ) care compun o rețea de puncte formînd nouă petice (părți) de suprafață (figura 10.42).

Ecuția suprafeței este de forma:

$$\bar{P}(u, v) = [B_1(u), B_2(u), B_3(u), B_4(u)] \times \\ \times \bar{B}_{4,4} [B_1(v), B_2(v), B_3(v), B_4(v)]^T$$

unde (fig. 10.43):

$$B_1(t) = (1 - t)^3 \\ B_2(t) = 3t(1 - t^2) \\ B_3(t) = 3t^2(1 - t) \\ B_4(t) = t^3$$

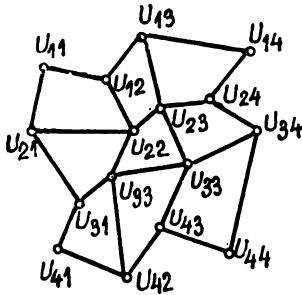


Fig. 10.42

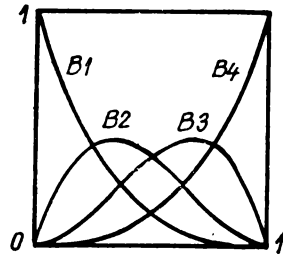


Fig. 10.43

„Amprenta” suprafeței este matricea:

$$\bar{B}_{4,4} = \begin{bmatrix} \bar{U}_{11} & \bar{U}_{12} & \bar{U}_{13} & \bar{U}_{14} \\ \bar{U}_{21} & \bar{U}_{22} & \bar{U}_{23} & \bar{U}_{24} \\ \bar{U}_{31} & \bar{U}_{32} & \bar{U}_{33} & \bar{U}_{34} \\ \bar{U}_{41} & \bar{U}_{42} & \bar{U}_{43} & \bar{U}_{44} \end{bmatrix}$$

în care elementele în fiecare loc răspund nodurilor rețelei din figura 10.42.

### 10.15.2. Calitățile plăcii bicubice Bézier

Suprafața este bicubică, iar punctele de colț ale rețelei sînt puncte de colț ale suprafeței.

Să alegem, de exemplu,  $u = v = 0$ . Deoarece  $B_1(0) = 1$  și  $B_2(0) = B_3(0) = B_4(0) = 0$  rezultă prin substituție în ecuația suprafeței

$$\bar{P}(0, 0) = \bar{U}_{11}$$

deci punctul nod de colț  $U_{11}$  este punctul de colț  $P(0, 0)$  al suprafeței. La fel se poate demonstra că și celelalte puncte de colț ale rețelei sînt puncte de colț ale suprafeței.

Curbele marginale ale suprafeței  $\bar{P}(u, v)$  sînt *cubicile Bézier* definite prin marginile respective ale rețelei de puncte.

Să alegem  $u = 0$ . Curba marginală are ecuația:

$$\bar{P}(0, v) = [1, 0, 0, 0] \cdot \bar{B}_{4,4} \cdot [B_1(v), B_2(v), B_3(v), B_4(v)]^T$$

astfel încît:

$$\bar{P}(0, v) = \bar{U}_{11}B_1(v) + \bar{U}_{12}B_2(v) + \bar{U}_{13}B_3(v) + \bar{U}_{14}B_4(v)$$

adică am obținut cubica Bézier definită de punctele  $V_i = U_{1i}$ ,  $i = 1, 2, 3, 4$ .

Deoarece valorile funcției  $B_i(t)$  sînt pozitive pentru  $0 \leq t \leq 1$ , se amplifică sau se reduc valorile funcției  $\bar{P}(u, v)$ , deci pot fi amplificate sau reduse valorile fiecărui element al matricei „amprentă”.

Curbele parametrice ale suprafeței  $\bar{P}(u, v)$  sînt cubice față de funcțiile cubice  $B_i$ . După cum s-a precizat mai sus, suprafața este bicubică.

### 10.15.3. Aplicație

Să se determine suprafața Bézier sub formă explicită considerînd  $u = x$ ,  $v = y$  și matricea nodurilor:

$$\bar{B}_{4,4} = \begin{bmatrix} 0, & 1, & 1, & 0 \\ 1, & 0, & 1, & 0 \\ -1, & -1, & 2, & 2, \\ 0, & -2, & -1, & 0 \end{bmatrix}$$

Înlocuind și făcînd înmulțirile se obține ecuația explicită a suprafeței Bézier sub forma:

$$Z = 3x(1 - 3x + 2x^2) + 3y(1 - 6x + 9x^2 + 6x^3) + 3y^2(-1 + 9x - 6x^2 + x^3) + 3y^3(-4x + 2x^2 + x^3)$$

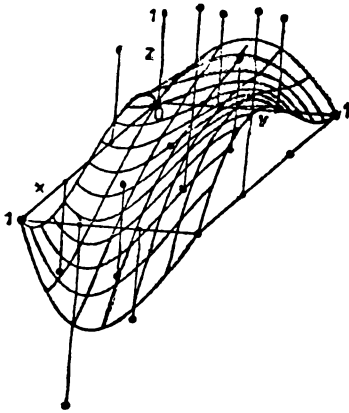


Fig. 10.44

În figura 10.44 sînt reprezentate imaginile perspective ale nodurilor inclusiv punctele de colț ale suprafeței.

Dacă, de exemplu, se transformă nodul arbitrar  $U_{23}$  astfel încît cota sa să fie zero, se transformă și valoarea funcției  $B_2(z)$   $B_3(y)$ , adică în expresia  $3x(1 - x^2) 3y^2(1 - y)$ . Punctul de proiecție orizontală  $\left(\frac{1}{3}, \frac{2}{3}\right)$  se trans-

formă în  $\frac{16}{81} = 0,2$ .

### 10.15.4. Placa bicubică Coons determinată printr-o rețea

Asemănător cu suprafața bicubică Bézier putem defini suprafața placă bicubică Coons prin ecuația:

$$\bar{P}(u, v) = [C_1(u), C_2(u), C_3(u), C_4(u)] \times \begin{bmatrix} \bar{U}_{11} & \bar{U}_{12} & \bar{U}_{13} & \bar{U}_{14} \\ \bar{U}_{21} & \bar{U}_{22} & \bar{U}_{23} & \bar{U}_{24} \\ \bar{U}_{31} & \bar{U}_{32} & \bar{U}_{33} & \bar{U}_{34} \\ \bar{U}_{41} & \bar{U}_{42} & \bar{U}_{43} & \bar{U}_{44} \end{bmatrix} \cdot \begin{bmatrix} C_1(v) \\ C_2(v) \\ C_3(v) \\ C_4(v) \end{bmatrix} / 36$$

în care elementele matricei sînt cubicele Coons, iar matricea „amprentă” a suprafeței conține  $4 \times 4$  noduri într-o rețea dată.

Această suprafață de interpolare nu trece prin nodurile date, similar cu cubica Coons care nu trece nici ea prin virfurile poligonului caracteristic (în cazul general).<sup>‡</sup>

## Aplicație

Să se determine suprafața placă bicubică Coons prin ecuațiile parametrice  $x, y, z$  pornind de la rețeaua de  $4 \times 4$  noduri utilizată pentru determinarea suprafeței  $P_{3,3}$  din „Aplicația 10.3.3” „Amprente” suprafeței căutate pentru coordonatele  $x, y, z$  sînt următoarele:

$$\bar{M}_x = \begin{bmatrix} 0, & 0, & 0, & 0 \\ \frac{1}{3}, & \frac{1}{3}, & \frac{1}{3}, & \frac{1}{3} \\ \frac{2}{3}, & \frac{2}{3}, & \frac{2}{3}, & \frac{2}{3} \\ 1, & 1, & 1, & 1 \end{bmatrix} \quad \bar{M}_y = \begin{bmatrix} 0, & \frac{1}{3}, & \frac{2}{3}, & 1 \\ 0, & \frac{1}{3}, & \frac{2}{3}, & 1 \\ 0, & \frac{1}{3}, & \frac{2}{3}, & 1 \\ 0, & \frac{1}{3}, & \frac{2}{3}, & 1 \end{bmatrix}$$

$$\bar{M}_z = \begin{bmatrix} 0, & 0, & 0, & 0 \\ 0, & K, & 0, & 0 \\ 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0 \end{bmatrix}$$

Ecuția parametrică pentru coordonata  $x$  este:

$$x = [C_1(u), C_2(u), C_3(u), C_4(u)] \cdot [\bar{M}_x] \cdot [C_1(v), C_2(v), C_3(v), C_4(v)]^T$$

sau

$$x = \frac{1+u}{3}$$

Analog se determină ecuațiile parametrice pentru coordonatele  $y$  și  $z$ . Astfel:

$$y = \frac{1+v}{3}$$

$$z = K(3u^3 - 6u^2 + 4) \frac{3v^3 - 6v^2 + 4}{36}$$

Imaginea perspectivă a acestei suprafețe pentru  $K = 1$  este redată (în figurile 10.45 a și 10.45.b)

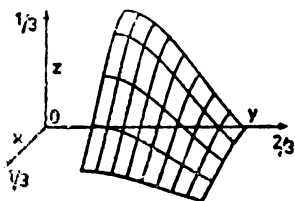


Fig. 10.45 a

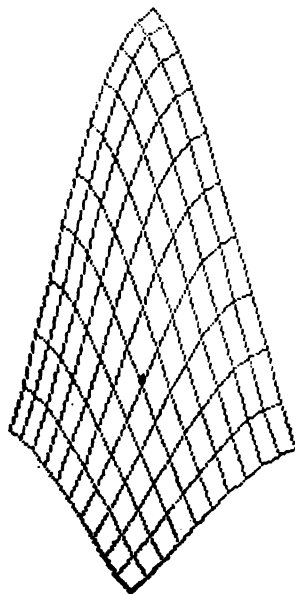


Fig. 10.45 b

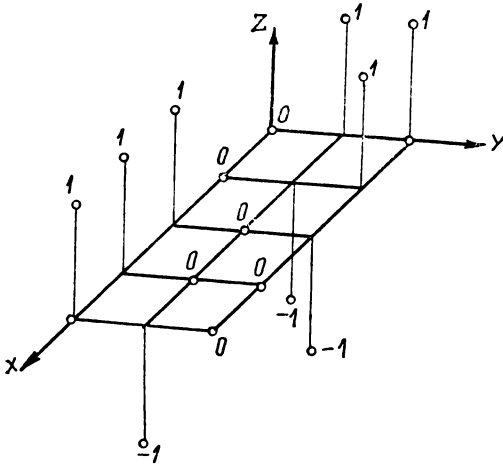


Fig. 10.46

### 10.15.5. Suprafața Bézier rețea 5 × 3. Aplicație

Să considerăm suprafața Bézier rețea 5 × 3 definită prin rețeaua punctelor  $B_{ij}$   $i = 1, 2, 3, 4, 5$ ;  $j = 1, 2, 3$  (figura 10.46).

Suprafața Bézier rețea  $\{5 \times 3\}$  se exprimă explicit prin ecuația:

$$\bar{Z} = [P(x), R(x), S(x), T(x), V(x)] \times \bar{B}_{5,3} \times [I(y), J(y), K(y)]^T$$

în care funcțiile  $P, R, S, T, V$  sînt cuarticele din fig. 10.47:

$$P(x) = (1-x)^4 = B_1(u)$$

$$R(x) = 4x(1-x)^3 = B_2(u)$$

$$S(x) = 6x^2(1-x)^2 = B_3(u)$$

$$T(x) = 4x^3(1-x) = B_4(u)$$

$$V(x) = 4x^3(1-x) = B_4(u)$$

iar funcțiile  $I, J, K$  sînt parabolele din figura 10.48.

$$I(y) = (1-y)^2 = B_1(v)$$

$$J(y) = 2y(1-y) = B_2(v)$$

$$K(y) = y^2 = B_3(v)$$

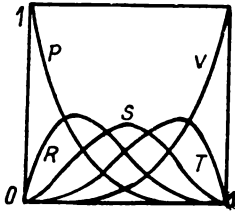


Fig. 10.47

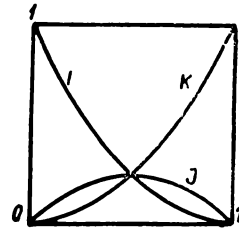


Fig. 10.48

iar  $\bar{B}_{5,3}$  este matricea coordonatelor în rețeaua de puncte

$$\bar{B}_{5,3} = \begin{bmatrix} Z_{11} & Z_{12} & Z_{13} \\ Z_{21} & Z_{22} & Z_{23} \\ Z_{31} & Z_{32} & Z_{33} \\ Z_{41} & Z_{42} & Z_{43} \\ Z_{51} & Z_{52} & Z_{53} \end{bmatrix} \bar{U}_{11} = \bar{Z}_{11}, \bar{U}_{21} = \bar{Z}_{21}, \dots$$

în care  $Z_{ij}$  este cota punctului  $B_{ij}$ ;  $i = 1, 2, 3, 4, 5$ ;  $j = 1, 2, 3$ .

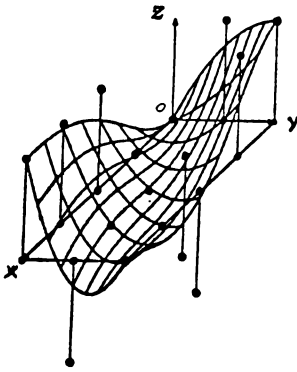


Fig. 10.49

Ca aplicație să determinăm suprafața Bézier a cărei rețea  $5 \times 3$  de puncte este dată în figura 10.46.

Matricea  $B_{5,3}$  este după relația de mai sus

$$\bar{B}_{5,3} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & -1 & 1 \\ 1 & 0 & -1 \\ 1 & 0 & 0 \\ -1 & -1 & 0 \end{bmatrix}$$

Prin înmulțirea matricelor și substituirea în funcțiile  $P, R, S, T, V, I, J, K$  și prin ordonare obținem ecuația explicită a suprafeței Bézier  $5 \times 3$ :

$$Z = x^2(6 - 8x + 3x^2) + 2y(1 - 8x + 12x^2 - 8x^3 + x^4) + y^2 \cdot (-1 + 16x - 42x^2 + 44x^3 - 14x^4)$$

Imaginea suprafeței este redată în figura 10.49.

## 10.16. Suprafețele Bézier. Generalități

Metoda lui Bézier pentru construirea interactivă a curbelor poate fi ușor extinsă pentru proiectarea suprafețelor cu formă liberă. Această generalizare la suprafețe a curbelor Bézier a fost prezentată în cadrul studiului curbelor Bézier. Dacă  $F: [0, 1] \times [0, 1] \rightarrow R^3$  este o funcție vector de valori din care se extrag valorile pentru aproximarea Bézier și dacă se consideră produsul cartesian, avem:

$$Q(u, v) = \sum_{i=0}^m \sum_{j=0}^n F\left(\frac{i}{m}, \frac{j}{n}\right) \binom{m}{i} u^i (1-u)^{m-i} \binom{n}{j} v^j (1-v)^{n-j}$$

Astfel multe din proprietățile cazului unidimensional pot fi ușor aduse în aproximarea bivariată. De exemplu suprafața  $Q(u, v)$  coincide cu funcția  $F$  în general numai în cele patru colțuri ale pătratului unitar  $(u, v)$ . Punctele  $F\left(\frac{i}{m}, \frac{j}{n}\right)$  sînt vîrfurile unei fețe  $mn$  ale segmentelor de linie, jucînd același rol cu poligonul Bézier din desenul curbei.

Figura 10.50 ilustrează de exemplu o producere a unei suprafețe Bézier. Partea de sus figurii arată suprafața bilineară pe porțiuni a echivalentului bivariat al poligonului Bézier.

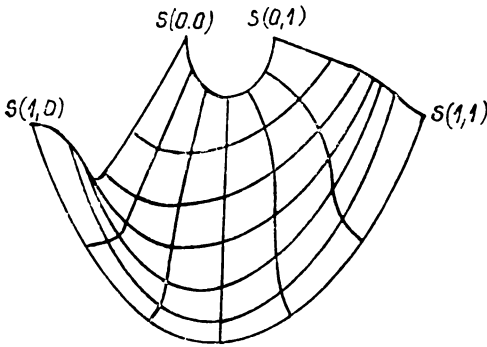
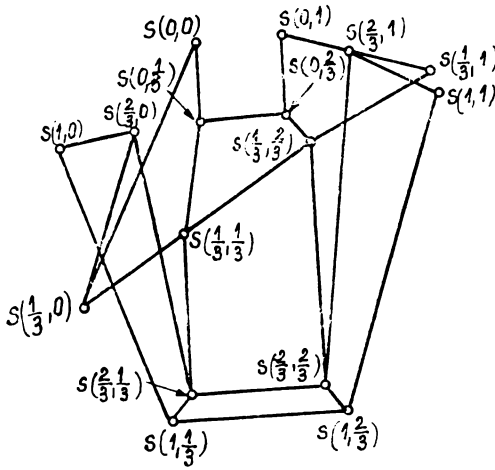


Fig. 10.50

Partea din dreapta figurii arată suprafața bicubică asociată.

Formula de aproximare a suprafeței în cazul când se consideră suma booleană în loc de produsul cartesian este:

$$\begin{aligned}
 Q(u, v) = & \\
 = & \sum_{i=0}^m \binom{m}{i} u^i (1-u)^{m-i} F\left(\frac{1}{m}, v\right) + \\
 & + \sum_{j=0}^n \binom{n}{j} (1-v)^{n-j} \cdot F\left(u, \frac{j}{n}\right) - \\
 & - \sum_{i=0}^m \sum_{j=0}^n \binom{m}{i} u^i (1-u)^{m-i} \binom{n}{j} \times \\
 & \times v^j (1-v)^{n-j} F\left(\frac{i}{m}, \frac{j}{n}\right)
 \end{aligned}$$

10.16.1. Suprafața Bézier generală

Suprafața Bézier generală este definită de ecuația:

$$\begin{aligned}
 \bar{P}(u, v) = & [B_1^m(u), \dots, B_m^m(u)] \times \\
 & \bar{B}_{m,n} \cdot [B_1^n(v), \dots, B_n^n(v)]^T
 \end{aligned}$$

Elementele matricelor sînt funcții. Elementele matricei „amprentă” a suprafeței sînt nodurile rețelei date în:

$$\bar{B}_{m,n} = \begin{bmatrix} \bar{U}_{11}, & \dots, & \bar{U}_{1n} \\ \dots & \vdots & \dots \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ \bar{U}_{m1}, & \dots, & \bar{U}_{mn} \end{bmatrix}$$

Această suprafață are calități similare cu suprafața placă bicubică Bézier.

Suprafața conține nodurile  $U_{11}, U_{1n}, U_{m1}, U_{mn}$ .

Astfel pentru  $u = v = 0$  și deoarece  $B_1^m(0) = 1, B_m^m(0) = 1$ , iar celelalte funcții sînt nule în origine avem  $\bar{P}(0, 0) = \bar{U}_{11}$ , deci punctele de colț ale suprafeței sînt puncte de colț ale rețelei nodurilor. În mod analog se poate face demonstrația și pentru celelalte noduri de colț  $U_{1n}, U_{m1}$  și  $U_{mn}$ .



Curbele marginale sînt curbe Bézier definite de laturile marginale ale rețelei. Astfel, pentru  $u = 0$  și deoarece  $B_1^m(0) = 1$  iar celelalte valori ale funcției Bézier sînt nule pentru  $u = 0$ , obținem curba Bézier.

$$\bar{P}(0, v) = \bar{U}_{11}B_1^n(v) + \dots + \bar{U}_{1n}B_n^n(v)$$

pentru  $V_1 = U_{11}, \dots, V_n = U_{1n}$ .

Analog se poate demonstra că și celelalte curbe marginale sînt curbe Bézier.

Curbele parametrice ale suprafeței sînt curbe algebrice de gradul  $m - 1$  respectiv  $n - 1$ .

Deoarece valorile funcției  $B_i^p(t)$  sînt pozitive pentru  $0 \leq t \leq 1$ , se amplifică sau se reduc valorile funcției  $P(u, v)$ , deci pot fi amplificate sau reduse valorile fiecărui element al matricei „amprentă”.

### 10.16.2. Aplicație

Să se determine suprafața Bézier generală pentru  $m = n = 2$  definită prin cele patru noduri de colț  $U_{11}, U_{12}, U_{22}, U_{21}$ .

Ecuția suprafeței este:

$$\bar{P}(u, v) = [1 - u, u] \cdot \begin{bmatrix} \bar{U}_{11} & \bar{U}_{12} \\ \bar{U}_{21} & \bar{U}_{22} \end{bmatrix} \cdot \begin{bmatrix} 1 - v \\ v \end{bmatrix}$$

Atît timp cît nodurile nu sînt în plan, suprafața este o cuadrică riglată.

### 10.16.3. Aplicație

Să se determine suprafața Bézier generală pentru  $m = 3$  și  $n = 2$  definită prin următoarele șase noduri (fig. 10.51)

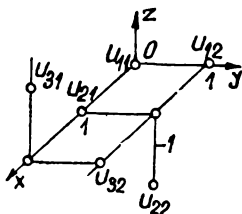


Fig. 10.51

$$\begin{array}{ll} U_{11} (0, 0, 0); & U_{12} (0, 1, 0) \\ U_{21} (1, 0, 0); & U_{22} (1, 1, -1) \\ U_{31} (2, 0, 1); & U_{32} (2, 1, 0) \end{array}$$

Din ecuația  $\bar{P}(u, v)$  a suprafeței Bézier generală obținem pentru  $m = 3$  și  $n = 2$ :

$$\bar{P}(u, v) = [1 - u]^2, 2u(1 - u), u^2 \cdot \begin{bmatrix} \bar{U}_{11} & \bar{U}_{12} \\ \bar{U}_{21} & \bar{U}_{22} \\ \bar{U}_{31} & \bar{U}_{32} \end{bmatrix} \cdot \begin{bmatrix} 1 - v \\ v \end{bmatrix}$$

Efectuînd înmulțirile obținem ecuațiile parametrice ale ecuației după înlocuirea valorilor date:

$$\begin{array}{l} x = 2u \\ y = v \\ z = u(u - 2v + uv) \end{array}$$

Imaginea perspectivă a suprafeței și câteva curbe parametrice ale suprafeței (drepte și parabole), este reprezentată în figura 10.52.

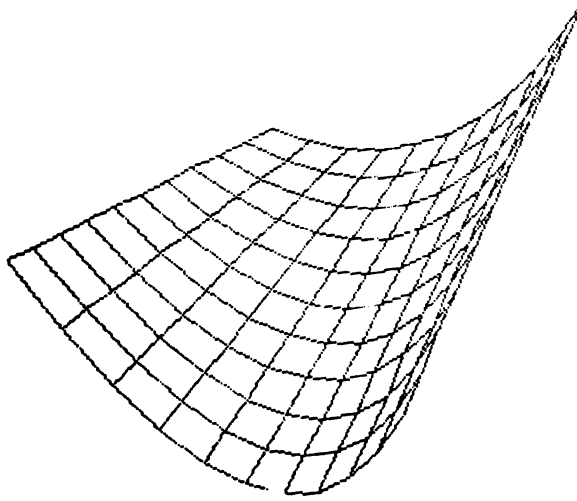


Fig. 10.52

#### 10.16.4. Suprafața placă Bézier

Să considerăm proprietățile suprafeței Bézier și plăcile alipite din figură cu două segmente marginale conținute în planul  $xy$  ( $U_{11} - U_{13}$  și  $U_{13} - U_{23} - U_{33}$ ). Plăcile  ${}^1P(u, v)$  și  ${}^2P(u, v)$  au poziție alăturată în care caz curba marginală comună  ${}^1P(u, 1)$  și  ${}^2P(u, 0)$  induce pentru noduri condițiile:

$${}^1U_{1n} = {}^2U_{11}, \dots, {}^1U_{mn} = {}^2U_{m1}$$

Placa netedă impune în lungul acestei curbe colinearitatea nodurilor:

$${}^1U_{1,n-1} \quad {}^1U_{1n} = {}^2U_{11}, {}^2U_{12}; \dots; {}^1U_{m,n-1}, {}^1U_{mn} = {}^2U_{m1}, {}^2U_{m2}$$

#### 10.16.5. Aplicație

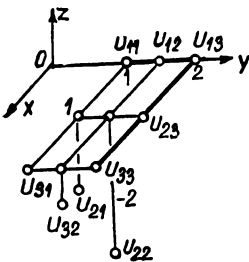


Fig. 10.53

Să se determine ecuațiile parametrice ale suprafețe placă Bézier pentru  $m = n = 3$  definită prin următoarele nouă noduri reprezentate pe figura 10.53.

$$U_{11} (0, 1, 0); \quad U_{12} (0; 1,5; 0); \quad U_{13} (0, 2, 0)$$

$$U_{21} (1, 1, -1); \quad U_{22} (1; 1,5; -1,5); \quad U_{23} (1, 2, 0)$$

$$U_{31} (2, 1, 0); \quad U_{32} (2; 1,5; -0,5); \quad U_{33} (2, 2, 0)$$

Matricea „amprentă“ a suprafeței este de forma  $3 \times 3$  iar nodurile  $U_{22}$  și  $U_{23}$  îndeplinesc condiția de colinearitate.

Din ecuația suprafeței Bézier  $P(u, v)$  obținem pentru  $m = n = 3$  următoarele ecuații parametrice ale suprafeței:

$$\begin{aligned} x &= 2u \\ y &= 1 + v \\ z &= 2u(1 - v) \left( u + \frac{3uv}{2} - 2v - 1 \right) \end{aligned}$$

În imaginile perspective din figurile 10.54, a, 10.54.b sunt reprezentate suprafețele din ultimele două aplicații, precum și secțiunea cu planul  $x = 1$  (figura 10.54.a).

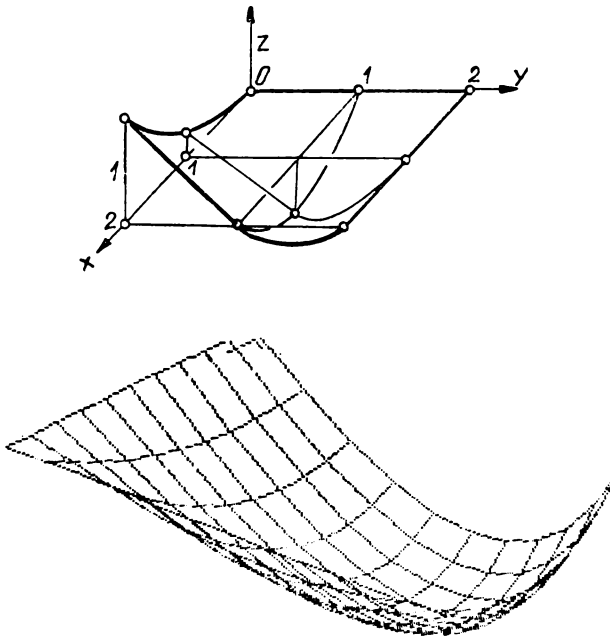


Fig. 10.54 a, b

## 10.17. Programe pentru construcția perspectivă a suprafețelor Bézier și Coons

### 10.17.1. Programul pentru construcția perspectivă a suprafețelor Bézier

Putem obține suprafața Bézier din curbele Bézier.  
Astfel fie  $\bar{r} = \bar{r}(u, v)$  ecuația suprafeței

$$\bar{r} = \bar{r}(u, v) = \sum_{i=0}^3 \sum_{j=0}^3 \bar{r}_{ij} \frac{3!}{(3-i)! i!} \frac{3!}{(3-j)! j!} u^i (1-u)^{3-i} v^j (1-v)^{3-j}$$

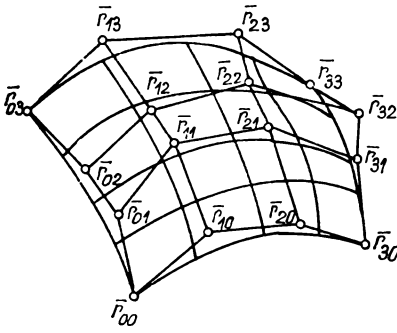


Fig. 10.55

unde  $\bar{r}_{ij}$  sînt vectorii de poziție ai vîrfurilor poliedrului director (figura 10.55).

Suprafața trece prin punctele (virfurile) de colț  $\bar{r}_{00}$ ,  $\bar{r}_{03}$ ,  $\bar{r}_{30}$ ,  $\bar{r}_{33}$ . Laturile (muchii) poliedrului director sînt respectiv tangente în punctele de colț curbilor frontieră.  $\bar{r}(u, 0)$ ,  $\bar{r}(u, 1)$ ,  $\bar{r}(0, v)$ ,  $\bar{r}(1, v)$ .

Sub forma matricială vom folosi ecuația suprafeței Bézier ca produs a următoarelor matrici:

$$\bar{r}(u, v) = \bar{U} \bar{M} \bar{B} \bar{M}^T \bar{V}$$

sau:

$$\bar{r} = \bar{r}(u, v) = [1, u, u^2, u^3] \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \cdot \begin{bmatrix} \bar{r}_{00} & \bar{r}_{01} & \bar{r}_{02} & \bar{r}_{03} \\ \bar{r}_{10} & \bar{r}_{11} & \bar{r}_{12} & \bar{r}_{13} \\ \bar{r}_{20} & \bar{r}_{21} & \bar{r}_{22} & \bar{r}_{23} \\ \bar{r}_{30} & \bar{r}_{31} & \bar{r}_{32} & \bar{r}_{33} \end{bmatrix}$$

$$\begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

După efectuarea înmulțirilor obținem tot sub forma matricială:

$$\bar{r} = \bar{r}(u, v) = [(1-u)^3, 3(1-u)^2 u, 3(1-u) u^2, u^3] \times$$

$$\times \begin{bmatrix} \bar{r}_{00} & \bar{r}_{01} & \bar{r}_{02} & \bar{r}_{03} \\ \bar{r}_{10} & \bar{r}_{11} & \bar{r}_{12} & \bar{r}_{13} \\ \bar{r}_{20} & \bar{r}_{21} & \bar{r}_{22} & \bar{r}_{23} \\ \bar{r}_{30} & \bar{r}_{31} & \bar{r}_{32} & \bar{r}_{33} \end{bmatrix} \cdot \begin{bmatrix} (1-v)^3 \\ 3(1-v)^2 v \\ 3(1-v) v^2 \\ v^3 \end{bmatrix}$$

Această formă a ecuației suprafeței Bézier o vom folosi în întocmirea programului, iar calculul coordonatelor punctelor se va face după ecuația dată în procedura **FUNCTION**.

Imaginile perspective ale suprafeței Bézier obținute sînt date în figurile 10.55 a și 10.55 b.

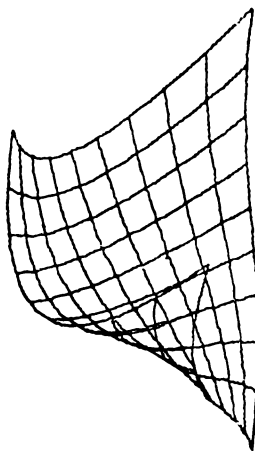


Fig.10.55 a

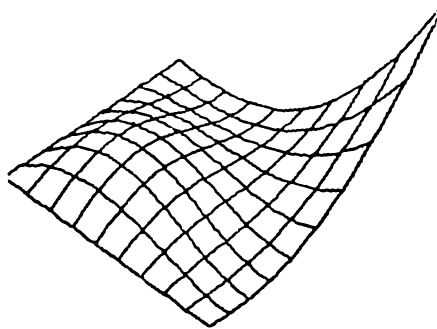


Fig. 10.55 b

### 10.17.2. Programul pentru construcția perspectivă a suprafețelor Coons

Considerăm cele patru curbe

$$\bar{a}_0(u), \bar{a}_1(u), \bar{b}_0(v), \bar{b}_1(v) \quad 0 \leq u \leq 1; 0 \leq v \leq 1$$

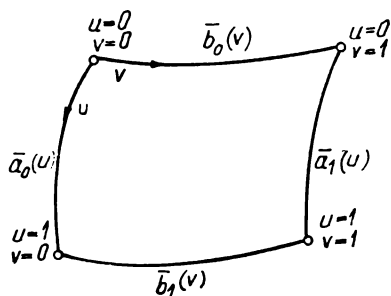


Fig. 10.56

care mărginesc suprafața (placa) din figura 10.56.

Curbele marginale  $\bar{a}_0$ ,  $\bar{a}_1$ ,  $\bar{b}_0$  și  $\bar{b}_1$  se obțin pentru valorile  $v = 0$ ,  $v = 1$ ,  $u = 0$  și  $u = 1$ .

Avem așadar:

$$\bar{r}(u, 0) = \bar{a}_0(u)$$

$$\bar{r}(u, 1) = \bar{a}_1(u)$$

$$\bar{r}(0, v) = \bar{b}_0(v)$$

$$\bar{r}(1, v) = \bar{b}_1(v)$$

Considerînd suprafața ca riglată avem pentru perechea de generatoare care se sprijină pe cîte două frontiere opuse:

$$\bar{r}_1(u, v) = (1 - v) \bar{r}(u, 0) + v \bar{r}(u, 1)$$

$$\bar{r}_2(u, v) = (1 - u) \bar{r}(0, v) + u \bar{r}(1, v)$$

așa cum putem remarca pe figurile 10.57 și 10.58.

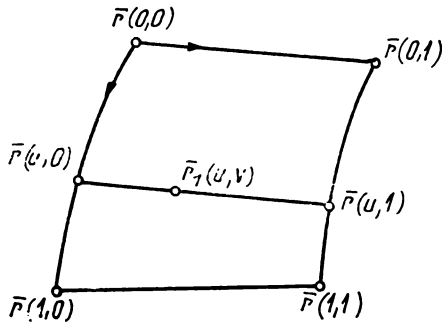


Fig. 10.57

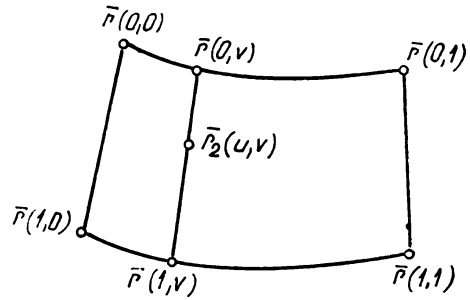


Fig. 10.58

Pentru a ajunge la suprafața Coons vom considera mai departe placa de suprafață definită astfel:

$$\bar{r}(u, v) = \bar{r}_1(u, v) + \bar{r}_2(u, v)$$

Substituind în această ecuație valorile anterioare și ținând seama de condițiile inițiale obținem o funcție de corecție  $\bar{r}_3(u, v)$ :

$$\begin{aligned} \bar{r}_3(u, v) = & (1 - u)(1 - v)\bar{r}(0, 0) + u(1 - v)\bar{r}(1, 0) + \\ & + (1 - u)v\bar{r}(0, 1) + uv\bar{r}(1, 1) \end{aligned}$$

Placa  $\bar{r}(u, v) = \bar{r}_1(u, v) + \bar{r}_3(u, v) + \bar{r}_2(u, v)$  îndeplinește condițiile și poate fi scrisă matricial sub forma:

$$\begin{aligned} \bar{r}(u, v) = & [1 - u, u] \cdot \begin{bmatrix} \bar{r}(0, v) \\ \bar{r}(1, v) \end{bmatrix} + [\bar{r}(u, 0), \bar{r}(u, 1)] \cdot \begin{bmatrix} 1 - v \\ v \end{bmatrix} - \\ & - [1 - u, u] \cdot \begin{bmatrix} \bar{r}(0, 0), \bar{r}(0, 1) \\ \bar{r}(1, 0), \bar{r}(1, 1) \end{bmatrix} \cdot \begin{bmatrix} 1 - v \\ v \end{bmatrix} \end{aligned}$$

Putem defini astfel o rețea specială de curbe parametrice pe suprafața  $\bar{r}(u, v)$  care se scrie:

$$\begin{aligned} \bar{r}(u, v) = & [F_1(u), F_2(u)] \begin{bmatrix} \bar{b}_0(v) \\ \bar{b}_1(v) \end{bmatrix} + [\bar{a}_0(u), a_1(u)] \cdot \begin{bmatrix} [F_1(v)] \\ [F_2(v)] \end{bmatrix} - \\ & - [F_1(u), F_2(u)] \cdot \begin{bmatrix} \bar{r}(0, 0), \bar{r}(0, 1) \\ \bar{r}(1, 0), \bar{r}(1, 1) \end{bmatrix} \begin{bmatrix} F_1(v) \\ F_2(v) \end{bmatrix} \end{aligned}$$

unde funcțiile  $F_1$  și  $F_2$  trebuie să îndeplinească condițiile:

$$\begin{aligned} F_1(0) = 1 & \quad F_1(1) = 0 & \quad (F_2 = 1 - F_1) \\ F_2(0) = 0 & \quad F_2(1) = 1 \end{aligned}$$

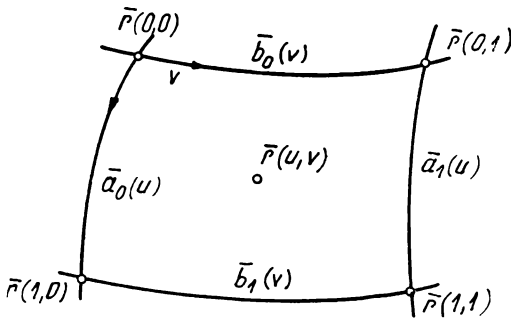


Fig. 10.59

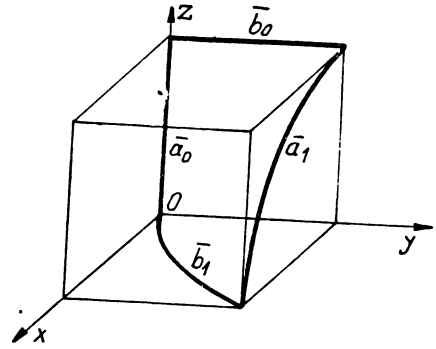


Fig. 10.60

Ecuția suprafeței poate fi scrisă și în felul următor într-o structură geometrică corespunzătoare figurii 10.59:

$$\bar{0} = \begin{bmatrix} F_1(u) \\ -1 \\ F_2(u) \end{bmatrix} \cdot \begin{bmatrix} \bar{r}(0,0) & \bar{b}_0(v) & \bar{r}(0,1) \\ \bar{a}_0(u) & \bar{r}(u,v) & \bar{a}_1(u) \\ \bar{r}(1,0) & \bar{b}_1(v) & \bar{r}(1,1) \end{bmatrix} \begin{bmatrix} F_1(v) \\ -1 \\ F_2(v) \end{bmatrix}$$

unde:

$$\begin{aligned} \bar{r}(0,0) &= \bar{a}_0(0) = \bar{b}_0(0) \\ \bar{r}(0,1) &= \bar{a}_1(0) = \bar{b}_0(1) \\ \bar{r}(1,0) &= \bar{a}_0(1) = \bar{b}_1(0) \\ \bar{r}(1,1) &= \bar{a}_1(1) = \bar{b}_1(1) \end{aligned}$$

Pentru întocmirea programului pentru construcția suprafețelor Coons vom folosi forma ecuației în care funcțiile parametrice  $F_1$  și  $F_2$  sînt definite independent în procedura de tip **FUNCTION**. Astfel alegem ca curbe marginale, deci ca limite ale suprafeței Coons două segmente de dreaptă  $\bar{a}_0$  și  $\bar{b}_0$ , arcul de parabolă  $\bar{b}_1$  și arcul de cosinusoidă  $\bar{a}_1$  din figura 10.60.

Alte detalii pot fi citite pe listarea programului, iar imaginile perspective ale suprafeței Coons obținute sînt date în figurile 10.61. a și 10.61.b.

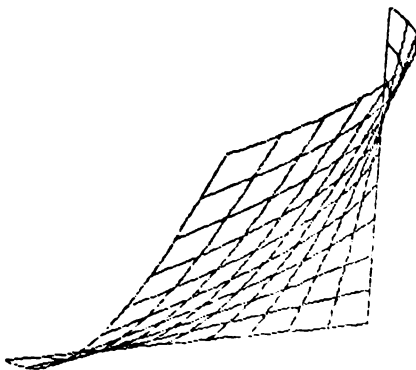


Fig. 10.61 a

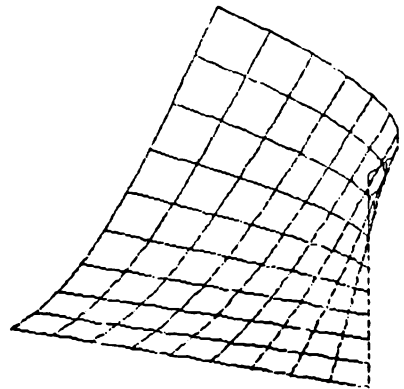


Fig. 10.61 b

```

PROGRAM BEZIER
C PROGRAM PENTRU GENERAREA MODELULUI SUPRAFETEI BEZIER
  DIMENSION K(4,4,3)
  REAL SEP1(3),SEP2(3),X,Y,Z
  DATA SEP1,SEP2/-5555.,-5555.,-5555.,-9999.,-9999.,-9999./
  DATA LU/Z/
  CALL ASSIGN(2,'BEZ.DAT')
C
C D1=DUMENIUL
C NBU=NUMARUL PUNCTELOR PE CURBA U
C NU=NUMARUL CURBELOR U
C NBV=NUMARUL PUNCTELOR PE CURBA V
C NV=NUMARUL CURBELOR V
  D1=10.
  NBU=30
  NU=10
  NBV=30
  NV=10
C
  DV1=1./FLOAT(NBU)
  DU1=1./FLOAT(NU-1)
  DV2=1./FLOAT(NV-1)
  DU2=1./FLOAT(NBV)
C CALCULUL VIKFURILOR POLIEDRULUI
  DO 50 I=1,4
  DO 50 J=1,4
  K(I,J,1)=D1*I
  K(I,J,2)=D1*J
  50 K(I,J,3)=0.
C VIKFURILE INTERACTIVE ALE POLIEDRULUI
  TYPE 10
  10 FORMAT (' ', 'DATI: N1, N2, Z: /')
  60 ACCEPT *, N1, N2, K3
  IF (N1.EQ.0) GO TO 90
  I=N1+1
  J=N2+1
  K(I,J,3)=K3
  GO TO 60
C
C CURBA U
  90 U=0.
  DO 120 M=1,NU
  V=0.
  DO 110 N=1,NBU
  IF (N.EQ.NBU) V=1.
  X=0.
  Y=0.
  Z=0.
  DO 100 K=1,4
  I=K-1
  DO 100 L=1,4
  J=L-1

```



```

      F=RNN3(I)*RNN3(J)*FT(U,I)*FT(V,J)
      X=X+F*K(K,L,1)
      Y=Y+F*K(K,L,2)
100  Z=Z+F*K(K,L,3)
      WRITE (LU) X,Y,Z
110  V=V+UV1
      WRITE (LU) SEP1
120  U=U+UU1
C   CAZUL IN CARE ESTE GENERATA CURBA V
      V=0.
      UU 140 M=1,MV
      U=0.
      UU 130 N=1,NBV
      IF (N.EQ.NBV) U=1.
      X=0.

      Y=0.
      Z=0.
      UU 150 K=1,4
      I=K-1
      UU 150 L=1,4
      J=L-1
      F=RNN3(I)*RNN3(J)*FT(U,I)*FT(V,J)
      X=X+F*K(K,L,1)
      Y=Y+F*K(K,L,2)
150  Z=Z+F*K(K,L,3)
      WRITE (LU) X,Y,Z
130  U=U+UU2
      WRITE (LU) SEP1
140  V=V+UV2
      WRITE (LU) SEP2
      CALL CLOSE(2)
      STOP
      END
      FUNCTION FT(T,I)
      FUNCTIA FT=(T**I)*(1.-T)**(3-I)
C   PENTRU I=0,1,2,3 A 0<=T<=1
      IF (I.EQ.0) GO TO 100
      IF (I.EQ.3) GO TO 130
      FT=0.
      IF (T.EQ.0. .OR. T.EQ.1.) RETURN
      FT=(T**I)*(1.-T)**(3-I)
      RETURN
100  FT=0
      IF (T.EQ.1.) RETURN
      FT=(1.-T)**3
      RETURN
130  FT=0
      IF (T.EQ.0.) RETURN
      FT=T**3
      RETURN
      END
C
      REAL FUNCTION RNN3(K)
C   FUNCTIA PENTRU CALCULUL 3 SPRE K
      RNN3=1
      IF (K.EQ.0 .OR. K.EQ.3) RETURN
      RNN3=3
      RETURN
      END

```

```

PROGRAM COONS
C PROGRAM PENTRU CONSTRUCTIA SUPRAFETEI COONS
C DEFINITE PRIN CURBELE MARGINALE
  DIMENSION R(3),SEP1(3),SEP2(3)
  DATA SEP1,SEP2/-5555.,-5555.,-5555.,-9999.,-9999.,-9999.
C FUNCTIA PARAMETRICA
C
  F1(T)=1.-T
  F2(T)=T
C
C NBU=NUMARUL PUNCTELOR PE CURBA U
C NBV=NUMARUL PUNCTELOR PE CURBA V
C NU=NUMARUL CURBELOR U
C NV=NUMARUL CURBELOR V
  CALL ASSIGN(2,'COONS.DAT')
  LU=2
  NBU=30
  NBV=30
  NU=10
  NV=10
  UV1=1./ (FLOAT (NBU))
  UV2=1./ (FLOAT (NU-1))
  UV3=1./ (FLOAT (NV-1))
  UV4=1./ (FLOAT (NBV))
C
C GENERAREA CURBELOR U
  U=0.
  DO 300 I=1,NU
    V=0.
    DO 200 J=1,NBU
      IF (J.EQ.NBU) V=1.
      DO 100 K=1,3
        R(K)=F1(U)*BU(K,V)+F2(U)*B1(K,V)+F1(V)*AU(K,U)+F2(V)*A1(K,U)
        * -F1(V)*F1(U)*AU(K,U.)-F1(V)*F2(U)*AU(K,1.)
        * -F2(V)*F1(U)*A1(K,U.)-F2(V)*F2(U)*A1(K,1.)
      100 CONTINUE
      WRITE (LU) R
    200 V=V+UV1
    WRITE (LU) SEP1
  300 U=U+UV2
C
C GENERAREA CURBELOR V
  V=0.
  DO 1300 I=1,NV
    U=0.
    DO 1200 J=1,NBU
      IF (J.EQ.NBU) U=1.
      DO 1100 K=1,3
        R(K)=F1(U)*BU(K,V)+F2(U)*B1(K,V)
        * +F1(V)*AU(K,U)+F2(V)*A1(K,U)
        * -F1(V)*F1(U)*AU(K,U.)-F1(V)*F2(U)*AU(K,1.)
        * -F2(V)*F1(U)*A1(K,U.)-F2(V)*F2(U)*A1(K,1.)
      1100 CONTINUE
      WRITE (LU) R
    1200 U=U+UV3
    WRITE (LU) SEP1

```

```
1500 V=V+DV2
      WRITE (L0)SEF2
      CALL CLOSE(L0)
      STOP
      END
      FUNCTION AU(1,T)
      GO TO (10,20,30), 1
10  AU=0.
      RETURN
20  AU=0.
      RETURN

30  AU=1.-T
      RETURN
      END

C
      FUNCTION A1(1,T)
      GO TO (10,20,30),1
10  A1=T
      RETURN
20  A1=1.
      RETURN
30  A1=COS(T*3.14/2.)
      RETURN
      END

C
      FUNCTION BU(1,T)
      GO TO (10,20,30),1
10  BU=0.
      RETURN
20  BU=T
      RETURN
30  BU=1.
      RETURN
      END

C
      FUNCTION B1(1,T)
      GO TO (10,20,30),1
10  B1=T
      RETURN
20  B1=T*T
      RETURN
30  B1=0.
      RETURN
      END
```



# Anexe (II)

## ANEXA C

### C.1. Programul PLANAR pentru desenarea modulară automată a planurilor de arhitectură și construcții

#### C.1.1. Introducere

Există posibilitatea elaborării unor algoritmi și programe pentru desenaarea modulară automată a elementelor care compun fie planuri de arhitectură și construcții civile sau industriale fie piese din domeniul construcțiilor de mașini.

Pe baza acestor elemente pot fi create numeroase alte module de desen în funcție de necesitățile impuse de o anumită reprezentare.

Ordonarea și legarea modulelor se face prin transformări geometrice uzuale iar epurele pot fi obținute în dialog interactiv fie prin desenarea automată la ploter, fie prin vizualizare pe displayul grafic.

Programul PLANAR este proiectat tocmai în vederea rezolvării acestor probleme și este constituit dintr-un set de subrutine care corespund diferitelor elemente constructive ale obiectului ce trebuie să fie reprezentat.

Acest set de subrutine poate fi extins pentru realizarea unor desene cu un grad de complexitate foarte mare astfel încât pe parcurs chiar obiectul deja reprezentat poate intra ca o subrutină într-un proces de elaborare al unui ansamblu complex.

Programul PLANAR preia datele de intrare din fișierul disc PLAN. DAT și produce un fișier de ieșire PLAN. PLO ce se va plota ulterior în mod OFF-LINE la masa de desen.

În forma sa inițială programul PLAISAR cuprinde subrutinele PERETE, UȘA, GEAM și COTA deocamdată minim pentru desenaarea modulară automată a planurilor de arhitectură și construcții sau a pieselor din domeniul construcțiilor de mașini.

#### C.1.2. Subrutina PERETE

Forma apelului este:

**CALL PERETE (INDC, DL1, DL2, DL, B, IPOZ, XI, YI, II)**

iar reprezentarea elementului (modulului) PERETE este dată în figura 1—C.

Semnificația parametrilor de apel este următoarea:

- *INDC* este un comutator cu trei valori care precizează ce punct este considerat punct inițial pentru perete

*INDC* = 0 punct nou; *XI*, *YI* specifică acest punct

*INDC* = 1 punctul *X3*

*INDC* = 2 punctul *X4*

- *DL1* este lungimea în centimetri a segmentului *X1 — X4*

- *DL2* este lungimea în centimetri a segmentului *X2 — X3*

- *DL* este lungimea în centimetri a segmentului *X1' — X2*

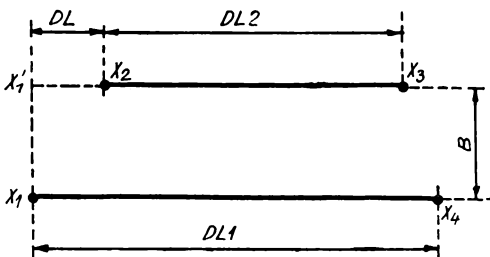


Fig. 1 — C

- $B$  este lățimea (grosimea) în centimetri a peretelui
- $IPOZ$  este un comutator cu patru valori care precizează direcția în care se desenează peretele.
  - $IPOZ = 0$  poziția de bază la  $0^\circ$
  - $IPOZ = 1$  perete la  $90^\circ$
  - $IPOZ = 2$  perete la  $180^\circ$
  - $IPOZ = 3$  perete la  $270^\circ$
- $XI$  este abscisa punctului inițial pentru cazul  $INDC = 0$
- $YI$  este ordonata punctului inițial pentru cazul  $INDC = 0$
- $II$  este un comutator cu patru valori care precizează închiderea peretelui
  - $II = 0$  perete deschis
  - $II = 1$  perete închis dreapta
  - $II = 2$  perete închis stânga
  - $II = 3$  perete închis la ambele capete.

### C.1.3. Subrutina USA

Forma apelului este:

**CALL USA (DL1, IPOZ, IT, IP, ID, XI, YI, B)**

Reprezentările elementului (modulului) USA sînt date în figurile 2—C a și 2—C b.

Subrutina USA calculează la ieșire valorile  $X3'$  și  $X4'$  (figura 2 — C a) pentru a facilita continuarea desenului cu un alt element.

Semnificația parametrilor de apel este următoarea:

- $DL1$  reprezintă lățimea ușii în centimetri
- $IPOZ$  are aceeași semnificație ca în subrutina PERETE
- $IT$  este indicator cu două valori care precizează tipul ușii:
  - $IT = 0$  tipul ușii este pe stînga pentru  $IPOZ = 0$  (figura 3 — C a).
  - $IT = 1$  tipul ușii este pe dreapta pentru  $IPOZ = 0$  (figura 3 — C b).

În cazul ușii duble  $IT$  nu are semnificația de mai sus ci:

- $IT = 0$  ușa este orientată stînga față de direcția în care se execută desenul pentru  $IPOZ = 0$  (fig. 4 — C a)
- $IT = 1$  ușa este orientată dreapta față de direcția în care se execută desenul pentru  $IPOZ = 0$  (fig. 4 — C b)

- $IP$  este un indicator cu două valori care codifică poziția punctului în juru căruia se rotește ușa față de punctul inițial de desen al ușii.

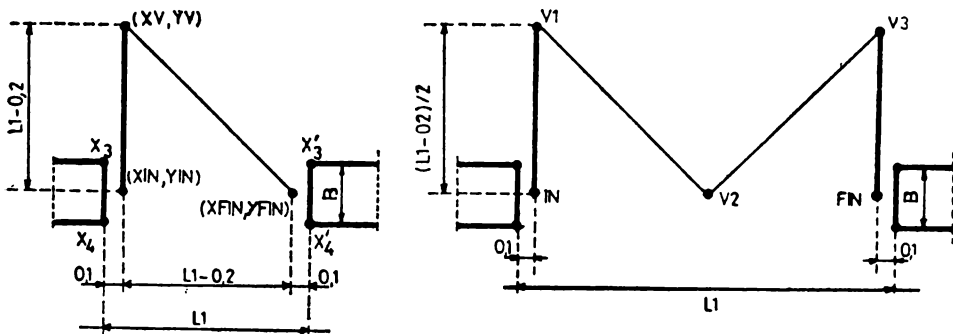


Fig. 2 — C

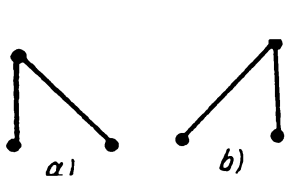


Fig. 3 - C

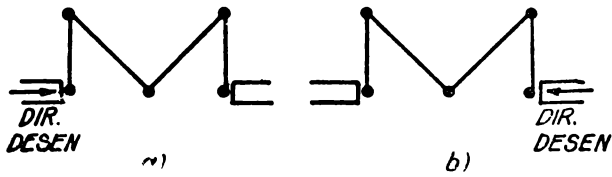


Fig. 4 - C

$IP = 0$  punct de rotație apropiat de punctul inițial, pentru  $IPOZ = 0$  (figura 5 - C a)

$IP = 1$  punct de rotație depărtat de punctul inițial, pentru  $IPOZ = 0$  (figura 5 - C a)

- $ID$  este un indicator cu două valori care specifică tipul ușii  
 $ID = 0$  ușa simplă (fig. 5 - C b)  
 $ID = 1$  ușa dublă (fig. 5 - C b)
- $XI$  este abscisa punctului inițial dacă ușa nu se desenează în continuarea altui element
- $YI$  este ordonata punctului inițial dacă ușa nu se desenează în continuarea altui element
- $B$  este lățimea în centimetri a elementului care urmează ușii în desen. Ea este necesară pentru omogenitatea programului.

#### C.1.4. Subrutina GEAM

Forma apelului este:

**CALL GEAM (INDC, DL, IPOZ, B, XI, YI)**

Figura 6-C ilustrează reprezentarea generală a elementului (modulului) **GEAM**.

Subrutina **GEAM** este concepută ca succesiunea a două apeluri ale subrutinei **PERETE**:

- un apel pentru construirea unui perete închis la ambele capete de direcție  $IPOZ$
- un apel pentru construirea unui perete deschis de direcție  $MOD_4 (IPOZ + 2)$

Semnificația parametrilor de apel este următoarea:

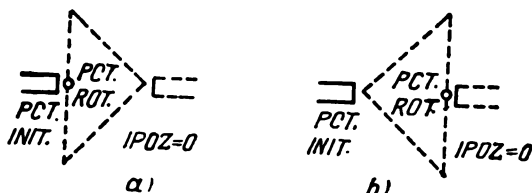


Fig. 5 - C

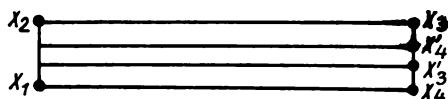


Fig. 6 - C

- $INDC$  are aceeași semnificație ca în subrutina **PERETE**
- $DL$  este lungimea geamului (ferestrei) în centimetri
- $IPOZ$  are aceeași semnificație ca în subrutina **PERETE**
- $B$  este lățimea ferestrei în centimetri
- $XI$  este abscisa punctului inițial dacă fereastra este primul element al secvenței de desenat
- $YI$  este ordonata punctului inițial dacă fereastra este primul element al secvenței de desenat

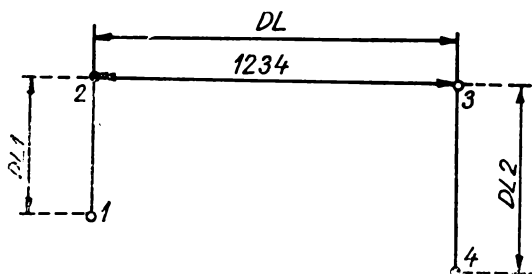


Fig. 7 - C

## C.1.5. Subrutina COTA

Forma apelului este:

**CALL COTA (XI, YI, DL, DL1, DL2, VAL, FI, K)**

Subrutina COTA poate fi folosită în cadrul oricărui program de desen care necesită cotări ale elementelor reprezentate. De asemenea ea poate avea diverse aspecte legate de specificul cotării, de standardizare sau de fantezie

în situații adecvate. Figura 7 - C ilustrează un mod de reprezentare generală a acestui element (modul) COTA. La fel figurile 7-C a și 7-C b.

Semnificația parametrilor de apel este următoarea:

- *XI, YI* sînt coordonatele punctului inițial al cotei
- *DL* este lungimea săgeții în centimetri
- *DL1* este lungimea segmentului (1 - 2) în centimetri
- *DL2* este lungimea segmentului (3 - 4) în centimetri
- *VAL* este valoare reală de patru cifre care reprezintă valoarea numerică a cotei
- *FI* este unghiul de rotație al cotei în grade, în jurul punctului inițial (*XI, YI*)
- *K* este un indicator cu două valori:

Dacă  $K = 0$  pentru  $VAL = 1234$ . pe desen apare 1234

Dacă  $K = 1$  pentru  $VAL = 1234$ . pe desen apare FI 1234

Toate subrutinele prezentate în cadrul acestui program conțin segmente orientate, valorile lor putînd fi și negative deci există echivalența inversării orientării segmentelor.

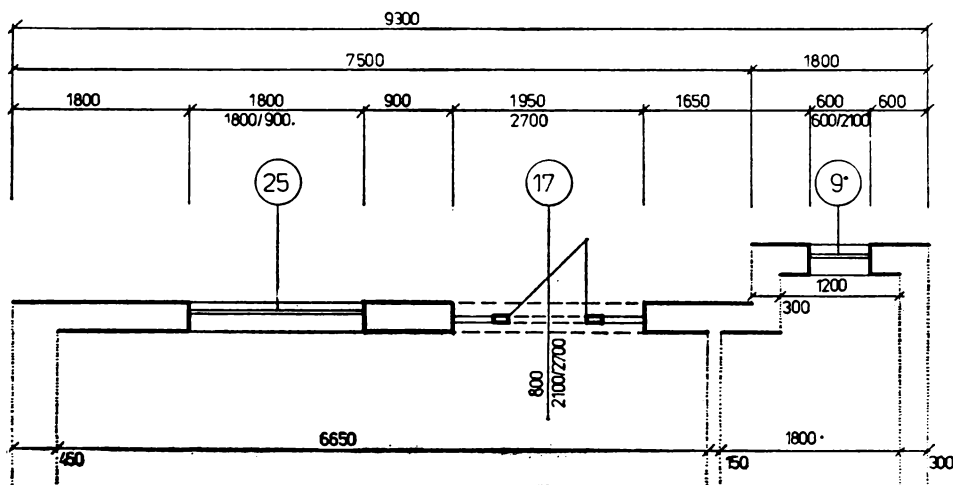


Fig. 7 - C a



**PROGRAM PRINCIPAL PLANAR**

```

        LOGICAL*1 FIS(15)
        TYPE 190
190    FORMAT(' FISIER DE INTRARE : ',3)
        ACCEPT 2,VRCAR,FIS
        CALL ASSIGN(2,FIS,VRCAR)
        CALL FDBSET(2,'R')
        TYPE 1
1      FORMAT(' FISIER DE IESIRE : ',3)
        ACCEPT 2,VRCAR,FIS
2      FORMAT(2,15A1)
        CALL ASSIGN(4,FIS,VRCAR)
        CALL FDBSET(4,'NEA')
        CALL PLJTS(0,0,0,4)
11     READ(2,4,END=21) ITIP
4      FORMAT(I2)
        GOTO (5,6,7,8,9),ITIP
5      READ(2,10) INDC,D_1,D_2,DL,3,IPJZ,11,XI,YI
10     FORMAT(I2,4F7.2,2I2,2F7.2)
        CALL PERETE(INDC,D_1,DL2,D_2,8,IPDZ,XI,YI,11)
        GOTO 11
6      READ(2,13) D_1,IPJZ,IT,IP,IO,XI,YI,B
13     FORMAT(F7.2,4I2,3F7.2)
        CALL JSA(D_1,IPJZ,IT,IP,IO,XI,YI,B)
        GOTO 11
7      READ(2,15) INDC,D_1,IPDZ,B,XI,YI
15     FORMAT(I2,F7.2,I2,3F7.2)
        CALL SEAM(INDC,D_1,IPJZ,B,XI,YI)
        GOTO 11
8      READ(2,18) XI,YI,D_1,DL1,DL2,VAL,FI,<
18     FORMAT(7F7.2,I2)
        CALL COFA(XI,YI,D_1,DL1,DL2,VAL,FI,<)
        GOTO 11
9      READ(2,20) A1,B1,C1
20     FORMAT(3F7.2)
        CALL PCAR(A1,B1,C1)
        GOTO 11
21     CALL PLDT(0,0,0,999)
        CALL CLJSE(4)
        CALL CLJSE(2)
        STOP
        END
    
```

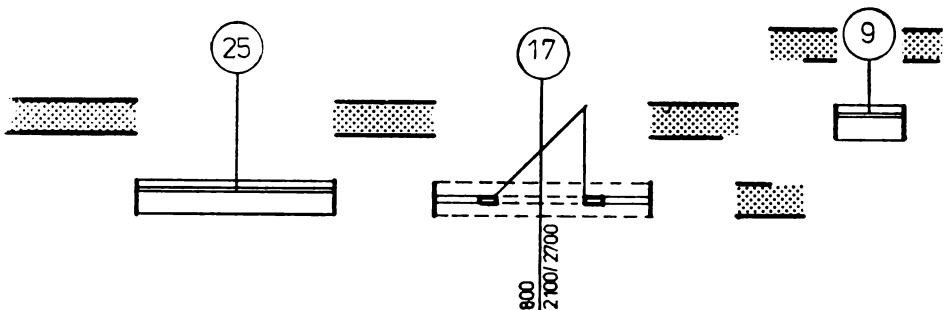


Fig. 7 - C b

## Subprogramul PERETE

```

SUBROUTINE PERETE (INDC,DL1,DL2,DL,8,IPJZ,KI,YI,I!)
COMMON /PL44/X3,X4,Y3,Y4
IF (INDC.EQ.0) GOTO 1
X4=XI
Y4=YI
1 IF (INDC.EQ.1) GOTO 2
X1=X4
Y1=Y4
GOTO 3
2 X1=X3
Y1=Y3
3 S=1.
IF (IPJZ.GT.1) S=-1.
IP =IPJZ+1
GOTO (10,20,10,20) ,IP
10 X2=X1+D_*S
Y2=Y1+B*S
X3=X2+D_*S
Y3=Y2
X4=X1+D_*S
Y4=Y1
GOTO 4
20 X2=X1-B*S
Y2=Y1+D_*S
X3=X2
Y3=Y2+D_*S
X4=X1
Y4=Y1+D_*S
4 I1=1
I2=1
CALL PLJT(X1,Y1,2)
CALL PLJT(X4,Y4,1)
IF (MOD(I1,2).EQ.0) I2=2
CALL PLJT(X3,Y3,I2)
CALL PLJT(X2,Y2,1)
IF (I1.LT.2) I1=2
CALL PLJT(X1,Y1,I1)
RETURN
END

```

## Subprogramul GEAM

```

SUBROUTINE GEAM(INDC,DL,IPJZ,8,XI,YI)
COMMON /PL44/X3,X4,Y3,Y4
CALL PERETE(INDC,DL,DL,0.,3,IPJZ,XI,YI,3)
X3N=X3
X4N=X4
Y3N=Y3
Y4N=Y4
S=1.
IF (IPJZ.GT.1) S=-1.
GOTO(10,20,10,20),IPJZ+1
10 Y4=Y3-B/3.*S
GOTO 30
20 X4=X3+B/3.*S
30 CALL PERETE(2,DL,DL,0.,B/3.,MOD(IPJZ+2,4),0.,0.,0)
X3=X3N
X4=X4N
Y3=Y3N
Y4=Y4N
RETURN
END

```

## SUBPROGRAMUL UȘA

```

SUBROUTINE UȘA(DL1,IPOZ,IT,IP,IO,XI,YI,B)
COMMON /P.AN/X3,X4,Y3,Y4
IPOZ1=IPOZ+1
I=(XI.EQ.0)..AND.(YI.EQ.0.) GOTO 1
S=1.
GOTO (10,20,30,40),IPOZ1
10 X3=XI
Y3=YI+B/2.*S
X4=XI
Y4=YI-B/2.*S
GOTO 1
20 X3=XI-B/2.*S
Y3=YI
X4=XI+B/2.*S
Y4=YI
GOTO 1
30 S=-1.
GOTO 10
40 S=-1.
GOTO 20
1 S=1.
GOTO (4,5,6,65),IPOZ1
4 XIN=X3+D.1*S
YIN=(Y3+Y4)/2.
XFIN=XIN+(DL1-D.2)*S
YFIN=YIN
X3=X3+D.1*S
X4=X3
IF(IO.EQ.1) GOTO 9
XV=XIN
IF(IP.EQ.1) XV=XFIN
YV=YIN-(DL1-D.2)*S
IF(IP.EQ.IT) YV=YIN+(DL1-D.2)*S
GOTO 3
5 XIN=(X3+X4)/2.
YIN=Y3+D.1*S
XFIN=XIN
YFIN=YIN+(DL1-D.2)*S
Y3=Y3+D.1*S
Y4=Y3
IF(IO.EQ.1) GOTO 11
YV=YIN
IF(IP.EQ.1) YV=YFIN
XV=XIN+(DL1-D.2)*S
IF(IP.EQ.IT) XV=XIN-(DL1-D.2)*S
GOTO 3
6 S=-1.
GOTO 4
55 S=-1.
GOTO 5
6 CALL P.LJT(XIN,YIN,2)
CALL P.LJT(XV,YV,1)
CALL P.LJT(XFIN,YFIN,1)
RETURN

```

```

9      XV2=(XIN+XFIN)/2.
      YV2=YIN
      YV1=YIN+(DL1-0.2)*S/2.
      XV1=XIN
      IF(IT.EQ.1) YV1=YIN-(DL1-0.2)*S/2.
      XV3=XFIN
      YV3=YV1
      GOTO 12
11     XV2=XIN
      YV2=(YIN+YFIN)/2.
      XV1=XIN-(DL1-0.2)*S/2.
      IF(IT.EQ.1) XV1=XIN+(DL1-0.2)*S/2.
      YV1=YIN
      YV3=YFIN
      XV3=XV1
12     CALL PLOT(XIN,YIN,2)
      CALL PLOT(XV1,YV1,1)
      CALL PLOT(XV2,YV2,1)
      CALL PLOT(XV3,YV3,1)
      CALL PLOT(XFIN,YFIN,1)
      RETURN
      END

```

### Subprogramul COTA

```

SUBROUTINE COTA(XI,YI,DL,DL1,DL2,VAL,FI,K)
COMMON /BCAR/3,AL,ALEFA,BETA/BD/IXO,IYO,SOX,SOY,IJ,MOD,CR,LF,F,S
LOGICAL*1 T(3),CR,LF
DIMENSION X(11),Y(11)
DATA T,'P','F','I',' ',',',3.1415926535/
FI1=FI*PI/180.
X(1)=0.
X(2)=0.
X(3)=0.3
X(4)=.3
X(5)=0.
X(6)=DL
X(7)=DL-.3
X(8)=X(7)
X(9)=DL
X(10)=DL
Y(1)=-DL
Y(2)=0.
Y(3)=.03
Y(4)=-.05
Y(5)=.0
Y(6)=0.
Y(7)=.05
Y(8)=-.05
Y(9)=0.
Y(10)=-.02
LVT=20
IF(K.EQ.1) LVT=30
DL=LVT*3
X(11)=(DL-DL1)/2.
Y(11)=2.*41
DO 1 I=1,11
CALL ROTATE(X(I),Y(I),FI1)
X(I)=X(I)+X1
Y(I)=Y(I)+Y1
CALL PDATA(X,Y,1,10)
IF(K.EQ.1)CALL TEXT(T,3,X(11),Y(11),FI)
CALL NUMBER(VAL,4,0,X(11),Y(11),FI)
RETURN
END

```

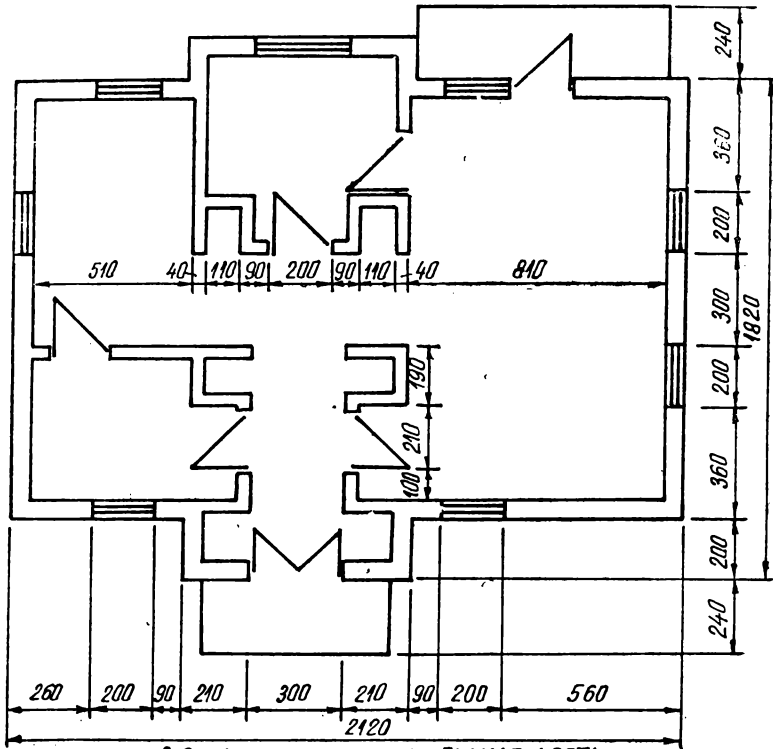
```

SUBROUTINE ROTATE(X,Y,FI)
X1=X
X=X1*COS(FI)-Y*SIN(FI)
Y=X*SIN(FI)+Y*COS(FI)
RETURN
END

```

### Subprogramul ROTATE

1



*\* Combinarea programelor PLANAR și COTA  
în reprezentarea planurilor de  
arhitectură și construcții*

Fig. 8 - C

În figura 8 - C este prezentat un plan de arhitectură realizat automat la ploter prin compunerea modulelor programului PLANAR.

Desenul poate fi completat mai departe cu orice alte detalii necesare asupra cărora nu mai insistăm.

## C.2. Program pentru trasarea conturilor care îmbracă cu grosimi variabile un graf dat prin noduri și legăturile dintre ele\*)

### 1. Introducere

Lucrarea conține algoritmul și programul pentru trasarea conturilor care îmbracă cu grosimi variabile grafuri date prin noduri și legăturile dintre ele. Algoritmul admite maximum 200 de puncte și maximum 200 segmente.

Variabilele de intrare sînt constituite din numărul de segmente ale grafului, din vectorii cu coordonatele pe  $x$  și pe  $y$  ale nodurilor, din vectorii care conțin referințele la primul punct al unui segment și la punctul final

\*) Au colaborat ing. Anca Dumitrescu și arh. Mihai Popescu

al segmentului și din vectorii grosimilor pe stînga și respectiv pe dreapta ale fiecărui segment.

Este prezentată căutarea de *descendenți* pentru *tatăl* de indice, rotirea intrărilor, căutarea referințelor stînga sau dreapta și secvențele de desen pentru succesiunea conturilor.

Sînt utilizate ca subrutine procedurile pentru calcularea unghiului dintre 2 segmente, pentru interschimbarea a doi întregi și pentru calcularea intersecției dintre două drepte.

Lucrarea prezintă mare interes prin aplicațiile sale în special în proiectarea de construcții și arhitectură precum și în alte domenii ale matematicii.

## 2. Analiza concretă a unui graf

Să considerăm graful definit aleator prin coordonatele nodurilor și prin legăturile dintre noduri din exemplul alăturat (Fig. 9.1 — C1).

Alegem, de exemplu, nodul 1 ca rădăcină (*tată*). Graful se reorganizează sub o formă de parcurgere „*most left*” adică spre cea mai din stînga ramură cu restricția că dacă un nod a apărut anterior drept „*tată*” atunci, el se va considera la următoarea apariție drept „*frunză*”.

Raționamentul este următorul:

Pornim din nodul 1. Avem fiii 2 și 4. Considerăm fiul 2 (cel mai din stînga). Pornim din nodul 2. Avem fiii 3 și 5. Considerăm fiul 3 (cel mai din stînga). Urmează nodul 11 care este „*frunză*”. Ne întoarcem la ultimul „*tată*” care mai are alți fii, de exemplu, 2. Alegem următorul fiu 5 apoi nodurile 6, 7 și 12 care este „*frunză*”. Ne întoarcem la nodul 7 cu fiul 10 și urmează apoi nodurile 9, 4 și 1. Dar nodul 1 a fost inițial „*tată*” și acum apare ca fiu, deci trebuie să fie considerat „*frunză*”. Urmează înapoi nodurile 4, apoi 5 și nodul 6 care este „*tată*” și ultimul segment 6—8 unde 8 este „*frunză*”. În felul acesta toate segmentele au fost parcurse. Succesiunea segmentelor este notată pe prima coloană din figura 1, adică graful modificat. Se remarcă cum nodul (1) (*tatăl*) din prima linie, se regăsește ca fiu în linia 11-a, situație în care apare ca „*frunză*” așa cum s-a arătat mai sus.

Pentru trasarea conturilor vom folosi tot graful modificat din prima coloană a figurii 9.1 — C2. Astfel să raționăm trasarea primului contur și în paralel să întocmim graful din figura 9.2 — C1.

Fiecare segment al grafului trebuie să apară parcurs o dată pe stînga și o dată pe dreapta. Este de reținut că prioritatea de alegere este fixă.

Astfel alegem segmentul 7—2 ca prim segment din graf și-l parcurgem pe stînga (linia 1 din coloana 1 din figura 9.1 — C2). Trecem acest segment 7—2 în linia 1 din coloana 2, din figura 1. Căutăm care segment apare cu referință stînga nesatisfăcută pornind de la fiul 2. Găsim 2—3 în linia a doua, apoi pentru fiul 3 găsim referința stînga în linia a treia. În figura 2 am notat nodurile 1—2—3—11 cu referința stînga. Pentru nodul 11 nu mai avem referința stînga deci vom lua referința dreapta și considerăm segmentul parcurs invers adică 11—3. Aceste elemente introduse în coloana 2 din figura 9.1 — C2 ne dau după terminarea întregului raționament trasarea primului contur, după ce s-a ajuns la rădăcina 1, ultimul segment trasat fiind 4—7 (coloana a doua din fig. 9.1 — C2). Primul contur trasat este deschis.

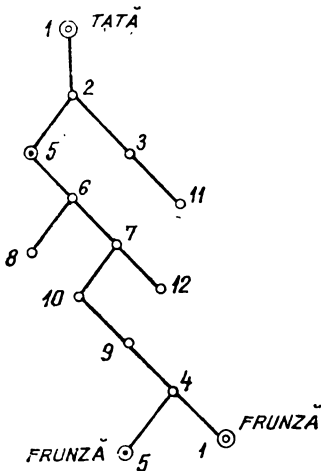
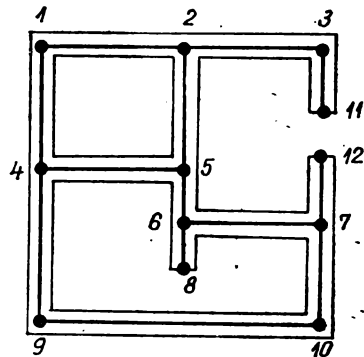
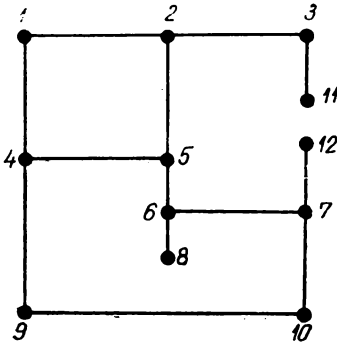


Fig. 9.1 - C1

GRAFUL MODIFICAT	TRASAREA PRIMULUI CONTUR	TRASAREA PTR. AL 2-LEA CONTUR
① 2	1 2	2 1
2 3	2 3	1 4
3 11	3 11	4 5
2 5	11 3	5 2
5 6	3 2	TRASAREA
6 7	2 5	PENTRU
7 12	5 6	AL 3-LEA
7 10	6 7	CONTUR
10 9	7 12	6 5
9 4	12 7	5 4
4 ①	7 10	4 9
4 5	10 9	9 10
6 8	9 4	10 7
	4 1	7 6
		6 8
		8 6

Fig. 9.1 - C2

Pentru trasarea celui de al doilea contur se ia prima referință liberă pe stînga sau pe dreapta. Alegem astfel 2—7 (toate referințele pe stînga din coloana 1, figura 9.1 - C2 au fost epuizate). Astfel nodul 1 din prima linie nu are pe stînga nici o referință și atunci alegem referința pe dreapta obținînd astfel 7—4 în linia unsprezece. Urmează trasarea segmentelor 4—5 apoi 5—2 închizîndu-se cu nodul 2 care este rădăcină și nu are referințe pe stînga. Al doilea contur trasat este închis (fig. 9.1 - C2 coloana 3). Pentru trasarea celui de al treilea contur se alege segmentul 6—5 după același raționament și se obține tot un contur închis (figura 9.1, coloana a treia).

După cum am arătat în introducere programul este structurat după acest algoritm și folosește pentru grosimile variabile pe stînga sau pe dreapta procedurile pentru calcularea unghiului dintre două segmente, pentru interschimbarea dintre doi întregi și pentru calculul punctelor de intersecție dintre două drepte. Alte amănunte privind programul pot fi citite pe listarea programului.

## PROGRAMUL ARBORE

```

C      PROGRAM PENTRU TRASAREA CONTURURILOR CARE IMBRACA CU GROSIMI
C      VARIABILE UN GRAF DAT PRIN NODURI SI LEGATURILE DINTRE ELE.
C      VARIABILE DE INTRARE
C      IFIN=NUMARUL DE SEGMENTE ALE GRAFULUI
C      XP=VECTORUL CU COORDONATELE PE X ALE NODURILOR
C      YP=VECTORUL CU COORDONATELE PE Y ALE NODURILOR
C      I1=VECTORUL CONTININD REFERINTA LA PRIMUL PUNCT AL UNUI SEGMENT
C      I2=VECTORUL CONTININD REFERINTA LA PUNCTUL FINAL AL SEGMENTULUI
C      GRS=VECTORUL GROSIMILOR PE STINGA ALE FIECARUI SEGMENT
C      GRD=VECTORUL GROSIMILOR PE DREAPTA ALE FIECARUI SEGMENT
C
C      DECLARATII
C
C      * DIMENSION XP(200),YP(200),GRD(200),GRS(200),UNGHI(200),I1(200)
C      * ,I2(200),GRARB(400)
C      * INTEGER IGRD(400),IGRS(400),IUNGHI(400),ARB1(400),ARB2(400)
C      * LOGICAL*1 FISI(15),FISE(15)
C      * EQUIVALENCE (ICRD(1),GRD(1)),(IGRS(1),GRS(1)),(IUNGHI(1),
C      * UNGHI(1))
C
C      CITIREA DATELOR DE INTRARE
C
C      TYPE 1000
1000   FORMAT(' NUME FISIER DE DATE : ',*)
      ACCEPT 1001,NRI,FISI
1001   FORMAT(0,15A1)
      TYPE 1000
1002   FORMAT(' NUME FISIER DE DESEN : ',*)
      ACCEPT 1001,NRE,FISE
      CALL ASSIGN(1,FISI,NRI)
      CALL ASSIGN(2,FISE,NRE)
      READ (1,2000)IFIN
      IF (IFIN.GT.200) GOTO 2006
2000   FORMAT(15)
      READ (1,2001)(I1(I),I2(I),GRS(I),GRD(I),I=1,IFIN)
2001   FORMAT(2I5,2F12.5)
      READ (1,2003,END=2004)(XP(I),YP(I),I=1,200)
2003   FORMAT(2F12.5)
      TYPE 2005
2005   FORMAT(' *** ALGORITMUL ADMITE MAXIMUM 200 DE PUNCTE ***')
      STOP
2006   TYPE 2007
2007   FORMAT(' *** ALGORITMUL ADMITE MAXIMUM 200 SEGMENTE ***')
      STOP
C      INITIALIZAREA UNOR VARIABILE GENERALE
2004   CALL INI(2)
      NRARB=1
      IARBI=1
      IRAD=I1(1)
      IDESC=I2(1)
      ICRT=1
      ICAUT=2
      IJOS=1
C
C      CAUTARE DE DESCENDENTI PENTRU TOTAL DE INDICE IDESC
C

```



```

6      NRDESC=0
      UMIN=2*3.141527
      IMIN=0
      DO 1 I=ICAUT,IFIN
        IF (IDESC.EQ.I1(I))GOTO 2
        IF (IDESC.EQ.I2(I))GOTO 3
        GOTO 1
3      CALL INTERS(I2(I),I1(I))
      CALL INTERS(IGRD(2*I),IGRS(2*I))
      CALL INTERS(IGRD(2*I-1),IGRS(2*I-1))
2      U=UNGHI(I)
      IF (IJOS.EQ.0)GOTO 5
      CALL ANGLE(XP,YP,I1(ICRT),IDESC,I2(I),U)
      UNGHI(I)=U
5      NRDESC=NRDESC+1
      IF (U.GE.UMIN)GOTO 1
      IMIN=I
      UMIN=U
1      CONTINUE
      IF (NRDESC.NE.0)GOTO 4
9      IF (ICRT.EQ.IARBI)GOTO 8
      ICRT=ICRT-1
      IF (IJOS.EQ.1)ICRT=ICRT+1
      IJOS=0
      IDESC=I1(ICRT)
      GOTO 6
4      IF (IMIN.EQ.ICAUT)GOTO 7
C
C      ROTIREA INTRARILOR IMIN SI ICAUT
C
      CALL INTERS(I1(IMIN),I1(ICAUT))
      CALL INTERS(I2(IMIN),I2(ICAUT))
      CALL INTERS(IGRD(2*IMIN-1),IGRD(2*ICAUT-1))
      CALL INTERS(IGRD(2*IMIN),IGRD(2*ICAUT))
      CALL INTERS(IGRS(2*IMIN-1),IGRS(2*ICAUT-1))
      CALL INTERS(IGRS(2*IMIN),IGRS(2*ICAUT))
      CALL INTERS(IUNGHI(2*IMIN-1),IUNGHI(2*ICAUT-1))
      CALL INTERS(IUNGHI(2*IMIN),IUNGHI(2*ICAUT))
7      ICRT=ICAUT
      ICAUT=ICAUT+1
      IDESC=I2(ICRT)
      DO 10 I=IARBI,ICRT-1
        IF (IDESC.EQ.I1(I))GOTO 9
10     CONTINUE
      IJOS=1
      IF (ICAUT.LE.IFIN)GOTO 6
8     TYPE 3000,(I1(K),I2(K),GRS(K),GRD(K),K=IARBI,ICAUT-1)
3000  FORMAT(2I5,2F12.5)
      IC=IARBI
21     IT=I2(IC)
      IR=I1(IC)
      ITC=1
      ARB1(ITC)=IR
      ARB2(ITC)=IT
      GRARB(ITC)=GRS(IC)
      I2(IC)=-I2(IC)

```

```

C
C      CAUTARE REFERINTA PENTRU IT
C
15     TYPE 3010,IARBI,ICAUT-1,IR,IT,IC
3010   FORMAT(S15)
      DO 11 I=IC+1,ICAUT-1
      IF (IT-I1(I))11,12,11
11     CONTINUE
      IF (IT.EQ.IR) GOTO 16
      DO 13 I=IARBI,ICAUT-1
      IF (IT+I2(I))13,14,13
13     CONTINUE
C
C      S-A GASIT REFERINTA STINGA
C
12     ITC=ITC+1
      ARB1(ITC)=I1(I)
      ARB2(ITC)=I2(I)
      GRARB(ITC)=GRS(I)
      IC=I
      IT=I2(IC)
      I2(IC)=-I2(IC)
      GOTO 15
C
C      S-A GASIT REFERINTA DREAPTA
C
14     ITC=ITC+1
      ARB1(ITC)=-I2(I)
      ARB2(ITC)=I1(I)
      GRARB(ITC)=GRD(I)
      IC=I
      IT=I1(IC)
      I1(IC)=0
      I2(IC)=0
      GOTO 15
C
C      SECVENTA DE DESEN
C
16     TYPE 4000,(ARB1(K),ARB2(K),GRARB(K),K=1,ITC)
4000   FORMAT(2I5,F12.5)
      TYPE 4001,(I1(K),I2(K),GRS(K),GRD(K),K=1,IFIN)
4001   FORMAT(2I5,2F12.5)
      IP1=ARB1(ITC)
      Y1=YP(IP1)
      X1=XP(IP1)
      IP2=ARB2(ITC)
      Y2=YP(IP2)
      X2=XP(IP2)
      A2=Y2-Y1
      B2=X1-X2
      DELT=SQRT(A2*A2+B2*B2)
      A2=A2/DELT
      B2=B2/DELT
      C2=-B2*Y1-A2*X1+GRARB(ITC)
      IORIG=0
      DO 22 I=1,ITC

```

```

A1=A2
B1=B2
C1=C2
IP1=ARB1(I)
IP2=ARB2(I)
X1=XP(IP1)
Y1=YP(IP1)
X2=XP(IP2)
Y2=YP(IP2)
A2=Y2-Y1
B2=X1-X2
DELT=SQRT(A2*A2+B2*B2)
A2=A2/DELT
B2=B2/DELT
C2=-B2*Y1-A2*X1+GRARB(I)
IF(A2*B1-A1*B2.NE.0.)GOTO 23
IF(C1.EQ.C2)GOTO 22
ADRP=-B2
BDRP=A2
CDRP=B2*X1-A2*Y1
CALL DRDR(A1,B1,C1,ADRP,BDRP,CDRP,X,Y)
IF(IORIG.NE.0)GOTO 24
XORIG=X
YORIG=Y
IORIG=1
CALL PLOT(X,Y,0)
GOTO 25
24 CALL PLOT(X,Y,1)
25 CALL DRDR(A2,B2,C2,ADRP,BDRP,CDRP,X,Y)
CALL PLOT(X,Y,1)
GOTO 22
23 CALL DRDR(A1,B1,C1,A2,B2,C2,X,Y)
IF(IORIG.NE.0)GOTO 26
XORIG=X
YORIG=Y
IORIG=1
CALL PLOT(X,Y,0)
GOTO 22
26 CALL PLOT(X,Y,1)
22 CONTINUE
CALL PLOT(XORIG,YORIG,1)
C
C URMATORUL CONTUR
C
DO 17 I=IARBI,ICAUT-1
IF(I2(I))20,17,18
17 CONTINUE
IF(ICAUT.GT.IFIN)GOTO 27
IARBI=ICAUT
IRAD=I1(IARBI)
IDESC=I2(IARBI)
ICRT=IARBI
ICAUT=ICRT+1
IJOS=1
NRARB=NRARB+1
GOTO 6

```

```

18      IC=I
        GOTO 21
20      IC=I
        IT=I1(IC,
        IR=-I2(IC)
        ITC=1
        ARB1(ITC)=IR
        ARB2(ITC)=IT
        GRARB(ITC)=GRD(IC)
        I1(I)=0
        I2(I)=0
        GOTO 15
27      CALL EOF
        STOP

```

```

C
C

```

```

END

```

### SUBPROGRAMUL ANGLE

```

C
C
C
C

```

PROCEDURA PENTRU CALCULAREA UNGHIULUI DINTRE 2 SEGMENTE

```

SUBROUTINE ANGLE(X,Y,I1,I2,I3,U)
DIMENSION X(1),Y(1)
A2=(X(I2)-X(I1))**2+(Y(I2)-Y(I1))**2
B2=(X(I3)-X(I2))**2+(Y(I3)-Y(I2))**2
C2=(X(I3)-X(I1))**2+(Y(I3)-Y(I1))**2
U=ACOS((A2+B2-C2)/(2.*SQRT(A2*B2)))
PV=(X(I2)-X(I1))*(Y(I3)-Y(I2))-(X(I3)-X(I2))*(Y(I2)-Y(I1))
IF(PV.LT.0.)U=2*3.14157-U
RETURN
END

```

### SUBPROGRAMUL INTERS

```

C
C
C

```

PROCEDURA DE INTERSCHIMBARE A DOI INTREGI

```

SUBROUTINE INTERS(I2,I1)
I1=IEOR(I1,I2)
I2=IEOR(I1,I2)
I1=IEOR(I1,I2)
RETURN
END

```

### SUBPROGRAMUL DRDR

```

C
C
C

```

PROCEDURA DE CALCULARE A INTERSECTIEI A DOUA DREPTE

```

SUBROUTINE DRDR(A1,B1,C1,A2,B2,C2,X,Y)
PROD=A2*B1-B2*A1
X=(B2*C1-B1*C2)/PROD
Y=(A1*C2-A2*C1)/PROD
RETURN
END

```

## SUBPROGRAMUL CONTUR

```

INTEGER TATA,SEG(2,1000),NREF(200),DATS(2,1000),MARCS(500),
* MARCP(200),TABF(15),ARB(100)
DIMENSION XP(200),YP(200),GROS(1000),TABU(15),GROS1(1000)
DATA PI/3.141529/
C CITIRE DATE DIN FISIER SI CALCUL NUMAR DE REFERINTE ***
CALL ASSIGN(1,'PLAN.DAT')
CALL FDBSET(1,'R')
CALL ASSIGN(3,'PLAN.FLO')
CALL FDBSET(3,'N')
READ (1,1)NRPCT,NRSEG
1 FORMAT(2I5)
READ(1,2)(XP(I),YP(I),I=1,NRPCT)
2 FORMAT(2F10.3)
DO 4 I=1,NRSEG
READ(1,3) SEG(1,I),SEG(2,I),GROS(I)
3 FORMAT(2I5,F10.3)
NREF(SEG(1,I))=NREF(SEG(1,I))+1
NREF(SEG(2,I))=NREF(SEG(2,I))+1
4 CONTINUE
C ORGANIZAREA SEGMENTELOR IN ARBORI ***
INDF=1
INDC=1
100 DO 5 I=1,NRPCT
IF(NREF(I).NE.1)GOTO 5
IF(MARCP(I).EQ.0) GOTO 6
5 CONTINUE
6 IF(I.EQ.NRPCT+1)GOTO 7
MARCP(I)=1
DO 8 J=1,NRSEG
IF(MARCS(J).EQ.1)GOTO 8
IF(SEG(1,J).EQ.I) GOTO 9
IF(SEG(2,J).EQ.I) GOTO 10
8 CONTINUE
10 CALL ROT(SEG,J)
9 DATS(1,INDF)=SEG(1,J)
DATS(2,INDF)=SEG(2,J)
GROS1(INDF)=GROS(J)
MARCS(J)=1
INDF=INDF+1
13 IF(INDF.EQ.INDC) GOTO 101
TATA=DATS(1,INDC)
NRC=DATS(2,INDC)
MARCP(NRC)=1
IF(NREF(NRC).NE.1)GOTO 12
INDC=INDC+1
GOTO 13
101 ARB(1+NRARB)=INDF-1
NRARB=NRARB+1
GOTO 100
C GASIRE FII AI UNUI PUNCT ***
12 ITF=1
DO 14 K=1,NRSEG
IF(MARCS(K).EQ.1)GOTO 14
IF(SEG(1,K).EQ.NRC)GOTO 15
IF(SEG(2,K).NE.NRC)GOTO 14
CALL ROT(SEG,K)

```

```

15     TABF(ITF)=K
      ITF=ITF+1
      MARCS(K)=1
      IF(ITF.EQ.NREF(NRC))GOTO 16
14     CONTINUE
16     ITF=ITF-1
C     ORGANIZARE TABF SI TABU IN FUNCTIE DE UNGHIURI ***
      IF(ITF.EQ.1)GOTO 18
      A2=(XP(NRC)-XP(TATA))**2+(YP(NRC)-YP(TATA))**2
      DO 19 I=1,ITF
        I1=SEG(2,TABF(I))
        B2=(XP(I1)-XP(NRC))**2+(YP(I1)-YP(NRC))**2
        C2=(XP(I1)-XP(TATA))**2+(YP(I1)-YP(TATA))**2
        COSFI=(A2+B2-C2)/(2*SQRT(A2*B2))
        FI=ATAN2(SQRT(1-COSFI*COSFI),COSFI)
        IF(FI.LT.0)FI=FI+PI
        FI=FI-FI
        PV=(XP(NRC)-XP(TATA))*(YP(I1)-YP(NRC))-(XP(I1)-XP(NRC))*
        * (YP(NRC)-YP(TATA))
        IF(PV.LT.0)FI=-FI
        TABU(I)=FI
19     CONTINUE
        DO 20 I=1,ITF-1
          DO 20 J=I+1,ITF
            IF(TABU(I).GT.TABU(J))GOTO 20
            XX=TABU(I)
            TABU(I)=TABU(J)
            TABU(J)=XX
            K=TABF(I)
            TABF(I)=TABF(J)
            TABF(J)=K
20     CONTINUE
18     DO 17 I=1,ITF
          DATS(1,INDP)=SEG(1,TABF(I))
          DATS(2,INDP)=SEG(2,TABF(I))
          GROS1(INDP)=GROS(TABF(I))
17     INDP=INDP+1
          INDC=INDC+1
          GOTO 13
7     DO 22 I=1,NRSEG
22     MARCS(I)=0
          INDI=0
          CALL PLOTS(0.,0.,3)
          DO 21 I=1,NRARB
            INDT=INDI+1
            INDA=1
            SEG(1,1)=DATS(1,INDT)
            SEG(2,1)=DATS(2,INDT)
            GROS(1)=GROS1(INDT)
24     MARCS(INDT)=1
            JSUC=DATS(2,INDT)
27     IF(NREF(JSUC).EQ.1)GOTO 25
            NREF(JSUC)=NREF(JSUC)-1
            DO 23 J=INDT,ARB(I)
              IF(MARCS(J).EQ.1)GOTO 23
              IF(DATS(1,J).NE.JSUC)GOTO 23

```

```

INDA=INDA+1
SEG(1,INDA)=DATS(1,J)
SEG(2,INDA)=DATS(2,J)
GROS(INDA)=GROS1(J)
INDT=J
23 GOTO 24
CONTINUE
25 DO 26 J=INDI+1,ARB(I)
IF(MARCS(J).EQ.0)GOTO 26
IF(DATS(2,J).NE.JSUC)GOTO 26
INDA=INDA+1
SEG(1,INDA)=JSUC
SEG(2,INDA)=DATS(1,J)
GROS(INDA)=GROS1(J)
JSUC=DATS(1,J)
INDT=J
26 GOTO 27
CONTINUE
I1=SEG(1,1)
I2=SEG(2,1)
XA=XP(I1)
YA=YP(I1)
XB=XP(I2)
YB=YP(I2)
D=GROS(1)/2.
A=YA-YB
B=XB-XA
XNORM=SQRT(A*A+B*B)
CP=(XA*YB-XB*YA)/XNORM-D
A=A/XNORM
B=B/XNORM
YPC=(-B*CP-A*(B*XA-A*YA))
XPC=(B*(B*XA-A*YA)-A*CP)
XI=XPC
YI=YPC
XI1=XA+(XA-XPC)
YI1=YA+(YA-YPC)
CALL PLOT(XI,YI,2)
DO 35 J=2,INDA
I3=SEG(2,J)
XC=XP(I3)
YC=YP(I3)
D=GROS(J)/2.
A1=YB-YC
B1=XC-XB
XNORM=SQRT(A1*A1+B1*B1)
CP1=(XB*YC-XC*YB)/XNORM-D
A1=A1/XNORM
B1=B1/XNORM
IF(I3.EQ.I1)GOTO 36
IF(A1*B.NE.A*B1)GOTO 37
C SEGMENTE IN PRELUNGIRE
XB=XC
YB=YC
I1=I2
I2=I3

```

```

      GOTO 35
C   TRATARE NORMALA A SEGMENTELOR
37   XPC=(-CP1*B+B1*CP)/(A1*B-A*B1)
      YPC=(CP1*A-CP*A1)/(A1*B-A*B1)
      CALL PLOT(XPC,YPC,1)
38   I1=I2
      I2=I3
      A=A1
      B=B1
      CP=CP1
      XB=XC
      YB=YC
      GOTO 35
C   SEGMENTUL ESTE FRUNZA
36   YPC=(-B1*CP1-A1*(B1*XB-A1*YB))
      XPC=(B1*(B1*XB-A1*YB)-A1*CP1)
      XPC1=XB+XB-XPC
      YPC1=YB+YB-YPC
      CALL PLOT(XPC1,YPC1,1)
      CALL PLOT(XPC,YPC,1)
      GOTO 38
35   CONTINUE
      CALL PLOT(XI1,YI1,1)
      CALL PLOT(XI,YI,1)
      INDI=ARB(I)
21   CONTINUE
      CALL PLOT(0.,0.,3)
      STOP
      END

SUBROUTINE ROT(SEG,J)
INTEGER SEG(2,500)
I1=SEG(2,J)
SEG(2,J)=SEG(1,J)
SEG(1,J)=I1
RETURN
END

SUBROUTINE PLOTS(X,Y,NR)
LOGICAL*1 F,ORD1,ORD2,V0(2),V1(2),A0(2),A1(2)
EQUIVALENCE (IV0,V0),(IV1,V1),(IA0,A0),(IA1,A1)
DATA F,ORD1,ORD2/'52','21','22'/
DATA IV0,IV1,IA0,IA1/'310','144,2,1'/
WRITE (3,1)F,ORD1,V0(2),V0(1),V1(2),V1(1)
FORM (6A1)
WRITE (3,1)F,ORD2,A0(2),A0(1),A1(2),A1(1)
RETURN
END

SUBROUTINE PLOT(X,Y,JPEN)
LOGICAL*1 F,ORD1,ORD2,ORD3,X0(4),Y0(4)
EQUIVALENCE (IX,X0),(IY,Y0)
DATA F,ORD1,ORD2,ORD3/'52,3,7','32'/
INTEGER*4 IX,IY
IX=JFIX(X*1000.)
IY=JFIX(Y*1000.)
GOTO(1,2,4),JPEN
1  WRITE(3,5)F,ORD2,X0(4),X0(3),X0(2),X0(1),Y0(4),Y0(3),Y0(2),Y0(1)
   RETURN
2  WRITE(3,5)F,ORD1,X0(4),X0(3),X0(2),X0(1),Y0(4),Y0(3),Y0(2),Y0(1)
   RETURN
5  FORMAT(10A1)
4  WRITE(3,6)F,ORD3
6  FORMAT(2A1)
   RETURN
   END

```

### 3. Aplicații

În fig. 10.1 — C2 și 10.2 — C2 sînt prezentate graful unui plan de arhitectură și „îmbrăcarea” automată a pereților cu lăsarea liberă a golurilor pentru uși și ferestre, în funcție de datele de intrare. Aplicația conține coordonatele pentru 55 de puncte și 46 segmente. Această aplicație conține numai trasarea unor contururi deschise.

Aplicația fig. 10.3 — C2 conținînd de asenienea reprezentarea unui plan de arhitectură mult mai sofisticat, exprimă posibilitățile reale ale programului prezentat în această lucrare.



## DATE DE INTRARE

70		
1,3,2,,2.	71,72,0,,3.	272.,242.
3,4,2,,2.	72,73,0,,3.	192.,125.
4,5,2,,2.	73,74,0,,3.	192.,170.
5,6,2,,2.	75,76,1,,1.	192.3,173.
6,7,2,,2.	77,74,1,,1.	193.,178.
7,8,2,,2.	77,78,1,,1.	194.,182.
8,9,2,,2.	78,79,1,,1.	198.,197.
9,10,0,,2.	79,80,1,,1.	199.,201.
10,11,2,,2.	80,81,1,,1.	200.5,204.6
11,12,2,,2.	81,82,1,,1.	203.,207.6
12,13,4,,2.	82,83,1,,1.	205.,209.
13,14,4,,2.	83,84,1,,1.	208.,210.
14,15,2,,2.	24,29,2,,2.	220.,210.
15,16,4,,2.	26,29,2,,2.	210.,188.
16,17,4,,2.		220.,188.
17,56,2,,2.		220.,198.
56,57,2,,2.		220.,162.
17,18,2,,2.	312.,164.	210.,162.
18,19,2,,2.	314.,160.	270.,186.
19,20,0,,2.	317.6,148.	270.,198.
20,21,2,,2.	318.8,144.	268.,162.
21,22,2,,2.	319.4,140.	280.,162.
22,23,0,,2.	319.7,136.	374.5,76.
21,23,2,,2.	320.,130.	354.,142.
23,24,2,,2.	320.,108.	334.,157.
24,25,2,,2.	297.,108.	348.,162.
25,26,2,,2.	289.,108.	306.,294.
24,27,2,,2.	270.,108.	200.,260.
26,28,0,,2.	270.,123.	0.,0.
30,31,0,,2.	268.,123.	34.,134.
31,32,0,,2.	260.,123.	30.,140.
33,32,2,,2.	230.,123.	10.,120.
32,34,2,,2.	222.,123.	30.,100.
34,35,2,,2.	220.,123.	50.,120.
35,40,2,,2.	220.,108.	42.,127.
41,42,1,,1.	199.,108.	350.,40.
42,43,1,,1.	191.,108.	351.2,34.8
43,44,1,,1.	138.,108.	355.,31.
44,45,1,,1.	134.,150.	360.,30.
45,46,1,,1.	138.,150.	365.,31.
46,47,1,,1.	138.,152.	368.8,34.8
47,48,1,,1.	138.,162.	370.,40.
48,49,1,,1.	148.,162.	368.8,45.2
49,50,1,,1.	148.,170.	365.,49.
50,51,1,,1.	175.,162.	360.,50.
51,52,1,,1.	148.,152.	
53,54,2,,2.	173.,171.	
54,55,2,,2.	173.,188.	
58,59,2,,2.	148.,188.	
60,61,2,,2.	148.,180.	
62,63,2,,2.	138.,188.	
64,65,2,,2.	138.,242.	
65,66,2,,2.	0.,0.	
66,67,2,,2.	0.,0.	
69,70,0,,3.	0.,0.	
70,71,0,,3.	0.,0.	

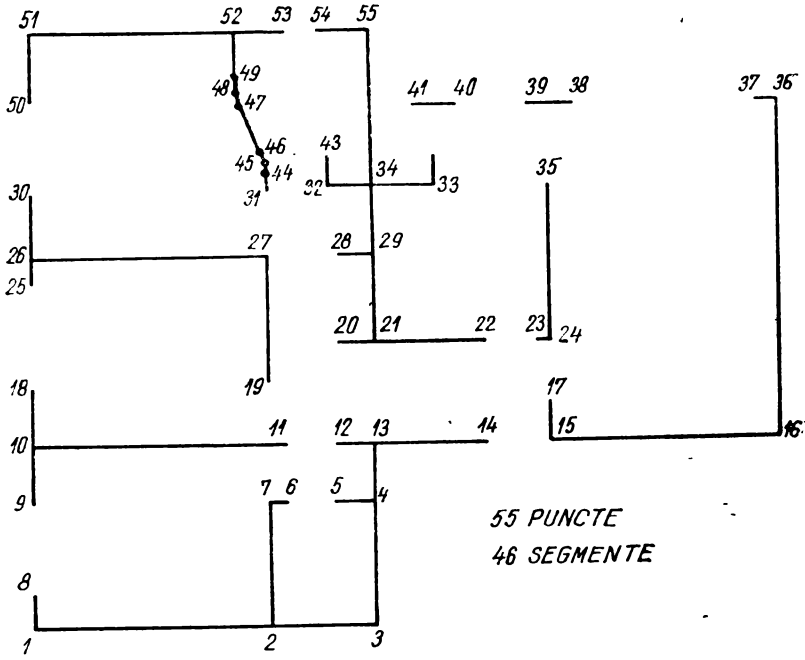


Fig. 10.1 - C2

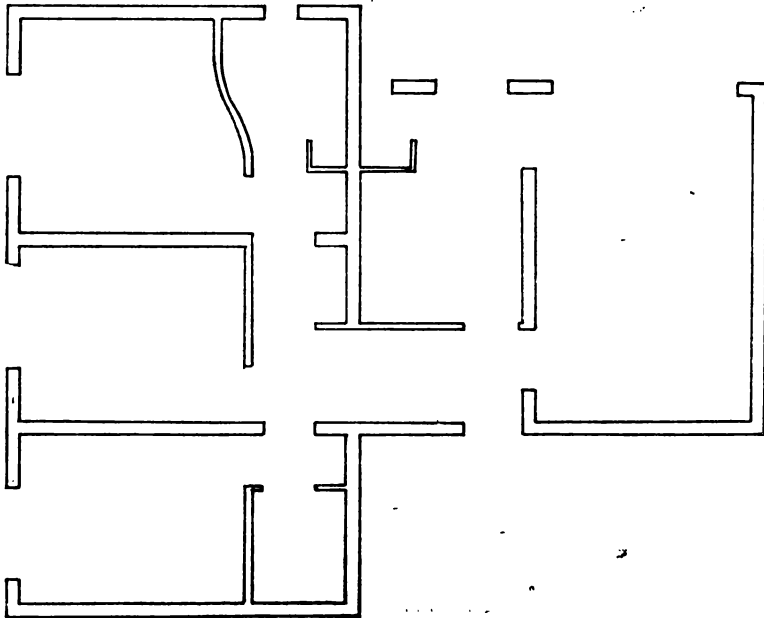


Fig. 10.2 - C2

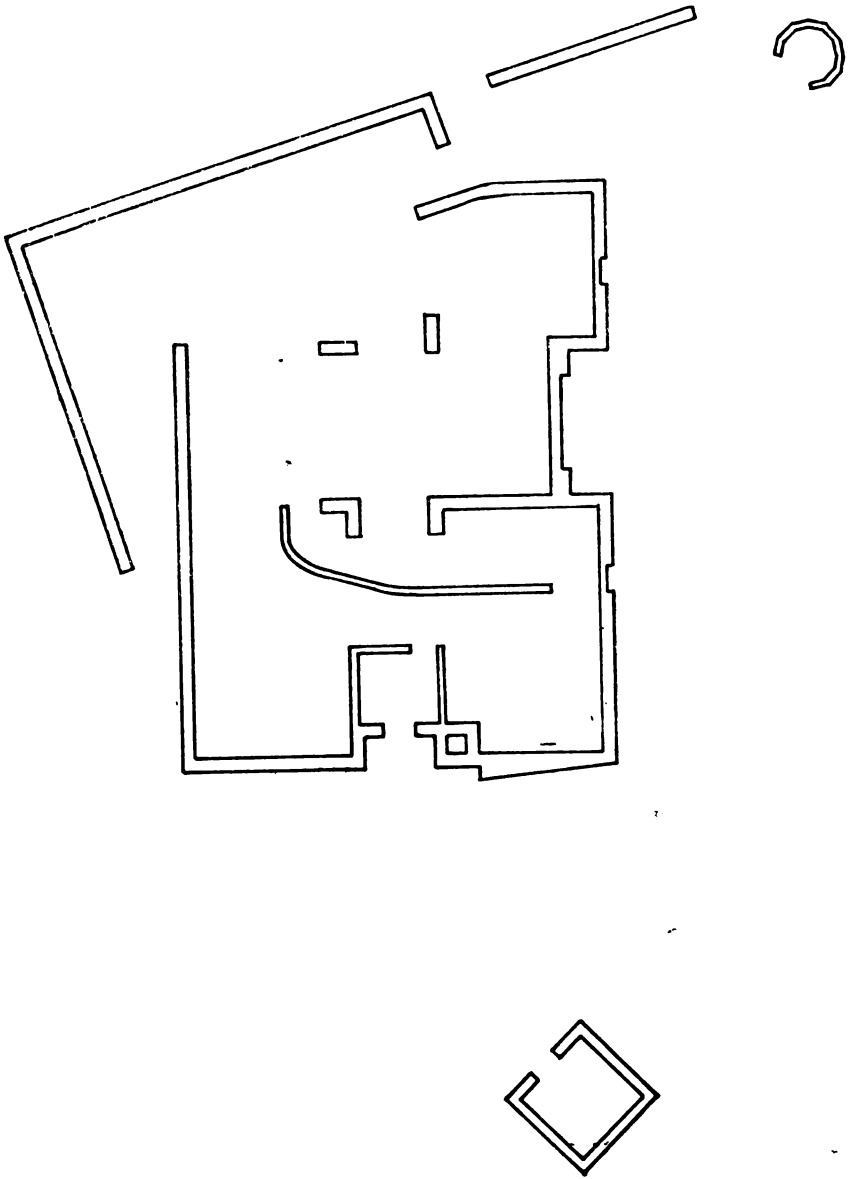


Fig. 10.3 - C2

## ANEXA D

### Programul PIESA

Programul „PIESA“ este un exemplu de desenare a unei piese din domeniul construcțiilor de mașini (un ansamblu de linii și cote) prin care am urmărit să arătăm posibilitățile realizării unui sistem integrat de proiectare a unor ansamble mai complexe pe baza unor module primare variabile.

Pentru piesa în cauză s-a căutat un set complet de dimensiuni care să permită o definiție neambiguă.

Executând diverse calcule care să ducă la obținerea tuturor dimensiunilor și coordonatelor piesei se realizează desenul și cotarea piesei.

Variabilitatea piesei este dată prin parametrizarea în funcție de setul de dimensiuni ales.

Din desenele prezentate în continuare se observă că s-a obținut același tip de piesă dar de diverse forme dând diverse dimensiuni parametrilor.

În exemplul ales setul de dimensiuni complet este alcătuit din cele 7 elemente cotate la care se adaugă un unghi  $U_1$  (necotat) și un coeficient de scădere  $S$  (figura 1 — D).

Datele numerice pot fi incluse într-o tabelă de forma următoare:

A	P	Q	D	V	W	T	Z	U1	S
100.	80.	65.	10.	36.	30.	12.	15.	0.27	1.
200.	150.	100.	8.	100.	80.	20.	20.	0.15	0.5
200.	150.	100.	8.	100.	80.	20.	20.	0.15	2.0

PROGRAMUL PIESA  
Dimensiuni variabile

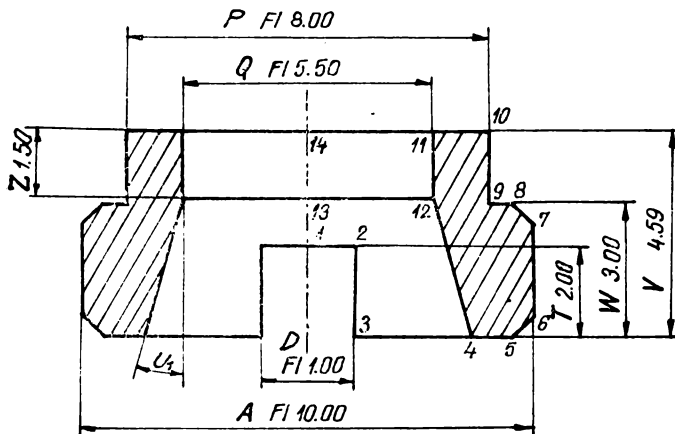


Fig. 1 — D

Programul poate fi foarte ușor modificat pentru a deveni o subrutină.

Dacă în plus se adaugă două valori de bazare a desenului și eventual un unghi de rotație se poate obține piesa în diverse poziții și unghiuri de rotație astfel încât să poată fi plasată într-un ansamblu.

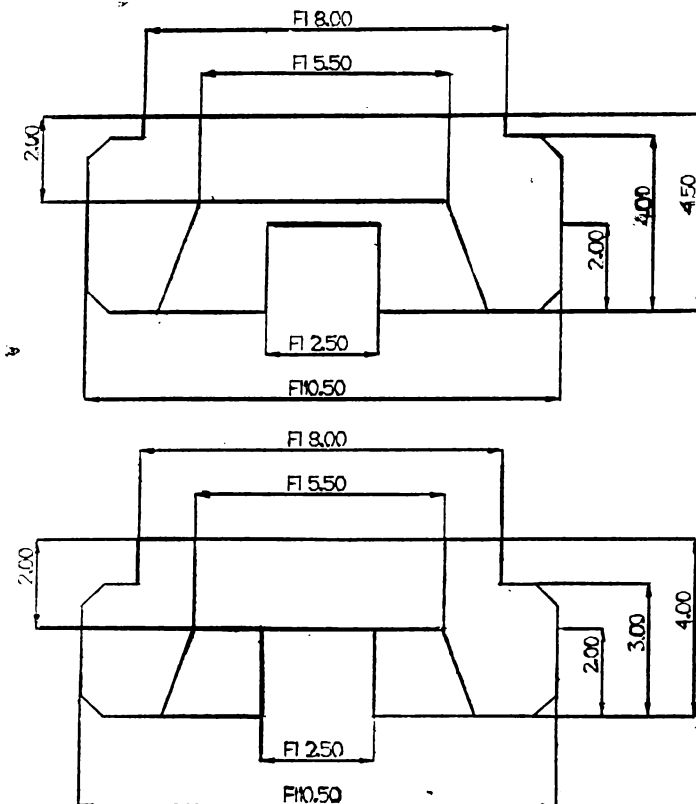
Presupunând în continuare existența unui număr de asemenea subrutine pentru piese primare (de exemplu: *CORP ROBINET*, *GHIDAJ*, *AX VENTIL*, *VENTIL*, *PIULIȚA OLANDEZĂ*, *PRESGARNITURI*, *GARNITURA*, *ROATA DE MANEVRĂ*, *ȘAIBĂ*, *ȘTIFT*, etc) un ansamblu alcătuit din asemenea piese ar putea fi desenat prin apeluri simple ale acestor subrutine specificând poziția de desenare și unghiul de rotație.

Problema se poate complica și mai mult dacă introducem și o parametrizare a liniilor vizibile și invizibile, diverși indicatori de cotare (cotarea unui ansamblu nu este echivalentă cu suma cotelor pieselor componente).

Am urmărit deci să demonstrăm o posibilitate de realizare a unui sistem integrat de proiectare a unui ansamblu, utilizatorilor fiindu-le pusă la dispoziție o unealtă de muncă modernă pe care o constituie calculatorul electronic împreună cu echipamentele sale periferice.

În privința prezentării finale a piesei sau a ansamblului nu mai sînt de rezolvat probleme deosebite, deoarece axele sau grosimile de linii pot fi obținute din schimbarea penițelor iar problema hașurilor în cazul poligoanelor pline sau goale situate arbitrar unele față de celelalte este realizată prin diverse subprograme (de exemplu subprogramul *SRA*) din Capitolul I.

Aplicații desenate la ploter sînt ilustrate în figurile 2—D,...,5—D.



## PROGRAM PRINCIPAL PIESĂ

```

LOGICAL*1 FIS(15)
DIMENSION X(14),Y(14)
DATA PI/3.1415926535/
TYPE 1
1  FORMAT(' FISIEM DE IESIRE',25)
ACCEPT 2,NRCAR,FIS
2  FORMAT(0,15A1)
CALL ASSIGN(4,FIS,NRCAR)
CALL FDBSET(4,'NEW')
CALL PCAR(.2,.3,90.)
TYPE 3
3  FORMAT(' INTRODUCETI DATELE (A,U,P,Q,T,W,V,Z,U) :',5)
ACCEPT 4,A,U,P,Q,T,W,V,Z,U
4  FORMAT(9F9.2)
U1=U*PI/180.
X(1)=0
Y(1)=T
X(2)=D/2.
Y(2)=T
X(3)=X(2)
Y(3)=0.
X(4)=U/2.+(V-Z)*SIN(U1)/COS(U1)
Y(4)=0.
X(5)=A/2.-.5
Y(5)=0.
X(6)=A/2.
Y(6)=.5
X(7)=X(6)
Y(7)=W-.5
X(8)=X(5)
Y(8)=W
X(9)=P/2.
Y(9)=W
X(10)=X(9)
Y(10)=V
X(11)=U/2.
Y(11)=V
X(12)=X(11)
Y(12)=V-Z
X(13)=0.
Y(13)=Y(12)
X(14)=0.
Y(14)=V
DO 12 I=1,14
12 X(I)=X(I)/S
Y(I)=Y(I)/S
CALL PDATA(X,Y,1,13)
CALL PLOT(X(11),Y(11),2)
CALL PLOT(X(14),Y(14),1)
CALL PLOT(X(12),Y(12),2)
CALL PLOT(X(4),Y(4),1)
DO 11 I=2,12
11 X(I)=-X(I)
CALL PDATA(X,Y,1,13)
CALL PLOT(X(11),Y(11),2)
CALL PLOT(X(14),Y(14),1)
CALL PLOT(X(12),Y(12),2)
CALL PLOT(X(4),Y(4),1)
CALL COTA(X(10),Y(10)+2./S,P/S,2./S,2./S,P,0.,1)
CALL COTA(X(11),Y(11)+1./S,Q/S,1./S,1./S,Q,0.,1)
CALL COTA(X(3),Y(3)-1./S,D/S,-1./S,-1./S,D,0.,1)
CALL COTA(X(6),Y(6)-2.5/S,A/S,-2.5/S,-2.5/S,A,0.,1)
CALL COTA(-X(5)+1.5,Y(5),T/S,-1.5/S,-1./S,T,90.,0)
CALL COTA(-X(5)+2.5/S,Y(5),W/S,-1./S,-2.5/S,W,90.,0)
CALL COTA(-X(5)+3.5/S,Y(5),V/S,-1./S,X(10)-X(7)+3/V,90.,0)
CALL COTA(X(12),Y(12),Z/S,1./S,X(10)-X(6)+1./S,Z,90.,0)
STOP
END

```

PROGRAMUL PIESĂ DIMENSIUNI VARIABLE

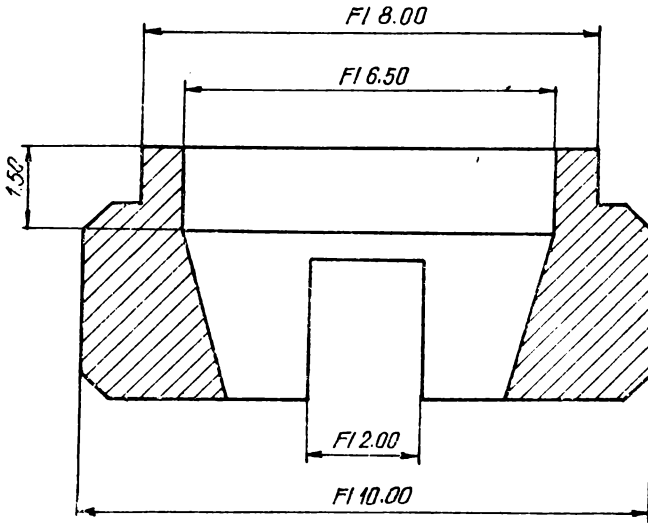


Fig. 3 - D

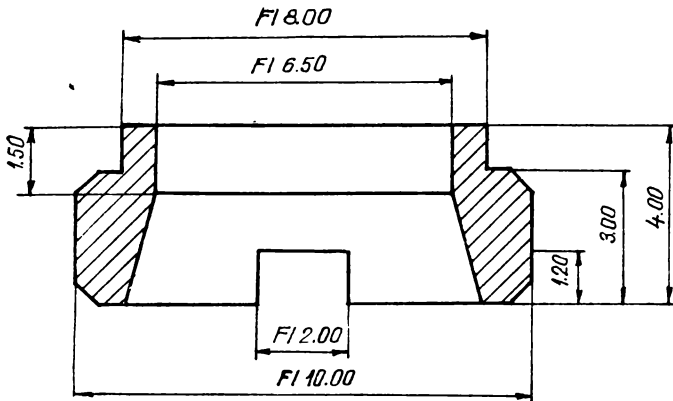


Fig. 4 - D

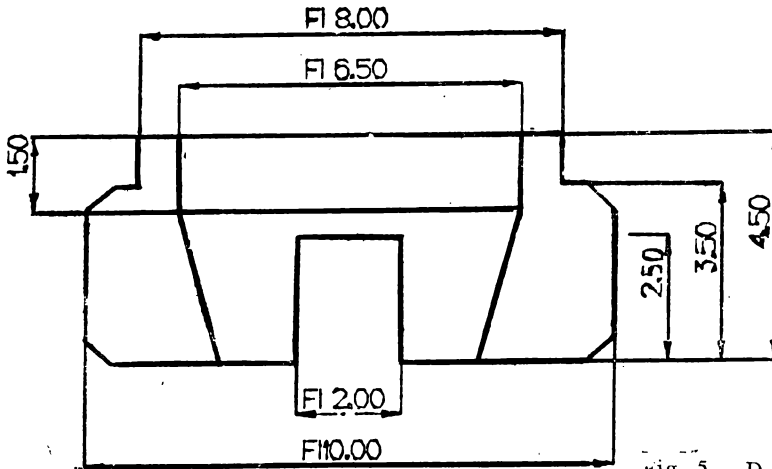


Fig. 5 - D

## ANEXA E

Program pentru simularea automată a unor curbe și suprafețe în  
grafică artistică

Automatizarea proceselor de orice natură este un proces ireversibil. Mijloacele moderne capabile să înlocuiască munca omului își dovedesc utilitatea în toate domeniile de activitate. Chiar în profesii cum ar fi creațiile artistice unde realizările depind într-o măsură accentuată de aptitudinile naturale ale individului, cibernetizarea poate să-și spună cuvântul. Din această cauză posibilitățile grafice ale calculatoarelor încă mai constituie un subiect de fascinație pentru mulți ingineri, arhitecți sau artiști ai penelului. Că acesta este un subiect general nu este surprinzător din moment ce un volum mare din munca atelierelor de proiectare este grafică într-o formă sau alta. Utilizarea echipamentelor periferice moderne ale calculatoarelor electronice permite să se întrezărească o dezvoltare a graficii prin aceste echipamente în arhitectură, inginerie și de ce nu în artele grafice.

În prezent sînt cunoscute numeroase tipuri foarte mult avansate de aplicații grafice ale calculatoarelor în desenarea unor schițe, în filme de desen animat, în alergări simulate, în selectări de perspective ale volumelor arhitecturale, etc deja realizate în alb negru sau color, dintre care unele pe scară mare.

Astfel vom prezenta cîteva simulări ale unor curbe și suprafețe ca modele artistice, simulări care își au geneza în compunerea unor mișcări oscilatori derivate din modele matematice corespunzătoare.

De exemplu dacă se consideră modelul matematic al vibrațiilor libere al sistemelor disipative

$$\ddot{x} + 2h\dot{x} + \omega^2 x = 0$$

soluția are forma

$$x = e^{-ht}(B_1 \sin \omega t + C_1 \cos \omega t)$$

Schimbînd variabila dependentă  $x$  prin  $y$  și admițînd alte condiții inițiale soluția are forma

$$y = e^{-ht}(D_1 \sin \omega t + E_1 \cos \omega t)$$

Însumînd două soluții de prima formă pentru condiții inițiale diferite și procedînd la fel și cu două soluții de forma a doua se obține

$$x = e^{h_1 t} B_1 \sin \omega_1 t + e^{-h_2 t} B_2 \sin \omega_2 t + e^{-h_1 t} C_1 \cos \omega_1 t + e^{-h_2 t} C_2 \cos \omega_2 t$$

$$y = e^{-h_1 t} D_1 \sin \omega_1 t + e^{-h_2 t} D_2 \sin \omega_2 t + e^{-h_1 t} E_1 \cos \omega_1 t + e^{-h_2 t} E_2 \cos \omega_2 t$$

Admițînd următoarele notații pentru programul de calcul

$$h_1 = A_1, \quad h_2 = A_2, \quad t = T, \quad \omega_1 = F_1, \quad \omega_2 = F_2$$

și grupînd convenabil relațiile se obține:

$$R1 = 100 * EXP(-A1 * .01 * T)$$

$$R2 = 100 * EXP(-A2 * .01 * T)$$

$$S1 = R1 * SIN(F1 * T)$$



$$S2 = R2 * \sin(F2 * T)$$

$$T1 = R1 * \cos(F1 * T)$$

$$T2 = R2 * \cos(F2 * T)$$

$$X = 511. + B1 * S1 + B2 * S2 + C1 * T1 + C2 * T2$$

$$Y = 511. + D1 * S1 + D2 * S2 + E1 * T1 + E2 * T2$$

Pentru diferite valori ale constantelor  $A_i, B_i, C_i, D_i, E_i, F_i (i = 1, 2)$  succesiunea punctelor  $(X, Y)$  crează imaginea unor curbe și suprafețe, teoretic foarte greu de realizat iar practic imposibil prin metodele cunoscute (fig. 1 — E, fig. 2 — E, fig. 3 — E, fig. 4 — E, fig. 5 — E, fig. 6 — E și fig. 7 — E).

```
* SEGMENT BENS0N,BENBUF
* DEFINE FILE Z2(DVT:MT,RCF:V,BFS:2488)=9 3
  COMMON /BENS0N/ XP,YP,X0,Y0,IX,IY,NPOS,PAS,ICAR,IBUF,NCAR,MBUF,EX,
  *EY,NBUF,INBUF,IBLOC,NTAPE,IBIT
  COMMON / BENBUF / IBUFF
  DIMENSION IBUFF(620)
  DIMENSION A(2),B(2),C(2),D(2),E(2),F(2)
  ND=9
  LBUF=620
  CALL IBENA(IBUFF,LBUF,ND)
  DO 1 J=1,5
    NBL0C=J
    READ 5,(A(1),B(1),C(1),D(1),E(1),F(1),A(2),B(2),C(2),D(2),E(2),F(2)
    1)
  5 FORMAT (12F6.3)
    PRINT 10,A(1),B(1),C(1),D(1),E(1),F(1),A(2),B(2),C(2),D(2),E(2),F(
    12)
  10 FORMAT(' ',A1=',F6.3,1X',B1=',F6.3,1X',C1=',F6.3,1X',D1=',F6.3,1X
  1',E1=',F6.3,1X',F1=',F6.3//',A2=',F6.3,1X',B2=',F6.3,1X',C2=',F
  26.3,1X',D2=',F6.3,1X',E2=',F6.3,1X',F2=',F6.3/)
    H=.05
    CALL PNUMA(0.,0.,NBL0C,0.,0.)
    CALL ECHEL(0.02,0.02,0.0,0.0)
    DO 20 I=1,4001
      T=(I-1)*H
      R1=100.*EXP(-A(1)*.01*T)
      R2=100.*EXP(-A(2)*.01*T)
      S1=R1*SIN(F(1)*T)
      S2=R2*SIN(F(2)*T)
      T1=R1*COS(F(1)*T)
      T2=R2*COS(F(2)*T)
      X=511.+B(1)*S1+B(2)*S2+C(1)*T1+C(2)*T2+(J-1)*1500./2.
      Y=511.+D(1)*S1+D(2)*S2+E(1)*T1+E(2)*T2
      IF(I-1)7,8,7
    8 CALL TRAS(X,Y,0)
    7 CALL TRAS(X,Y,1)
  20 CONTINUE
  1 CONTINUE
  CALL PNUMA(0.0,0.0,999,0.0,0.0)
  STOP
  END
```

```

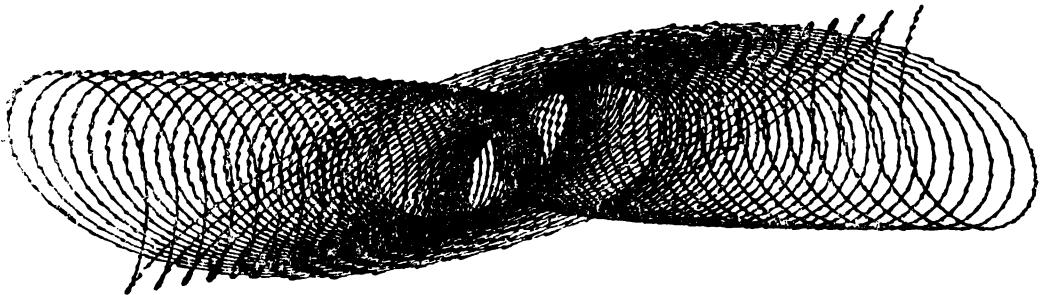
10 DIM A[2], B[2], C[2], D[2], E[2], F[2]
20 DISP #VAL(1) #I #VAL(2) #J
30 INPUT A[1], B[1], C[1], D[1], E[1], F[1], A[2], B[2], C[2], D[2], E[2], F[2]
40 H=0.05
50 SCALE 800, 1700, 2, 200
60 FOR I=1 TO 4000 STEP 1
70 T=(I/1)*H
80 R1=100*EXP(A[1]*0.01*T)
90 R2=100*EXP(A[2]*0.01*T)
100 S1=R1*SIN(B[1]*T)
110 S2=R2*SIN(B[2]*T)
120 T1=R1*COS(C[1]*T)
130 T2=R2*COS(C[2]*T)
140 X=511+B[1]*S1+B[2]*S2+C[1]*T1+C[2]*T2
150 Y=511+D[1]*S1+D[2]*S2+E[1]*T1+E[2]*T2
160 IF I=1 THEN 190
170 PLOT X, Y, 2
180 GOTO 200
190 PLOT Y, Y, 1
200 NEXT I
210 END

```

```

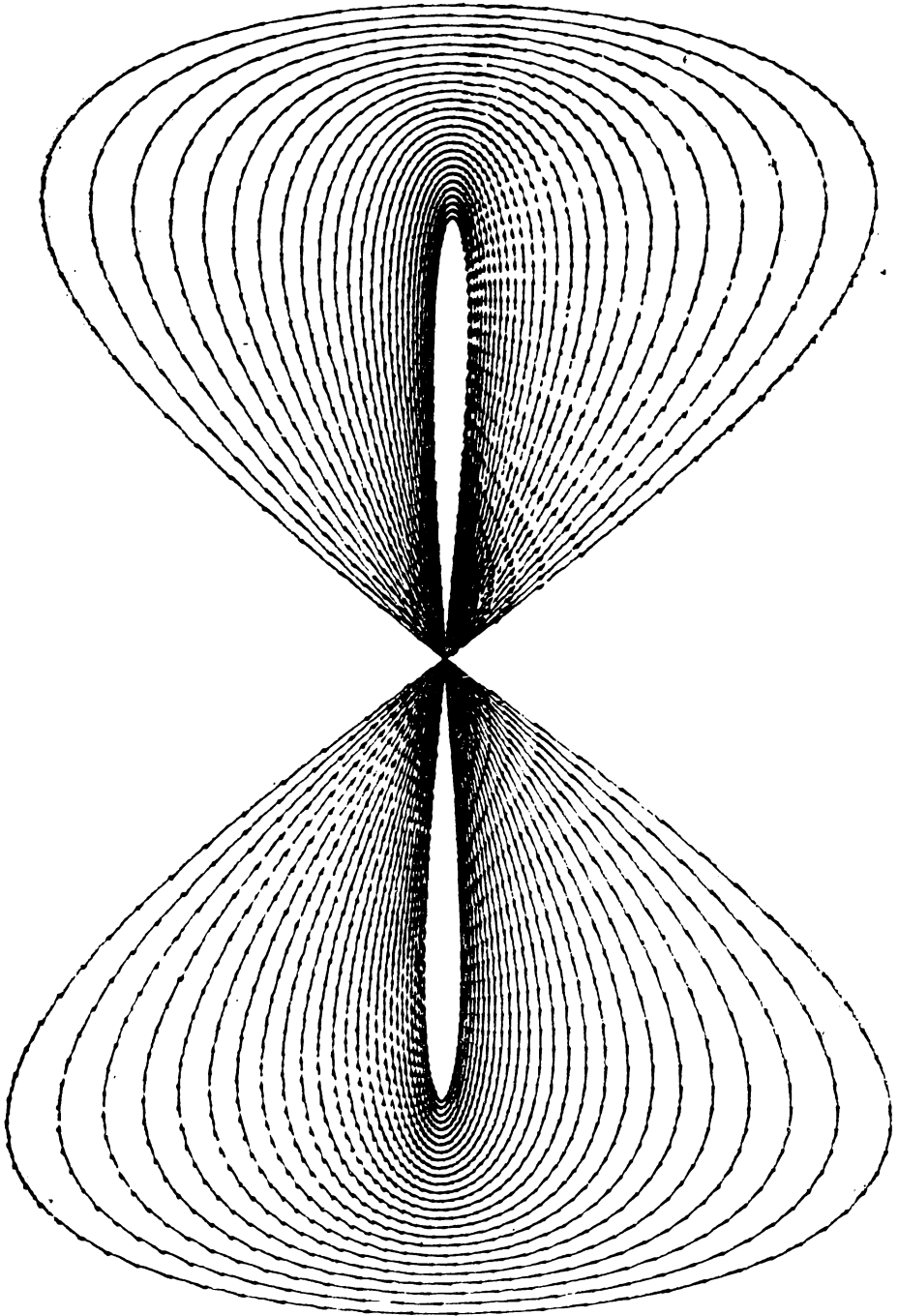
10 A=1
20 B=1.5
30 FOR I=1 TO 6283 STEP 1
40 T=I
50 X1=A*COS(T/1000)
60 Y1=A*SIN(T/1000)
70 R=SQR(Y1^2+(X1+B)^2)
80 X=511+150*(X1+R*COS(T/10))
90 Y=611+150*(Y1+R*SIN(T/10))
95 SCALE 100, 1100, 200, 1100
100 IF I=1 THEN 190
110 PLOT X, Y, 2
120 GOTO 140
130 PLOT X, Y, 1
140 NEXT I
150 END.

```



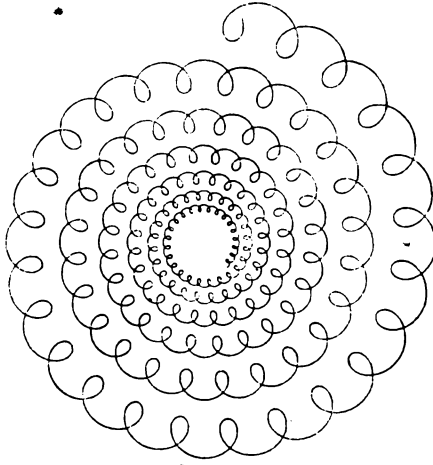
$A1 = .300$   $B1 = 1.500$   $C1 = 1.500$   $D1 = -1.500$   $E1 = 1.500$   $F1 = 2.800$   
 $A2 = .100$   $B2 = 1.550$   $C2 = 1.550$   $D2 = 1.550$   $E2 = 1.550$   $F2 = 2.864$

Fig. 1 - E



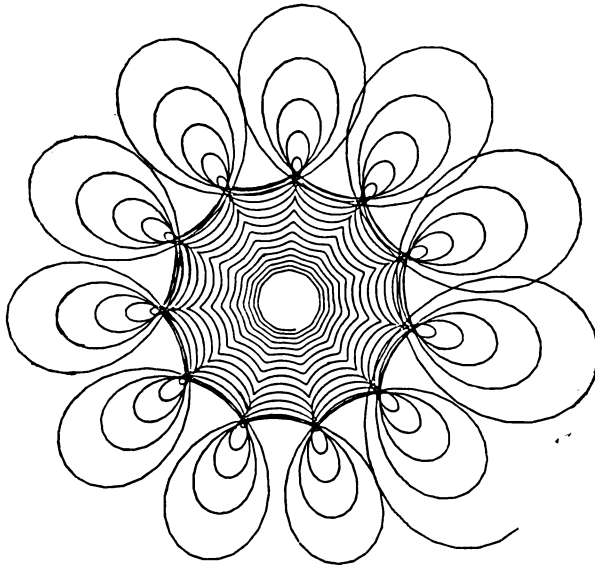
A1 = .250 B1= 4.000 C1 .000 D1= .000 E1= .000 F1=1.000  
A2= 2.000 B2= .000 C2= .000 D2=4.000 E2= .000 F2=2.000

Fig. 2. - E



A1= 1.000 B1= 3.000 C1= .000 D1= .000 E1= 3.000 F1= .200  
 A2= 1.000 B2= .000 C2= .300 D2= -.300 E2= .000 F2= 5.000

Fig. 3 — E



A1= 1.000 B1= 2.000 C1= 2.000 D1= 2.000 E1=-2.000 F1= .500  
 A2= 3.000 B2=-1.000 C2= 1.000 D2=-1.000 E2=-1.000 F2= 5.000

Fig. 4 — E

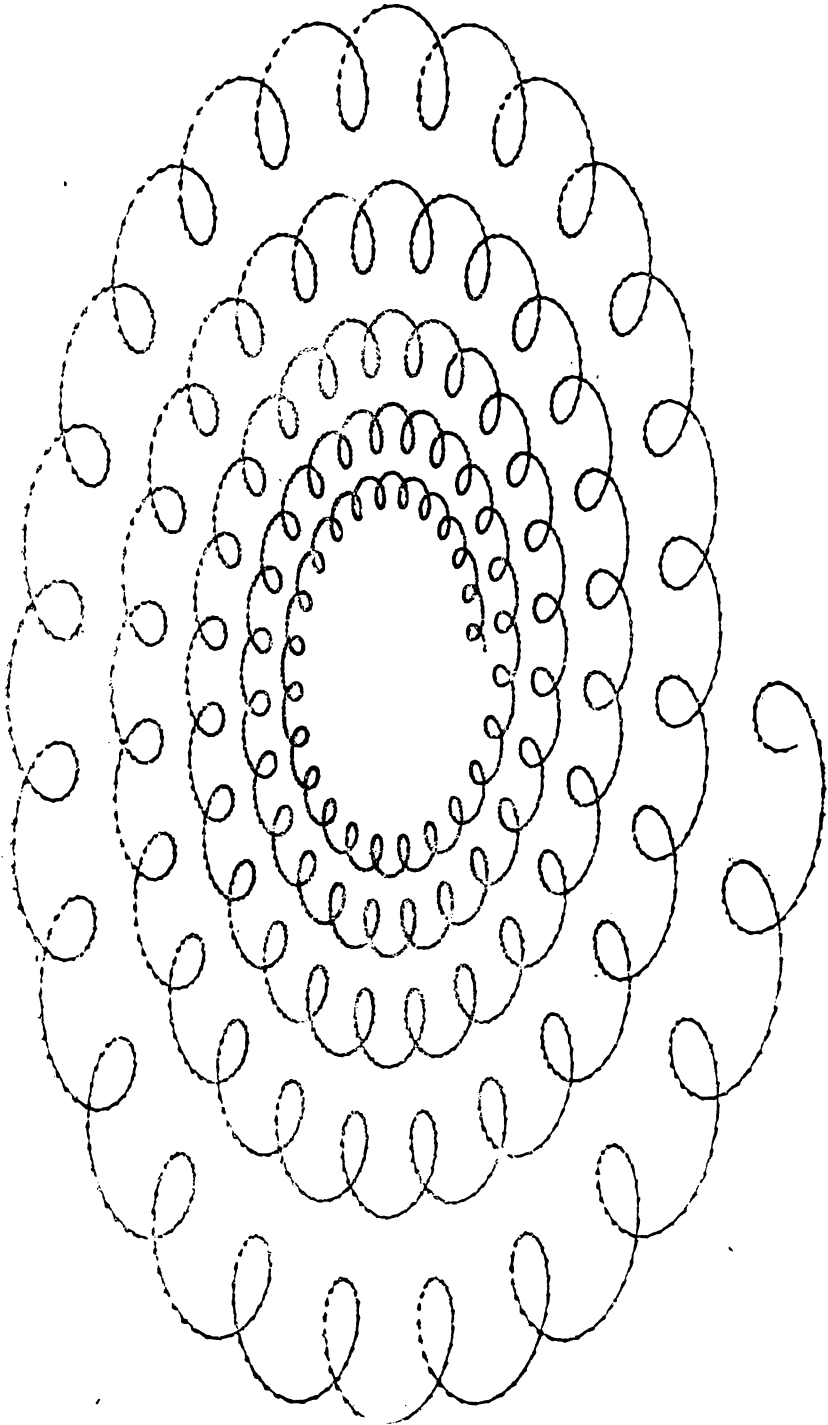


Fig. 5 — E

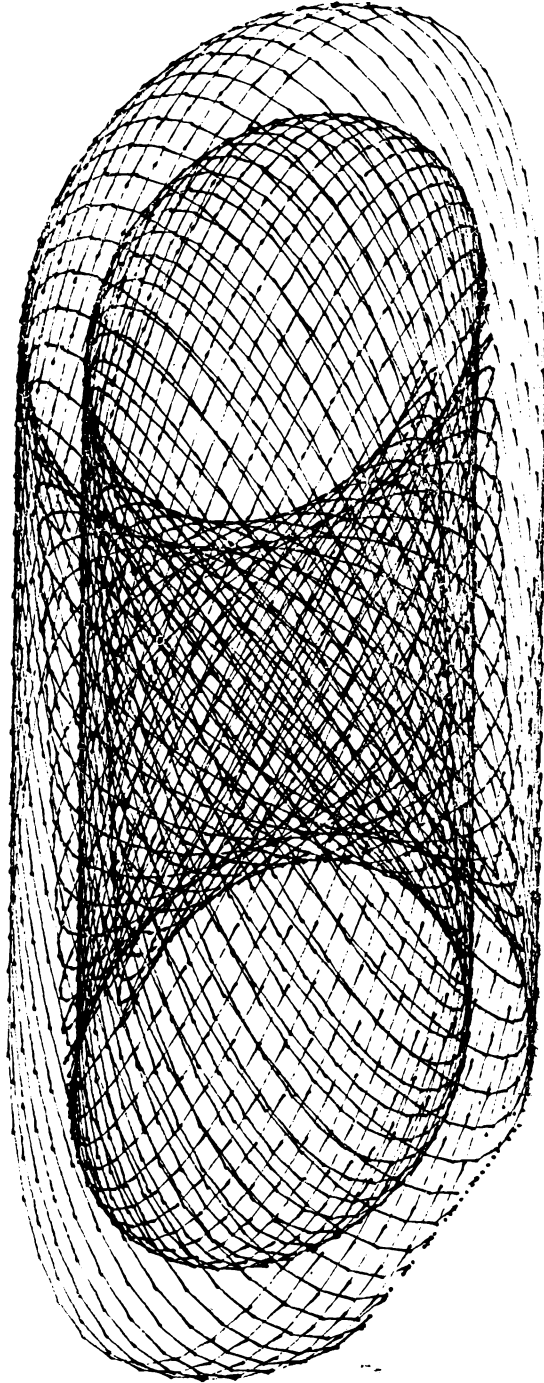


Fig. 6 — E

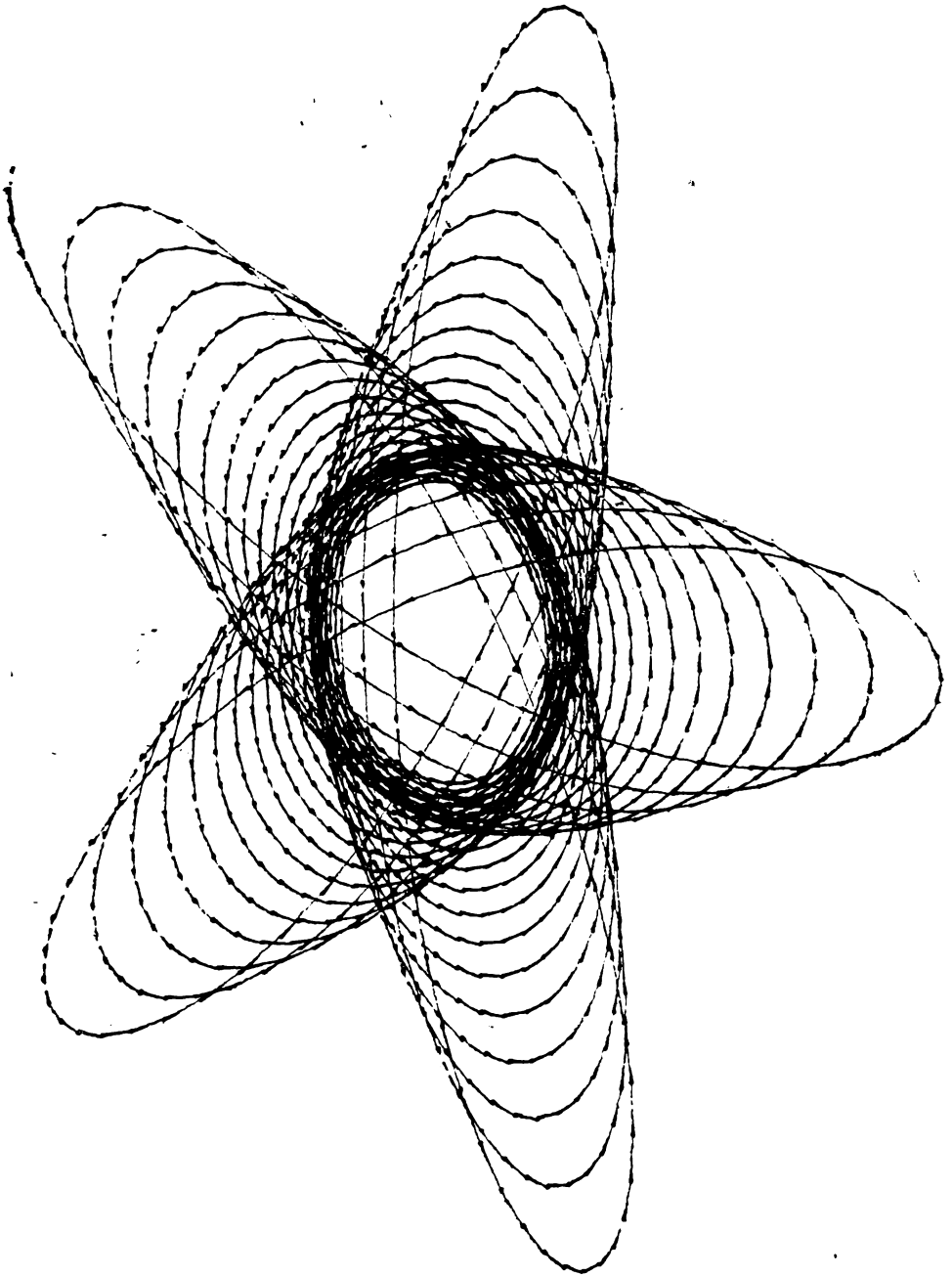


Fig. 7 — E

Alte aplicații deosebite se pot obține pentru următoarele valori ale parametrilor:

- $A1 = .400$   $B1 = 1.400$   $C1 = 1.400$   
 $A2 = 1.000$   $B2 = -1.500$   $C2 = 1.500$   
 $D1 = -1.400$   $E1 = 1.400$   $F1 = 2.000$   
 $D2 = 1.500$   $E2 = 1.500$   $F2 = 3.000$
- $A1 = .250$   $B1 = 4.000$   $C1 = .000$   
 $A2 = 2.000$   $B2 = .000$   $C2 = .000$   
 $D1 = .000$   $E1 = .000$   $F1 = 1.000$   
 $D2 = 4.000$   $E2 = .000$   $F2 = 2.000$
- $A1 = 0.400$   $B1 = 1.400$   $C1 = 1.400$   
 $A2 = 1.000$   $B2 = -1.500$   $C2 = 1.500$   
 $D1 = -1.400$   $E1 = 1.400$   $F1 = 2.000$   
 $D2 = 1.500$   $E2 = 1.500$   $F2 = 3.000$
- $A1 = 1.000$   $B1 = 2.000$   $C1 = 2.000$   
 $A2 = -0.300$   $B2 = -0.650$   $C2 = 0.650$   
 $D1 = -2.000$   $E1 = -2.000$   $F1 = 1.000$   
 $D2 = 0.650$   $E2 = 0.650$   $F2 = 3.010$
- $A1 = 0.300$   $B1 = 0.300$   $C1 = 0.000$   
 $A2 = 1.500$   $B2 = 0.000$   $C2 = 0.300$   
 $D1 = 1.500$   $E1 = 2.000$   $F1 = 1.500$   
 $D2 = 1.500$   $E2 = 2.000$   $F2 = 0.000$

## BIBLIOGRAFIE

- [ 1 ] AHLBERG, J. M.: *SPLINE APROXIMATION AND COMPUTER AIDED DESIGN*, IN: ADVANCES IN COMPUTERS, No. 10, NEW-YORK, ACADEMIC PRESS, 1970.
- [ 2 ] AHLBERG, J. M. NILSON, E. N.: *THE THEORY OF SPLINES AND THEIR APPLICATIONS*, NEW-YORK, ACADEMIC PRESS, 1967.
- [ 3 ] BERNHILL, R. E.: *COMPUTER AIDED GEOMETRIC DESIGN*, NEW-YORK, ACADEMIC PRESS, 1979.
- [ 4 ] BALTAČ, V. și colectiv: *CALCULATOARE ELECTRONICE, GRAFICĂ INTERACTIVĂ și PRELUCRAREA IMAGINILOR* Editura Tehnică, București, 1985.
- [ 5 ] BÉZIER, P.: *DEFINITION NUMERIQUE DES COURBES ET SURFACES NON MATHEMATIQUES*, AUTOMATISME 11-1966, 13-1968.
- [ 6 ] COONS, S. A.: *SURFACES FOR COMPUTER AIDED DESIGN OF SPACE FORMS* MIT PROJECT MAC. TR. -41, 1967.
- [ 7 ] DRS, L. NOVAK, J. JEZEC, F.: *POCITACOVA GRAFIKA*, CVUT, PRAHA, 1980.
- [ 8 ] DRS, L.: *PLOCHY VE VYPOCETNI TEHNICE*, MS-SNTL PRAHA, 1984.
- [ 9 ] ENCARNACAO, J. L.: *COMPUTER GRAPHICS*, R. OLDENBOURG VERLAG, MÜNCHEN WIEN, 1975.
- [ 10 ] FAUX, I. D. PRATT, M. J.: *COMPUTATIONAL GEOMETRY FOR DESIGN AND MANUFACTURE*, ELLIS GARWOOD LTD., ENGLAND 1981
- [ 11 ] FOKS, A. PRATT, M.: *VICISLITELANIA GEOMETRIIA PRIMENENIA V IROVANIII NA PROIZVODSTVE*, MOSKVA, - MIR -, 1982.
- [ 12 ] FOLEY, I. D. A van DAM.: *FUNDAMENTALS OF INTERACTIVE COMPUTER GRAPHICS*, THE SYSTEMS PROGRAMMING SERIES, ADDISON-WESLEY PUBLISHING COMPANY 1983.



- [13] FERGUSON, J.: *MULTIVARIABLE CURVE INTERPRETATION*, JACM, 11—1964.
- [14] GRANAT, L. SECHOVSKY, H.: *POČITAČOVA GRAFICA ČVUT PRAHA*, SNTL 1980.
- [15] JEZEK, F.: *INTERPOLACE KRIVEK A PLOCH PRACE K ODBORNE ASPIRANTSKE ZKOUSCE*, ČVUT, PRAHA, 1977.
- [16] KUBERT, B.: *THE PERSPECTIVE REPRESENTATION OF FUNCTION OF TWO VARIABLES*, JACM, V 15, 1968, N2, 193.
- [17] MORVAL, P. LUCAS, M.: *IMAGES ET ORDINATEURS*, LAROUSSE, PARIS, 1980.
- [18] MARINESCU, I. D. TĂNĂSESCU, A. CONSTANTINESCU, R.: *DESFĂȘURAREA ASISTATĂ DE CALCULATOR*, EDITURA TEHNICĂ, BUCUREȘTI, 1987.
- [19] NEWMANN, W. M. SPROULL, R. F.: *PRINCIPLES OF INTERACTIVE COMPUTER GRAPHICS* NEW-YORK, MCGRAW HILL, 1973.
- [20] NEWMANN, W. M.: *COMPUTER GRAPHICS*, NEW-YORK, 1980.
- [21] RIESENFELD, R. F.: *APPLICATIONS OF B-SPLINE APPROXIMATION TO GEOMETRIC PROBLEMS OF COMPUTER AIDED DESIGN* PH. D. THESIS, SYRACUSE 1973.
- [22] ROGERS, D. F. ADAMS, J. A.: *MATHEMATICAL ELEMENTS FOR COMPUTER GRAPHICS* MCGRAW-HILL, 1976.
- [23] SAVA, I. TĂNĂSESCU, A.: *PROBLEMA ELIMINĂRII LINIILOR INVIZIBILE ÎN PERSPECTIVA ARHITECTURALĂ* ARHITECTURĂ Nr.2/1975.
- [24] SUTHERLAND, E. I. SPROULL, R. F. SCHUMACKER, A. R.: *A CHARACTERIZATION OF TEN HIDDEN-SURFACE ALGORITHMS*, ACM COMPUTING SURVEYS, 1—1979.
- [25] TĂNĂSESCU, A.: *UTILIZAREA CALCULATORILOR ELECTRONICE ÎN ARHITECTURA ȘI PERSPECTIVA ARHITECTURALĂ* ED. DIDACTICĂ ȘI PEDAGOGICĂ, BUCUREȘTI, 1972.
- [26] TĂNĂSESCU, A.: *ALGORITMUL GENERAL AL REPREZENTĂRIILOR GEOMETRICE AUTOMATE*, ARHITECTURA, 1/1981.
- [27] TĂNĂSESCU, A.: *REPREZENTĂRI AUTOMATE ÎN GEOMETRIA DESCRIPTIVĂ* ARHITECTURA 3/1983.
- [28] TĂNĂSESCU, A. COLECTIV: *GRAFICA COMPUTER VOL. I—XII* BIBLIOTECA TEHNICĂ ISPIF, BUCUREȘTI, 1982—1986.
- [29] TĂNĂSESCU, A. MARINESCU, I. D. CONSTANTINESCU, R. PRECUPEȚU, P. BUSUIOC, L.: *REPREZENTĂRI GEOMETRICE ÎN PROIECTAREA ASISTATĂ DE CALCULATOR*. BUCUREȘTI, OFICIUL DE DOCUMENTARE ȘI INFORMARE AL M.I.Et., 1986.
- [30] TĂNĂSESCU A.: *POLYEDER UND COMPUTERGRAPHIK* POTSDAMER FORSCHUNGEN, R. B. HEFT 43 S. 81—92, DDR, 1984.
- [31] TĂNĂSESCU A.: *GEOMETRIE DESCRIPTIVĂ, PERSPECTIVĂ. AXONOMETRIE* ED. DID. ȘI PED., BUCUREȘTI 1975.
- [32] URBAN J.: *POČITAČOVA GRAFICA V PROJEKTOVANI* ČVUT, PRAHA, 1984.
- [33] URBAN J.: *POČITAČOVA GRAFICA '78* GEOMETRICKA PROBLEMATICA ZBORNIK KONFERENCIE, SMOLENICE, ČSSR, 1978.
- [34] WILLIAMSON, H.: *HIDDEN-LINE PROGRAM*, CACM, V.15, N.2, 100.
- [35] WRIGHT, T. J.: *A TWO SPACE SOLUTION TO THE HIDDEN-LINE PROBLEM OR PLOTTING FUNCTIONS, OF TWO VARIABLES*, IEEE TRANS ON COMP., V—C—22, 1979, N1, 28.

# Biblioteca de automată, informatică, electronică, management CĂRȚI ÎN PREGĂTIRE

- |   |  |
|---|--|
| 1. Colectiv învățămînt,<br>centre de calcul | Învățăm FORTRAN 77.... conversînd cu<br>calculatorul,                                      |
| 2. Colectiv învățămînt,<br>centre de calcul | Învățăm PASCAL... conversînd cu calcu-<br>latorul  |
| 3. Baltac V. și colectiv,                   | Manual practic de informatică și tehnică de<br>calcul                                      |
| 4. Petrescu A. și colectiv,                 | Totul despre... Felix PC   |
| 5. Munteanu V. și a.                        | Limbaajul C pe mini și micro   |
| 6. R. W. Hockney, C. R. Jesshope            | Calculatoare paralele. Arhitectură, progra-<br>mare, algoritmi (traducere din l. engleză). |
| 7-10. * * *                                 | Automatică, management, calculatoare (AMC)<br>vol. 61-64.                                  |
| 11. Costache N, Coojcaru I,                 | Microeconomie industrială, Concepte, metode,<br>aplicații                                  |
| 12. Colectiv ITCI, M.I.A.,                  | Practica informaticii și tehnicii de calcul în<br>industria alimentară                     |
| 13. Bădea N. ș.a.                           | Optimizarea organizării și conducerii între-<br>prinderilor                                |
| 14. Trandafir I, Marinescu V. și a.         | Ingineria programării  |

## Seria „Tehnica la zi“

- |                      |                                       |
|----------------------|---------------------------------------|
| 15. Isaic Maniu Al.  | Statistică asistată de calculator     |
| 16. Hângănuț I. ș.a. | Testare automată                      |
| 17. Sabău Gh.        | Baze de date. De la concept la mașini |

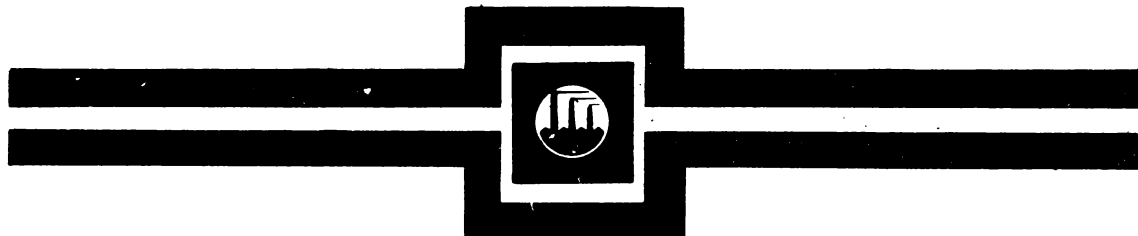
Vom anunța în timp util cînd unele dintre aceste lucrări vor fi în faze avansate de apariție. Cărți apărute sau în curs de apariție sînt anunțate în ultimele pagini ale volumului 1.



● Volumul 1: biblioteca software de rutine grafice a mesei de desen DIGIGRAF (I), subprogramele pentru punct, dreaptă, plan, paralelism, perpendicularitate (II), programe și aplicații pentru secțiuni și intersecții de poliedre (III), pentru conice, quadrice, suprafețe de rotație și translație (IV), pentru reprezentarea spațiului 3D pe 2D (V), și a obiectelor spațiale (VI), cum și anexe cu formulele fundamentale, respectiv cu utilizarea mesei de desen ARISTO.

● Volumul 2: concepte, algoritmi, programe și aplicații pentru linii ascunse și suprafețe invizibile (VII), pentru vizualizarea sau reprezentarea grafică a curbelor și suprafețelor (VIII), pentru interpolări cu curbe speciale (Bézier, Ferguson, B-spline, COONS), pentru curbe (IX) sau suprafețe (X), ca și anexe cu programe de desenare automată a planurilor în arhitectură și construcții, în construcții de mașini sau în grafica artistică.

● Adresată, îndeosebi, proiectanților, studenților, cadrelor didactice, utilizatorilor informaticieni din toate domeniile, cu o atenție aparte pentru cei din construcții, sistematizare, arhitectură.



EDITURA TEHNICĂ