

A. TĂNĂSESCU
I. D. MARINESCU

R. CONSTANTINESCU
L. BUSUIOC

GRAFICĂ ASISTATĂ

Programe FORTRAN pentru reprezentări geometrice

1

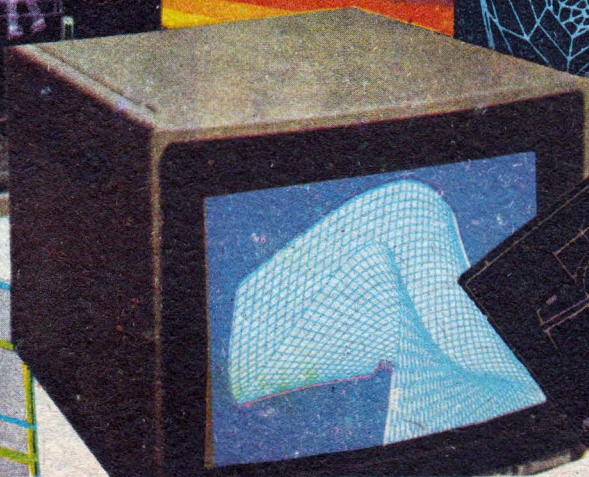
AUTOMATICA

ELECTRONICA

INFORMATICA

MANAGEMENT

SERIA PRACTICĂ





TECHNOLOGY OF AUTOMATIC INFORMATION SYSTEMS
 ELECTRONIC MANAGEMENT

1. Introduction to the field of automatic management systems.
 2. The role of automatic management systems in industry.
 3. The structure of automatic management systems.
 4. The classification of automatic management systems.
 5. The development of automatic management systems.
 6. The application of automatic management systems.
 7. The advantages of automatic management systems.
 8. The disadvantages of automatic management systems.
 9. The future of automatic management systems.
 10. The conclusion.

11. The role of automatic management systems in industry.
 12. The structure of automatic management systems.
 13. The classification of automatic management systems.
 14. The development of automatic management systems.
 15. The application of automatic management systems.
 16. The advantages of automatic management systems.
 17. The disadvantages of automatic management systems.
 18. The future of automatic management systems.
 19. The conclusion.

20. The role of automatic management systems in industry.
 21. The structure of automatic management systems.
 22. The classification of automatic management systems.
 23. The development of automatic management systems.
 24. The application of automatic management systems.
 25. The advantages of automatic management systems.
 26. The disadvantages of automatic management systems.
 27. The future of automatic management systems.
 28. The conclusion.

29. The role of automatic management systems in industry.
 30. The structure of automatic management systems.
 31. The classification of automatic management systems.
 32. The development of automatic management systems.
 33. The application of automatic management systems.
 34. The advantages of automatic management systems.
 35. The disadvantages of automatic management systems.
 36. The future of automatic management systems.
 37. The conclusion.

38. The role of automatic management systems in industry.
 39. The structure of automatic management systems.
 40. The classification of automatic management systems.
 41. The development of automatic management systems.
 42. The application of automatic management systems.
 43. The advantages of automatic management systems.
 44. The disadvantages of automatic management systems.
 45. The future of automatic management systems.
 46. The conclusion.

47. The role of automatic management systems in industry.
 48. The structure of automatic management systems.
 49. The classification of automatic management systems.
 50. The development of automatic management systems.
 51. The application of automatic management systems.
 52. The advantages of automatic management systems.
 53. The disadvantages of automatic management systems.
 54. The future of automatic management systems.
 55. The conclusion.

56. The role of automatic management systems in industry.
 57. The structure of automatic management systems.
 58. The classification of automatic management systems.
 59. The development of automatic management systems.
 60. The application of automatic management systems.
 61. The advantages of automatic management systems.
 62. The disadvantages of automatic management systems.
 63. The future of automatic management systems.
 64. The conclusion.

65. The role of automatic management systems in industry.
 66. The structure of automatic management systems.
 67. The classification of automatic management systems.
 68. The development of automatic management systems.
 69. The application of automatic management systems.
 70. The advantages of automatic management systems.
 71. The disadvantages of automatic management systems.
 72. The future of automatic management systems.
 73. The conclusion.

74. The role of automatic management systems in industry.
 75. The structure of automatic management systems.
 76. The classification of automatic management systems.
 77. The development of automatic management systems.
 78. The application of automatic management systems.
 79. The advantages of automatic management systems.
 80. The disadvantages of automatic management systems.
 81. The future of automatic management systems.
 82. The conclusion.

83. The role of automatic management systems in industry.
 84. The structure of automatic management systems.
 85. The classification of automatic management systems.
 86. The development of automatic management systems.
 87. The application of automatic management systems.
 88. The advantages of automatic management systems.
 89. The disadvantages of automatic management systems.
 90. The future of automatic management systems.
 91. The conclusion.

92. The role of automatic management systems in industry.
 93. The structure of automatic management systems.
 94. The classification of automatic management systems.
 95. The development of automatic management systems.
 96. The application of automatic management systems.
 97. The advantages of automatic management systems.
 98. The disadvantages of automatic management systems.
 99. The future of automatic management systems.
 100. The conclusion.

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. Traducătoare 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. Samuelli, J. Pignaret, A. Sarazin. Instrumentația electronică în fizica nucleară
T. Homoș. Capacitatea de producție în construcții de mașini
S. Radu, D. Floti. Centrale telefonice automate. Sisteme de comutație
M. Bodea ș.a. Tranzistoare cu efect de cîmp
D. N. Șapiro. Proiectarea radioreceptoarelor
V. Antonescu, M. Popovici. Ghid pentru controlul statistic al calității producției
N. Stanciu ș.a. Tehnica imaginii în cinematografe și televiziune
P. Vezeanu, Șt. Pătrașcu. Măsurarea temperaturii în tehnică
T. Pănescu, 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 maiștri
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, automată, 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 ierarhizate, 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

1



EDITURA TEHNICĂ
București, 1989

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

Red actor: 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. 1411 la
Întreprinderea poligrafică
„13 Decembrie 1918”,
str. Grigore Alexandrescu nr. 89-97
București,
Republica Socialistă România

Să menținem întotdeauna industria noastră
la nivelul celor mai noi cuceriri ale tehnicii
și științei!

NICOLAE CEAUȘESCU

(Din cuvântarea la Plenara Consiliului Național
al Oamenilor Muncii, 31 iulie 1989).

CUVÎNT ÎNAINTE

În acest an — în primăvara căruia România a rambursat în întregime datoria externă, devenind, cu adevărat, independentă economic și politic — producția industrială va fi de cca 135 ori mai mare decât în 1945, o creștere de 120 ori fiind realizată după Congresul al IX-lea al Partidului Comunist Român.

Rezultatele de pînă acum și prevederile pentru anul 1990 asigură îndeplinirea hotărîrilor Congresului al XIII-lea al partidului privind dezvoltarea economico-socială a țării, realizarea obiectivului strategic al celui de al 8-lea cincinal — trecerea României la stadiul de țară socialistă mediu dezvoltată.

Din obiectivul fundamental al celui de al 9-lea cincinal —, care este asigurarea trecerii la cea de a doua fază, superioară, a construirii societății socialiste multilateral dezvoltate în țara noastră, pe baza și a celor mai avansate cuceriri ale științei și tehnicii — desprindem sarcina de continuare fermă a politicii de dezvoltare a industriei prelucrătoare, acordînd prioritate sectoarelor tehnicii de vîrf. Astfel, industria electronică va dezvolta, cu precădere, producția de componente perfecționate și de echipamente de electronică industrială.

Prin efortul cercetării științifice — orientat spre îndeplinirea programelor de organizare, modernizare și dezvoltare, vor fi asimilate în fabricație, în toate ramurile economiei, noi tipuri de mașini și instalații tehnologice, la nivelul realizărilor înalte de pe plan mondial.

Învățămîntul va asigura în 1991—95 pregătirea profesională a aproape 2 milioane persoane (întregul tineret urmînd școala de 12 ani), iar programele de reciclare, pentru perfecționarea cunoștințelor profesionale ale tuturor categoriilor de oameni ai muncii, vor cuprinde peste 3,2 milioane persoane anual.

În acest larg context, prezintă o mare importanță producerea și folosirea eficientă, bine direcționată, a echipamentelor și utilajelor de înaltă tehnicitate cu care sînt sau vor fi dotate unitățile din industrie, din cercetare sau învățămînt.

Ne îndreptăm, spre exemplu, atenția asupra mijloacelor de electronică industrială, cu o extinsă paletă a aplicațiilor, de la conducerea proceselor industriale și cercetarea-proiectarea-fabricarea produselor de calitate pînă la instruirea în toate formele de învățămînt. Multe dintre aceste produse ale industriei electronice și de tehnică de calcul au interfețe grafice, pentru care există sau trebuie scrise adevărate pachete de programe de aplicații. Bazată și pe o producție proprie de microprocesoare s-a dezvoltat fabricația de calculatoare profesionale, de microcalculatoare ca și de echipamente periferice adecvate: display-uri grafice, mese de desen (plotere), imprimante grafice etc.; alături de acestea se produc minicalculatoare interactive, compatibile cu familii uzuale pe plan internațional, ca și calculatoare medii, cu performanțe ridicate.

Prelucrarea grafică a datelor necesită tehnici diferite de ale prelucrării clasice, sistemele grafice — bazele pe concepte matematice și logice evaluate — impunând *standardizări* în hardware și software, pentru a-și asigura independența față de aplicații și față de echipamentele utilizate; realizarea acestei *portabilități* rezultă și din susținute activități de proiectare pentru obținerea de nuclee grafice standardizate, ce interfațează aplicația și operatorul, asigurând independența de limbajul programului de aplicație și de sistemul de operare al echipamentului, în adevărate *stații grafice*.

Industria producătoare de echipamente electronice interactive este impulsionată de *cerințele crescînde și diversificate ale economiei*, de rezultatele cercetărilor în toate domeniile, de performanțele impuse aplicațiilor; se dezvoltă atât la producători cît și la utilizatori adevărate „*industrii de programe*“ de sistem și de aplicații. Deși producătorii de tehnică de calcul reușesc să încorporeze în echipamente și în sistemele de operare facilități grafice din ce în ce mai adîncite, *programele de aplicații* scrise de utilizatori nu se reduc numeric *ci sporesc* și în complexitate, pe măsura dezvoltării cercetării-proiectării asistate, a fabricației și conducerii producției asistate, a interesului pentru asistarea cu calculatorul în toate domeniile economiei.

Mijloacele românești de comunicare în masă (presa centrală, radioul, televiziunea ș.a.), au reflectat *extinderea eficientă, a feluritelor aplicații de asistare grafică* ce asigură competitivitatea la export, calitatea superioară, economia de materii prime și materiale, combustibili și energie, modernizarea industriei, ș.a.: proiectarea și fabricarea corpului și compartimentelor navelor, conducerea roboților, proiectarea, testarea și execuția circuitelor electronice imprimate, proiectarea clădirilor, optimizarea și sistematizarea ansamblurilor arhitecturale, prelucrarea imaginilor în medicină, fotogrammetrie, teledetecție ș.a., simbolizate prin CAD, CAM, CAE (sau PAC, IAC, FAC), deci proiectare, fabricație, inginerie — asistate de calculator, corelate cu suportul de instruire și educare asistată (IEAC), prezent atât în programele de reciclare cît și în învățămînt.

Editurile, cu precădere Editura Tehnică, și-au adus contribuția în domeniu *publicînd cărți* privind grafica interactivă și prelucrarea imaginilor, ca și module privind terminalele grafice românești, aplicații grafice — inclusiv pe calculatoare personal-profesionale — în manuale și publicații seriale (exemplu AMC — Automatică, management, calculatoare) ș.a.

Din aceleași cicluri continue fac parte și *volumele de față, orientate* (conform și unor preocupări îndelungate ale autorilor în învățămîntul și cercetarea din arhitectură-construcții (mai ales) dar și din domeniile similare politehnice, pentru modelarea geometrică asistată de calculator) *spre manuale practice, fundamentate teoretic*; ele cuprind sute de subprograme și programe scrise în FORTRAN (limbajul de departe cel mai răspîdit în aplicații grafice industriale), care rezolvă toate problemele esențiale ale *reprezentărilor geometrice, în 2D și 3D*.

Adîncind acest *nucleu al aplicațiilor grafice, din punctul de vedere al utilizatorului proiectant*, prin numeroase exemple și aplicații concrete, finalizate în programe sursă, amply explicate și comentate, autorii și editura (care au realizat din programele sursă ale cărții și un *produs program* de reprezentări geometrice, înregistrat, după testări, în Biblioteca Națională de Programe (BNP) a ITCI din MIET), nu și-au extins preocupările spre celelalte domenii menționate ale graficii interactive asistate, ce interesează îndeosebi pe producătorii de tehnică de calcul și pe informaticienii de sisteme și stații grafice.

Acestea, însă, sînt acoperite de alte cărți și articole apărute sau în curs de apariție la Editura Tehnică și la alte edituri, cum sînt — mai ales — cele privind echipamentele cu facilități grafice și cele privind programarea aplicațiilor grafice (de ex. în BASIC, pe calculatoare personal-profesionale), ce apar în 1989—90.

Felicîtînd atât autorii cît și redacția de informatică-tehnică de calcul a Editurii Tehnice pentru prezenta realizare — ce intersectează preocupările industriei și cercetării-proiectării de profil — recomandăm tuturor celor care produc și utilizează sisteme grafice să consulte și să aplice critic cunoștințele de valoare din această lucrare.

ing. NICOLAE VAIDESCU
Ministrul Industriei Electrotehnice

PREFAȚĂ

Cartea constituie un instrument concret de lucru pentru toți cei ce sînt interesați în cunoașterea sau în aprofundarea problemelor legate de grafica pe calculator, în domeniul modelării geometrice.

Totalitatea programelor sursă din carte alcătuiește un produs-program, care a fost testat la Institutul de Tehnică de Calcul și Informatică și catalogat, în 1988—1989, în Biblioteca Națională de Programe.

Dacă utilizatorii consideră că un anumit program nu îi satisface, ei pot modifica sursa după dorință. Astfel, posibilitățile de îmbunătățire și lărgire a programelor sînt practic nelimitate.

În definitiv, considerăm că acest fapt reprezintă un punct forte al lucrării și constituie una dintre ideile călăuzitoare ale noastre în elaborarea sa. Este foarte important — aproape esențial — să ai în mîină un program sursă și o aplicație de la care să poți porni în studiul propus. Acest fapt îl cunoaștem din propria experiență. Problema limitelor și a performanțelor nu reprezintă ceva acut pentru lucrare, deoarece doar cîțiva dintre algoritmi ar necesita probleme de viteză de calcul și de limitare de memorie, deși pentru acești algoritmi limitele reies din lucrare, vezi APIS, ALPLA, HIDE, ARBORE etc., fiind specificate la momentul respectiv.

Programele principale precum și procedurile apelate în aceste programe sînt scrise în limbajele FORTRAN IV și FORTRAN 77, iar testările respective precum și elaborarea definitivă au fost făcute pe calculatoarele românești INDEPENDENT, CORAL și FELIX C. Vizualizarea desenelor a fost făcută pe displayul grafic românesc DAF 2020, iar desenele (majoritatea) au fost executate pe masa de desen DIGIGRAF 1712, folosind biblioteca software omonimă. Alte desene au fost executate pe imprimanta grafică / în general copiile de pe displayul grafic /.

Precizăm faptul că rutinele cuprinse în Biblioteca grafică DIGIGRAF urmăresc întocmai setul de rutine prezent în programul de firmă care însoțește masa de desen DIGIGRAF 1712. În prezent această masă de desen este cea mai larg utilizată în țara noastră. Introducerea în lucrare a Bibliotecii grafice DIGIGRAF 1712 are scopul de a da posibilitatea utilizării un instrument de interfațare cu mese de desen DIGIGRAF 1712 sau cu imprimanta grafică. Pornind de la aceste fapte precizăm că toți utilizatorii își pot înlocui, pentru partea de desen, apelurile la biblioteca grafică cu sistemele grafice pe care le au la dispoziție pe propriile lor calculatoare.

Menționăm și faptul că problema standardelor grafice dezvoltată în ultimul timp nu a făcut obiectul nici unui capitol al lucrării.

Precizăm, de asemenea, faptul că utilizarea mai multor compilatoare și rularea programelor pe diverse calculatoare nu reprezintă un impediment, modificarea surselor programelor de la un compilator la altul fiind o problemă minoră pentru orice cunoscător al limbajelor FORTRAN.

Cartea, care se bazează atît pe cercetări și realizări originale cît și pe sintetizarea unui bogat material bibliografic a fost îmbunătățită în tot cursul procesului îndelung de editare, fiind prezentată cititorilor structurată în

două volume, cu 10 capitole — cuprinzând teoria reprezentărilor geometrice, sute de algoritmi, subprograme și programe, zeci de aplicații esențiale —, 5 anexe consistente și indicații bibliografice de referință.

Capitolul I — conține Biblioteca software pentru funcții grafice DIGIGRAF 1712, formate din 51 subprograme, cu ajutorul cărora pot fi realizate cele mai sofisticate reprezentări geometrice în 2 D, cât și în 3 D.

De asemenea sînt prezentate o serie de programe test, pentru familiarizarea utilizatorilor.

Capitolul II — conține studiul reprezentărilor geometrice automate în geometria descriptivă. Sînt, astfel, prezentate programele construcțiilor geometrice referitoare în spațiu la punct, dreaptă și plan, precum și programele de desen, împreună cu epurele respective, desenate automat. Sînt tratate, de asemenea, prin particularizarea bazelor matematice, programele construcțiilor geometrice în plan.

Capitolul III — cuprinde programele și subprogramele necesare efectuării secțiunilor plane în suprafețele poliedrale în triplă proiectare ortogonală, cu aplicații posibile în desenul tehnic, arhitectural și de construcții de mașini. Acest capitol conține, de asemenea, programele și subprogramele necesare efectuării intersecției dintre două poliedre convexe în triplă proiectie ortogonală.

Capitolul IV — conține programele secțiunilor plane în sferă și în suprafețele conice sau cilindrice, în triplă proiectie ortogonală, precum și programele necesare reprezentării suprafețelor de rotație generale (sferă, tor, hiperboloid, suprafețe definite parametric etc.). Sînt prezentate, în continuare, programele necesare reprezentării suprafețelor de translație generale, precum și programe de intersecții mixte de suprafețe (sfere cu paraboloidi hiperbolici etc.).

Capitolul V — cuprinde studiul generalizării reprezentării spațiului tridimensional pe un spațiu bidimensional. Astfel, este generalizată metoda coordonatelor perspective pe un tablou înclinat oarecare și este studiată proiectarea centrală sau paralelă a spațiului S^3 pe un tablou plan. Sînt prezentate programele pentru reprezentarea perspectivă a curbilor și suprafețelor exprimate explicit sau parametric, precum și generalități asupra vizualizării obiectelor spațiale, transformările în S^2 și S^3 , decuparea tridimensională a imaginii etc.

Capitolul VI — conține generalități asupra reprezentării obiectelor spațiale, studiul „liniilor și suprafețelor ascunse”, studiul testelor de interioritate, clasificarea algoritmilor de determinare a suprafețelor ascunse (Appel, Galimberti, Montanari, Warnok, Watkins, Loutrel, Roberts, Schumacker, Encarnação, Romney, Bouknight), precum și descrierea algoritmilor Appel, Encarnação, Warnock și Watkins.

Capitolul VII — cuprinde prezentarea generală a algoritmilor pentru rezolvarea problemei liniilor ascunse (ALPLA) și a suprafețelor invizibile (APIS), precum și programele și subprogramele necesare efectuării reprezentărilor geometrice automate în spațiul tridimensional pentru poliedre convexe și poliedre concave (neconvexe) sau pentru poliedre cu goluri sau corpuri deschise. Aceste programe realizează, de asemenea, intersecția dintre poliedre convexe sau neconvexe cu eliminarea liniilor ascunse sau trasarea lor diferențiată. Sînt prezentate aplicațiile rezolvate prin programele ALPLA și APIS.

Capitolul VIII — programele pentru reprezentarea spațială, grafică și vizuală a curbilor și suprafețelor, algoritmi Williamson și Wright pentru vizualizarea funcțiilor de z variabile, rețele de curbe pe suprafață, cu numeroase

aplicații de reprezentare a curbelor și suprafețelor în perspectivă (cu studiul porțiunilor ascunse). Este tratată, în continuare, interpolarea bivariată și montarea suprafețelor netede, bazată pe proceduri locale, împreună cu programele pentru vizualizarea acestor suprafețe. Este realizată trasarea și scrierea curbelor de nivel cu evitarea anumitor restricții, precum și legătura dintre interpolarea bivariată și vizualizarea perspectivă a funcțiilor de 2 variabile.

Capitolul IX — se ocupă de interpolarea și aproximarea curbelor în grafica pe calculator. Astfel, sînt tratate interpolările Hermite, Largange, Achimov, Ferguson, funcția și curbele Spline, curbele Bézier, curbele Bernstein-Bézier, curbele Spline cubice, funcția și curbele B-Spline, interpolarea cu B-Spline. Sînt prezentate numeroase exerciții și probleme rezolvate, cu programele și grafica respectivă.

Capitolul X — se ocupă, în notație matriceală, de problemele generale de determinare și de construcție a suprafețelor în grafica pe calculator. Sînt, astfel, tratate suprafețele riglate, curbele de frontieră, suprafețele COONS, modelarea suprafețelor, suprafețele plăci biliniare și bicubice, suprafețele-plăci generale de diverse tipuri, suprafețele Bézier, interpolarea prin suprafețe Bézier, alipirea netedă a suprafețelor și cuplarea netedă de colț sau în serie a șeticelor de suprafață. Sînt prezentate programele pentru construcția perspectivă a suprafețelor Bézier și COONS și pentru construcția perspectivă a celorlalte tipuri de suprafețe.

Anexa A cuprinde un studiu sistematic al formulelor fundamentale necesare reprezentărilor geometrice în grafica pe calculator.

Anexa B cuprinde programul HAȘURA utilizat pe mesele ARISTO, cum și programul pentru efectuarea produsului dintre 2 matrice.

Anexa C cuprinde programe pentru realizarea și desenarea modulară automată a planurilor de arhitectură și construcții, ș.a.

Anexa D — conține programul pentru desenarea automată a unei piese parametrizate din domeniul construcțiilor de mașini, ca prim pas pentru realizarea desenului automat al unui ansamblu.

Anexa E — conține programe pentru simularea automată a unor curbe și suprafețe în grafica artistică.

Editarea acestor volume de către Editura Tehnică va fi, în mod cert, utilă unui număr mare de cititori, ingineri, informaticieni, arhitecți, cadre didactice și studenți din cadrul institutelor de profil, tuturor celor implicați în dezvoltarea electronicii și informaticii — domenii de maximă importanță pentru progresul tehnico-științific.

AUTORII

DESPRE CARTE

Lucrarea se include într-un program mult mai larg al Editurii Tehnice privind „grafica pe calculator“, adresat deopotrivă specialiștilor în tehnică de calcul — cercetători și producători — și utilizatorilor — proiectanți și realizatori de aplicații informatice — tuturor celor care, în învățămînt-cercetare-producție, folosesc sau vor folosi calculatoare cu terminale grafice — display-uri, plote, imprimante grafice —, cu deosebire în proiectarea și fabricația asistată. Din acest program menționăm „Grafica interactivă și prelucrarea imaginilor“ de V. Baltac și colectivul ITCI, cum și modulele curente din seria „Automatică, management, calculatoare (AMC)“. Cartea de față, oferită de autori ca urmare a unor preocupări inițiate în învățămîntul și cercetarea din construcții-arhitectură, extinse ulterior la aplicații în construcții de mașini, își propune — și atinge — un obiectiv limitat, cel de a constitui o colecție consistentă de algoritmi și programe destinate modelării geometrice în două și trei dimensiuni, fundamentată științific pe cunoștințe solide de reprezentări geometrice, cu aplicații preferențiale din arhitectură-construcții, fără a neglija și domeniul general al construcțiilor de mașini. După o activitate intensă de îmbunătățire în procesul editorial îndelung, programele sursă FORTRAN pentru reprezentări geometrice pe mese de desen și imprimante grafice au fost înregistrate — la propunerea redacției de informatică și tehnică de calcul și a autorilor — în Biblioteca Națională de Programe, după noi experimentări și testări în cadrul Institutului de Tehnică de Calcul și Informatică și al MIET și al consultării unor recenzenti.

Astfel concepută și focalizată cartea nu are în vedere o reflectare a realizărilor din grafica interactivă, din punctul de vedere al producătorilor de echipamente recente de grafică interactivă, nici al proiectanților de aplicații și pachete sofisticate, neatacînd frontal domeniul „fierbinte“ al standardelor grafice — de independență față de dispozitive și de aplicații — și nici cele ale portabilității desenelor și programelor sau al performanțelor absolute ale programelor elaborate.

DESPRE AUTORI

AURELIAN TĂNAȘESCU, n. 1926, Făgăraș, licențiat în matematici al Universității din București, în 1949, arhitect diplomat al Institutului de Arhitectură din București în 1953, și doctor în arhitectură al Institutului de Arhitectură „Ion Mincu“ din București în 1970. Activitate didactică intensă din 1949, devenind din 1955 șef de lucrări la Institutul Politehnic București și conferențiar la Institutul de Arhitectură „Ion Mincu“ din București. Autor al numeroase cărți din domeniul geometriei descriptive, perspectivei și axonometriei, tipărite începînd cu 1962. A elaborat peste 150 de lucrări științifice, tipărite în țară și în străinătate. A participat la numeroase congrese și sesiuni științifice în străinătate. Specializări în Olanda și în Cehoslovacia privind proiectarea asistată de calculator, precum și implementarea graficei pe calculator în domeniile sale de specialitate. În prezent coordonator al grupei de grafică pe calculator din colectivul de proiectare asistată de calculator al Institutului de studii și proiectări pentru îmbunătățiri funciare din București.

CONSTANTINESCU RADU, n. 1956, la București, inginer diplomat al Facultății de Automatică a Institutului Politehnic din București în 1981. A lucrat, din 1981, la Centrul de calcul al Institutului de studii și proiectări pentru îmbunătățiri funciare din București, ocupîndu-se de probleme legate de stocarea și prelucrarea imaginilor digitale, iar în ultimii ani de studiu și implementarea unor algoritmi din domeniul graficei pe calculator, în parte prezentați în această lucrare. A participat la diverse simpozioane și sesiuni de comunicări științifice din țară și străinătate. Din 1989 lucrează la Institutul de Tehnică de Calcul și Informatică.

IOAN D. MARINESCU, absolvent al Institutului Politehnic București, Facultatea Tehnologie Construcției de Mașini, promoția 1976 și asistent universitar, din 1980, în cadrul catedrei Geometrie Descriptivă și Desen. Predă cursurile: Geometrie Descriptivă și Reprezentări Geometrice în Proiectarea Asistată de Calculator. A publicat numeroase lucrări științifice în țară și în străinătate. Din 1981 este membru al Comitetului de Conducere al Fundației Internaționale de Cercetări în Producție și din 1983 este membru al Societății Americane a Inginerilor Mecanici.

BUSUIOC LILIANA, n. 1951, la Ploiești, licențiată a facultății de matematică-mecanică a Universității din București în 1975. Din acel an lucrează la Institutul de Studii și proiectări pentru îmbunătățiri funciare din București, în colectivul Cellatron, apoi în colectivul de proiectare și modelare matematică asistată de calculator. A elaborat numeroase programe privind eforturi și deformații în terenuri de fundare, analiza stabilității taluzelor; corelații statistice ale datelor experimentale, baza de date privind evidența studiilor geotehnice, un pachet de subprograme care fundamentează, în sens geometric, proiectarea asistată de calculator.

CUPRINS GENERAL

VOLUMUL 1

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

VOLUMUL 2

Cuprins general	5
Cuprinsul Volumului 2	6
<i>Capitolul VII</i>	
Algoritmi pentru rezolvarea problemelor liniilor ascunse (ALPLA) și a suprafețelor invizibile (APIS)	9
<i>Capitolul VIII</i>	
Reprezentarea spațială grafică și vizuală a curbilor și suprafețelor	91
<i>Capitolul IX</i>	
Curbele în grafica pe calculator	145
<i>Capitolul X</i>	555
Suprafețele în grafica pe calculator	187
ANEXE (II)	
ANEXA C — Programul PLANĂR pentru desenarea modulară automată a planurilor de arhitectură și construcții	259
— Programe pentru trasarea conturilor care îmbracă cu grosimi variabile un graf dat prin noduri și legăturile dintre ele	267
ANEXA D — Programul PIESA pentru desenarea automată a unei piese parametrizate din domeniul construcțiilor de mașini	282
ANEXA E — Program pentru simularea automată a unor curbe și suprafețe în grafica artistică	286
BIBLIOGRAFIE	294

CUPRINS

Cuvint înainte	5
Prefața autorilor	7
Despre carte și autori	10
Cuprins general	11
Cuprins volumul 1	12

Capitolul 1

BIBLIOTECA SOFTWARE PENTRU
FUNCTII DIGIGRAF 1712 ÎN 2D ȘI 3D

A. SPAȚIUL BIDIMENSIONAL (2D)	15
1.1. Subprogramul INI	16
1.2. Subprogramul BEG	17
1.3. Subprogramul SCA	19
1.4. Subprogramul TRA	20
1.5. Subprogramul SSCA	20
1.6. Subprogramul DEG	21
1.7. Subprogramul ROT	21
1.8. Subprogramul ARC	22
1.9. Subprogramul ANG	22
1.10. Subprogramul RCT	22
1.11. Subprogramul TRXY	23
1.12. Subprogramul SPG	23
1.13. Subprogramul SPR	24
1.14. Subprogramul TLA	24
1.15. Subprogramul EOF	24
1.16. Subprogramul SPD	25
1.17. Subprogramul SSPD	26
1.18. Subprogramul BLH	26
1.19. Subprogramul BLD	27
1.20. Subprogramul SAV	29
1.21. Subprogramul PLOT	29
1.22. Subprogramul RET	30
1.23. Subprogramul LIN	31
1.24. Subprogramul NOT	32
1.25. Subprogramul PLG	32
1.26. Subprogramul SRA	33
1.27. Subprogramul BOW	35
1.28. Subprogramul AXY	35
1.29. Subprogramul CSC	36
1.30. Subprogramul CPP	37
1.31. Subprogramul CERC	38
1.32. Subprogramul CIS	39
1.33. Subprogramul CIT	40
1.34. Subprogramul IPO	40
1.35. Subprogramul IPR	41
1.36. Subprogramul IPC	42
1.37. Subprogramul TEX	43
1.38. Subprogramul POS	47
1.39. Subprogramul NUM	48
1.40. a Lista erorilor returnate de sub- programele grafice	
1.40 b Aplicații. Programe test	48

B. SPAȚIUL TRIDIMENSIONAL
(3D)

1.41. Subprogramul INISP	60
1.42. Subprogramul SETV	60
1.43. Subprogramul SETC	60
1.44. Subprogramul SETD	61
1.45. Subprogramul SETDIM	61
1.46. Subprogramul SETS	61
1.47. Subprogramul PLANE	62
1.48. Subprogramul NEWC	62
1.49. Subprogramul PRO	62
1.50. Subprogramul CLIPS	63
1.51. Program principa aplicativ pen- tru utilizarea softului de rutine grafice în spațiul 3D	64
1.52. Aplicații	65

Capitolul II

REPREZENTĂRI AUTOMATE ÎN GEOME-
TRIA DESCRIPTIVĂ PUNCTUL. DREAPTA-
PLANUL. PROBLEME DE PARALELISM,
INCIDENTĂ ȘI PERPENDICULARITATE.
SUBPROGRAME GEOMETRICE ȘI DE
DESEN

2.1. Geometrie descriptivă asistată de calculator (I)	69
2.1.1. Subprogramul REORPU ..	70
2.1.2. Subprogramul REORDR ..	71
2.1.3. Subprogramul CLASDR (Exemple)	72
2.1.4. Subprogramul PCTDR	74
2.1.5. Subprogramul DPCTDR	74
2.1.6. Subprogramul PL3P	75
2.1.7. Subprogramul CLASPL (Exemple)	77
2.1.8. Subprogramul PPADD	80
2.1.9. Subprogramul INPLPL	82
2.1.10. Subprogramul INDRPL ..	85
2.1.11. Subprogramul PPPDDR ..	87
2.2. Geometrie descriptivă asistată de calculator (II)	89
2.2.1. Subprogramul DPRPED	89
2.2.2. Subprogramul DPRPOD ..	92
2.2.3. Subprogramul PECO	94

Capitolul III

SECȚIUNI PLANE ÎN POLIEDRE ȘI
INTERSECȚIA DINTRE DOUĂ POLIEDRE

3.1. Secțiuni plane în poliedre	97
3.1.1. Generalități	97

3.1.2. Programul SECPOS pentru determinarea secțiunilor plane în suprafețele poliedrale	97
3.1.3. Subprogramul CALCUL	102
3.1.4. Subprogramul LIMITE	103
3.1.5. Subprogramul BAZA	104
3.1.6. Subprogramul INLAT	105
3.1.7. Subprogramul CLSDR	107
3.1.8. Subprogramul INTMU	107
3.2. Programele de desen autmat sau de vizualizare pentru poliedre și pentru poligonul de secțiune..	108
3.2.1. Generalități	108
3.2.2. Subprogramul DRIP	109
3.2.3. Subprogramul DRFP	109
3.2.4. Subprogramul POLIGO(N)	110
3.2.5. Subprogramul PIR(AMIDA)	112
3.2.6. Subprogramul CUB(PRISMA)	113
3.3. Aplicație la secțiuni plane în poli- edre	114
3.3.1. Aplicația 1	114
3.3.2. Aplicația 2	115
3.3.3. Aplicația 3	116
3.3.4. Aplicația 4	118
3.4. Intersecția dintre două poliedre	
3.4.1. Subprogramul INTPOL	119
3.4.2. Subprogramul PCTINT	119
3.4.3. Subprogramul UPIP	122
3.4.4. Aplicație la programul UPIP	125
3.4.5. Descrierea programului INTPOL	127
3.4.6. Program principal INTPOL (Listare)	128
3.4.7. Subrutina CALCUL	130
3.4.8. Subrutina PARAM	131
3.4.9. Subrutina PCTINT (Listare)	132
3.4.10. Subrutina LIMITE	135
3.4.11. Program principal desen INTPOL (Listare)	136
3.5. Aplicații rezolvate prin pro- gramul INTPOL	139
3.5.1. Intersecția dintre doi tetra- edri regulați	139
3.5.2. Intersecția dintre un tetra- edru și un octaedru	140
3.5.3. Intersecția dintre doi octaedri	140
3.5.4. Idem. Date schimbate	140
3.6. Extinderea programului INTPOL	142
3.6.1. Programul INNPOL	142
3.6.2. Subprogramul MATRICE ..	147
3.6.3. Subprogramul ORD	148
3.6.4. Subprogramul PARAM	148
3.6.5. Subprogramul CALCUL ..	144
3.6.6. Subprogramul LIMITE	151
3.6.7. Subprogramul POLIG	152
3.6.8. Subprogramul PCTINT	153
3.6.9. Aplicație pentru programul INNPOL. Intersecția dintre două cuburi P1 și P2	156

Capitolul IV

CONICE. CUADRICE, SUPRAFETE DE ROTAȚIE. SUPRAFETE DE TRANSLAȚIE. REPREZENTARE SECȚIUNI PLANE. INTERSECȚII MIXTE DE SUPRAFETE	
4.1. Secțiuni plane în sferă. Intersec- ția dintre două sfere	159
4.1.1. Generalități (baza matema- tică)	159
4.1.2. Plotarea elipselor	160
4.1.3. Programul SFERA	169
4.1.4. Intersecția dintre două sfere. Programul principal	167
4.1.5. Reprezentarea sferei în izo- metrie	171
4.2. Secțiuni plane în suprafețele conice	172
4.2.1. Secțiunea eliptică în conul circular oblic	172
4.2.2. Aplicație	176
4.2.3. Programul principal CON ..	178
4.2.4. Secțiuni plane punctuale în suprafețele conice sau cilin- dric	179
4.2.5. Programul INTCIL (sau INTCON)	181
4.3. Programul pentru construcția perspectivă a suprafețelor de ro- tație generale	186
4.3.1. Baza matematică	186
4.3.2. Programul ROTIZO, Subpro- gramele PRIZOM și PRA- DIM. Aplicații	192
4.4. Programul pentru reprezentarea perspectivă a hiperboloidului de rotație cu o pînă	195
4.4.1. Baza matematică	195
4.4.2. Programul HIP1P	196
4.5. Programul pentru construcția perspectivă a suprafețelor de translație (I)	198
4.5.1. Baza matematică	198
4.5.2. Programul TRANS pentru gene- rarea modelului suprafeței de translație	198
4.6. Programul pentru construcția perspectivă a suprafețelor de translație (II) speciale care trec prin frontierele domeniului	200
4.6.1. Baza matematică	200
4.6.2. Programul TRAN2	201
4.7. Program pentru determinarea intersecției dintre o sferă și un paraboloid hiperbolic	202
4.7.1. Baza matematică	202
4.7.2. Programul HIPSF	205
4.8. Program SELI pentru construcția și desenarea anumitor suprafețe de tip elicoidal. Subprogramul CARPL	208

4.9. Programul INSFCI. Intersecția dintre o sferă și un cilindru în axonometrie. Subprogramele CERC2 și PRPLOT	210
--	-----

Capitolul V

GENERALIZAREA REPREZENTĂRII SPAȚIULUI S^3 TRIDIMENSIONAL PE UN SPAȚIU BIDIMENSIONAL S^2

5.1. Generalizarea metodei coordonatelor perspective pe un tablou înclinat oarecare	213
5.1.1. Determinarea coordonatelor perspective absolute.....	213
5.1.2. Determinarea coordonatelor perspective relative	214
5.1.3. Determinarea cosinuşilor directori $\alpha_i, \beta_i, \gamma_i$	215
5.1.4. Observația 1	217
5.1.5. Observația 2	218
5.1.6. Observația 3	219
5.2. Proiecția centrală sau paralelă a spațiului S^3 pe planul P	219
5.2.1. Proiecția centrală	219
5.2.2. Proiecția paralelă	220
5.2.3. Perspectiva pe tablou înclinat	221
5.2.4. Perspectiva pe tablou vertical (frontal).....	223
5.2.5. Perspectiva pe tablou orizontal	225
5.3. Formularea problemei trasării perspectivei	227
5.3.1. Notații cu caracter general pentru întocmirea programului pe calculator	227
5.3.2. Program principal: PROP, CENT sau IZOM pentru perspectiva paralelă oarecare, centrală sau izometrică	228
5.4. Reprezentarea perspectivă pe imprimantă a curbelor și suprafețelor pentru o primă aproximare	229
5.4.1. Subprogramul CALCF	229
5.4.2. Subprogramul TRASXY ..	230
5.4.3. Subprogramul CALCFU ..	232
5.4.4. Aplicații. Suprafețe desenate pe imprimanta obișnuită....	232
5.5. Reprezentarea perspectivă la plotter (display) sau pe imprimanta grafică a curbelor și suprafețelor	235
5.5.1. Programul interactiv pentru cazul în care suprafața este exprimată explicit	235
5.5.2. Aplicații	238
5.5.3. Programul interactiv pentru cazul în care suprafața este exprimată parametric	241
5.5.4. Programul interactiv pentru cazul în care curba este exprimată parametric	243

5.5.5. Aplicații	248
5.6. Generalități asupra vizualizării obiectelor spațiale	248
5.6.1. Influența echipamentelor electronice utilizate	248
5.7. Transformări în S^2 și S^3	249
5.7.1. Coordonate omogene	249
5.7.2. Puncte și drepte (forma implicită)	250
5.7.3. Transformări în S^2	251
5.7.4. Reprezentarea conicelor. Forma implicită	251
5.7.5. Ecuațiile parametrice	252
5.7.6. Transformări în S^3	253
5.7.7. Proiecția ortogonală.....	255
5.8. Proiecția perspectivă	256
5.8.1. Sistemul de coordonate al ochiului	256
5.8.2. Decuparea tridimensională a imaginii	258
5.8.3. Aplicații. Vederea perspectivă a unui cub	260

Capitolul VI

REPREZENTAREA OBIECTELOR SPAȚIALE

6.1. Generalități asupra reprezentării obiectelor spațiale.....	263
6.1.1. Tehnici de simulare.....	263
6.1.2. Descrierea obiectelor	264
6.2. Problema „suprafețelor ascunse“	266
6.2.1. O descriere a problemei....	266
6.2.2. Calcule geometrice	267
6.3. Teste de interioritate	271
6.3.1. Teste prin calcularea sumei unghiurilor	272
6.3.2. Test prin calcularea numărului de intersecții	272
6.4. Clasificări ale algoritmilor de suprafețe ascunse	273
6.5. Descrierea citorva algoritmi....	275
6.5.1. Metoda invizibilității cantitative a lui APPEL	275
6.5.2. Metoda de prioritate a lui ENCARNĂO	277
6.5.3. Algoritmul lui WARNOCK ..	279
6.5.4. Algoritmul lui WATKINS ..	279
6.5.5. Metoda rețelei de explorare a lui ENCARNĂO	280

ANEXE (I)

ANEXA A — Formule fundamentale necesare înțelegerii reprezentărilor geometrice în grafica pe calculator.....	282
ANEXA B — Subprogramul HASURA utilizat pe mesele de desen de tip ARISTO.....	307
Programul pentru produsul a două matrice	310

A. SPAȚIUL BIDIMENSIONAL (2D)

Printre dispozitivele periferice intrate în dotarea unităților informatice din țara noastră în ultimii ani, se numără și mesele de desen **DIGIGRAF** de producție cehoslovacă, produse de firma **KOVO**. Existent în două variante constructive, dispozitivul se remarcă prin bune caracteristici de viteză de trasare și precizie (0.01 mm).

În continuare referirile se vor face la ambele variante constructive, care diferă numai prin dimensiunile suprafeței utile de desen.

Din punct de vedere al utilizatorului mesei de desen, sînt importante modul de transmitere a datelor către dispozitiv, codificarea lor cît și comenzile pe care le poate programa.

Funcțiile de bază pe care le poate îndeplini **DIGIGRAF** sînt următoarele:

a. trasarea unui segment de dreaptă din poziția curentă a capului de trasare pînă într-un punct specificat.

b. poziționarea capului de trasare într-un punct specificat.

c. trasarea unui arc de cerc începînd din poziția curentă a capului de trasare pînă într-un punct specificat. Pentru unicitatea trasării se specifică în plus un punct drept centru al cercului și un sens de trasare (arcul cel mare, respectiv arcul cel mic).

d. scrierea unui text cu o înclinare și o mărime a caracterului definite de utilizator. Poziționarea textului se face cu ajutorul unui punct specificat și a unui

indicator de centrare a textului, care poate lua 3 valori.

0 — marginea din stînga a textului

1 — mijlocul textului

2 — marginea din dreapta a textului

e. specificarea capului de trasare la care se vor referi comenzile de desen ulterioare (există variante constructive cu 2 sau 4 capete de trasare).

f. alegerea unui tip de linie cu care se vor efectua trasările următoare, dintr-o tabelă predefinită.

g. modificarea tabelii de tipuri de linie

h. modificarea vitezei și accelerației pentru comenzile de trasare și sau poziționare

Transmiterea comenzilor către dispozitiv se realizează în format hexazecimal codificat *ASCII*.

În descrierea fiecărei rutine care transmite efectiv comenzi mesei de desen, s-a specificat forma hexa *ASCII* a comenzii respective.

Biblioteca de rutine prezentată în acest subcapitol este alcătuită dintr-un set de proceduri pentru lucrul în spațiul 2-D. Rutinele din bibliotecă pot

* Programele au fost elaborate în colaborare cu Ing. Anca Dumitrescu

fi structurate în mai multe categorii, în funcție de acțiunile pe care le realizează:

a. rutine care implementează comenzi pentru masă

(PLOT, CERC, TEX, TLA, BLH, BLD, SPD, SSD, EOF)

b. rutine necesare realizării transformărilor geometrice uzuale în spațiul 2-D. Aceste transformări sînt: translații, rotații, scalări, ogîndiri, salvări și restaurări de context geometric.

(BEG, TRA, ANG, ROT, SCA, SSCA, SPR SPG, SAV, RET)

c. rutine de desen cu scopuri generale pentru desenarea de dreptunghiuri, poligoane, interpolări de curbe, hașurări, racordări, diverse tipuri de definire a cercurilor, trasări de axe.

(LIN, ROT, PLG, IPO, IPR, IPC, SRA, BOW, NOT, CSC, CPP, CIT, CIS, AXI)

d. alte rutine apelate de procedurile bibliotecii

(TRXY, INV2, INV4, POL4, POL3, SCHAR, CHAR)

e. rutina de editare de numere (NUM)

Toate rutinele sînt scrise în limbajul **FORTRAN 77** pentru calculatoare din seria *I100, I102F, CORAL*.

Pentru acei utilizatori care doresc să implementeze această bibliotecă de rutine pe alte calculatoare decît cele menționate, trebuie verificate condițiile de compatibilitate cu compilatoarele **FORTRAN** de pe respectivele echipamente.

1.1. Subprogramul INI

— DENUMIRE

INI

— FUNCȚIE

Subprogramul **INI** inițiază masa de desen, deci stabilește legătura dintre un număr logic ales arbitrar și dispozitivul fizic de ieșire pe care va fi scos fișierul de desen. Din acest motiv subprogramul **INI** trebuie să fie apelat înaintea tuturor celorlalte apeluri de subprograme din biblioteca grafică.

— APEL

CALL INI (NL)

— PARAMETRII

NL — numărul logic al perifericului de editare în cadrul sistemului de operare generat.

Parametrul **NL** se definește prin **INTEGER**.

— OBSERVAȚII

Fișierul de desen poate fi scos direct pe perforatorul de bandă sau poate fi trecut pe disc urmînd ca în final acesta să fie copiat pe bandă perforată cu ajutorul utilitarului **PIP**.

Dacă se urmărește obținerea desenului pe imprimanta grafică atunci fișierul este obligatoriu să se creeze pe disc. Specificarea dispozitivului de ieșire se face înaintea apelului subprogramului **INI**, cu ajutorul subprogramului **ASSIGN** din biblioteca **FORTRAN**. Dacă se asigîtează un număr logic mai mare ca 5, fișierul de desen va putea fi folosit la imprimanta grafică, altfel fișierul rezultat va avea formatul compatibil mesei de desen.

Exemplu:

Masa de desen

CALL ASSIGN (1, 'PP:')

CALL INI (1)

 sau

CALL ASSIGN (1, 'PLOT.DAT')

CALL INI (1)

A. Spațiul bidimensional (2D)

Printre dispozitivele periferice intrate în dotarea unităților informatice din țara noastră în ultimii ani, se numără și mesele de desen **DIGIGRAF** de producție cehoslovacă, produse de firma **KOWO**. Existent în două variante constructive, dispozitivul se remarcă prin bune caracteristici de viteză de trasare și precizie (0.01 mm).

În continuare referirile se vor face la ambele variante constructive, care diferă numai prin dimensiunile suprafeței utile de desen.

Din punct de vedere al utilizatorului mesei de desen, sînt importante modul de transmisie a datelor către dispozitiv, codificarea lor cît și comenzile pe care le poate programa.

Funcțiile de bază pe care le poate îndeplini **DIGIGRAF** sînt următoarele:

a. trasarea unui segment de dreaptă din poziția curentă a capului de trasare pînă într-un punct specificat.

b. poziționarea capului de trasare într-un punct specificat.

c. trasarea unui arc de cerc începînd din poziția curentă a capului de trasare pînă într-un punct specificat. Pentru unicitatea trasării se specifică în plus un punct drept centru al cercului și un sens de trasare (arcul cel mare, respectiv arcul cel mic).

d. scrierea unui text cu o înclinare și o mărime a caracterului definite de utilizator. Poziționarea textului se face cu ajutorul unui punct specificat și a unui

indicator de centrare a textului, care poate lua 3 valori

0 — marginea din stînga a textului

1 — mijlocul textului

2 — marginea din dreapta a textului

e. specificarea capului de trasare la care se vor referi comenzile de desen ulterioare (există variante constructive cu 2 sau 4 capete de trasare).

f. alegerea unui tip de linie cu care se vor efectua trasările următoare, dintr-o tabelă predefinită.

g. modificarea tabelii de tipuri de linie

h. modificarea vitezei și accelerației pentru comenzile de trasare și/sau poziționare

Transmiterea comenzilor către dispozitiv se realizează în format hexazecimal codificat *ASCII*.

În descrierea fiecărei rutine care transmite efectiv comenzi mesei de desen s-a specificat forma hexa *ASCII* a comenzii respective.

Biblioteca de rutine prezentată în acest capitol este alcătuită dintr-un set de proceduri pentru lucrul în spațiul 2-D. Rutinele din bibliotecă pot fi structurate în mai multe categorii, în funcție de acțiunile pe care le realizează:

* Programele au fost elaborate în colaborare cu ing. Anca Dumitrescu

a. rutine care implementează comenzi pentru masă

(PLOT, CERC, TEX, TLA, BLH, BLD, SPD, SSPD, EOF)

b. rutine necesare realizării transformărilor geometrice uzuale în spațiul 2-D. Aceste transformări sînt: translații, rotații, scalări, ogîndiri, salvări și restaurări de context, geometric.

(BEG, TRA, ANG, ROT, SCA, SSCA, SPR, SPG, SAV, RET)

c. rutine de desen cu scopuri generale pentru desenarea de dreptunghiuri, poligoane, interpolări de curbe, hașurări, racordări, diverse tipuri de definire a cercurilor, trasări de axe.

(LIN, ROT, PLG, IPO, IPR, IPC, SRA, BOW, NOT, CSC, CPP, CIT, CIS, AXI)

d. alte rutine apelate de procedurile bibliotecii

(TRXY, INV2, INV4, POL4, POL3, SCHAR, CHAR)

e. rutina de editare de numere (NUM)

Toate rutinele sînt scrise în limbajul **FORTRAN77** pentru calculatoare din seria *I100*, *I102F*, *CORAL*.

Pentru acei utilizatori care doresc să implementeze această bibliotecă de rutine pe alte calculatoare decît cele menționate, trebuie verificate condițiile de compatibilitate cu compilatoarele **FORTRAN** de pe respectivele echipamente.

1.1. Subprogramul INI

- DENUMIRE

INI

- FUNCȚIE

Subprogramul **INI** inițializează masa de desen, deci stabilește legătura dintre un număr logic ales arbitrar și dispozitivul fizic de ieșire pe care va fi scos fișierul de desen. Din acest motiv subprogramul **INI** trebuie să fie apelat înaintea tuturor celorlalte apeluri de subprograme din biblioteca grafică.

- APEL

CALL INI (NL)

- PARAMETRII

NL — numărul logic al perifericului de editare în cadrul sistemului de operare generat.

Parametrul *NL* se definește prin *INTEGER*.

- OBSERVAȚII

Fișierul de desen poate fi scos direct pe perforatorul de bandă sau poate fi trecut pe disc urmînd ca în final acesta să fie copiat pe bandă perforată cu ajutorul utilitarului **PIP**.

Dacă se urmărește obținerea desenului pe imprimanta grafică atunci fișierul este obligatoriu să se creeze pe disc. Specificarea dispozitivului de ieșire se face înaintea apelului subprogramului **INI**, cu ajutorul subprogramului **ASSIGN** din biblioteca **FORTRAN**. Dacă se asignează un număr logic mai mare ca 5, fișierul de desen va putea fi folosit la imprimanta grafică, altfel fișierul rezultat va avea un format compatibil mesei de desen.

Exemplu:

Masa de desen

CALL ASSIGN (1, 'PP:')

sau

CALL ASSIGN (1, 'PLOT. DAT')

CALL INI (1)

CALL INI (1)

Imprimanta grafică

CALL ASSIGN (5, 'SCAMP.DAT')

CALL INI (5)

În afara asigurării numărului logic, subprogramul **INI** face inițializarea parametrilor folosiți de rutinele de transformare după cum urmează:

ALFA = 0.	echivalent cu următoarea secvență de apeluri:
XR = 1.	CALL ROT (0.)
YR = 1.	CALL SSCA (1. 1.)
DX = 0.	CALL BEG (0. 0.)
DY = 0.	CALL SPR
IOGL = 0	CALL ARC
IRAD = 0	CALL SCHAR (PI/2.)
BET = PI/2	

(Apelul subprogramului **SCHAR** are sens doar pentru lucrul cu imprimanta grafică, sau dacă se dorește folosirea generatorului de caractere soft pentru a folosi caractere înclinate).

De asemenea, subprogramul **INI** reprezintă începutul pentru următoarele subprograme:

SCA (1., 1.)
 TRA (0., 0.)
 ROT (0.)
 TLA (1)

```

SUBROUTINE INI(NI)
COMMON/BCAR/BET
COMMON /HCAR/UNGHT,H,IP07
COMMON /GRI /XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
COMMON /GRAF/XOT,YOT,NI R,XO,YO
PI=3.141529
UNGHI=0.
H=0.
IP07=0
NLU=NL
XR=1.
YR=1.
BET=PI/2.
ALFA=0.
DX=0.
UY=0.
IOGL=0
IRAD=0
IF (NL.GT.5)GOTO 1
RETURN
NLG=NL
RETURN
END
  
```

1.2. Subprogramul BEG

— DENUMIRE
 — FUNCȚIE

BEG

Subprogramul **BEG** definește translația absolută a sistemului de coordonate în punctul (**DX**, **DY**) fără definițiile anterioare făcute de subprogramele **TRA** și **BEG** și fără influența subprogramelor **ROT**; **ANG**; **SCA**; **SSCA**; **SPG**

— APEL
 — PARAMETRII

CALL BEG (DX, DY)

DX — coordonata noii origini a sistemului de coordonate pe axa **X**
DY — coordonata noii origini a sistemului de coordonate pe axa **Y**

Parametri **DX**, **DY** se definesc prin **REAL**

```

SUBROUTINE BEG(DX1,DY1)
COMMON /GRI /XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
DX=DX1
DY=DY1
RETURN
END
  
```

1.3. Subprogramul SCA

- DENUMIRE **SCA**
- FUNCȚIE Subprogramul **SCA** modifică scara desenului pentru axa *OX* sau axa *OY* față de valorile anterior definite (factor de scară relativ)
- APEL **CALL SCA (XR, YR)**
- PARAMETRI *XR* – factorul de scară pentru axa *X*
 YR – factorul de scară pentru axa *Y*
 XR, YR > 0
 XR, YR > 1 – mărire
 XR, YR < 1 – micșorare
 Parametrii *XR, YR* se definesc prin *REAL*
- OBSERVAȚII Noile scări vor fi egale cu produsul dintre vechile valori ale scărilor setate de ultimul apel al subprogramelor **SCA** sau **SSCA** și valoarea parametrului *XR*, respectiv *YR*
- ERORI Subprogramul **INI** stabilește pentru parametrii *XR* și *YR* valoarea, 1.
 RETURNATE Eroare 1 – $XR \leq 0$ sau $YR \leq 0$

```

SUBROUTINE SCA(XR1,YR1)
COMMON /GB1/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
IF(XR1.LT.0.OR.YR1.LT.0) CALL ERR(1)
IF(XR1.EQ.0.OR.YR1.EQ.0) GOTO 1
XR=XR*ABS(XR1)
YR=YR*ABS(YR1)
RETURN
CALL ERR(1)
RETURN
END

```

1.4. Subprogramul TRA

- DENUMIRE **TRA**
- FUNCȚIE Subprogramul **TRA** definește translația relativă a originii sistemului de coordonate cu *DX* pe axa *OX* și respectiv cu *DY* pe axa *OY*
 Vechilor valori ale parametrilor de translație definite cu ajutorul subprogramelor **TRA** sau **BEG** li se adaugă valorile *DX*, respectiv *DY*
- APEL **CALL TRA (DX, DY)**
- PARAMETRI *DX* – deplasarea în direcția axei *X*
 DY – deplasarea în direcția axei *Y*
 Parametrii *DX* și *DY* se definesc prin *REAL*
 Parametrii de translație nu sînt influențați de apelurile anterioare ale subprogramelor **ROT, ANG, SCA, SSCA, SPG**
- OBSERVAȚII Toate coordonatele date ulterior apelului vor fi considerate față de noul sistem traslatat *X'OY'*

```

SUBROUTINE TRA(DX1,DY1)
COMMON /GB1/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
DX=DX+DX1
DY=DY+DY1
RETURN
END

```

1.5. Subprogramul SSCA

- DENUMIRE **SSCA**
- FUNCȚIE Subprogramul **SSCA** stabilește în mod absolut scara desenului pentru axa *OX* respectiv *OY*. Aceasta înseamnă că valorile inițiale *SCAX* și *SCAY* sînt anterior definite

- APEL **CALL SSCA (XR, YR)**
- PARAMETRI *XR* — factorul de scară pentru axa *X*
 YR — factorul de scară pentru axa *Y*
 XR, YR > 0
 XR, YR > 1 — mărire
 XR, YR < 1 — micșorare
 Parametrii *XR* și *YR* se definesc prin *REAL*
- OBSERVAȚII Subprogramul *INI* stabilește pentru parametrii *XR* și *YR* valoarea 1
- ERORI
- RETURNATE Eroare 2 — $XR \leq 0$ sau $YR \leq 0$

```

SUBROUTINE SSCA(XR1,YR1)
COMMON /GBI/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
IF(XR1.LT.0.OR.YR1.LT.0) CALL ERR(2)
IF(XR1.EQ.0.OR.YR1.EQ.0) GOTO 1
XR=ABS(XR1)
YR=ABS(YR1)
RETURN
CALL ERR(2)
RETURN
END

```

1.6. Subprogramul DEG

- DENUMIRE **DEG**
- FUNCȚIE Subprogramul **DEG** asigură ca toate valorile unghiurilor să fie introduse numai în grade pentru toate apelurile subprogramelor ce apar în continuare
- APEL **CALL DEG**
- PARAMETRI Nu are
- OBSERVAȚIE Subprogramul *INI* definește implicit valorile de intrare în radiani
 Subprogramul **DEG** setează în unu valoarea parametrului *IRAD*.

```

SUBROUTINE DEG
COMMON /GBI/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
IRAD=1
RETURN
END
SUBROUTINE ARC
COMMON /GBI/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
IRAD=0
RETURN
END

```

1.7. Subprogramul ROT

- DENUMIRE **ROT**
- FUNCȚIE Subprogramul **ROT** definește rotirea axelor sistemului de coordonate cu unghiul *ALFA*, ceea ce înseamnă că în raport cu rotația anterior definită, între axa *OX* și axa *OY* se păstrează unghiul de 90 grade.
- APEL **CALL ROT (ALFA)**
- PARAMETRI *ALFA* — unghi exprimat în radiani (sau în grade cu care axa *OX*' se rotește față de axa *OX* și axa *OY*' față de *OY* în direcția pozitivă (sens direct trigonometric)
 Parametrul *ALFA* se definește prin *REAL*.
- OBSERVAȚII Subprogramul *INI* stabilește pentru *ALFA* valoarea 0.
 Dacă utilizatorul dorește să execute rotația sistemului de coordonate în grade, este nevoie ca înainte de aplicarea subprogramului **ROT** să se apeleze subprogramul **DEG**.
 Noul unghi de rotație dintre cele două sisteme de coordonate va fi suma dintre unghiul vechi de rotație, setat de apelul anterior al subprogramului **ROT** sau **ANG** și *ALFA*.

```

SUBROUTINE ROT(ALFA1)
COMMON /GHI /XR,YR,ALFA,DX,DY,IOGL,TRAD,NLU,PI
ALFA2=ALFA1
IF (IRAD.EQ.1) ALFA2=ALFA1*PI/180.
ALFA=ALFA+ALFA2
RETURN
END

```

1.8. Subprogramul ARC

- DENUMIRE ARC
- FUNCȚIE Subprogramul ARC asigură ca toate mărimile unghiurilor să fie introduse prin măsura arcului 1 Rad $57^{\circ}17'45''$
 $1^{\circ} = 0,01745329$ Rad
- APEL CALL ARC
- PARAMETRI Nu are
- OBSERVAȚIE Vezi observațiile subprogramului DEG.
 Subprogramul ARC setează la zero valoarea parametrului IRAD.
 În urma acestui apel, toți parametrii de tip unghi vor trebui să fie dați în radiani

1.9. Subprogramul ANG

- DENUMIRE ANG
- FUNCȚIE Subprogramul ANG definește rotația axelor de coordonate cu unghiul ALFA, în mod absolut, indiferent de comenzile anterioare ale subprogramelor ANG sau ROT. Între axele OX și OY rămân 90 grade
- APEL CALL ANG (ALFA)
- PARAMETRI ALFA — unghi, în radiani sau grade cu care se rotește axa OX' față de axa OX sau OY' față de OY.
 Parametrul ALFA se definește prin REAL
- OBSERVAȚII Subprogramul INI stabilește pentru ALFA valoarea 0.
 Tot ce se va desena în urma apelului subprogramului ANG, va fi rotit față de sistemul XOY, cu unghiul ALFA
 Unghiul ALFA trebuie dat în grade sau radiani în concordanță cu ultima valoare setată a parametrului IRAD (după INI, IRAD = 0) deci unghiurile se vor da în radiani
 Pentru a se putea da valori în grade, trebuie apelat în prealabil subprogramul DEG

```

SUBROUTINE ANG(ALFA1)
COMMON /GHI /XR,YR,ALFA,DX,DY,IOGL,TRAD,NLU,PI
ALFA2=ALFA1
IF (IRAD.EQ.1) ALFA2=ALFA1*PI/180.
ALFA=ALFA2
RETURN
END

```

1.10. Subprogramul RCT

- DENUMIRE RCT
- FUNCȚIE Subprogramul RCT desenează un patrulater rectangular, care este definit prin punctul minim și punctul maxim, în raport cu axele OX și OY cu care vor fi paralele laturile sale.
- APEL CALL RCT (X MIN, Y MIN, X MAX, Y MAX)
- PARAMETRI X MIN, Y MIN — definesc coordonatele pe axa X sau Y ale punctului din stînga jos al parametrului.
 X MAX, Y MAX — definesc coordonatele punctului din dreapta sus.

- OBSERVAȚIE Subprogramul RCT este influențat de toate subprogramele de transformare TRA; BEG; ANG; ROT; SCA; SSCA; SPG.

```

SUBROUTINE RCT(XMIN,YMTN,XMAX,YMAX)
CALL PLOT(XMIN,YMIN,0)
CALL PLOT(XMIN,YMAX,1)
CALL PLOT(XMAX,YMAX,1)
CALL PLOT(XMAX,YMIN,1)
CALL PLOT(XMIN,YMIN,1)
RETURN
END

```

1.11. Subprogramul TRXY

- DENUMIRE **TRXY**
- FUNCȚIE Subprogramul TRXY aplică punctului (X, Y) din sistemul de coordonate X'OY' transformarea complexă definită cu ajutorul subprogramelelor ANG; ROT; TRA; BEG; SCA; SSCA; SPR, calculându-i coordonatele (XT, YT) din sistemul inițial XOY
- APEL **CALL TRXY (X, Y, XT, YT)**
- PARAMETRII X, Y — coordonatele punctului
XT, YT — coordonatele transformate ale punctului
Parametrii X, Y, XT, YT se definesc prin REAL
- OBSERVAȚII Relația de calcul a transformării este:

$$XT = DX + X \star XR \star IV \star \cos(\text{ALFA}) - Y \star YR \star \sin(\text{ALFA})$$

$$YT = DY + X \star XR \star IV \star \sin(\text{ALFA}) + Y \star YR \star \cos(\text{ALFA})$$
unde: DX, DY — parametri de translație
XR, YR — scările pe cele 2 axe
ALFA — unghiul de rotație între OX și OX' respectiv OY și OY'
IV = 1 pentru IOGL = 0
— 1 pentru IOGL = 1 parametru de oglindire față de axa X
Subprogramul TRXY este conceput pentru uz intern el fiind apelat de subprogramele de desen (PLOT; CERC), deoarece acestea lucrează în mod unitar cu coordonate din sistemul inițial XOY

```

SUBROUTINE TRXY(X,Y,XT,YT)
COMMON/GBL/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
IV=1
IF(IOGL.EQ.1)IV=-1
XT1=DX+X*XR*IV*COS(ALFA)-Y*YR*SIN(ALFA)
YT1=DY+X*IV*XR*SIN(ALFA)+Y*YR*COS(ALFA)
XT=XT1
YT=YT1
RETURN
END

```

1.12. Subprogramul SPG

- DENUMIRE **SPG**
- FUNCȚIE Subprogramul SPG definește oglindirea față de axa OY luind în considerare subprogramele care transformă de fapt pe X în -X
- APEL **CALL SPG**
- OBSERVAȚII Subprogramul SPG setează la 1 parametrul de oglindire față de axa OX, IOGL
Tot ce se va desena în continuare va fi oglindit față de axa OY.

```

SUBROUTINE SPG
COMMON /GBI/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
IOGL=1
RETURN
END

```

1.13. Subprogramul SPR

- DENUMIRE **SPR**
- FUNCȚIE Subprogramul **SPR** invalidează transformarea definită prin apelul subprogramului **SPG**. Wșadar **SPR** setează la 0 parametrul **IOGL** invalidind oglindirea
- APEL **CALL SPR**
- OBSERVAȚIE Subprogramul setează la zero parametrul **IOGL**, invalidind oglindirea

SUBROUTINE SPR

```
COMMON /GBL/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
IOGL=0
RETURN
END
```

1.14. Subprogramul TLA

- DENUMIRE **TLA**
- FUNCȚIE Subprogramul **TLA** precizează numărul capului de scriere cu care se va lucra. Trecerea de la un cap la altul se va face automat, fără să fie programată.
- APEL **CALL TLA (KS)**
- PARAMETRI **KS** — definește numărul capului de scriere cu care se va lucra
 $1 < KS < 4$
Parametrul se definește prin **INTEGER**
- OBSERVAȚII Prin subprogramul **INI** lui **K** i se atribuie valoarea 1
Subprogramul transmite mesei de desen ordinul 2 A 10 P unde P = 01, 02 (numărul capului de scriere)
- ERORI Eroare 3 — $KS > 2$. Se semnalează eroare dar se consideră
- RETURNATE $\sqrt{(KS \text{ mod } 2 + 1)}$
La inițializarea mesei $KS = 1$

SUBROUTINE TLA(KS)

```
COMMON /GBL/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
LOGICAL*1 F,ORD
DATA F,ORD/42,16/
IF (NLU.GT.5) RETURN
IF (KS.GT.2) CALL ERR(3)
KS1=MOD(KS,2)+1
CALL INV2(KS1,KS1)
WRITE (NLU,1)F,ORD,KS1
FORMAT (2A1,A2)
RETURN
END
```

1.15. Subprogramul EOF

- DENUMIRE **EOF**
- FUNCȚIE Subprogramul **EOF** inchee editarea grafică și duce la decuplarea aparatului de editare (trecerea mesei de desen în mod manual prin ordinul 2 A11A) Simultan se va tipări o informare despre mersul programului.
- APEL **CALL EOF**
- OBSERVAȚII Acest subprogram se va poziționa la sfârșit.
În cazul lucrului cu imprimanta grafică, subprogramul **EOF** incheie fișierul de desen.

SUBROUTINE EOF

```
COMMON /GBL/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
LOGICAL*1 F,ORD
DATA F,ORD/42,26/
IF (NLU.GT.5) RETURN
WRITE (NLU,1)F,ORD
FORMAT (2A1)
F=0
DO 2 I=1,20
WRITE (NLU,1)F,F
RETURN
END
```

1.16. Subprogramul SPD

- DENUMIRE **SPD**
 — FUNCȚIE Subprogramul SPD definește viteza în mm/sec. și treptele de accelerație pentru deplasarea capului de scriere.
 — APEL **CALL SPD (IV1, IV0, IA1, IA0)**
 — PARAMETRII **IV1** — viteza de deplasare a capului de scriere cu penița jos
IV0 — viteza de deplasare a capului de scriere cu penița sus
IV1, IV0, IA1, IA0 > 0.
IA1 — treapta de accelerație a capului cu penița jos
IA0 — treapta de accelerație cu penița sus
 Parametrii **IV1, IV0, IA1, IA0** au caracter **INTEGER**.
 Valorile treptelor de accelerație sînt date în tabela 1.
 — OBSERVAȚIE Maximele pentru vitezele admisibile ale capului de desen stabilit sînt date în tabela 2

tabela 1

treapta de accelerație	accelerația mm/sec ²
1	50
2	200
3	450
4	800
5	1.250
6	1.800
7	2.450
8	3.200

tabela 2

viteza și accelerația maxim admise pentru capul de desen

tipul capului	valoarea de început				valoarea maximă			
	IV0	IV1	IA0	IA1	IV0	IV1	IA0	IA1
— cap de desen	400	200	6	4	400	400	7	7
— cap de înțepat	400	200	6	4	400	400	7	7
— cap de gravat	250	60	4	2	400	400	6	2
— cap de tăiat	250	10	4	2	400	50	6	4
— cap luminos	200	100	2	1	200	100	2	1

```

SUBROUTINE SPD (IV1, IV0, IA1, IA0)
COMMON /GRI/XR,YR,ALFA,DX,DY,I0GL,IRAD,NLU,PI
LOGICAL*1 F,ORD,ORD1
DATA F,ORD,ORD1/42,17,1A/
IF (NLU.GT.5) RETURN
IV0=IABS (IV0)
IV1=IABS (IV1)
IA0=IABS (IA0)
IA1=IABS (IA1)
IF (IV0.GT.400.OR.IV1.GT.400.OR.IA0.GT.7.OR.IA1.GT.7) GOTO 2
CALL INV2 (IV0,IV01)
CALL INV2 (IV1,IV11)
CALL INV2 (IA0,IA01)
CALL INV2 (IA1,IA11)
WRITE (NLU,1) F,ORD,IV01,IV11,F,ORD1,IA01,IA11
1  FORMAT (2A1,2A2)
RETURN
2  CALL ERR (5)
RETURN
END
  
```

Subprogramul SPD trimite mesei de desen ordinele:

2 A 11 IV 0 IV 1

2 A 12 IA 0 IA 1

- **RESTRICȚII** Viteza și accelerația trebuie exprimate prin numere pozitive și este interzisă depășirea valorilor maxime incluse în tabelul anexat.
- **OBSERVAȚIE** Valoarea *IA 1* influențează în execuție calitatea (precizia) desenului.
PT. PROGRAMA
— **MARE** Pentru lucrul cu capul de desen, este bine să fie folosite următoarele valori:
 - lucrare foarte precisă — treapta 3
 - lucrare cu precizie normală — treapta 4
 La inițializarea mesei de desen parametrii au următoarele valori
 - *IV1* = 200 *IVO* = 400
 - *IA1* = 4 *IA0* = 6
- **ATENȚIE** În cazul lucrului cu imprimanta grafică apelul subprogramelor **SPD**, **SSPD** nu are nici un efect.
- **ERORI** EROARE 5 — *IVO* > 400 sau *IV1* > 400 sau
RETURNATE *IA0* > 7 sau *IA1* > 7

1.17. Subprogramul SSPD

- **DENUMIRE** **SSPD**
- **FUNCȚIA** Programul **SSPD** definește viteza în mm/sec. și treptele de accelerație (vezi tabela la subprogramul **SPD**) numai pentru Instrucțiunile desenate cu capul de scriere jos.
CALL SSPD (IV1, IA1)
- **APEL** **IV1** — reprezentarea vitezei în mm/sec
- **PARAMETRI** **IA 1** — treptele de accelerație
- **OBSERVAȚIE** Parametrii *IV 1* și *IA 1* aparțin tipului **INTEGER IV 1**, *IA 1* > 0
Pentru definirea vitezei și accelerației sînt valabile toate restricțiile și completările care au fost expuse în subprogramul **SPD**
Subprogramul **SSPD** trimite mesei de desen ordinul 2 A 15 IV 1 IA 1

SUBROUTINE SSPD(IV1,IA1)

COMMON /GRI/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI

LOGICAL*1 F,ORD

DATA F,ORD/42,21/

IF(NLU.GT.5)RETURN

IV1=IAHS(IV1)

IA1=IABS(IA1)

IF(IV1.GT.400.OR.IA1.GT.7)GOTO 2

CALL INV2(IV1,IV11)

CALL INV2(IA1,IA11)

WRITE(NLU,1)F,ORD,IV11,IA11

1 FORMAT(2A1,2A2)

RETURN

2 CALL ERR(5)

RETURN

END

1.18. Subprogramul BLH

- **DENUMIRE** **BLH**
- **FUNCȚIE** Subprogramul **BLH** folosește la alegerea uneia din cele 7 feluri de linie de trasare (vezi tabela) care va fi aplicată în prelucrarea diferitelor programe.
KL = 0 arată că desenul conține numai linii continue. Pe parcurs **BLH** stabilește tipul de linie cu care se va desena în continuare.
- **APEL** **CALL BLH (KL)**
Apelul este inefectiv în cazul lucrului cu imprimanta grafică.

- PĂRAMEȚRII $KL \in (0, 7)$
 KL — definește tipul corespunzător al liniei de trasare, în total 8 tipuri de linie.
 Parametrul aparține valorilor de tipul *INTEGER*.
- OBSERVAȚIE Tabela cu tipul liniei de trasare:
 Valorile sînt date în sutimi de mm

felul liniei (KL)	L 1	M 1	L 2	M 2	L 3	M 3	L 4	M 4	Observație
0									linie continuă
1	500	200	500	200	500	200	500	200	linie întreruptă
2	800	200	800	200	800	200	800	200	linie întreruptă segmente mari
3	200	200	200	200	200	200	200	200	linie întreruptă segmente mici
4	800	200	200	200	800	200	200	200	linie întreruptă segment mare, segment mic
5	800	200	200	200	200	200	0	0	linie întreruptă segment mare, 2 segmente mici
6	800	200	800	200	200	200	0	0	linie întreruptă, 2 segmente mari, 1 mic
7	800	200	800	200	200	200	200	200	linie întreruptă 2 segmente mari 2 segmente mici

La inițializare $KL = 0$, iar celelalte tipuri de linie sînt predefinite prin valorile din tabela de mai sus.

Subprogramul **BLH** transmite mesei de desen ordinul:

2 A 13 D

unde $D = 00, 01 \dots 07$ (nr. tipului de linie selectat)

- ERORI
 RETURNATE EROARE 4 — $KL > 7$
 Se semnalează eroare dar se ia $KL \bmod 8$.

SUBROUTINE RLH(KL)

COMMON /GBI /XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI

LOGICAL*1 F,ORD

DATA F,ORD/42,19/

IF(NLU.GT.5)RETURN

IF(KL.LT.0.OR.KL.GT.7) GOTO 2

KL1=KL

IF(KL.GT.7)KL1=MOD(KL,8)

CALL INV2(KL1,KL1)

WRITE(NLU,1)F,ORD,KL1

1 FORMAT(2A1,A2)

RETURN

2 CALL ERR(4)

RETURN

END

1.19. Subprogramul BLD

- DENUMIRE **BLD**
- FUNCȚIE Subprogramul **BLD** permite o redefinire a intrării în tabela **BLH** prin care să se schimbe definiția originală a tipului de linie cu care se va desena.
- APEL **CALL BLD (IT, POLE, N)**

- PARAMETRI $IT \in (1, 7)$ — desemnează tipul liniei ce se urmărește să se redefinească.
 POLE — este un vector de N elemente. $N \in (1, 4)$. Dacă $N < 4$ zonele rămase nedefinite se consideră egale cu 0.
 Parametrii IT , $POLE$, N au caracter *INTEGER*.
- OBSERVAȚII Tipurile de linii implicite sînt date în următorul tabel:

tip linie	L 1	M 1	L 2	M 2	L 3	M 3	L 4	M 4	Observații
1	500	200	500	200	500	200	500	200	linie întreruptă
2	800	200	800	200	800	200	800	200	linie întreruptă, segmente mari
3	200	200	200	200	200	200	200	200	linie întreruptă, segmente mici
4	800	200	200	200	800	200	200	200	linie întreruptă, segment mare, segment mic
5	800	200	200	200	200	200	0	0	linie întreruptă, segment mare, 2 segmente mici
6	800	200	800	200	200	200	0	0	linie întreruptă; 2 segmente mari, segment mic
7	800	200	800	200	200	200	200	200	linie întreruptă, 2 segmente mari, 2 segmente mici
0 LINIA CONTINUĂ NU SE POATE REDEFINI									

Dimensiunile sînt date în sutimi de mm.

Un tip de linie este alcătuit din 8 zone unde L 1, L 2, L 3, L 4, sînt dimensiuni de segmente trasate iar M 1, M 2, M 3, M 4, sînt dimensiuni de segmente netrasate.

Vectorul *POLE* din apelul subprogramului **BLD** trebuie să conțină descrierea zonelor L 1, L 2, L 3, L 4, M 1, M 2, M 3, M 4.

Apelul subprogramului **BLD** nu are efect în cazul lucrului cu imprimanta grafică.

Subprogramul **BLD** trimite mesei de desen ordinul

2 A 14 d L 1 M 1 L 2 M 2 L 3 M 3 L 4 M 4

- ERORI
 RETURNATE

EROARE 6 — $N < 1$ sau $N > 4$

La inițializarea mesei de desen tipul implicit de linie este 0 iar celelalte tipuri de linie sînt cele predefinite.

SUBROUTINE PLD(IT,POLE,N)

COMMON /GRI/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PT

INTEGER POIF1(8)

DIMENSION POLE(1)

LOGICAL*1 F,ORD

DATA F,ORD/42,20/

IF(NLU.GT.5) RETURN

IF(N.LT.1.OR.N.GT.4) GOTO 5

IT=MOD(IT,7)+1

DO 1 I=1,2*N

POLE1(I)=POLE(I)*100

1 CALL INV2(POLE1(I),POLF1(I))

IF(N.EQ.4) GO TO 2

DO 3 I=2*N+1,8

3 POLE1(I)=0

2 WRITE(NLU,4) F,ORD,(POIF1(I),I=1,8)

4 FORMAT(2A1,8A2)

RETURN

5 CALL ERR(6)

RETURN

END

1.20. Subprogramul SAV

- DENUMIRE SAV
 — FUNCȚIE Subprogramul SAV asigură salvarea valorilor momentane ale translației, ale rotației, ale scării, ale definiției unghiurilor și al indicatorului de oglindire, adică a acelor valori care sînt definite prin subprogramul **INI** sau sînt definite eventual în program prin apelul subprogramelor anterioare **BEG**; **TRA**; **SCA**; **SSCA**; **ROT**; **ANG**; **DEG**; **SPR**; **ARC**; **SPG**.
 Dacă nici unul din aceste programe nu a fost apelat pînă în momentul respectiv, sînt salvate valorile inițiale ale parametrilor setați de subprogramul **INI**.
 — APEL **CALL SAV**
 — PARAMETRI Nu are
 — OBSERVAȚIE Există posibilitatea salvării unui singur context la un moment dat. Salvarea contextului se poate face pe un singur nivel, o nouă salvare distruge vechiul context salvat.

SUBROUTINE SAV

```
COMMON /GBI/ XR, YR, ALFA, DX, DY, IOGL, IRAD, NLU, PI /
SALV / DXS, DYS, XRS, YRS, AIFAS, IRADS, IOGLS, ISALV
ISALV=1
DXS=DX
DYS=DY
YRS=YR
XRS=XR
ALFAS=ALFA
IRADS=IRAD
IOGLS=IOGL
DX=0.
DY=0.
XR=1.
YR=1.
ALFA=0.
IOGL=0
RETURN
END
```

1.21. Subprogramul PLOT

- DENUMIRE **PLOT**
 — FUNCȚIE Subprogramul **PLOT** comandă deplasarea capului de scriere din punctul curent în punctul (X, Y), cu penița jos dacă $IT = 1$ și cu penița sus dacă $IT = 0$.
 — APEL **CALL PLOT (X, Y, IT)**
 — PARAMETRI X, Y — coordonatele punctului
 $IT = 0$ — deplasare în punctul dorit (cu penița sus)
 $IT = 1$ — deplasare cu desen (penița jos)
 X, Y — se definesc prin **REAL**
 IT — se definește prin **INTEGER**
 — OBSERVAȚII În funcție de valoarea parametrului IT și de mărimea coordonatelor X, Y, subprogramul **PLOT** transmite mesei de desen unul din ordinele:
 2 A 02 X Y — deplasare cu penița sus în punctul de coord. X, Y (cu X, Y pe cite 2 octeți)
 2 A 03 X Y — deplasare identică cu cea anterioară dar cu X, Y pe cite 4 octeți
 2 A 06 X Y — deplasare cu penița jos în punctul de coord. X, Y (cu X, Y pe cite 2 octeți)
 2 A 07 X Y — deplasare identică cu cea anterioară dar cu X, Y pe cite 4 octeți
 Subprogramul **PLOT** este apelat de toate celelalte subprograme de desen. În cazul lucrului cu imprimanta grafică, subprogramul **PLOT** scrie în fișierul de desen coordonatele punctului curent și coordonatele punctului (X, Y).
 Parametrii X, Y sînt afectați de toate transformările definite prin apelurile anterioare ale subprogramelor **TRA**, **BEG**, **ANG**, **ROT**, **SCA**, **SSCA**, **SGP**.

```

SUBROUTINE PLOT(X,Y,IT)
COMMON /GRI/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
COMMON /GRAF/XOT,YOT,NI,G,XO,YO
LOGICAL*1 F,ORD,ORD1,ORD2
DATA F,ORD1,ORD2/42,2,3/
INTEGER*4 X4,Y4
CALL TRXY(X,Y,XT,YT)
IF(NLG.GT.5) GOTO 4
XT=XT*100.
YT=YT*100.
IF(ABS(XT).GT.32767.OR.ABS(YT).GT.32767) GO TO 1
ORD=ORD1+4*IT
IX2=XT
IY2=YT
CALL INV2(IX2,IX2)
CALL INV2(IY2,IY2)
WRITE(NLU,2) F,ORD,IX2,IY2
2  FORMAT(2A1,2A2)
RETURN
1  ORD=ORD2+4*IT
X4=JFIX(XT)
Y4=JFIX(YT)
CALL INV4(X4,X4)
CALL INV4(Y4,Y4)
WRITE(NLU,3) F,ORD,X4,Y4
3  FORMAT(2A1,2A4)
RETURN
4  IF(IT.EQ.0) GOTO 5
WRITE(NLG),XOT,YOT,XT,YT
5  XOT=XT
YOT=YT
XO=X
YO=Y
RETURN
END

```

```

SUBROUTINE INV2(IX,IX1)
LOGICAL*1 X0(2),X
EQUIVALENC F (IX2,X0)
IX2=IX
X=X0(1)
X0(1)=X0(2)
X0(2)=X
IX1=IX2
RETURN
END

```

```

SUBROUTINE INV4(IX,IX1)
INTEGER*4 IX,IX1,IX2
LOGICAL*1 X0(4),X
EQUIVALENC F (IX2,X0)
IX2=IX
X=X0(1)
X0(1)=X0(4)
X0(4)=X
X=X0(2)
X0(2)=X0(3)
X0(3)=X
IX1=IX2
RETURN
END

```

1.22. Subprogramul RET

- DENUMIRE **RET**
- FUNCȚIE Subprogramul **RET** reînnoiește valoarea parametrilor care au fost memorati prin apelul subrutinei **SAV**
Dacă nu a existat un apel **SAV**, subprogramul **RET** nu execută nici o acțiune
- APEL **CALL RET**
- PARAMETRI Nu are

- OBSERVAȚIE Între 2 salvări se pot face oricâte reveniri la vechiul context. Subprogramul SRA apelează SAV, astfel încât dacă anterior s-a făcut o salvare și contextul nu a fost restaurat printr-un apel RET, contextul vechi va fi distrus la ieșirea din SRA.

SUBROUTINE RET

```

COMMON /GBI/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI/
* SALV/DXS,DYS,XRS,YRS,ALFAS,IRADS,IOGLS,ISALV
IF (ISALV.EQ.0) RETURN
DX=DXS
DY=DYS
XR=XRS
YR=YRS
ALFA=ALFAS
IOGL=IOGLS
IRAD=IRADS
RETURN
END

```

1.23. Subprogramul LIN

- DENUMIRE LIN
 — FUNCȚIE Subprogramul LIN asigură unirea a două puncte printr-o linie.
 — APEL CALL LIN (X1, Y1, X2, Y2)
 — PARAMETRI X1, Y1 — coordonatele punctului de început
 X2, Y2 — coordonatele punctului final
 X1, Y1, X2, Y2 se definesc prin REAL
 — OBSERVAȚII Subprogramul LIN este influențat de toate subprogramele de transformare TRA; BEG; ANG; ROT; SCA; SSCA; SPG.

SUBROUTINE LIN(X1,Y1,X2,Y2)

```

COMMON /VARY/XM2,YM2,XM1,YM1,INOT,IBOW,IPBOW
IF (IBOW.EQ.1) GOTO 1
XM2=XM1
YM2=Y1
XM1=X2
YM1=Y2
IF (INOT.EQ.1) GOTO 2
CALL PLOT(X1,Y1,0)
CALL PLOT(X2,Y2,1)
RETURN
2 INOT=0
RETURN
1 CALL INTER(XM2,YM2,XM1,YM1,X1,Y1,X2,Y2,IPBOW)
IBOW=0
GOTO 3
END

```

```

SUBROUTINE INTER(X0,Y0,X1,Y1,X2,Y2,X3,Y3,IP)
P=IP/10.
ALFA1=ATAN2(ABS(Y1-Y0),ABS(X1-X0))
PX1=COS(ALFA1)*P
PY1=SIN(ALFA1)*P
IF(X1.LT.X0)PX1=-PX1
IF(Y1.LT.Y0)PY1=-PY1
ALFA2=ATAN2(ABS(Y3-Y2),ABS(X3-X2))
PX2=COS(ALFA2)*P
PY2=SIN(ALFA2)*P
IF(X3.LT.X2)PX2=-PX2
IF(Y3.LT.Y2)PY2=-PY2
C1X=2.*X1-2.*X2+PX1+PX2
C2X=-3.*X1+3.*X2-2.*PX1-PX2
C3X=PX1
C4X=X1
C1Y=2.*(Y1-Y2)+PY1+PY2
C2Y=-3.*(Y1-Y2)-2.*PY1-PY2
C3Y=PY1
C4Y=Y1
CALL PUL3(C1X,C2X,C3X,C4X,C1Y,C2Y,C3Y,C4Y,X1,Y1)
RETURN
END

```

1.24. Subprogramul NOT

- DENUMIRE **NOT**
— FUNCȚIE Apelul subprogramului **NOT** invalidează trasarea dreptei definită de următorul apel al subprogramului **LIN** sau a primei drepte definită de următorul apel al subprogramului **PLG** care urmează după **BOW**.
- APEL **CALL NOT**
— PARAMETRI Nu are.
— OBSERVAȚII Exemple:
- | | | |
|-----------------|-----|-----------------|
| CALL NOT | sau | CALL NOT |
| CALL LIN | | CALL BOW |

```

SUBROUTINE NOT
COMMON /VART/XM2,YM2,XM1,YM1,INOT,IBOW,IPROW
INOT=1
RETURN
END
SUBROUTINE ROW(N)
COMMON /VART/XM2,YM2,XM1,YM1,INOT,IBOW,IPROW
IPROW=N
IBOW=1
RETURN
END

```

1.25. Subprogramul PLG

- = DENUMIRE **PLG**
= FUNCȚIE Subprogramul **PLG** unește prin segmente punctele ale căror coordonate sînt date în cîmpul **POLE**, în ordinea în care le citește.
Dependent de valoarea parametrului **IT** punctul va fi sau nu unit cu punctul final al poligonului.
- APEL **CALL PLG (POLEX, POLEY, I, J, IT)**
— PARAMETRII **POLEX**; **POLEY** — cîmpul (ordinea) în care sînt scrise coordonatele **X**, **Y** ale parametrilor în acești vectori. O aceeași descriere **POLEX**, **POLEY**, poate fi reprezentată pe porțiuni alegînd diverse perechi de indici **I**, **J**. Nu există constrîngerii asupra dimensiunilor celor doi vectori **POLEX**, **POLEY**, evident cu amendamentul ca aceste dimensiuni să fie cel puțin egale cu **J**. Punctul de început este **POLEX(I) POLEY(I)** iar punctul

de sfârșit este $POLEX(J)$, $POLEY(J)$ I, J – indici de început, respectiv de sfârșit în vectorii $POLEX$, $POLEY$.

Semnificația parametrului IT este următoarea:

$IT = 0$ – punctul de început nu va fi unit cu punctul de sfârșit al poligonului

$IT = 1$ – cele două puncte vor fi unite (cel de început cu cel de sfârșit al poligonului)

Parametrii $POLEX$ și $POLEY$ se definesc prin $REAL$.

Parametrii I, J și IT se definesc prin $INTEGER$.

— OBSERVAȚII

Subprogramul este influențat de toate subprogramele de transformare deci coordonatele conținute în cei doi vectori $POLEX$, $POLEY$ sînt supuse transformării complexe definite de eventualele apeluri anterioare ale subprogramelor ROT , ANG , TRA , BEG , SPG , SCA , $SSCA$.

— ERORI

RETURNATE EROARE 11 – $I \geq J$

```

SUBROUTINE PLG(VX,VY,I,J,IT)
DIMENSION VX(1),VY(1)
IF (I.GE.J) GOTO 3
XE=VX(I)
YE=VY(I)
DO 1 N=I+1,J
CALL LIN(VX(N-1),VY(N-1),VX(N),VY(N))
IF (VX(N).EQ.XE.AND.VY(N).EQ.YE) GOTO 2
GOTO 1
2 N=N+1
XE=VX(N)
YE=VY(N)
1 CONTINUE
IF (IT.EQ.1) CALL LIN(VX(I),VY(I),XE,YE)
RETURN
3 CALL ERR(11)
RETURN
END

```

1.26. Subprogramul SRA

— DENUMIRE

SRA

— FUNCȚIE

Subprogramul SRA hașurează un poligon sub un unghi U și cu distanța dintre hașuri – D – dată.

— APEL

CALL SRA (POLEX, POLEY, I, J, U, D)

— PARAMETRI

$POLEX$; $POLEY$ – cîmpul (vectori) în care se dau coordonatele X, Y ale vîrfurilor poligonului.

Dimensionarea maximă a celor doi vectori a fost aleasă 100, deci se pot hașura poligoane cu cel mult 100 laturi. Descrierea poate conține înlănțuiri de poligoane închise ținînd seama însă ca ultimul punct $POLEX(J)$, $POLEY(J)$ să nu fie identic cu primul punct $POLEX(I)$, $POLEY(I)$. În acest mod se poate hașura simultan o înlănțuire de poligoane. De asemenea, poligoanele pot fi interioare unul celuilalt, fapt ce permite hașurarea corpurilor cu găuri.

Subprogramul SRA consideră în mod implicit ca fiind unite ultimul punct cu primul punct.

U – unghiul de hașurare față de axa OX care se va da în grade sau în radiani, funcție de ultima valoare setată a parametrului $IRAD$

I, J – indici de început și respectiv de sfârșit în vectorii $POLEX$ și $POLEY$

D – distanța dintre liniile de hașurare exprimată în milimetri

$POLEX$, $POLEY$, U , D se definesc în $REAL$, iar I, J se definesc prin $INTEGER$.

— OBSERVAȚII

Subprogramul SRA este influențat de toate apelurile anterioare ale subprogramelor de transformare.

— ERORI

RETURNATE EROARE 14 $J < I + 2$ sau $J - I + 1 > 100$

```

SUBROUTINE SRA(X,Y,I,J,11,D)
DIMENSION X(1),Y(1),A(100),B(100),C(100),XI(100)
IF(J-I+1.LT.3.OR.J-I+1.GT.100) GOTO 11
CALL SAV
CALL ANG(-11)
YMIN=1000000000.
YMAX=-YMIN
DO 1 K=I,J
CALL TRXY(X(K),Y(K),X(K),Y(K))
IF(Y(K).GT.YMAX)YMAX=Y(K)
IF(Y(K).LT.YMIN)YMIN=Y(K)
CONTINUE
CALL RET
CALL ROT(U)
K1=1
XE=X(I)
YE=Y(I)
IORD=I
DO 2 K=I+1,J
A(K1)=Y(K)-Y(K-1)
B(K1)=X(K-1)-X(K)
C(K1)=X(K)*Y(K-1)-X(K-1)*Y(K)
IF(X(K).NE.XE.OR.Y(K).NE.YF)GOTO 2
K1=K1+1
A(K1)=0.
K=K+1
XE=X(K)
YE=Y(K)
IORD=K
2
K1=K1+1
A(K1)=YE-Y(J)
B(K1)=X(J)-XE
C(K1)=XE*Y(J)-X(J)*YE
NRH=(YMAX-YMIN)/D
DO 3 K=1,NRH
INDI=0
YI=YMIN+D*K
DO 4 L=1,K1
IF(A(L).EQ.0)GOTO 4
II=L+I-1
IF1=L+I
IF(L.EQ.K1)IF1=IORD
IF(Y(II).GT.YI.AND.Y(IF1).GT.YI.OR.Y(II).LT.YI.AND.Y(IF1).
LT.YI) GOTO 4
INDI=INDI+1
XI(INDI)=(-C(L)-B(L)*YI)/A(L)
CONTINUE
IF(INDI.EQ.0) GOTO 3
IF(MOD(INDI,2).EQ.1)GOTO 6
DO 5 L1=1,INDI-1
DO 5 L2=L1+1,INDI
IF(XI(L1).LF.XI(L2))GOTO 5
Z=XI(L1)
XI(L1)=XI(L2)
XI(L2)=Z
CONTINUE
GOTO 7
6
YI=YI+.01
INDI=0
GOTO 8
7
DO 9 L1=1,INDI,2
9
CALL LIN(XI(L1),YI,XI(L1+1),YI)
3
CONTINUE
CALL ROT(-U) 10
CALL SAV
CALL ANG(U)
DO 10 K=I,J 11
CALL TRXY(X(K),Y(K),X(K),Y(K))
CALL RET
RETURN
CALL ERR(14)
RETURN
END

```

1.27. Subprogramul BOW

- DENUMIRE **BOW**
- FUNCȚIE Subprogramul **BOW** asigură racordarea a două linii printr-un arc de trecere (curbă).
Apelul trebuie plasat între două apeluri ale subprogramelor **LIN** sau **PLG**.
- APEL **CALL BOW (N)**
- PARAMETRI N — factorul de curbură al arcului
Se recomandă să i se atribuie următoarele plaje $500 < N < 1.500$
Parametrul N se definește prin **INTEGER** și reprezintă de fapt mărimea vectorului tangentei în cele două puncte de racordare
- OBSERVAȚII Dacă $N < 1.000$ arcul se termină în tangentă
Dacă $N > 1.000$ arcul se termină în secantă
Apelul subprogramului **BOW** poate fi plasat și între o combinație de apeluri **LIN**, **PLG** sau **PLG**, **LIN**
- EXEMPLE 1) **CALL LIN (X 1, Y 1, X 2, Y 2)**
 CALL BOW (1.000)
 CALL LIN (X 3, Y 3, X 4, Y 4)
Se racordează cele două drepte (linii) printr-un arc de curbă ce trece prin punctele $(X 2, Y 2)$ și $(X 3, Y 3)$.
Se cere atenție în păstrarea sensului în definiție!
2) **CALL PLG (POLEX, POLEY, I, J, IT)**
 CALL BOW (1.500)
 CALL PLG (POLE 1 X, POLE 1 Y, I, J, IT)
Se racordează ultima dreaptă trasată din descrierea **POLEX**, **POLEY**, cu prima dreaptă trasată din descrierea **POLE 1 X**, **POLE 1 Y**.

```

0001            SUBROUTINE BOW(N)
0002            COMMON /VARI/XM2,YM2,XM1,YM1,INOT,IBOW,IPBOW
0003            IPBOW=N
0004            IBOW=1
0005            RETURN
0006            END

```

1.28. Subprogramul AXI

- DENUMIRE **AXI**
- FUNCȚIE Subprogramul **AXI** desenează și gradează axele **OX** și **OY** din primul cadran, în părți egale. Gradarea începe din punctul de origine al axelor și se face cu două rînduri de diviziuni.
- APEL **CALL AXI (XM, YM, XI, YI, N, M)**
- PARAMETRI XM, YM — lungimea axei **OX** și **OY** — **REAL**
 XI, YI — distanța la care se vor executa subdiviziunile — **REAL**
 N — numărul liniuțelor scurte (de desenare a subdiviziunilor) între două liniuțe lungi (de desenare a diviziunilor) pe axa **OX**.
 M — același lucru pentru axa **OY**
 N și M sint de tip **INTEGER**
Lungimea liniuțelor este constantă
1 mm — pentru cele scurte
2,5 mm — pentru cele lungi
- OBSERVAȚII Subprogramul **AXI** este influențat de toate subprogramele de transformare.

```

SUBROUTINE AXI(XM, YM, XI, YI, N, M)
IF (XM.EQ.0.) GOTO 5
CALL PLOT(0., 0., 0)
CALL PLOT(XM, 0., 1)
NRPX=INT(XM/XI)+1
DO 1 NR=1, NRPX
IF (MOD(NR, N+1).EQ.1) GO TO 2
CALL LIN((NR-1)*XI, 0., (NR-1)*XI, -1.)
GO TO 1
2 CALL LIN((NR-1)*XI, 0., (NR-1)*XI, -2.5)
1 CONTINUE
5 IF (YM.EQ.0.) GOTO 6
CALL PLOT(0., YM, 0)
CALL PLOT(0., 0., 1)
NRPX=INT(YM/YI)+1
DO 3 NR=1, NRPX
IF (MOD(NR, M+1).EQ.1) GO TO 4
CALL LIN(0., (NR-1)*YI, -1., (NR-1)*YI)
GO TO 3
4 CALL LIN(0., (NR-1)*YI, -2.5, (NR-1)*YI)
3 CONTINUE
6 RETURN
END

```

1.29. Subprogramul CSC

- DENUMIRE **CSC**
- FUNCȚIE Subprogramul **CSC** desenează un segment de cerc sau un cerc întreg definit prin centrul cercului, rază, punctul de început și punctul de sfârșit. Dacă *ALFA* nu este egal cu *BETA* va fi desenat un segment de cerc. Dacă *ALFA* = *BETA* va fi desenat un cerc întreg.
- APEL **CALL CSC (X, Y, R, ALFA, BETA)**
- PARAMETRI *X, Y* — coordonatele centrului cercului
R — raza cercului
ALFA — unghiul de început în radiani, unghiul făcut de axa *OX* cu o rază dusă din punctul de început al arcului de desenat
BETA — unghiul de sfârșit în radiani, unghiul făcut de axa *OY* cu o rază dusă din punctul de sfârșit al arcului ce trebuie desenat
R > 0 — cercul va fi desenat în sens trigonometric
R < 0 — cercul va fi desenat în sens Invers trigonometric
Parametrii *X, Y, R, ALFA, BETA* se definesc prin *REAL*.
- OBSERVAȚII Unghiurile *ALFA* și *BETA* se dau în grade sau radiani, funcție de valoarea setată a parametrului *IRAD* și nu sînt influențate de rotații.
Parametrul *R* nu este influențat de setarea scării.
Parametrii *X, Y* sînt influențați de toate subprogramele de transformare apelate anterior.
Dacă utilizatorul dorește să lucreze în grade trebuie mai întii să apeleze subprogramul **DEG**.
Se precizează că 1 RAD = 57°17'45"
- | | |
|-----------------|----------------|
| 1° = 0,01745329 | 3π/2 = 4,71239 |
| π = 3,14160 | |
| π/2 = 1,57080 | 2 π = 6,28318 |

```

SUBROUTINE CSC(X,Y,R,ALFA,BETA)
COMMON/GHL/XR,YR,ALFA,DX,DY,IOGL,IRAD,MLU,PI
ALFAT=ALFA1
BETAT=BETA
IF (IRAD.LT.1)GOTO 1
ALFAT=ALFAT*PI/180.
BETAT=BETAT*PI/180.
1 ALFAT=ALFAT+ALFA
  BETAT=BETAT+ALFA
  IS=0
  IF (R.LT.0) IS=1
  R1=ARS(R)
  IF (R1.LT.0.3) GOTO 2
  XI=X+R1*COS(ALFAT)
  YI=Y+R1*SIN(ALFAT)
  CALL PLOT(XI,YI,0)
  XI=X+R1*COS(BETAT)
  YI=Y+R1*SIN(BETAT)
  CALL CERC(XI,YI,X,Y,IS)
  RETURN
2 CALL ERR(12)
  RETURN
  END

```

1.30. Subprogramul CPP

- DENUMIRE **CPP**
- FUNCȚIE Subprogramul **CPP** desenează un cerc sau un arc de cerc definit prin trei puncte care se află pe circumferința sa.
Generarea începe din punctul (X_1 ; Y_1).
- APEL **CALL CPP** (X_1 , Y_1 , X_2 , Y_2 , X_3 , Y_3 , IT)
- PARAMETRI (X_1 , Y_1), (X_2 , Y_2), (X_3 , Y_3) — coordonatele celor trei puncte.
Semnificația parametrului IT este:
 - $IT = 0$ — se trasează un arc de cerc în sens trigonometric din punctul (X_1 , Y_1) în punctul (X_3 , Y_3) prin punctul (X_2 , Y_2)
 - $IT = 1$ — se trasează un cerc întreg definit prin cele trei puncte; capul de ieșire rămâne în punctul (X_1 , Y_1)

SUBROUTINE CPP(X1,Y1,X2,Y2,X3,Y3,IT)

```

A=Y2-Y1
B=X1-X2
A1=Y3-Y1
B1=X1-X3
PROD=B*A1-R1*A
IF (PROD.EQ.0)GOTO 1
XC=(A*A1*(Y3-Y2)+B*A1*(X1+X2)-A*B1*(X1+X3))*0.5/PROD
YC=-(R1*B1*(X3-X2)-A1*B*(Y1+Y3)+A*B1*(Y1+Y2))*0.5/PROD
CALL PLOT(X1,Y1,0)
XF=X3
YF=Y3
IF (IT.NE.1)GOTO 2
XF=X1
YF=Y1
2 IT1=0
  IF (IT.FO.2) IT1=1
  CALL CERC(XF,YF,XC,YC,IT1)
  RETURN
1 CALL PLOT(X3,Y3,0)
  CALL ERR(13)
  RETURN
  END

```

- $IT = 2$ — se trasează un arc de cerc în sens invers trigonometric din punctul $(X 1, Y 1)$ în punctul $(X 3, Y 3)$, deci care nu trece prin punctul $(X 2, Y 2)$
- **OBSERVAȚII** Parametrii $X 1, Y 1, X 2, Y 2, X 3, Y 3$, sînt influențați de toate sub-programele de transformare apelate anterior
- **ERORI**
- RETURNATE** Eroare 13 — cele trei puncte sînt coliniare.

1.31. Subprogramul CERC

- **DENUMIRE** CERC
- **FUNCȚIE** Subprogramul CERC desenează un arc de cerc din punctul curent pînă în punctul de coordonate (X, Y) , cu centrul în punctul de coordonate (XC, YC)
- **APEL** CALL CERC (X, Y; XC, YC, IS)
- **PARAMETRI** X; Y; XC; YC — de tip REAL
IS — de tip INTEGER
Dacă IS = 0 arcul de cerc se trasează în sens trigonometric.
Dacă IS = 1 arcul de cerc se trasează în sens invers trigonometric.
- **OBSERVAȚII** Subprogramul CERC transmite mesei de desen funcție de valoarea parametrului IS și de mărimea parametrilor X, Y, XC, YC unul din ordinele:
— 2A OA X Y XC YC — trasare arc de cerc în sens trigonometric cu coordonatele X; Y; XC; YC; pe 2 octeți
— 2A OB X Y XC YC — IDEM pe 4 octeți
— 2A OE X Y XC YC — trasare arc de cerc în sens invers trigonometric cu coordonatele X; Y; XC; YC transmise pe cite 2 octeți;
— 2A OF X Y XC YC — IDEM pe 4 octeți.
Dacă se lucrează pe imprimanta grafică arcul de cerc se trasează prin segmente aproximîndu-se cercul prin 90 de segmente (arcul sub 5° se poate bine aproxima cu coarda).
Coordonatele X, Y, XC, YC, sînt supuse transformării complexe definite prin apelurile subprogramelor ROT; ANG; SCA; SSCA; TRA; BEG; SPG.

```

SUBROUTINE CERC(X,Y,XC,YC,IS)
C
IS=0 SENS TRIGONOMETRIC
COMMON /GBI /XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
COMMON /GRAF /XOT,YOT,NLG,XO,YO
LOGICAL*1 F,ORD,ORD1,ORD2
DATA F,ORD1,ORD2/42,10,11/
INTEGER*4 X4,Y4,XC4,YC4
CALL TRXY(X,Y,XT,YT)
CALL TRXY(XC,YC,XTC,YTC)
IF(NLU.GT.5)GOTO 4
XT=XT*100.
YT=YT*100.
XTC=XTC*100.
YTC=YTC*100.
IF(ABS(XT).GT.32767.OR.ABS(YT).GT.32767) GO TO 1
IF(ABS(XTC).GT.32767.OR.ABS(YTC).GT.32767) GO TO 1
ORD=ORD1+4*TS
CALL INV2(TNT(XT),IX2)
CALL INV2(TNT(YT),IY2)
CALL INV2(TNT(XTC),IXC?)
CALL INV2(TNT(YTC),IYC?)

```



```

2      WRITE(NLU,2) F,ORD,IX2,IY2,IXC2,IYC2
      FORMAT(2A1,4A2)
      RETURN
1      ORD=ORD2+4*IS
      CALL INV4(JFIX(XT),X4)
      CALL INV4(JFIX(YT),Y4)
      CALL INV4(JFIX(XTC),XC4)
      CALL INV4(JFIX(YTC),YC4)
      WRITE(NLU,3) F,ORD,X4,Y4,XC4,YC4
      FORMAT(2A1,4A4)
      RETURN
4      C=(XT-XOT)**2+(YT-YOT)**2
      A=(XT-XTC)**2+(YT-YTC)**2
      B=(XOT-XTC)**2+(YOT-YTC)**2
      CU=(A+B-C)/(2*SQR(A*B))
      U=ATAN2(1.-CU*CU),CU
      IF(U.LT.0)U=U+PI
      SEMN=0.
      PROD=(XOT-XTC)*(YT-YTC)-(XT-XTC)*(YOT-YTC)
      IF(PROD.LT.0)SEMN=1.
      IF(IS.NE.SFMN)U=U-2*PI
      IF(SEMN.EQ.1)U=-U
      IF(PROD.NE.0) GOTO 7
      U=2*PI
      IF(XOT.EQ.XT.AND.YOT.EQ.YT) GOTO 7
      U=PI
      IF(XOT.EQ.XT) GOTO 8
      U=SIGN(U,(XT-XOT))
      GOTO 7
8      U=SIGN(U,(YOT-YT))
7      KU=ABS(U)*45./PI
      IF(KU.EQ.0) GOTO 6

```

```

      X1=XO
      Y1=YO
      SA=SIN(SIGN(PI/45.,U))
      CA=COS(PI/45.)
      DO 5 I=1,KU
      X2=XC+(X1-XC)*CA-(Y1-YC)*SA
      Y2=YC+(X1-XC)*SA+(Y1-YC)*CA
      CALL PLOT(X2,Y2,1)
5      X1=X2
      Y1=Y2
6      CALL PLOT(X,Y,1)
      RETURN
      END

```

1.32. Subprogramul CIS

- DENUMIRE CIS
- FUNCȚIE Subprogramul CIS trasează un cerc definit prin centru și rază. Sensul în care va fi desenat cercul va fi hotărît prin sensul razei R.
- APEL CALL CIS (X, Y, R)
- PARAMETRI X, Y – coordonatele centrului cercului
R – raza cercului. Parametrul R nu este afectat de setarea scăriiilor.
R > 0 – cercul va fi desenat în sens pozitiv (trigonometric)
R < 0 – cercul va fi desenat în sens negativ
- OBSERVAȚIE Parametrii X, Y, R, aparțin valorilor de tip REAL
Subprogramul este influențat prin X, Y de toate subprogramele de organizare și transformare apelate anterior.

SUBROUTINE CIS(X,Y,R)

CALL CIT(X,Y,R,0.)

RETURN

END

1.33. Subprogramul CIT

- DENUMIRE **CIT**
 — FUNCȚIE Subprogramul CIT trasează un cerc de rază R , cu centrul în punctul de coordonate (X, Y) și fixează unghiul de ieșire. Sensul de generare este dat de semnul razei.
- APEL **CALL CIT (X, Y, R, U)**
 — PARAMETRII X, Y — coordonatele centrului cercului
 R — raza cercului
 $R > 0$ — sens de generare pozitiv, în sens trigonometric
 $R < 0$ — sens de generare negativ, în sens invers trigonometric
 U — reprezintă unghiul dintre axa OX și raza dusă prin punctul de început și de sfârșit al cercului. Acest unghi determină punctul unde rămâne capul de ieșire la ieșirea din acest subprogram
 Unghiul U se dă în grade sau în radiani, funcție de valoarea setată a parametrului $IRAD$.
 X, Y, R, U — se definesc prin valori de tip *REAL*.
- OBSERVAȚII Dacă utilizatorul dorește să exprime unghiul în grade trebuie să apeleze întâi subprogramele **DEG**. Subprogramul **CIT** este influențat de toate subprogramele de organizare și transformare. Raza R nu este influențată de setarea scărilor.

SUBROUTINE CIT(X,Y,R,U)

CALL CSC(X,Y,R,U,U)

RETURN

END

1.34. Subprogramul IPO

- DENUMIRE **IPO**
 — FUNCȚIE Subprogramul IPO unește cu o curbă netedă punctele date ale căror coordonate sînt memorate în vectorii *POLEX*, *POLEY* în ordinea în care coordonatele acestor puncte apar în cei doi vectori. În punctele inițial și final valoarea curburii derivatei a doua este nulă.
- APEL **CALL IPO (POLEX, POLEY, I, J)**
 — PARAMETRII *POLEX*; *POLEY* — vectori de tip *REAL* în care sînt memorate coordonatele punctelor ce vor fi unite
 I, J — reprezintă indici de început și respectiv de sfârșit în cei doi vectori de coordonate. Numărul punctelor date trebuie să fie mai mare ca 3, deci $J - I \geq 2$.
 Parametrii I, J sînt de tip *INTEGER*.
- OBSERVAȚII În acest subprogram s-a folosit metoda de interpolare de tip *SPLINE* cu funcții de gradul 3 pe fiecare interval și se recomandă să se aleagă o distribuție regulată a punctelor pe curbă. Curbele individuale, între fiecare două puncte, sînt desenate prin cîte 20 de segmente.
 Subprogramul este influențat de toate subprogramele de organizare și transformare ca de exemplu **ROT**, **ANG**, **TRA**, **BEG**, **SCA**, **SSCA**, **SPG**.
- ERORI
 RETURNATE EROARE 15 $J - I < 2$

În afară de aplicațiile și programele test de la sfîrșitul acestui capitol este util ca în privința trasării diferitelor curbe să fie cercetate de către cititor și celelalte programe și subprogramele incluse în lucrare, cum ar fi, de exemplu, programele **CURBA** din Capitolul VIII sau programele speciale din Capitolele IX și X care rezolvă problemele de interpolare Lagrange, Hermite, Coons, Akimov, B-spline, Bézier sau care rezolvă problemele de construcție a suprafețelor în grafica pe calculator. Un interes deosebit prezintă și programele din Capitolul V pentru trasarea curbelor exprimate explicit sau parametric în reprezentarea ortogonală sau în perspectiva centrală sau axonometrică.

```

SURROUTINE IPO(X,Y,II,TF1)
DIMENSION X(1),Y(1)
REAL*8 CD,RI1,CI,RI,VALX,VALY
IF(IF1-II+1.LT.3) GOTO 3
CD=-3.
BI=-2.
CI=3.

```

```

VALX=CD*(X(II+1)-X(II))
VALY=CD*(Y(II+1)-Y(II))
DO 1 IND=II+2,IF1-1
CD=(CI-BI)*3.
BI1=CI-BI*2
CI=3.*BI-2.*CI
HI=RI1
VALX=VALX+CD*(X(IND)-X(TND-1))
VALY=VALY+CD*(Y(IND)-Y(TND-1))
CONTINUE
BNX=(-VALX-CI*(X(IND)-X(IND-1)))/(RI-2.*CI/3.)
CNX=X(IND)-X(IND-1)-2.*BNX/3.
ANX=-BNX/3.
BNY=(-VALY-CI*(Y(IND)-Y(IND-1)))/(RI-2.*CI/3.)
CNY=Y(IND)-Y(IND-1)-2.*BNY/3.
ANY=-BNY/3.
CALL POL3(ANX,BNX,CNX,X(IND-1),ANY,BNY,CNY,Y(IND-1),X(IND-1),
*Y(IND-1))
N=IF1-II+1
DO 2 J=1,N-2
DELX=X(IF1-J)-X(IF1-J-1)
DELY=Y(IF1-J)-Y(IF1-J-1)
AX=DELX+BNX-CNX
AY=DELY+BNY-CNY
BX=3*CNX-2*ANX-3.*DELX
BY=3*CNY-2*BNY-3.*DELY
CX=-2.*CNX+BNX+3.*DELX
CY=-2.*CNY+BNY+3.*DELY
CALL POL3(AX,BX,CX,X(IF1-J-1),AY,BY,CY,Y(IF1-J-1),X(IF1-J-1),
*Y(IF1-J-1))
BNX=BX
CNX=CX
BNY=BY
CNY=CY
CONTINUE
RETURN
CALL ERR(15)
RETURN
END

```

```

SUBROUTINE POL3(C1X,C2X,C3X,C4X,C1Y,C2Y,C3Y,C4Y,X1,Y1)

```

```

CALL PLOT(X1,Y1,0)
DEL=1./19.
V0=DEL*DEL
D3V0X=6.*C1X*DEL*V0
D2VKX=D3V0X+2.*C2X*V0
D1VKX=C1X*V0*DEL+C2X*V0+C3X*DEL
D3V0Y=6.*C1Y*DEL*V0
D2VKY=D3V0Y+2.*C2Y*V0
D1VKY=C1Y*V0*DEL+C2Y*V0+C3Y*DEL
V1X=X1
V1Y=Y1
DO 1 K=2,20
V1X=V1X+D1VKX
V1Y=V1Y+D1VKY
D1VKX=D1VKX+D2VKX
D1VKY=D1VKY+D2VKY
D2VKX=D2VKX+D3V0X
D2VKY=D2VKY+D3V0Y
CALL PLOT(V1X,V1Y,1)
RETURN
END

```

1.35. Subprogramul IPR

- DENUMIRE **IPR**
- FUNCȚIE Subprogramul **IPR** unește printr-o curbă netedă închisă punctele ale
căror coordonate sînt memorate în vectorii **POLEX**, **POLEY** în ordinea
în care coordonatele acestor puncte apar în cei doi vectori (sînt necesare
cel puțin patru puncte)
- APEL **CALL IPR (POLEX, POLEY, I, J)**
- PARAMETRII **POLEX**, **POLEY** — vectorii care conțin coordonatele succesive ale puncte-
lor ce vor fi unite
I, J — indici de început, respectiv de sfîrșit în vectorii de coordonate
Numărul punctelor trebuie să fie mai mare ca 3, deci $J - I \geq 2$.
POLEX, **POLEY** aparțin valorilor de tip **REAL**, iar *I, J* aparțin valorilor
INTEGER.
- OBSERVAȚII S-a folosit metoda de interpolare de tip **SPLINE** cu funcții de gradul 3
ca și în cazul subprogramului **IPO** cu diferența că, deoarece punctul final
POLEX (J), **POLEY (J)** este unit cu punctul inițial **POLEX (I)**, **POLEY (I)**
se impune egalitatea derivatelor întîia cu a doua, pentru prima și ultima
funcție în punctul **POLEX (I)**, **POLEY (I)**.
Valorile coordonatelor memorate în cei doi vectori sînt supuse transformării
definite în urma apelurilor subprogramelor **ROT**, **ANG**, **TRA**, **BEG**, **SCA**,
SSCA, **SPG**.
- ERORI
RETURNATE EROARE 16 $J - I < 2$

```

SUBROUTINE IPR(X,Y,II,IF1)
COMMON /INC/ID
DIMENSION X(1),Y(1)
ID=1
CALL IPC(X,Y,II,IF1)
RETURN
END
SUBROUTINE FRR(N)
TYPE 1,N
FORMAT(' DIGILIB ***EROARE ',I3,10000)
RETURN
END

```

1.36. Subprogramul IPC

- DENUMIRE **IPC**
- FUNCȚIE Subprogramul **IPC** este identic cu subprogramul **IPR** singura deosebire
constind în faptul că ultima curbă, cea care unește punctul final cu cel
inițial, nu este reprezentată (sînt necesare cel puțin patru puncte).
- APEL **CALL IPC (POLEX; POLEY, I, J)**
- PARAMETRII **POLEX**; **POLEY** — au semnificația identică cu cea din subprogramul
IPR.
Parametrii **POLEX**, **POLEY**, sînt vectori de tip **REAL**.
I, J — au semnificația identică cu cea din subprogramul **IPR**.
I, J sînt parametrii de tip **INTEGER**.

```

SUBROUTINE IPC(X,Y,II,IF1)
COMMON /INC/ID
REAL*8 CBB,CCH,CRC,CCC,SBX,SBY,SCX,SCY,CDR,CDC,CBB1,CBC1
DIMENSION X(1),Y(1)
N=IF1-II+1
IF(N.LT.3) GOTO 3
DO 2 J=1,IF1-II+ID
  CBF=-2.
  CCH=3.
  CBC=1.
  CCC=-2.
  SBX=-3.*(X(II+MOD(J,N))-X(II+MOD(J-1,N)))
  SBY=-3.*(Y(II+MOD(J,N))-Y(II+MOD(J-1,N)))
  SCX=-SBX
  SCY=-SBY
  DO 1 IND=1,N-1
    CDR=(CCB-CRB)*3.
    CDC=(CCC-CRC)*3.
    CBB1=CCR-2.*CBB
    CR1=CCC-2.*CBC
    CCB=3.*CBB-2.*CCB
    CCC=3.*CRC-2.*CCC
    CBB=CBB1
    CBC=CBC1
    SBX=SBX+CDR*(X(II+MOD(J+IND,N))-X(II+MOD(J+IND-1,N)))
    SBY=SBY+CDR*(Y(II+MOD(J+IND,N))-Y(II+MOD(J+IND-1,N)))
    SCX=SCX+CDR*(X(II+MOD(J+IND,N))-X(II+MOD(J+IND-1,N)))
    SCY=SCY+CDR*(Y(II+MOD(J+IND,N))-Y(II+MOD(J+IND-1,N)))
1  CONTINUE
  B1X=(SCX*CCR-SBX*(CCC-1.))/((CCC-1.)*(CBB-1.)-CCB*CBC)
  B1Y=(SCY*CCR-SBY*(CCC-1.))/((CCC-1.)*(CBB-1.)-CCB*CBC)
  C1X=(-SBX-(CBB-1.)*B1X)/CCB
  C1Y=(-SBY-(CBB-1.)*B1Y)/CCB
  A1X=X(II+MOD(J,N))-X(II+MOD(J-1,N))-B1X-C1X
  A1Y=Y(II+MOD(J,N))-Y(II+MOD(J-1,N))-B1Y-C1Y
100 TYPE 100,A1X,B1X,C1X,X(II),A1Y,B1Y,C1Y,Y(II)
  FORMAT(8F12.3)
  CALL POL3(A1X,B1X,C1X,X(II+MOD(J-1,N)),A1Y,B1Y,C1Y,
  *Y(II+MOD(J-1,N)),X(II+MOD(J-1,N)),Y(II+MOD(J-1,N)))
2  CONTINUE
  ID=0
  RETURN
3  CALL ERR(16)
  RETURN
END

```

1.37. Subprogramul TEX

- DENUMIRE **TEX**
- FUNCȚIE Subprogramul **TEX** tipărește caracterele ale căror coduri *ASCII* se află în vectorul **TEX** cu lungimea *NRCAR*, unde $NRCAR \leq 80$. Cu alte cuvinte subprogramul **TEX** realizează scrierea unui text care poate fi format din litere mari și litere mici ale alfabetului latin, precum și cifre și semne speciale
- APEL **CALL TEX (X, Y, ALFAT, VEL, IREF, TEXT, NRCAR)**
- PARAMETRII **X, Y, ALFAT, VEL** — de tip *REAL*
IREF, NRCAR — de tip *INTEGER*
TEXT — vector *LOGICAL* ★ 1
X, Y — coordonatele punctului de referință
 Funcție de valoarea parametrului de aliniere *IREF*, punctul de coordonate **X, Y** reprezintă:

$IREF = 0$; X, Y — coordonatele punctului de început al textului (aliniere la stînga)

$IREF = 1$; X, Y — coordonatele punctului de mijloc al textului (aliniere la mijloc)

$IREF = 2$; X, Y — coordonatele punctului de sfîrșit al textului (aliniere la dreapta)

Parametrii X, Y sînt afectați de transformarea definită prin apelurile anterioare ale subprogramelor **ROT, ANG, TRA, BEG, SCA, SSCA, SPG**.

Parametrul $ALFAT$ este unghiul care dă înclinarea textului față de axa OX , unghiul fiind considerat în sens direct trigonometric. Valoarea $ALFAT$ se va da în grade sau în radiani funcție de valoarea parametrului $IRAD$. Unghiul $ALFAT$ nu este influențat de rotații.

Parametrul VEL dă înălțimea caracterelor în milimetri și nu este afectat de scară.

Vectorul $TEXT$ conține textul ce trebuie scris avînd lungimea de $NRCAR \leq 80$ înțelegînd în numărul de caractere și spațiul dintre cuvinte.

```

SUBROUTINE TEX(X,Y,ALFAT,VEL,IREF,TEXT,NRCAR)
COMMON /GBI /XR,YR,ALFA,DX,DY,IOGL,IRAD, NLU,PI
COMMON /HCAR/ALFAD,H,IPOZ1
COMMON/BCAR/BET
LOGICAL*1 TEXT(80),F,ORD1,ORD2,ORD3,ORD4,TAB(91),TEXTC(80),ZERO
DATA F,ORD1,ORD2,ORD3,ORD4,ZERO/42,28,29,30,31,0/
DATA TAB/64,79,87,65,140,108,80,86,77,93,92,78,107,96,75,97,
1 240,241,242,243,244,245,246,247,248,249,84,94,76,66,110,111,
1 23,193,194,195,196,197,198,199,200,201,209,210,211,212,213,214,
1 215,216,217,226,227,228,229,230,231,232,233,74,0,90,0,0,
1 0,129,130,131,132,133,134,135,136,137,145,146,147,148,149,150,
1 151,152,153,162,163,164,165,166,167,168,169/
IF(NRCAR.LF.0.OR.NRCAR.GT.80) GOTO 12
ALFA1=ALFAT
IF(IRAD.EQ.1) ALFA1=ALFA1*PI/180.
ALFA1=ALFA1+ALFA
IF(NLU.GT.5.OR.PET.NE.PI/2.)GOTO 7
IF(ALFA1.EQ.ALFAD)GO TO 1
ALFAD=ALFA1
SA=16384.*STN(ALFA1)
CA=16384.*COS(ALFA1)
CALL INV2(TNT(SA),ISA)
CALL INV2(TNT(CA),ICA)
WRITE(NLU,2)F,ORD2,ISA,TCA
2 FORMAT(2A1,2A2)
1 IF(VEL.EQ.H) GO TO 3
H=VEL
CALL INV2(TNT(VEL*100.),IH)
WRITE(NLU,2)F,ORD1,IH
3 IF(IREF.EQ.IPOZ) GO TO 4
IPOZ=IREF
IF(IPOZ.GT.2.OR.IPOZ.LT.0)GOTO 13
CALL INV2(TPOZ,IPOZ1)
WRITE(NLU,2)F,ORD3,IPOZ1
4 IF(NRCAR.GT.80) NRCAR=80
NRC=NRCAR
IF(MOD(NRCAR,2).EQ.1) NRC=NRC+1
CALL INV2(NRC,NRC)
CALL PLOT(X,Y,0)
DO 14 I=1,NRCAR
IF(TEXT(I).IT.32.OR.(TEXT(I)-31).GT.91) GOTO 15
14 CONTINUE
WRITE(NLU,5) F,ORD4,NRC.(TAB(TEXT(I)-31),I=1,NRCAR)
5 FORMAT(2A1,A2,80A1)
IF(MOD(NRCAR,2).EQ.1)WRITE(NLU,6)ZERO
6 FORMAT(A1)

```

```

7      H1=VEL/6.
      IF (NLU.LT.5) H1=H1*100.
      B=H1*7./8.
      X2=X
      Y2=Y
      IF (IREF.LT.1) GOTO 8
      X1=X
      Y1=Y
      DO 9 I=1,NRCAR
9      CALL CHAR(TEXT(I),X1,Y1,0,B,H1,ALFA1)
      IF (IREF.LT.2) GOTO 10
      X2=2*X-X1
      Y2=2*Y-Y1
      GOTO 8
10     X2=(3*X-X1)/2.
      Y2=(3*Y-Y1)/2.
      CALL PLOT(X2,Y2,0)
8      DO 11 I=1,NRCAR
11     CALL CHAR(TEXT(I),X2,Y2,1,B,H1,ALFA1)
      RETURN
12     CALL ERR(8)
      RETURN
13     CALL ERR(9)
      RETURN
15     CALL ERR(10)
      RETURN
      END

```

```

SUBROUTINE SCHAR(U)
COMMON /BCAR/BET
COMMON /GBL/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI
BET=U
IF (IRAD.GT.0) BET=U*PI/180.
BET=BET+ALFA
RETURN
END

```

Subprogramul TEX trimite mesei de desen ordinea:

```

2 A 1 C VEL
2 A 1 D sin (ALFAT) cos (ALFAT)
2 A 1 E IREF
2 A 1 F NRCAR TEXT (1) TEXT (2) ... TEXT (NRCAR)

```

— OBSERVAȚII Pentru lucrul cu imprimanta grafică se folosește un generator de caractere SOFT, subprogramul TEX făcînd apel la rutina CHAR. Această rutină ia în considerație și înclinarea, parametru ce se poate seta prin apelul rutină SCHAR.

CALL SCHAR (BET)

unde BET este unghiul de înclinare al caracterelor față de axa OX, definit în sens trigonometric.

Chiar dacă se lucrează pentru masa de desen ($NL < 5$) se poate folosi generatorul de caractere SOFT dacă se dorește utilizarea caracterelor cu o înclinare $BET \neq \pi/2$, valoare setată de către INI. Rezultă deci că un apel al rutinei SCHAR cu un unghi $BET \neq \pi/2$ va determina ca toate apelurile ulterioare ale subprogramelor TEX și NUM să folosească generatoarele de caractere SOFT (subprogramul NUM apelează subprogramul TEX).

— ERORI EROARE 8 — $NRCAR \leq 0$ sau $NRCAR > 80$

RETURNATE EROARE 9 — $IPOZ < 0$ sau $IPOZ \geq 2$

EROARE 10 — Vectorul TEXT conține un cod de caractere nepermis /nu face parte din codurile ASCII/

SUBROUTINE CHAR(LTCOD,XR,YB,IT,8,H1,ALFA1)

LOGICAL*1 MATCAR(659,2),LTCOD

DIMENSION INDMAT(92)

COMMON /BCAR/BET

```

DATA INDMAT/1.2,9,14,23,36,48,58,61,65,70,79,84,88,91,96,99,109,
112,119,129,134,145,158,162,179,192,203,212,216,221,225,236,252,
258,271,280,288,295,301,312,318,320,327,333,336,341,346,356,364,
1 375,384,397,402,409,413,419,423,429,433,437,439,444,448,450,450,
1 461,472,481,492,503,510,524,531,536,541,547,551,561,568,578,589,
1 600,607,618,624,631,635,641,645,656,660/
DATA MATCAR/-3,-2,3,1,2,-2,2,-3,-1,1,-2,2,-2,-2,2,3,-4,3,-5,1,-1,
1 5,-5,-1,2,4,5,5,1,1,2,4,5,-3,3,-5,-1,1,2,2,1,-5,1,-5,5,4,4,5,-5,
1 3,2,1,1,4,3,2,1,5,-1,1,-1,-2,1,1,2,-1,2,2,1,-2,-1,3,-2,2,-3,1,
1 -1,3,-3,-1,3,-2,2,5,-2,1,-2,-1,3,-3,-2,1,1,2,2,-1,3,-3,-2,1,
1 1,2,4,5,5,4,2,-5,-1,2,2,-1,2,4,5,5,1,5,-1,2,4,5,5,4,2,5,1,-5,-4,
1 4,1,5,-5,-1,2,4,5,5,4,2,1,1,5,-5,-1,2,4,5,5,4,2,1,1,2,4,5,-5,-2,
1 5,1,-5,-2,1,1,2,4,5,5,4,2,1,1,2,4,5,5,4,-5,-1,2,4,5,5,4,2,1,1,
1 2,4,5,-5,-1,1,2,2,1,-1,1,2,2,1,-2,-1,2,2,1,1,-1,2,1,-2,-3,1,3,
1 -3,-1,3,-3,1,-3,-1,3,1,-3,-1,2,4,5,5,4,3,3,-3,3,-5,-4,3,2,2,3,4,
1 4,5,5,4,2,1,1,2,3,-5,-1,3,5,-2,4,-5,-1,1,4,5,5,4,1,4,5,5,4,1,
1 -5,-5,4,2,1,1,4,5,5,4,1,-5,-1,1,4,5,5,4,1,-5,-1,1,5,-4,1,1,5,-1,1,5,
1 -4,1,-5,-3,5,5,4,2,1,1,2,4,5,-5,-1,1,1,5,5,5,-1,1,-1,2,4,5,5,1,
1 -5,-1,1,-5,1,2,5,-1,1,5,-1,1,3,5,5,-1,1,5,5,-5,-2,1,1,2,4,5,5,4,
1 2,-5,-1,1,4,5,5,4,1,-5,-2,1,1,2,4,5,5,4,2,4,5,-1,1,4,5,5,4,1,4,
1 5,-1,2,4,5,5,4,2,1,1,2,4,5,-5,-3,3,1,5,-5,-1,1,2,4,5,5,-5,-1,3,
1 5,-5,-1,2,3,4,5,-5,-1,5,-1,5,-3,3,1,-5,3,-5,-1,5,1,5,-3,1,1,3,-1,
1 3,-1,3,3,1,-5,-1,2,3,-3,-1,4,-4,4,4,3,2,1,1,2,3,4,-4,-1,1,1,2,3,
1 4,4,3,2,1,-4,-4,3,2,1,1,2,3,4,-4,-4,4,4,3,2,1,1,2,3,4,-4,-1,4,4,
1 3,2,1,1,2,3,4,-4,-2,2,3,4,-3,1,-4,-1,2,3,4,4,4,3,2,1,1,2,3,4,-4,
1 -1,1,1,2,3,4,4,-1,1,2,3,-3,-1,2,3,3,3,-1,1,1,2,4,-1,1,2,2,-2,
1 -1,1,1,2,3,3,3,4,5,5,-1,1,1,2,3,4,4,-2,1,1,2,3,4,4,3,2,-4,-1,1,
1 1,2,3,4,4,3,2,1,-4,-4,4,4,3,2,1,1,2,3,4,-4,-1,1,1,2,3,4,-4,-1,2,
1 3,4,3,2,1,2,3,4,-4,-1,3,-2,2,3,-3,-1,1,2,3,4,4,-1,2,4,-4,-1,2,
1 3,4,5,-5,-1,4,-1,4,-1,2,3,4,4,-1,1,2,3,4,-4,-1,4,4,0,1,6,6,1,
1 0,0,0,4,5,5,4,0,0,6,6,0,2,2,2,4,4,0,2,1,1,2,3,3,4,5,5,4,6,0,4,6,
1 6,4,4,6,0,0,2,2,0,0,2,0,0,1,2,5,6,6,4,0,4,5,0,6,5,1,0,0,1,5,6,0,
1 2,4,4,2,2,4,3,3,0,3,3,4,2,0,1,1,0,0,3,3,0,0,1,1,0,0,6,0,2,2,4,
1 6,6,4,2,0,0,0,4,6,0,5,6,6,5,4,0,0,1,0,0,1,3,4,4,6,6,0,0,6,2,2,0,
1 1,0,0,1,3,4,4,3,6,6,0,3,4,4,3,1,0,0,1,5,6,6,5,0,0,6,6,0,3,4,5,6,
1 6,5,4,3,3,2,1,0,0,1,2,3,0,1,0,0,1,5,6,6,5,3,2,2,3,2,3,4,3,3,2,
1 1,1,2,2,0,3,3,4,4,3,1,1,0,0,4,3,2,0,2,2,3,3,0,2,3,4,0,5,6,6,5,4,
1 3,3,1,0,0,0,1,0,1,3,4,3,0,0,5,6,6,5,0,-1,-1,0,0,6,0,3,3,0,0,6,6,
1 5,4,3,3,2,1,0,0,0,5,6,6,5,1,0,0,1,0,0,6,6,5,1,0,0,0,0,6,6,3,3,
1 0,0,0,6,6,3,3,0,3,3,1,0,0,1,5,6,6,5,0,0,6,3,3,6,0,6,0,1,0,0,1,6,
1 6,0,0,6,6,2,3,0,6,0,0,6,6,3,6,0,0,6,0,6,0,6,0,1,5,6,6,5,1,0,0,0,
1 6,6,5,4,3,3,0,0,1,5,6,6,5,1,0,0,1,0,0,6,6,5,4,3,3,3,0,1,0,0,1,2,
1 3,3,4,5,6,6,5,0,0,6,6,6,0,6,1,0,0,1,6,0,6,0,6,0,6,0,3,0,6,0,0,6,
1 6,0,0,3,6,6,3,0,6,6,0,0,6,6,0,0,6,0,0,0,6,6,0,3,5,3,0,0,0,0,4,3,
1 4,4,3,1,0,0,1,0,0,6,3,4,4,3,1,0,0,1,0,3,4,4,3,1,0,0,1,0,0,6,3,4,
1 4,3,1,0,0,1,0,2,2,3,4,4,3,1,0,0,1,0,0,5,6,5,2,2,0,-1,-2,-2,-1,4,
1 3,4,4,3,1,0,0,1,0,0,6,3,4,4,3,0,4,1,0,1,0,-1,-2,-1,4,0,0,6,4,1,
1 2,0,6,0,1,0,0,4,3,4,3,0,3,4,3,0,0,4,3,4,4,3,0,0,1,3,4,4,3,1,0,0,
1 0,-2,4,3,4,4,3,1,0,0,1,0,-2,4,3,4,4,3,1,0,0,1,0,0,4,3,4,4,3,0,1,
1 0,0,1,2,2,3,4,4,3,0,4,4,6,0,1,0,4,1,0,0,1,4,0,4,0,4,0,4,0,2,0,4,
1 0,0,4,4,0,-1,-2,-2,-1,4,4,1,0,0,1,0,4,4,0,0/

```



```

LITCOD=LTCOD-31
INDI=INDMAT(LITCOD)
INDF=INDMAT(LITCOD+1)-1
IF(INDF.LT.INDI)RETURN
D=0.
C=COS(ALFA1)
S=SIN(ALFA1)
D=COS(BET)/SIN(BET)
1 DO 4 J=INDT.INDF
  JPEN=1
  I1=MATCAR(T,1)
  IF(I1.GT.0)GOTO 2
  JPEN=0
  I1=-I1
2 I2=MATCAR(T,2)
  XD=H1*D*I2+R*I1
  YD=H1*I2
  IF(ALFA1.EQ.0.)GOTO 3
  X1=XD*C-YD*S
  YD=XD*S+YD*C
  XD=X1
3 XD=XB+XD
  YD=YB+YD
  IF(IT.LT.1)GOTO 4
  CALL PLOT(XD,YD,JPEN)
4 CONTINUE
  XB=XD
  YB=YD
  RETURN
END

```

1.38. Subprogramul POS

= DENUMIRE POS
= FUNCȚIE

= APEL
= PARAMETRI

= OBSERVAȚIE

POS

Subprogramul POS trimete mesei de desen ordinul 2 A 18 N, de trecere în mod manual, cu deplasarea capului de scriere în punctul (X, Y) și cu afișarea pe display-ul de pe panoul de comandă al mesei, a numărului N

CALL POS (X, Y, N)

X, Y — coordonatele punctului de tranziție

Parametrii X, Y, aparțin valorilor de tip REAL. N - INTEGER

Subprogramul va fi influențat de toate subprogramele de organizare și transformare, cu excepția subprogramului LMT.

Cu ajutorul acestui subprogram se pot introduce în program puncte de oprire, pentru a se putea introduce comenzi manuale sau să se schimbe hîrtia.

Numărul N nu trebuie să depășească valoarea 8191.

În cazul lucrului cu imprimanta grafică, apelul subprogramului POS nu are nici un efect.

SUBROUTINE POS(X,Y,N)

COMMON /GHI/XR,YR,ALFA,DX,DY,IOGL,IRAD,NLU,PI

LOGICAL*1 F,ORD

DATA F,ORD/42,24/

IF(NLU.GT.5)RETURN

CALL PLOT(X,Y,0)

IF(N.GT.8191)N=8191

CALL INV2(N,N1)

WRITE(NLU,1) F,ORD,N1

FORMAT(2A1,A2)

RETURN

END

1.39. Subprogramul NUM

- DENUMIRE NUM
 — FUNCȚIE Subprogramul NUM tipărește numărul real C cu N zecimale. Numărul de zecimale trebuie să fie mai mare decât zero și mai mic decât 15. Numărul este scris cu 15 cifre dintre care N cifre sunt zecimale.
- APEL CALL NUM (X, Y, ALFAT, VEL, IREF, C, N)
- PARAMETRII X, Y — coordonatele punctului de referință
 $ALFAT$ — înclinarea rîndului de scriere față de axa OX , în grade
 C — număr real de maximum 15 cifre
 N — numărul de zecimale > 0
 VEL — înălțimea cifrelor
 $IREF$ — specifică poziția rîndului scris în funcție de punctul de referință
 $0 < = IREF = < 2$
- Parametrul C se definește în *REAL* iar parametrul N se definește în *INTEGER*.
 Parametrii $X, Y, ALFAT, VEL, IREF$ — au aceeași semnificație ca și în cadrul subprogramului *TEX*.
- ERORI
 RETURNATE Eroare 7 — $N < 0$ sau $N > 15$.

```

SUBROUTINE NUM(X,Y,ALFA,VEL,TREF,C,N)
  LOGICAL*1 TFXT(17)
  DOUBLE PRECISION FOR(3),F(15)
  DATA FOR(1),FOR(3),F(1),F(15)
  DATA F/'F17.1','F17.2','F17.3','F17.4','F17.5','F17.6','F17.7',
  1 'F17.8','F17.9','F17.10','F17.11','F17.12','F17.13',
  1 'F17.14','F17.15'/
  IF(N.LT.0.OR.N.GT.15) GOTO 8
  IF(N.EQ.0) GOTO 1
  FOR(2)=F(N)
  ENCODE(17,FOR,TEXT) C
  GOTO 4
1  NR=C
  ENCODE(17,3,TEXT) NR
3  FORMAT(I17)
4  II=1
  IF(TEXT(1).EQ.45) II=2
  DO 5 J=II,17
  IF(TEXT(J).NE.32) GOTO 6
5  CONTINUE
6  DO 7 K=J,17
  TEXT(II+K-J)=TEXT(K)
7  CONTINUE
  CALL TEX(X,Y,ALFA,VEL,TREF,TEXT,II+K-J-1)
  RETURN
8  CALL ERR(7)
  RETURN
  END
  
```

1.40. Lista erorilor returnate de subprogramele grafice. Aplicații. Programe test.

Subprogramele grafice din biblioteca *DIGILIB* returnează la consolă în caz de eroare, următorul mesaj:

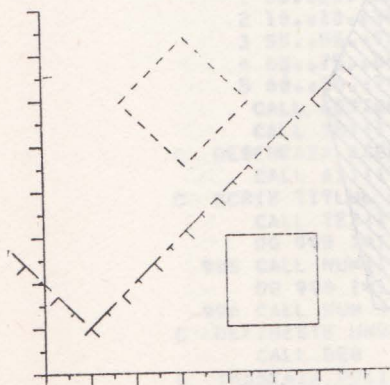
xxx DIGILIB EROARE N xxx

unde N reprezintă numărul erorii întâlnite.

În tabelul de mai jos sînt date valorile posibile ale lui N , cu semnificațiile corespunzătoare, precum și acțiunea întreprinsă în caz de eroare:

Număr eroare	Sub program	Semnificație
1	SCA	– factorul de scară pentru axele OX sau OY negativ sau nul. Dacă valoarea este negativă, se ia valoarea absolută. Dacă valoarea este nulă, subprogramul nu execută nici o acțiune.
2	SSCA	– aceeași semnificație ca eroare 1.
3	TLA	– definire incorectă a capului de scriere ($KS > 2$) se consideră $KS \bmod 2 + 1$
4	BLH	– tipul liniei definit incorect ($KL < 0$ sau $KL > 7$) nu se modifică tipul liniei.
5	SPD	– una din viteze este mai mare decît 400 mm/s sau una din treptele de accelerație este mai mare decît 7.
	SSPD	Vitezele de accelerație rămîn nemodificate.
6	BLD	– numărul de zone din definiția tipului de linie este definit incorect ($N > 4$ sau $N < 1$). Nu se redefineste tipul specificat de linie.
7	NUM	– numărul de zecimale este incorect specificat ($N < 0$ sau $N > 15$). Subprogramul NUM nu tipărește numărul C.
8	TEX	– numărul caracterelor incorect specificat ($NRCAR < 0$ sau $NRCAR > 80$) Subprogramul TEX nu tipărește șirul dat în vectorul TEXT
9	TEX	– parametrul de poziționare incorect specificat ($IPOZ < 0$ sau $IPOZ > 2$) textul nu va fi tipărit.
10	TEX	– în vectorul TEXT există un cod nepermis (nu este ASCII) textul nu va fi tipărit
11	PLG	– indicii I, J sînt incorecți definiți ($I \leq J$); nu se desenează nimic.
12	CSC	– raza cercului este mai mică de 0,3 mm; nu se desenează nimic.
13	CPP	– cele trei puncte sînt coliniare; se trage o dreaptă din punctul ($X 1, Y 1$) în punctul ($X 3, Y 3$).
14	SRA	– s-au specificat mai puțin de 3 puncte; nu se desenează nimic.
15	IPO	– numărul punctelor de interpolat este mai mic decît 3; nu se desenează nimic.
16	IPC	– IDEM eroare 15

Aplicații. Programe test



```

C PROGRAM TEST TRANSLATIE SI ROTATIE
CALL ASSIGN(1,'PP1')
CALL INI(1)
CALL AXI(80.,80.,5.,5.,1,1)
CALL RCT(40.,10.,60.,30.)
CALL BEG(10.,10.)
CALL DEG
CALL ROT(45.)
CALL BLH(4)
CALL AXI(80.,80.,5.,5.,1,1)
CALL BLH(3)
CALL RCT(40.,10.,60.,30.)
CALL EOF
STOP
END

```

Fig. 1.1

```

PROGRAM TEST  DESEN DE OPERATII
  CALL ASSIGN(1,'PP:1')
  CALL INT(1)
  DIMENSION X(16),Y(16)
  DATA X/15.,35.,40.,40.,35.,30.,20.,5.,.
    * 65.,85.,95.,80.,70.,65.,60.,60./
  DATA Y/15.,15.,20.,20.,35.,40.,90.,90.,55.,.
    1 15.,15.,55.,90.,90.,40.,35.,20./
C TRASEAZA AXELF DESENFUII
  CALL AXI(110.,110.,5.,5.,1,1)
C SCRIE TITLUL DESENULUI SI AXFLF
  CALL TEX(25.,100.,0.,4.,0,'DESEN DE OPERATII',17)
  DO 995 I=1,11
995 CALL NUM(10.*I,-5.,0.,2.5,1.10.*I,0)
  DO 996 I=1,11
996 CALL NUM(-3.,10.*I,0.,2.5,2.10.*I,0)
C DEFINESTE UNGHIIUL IN GRAF
  CALL DEG
C TRASEAZA CONTURUL PIESEI SI HASUREAZA
  CALL PLG(X,Y,1,8,1)
  CALL SRA(X,Y,1,8,45.,2.)
  CALL PLG(X,Y,9,16,1)
  CALL SRA(X,Y,9,16,45.,2.)
C COMPLETEAZA PIESEA
  CALL LIN(30.,90.,70.,90.)
  CALL LIN(35.,40.,65.,40.)
  CALL LIN(40.,35.,60.,35.)
  CALL LIN(40.,20.,60.,20.)
  CALL LIN(35.,15.,65.,15.)
  CALL BLH(1)
C TRASEAZA DREPTUNGHIIUL INTERIOR
  CALL RCT(25.,20.,75.,80.)
  CALL BLH(4)
C TRASEAZA AXA VERTICALA A PIESEI
  CALL LIN(50.,5.,50.,95.)
  CALL EOF
  STOP
  FND

```

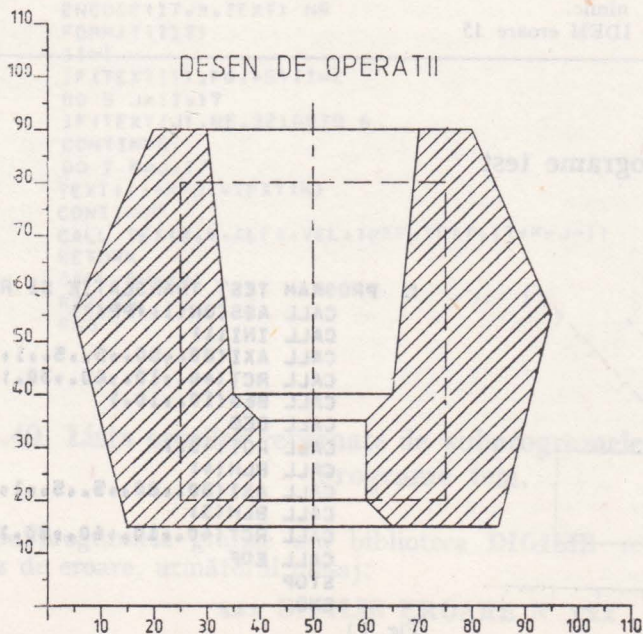
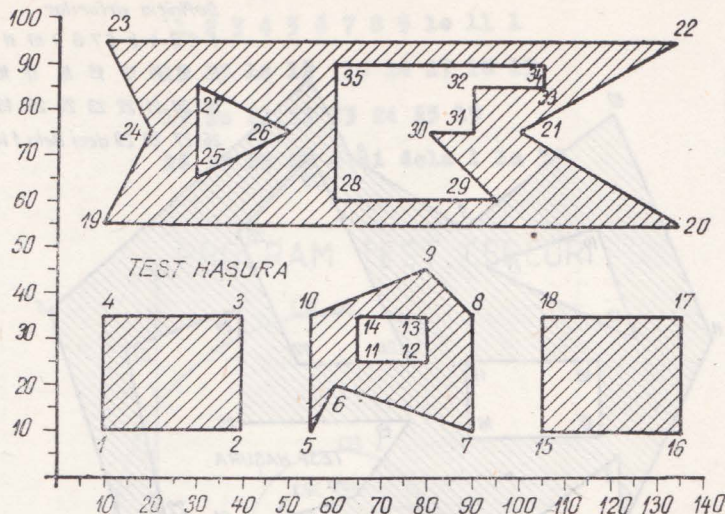


Fig. 1.2



Definitia virfurilor:

1 2 3 4 1 5 6 7 8 9 10 5 11 12 13 14 11

15 16 17 18 15 19 20 21 22 23 24 19

25 26 27 25 28 29 30 31 32 33 34 35

deci dela 1 la 41

Fig. 1.3

```

C PROGRAM TEST HASURA
  DIMENSION X(41),Y(41)
  DATA X/10.,40.,40.,10.,10.,
  * 55.,60.,90.,90.,80.,55.,55.,
  1 65.,80.,80.,65.,65.,
  2 105.,135.,135.,105.,105.,
  3 10.,135.,100.,135.,10.,20.,10.,
  4 30.,50.,30.,30.,
  5 60.,95.,80.,90.,105.,105.,60./
  DATA Y/10.,10.,35.,35.,10.,
  * 10.,20.,10.,35.,45.,35.,10.,
  1 25.,25.,35.,35.,25.,
  2 10.,10.,35.,35.,10.,
  3 55.,55.,75.,95.,95.,75.,55.,
  4 65.,75.,85.,65.,
  5 60.,60.,75.,75.,85.,85.,90.,90./
  CALL ASSIGN(1,'PP1')
  CALL INI(1)
C DESENEAZA AXELE
  CALL AXI(140.,100.,5.,5.,1.1)
C SCRIE TITLUL DESFNULUI SI AXELE
  CALL TEX(15.,45.,0.,4.,0.,'TEST HASURA',11)
  DO 995 I=1,14
  995 CALL NUM(10.*I,-5.,0.,2.5,1,10.*I,0)
  DO 996 I=1,10
  996 CALL NUM(-3.,10.*I,0.,2.5,2,10.*I,0)
C DEFINESTE UNGHIIUL IN GRADE
  CALL DEG
C TRASEAZA POLIGOANELE SI HASUREAZA
  CALL PLG(X,Y,1,41,1)
  CALL SRA(X,Y,1,41,45.,2.)
  CALL EOF
  STOP
  FND

```

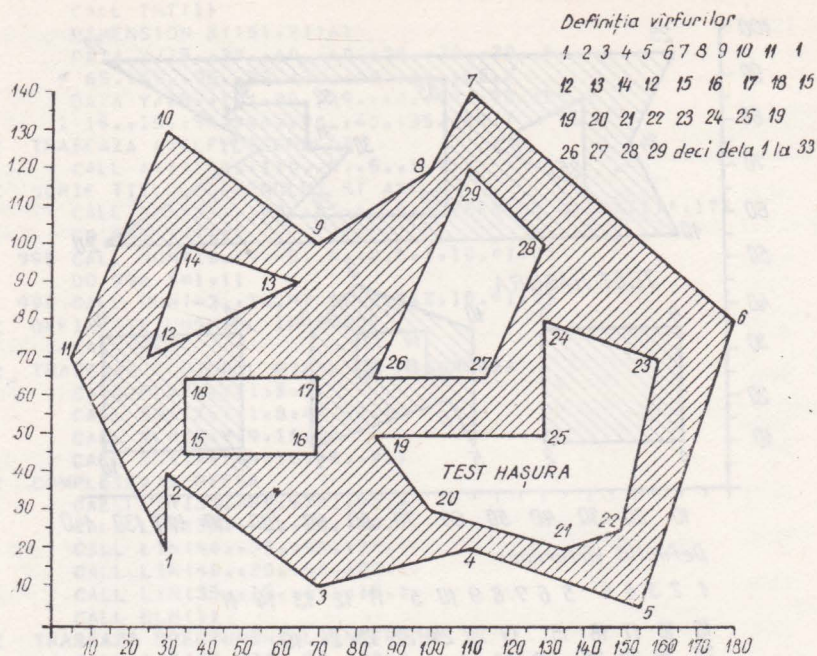


Fig. 1.4

```

C PROGRAM TEST HASURA
  DIMENSION X(33),Y(33)
  DATA X/30.,30.,70.,110.,155.,180.,110.,100.,70.,30.,5.,30.,
  * 25.,65.,35.,25.,
  1 35.,70.,70.,35.,35.,
  2 85.,100.,135.,150.,160.,130.,130.,85.,
  3 85.,115.,130.,110./
  DATA Y/20.,40.,10.,20.,5.,80.,140.,120.,180.,130.,70.,20.,
  * 70.,90.,100.,70.,
  1 45.,45.,65.,65.,45.,
  2 50.,30.,20.,25.,70.,80.,50.,50.,
  3 65.,65.,100.,120./
  CALL ASSIGN(1.,'PP:1')
  CALL INI(1)
C DESENEAZA AXELE
  CALL AXI(180.,140.,5.,5.,1,1)
C SCRIE TITLUL DESENULUI SI AXFLE
  CALL TEX(100.,40.,0.,4.,0.,'TEST HASURA',11)
  DO 995 I=1,18
  995 CALL NUM(10.*I,-5.,0.,2.5,1,10.*I,0)
  DO 996 I=1,14
  996 CALL NUM(-3.,10.*I,0.,2.5,2,10.*I,0)
C DEFINESTE UNGHIUL IN GRAD
  CALL DEG
C TRASFATA POLIGOANELE SI HASUREAZA
  CALL PLG(X,Y,1,33,1)
  CALL SRA(X,Y,1,33,65.,2.)
  CALL EOF
  STOP
  END

```

Definiția virfurilor :

1 2 3 4 5 6 7 8 9 10 11 1
 12 13 14 12 15 16 17 18 15
 19 20 21 22 23 24 25 19
 26 27 28 29 deci dela 1 la 33

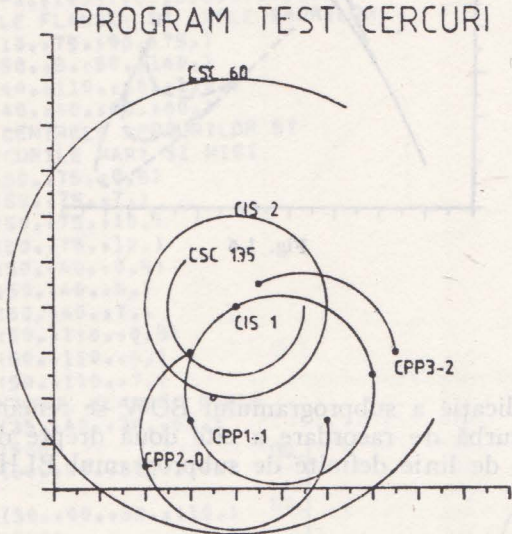


Fig. 1.5

```

C PROGRAM TEST CERC,CIS,CPP,CSC
  CALL ASSIGN(1,'PP:1')
  CALL INI(1)
  CALL AXI(100.,100.,5.,5.,1,1)
  CALL CIS(40.,40.,0.5)
  CALL CIS(40.,40.,20.)
  CALL CIS(30.,30.,0.5)
  CALL CIS(45.,45.,0.5)
  CALL CIS(30.,15.,0.5)
  CALL CIS(20.,5.,0.5)
  CALL CIS(35.,20.,0.5)
  CALL CIS(70.,25.,0.5)
  CALL CIS(60.,15.,0.5)
  CALL CIS(75.,30.,0.5)
  CALL DEG
  CALL CSC(40.,40.,50.,60.,330.)
  CALL CSC(40.,40.,15.,135.,360.)
  CALL CPP(40.,40.,30.,15.,70.,25.,1)
  CALL CPP(30.,30.,20.,5.,60.,15.,0)
  CALL CPP(45.,45.,35.,20.,75.,30.,2)
  CALL TEX(5.,100.,0.,6.,0,'PROGRAM TEST CERCURI',20)
  CALL TEX(40.,35.,0.,3.,0,'CIS 1',5)
  CALL TEX(40.,60.,0.,3.,0,'CIS 2',5)
  CALL TEX(30.,50.,0.,3.,0,'CSC 135',7)
  CALL TEX(30.,90.,0.,3.,0,'CSC 60-330',6)
  CALL TEX(35.,10.,0.,3.,0,'CPP1-1',6)
  CALL TEX(20.,5.,0.,3.,0,'CPP2-0',6)
  CALL TEX(75.,25.,0.,3.,0,'CPP3-2',6)
  CALL EOF
  STOP
  END
  
```

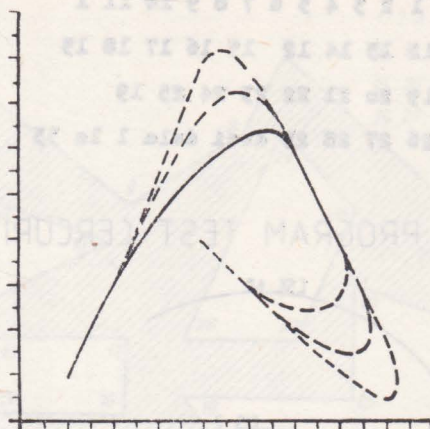


Fig. 1.6

În această aplicație a subprogramului BOW se remarcă utilizarea diferitelor arce de curbă de racordare a câte două drepte date, arcele fiind trasate cu tipurile de linie definite de subprogramul BLH.

```

C PROGRAM TEST
CALL ASSIGN(1,'PP:1')
CALL INT(1)
CALL AXI(90.,90.,5.,5.,1,1)
CALL LIN(10.,10.,20.,30.)
CALL ROW(500)
CALL LTN(60.,60.,70.,40.)
CALL BLH(1)
CALL NOT
CALL LIN(10.,10.,20.,30.)
CALL BOW(1000)
CALL NOT
CALL LTN(60.,60.,70.,40.)
CALL BLH(3)
CALL NOT
CALL LIN(10.,10.,20.,30.)
CALL ROW(1500)
CALL NOT
CALL LTN(60.,60.,70.,40.)
CALL LTN(40.,40.,50.,30.)
CALL BOW(500)
CALL NOT
CALL LTN(70.,40.,60.,60.)
CALL BLH(1)
CALL NOT
CALL LTN(40.,40.,50.,30.)
CALL BOW(1000)
CALL NOT
CALL LTN(70.,40.,60.,60.)
CALL BLH(3)
CALL NOT
CALL LTN(40.,40.,50.,30.)
CALL BOW(1500)
CALL NOT
CALL LTN(70.,40.,60.,60.)
CALL EOF
STOP
END
  
```



```

C PROGRAM TEST FLANSE OVALA
  CALL ASSIGN(1,'PP:1')
  CALL INI(1)
C TRASEAZA AXELE DE COORDONATE
  CALL AXI(90.,140.,5.,5.,1:1)
C SCRIE TITLUL DESENULUI SI AXELE
  CALL TEX(5.,5.,0.,4.,0.,'FLANSE OVALA',12)
  DO 995 I=1,9
995 CALL NUM(10.*I,-5.,0.,2.5,1,10.*I,0)
  DO 996 J=1,14
996 CALL NUM(-3.,10.*I,0.,2.5,2,10.*I,0)
C TRASEAZA AXELE FLANSET SI AXELE GAURILOR
  CALL LIN(10.,75.,90.,75.)
  CALL LIN(50.,5.,50.,140.)
  CALL LIN(40.,110.,60.,110.)
  CALL LIN(40.,40.,60.,40.)
C BALUSTREAZA CENTRELE CERCURILOR SI
C TRASEAZA CERCIILE MARI SI MICI
  CALL CIS(50.,75.,0,5)
  CALL CIS(50.,75.,7.)
  CALL CIS(50.,75.,10.)
  CALL CIS(50.,75.,12.)
  CALL CIS(50.,40.,0,5)
  CALL CIS(50.,40.,5.)
  CALL CIS(50.,40.,7.)
  CALL CIS(50.,110.,0,5)
  CALL CIS(50.,110.,5.)
  CALL CIS(50.,110.,7.)
C TRASEAZA CONTURUL FLANSET OVALE
  CALL LIN(35.,40.,30.,60.)
  CALL BOW(500)
  CALL LIN(30.,90.,35.,110.)
  CALL NOT
  CALL LIN(30.,90.,35.,110.)
  CALL ROW(750)
  CALL LIN(65.,110.,70.,90.)
  CALL NOT
  CALL LIN(65.,110.,70.,90.)
  CALL BOW(500)
  CALL LIN(70.,60.,65.,40.)
  CALL NOT
  CALL LIN(70.,60.,65.,40.)
  CALL BOW(750)
  CALL NOT
  CALL LIN(35.,40.,30.,60.)
  CALL EOF
  STOP
  END

```

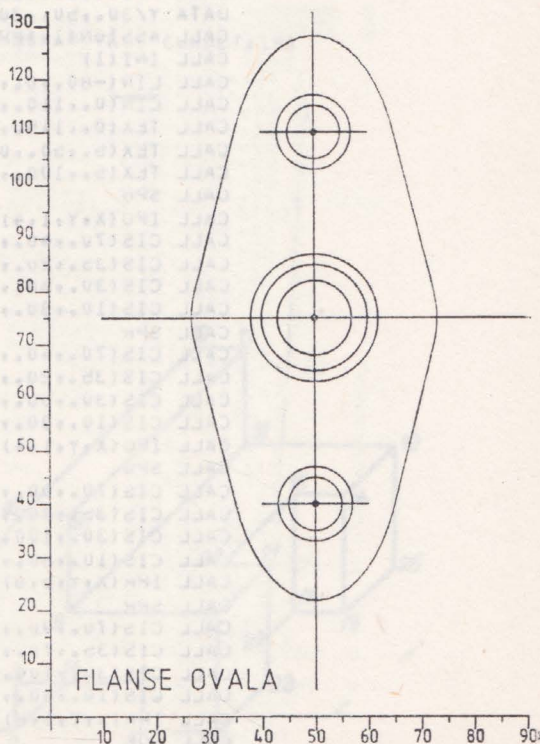


Fig. 1.7

TEST

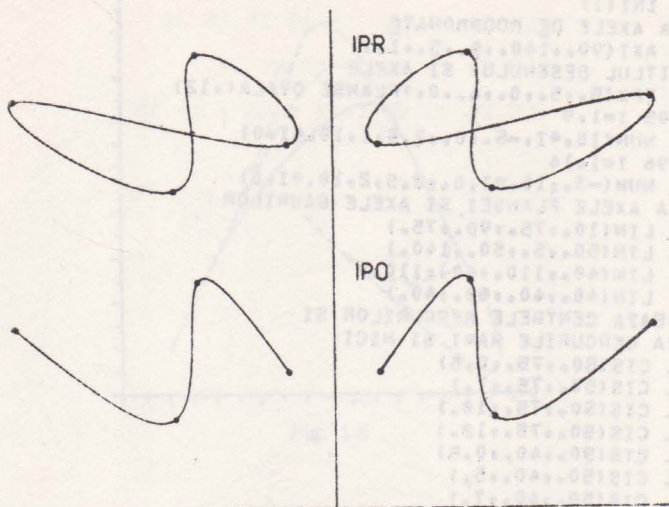


Fig. 1.8

```

PROGRAM TEST CURBE IPO SI IPR
DIMENSION X(8),Y(8)
DATA X/10.,30.,35.,70.,10.,30.,35.,70./
DATA Y/30.,50.,20.,40.,80.,100.,70.,90./
CALL ASSIGN(1,'PP:1')
CALL INI(1)
CALL LIN(-80.,0.,80.,0.)
CALL LINT(0.,110.,0.,0.)
CALL TEX(0.,115.,0.,8.,1,'TEST',4)
CALL TEX(5.,50.,0.,4.,0,'IPO',3)
CALL TEX(5.,100.,0.,4.,0,'IPR',3)
CALL SPG
CALL IPO(X,Y,1,4)
CALL CIS(70.,40.,0.5)
CALL CIS(35.,20.,0.5)
CALL CIS(30.,50.,0.5)
CALL CIS(10.,30.,0.5)
CALL SPR
CALL CIS(70.,40.,0.5)
CALL CIS(35.,20.,0.5)
CALL CIS(30.,50.,0.5)
CALL CIS(10.,30.,0.5)
CALL IPO(X,Y,1,4)
CALL SPG
CALL CIS(70.,90.,0.5)
CALL CIS(35.,70.,0.5)
CALL CIS(30.,100.,0.5)
CALL CIS(10.,80.,0.5)
CALL IPR(X,Y,5,8)
CALL SPR
CALL CIS(70.,90.,0.5)
CALL CIS(35.,70.,0.5)
CALL CIS(30.,100.,0.5)
CALL CIS(10.,80.,0.5)
CALL IPR(X,Y,5,8)
CALL EOF
STOP
END

```

PROGRAM TEST CURBE

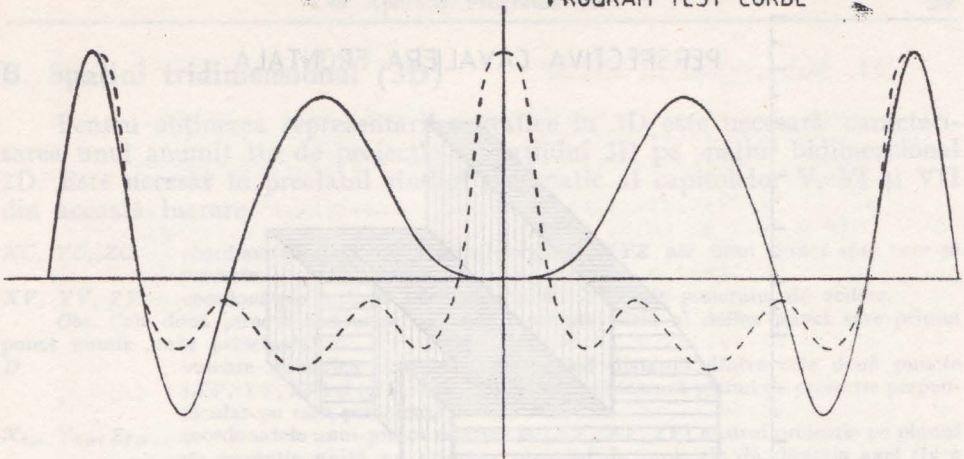


Fig. 1.9

C PROGRAM TEST CURBE

```

DIMENSION X(8),Y(8)
DATA X/0.,20.,40.,60.,70.,80.,90.,100./
DATA Y/0.,10.,40.,0.,-30.,0.,50.,0./
DIMENSION X1(11),Y1(11)
DATA X1/0.,5.,10.,20.,30.,45.,60.,70.,80.,85.,90./
DATA Y1/50.,40.,0.,-15.,-5.,-20.,-5.,-15.,0.,40.,50./
CALL ASSIGN(1,'PP:')
CALL INI(1)
CALL DEG
CALL TEX(10.,60.,0.,4.,0,'PROGRAM TEST CURBE',18)
CALL LIN(-110.,0.,110.,0.)
CALL LIN(0.,-40.,0.,60.)
CALL SPG
CALL IPO(X,Y,1,8)
CALL SPR
CALL IPO(X,Y,1,8)
CALL BLH(3)
CALL SPG
CALL IPO(X1,Y1,1,11)
CALL SPR
CALL IPO(X1,Y1,1,11)
CALL: EOF
STOP
END
    
```

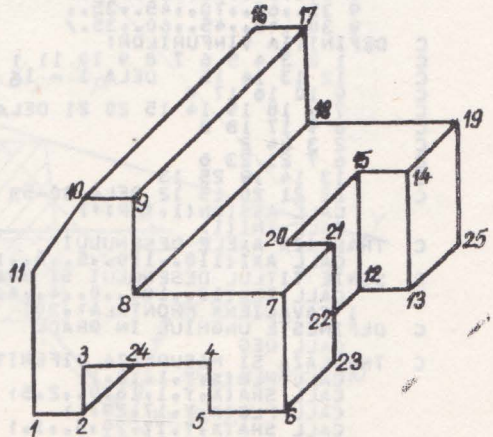


Fig. 1.10 a

PERSPECTIVA CAVALIERA FRONTALA

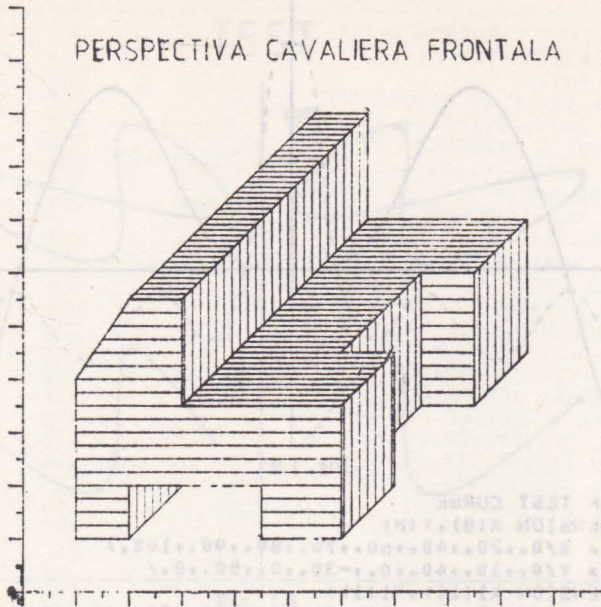


Fig. 1.10 b

```

C PROGRAM TEST
C PERSPECTIVA CAVALIERA FRONTALA
  DIMENSION X(53),Y(53)
  DATA X/10.,20.,20.,45.,45.,60.,60.,30.,30.,20.,10.,10.,
1 75.,85.,85.,75.,
2 30.,20.,55.,65.,30.,
3 60.,30.,65.,95.,85.,75.,60.,70.,
4 30.,30.,65.,65.,30.,
5 20.,20.,30.,20.,
6 60.,60.,70.,70.,60.,
7 85.,85.,95.,95.,85.,
8 70.,70.,60.,75.,75./
  DATA Y/10.,10.,20.,20.,10.,10.,35.,35.,55.,55.,40.,10.,
9 35.,35.,60.,60.,
10 55.,55.,90.,90.,55.,
11 35.,35.,70.,70.,60.,60.,45.,45.,
12 35.,55.,90.,70.,35.,
13 10.,20.,20.,10.,
14 10.,35.,45.,20.,10.,
15 35.,60.,70.,45.,35.,
16 30.,45.,45.,60.,35./
C DEFINITIA VIRFURILOR:
1 2 3 4 5 6 7 8 9 10 11 ;
12 13 14 15 DELA 1 - 16
9 10 16 17 9
7 8 16 19 14 15 20 21 DELA 17-29
8 9 17 18 8
0 0 3 4 5
0 0 21 23 6
13 14 19 25 13
C 22 21 20 15 12 DELA 30-53
  CALL ASSIGN(1,'PP:1')
  CALL INI(1)
C TRASEAZA AXELE DESENULUI
  CALL AXI(110.,110.,5.,5.,1,1)
C SCRIE TITLUL DESENULUI SI GRADEAZA AXELE
  CALL TEX(10.,100.,0.,4.,8,'PERSPECTIVA
1 CAVALIERA FRONTALA',30)
C DEFINESTE UNghiUL IN GRADE
  CALL DEG
C TRASEAZA SI MASUREAZA DIFERIT PIESA
  CALL PLG(X,Y,1,16,1)
  CALL SRA(X,Y,1,16,0.,2.5)
  CALL PLG(X,Y,17,29,1)
  CALL SRA(X,Y,17,29,0.,1.)
  CALL PLG(X,Y,30,53,1)
  CALL SRA(X,Y,30,53,90.,1.5)
  CALL EOF
  STOP
  END

```

B. Spațiul tridimensional (3D)

Pentru obținerea reprezentărilor grafice în 3D este necesară caracterizarea unui anumit tip de proiecție a spațiului 3D pe spațiul bidimensional 2D. Este necesar în prealabil studiul sistematic al capitolelor V, VI și VII din această lucrare.

XC, YC, ZC coordonatele față de triedrul original XYZ ale unui punct spre care se privește.

XV, YV, ZV coordonatele față de triedrul original XYZ ale punctului de vedere.

Obs. Cele două puncte alcătuiesc un vector orientat dela al doilea punct spre primul punct numit „raăz principală“.

D valoare numerică pozitivă reprezentind distanța dintre cele două puncte (XV, YV, ZV) și (XC, YC, ZC) la care se plasează planul de proiecție perpendicular pe raza principală.

$X_{spr}, Y_{spr}, Z_{spr}$ coordonatele unui punct relative la (XY, YV, ZV) a cărui proiecție pe planul de proiecție unită cu originea planului de proiecție dă direcția axei Oy a noului triedru. Originea O'' a planului de proiecție se alege la intersecția razei principale cu planul de proiecție. Axa $O'z'$ a noului triedru este chiar raza principală iar sensul axei este dela O' către (XV, YV, ZV) .

D_x, D_y valori numerice pozitive reprezentind mărimea unui sfert din fereastra din planul de proiecție față de care se execută procedura de clipping. Fereastra este un dreptunghi cu laturile paralele cu axele $O'x'$ și $O'y'$ ale noului triedru și cu mijlocul în O' .

Obs. Toate elementele sistemului de proiecție pot fi urmărite pe figura 1.11.

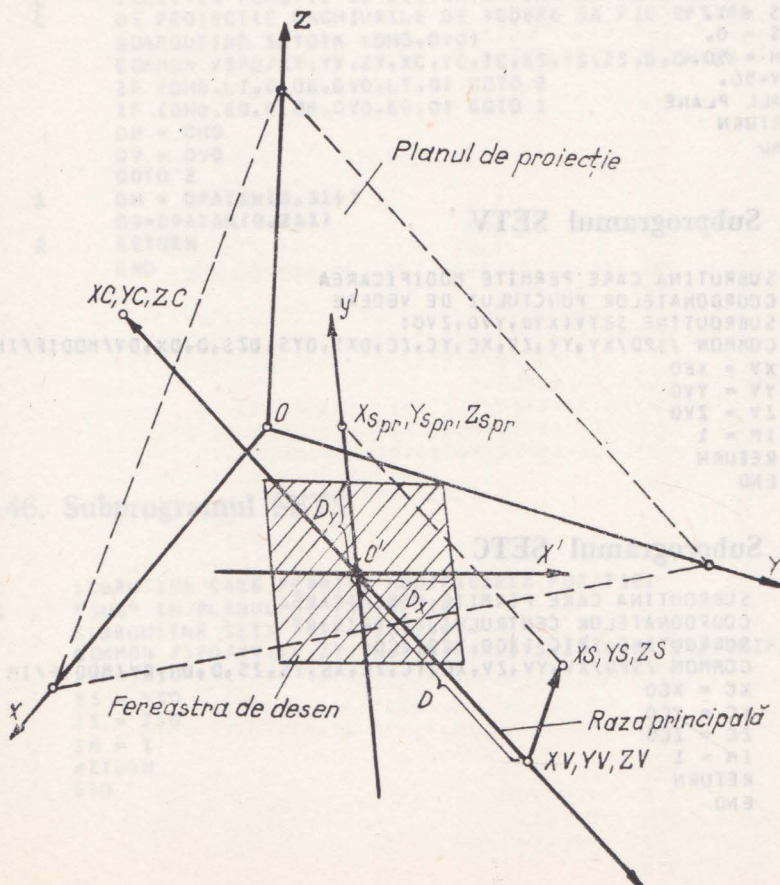


Fig. 1.11

1.41. Subprogramul INISP

```

SUBROUTINA PENTRU INITIALIZAREA VARIABILELOR SPATIALE
SUBROUTINE INISP
COMMON /SPR/CX1X,CX1Y,CX1Z,CY1X,CY1Y,CY1Z,CZ1C,CZ1Y,CZ1Z,
* XO,YO,ZO/
*   SPD/XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,D,DH,DV/MODIF/IM
C   SPR - BLOC DE COMUN IN CARE SE VOR PLASA COSINUSII DIRECTORI
C   AI AXELOR NOULUI SISTEM DE COORDONARE SI DEPLASAMENTUL
C   ORIGINII NOULUI SISTEM
C   SPD - BLOC DE COMUN IN CARE SE VOR PLASA DATELE SISTEMULUI
C   DE PROIECTIE
C   XV,YV,ZV, - COORDONATE PUNCT DE VEDERE
C   XC,YC,ZC, - COORDONATE CENTRU DE PRIVIRE
C   XS,YS,ZS- COORDONATE RELATIVE LA XV,YV,ZV CARE DAU
C   DIRECTIA SUS PE PLANUL DE PROIECTIE
C   D -DISTANTA DE LA XV,YV,ZV LA PLANUL DE PROIECTIE
C   DH,DV -DIMENSIUNILE IMAGINII IN PLANUL DE PROIECTIE

XC = 0.
YC = 0.
ZC = 0.
XV = 0.
YV = 0.
ZV = 100.
D = 100.
XS = 0.
YS = 1.
ZS = 0.
DH = 50.
DV = 50.
CALL PLANE
RETURN
END

```

1.42. Subprogramul SETV

```

C   SUBROUTINA CARE PERMITE MODIFICAREA
C   COORDONATELOR PUNCTULUI DE VEDERE
SUBROUTINE SETV(XVO,YVO,ZVO)
COMMON /SPD/XV,YV,ZV,XC,YC,ZC,DXS,DYS,DZS,D,DX,DY/MODIF/IM
XV = XVO
YV = YVO
ZV = ZVO
IM = 1
RETURN
END

```

1.43. Subprogramul SETC

```

C   SUBROUTINA CARE PERMITE MODIFICAREA
C   COORDONATELOR CENTRULUI DE PRIVIRE
SUBROUTINE SETC(XCO,YCO,ZCO)
COMMON /SPD/XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,D,DH,DV/MODIF/IM
XC = XCO
YC = YCO
ZC = ZCO
IM = 1
RETURN
END

```

1.44. Subprogramul SETD

```

C   SUBROUTINA CARE PERMITE MODIFICAREA
C   DISTANTEI PINA LA PLANUL DE PROIECTIE
SUBROUTINE SETD (DO)
COMMON /SPD/XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,D,DH,DV/MODIF/IM
IF(DO.EQ.0) GOTO 1
D=DO
IM=1
1   RETURN
END

```

1.45. Subprogramul SETDIM

```

C   SUBROUTINA CARE PERMITE MODIFICAREA
C   DIMENSIUNILOR FERESTREI DIN PLANUL DE PROIECTIE.
C   DACA APELUL SE FACE FARA PARAMETRI ATUNCI
C   SE CALCULEAZA PENTRU DX SI DY VALORI ASTFEL
C   INCIT IN FUNCTIE DE DISTANTA PINA LA PLANUL
C   DE PROIECTIE UNghiURILE DE VEDERE SA FIE OPTIME
SUBROUTINE SETDIM (DHO,DVO)
COMMON /SPD/XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,D,DH,DV
IF (DHO.LT.0.OR.DVO.LT.0) GOTO 2
IF (DHO.EQ.0.OR.DVO.EQ.0) GOTO 1
DH = DHO
DV = DVO
GOTO 2
1   DH = D*ATAN(0.314)
   DV = D*ATAN(0.261)
2   RETURN
END

```

1.46. Subprogramul SETS

```

C   SUBROUTINA CARE PERMITE MODIFICAREA POZITIEI
C   "SUS" IN PLANUL DE PROIECTIE
SUBROUTINE SETS (XS0,YS0,ZS0)
COMMON /SPD/XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,D,DH,DV/MODIF/IM
XS = XS0
YS = YS0
ZS = ZS0
IM = 1
RETURN
END

```

1.47. Subprogramul PLANE

```

SUBROUTINA CARE CALCULEAZA CUSINUSII DIRECTORI AI
AXELOR NUULUI SISTEM
SUBROUTINE PLANE
COMMON /SPD/XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,D,DH,DV
*   /SPR/CX1X,CX1Y,CX1Z,CY1X,CY1Y,CY1Z,CZ1X,CZ1Y,CZ1Z,
*   XO,YO,ZO/MODIF/IM
DELT=SQRT((XV-XC)*(XV-XC)+(YV-YC)*(YV-YC)+(ZV-ZC)*(ZV-ZC))
CZ1X=(XV-XC)/DELT
CZ1Y=(YV-YC)/DELT
CZ1Z=(ZV-ZC)/DELT
XO=XC+CZ1X*(DELT-D)
YO=YC+CZ1Y*(DELT-D)
ZO=ZC+CZ1Z*(DELT-D)
PS=XO*CX1X+YO*CY1Y+ZO*CZ1Z
P1=XS-CZ1X*PS
P2=YS-CZ1Y*PS
P3=ZS-CZ1Z*PS
DELT1=SQRT(P1*P1+P2*P2+P3*P3)
CY1X=P1/DELT1
CY1Y=P2/DELT1
CY1Z=P3/DELT1
CX1X=CY1Y*CZ1Z-CY1Z*CZ1Y
CX1Y=CY1Z*CZ1X-CY1X*CZ1Z
CX1Z=CY1X*CZ1Y-CY1Y*CZ1X
IM=0
RETURN
END

```

1.48. Subprogramul NEWC

```

C   SUBROUTINA PENTRU CALCULUL NOILOR COORDONATE
C   IN SISTEMUL IMAGINE
SUBROUTINE NEWC(X,Y,Z,X1,Y1,Z1)
COMMON /SPR/CX1X,CX1Y,CX1Z,CY1X,CY1Y,CY1Z,CZ1X,CZ1Y,CZ1Z,
*   XO,YO,ZO/MODIF/IM
IF(IM.EQ.1) CALL PLANE
X1=(X-XO)*CX1X+(Y-YO)*CX1Y+(Z-ZO)*CX1Z
Y1=(X-XO)*CY1X+(Y-YO)*CY1Y+(Z-ZO)*CY1Z
Z1=(X-XO)*CZ1X+(Y-YO)*CZ1Y+(Z-ZO)*CZ1Z
RETURN
END

```

1.49. Subprogramul PRO

```

C   SUBROUTINA PENTRU OBTINEREA PROIECTIEI UNUI PUNCT IN
C   PLANUL-IMAGINE
SUBROUTINE PRO(X1,Y1,Z1,XP,YP,I)
C   PENTRU I=0 SE OBTINE PROIECTIA PARALELA
C   I#0 SE OBTINE PROIECTIA CENTRALA
COMMON /SPD/XV,YV,ZV,XC,YC,ZC,XS,YS,ZS,D,DH,DV
XP=X1
YP=Y1
IF(I.EQ.0) RETURN
XP=XP*U/(D-Z1)
YP=YP*U/(D-Z1)
RETURN
END

```


1.50. Subprogramul CLIPS

```

C      PROCEDURA PENTRU REALIZAREA CLIPPINGULUI IN PLANUL DE
C      PROIECTIE (FEREAȘTRA DE DESEN)
C      X1,Y1,Z1=COORDONATELE UNUI CAPAT DE SEGMENT IN TRIEDRUL
C      ATASAT PLANULUI DE PROIECTIE
C      X2,Y2,Z2=COORDONATELE CELUIALTI CAPAT AL SEGMENTULUI
C      XP1,YP1,XP2,YP2=COORDONATELE PROIECTIILOR PE PLAN
C      I = TIPUL PROIECTIEI (DESCRIS ANTERIOR)
C      NP = INDICATOR DE LIPSA A PROIECTIEI
SUBROUTINE CLIPS(X1,Y1,Z1,X2,Y2,Z2,I,XP1,YP1,XP2,YP2,NP)
COMMON /SPU/XV,YX,ZV,XC,YC,ZC,XS,YS,ZS,D,DX,DY
DIMENSION XI(2),YI(2),XK(2)
DATA EPS/0.01/
NP=0
1  XT1=X1
   YT1=Y1
   ZT1=Z1
   XT2=X2
   YT2=Y2
   ZT2=Z2
   IF(I.EQ.0)GOTO 4
   IT=1
   IF(Z1.GE.0) IT=IT+1
   IF(Z2.GE.0) IT=IT+2
   GOTO (4,2,3,1),IT
2  NP=1
   RETURN
3  XK1=(D-Z1-EPS)/(Z2-Z1)
   XT1=X1+XK1*(X2-X1)
   YT1=Y1+XK1*(Y2-Y1)
   ZT1=D-EPS
   GOTO 4
4  XK1=(D-Z1-EPS)/(Z2-Z1)
   XT2=X1+XK1*(X2-X1)
   YT2=Y1+XK1*(Y2-Y1)
   ZT2=D-EPS
   CALL PRO(XT1,YT1,ZT1,XP1,YP1,I)
   CALL PRO(XT2,YT2,ZT2,XP2,YP2,I)
   IT=1
   IF(ABS(XP1).LE.DX.AND.ABS(YP1).LE.DY)IT=IT+1
   IF(ABS(XP2).LE.DX.AND.ABS(YP2).LE.DY)IT=IT+2
   IF(IT.EQ.4)RETURN
   INT=1
   IF(XP1.EQ.XP2)GOTO 6
   XK1=(DX-XP1)/(XP2-XP1)
   IF(XK1.LT.0..OR.XK1.GT.1.)GOTO 5
   YI(INT)=YP1+XK1*(YP2-YP1)
   IF(ABS(YI(INT)).GT.DY)GOTO 5
   XI(INT)=DX
   IF(IT.GT.1)GOTO 9
   XK(INT)=XK1
   INT=INT+1
5  XK1=(-DX-XP1)/(XP2-XP1)
   IF(XK1.LT.0..OR.XK1.GT.1.)GOTO 6
   YI(INT)=YP1+XK1*(YP2-YP1)
   IF(ABS(YI(INT)).GT.DY)GOTO 6
   XI(INT)=-DX

```

```

IF(IT.GT.1)GOTO 9
XK(INT)=XK1
IF(INT.EQ.2)GOTO 12
INT=INT+1
6 IF(YP1.EQ.YP2)GOTO 8
XK1=(DY-YP1)/(YP2-YP1)
IF(XK1.LT.0..OR.XK1.GT.1)GOTO 7
X1(INT)=XP1+XK1*(XP2-XP1)
IF(ABS(X1(INT)).GT.DX)GOTO 7
Y1(INT)=DY
IF(IT.GT.1)GOTO 9
XK(INT)=XK1
IF(INT.EQ.2)GOTO 12
INT=INT+1
7 XK1=(-DY-YP1)/(YP2-YP1)
IF(XK1.LT.0..OR.XK1.GT.1)GOTO 8
X1(INT)=XP1+XK1*(XP2-XP1)
IF(ABS(X1(INT)).GT.DX)GOTO 8
Y1(INT)=-DY
IF(IT.GT.1)GOTO 9
XK(INT)=XK1
GOTO 12
8 NP=1
RETURN
9 GOTO(10,11),IT-1
10 YP2=XI(1)
YP2=YI(1)
RETURN
11 XP1=XI(1)
YP1=YI(1)
RETURN
12 IF(XK(1)-XK(2))13,8,14
13 XP1=XI(1)
YP1=YI(1)
XP2=XI(2)
YP2=YI(2)
RETURN
14 XP1=XI(2)
YP1=YI(2)
XP2=XI(1)
YP2=YI(1)
RETURN
END

```

1.51. Program principal aplicativ pentru utilizarea softului de rutine grafice în spațiul 3D

```

DIMENSION X1(50),Y1(50),Z1(50),X11(50),Y11(50),Z11(50)
COMMON /SPR/CX1X,CX1Z,CY1X,CY1Z,CZ1X,CZ1Y,CZ1Z,XC,YC,Z
CALL ASSIGN(2,'DT:')
CALL INI(2)
CALL INISP
TYPE 1
1 FORMAT(' DATE PENTRU SISTEM DE PROIECTIE',/
* ' XC,YC,ZC,XS,YS,ZS,D,DH,DV')
2 ACCEPT 2,XCO,YCO,ZCO,XSO,YSO,ZSO,DO,DHO,DVO
FORMAT(9F10.3)
CALL SETC(XCO,YCO,ZCO)
CALL SETD(DO)
CALL SETDIM(DHO,DVO)
CALL SETS(XSO,YSO,ZSO)
200 TYPE 111
111 FORMAT(' XV,YV,ZV= ',S)
ACCEPT 2,XVO,YVO,ZVO
IF(XVO.EQ.-1000.) GOTO 300
CALL SETV(XVO,YVO,ZVO)
CALL PLANE
CALL TRA(2.1*DHO,0.)
CALL RCT(-DHO,-DVO,DHO,DVO)
CALL ASSIGN(1,'A.DAT')
READ(1,11)NRP
11 FORMAT(I2)
DO 12 J=1,NRP
READ(1,13)X1(J),Y1(J),Z1(J)
13 FORMAT(3F10.5)
CALL NEWC(X1(J),Y1(J),Z1(J),X11(J),Y11(J),Z11(J))
12 CONTINUE
READ(1,11) NRS
DO 14 J=1,NRS
READ(1,15)NR1,NR2
CALL CLIPS(X11(NR1),Y11(NR1),Z11(NR1),X11(NR2),Y11(NR2),
* Z11(NR2),1,X1(NR1),Y1(NR1),X1(NR2),Y1(NR2),NP)
IF(NP.EQ.0) CALL LIN(X1(NR1),Y1(NR1),X1(NR2),Y1(NR2))
15 FORMAT(2I3)
14 CONTINUE
CALL CLOSE(1)
GOTO 200
300 STUP
END

```

1.52. Aplicații

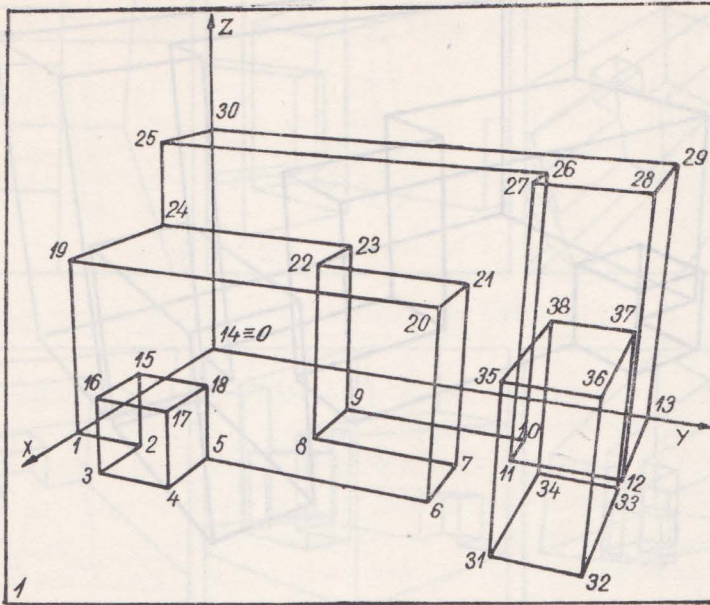


Fig. 1.12

COORDONATE

VERTICALE ȘI SEGMENTE

P	X	Y	Z	P	X	Y	Z
1	200	0	0	20	200	250	120
2	200	50	0	21	140	250	120
3	250	50	0	22	140	150	120
4	250	100	0	23	80	150	120
5	200	100	0	24	80	0	120
6	200	250	0	25	80	0	180
7	140	250	0	26	80	280	180
8	140	150	0	27	120	280	180
9	80	150	0	28	120	350	180
10	80	280	0	29	0	350	180
11	120	280	0	30	0	0	180
12	120	350	0	31	250	300	0
13	0	350	0	32	250	350	0
14	0	0	0	33	130	350	0
15	200	50	50	34	130	300	0
16	250	50	50	35	250	300	100
17	250	100	50	36	250	350	100
18	200	100	50	37	130	350	100
19	200	0	120	38	130	300	100

1-2	1-19	15-16	31-35
2-3	2-15	16-17	32-36
3-4	3-16	17-18	33-37
4-5	4-17	18-15	34-38
5-6	5-18	19-20	35-36
6-7	6-20	20-21	36-37
7-8	7-21	21-22	37-38
8-9	8-22	22-23	38-35
9-10	9-23	23-24	
10-11	10-26	24-19	
11-12	11-27	25-26	
12-13	12-28	26-27	
13-14	13-29	27-28	
14-1	14-30	28-29	
		29-30	
31-32	33-34	30-25	
32-33	34-31	24-25	

POLIGOANE

- 1, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1
- 16, 17, 18, 15, 16
- 19, 20, 21, 22, 23, 24, 19
- 25, 26, 27, 28, 29, 30, 25
- 3, 4, 17, 16, 3
- 1, 2, 15, 18, 5, 6, 20, 19, 1

- 7, 8, 22, 21, 7
- 11, 12, 28, 27, 11
- 24, 23, 9, 10, 26, 25, 24
- 14, 30, 29, 13, 14
- 1, 19, 25, 24, 30, 14, 1
- 3, 16, 15, 2, 3
- 4, 5, 18, 17, 4

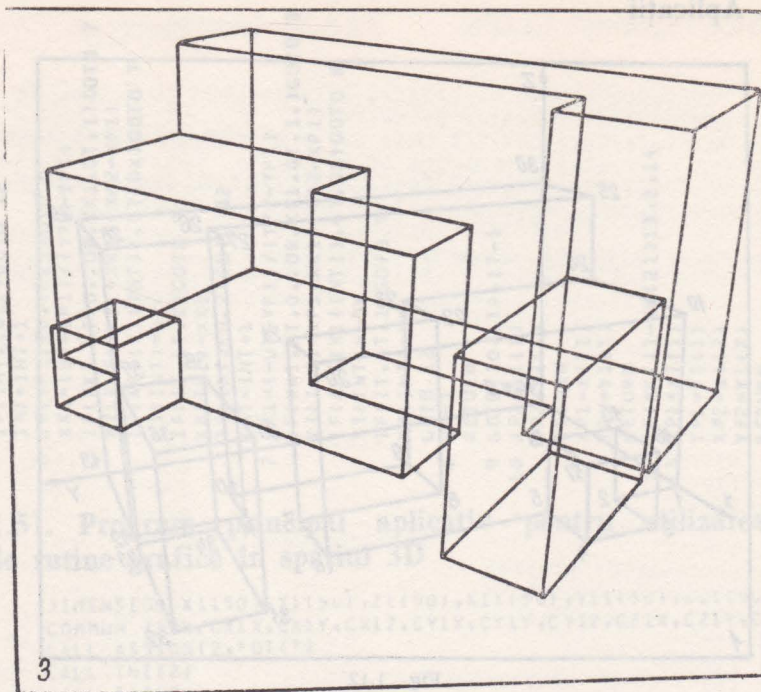


Fig. 1.13

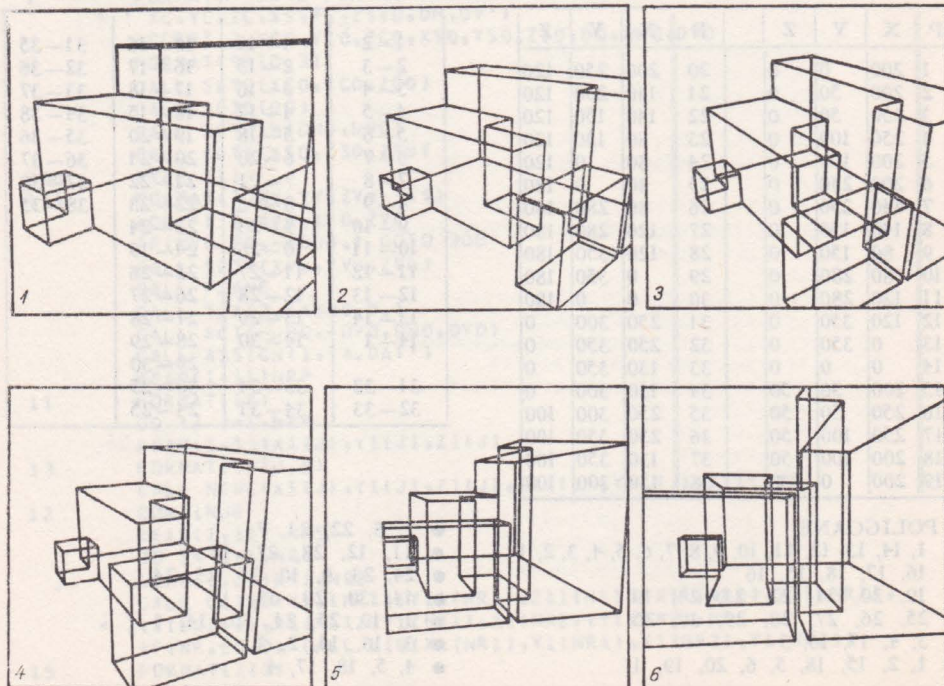


Fig. 1.14

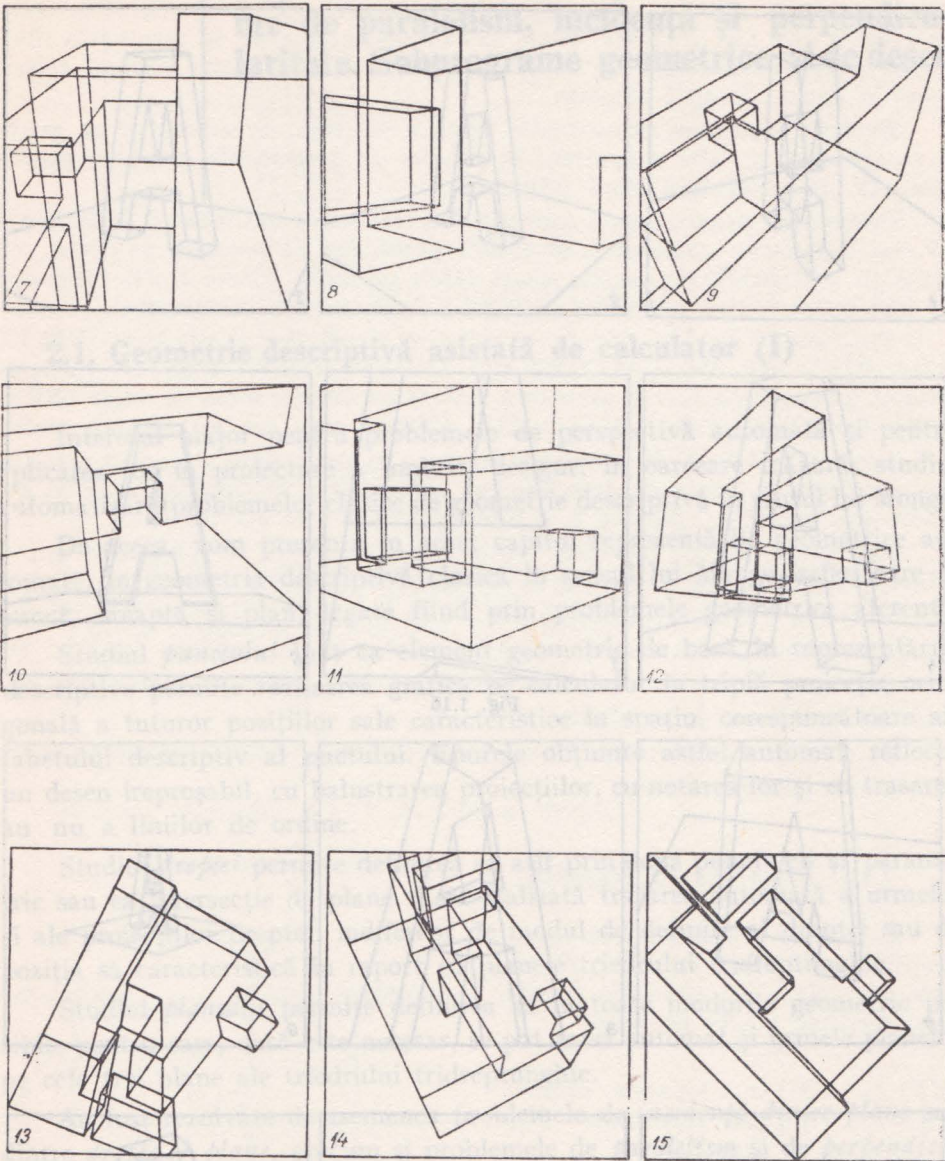
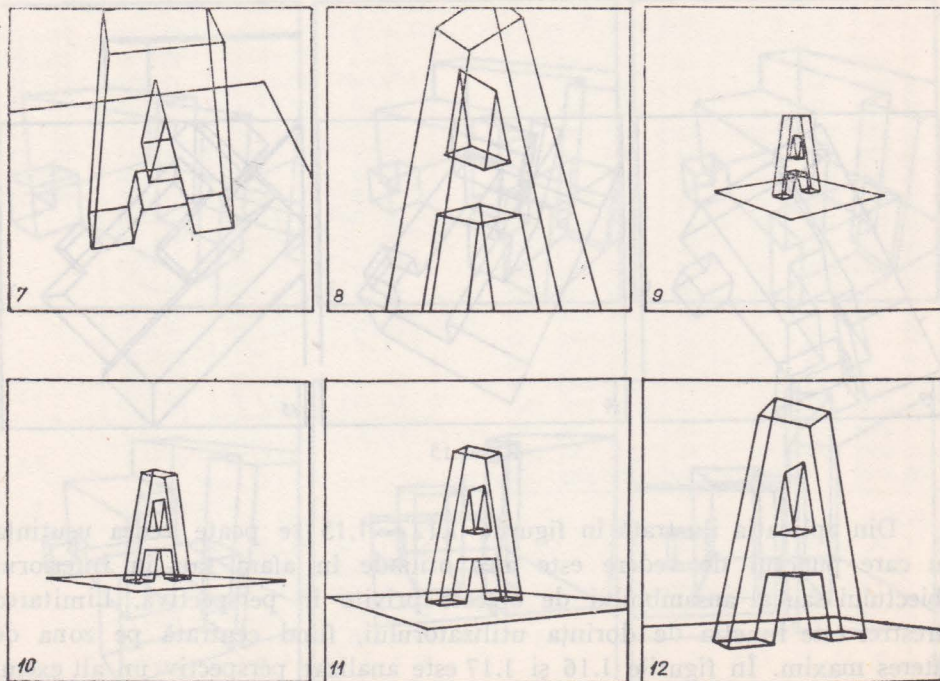
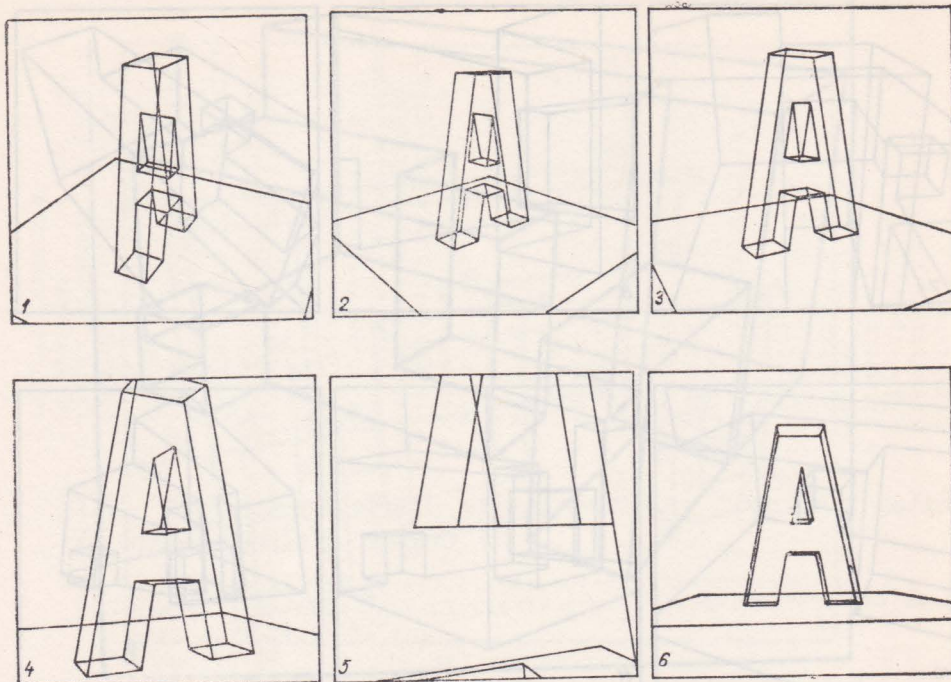


Fig. 1.15

Din aplicația ilustrată în figurile 1.12 — 1.15 se poate vedea ușurința cu care punctul de vedere este ales oriunde în afara sau în interiorul obiectului sau al ansamblului de obiecte privite în perspectivă. Limitarea ferestrei este funcția de dorința utilizatorului, fiind centrată pe zona de interes maxim. În figurile 1.16 și 1.17 este analizat perspectiv un alt exemplu, ca aplicație.



Reprezentări automate în geometria descriptivă. Punctul. Dreapta. Planul. Probleme de paralelism, incidență și perpendicularitate. Subprograme geometrice și de desen

2.1. Geometrie descriptivă asistată de calculator (I)

Interesul major pentru problemele de perspectivă automată și pentru aplicarea lor în proiectare a umbrit, desigur, în oarecare măsură, studiul automatizării problemelor clasice de geometrie descriptivă în sensul lui Monge.

De aceea, vom prezenta în acest capitol reprezentările geometrice automate, în geometria descriptivă clasică în sensul lui Monge, referitoare la punct, dreaptă și plan, legate fiind prin problemele geometrice aferente.

Studiul *punctului* luat ca element geometric de bază în reprezentările descriptive permite realizarea grafică pe calculator în triplă proiecție ortogonală a tuturor pozițiilor sale caracteristice în spațiu, corespunzătoare alfabetului descriptiv al punctului. Epurele obținute astfel automat reflectă un desen ireproșabil, cu balustrarea proiecțiilor, cu notarea lor și cu trasarea au nu a liniilor de ordine.

Studiul *dreptei* permite definirea sa atât prin două puncte cât și parametric sau ca intersecție de plane. Este realizată trasarea automată a urmelor și ale proiecțiilor dreptei, indiferent de modul de definire al dreptei sau de poziția sa caracteristică în raport cu planele triedrului tridreptunghic.

Studiul *planului* permite definirea sa în toate modurile geometrice posibile pentru care, dacă este necesar, se pot trasa automat și urmele planelor pe cele trei plane ale triedrului tridreptunghic.

Au fost rezolvate de asemenea problemele de *incidență dintre plane* sau *dintre drepte și plane*, precum și problemele de *paralelism* și de *perpendicularitate între plane și dintre drepte și plane*.

Problemele *metrice* privind determinarea diferitelor distanțe dintre elementele geometrice punct, dreaptă, plan sau determinarea bisectoarelor unui unghi fac parte integrantă din acest studiu.

Este realizată astfel o transpunere a principalelor rezolvări ale problemelor geometrice și de geometrie descriptivă pe calculatorul electronic, cu desenaerea automată a epurelor în triplă proiecție ortogonală.

Acest lucru a fost posibil prin definirea unor mulțimi de subrutine (subprograme) care sînt chemate, odată sau de mai multe ori, în rezolvarea diferitelor probleme specifice de astădată geometriei descriptive clasice, referitoare la punct, dreptă și plan.

Evident mulțimea subrutilnelor rămîne deschisă, în sensul că ea poate fi completată, cu alte subrutine, în funcție de necesitățile ce apar, pentru moment, în rezolvarea unei probleme.

Programele pentru aceste subrutine au fost scrise în limbajul FORTRAN IV, iar din forma apelului și structura problemei de rezolvat se deduc, cu ușurință, atît datele de intrare, cît și datele de ieșire. Unele dintre programe au fost scrise pentru realizarea dialogului interactiv cu calculatorul.

Programele au fost testate pe diferite tipuri de calculatoare (FELIX 1024, PDP, IBM 370, CALCOMP 925, INDEPENDENT etc.), iar execuția grafică a epurelor a fost de asemenea testată pe diferite mese de desen (BENSON, CALCOMP 947, ARISTO, HP, KOWO etc.) sau pe display grafic.

Acest set de bază de programe servește în continuare și pentru rezolvarea problemelor legate de determinarea automată a secțiunilor plane în poliedre sau suprafețe, precum și a intersecțiilor dintre acestea.

Este important de remarcat faptul că aceste seturi de programe pot fi concepute, în variante simple sau în variante extinse, adică o anumită subrutină poate rezolva o singură problemă pentru un singur rînd de date sau, dimpotrivă, ea poate rezolva aceeași problemă pentru un șir multiplu de date. Diferențele apar și în afișarea datelor de ieșire, în funcție de necesitățile impuse de problema de rezolvat.

În studiile și testările pe care le-am efectuat am folosit ambele variante ale setului de subrutine.

2.1.1. Subprogramul REORPU

Calculează coordonatele proiecțiilor punctului $P(X, Y, Z)$ pe cele trei plane de proiecție (eventual trasează aceste proiecții)

$XP1 = -X$ } coordonatele proiec-
 $YP1 = -Y$ } ției orizontale $P1$,
 (fig. 2.1)

$XP2 = -X$ } coordonatele proiec-
 $YP2 = Z$ } ției verticale $P2$,

$XP3 = Y$ } coordonatele proiec-
 $YP3 = Z$ } ției laterale $P3$ ale
 punctului P

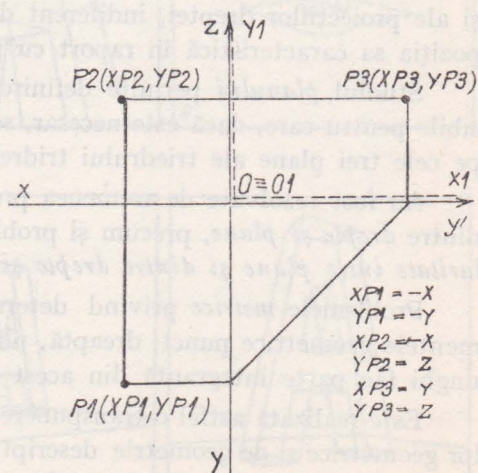


Fig. 2.1.


```

SUBROUTINE REORPU(X,Y,Z,XP,YP,T)
LOGICAL*1 T(1)
DIMENSION XP(3),YP(3)
XP(1)=-X
YP(1)=-Y
XP(2)=-X
YP(2)=Z
XP(3)=Y
YP(3)=Z
DO 1 I=1,3
CALL CIS(XP(I),YP(I),.5)
T(2)=48+I
CALL TEX(XP(I)+1.,YP(I)+1.,0.,2.5,0,T,2)
1 CONTINUE
RETURN
END

```

2.1.2. Subprogramul REORDR

Efectuează trasarea dreptei definite de două puncte.

De asemenea, determină dacă este cazul și urmele dreptei pe cele trei plane de proiecție deduse din următoarele relații:

— urma orizontală $H(XH, YH, O)$ a dreptei

$$XH = -Z1 \frac{AL}{AN} + X1$$

$$YH = -Z1 \frac{AM}{AN} + Y1$$

— urma verticală $V(XV, O, ZV)$ a dreptei

$$XV = -Y1 \frac{AL}{AM} + X1$$

$$ZV = -Y1 \frac{AN}{AM} + Z1$$

— urma laterală $W(O, YW, ZW)$ a dreptei

$$YW = -X1 \frac{AM}{AL} + Y1$$

$$ZW = -X1 \frac{AN}{AL} + Z1$$

```

SUBROUTINE REORDR(XT, YT, ZT, XR, YR, ZR, XP, YP, XQ, YQ, T, U)
LOGICAL*1 T(1), U(1)
DIMENSION XP(3), YP(3), XQ(3), YQ(3)
CALL REORPU(XT, YT, ZT, XP, YP, T)
CALL REORPU(XR, YR, ZR, XQ, YQ, U)
DO 10 I=1,3
10 CALL LIN(XP(I), YP(I), XQ(I), YQ(I))
RETURN
END

```

2.1.3. Subprogramul CLASDR. Exemple.

Efectuează clasificarea dreptelor în raport cu fețele triedrului de rețea. Dreptele pot fi definite prin două puncte sau prin parametrii directori.

```

SUBROUTINE CLASDR(N,X1,Y1,Z1,X2,Y2,Z2,AL,AM,AN)
DIMENSION X1(200),Y1(200),Z1(200),X2(200),Y2(200),Z2(200)
READ(105,1) N
1 FORMAT(I4)
READ(105,2) (X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),I=1,N)
2 FORMAT(6F10.3,20X)
DO 9 I=1,N
WRITE(108,4) X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I)
4 FORMAT(' ', 'COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA' /
* ' ', 'X1=', F10.3, 2X, 'Y1=', F10.3, 2X, 'Z1=', F10.3/ ' ', 'X2=', F10.3, 2X,
* 'Y2=', F10.3, 2X, 'Z2=', F10.3/)
AL=X2(I)-X1(I)
AM=Y2(I)-Y1(I)
AN=Z2(I)-Z1(I)
IF(AL.EQ.0) GO TO 5
IF(AM.EQ.0) GO TO 10
IF(AN.EQ.0) GO TO 12
WRITE(108,30) AL,AM,AN
21 FORMAT(' ', 2X, 'DREAPTA ESTE OARECARE'////)
GO TO 9
12 WRITE(108,30) AL,AM,AN
WRITE(108,22)
22 FORMAT(' ', 2X, 'DREAPTA ESTE ORIZONTAL'////)
GO TO 9
10 IF(AN.EQ.0) GO TO 11
WRITE(108,30) AL,AM,AN
WRITE(108,23)
23 FORMAT(' ', 2X, 'DREAPTA ESTE FRONTALA'////)
GO TO 9
11 WRITE(108,30) AL,AM,AN
WRITE(108,24)
24 FORMAT(' ', 2X, 'DREAPTA ESTE FRONTODRIZONTALA'////)
GO TO 9
5 IF(AM.EQ.0) GO TO 6
IF(AN.EQ.0) GO TO 8
WRITE(108,30) AL,AM,AN
WRITE(108,25)
25 FORMAT(' ', 2X, 'DREAPTA ESTE DE PROFIL'////)
GO TO 9
8 WRITE(108,30) AL,AM,AN
WRITE(108,26)
26 FORMAT(' ', 2X, 'DREAPTA ESTE DE CAPAT'////)
GO TO 9
6 IF(AN.EQ.0) GO TO 7
WRITE(108,30) AL,AM,AN
WRITE(108,27)
27 FORMAT(' ', 2X, 'DREAPTA ESTE VERTICALA'////)
GO TO 9
7 WRITE(108,30) AL,AM,AN
WRITE(108,28)
28 FORMAT(' ', 2X, 'ERDARE'////)
30 FORMAT(' ', 2X, 'AL=', F10.3/ ' ', 2X, 'AM=', F10.3/ ' ', 2X, 'AN=', F10.3/)
9 CONTINUE
STOP
END

```

Acest program poate fi extins în sensul de a cuprinde și alte clasificări posibile ale dreptelor cum ar fi de exemplu:

- dreptele paralele cu planele bisectoare;
- dreptele ce intersectează linia de pământ;
- dreptele perpendiculare pe planele bisectoare ca o categorie specială de drepte de profil.

```

COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA
X1= 5.000 Y1= 2.000 Z1= 3.000
X2= 2.000 Y2= 6.000 Z2= 3.000
AL= -3.000
AM= 4.000
AN= 0.000
DREAPTA ESTE ORIZONTALA

```

```

COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA
X1= 3.000 Y1= 4.000 Z1= 2.000
X2= 6.000 Y2= 4.000 Z2= 5.000
AL= -3.000
AM= 0.000
AN= 3.000
DREAPTA ESTE FRONTALA

```

```

COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA
X1= 3.000 Y1= 6.000 Z1= 1.000
X2= 3.000 Y2= 2.000 Z2= 5.000
AL= 0.000
AM= -4.000
AN= 4.000
DREAPTA ESTE DE PROFIL

```

```

COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA
X1= 4.000 Y1= 2.000 Z1= 4.000
X2= 6.000 Y2= 4.000 Z2= 3.000
AL= -4.000
AM= 0.000
AN= 0.000
DREAPTA ESTE FRONTORIZONTALA

```

```

COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA
X1= 4.000 Y1= 3.000 Z1= 3.000
X2= 4.000 Y2= 3.000 Z2= 3.000
AL= 0.000
AM= 0.000
AN= 0.000
DREAPTA ESTE DARECARE

```

```

COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA
X1= 3.000 Y1= 3.000 Z1= 3.000
X2= 3.000 Y2= 3.000 Z2= 3.000
AL= 0.000
AM= -3.000
AN= 0.000
DREAPTA ESTE DE CAPAT

```

```

COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA
X1= 3.000 Y1= 3.000 Z1= 5.000
X2= 3.000 Y2= 2.000 Z2= 2.000
AL= 0.000
AM= 0.000
AN= 3.000
DREAPTA ESTE VERTICALA

```

```

COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA
X1= 4.000 Y1= 3.000 Z1= 3.000
X2= 4.000 Y2= 3.000 Z2= 3.000
AL= 0.000
AM= 0.000
AN= 3.000
DREAPTA ESTE VERTICALA

```

```

COORDONATELE CELOR DOUA PUNCTE CARE DEFINESC DREAPTA
X1= 9.000 Y1= 9.000 Z1= 2.000
X2= 9.000 Y2= 2.000 Z2= 2.000
AL= 0.000
AM= 0.000
AN= 0.000
EROARE

```

2.1.4. Subprogramul PCTDR

Determină coordonatele unui punct arbitrar de pe o dreaptă definită printr-un punct dat și direcție dată.

```

C   SUBROUTINE PCTDR(N,AL,AM,AN,X1,Y1,Z1,X,Y,Z,XP1,YP1,XP2,YP2,XP3,YP3
C   )
C   * DETERMINAREA UNUI PUNCT ARBITRAR X,Y,Z PE O DREAPTA DEFINITA PRIN
C   DIRECTIA AL,AM,AN, SI PUNCTUL X1,Y1,Z1
      DIMENSION AL(100),AM(100),AN(100),X1(100),Y1(100),Z1(100)
      READ (105,1) N
      1  FORMAT (I4)
      READ (105,2) (AL(I),AM(I),AN(I),X1(I),Y1(I),Z1(I),I=1,N)
      2  FORMAT (6F10.3,20X)
      DO 10 I=1,N
        WRITE(108,11) I, 'DATELE DE INTRARE'//
      11  FORMAT(///)
        WRITE(108,4)AL(I),AM(I),AN(I),X1(I),Y1(I),Z1(I)
        4  FORMAT(' ',AL=' ',F10.3,2X,'AM=',F10.3,2X,'AN=',F10.3,
        2X,'X1=',F10.3,2X,'Y1=',F10.3,2X,'Z1=',F10.3/)
        D=SQRT(AL(I)**2+AM(I)**2+AN(I)**2)
        X=X1(I)+AL(I)/D**2
        Y=Y1(I)+AM(I)/D**2
        Z=Z1(I)+AN(I)/D**2
        WRITE(106,12)
      12  FORMAT(' ', 'REZULTATE'//)
        3  WRITE(108,3) X,Y,Z
        3  FORMAT (' ', 'X=',F10.3,4X,'Y=',F10.3,4X,'Z=',F10.3/)
        CALL FLT(X)
        CALL FLT(Y)
        CALL FLT(Z)
        XP1=-X
        YP1=-Y
        XP2=-X
        YP2=Z
        XP3=Y
        YP3=Z
        WRITE(108,5) XP1,YP1,XP2,YP2,XP3,YP3
        5  *FORMAT(' ', 'XP1=',F10.3,2X,'YP1=',F10.3,2X,'XP2=',F10.3,2X,'YP2=',
        *F10.3,2X,'XP3=',F10.3,2X,'YP3=',F10.3/)
      10 CONTINUE
      STOP
      END

      SUBROUTINE FLT(X)
      EQUIVALENCE(Y,IY)
      Y=X
      IF(ABS(IY).GT.10000) RETURN
      X=IY
      RETURN
      END

```

Subprogramul **FLT(X)** flotează parametri întregi.

2.1.5. Subprogramul DPCTDR

Determină coordonatele unui punct arbitrar de pe o dreaptă definită prin două puncte date și un parametru.

```

C   SUBROUTINE DPCTDR(N,X1,Y1,Z1,X2,Y2,Z2,AK,X,Y,Z,XP1,YP1,XP2,YP2,XP3
C   *,YP3)
C   DETERMINAREA UNUI PUNCT ARBITRAR PE O DREAPTA DEFINITA DE DOUA
C   PUNCTE SI PARAMETRUL AK
C   DIMENSION X1(100),Y1(100),Z1(100),X2(100),Y2(100),Z2(100),AK(100)
  READ(105,1) N
  1  FORMAT(I4)
  READ(105,2) (X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),AK(I),I=1,N)
  2  FORMAT(7F10.3,10X)
  DO 10 I=1,N
    WRITE(103,11)
  11  FORMAT(//',', 'DATELE DE INTRARE'/)
    WRITE(103,5) X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),AK(I)
  5  *  FORMAT(' ', 'X1=', F10.3,2X, 'Y1=', F10.3,2X, 'Z1=', F10.3,2X, 'X2=',
    *  F10.3,2X, 'Y2=', F10.3,2X, 'Z2=', F10.3,2X, 'AK=', F10.3/)
    CALL FLT(AK)
    AL=X2(I)-X1(I)
    AM=Y2(I)-Y1(I)
    AN=Z2(I)-Z1(I)
    X=AK(I)*AL+X1(I)
    Y=AK(I)*AM+Y1(I)
    Z=AK(I)*AN+Z1(I)
    WRITE(103,12)
  12  FORMAT(' ', 'REZULTATE'/)
    WRITE(103,3) X,Y,Z
  3  *  FORMAT(' ', 'X=', F10.3,4X, 'Y=', F10.3,4X, 'Z=', F10.3/)
    CALL FLT(X)
    CALL FLT(Y)
    CALL FLT(Z)
    XP1=-X
    YP1=-Y
    XP2=-X
    YP2=Z
    XP3=Y
    YP3=Z
    WRITE(103,4) XP1,YP1,XP2,YP2,XP3,YP3
  4  *  FORMAT(' ', 'XP1=', F10.3,2X, 'YP1=', F10.3,2X, 'XP2=', F10.3,2X, 'YP2=',
    *  F10.3,2X, 'XP3=', F10.3,2X, 'YP3=', F10.3/)
  10 CONTINUE
  STP
  END

```

2.1.6. Subprogramul PL3P

Determină planul care trece prin trei puncte date. Eventual desenează ceea ce îi este indicat. Epura este reprezentată în figura 2.2.

Astfel, au fost date coordonatele celor trei puncte care definesc un triunghi, deci de o suprafață plană, iar pe epura obținută au fost reprezentate și urmele planului în triplă proiecție ortogonală.

```

C   SUBROUTINE PL3P(N,X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3,A,B,C,D)
C   DETERMINAREA PLANULUI CARE TRECE PRIN 3 PUNCTE (AX+BY+CZ+D=0)
C   DACA IND=0 SE DESENEAZA NUMAT URMELE PLANULUI
C   DACA IND=1 SE DESENEAZA URMELE PLANULUI SI PROIECTIILE TRIUNGHILUI
C   DACA IND=2 SE DESENEAZA NUMAT PROIECTIILE TRIUNGHILUI DEFINIT DE
C   CELE TREI PUNCTE
C
  DIMENSION X1(100),Y1(100),Z1(100),X2(100),Y2(100),Z2(100)
  DIMENSION X3(100),Y3(100),Z3(100)
  DIMENSION XP(3),YP(3),XQ(2),YQ(3)
  LOGICAL*1 T(2),U(2)
  CALL ASSIGN(3,'PP:1)

```

```

CALL ASSIGN(1,'CR:')
CALL ASSIGN(2,'LP:')
CALL INI(3)
READ(1,7) IND
7 FORMAT(I2)
READ(1,1) N
1 FORMAT(I4)
READ(1,2) (X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),X3(I),Y3(I),Z3(I),I
* =1,N)
2 FORMAT(9F6.2,8X)
DO 10 I=1,N
C TRASEAZA AXELE EPUREI
CALL IIN(-130.,0.,130.,0.)
CALL IIN(0.,-95.,0.,95.)
CALL TEX(10.,-20.,0.,4.,0.,'SURROUTINE PL3P',15)
CALL TEX(10.,-30.,0.,3.5,0.,'PIAN DEFINIT PRIN 3 PUNCTE',26)
CALL TEX(-130.,1.,0.,2.5,0.,'X',1)
CALL TEX(130.,1.,0.,2.5,0.,'Y',2)
CALL TEX(1.,-95.,0.,2.5,0.,'Z',3)
CALL TEX(1.,90.,0.,2.5,0.,'7',1)
CALL TEX(2.,2,0.,2.5,0.,'0',1)
WRITE(2,5)
5 FORMAT(' ', 'DATELE DE INTRARE SINT')
WRITE(2,3) X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),X3(I),Y3(I),Z3(I)
3 FORMAT(' ', 'X1=',F10.3,2X,'Y1=',F10.3,2X,'Z1=',F10.3/
*1 ', 'X2=',F10.3,2X,'Y2=',F10.3,2X,'Z2=',F10.3/
*1 ', 'X3=',F10.3,2X,'Y3=',F10.3,2X,'Z3=',F10.3/)
IF(IND.EQ.2) GO TO 11
C CALCULEAZA COEFICIENTII ECUATIEI PLANULUI
A=Y1(I)*Z2(I)-Y1(I)*Z3(I)+Y2(I)*Z3(I)-Y2(I)*Z1(I)+Y3(I)*Z1(I)-
*Y3(I)*Z2(I)
H=X1(I)*Z3(I)-X1(I)*Z2(I)+X2(I)*Z1(I)-X2(I)*Z3(I)-X3(I)*Z1(I)+
*X3(I)*Z2(I)
C=X1(I)*Y2(I)-X1(I)*Y3(I)+X2(I)*Y3(I)-X2(I)*Y1(I)+X3(I)*Y1(I)-
*X3(I)*Y2(I)
D=X1(I)*Y2(I)*Z3(I)+X2(I)*Y3(I)*Z1(I)+X3(I)*Y1(I)*Z2(I)-X3(I)*Y2(I
*)*Z1(I)-X2(I)*Y1(I)*Z3(I)-X1(I)*Y3(I)*Z2(I)
D=D-D
C CALCULEAZA SI DESENEAZA TAIFURILE PLANULUI
S1=-D/A
S2=-D/B
S3=-D/C
WRITE(2,6) S1,S2,S3
6 FORMAT(' ', 'S1=',F10.3,2X,'S2=',F10.3,2X,'S3=',F10.3/)
CALL PLOT(0.,-S2,0)
CALL PLOT(-S1,0.,1)
CALL PLOT(0.,S3,1)
CALL PLOT(S2,0.,1)
CALL CIS(S2,0.,.5)
CALL CIS(0.,-S2,.5)
CALL CIS(0.,S3,.5)
CALL CIS(-S1,0.,.5)
CALL TEX(1.,-S2,0.,2.5,0.,S2,2)
CALL TEX(-S1,1.,0.,2.5,0.,S1,2)
CALL TEX(1.,S3,0.,2.5,0.,S3,2)

```

```

CALL TEX(S2,1.,0.,2.5,0.,S2',2)
IF(IND.EG.0) GO TO 10
C DESENEAZA PROIECTIILE TRIUNGHIULUI ABC
11 T(1)=65
   H(1)=66
   CALL REORDR(X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),XP,YP,XQ,YQ,T,U)
   T(1)=66
   U(1)=67
   CALL REORDR(X2(I),Y2(I),Z2(I),X3(I),Y3(I),Z3(I),XP,YP,XQ,YQ,T,U)
   T(1)=67
   U(1)=65
   CALL REORDR(X3(I),Y3(I),Z3(I),X1(I),Y1(I),Z1(I),XP,YP,XQ,YQ,T,U)
10 CONTINUE
CALL FOF
STOP
END

```

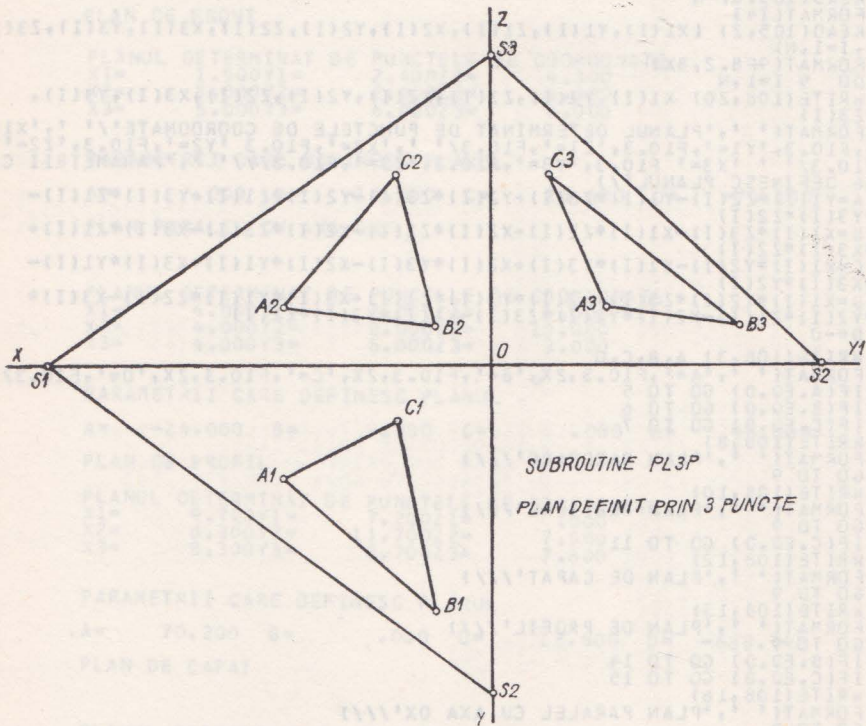


Fig. 2.2

2.1.7. Subprogramul CLASPL. Exemple

Clasifică planele în raport cu fețele triedrului ortogonal de referință. Clasificarea se face în funcție de coeficienții planului (parametrii directori A, B, C și coeficientul D): Planul poate fi definit în diferite feluri.

Discuția poate fi rezumată în următorul tabel:

Felul planului	Coeficienții ecuației planului		
Plan oarecare	$A \neq B \neq C$		
Plan vertical	$C = 0$	$A \neq 0$	$B \neq 0$
Plan de capăt	$B = 0$	$A \neq 0$	$C \neq 0$
Plan paralel cu linia de pământ	$A = 0$	$B \neq 0$	$C \neq 0$
Plan frontal	$A = 0$	$C = 0$	$B \neq 0$
Plan orizontal	$A = 0$	$B = 0$	$C \neq 0$
Plan de profil	$B = 0$	$C = 0$	$A \neq 0$
Plan prin origine	$D = 0$		
Plan perpendicular pe primul plan bisector	$B = C$	$A \neq 0$	
Plan perpendicular pe al 2-lea plan bisector	$A = C$	$B \neq 0$	
Plan perpendicular pe bisectoarea triedrului OXYZ (Plan axonometric)	$A = B = C$		

```

DIMENSION X1(100),Y1(100),Z1(100),X2(100),Y2(100),Z2(100),X3(100),
*Y3(100),Z3(100)
READ(105,1) N
1 FORMAT(I4)
READ(105,2) (X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),X3(I),Y3(I),Z3(I)
*,I=1,N)
2 FORMAT(9F8,2,8X)
DO 9 I=1,N
WRITE(108,20) X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),X3(I),Y3(I),
*Z3(I)
20 FORMAT(' ','PLANUL DETERMINAT DE PUNCTELE DE COORDONATE',' ','X1='
1,F10.3,'Y1=',F10.3,'Z1=',F10.3,'X2=',F10.3,'Y2=',F10.3,'Z2=',F
210.3,'X3=',F10.3,'Y3=',F10.3,'Z3=',F10.3/' ','PARAMETRII CAR
3E DEFINESC PLANUL')
A=Y1(I)*Z2(I)-Y1(I)*Z3(I)+Y2(I)*Z3(I)-Y2(I)*Z1(I)+Y3(I)*Z1(I)-
*Y3(I)*Z2(I)
B=X1(I)*Z3(I)-X1(I)*Z2(I)+X2(I)*Z1(I)-X2(I)*Z3(I)-X3(I)*Z1(I)+
*X3(I)*Z2(I)
C=X1(I)*Y2(I)-X1(I)*Y3(I)+X2(I)*Y3(I)-X2(I)*Y1(I)+X3(I)*Y1(I)-
*X3(I)*Y2(I)
D=X1(I)*Y2(I)*Z3(I)+X2(I)*Y3(I)*Z1(I)+X3(I)*Y1(I)*Z2(I)-X3(I)*
*Y2(I)*Z1(I)-X2(I)*Y1(I)*Z3(I)-X1(I)*Y3(I)*Z2(I)
D=-D
WRITE(108,3) A,B,C,D
3 FORMAT(' ','A=',F10.3,2X,'B=',F10.3,2X,'C=',F10.3,2X,'D=',F10.3/)
IF(A.EQ.0) GO TO 5
IF(B.EQ.0) GO TO 6
IF(C.EQ.0) GO TO 7
WRITE(108,8)
8 FORMAT(' ','PLAN OARECARE'////)
GO TO 9
7 WRITE(108,10)
10 FORMAT(' ','PLAN VERTICAL'////)
GO TO 9
6 IF(C.EQ.0) GO TO 11
WRITE(108,12)
12 FORMAT(' ','PLAN DE CAPAT'////)
GO TO 9
11 WRITE(108,13)
13 FORMAT(' ','PLAN DE PROFIL'////)
GO TO 9
5 IF(B.EQ.0) GO TO 14
IF(C.EQ.0) GO TO 15
WRITE(108,16)
16 FORMAT(' ','PLAN PARALEL CU AXA OX'////)
GO TO 9
15 WRITE(108,17)
17 FORMAT(' ','PLAN DE FRONT'////)
GO TO 9
14 IF(C.EQ.0) GO TO 18
WRITE(108,19)
19 FORMAT(' ','PLAN DE NIVEL'////)
GO TO 9
18 WRITE(108,21)
21 FORMAT(' ','ERDARE'////)
9 CONTINUE
STOP
END

```


Exemplificări ale subprogramului CLASPL

PLANUL DETERMINAT DE PUNCTELE DE COORDONATE
 X1= 7.000Y1= 3.000Z1= 5.000
 X2= -4.000Y2= 2.000Z2= 5.000
 X3= 2.000Y3= 8.000Z3= 5.000

PARAMETRII CARE DEFINESC PLANUL

A= .000 B= .000 C= -60.000 D= 300.000

PLAN DE NIVEL

PLANUL DETERMINAT DE PUNCTELE DE COORDONATE
 X1= 8.500Y1= 4.000Z1= 4.000
 X2= 4.000Y2= 4.000Z2= 10.000
 X3= 2.000Y3= 4.000Z3= 2.000

PARAMETRII CARE DEFINESC PLANUL

A= .000 B= -48.000 C= .000 D= 192.000

PLAN DE FRONT

PLANUL DETERMINAT DE PUNCTELE DE COORDONATE
 X1= 1.500Y1= 2.400Z1= 4.300
 X2= 8.000Y2= 2.400Z2= 4.300
 X3= 5.000Y3= 6.000Z3= 6.000

PARAMETRII CARE DEFINESC PLANUL

A= .000 B= -11.050 C= 23.400 D= -74.100

PLAN PARALEL CU AXA OX

PLANUL DETERMINAT DE PUNCTELE DE COORDONATE
 X1= 4.000Y1= 2.000Z1= 4.000
 X2= 4.000Y2= 2.000Z2= 10.000
 X3= 4.000Y3= 6.000Z3= 3.000

PARAMETRII CARE DEFINESC PLANUL

A= -24.000 B= .000 C= .000 D= 96.000

PLAN DE PROFIL

PLANUL DETERMINAT DE PUNCTELE DE COORDONATE
 X1= 9.700Y1= 7.200Z1= .000
 X2= 8.300Y2= 11.700Z2= 7.800
 X3= 8.300Y3= 2.700Z3= 7.800

PARAMETRII CARE DEFINESC PLANUL

A= 70.200 B= .000 C= 12.500 D= -680.940

PLAN DE CAPAT

PLANUL DETERMINAT DE PUNCTELE DE COORDONATE
 X1= 6.000Y1= 2.000Z1= 2.000
 X2= 6.000Y2= 2.000Z2= 10.000
 X3= 3.000Y3= 4.000Z3= 3.000

PARAMETRII CARE DEFINESC PLANUL

A= -16.000 B= -24.000 C= .000 D= 144.000

PLAN VERTICAL

2.1.8. Subprogramul PPADD

Determină planul care trece printr-un punct dat și este paralel cu două direcții date. Epura este reprezentată în figura 2.3 fiind executată integral la plotter.

```

C SUBROUTINE PPADD(N,X,Y,Z,AL1,AM1,AN1,AL2,AM2,AN2,A,B,C,D,S1,S2,S3)
C DETERMINAREA PLANULUI DUS PRINTR-UN PUNCT X,Y,Z SI PARALEL CU DOUA
C DIRECTII DATE AL1,AM1,AN1,AL2,AM2,AN2
C DIMENSION X(100),Y(100),Z(100),AL1(100),AM1(100),AN1(100),AL2(100)
C DIMENSION AM2(100),AN2(100),XP(3),YP(3),XQ(3),YQ(3)
C LOGICAL*1 T(2),U(2)
C CALL ASSIGN(3,'PP:')
C CALL ASSIGN(1,'CR:')
C CALL ASSIGN(2,'LP:')
C CALL INT(3)
C READ(1,1) N
1 FORMAT(I4)
C READ(1,2) (X(I),Y(I),Z(I),AL1(I),AM1(I),AN1(I),AL2(I),AM2(I),AN2(I)
C *) ,I=1,N)
2 FORMAT(3F10.3,50X/6F10.3,20X)
C TRASEAZA AXELE EPUREI
C CALL LINC(-130,0,130,0)
C CALL LINC(0,-95,0,95)
C CALL TEX(1,-90,0,2.5,0,'Z',1)
C CALL TEX(1,-95,0,2.5,0,'Y',1)
C CALL TEX(130,1,0,2.5,0,'Y1',2)
C CALL TEX(-130,1,0,2.5,0,'X',1)
C CALL TEX(2,2,0,2.5,0,'O',1)
C DO X I=1,N
C WRITE(2,4) X(I),Y(I),Z(I),AL1(I),AM1(I),AN1(I),AL2(I),AM2(I),AN2(I)
C *)
4 FORMAT(' ','DATELE DE INTRARE SINT',' ',4X,'X=' ,F10.3/' ',4X,'Y=' ,
C *F10.3/' ',4X,'Z=' ,F10.3/' ',4X,'AL1=' ,F10.3,2X,'AM1=' ,F10.3,2X,'AN
C *1=' ,F10.3/' ',4X,'AL2=' ,F10.3,2X,'AM2=' ,F10.3,2X,'AN2=' ,F10.3//)
C A=AM1(I)*AN2(I)-AN1(I)*AM2(I)
C B=AM1(I)*AL2(I)-AL1(I)*AM2(I)
C C=AL1(I)*AM2(I)-AM1(I)*AL2(I)
C D=A*X(I)-B*Y(I)-C*Z(I)
C WRITE(2,5) A,B,C,D
5 FORMAT(' ','PLANUL REZULTAT ESTE DETERMINAT DE PARAMETRII',' ',2X,
C *A=' ,F10.3,2X,'B=' ,F10.3,2X,'C=' ,F10.3,2X,'D=' ,F10.3//)
C S1=-D/A
C S2=-D/B
C S3=-D/C
C WRITE(2,6) S1,S2,S3
6 FORMAT(' ',2X,'S1=' ,F10.3,2X,'S2=' ,F10.3,2X,'S3=' ,F10.3//)
C CALL TEX(10,-25,0,4,0,'SUBROUTINE PPADD',16)
C CALL TEX(10,-35,0,4,0,'PLAN DUS PRINTR-UN PUNCT',24)
C CALL TEX(10,-45,0,4,0,'PARALEL CU DOUA DIRECTII',24)
C CALL TEX(10,-55,0,4,0,'DATE',4)
C CALL CISC(S2,0,5)
C CALL CISC(0,-S2,5)
C CALL CISC(0,S3,5)
C CALL CISC(-S1,0,5)
C CALL PLOT(0,-S2,0)
C CALL PLOT(-S1,0,1)
C CALL PLOT(0,S3,1)
C CALL PLOT(S2,0,1)
C CALL TEX(1,-S2,0,2.5,0,'S2',2)
C CALL TEX(-S1,1,0,2.5,0,'S1',2)

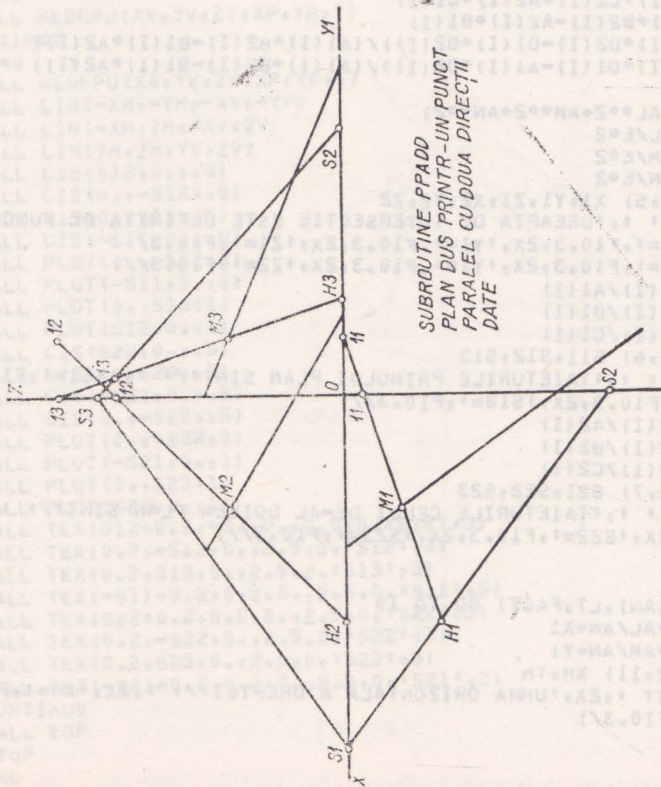
```

În acest program cele două direcții date au fost definite inițial prin perechile de parametri directori date ca atare. Dar cele două direcții pot fi date și, de exemplu, prin intersecțiile a câte două plane, caz în care este necesară deducerea parametrilor directori ai celor două direcții. Acest calcul este arătat în alte programe ale lucrării.

```

CALL TLX(1,53,0,-2.5,J,53,2)
CALL TX(S2,1,0,-2.5,J,52,2)
T(I)=77
CALL REORPU(X,Y,Z,XP,YP,T)
XHEQ
IF(CABS(AM1(I)),LT,1E-8) GO TO 8
XW=-Z(I)*AM1(I)/AM1(I)+X(I)
YW=-Z(I)*AM1(I)/AM1(I)+Y(I)
ZHEQ
IF(CABS(AM1(I)),LT,1E-8) GO TO 10
XW=-Y(I)*AL1(I)/AM1(I)+X(I)
YW=-Y(I)*AM1(I)/AM1(I)+Z(I)
ZHEQ
IF(CABS(AL1(I)),LT,1E-8) GO TO 12
ZW=-X(I)*AM1(I)/AL1(I)+Z(I)
YI(I)=Z
ZHEQ
CALL REORPU(XH,YH,ZH,XP,YP,T)
T(I)=75
YHEQ
CALL REORPU(XV,YV,ZV,XP,YP,T)
T(I)=74
XHEQ
CALL REORPU(XW,YW,ZW,XP,YP,T)
CALL LIN(-XH,-YH,-XV,-YV)
CALL LIN(-XW,-YH,-XV,-ZV)
CALL LIN(YH,ZH,YV,ZV)
XHEQ
IF(CABS(AM2(I)),LT,1E-8) GO TO 80
XW=-Z(I)*AL2(I)/AM2(I)+X(I)
YW=-Z(I)*AM2(I)/AM2(I)+Y(I)
ZHEQ
IF(CABS(AM2(I)),LT,1E-8) GO TO 100
XW=-Y(I)*AL2(I)/AM2(I)+X(I)
YW=-Y(I)*AM2(I)/AM2(I)+Z(I)
ZHEQ
IF(CABS(AL2(I)),LT,1E-8) GO TO 120
CONTINUE
CALL LIN(-XW,-YH,-XV,-YV)
CALL LIN(-XW,-YH,-XV,-ZV)
CONTINUE
CALL TISE
STOP
END

```



SUBROUTINE PRADD
 PLAN DUS PRINTR-UN PUNCT
 PARALEL CU DOUA DIRECTII
 DATE

Fig. 2.3.

2.1.9. Subprogramul INPLPL

Determină dreapta de intersecție dintre două plane.

```

C   SUBROUTINE INPLPL(N,A1,B1,C1,D1,A2,B2,C2,D2,S11,S12,S13,S21,S22,S2
C   *3,X1,Y1,Z1,AL,AM,AN,XH,YH,XV,ZV,YW,ZW)
C   DETERMINAREA DREPTEI DE INTERSECȚIE A DOUA PLANE EXPRESĂ PRIN
C   PUNCTUL X0,Y0,Z0 ȘI DIRECȚIA AL,AM,AN
      DIMENSION A1(100),B1(100),C1(100),D1(100),A2(100),B2(100),C2(100)
      DIMENSION D2(100),XP(3),YP(3),XQ(3),YQ(3)
      LOGICAL*1 T(2),U(2)
      CALL ASSIGN(3,'PP:')
      CALL ASSIGN(1,'CR:')
      CALL ASSIGN(2,'LP:')
      CALL INI(3)
      READ(1,1) N
1   FORMAT(I4)
      READ(1,2) (A1(I),B1(I),C1(I),D1(I),A2(I),B2(I),C2(I),D2(I),I=1,N)
2   FORMAT(8F10.3)
      FACT=1./10**8
      DO 3 I=1,N
      WRITE(2,4) A1(I),B1(I),C1(I),D1(I),A2(I),B2(I),C2(I),D2(I)
4   FORMAT(' ','DATELE DE INTRARE'/ ' ',2X,'PARAMETRII CE DEFINESC CELE
      *DOUA PLANE SINT'/ ' ',A1=' ,F10.3,2X,'B1=' ,F10.3,2X,'C1=' ,F10.3,2X,
      *D1=' ,F10.3/' ' ',A2=' ,F10.3,2X,'B2=' ,F10.3,2X,'C2=' ,F10.3,2X,'D2='
      *F10.3/)
      AL=B1(I)*C2(I)-B2(I)*C1(I)
      AM=-A1(I)*C2(I)+A2(I)*C1(I)
      AN=A1(I)*B2(I)-A2(I)*B1(I)
      X1=(B1(I)*D2(I)-D1(I)*B2(I))/(A1(I)*B2(I)-B1(I)*A2(I))
      Y1=(A2(I)*D1(I)-A1(I)*D2(I))/(A1(I)*B2(I)-B1(I)*A2(I))
      Z1=0
      E=SQRT(AL**2+AM**2+AN**2)
      X2=X1+AL/E*2
      Y2=Y1+AM/E*2
      Z2=Z1+AN/E*2
      WRITE(2,5) X1,Y1,Z1,X2,Y2,Z2
5   FORMAT(' ','DREAPTA DE INTERSECȚIE ESTE DEFINITĂ DE PUNCTELE'/
      *' ',X1=' ,F10.3,2X,'Y1=' ,F10.3,2X,'Z1=' ,F10.3/
      *' ',X2=' ,F10.3,2X,'Y2=' ,F10.3,2X,'Z2=' ,F10.3//)
      S11=-D1(I)/A1(I)
      S12=-D1(I)/B1(I)
      S13=-D1(I)/C1(I)
      WRITE(2,6) S11,S12,S13
6   FORMAT(' ','TAIETURILE PRIMULUI PLAN SINT'/ ' ',S11=' ,F10.3,2X,
      *S12=' ,F10.3,2X,'S13=' ,F10.3//)
      S21=-D2(I)/A2(I)
      S22=-D2(I)/B2(I)
      S23=-D2(I)/C2(I)
      WRITE(2,7) S21,S22,S23
7   FORMAT(' ','TAIETURILE CELUI DE-AL DOILEA PLAN SINT'/ ' ',S21=' ,
      *F10.3,2X,'S22=' ,F10.3,2X,'S23=' ,F10.3//)
      XH=0
      YH=0
      IF (ABS(AN).LT.FACT) GO TO 10
      XH=-Z1*AL/AN+X1
      YH=-Z1*AM/AN+Y1
10  WRITE(2,11) XH,YH
11  FORMAT(' ','DREPTA ORIZONTALĂ A DREPTEI'/ ' ',2X,'XH=' ,F10.3,2X,
      *YH=' ,F10.3/)
      XV=0
      ZV=0

```

```

IF (ABS(AM).LT.FACT) GO TO 12
XV=-Y1*AL/AM+X1
ZV=-Y1*AN/AM+Z1
12 WRITE(2,13) XV,ZV
13 FORMAT(' ',2X,'URMA VERTICALA A DREPTII'/' ',2X,'XV=',F10.3,'Z',
*'XV=',F10.3/)
YW=0
ZW=0

IF (ABS(AL).LT.FACT) GO TO 14
Yw=-X1*AM/AL+Y1
Zw=-X1*AN/AL+Z1
14 WRITE(2,15) Yw,Zw
15 FORMAT(' ',2X,'URMA LATERALA A DREPTII'/' ',2X,'Yw=',F10.3,'Z',
*'Zw=',F10.3//)
C TRASEAZA AXELE EPUREI
CALL LIN(0.,-95.,0.,95.)
CALL LIN(-130.,0.,130.,0.)
CALL TEX(1.,90.,0.,2.5,0,'Z',1)
CALL TEX(1.,-95.,0.,2.5,0,'Y',1)
CALL TEX(130.,1.,0.,2.5,0,'Y1',2)
CALL TEX(-130.,1.,0.,2.5,0,'X',1)
CALL TEX(10.,-20.,0.,4.,0,'SUBROUTINE INPLPL',17)
CALL TEX(10.,-30.,0.,4.,0,'INTERSECTIA A DOUA PLANE',24)
T(1)=72
ZH=0
CALL REORPU(XH,YH,ZH,XP,YP,T)
T(1)=86
YV=0
CALL REORPU(XV,YV,ZV,XP,YP,T)
T(1)=87
XW=0
CALL REORPU(XW,YW,ZW,XP,YP,T)
CALL LIN(-XW,-YH,-XV,-YV)
CALL LIN(-XW,ZH,-XV,ZV)
CALL LIN(YH,ZH,YV,ZV)
CALL CIS(S12,0.,.5)
CALL CIS(0.,-S12,.5)
CALL CIS(0.,S13,.5)
CALL CIS(-S11,0.,.5)
CALL PLOT(0.,-S12,0)
CALL PLOT(-S11,0.,1)
CALL PLOT(0.,S13,1)
CALL PLOT(S12,0.,1)
CALL CIS(S22,0.,.5)
CALL CIS(0.,S23,.5)
CALL CIS(-S21,0.,.5)
CALL CIS(0.,-S22,.5)
CALL PLOT(0.,-S22,0)
CALL PLOT(-S21,0.,1)
CALL PLOT(0.,S23,1)
CALL PLOT(S22,0.,1)
CALL TEX(S12+0.2,-5.,0.,2.5,0,'S12',3)
CALL TEX(0.2,-S12,0.,2.5,0,'S12',3)
CALL TEX(0.2,S13,0.,2.5,0,'S13',3)
CALL TEX(-S11+0.2,0.2,0.,2.5,0,'S11',3)
CALL TEX(S22+0.2,0.2,0.,2.5,0,'S22',3)
CALL TEX(0.2,-S22,0.,2.5,0,'S22',3)
CALL TEX(0.2,S23,0.,2.5,0,'S23',3)
CALL TEX(-S21+0.2,0.2,0.,2.5,0,'S21',3)
3 CONTINUE
CALL EOF
STOP
END

```

```

SUBROUTINE REORUR(AT,YT,ZT,AX,AY,ZR,AP,YP,XU,YU,T,U)
LOGICAL*1 T(1),U(1)
DIMENSION AP(3),YP(3),XU(3),YU(3)
CALL REORPU(AT,YT,ZT,AX,AY,T)
CALL REORPU(XR,YR,ZR,X,Y,U)
DO 10 I=1,3
CALL LIN(AP(I),YP(I),XU(I),YU(I))
RETURN
END
SUBROUTINE REORPU(X,Y,Z,XP,YP,T)
LOGICAL*1 T(1)
DIMENSION XP(3),YP(3)
AP(1)=-X
YP(1)=-Y
AP(2)=-X
YP(2)=Z
AP(3)=Y
YP(3)=Z
DO 1 I=1,3
CALL CIS(AP(I),YP(I),.5)
T(2)=40+1
CALL TEX(AP(I)+1.,YP(I)+1.,U,.2,0,U,T,2)
CONTINUE
RETURN
END

```

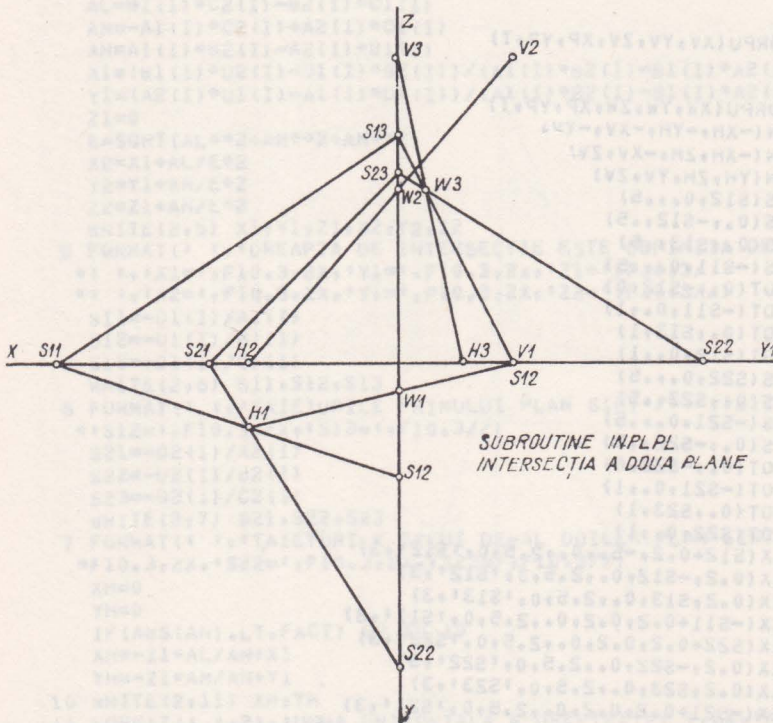


Fig. 2.4

Epura executată integral automat la plotter este dată în figura 2.4.

2.1.10. Subprogramul INDRPL

Determină intersecția dintre o dreaptă și un plan.

```

C   SUBROUTINE INDRPL(N,X1,Y1,Z1,X2,Y2,Z2,X,Y,Z)
C   DETERMINA PUNCTUL DE INTERSECȚIE AL UNEI DREAPTE
C   DATA PRIN DOUA PUNCTE X1,Y1,Z1 SI X2,Y2,Z2 CIL UN PLAN DEFINIT
C   PRIN COEFICIENTII A,R,C,D
C   IND=0 SE DESFEEAZA NUMAI DREAPTA SI PROIECȚIILE PUNCTULUI DE INTERSECȚIE
C   IND=1 SE DESFEEAZA EPIURA COMPLETA, DECI URMELE PLANULUI, DREAPTA SI
C   PUNCTUL DE INTERSECȚIE
      DIMENSION XP(3),YP(3),XQ(3),YQ(3)
      DIMENSION A(100),R(100),C(100),D(100),X1(100),Y1(100),Z1(100),AL(1
*00),AM(100),AN(100),X2(100),Y2(100),Z2(100)
      LOGICAL*1 T(2),U(2)
      CALL ASSIGN(3,'PP:')
      CALL ASSIGN(1,'CP:')
      CALL ASSIGN(2,'LP:')
      CALL INI(3)
      CALL TEX(10.,-20.,0.,4.,0.,'SUBROUTINE INDRPL',17)
      CALL TEX(10.,-30.,0.,4.,0.,'INTERSECȚIE DREAPTA-PLAN',24)
      READ(1,7) IND
      7 FORMAT(I2)
      READ(1,1) N
      1 FORMAT(I4)
C   TRASEAZA AXELE EPIUREI
      CALL IIN(-130.,0.,130.,0.)
      CALL IIN(0.,-95.,0.,95.)
      CALL TEX(1.,90.,0.,2.5,0.,7.,1)
      CALL TEX(1.,-95.,0.,2.5,0.,7.,1)
      CALL TEX(130.,1.,0.,2.5,0.,7.,2)
      CALL TEX(-130.,1.,0.,2.5,0.,7.,1)
      CALL TEX(.2.,2.,0.,2.5,0.,7.,1)
      READ(1,2) (A(I),R(I),C(I),D(I),X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I)
* ,I=1,N)
      2 FORMAT(4F10.3,4X,4F10.3,10X)
      DO 3 I=1,N
      WRITE(2,4) A(I),R(I),C(I),D(I)
      4 FORMAT(' ','DATELE DE INTRARE SINT:/' ' ','PARAMETRII CARE DEFINESC
* PLANUL/' ' ','A=' ,F10.3,2X,'B=' ,F10.3,2X,'C=' ,F10.3,2X,'D=' ,F10.3/
* /)
      IF (IND.EQ.0) GO TO 10
C   CALCULEAZA, TRASEAZA SI HALUSTREAZA TATFTURILE PLANULUI
      S1=-D(I)/A(I)
      S2=-D(I)/R(I)
      S3=-D(I)/C(I)
      CALL PLOT(0.,-S2,0)
      CALL PLOT(-S1,0.,1)
      CALL PLOT(0.,S3,1)
      CALL PLOT(S2,0.,1)
      CALL CIT(S2,0.,.5)
      CALL CIT(0.,-S2,.5)
      CALL CIT(0.,S3,.5)
      CALL CIT(-S1,0.,.5)
      CALL TEX(S2,1.,0.,2.5,0.,'S2',2)
      CALL TEX(-S1,1.,0.,2.5,2.,'S1',2)
      CALL TEX(1.,S3,0.,2.5,0.,'S3',2)
      CALL TEX(1.,-S2,0.,2.5,0.,'S2',2)
      WRITE(2,6) S1,S2,S3
      6 FORMAT(' ','S1=' ,F10.3,2X,'S2=' ,F10.3,2X,'S3=' ,F10.3/

```

```

C DETERMINA PARAMETRII DIRECTORII AL,AM,AN AT DREPTII
10 AL(I)=X2(I)-X1(I)
   AM(I)=Y2(I)-Y1(I)
   AN(I)=Z2(I)-Z1(I)
   WRITE(2,8) X(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),AL(I),AM(I),AN(I)
8  FORMAT(' ', 'PARAMETRII CARE DEFINESC DREAPTA'//', ', 'X1=',F10.3,2X,
   *'Y1=',F10.3,2X, 'Z1=',F10.3//', ', 'X2=',F10.3,2X, 'Y2=',F10.3,2X, 'Z2=',
   *F10.3//', ', 'AL=',F10.3,2X, 'AM=',F10.3,2X, 'AN=',F10.3//')
   AK=(A(I)*X1(I)+B(I)*Y1(I)+C(I)*Z1(I)+D(I))/(A(I)*AL(I)+B(I)*AM(I)+
   *C(I)*AN(I))
C CALCULEAZA COORDONATILE PUNCTULUI DE INTERSECIE DINTRE DREAPTA SI PLAN
   X=X1(I)-AL(I)*AK
   Y=Y1(I)-AM(I)*AK
   Z=Z1(I)-AN(I)*AK
   WRITE(2,5) X,Y,Z
5  FORMAT(' ', 'COORDONATELE PUNCTULUI DE INTERSECIE SINT'//', ', 'X=',
   *F10.3//', ', 'Y=',F10.3//', ', 'Z=',F10.3//')
C DESENEAZA PROIECTIILE DREPTII SI ALE PUNCTULUI DE INTERSECIE I(X,Y,Z)
C CU PLANUL
   T(1)=65
   U(1)=66
   CALL REORDR(X1,Y1,Z1,X2,Y2,Z2,XP,YP,XQ,YQ,T,U)
   T(1)=73
   CALL REORPI(X,Y,Z,XP,YP,T)
   CALL CIT(-X,-Y,1.)
   CALL CIT(-X,Z,1.)
   CALL CIT(Y,Z,1.)
3 CONTINUE
CALL EOF
STOP
END

DATELE DE INTRARE SINT
PARAMETRII CARE DEFINESC PLANUL
A= 2.000 H= 2.000 C= 4.000 D= -150.000
S1= 75.000 S2= 75.000 S3= 37.500
PARAMETRII CARE DEFINESC DREAPTA
X1= 85.000 Y1= 30.000 Z1= 40.000
X2= -15.000 Y2= -23.000 Z2= -57.000
AL= -100.000 AM= -53.000 AN= -97.000
COORDONATELE PUNCTULUI DE INTERSECIE SINT
X= 50.418
Y= 11.671
Z= 6.455

```

Epura executată integral automat la plotter este dată în figura 2.5.

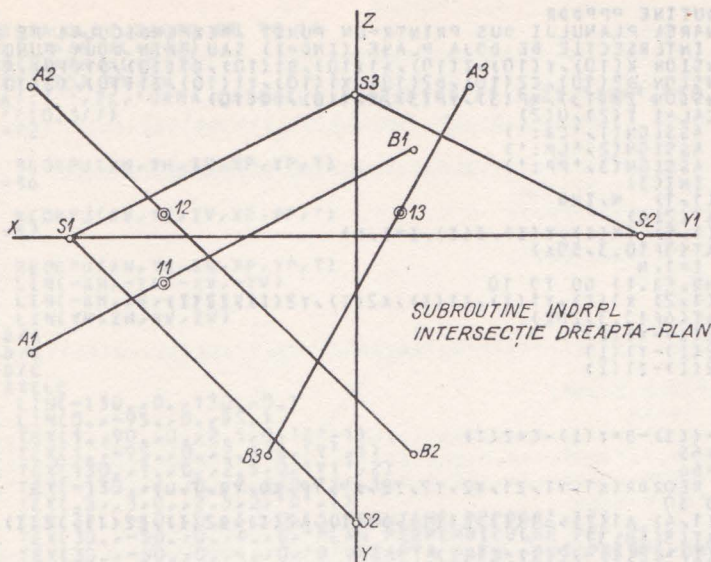


Fig. 2.5

2.1.11. Subprogramul PPPDDR

Determină planul dus printr-un punct dat, perpendicular pe o dreaptă definită ca intersecție de două plane.

Dacă datele de intrare sînt: X, Y, Z —coordonatele punctului și

A_1, B_1, C_1 } parametri directori
 A_2, B_2, C_2 } ai celor două plane

se calculează parametri directori
 ai planului perpendicular

$$AL = A = B_1C_2 - B_2C_1$$

$$AM = B = A_2C_1 - A_1C_2$$

$$AN = C = A_1B_2 - A_2B_1$$

$D = -AX - BY - CZ$ astfel încît ecuația planului căutat este

$$Ax + By + Cz + D = 0$$

```

SUBROUTINE PPPDDR
DETERMINAREA PLANULUI DUS PRINTR-UN PUNCT , PERPENDICULAR PE O DREAPTA
DATA CA INTERSECȚIE DE DOUA PLANE (IND=1) SAU PRIN DOUA PUNCTE 2(IND=2)
DIMENSION X(10),Y(10),Z(10),A1(10),B1(10),C1(10),D1(10),A2(10)
DIMENSION B2(10),C2(10),D2(10),X1(10),Y1(10),Z1(10),X2(10),Y2(10)
DIMENSION Z2(10),XP(3),YP(3),XQ(10),YQ(10)
LOGICAL*1 T(2),U(2)
CALL ASSIGN(1,'CR:')
CALL ASSIGN(2,'LP:')
CALL ASSIGN(3,'PP:')
CALL INI(3)
READ(1,1) N,IND
1  FORMAT(2I4)
READ(1,5) (X(I),Y(I),Z(I),I=1,N)
5  FORMAT(3F10.3,50X)
DO 3 I=1,N
IF(IND.EQ.1) GO TO 10
READ(1,2) X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I)
2  FORMAT(6F10.3,20X)
AL=X2(I)-X1(I)
AM=Y2(I)-Y1(I)
AN=Z2(I)-Z1(I)
A=AL
B=AM
C=AN
D=-A*(I)-B*Y(I)-C*Z(I)
T(1)=.55
U(1)=.66
CALL REORDR(X1,Y1,Z1,X2,Y2,Z2,XP,YP,XQ,YQ,T,U)
GO TO 30
10 READ(1,4) A1(I),B1(I),C1(I),D1(I),A2(I),B2(I),C2(I),D2(I)
4  FORMAT(8F10.3)
A=B1(I)*C2(I)-B2(I)*C1(I)
B=A2(I)*C1(I)-A1(I)*C2(I)
C=A1(I)*B2(I)-A2(I)*B1(I)
D=-A*X(I)-B*Y(I)-C*Z(I)
AL=B1(I)*C2(I)-B2(I)*C1(I)
AM=-A1(I)*C2(I)+A2(I)*C1(I)
AN=A1(I)*B2(I)-A2(I)*B1(I)
FACT=1E-3
XH=0
YH=0
IF(ABS(AN).LT.FACT) GO TO 20
XH=-Z(I)*AL/AN+X(I)
YH=-Z(I)*AM/AN+Y(I)
20 WRITE(2,11) XH,YH
11  FORMAT(' ',2X,'URMA ORIZONTALA A DREPTEI '/' ' ',2X,'XH=' ,F10.3,2X,
* 'YH=' ,F10.3/)
XV=0
ZV=0
IF(ABS(AM).LT.FACT) GO TO 12
XV=-Y(I)*AL/AM+X(I)
ZV=-Y(I)*AN/AM+Z(I)
12  WRITE(2,13) XV,ZV
13  FORMAT(' ',2X,'URMA VERTICALA A DREPTEI '/' ' ',2X,'XV=' ,F10.3,2X,
* 'ZV=' ,F10.3/)

```

Și în acest exemplu, ca și în celelalte subprograme, dreapta poate fi definită altfel de exemplu: prin două puncte, prin parametri directori, etc., iar planul perpendicular căutat poate fi și el exprimat și desenat prin orice alte elemente care-l pot defini și care servesc astfel mai bine, eventual ca date de intrare într-o nouă sau aceeași problemă în curs de rezolvare.

```

YW=0
ZW=0
IF (ABS(AL).LT.FACT) GO TO 14
YW=-X(I)*AM/AL+Y(I)
ZW=-X(I)*AN/AL+Z(I)
14 WRITE(2,15) YW,ZW
15 FORMAT('1,2X,'URMA LATERALA A DREPTEI'/' ' ,2X,'YW=' ,F10.3,2X,
* 'ZW=' ,F10.3//)
T(1)=72
ZH=0
CALL REORPU(XH,YH,ZH,XP,YP,T)
T(1)=86
YV=0
CALL REORPU(XV,YV,ZV,XP,YP,T)
T(1)=87
XW=0
CALL REORPU(XW,YW,ZW,XP,YP,T)
CALL LIN(-XH,-YH,-XV,-YV)
CALL LIN(-XH,ZH,-XV,ZV)
CALL LIN(YH,ZH,YV,ZV)
30 S1=-D/A
S2=-D/B
S3=-D/C
E TRASAM AXELE
CALL LIN(-130,0,130,0)
CALL LIN(0,-95,0,95)
CALL TEX(1,90,0,2.5,0,'Z',1)
CALL TEX(1,-95,0,2.5,0,'Y',1)
CALL TEX(130,1,0,2.5,0,'Y1',2)
CALL TEX(-130,1,0,2.5,0,'X',1)
CALL TEX(3,3,0,2.5,0,'0',1)
CALL TEX(30,-40,0,4,0,'SUBROUTINE PPPDDR',17)
CALL TEX(30,-50,0,4,0,'PLAN PERPENDICULAR PE',21)
CALL TEX(30,-60,0,4,0,'DREAPTA DATA ,DUS PRINTR-UN',29)
CALL TEX(30,-70,0,4,0,'PUNCT OARECARE',14)
CALL CIS(S2,0,5)
CALL CIS(0,-S2,5)
CALL CIS(0,S3,5)
CALL CIS(-S1,0,5)
CALL PLOT(J,S2,0)
CALL PLOT(-S1,S3,1)
CALL PLOT(J,S3,1)
CALL PLOT(S2,0,1)
T(1)=77
CALL REORPU(X,Y,Z,XP,YP,T)
CALL TEX(1,-S2,0,2.5,0,'S2',2)
CALL TEX(-S1,1,0,2.5,0,'S1',2)
CALL TEX(1,S3,0,2.5,0,'S3',2)
CALL TEX(S2,1,0,2.5,0,'S2',2)
3 CONTINUE
CALL EOF
STOP
END

```

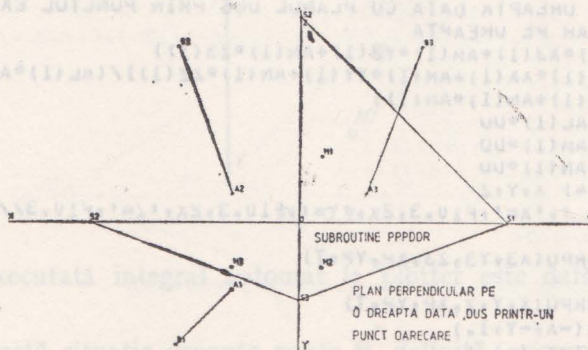


Fig. 2.6

Epura executată integral automat la plotter este dată în figura 2.6.

2.2. Geometrie descriptivă asistată de calculator (II)

2.2.1. Subprogramul DPRPED

Determină proiecția unui punct exterior pe o dreaptă definită prin două puncte.

```

SUBPROGRAM DPRPD
C DETERMINA PROIECTIA UNUI PUNCT EXTERIOR X3,Y3,Z3, PE O DREAPTA DATA
C PRIN DOUA PUNCTE X1,Y1,Z1, SI XX,YY,ZZ
C
  DIMENSION X1(50),Y1(50),Z1(50),XX(50),YY(50),ZZ(50),X3(50),Y3(50)
  DIMENSION Z3(50)
  DIMENSION XP(3),YP(3),XQ(3),YQ(3),AL(50),AM(50),AN(50)
  LOGICAL *1 T(2),U(2)
  CALL ASSIGN(3,'PP:')
  CALL ASSIGN(1,'CR:')
  CALL ASSIGN(2,'LP:')
  CALL INI(3)
  READ(1,1) N
1  FORMAT(14)
  READ(1,2) (X1(I),Y1(I),Z1(I),XX(I),YY(I),ZZ(I),X3(I),Y3(I),Z3(I),I
  *1,N)
2  FORMAT(9F5.2,35X)
  DO 10 I=1,N
  WRITE(2,3) AL(I),AM(I),AN(I),XX(I),YY(I),ZZ(I),X3(I),Y3(I),Z3(I)
3  FORMAT(' ',AL=F10.3,2X,AM=F10.3,2X,AN=F10.3/' ',XX=F10.3,2X,YY=F10.3,2X,ZZ=F10.3/' ',X3=F10.3,2X,Y3=F10.3,2X,Z3=F10.3//)
  CALL TEX(10.,-25.,0.,4.,0,'SUBROUTINE DPRPD',17)
  CALL TEX(10.,-35.,0.,4.,0,'DETERMINAREA PROIECTIEI UNUI',28)
  CALL TEX(10.,-45.,0.,4.,0,'PUNCT EXTERIOR PE O DREAPTA',27)
  CALL TEX(10.,-55.,0.,4.,0,'DATA PRIN DOUA PUNCTE',21)
C TRASEAZA AXELE EPUREI
  CALL LIN(-130.,0.,130.,0.)
  CALL LIN(0.,-95.,0.,95.)
  CALL TEX(1.,90.,0.,2.5,0,'Z',1)
  CALL TEX(1.,-95.,0.,2.5,0,'Y',1)
  CALL TEX(130.,1.,0.,2.5,0,'Y1',2)
  CALL TEX(-130.,1.,0.,2.5,0,'X',1)
  CALL TEX(.3,.3,0.,2.5,0,'U',1)
  AL(I)=X1(I)-XX(I)
  AM(I)=Y1(I)-YY(I)
  AN(I)=Z1(I)-ZZ(I)

C INTERSECTAM DREAPTA DATA CU PLANUL DUS PRIN PUNCTUL EXTERIOR
C PERPENDICULAR PE DREAPTA
  UU=(AL(I)*X3(I)+AM(I)*Y3(I)+AN(I)*Z3(I))
  UU=(U+AL(I)*XX(I)+AM(I)*YY(I)+AN(I)*ZZ(I))/(AL(I)*AL(I)+
  *AM(I)*AM(I)+AN(I)*AN(I))
  X=XX(I)-AL(I)*UU
  Y=YY(I)-AM(I)*UU
  Z=ZZ(I)-AN(I)*UU
  WRITE(2,4) X,Y,Z
4  FORMAT(' ',X=F10.3,2X,Y=F10.3,2X,Z=F10.3//)
  T(1)=77
  CALL KEURPU(X3,Y3,Z3,XP,YP,T)
  T(1)=78
  CALL KEURPU(X,Y,Z,XP,YP,T)
  CALL CIS(-X,-Y,1.)
  CALL CIS(-X,Z,1.)
  CALL CIS(Y,Z,1.)
  T(1)=65
  U(1)=65
  CALL REORDR(X1,Y1,Z1,XX,YY,ZZ,XP,YP,XQ,YQ,T,U)
10 CONTINUE
  CALL EOF
  STOP
  END

```

```

AL=      0.000  AM=      0.000  AN=      0.000
AX=     -30.000  AY=      0.000  AZ=      00.000
AJ=     -30.000  JZ=      80.000  JZ=     -10.000

```

$A = \begin{matrix} X & Y & Z \\ 21.704 & 12.557 & 20.909 \end{matrix}$

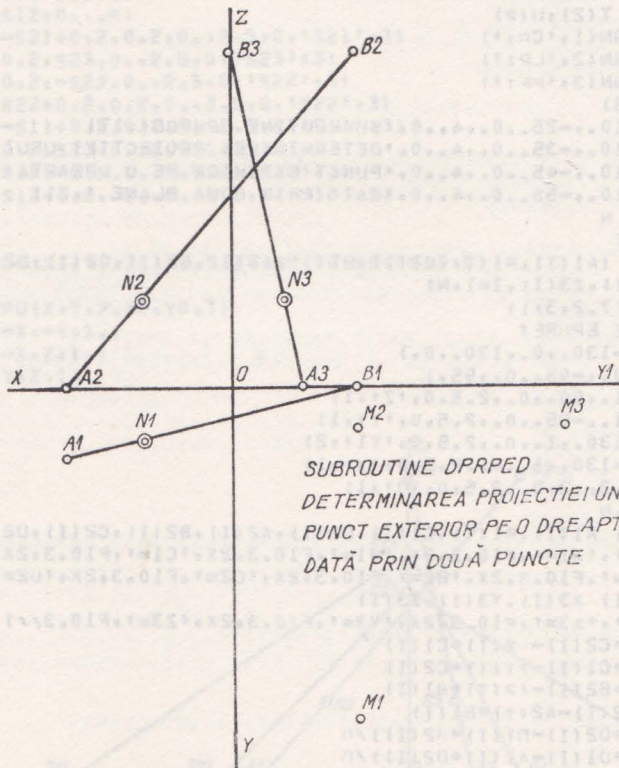


Fig. 2.7

Epura executată integral automat la plotter este dată în figura 2.7.

Și în această situație dreapta poate fi definită oricum, astfel ca, de exemplu, intersecție de două plane, printr-un punct și o direcție dată etc.

2.2.2. Subprogramul DPRPOD

Determină proiecția unui punct exterior pe o dreaptă definită ca intersecție de două plane.

```

C      SUBROUTINE DPRPOD(N,A1,B1,C1,D1,A2,B2,C2,D2,X3,Y3,Z3,X,Y,Z,XP1,YP1
C      *,XP2,YP2,XP3,YP3)
C      DETERMINAREA PROIECTIEI UNUI PUNCT EXTERIOR X3,Y3,Z3 PE O DREAPTA
C      DATA CA INTERSECȚIE DE PLANE
C
      DIMENSION A1(50),B1(50),C1(50),D1(50),A2(50),B2(50),C2(50),D2(50)
      DIMENSION X3(30),Y3(30),Z3(30),XP(3),YP(3),XQ(3),YQ(3)
      LOGICAL*1 T(2),U(2)
      CALL ASSIGN(1,'CD:')
      CALL ASSIGN(2,'LP:')
      CALL ASSIGN(3,'PF:')
      CALL INI(3)
      CALL TEX(10.,-25..0..4..0,'SUBROUTINE DPRPOD',17)
      CALL TEX(10.,-35..0..4..0,'DETERMINAREA PROIECTIEI UNUI',28)
      CALL TEX(10.,-45..0..4..0,'PUNCT EXTERIOR PE O DREAPTA',27)
      CALL TEX(10.,-55..0..4..0,'DATA PRIN DOUA PLANE ',21)
      READ(1,1) N
1     FORMAT(I4)
      READ(1,2) (A1(I),B1(I),C1(I),D1(I),A2(I),B2(I),C2(I),D2(I),
      *,X3(I),Y3(I),Z3(I),I=1,N)
2     FORMAT(8(F7.2,3X))
C TRASEAZA AXELE EPIUREI
      CALL LIN(-130.,0.,130.,0.)
      CALL LIN(0.,-95.,0.,95.)
      CALL TEX(1.,90.,0.,2.5,0,'Z',1)
      CALL TEX(1.,-95.,0.,2.5,0,'Y',1)
      CALL TEX(130.,1.,0.,2.5,0,'Y1',2)
      CALL TEX(-130.,1.,0.,2.5,0,'X',1)
      CALL TEX(.3.,.3,0.,2.5,0,'O',1)
      DO 10 I=1,N
      WRITE(2,4) A1(I),B1(I),C1(I),D1(I),A2(I),B2(I),C2(I),D2(I)
4     FORMAT(' ',A1='F10.3,2X',B1='F10.3,2X',C1='F10.3,2X',D1='F10.
      *,3/' ',A2='F10.3,2X',B2='F10.3,2X',C2='F10.3,2X',D2='F10.3//)
      WRITE(2,11) X3(I),Y3(I),Z3(I)
11    FORMAT(' ',X3='F10.3,2X',Y3='F10.3,2X',Z3='F10.3//)
      A1=B1(I)*C2(I)-C1(I)*C2(I)
      A11=A2(I)*C1(I)-C1(I)*C2(I)
      A1=A1(I)*B2(I)-C1(I)*B1(I)
      U=A1(I)*B2(I)-A2(I)*B1(I)
      X1=(B1(I)*D2(I)-D1(I)*D2(I))/D
      Y1=(A2(I)*D1(I)-A1(I)*D2(I))/D
      Z1=0
      XV=-Y1*A1/AM1+X1
      ZV=-Y1*AN1/AM1+Z1
      D=-((A11*X3(I)+AM1*Y3(I)+AN1*Z3(I)
      DD=(D+A11*X1+AM1*Y1)/(A11*A1+AM1*AN1+AN1*AN1)
      X=X1-AL1*DD
      Y=Y1-AM1*DD
      Z=Z1-AN1*DD
      WRITE(2,12) X,Y,Z
12    FORMAT(' ',X='F10.3,2X',Y='F10.3,2X',Z='F10.3//)
      T(1)=77
      CALL REORPU(X3,Y3,Z3,XP,YP,T)
      S11=-D1(I)/A1(I)
      S12=-D1(I)/B1(I)
      S13=-D1(I)/C1(I)

```

```

S21=-D2(I)/A2(I)
S22=-D2(I)/R2(I)
S23=-D2(I)/C2(I)
CALL PLOT(0.,-S12.0.)
CALL PLOT(-S11.0.,.1)
CALL PLOT(0.,S13.1)
CALL PLOT(S12.0.,.1)
CALL PLOT(0.,-S22.0.)
CALL PLOT(-S21.0.,.1)
CALL PLOT(0.,S23.1)
CALL PLOT(S22.0.,.1)
CALL CIS(-S21,0...5)
CALL CIS(0.,-S22..5)
CALL CIS(0.,S23..5)
CALL CIS(S22,0...5)
CALL CIS(-S11,0...5)
CALL CIS(0.,S13..5)
CALL CIS(0.,-S12..5)
CALL CIS(S12,0...5)
CALL TEX(-S21+0.2,0.2,0.,2.5,0,'S21',3)
CALL TEX(0.2,S23,0.,2.5,0,'S23',3)
CALL TEX(0.2,-S22,0.,2.5,0,'S22',3)
CALL TEX(S22+0.2,0.2,0.,2.5,0,'S22',3)
CALL TEX(-S11+0.2,0.2,0.,2.5,0,'S11',3)
CALL TEX(0.2,S13,0.,2.5,0,'S13',3)
CALL TEX(0.2,-S12,0.,2.5,0,'S12',3)
CALL TEX(S12+0.2,-5.,0.,2.5,0,'S12',3)
T(1)=65
U(1)=66
CALL REORDR(X1,Y1,Z1,XV,0,ZV,XP,YP,XQ,YQ,T,U)
T(1)=78
CALL REORPU(X,Y,7,XP,YP,T)
CALL CIS(-X,-Y,1.)
CALL CIS(-X,Z,1.)
CALL CIS(Y,Z,1.)
10 CONTINUE
CALL EOF
STOP
END

```

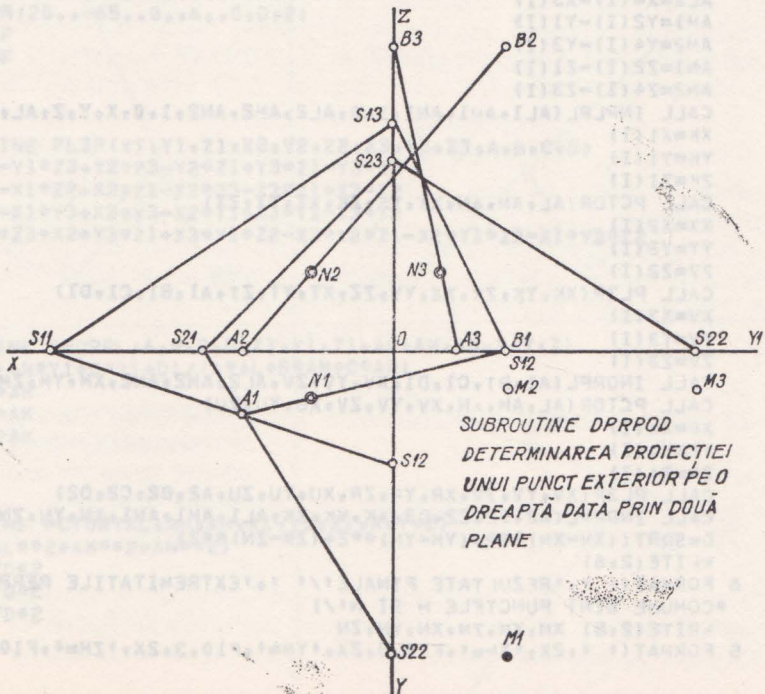


Fig. 2.8

Epura executată integral automat la plotter este dată în figura 2.8.

2.2.3. Subprogramul PECO

Determină și desenează perpendiculara comună dintre două drepte disjuncte, precum și lungimea perpendicularei comune.

```

C PROGRAM PECO PENTRU DETERMINAREA PERPENDICULAREI
C COMUNE DINTRE DOUA DREPTI DISJUNCTE DEFINITE PRIN
C CITE DOUA PUNCTE: X1,Y1,Z1-X2,Y2,Z2
C
  DIMENSION X1(100),Y1(100),Z1(100),X2(100),Y2(100),Z2(100)
  DIMENSION X3(100),Y3(100),Z3(100),X4(100),Y4(100),Z4(100)
  DIMENSION XP(3),YP(3),XQ(3),YQ(3)
  LOGICAL*1 T(2),U(?)
  CALL ASSIGN(3,'PP:')
  CALL ASSIGN(1,'CR:')
  CALL ASSIGN(2,'LP:')
  CALL INI(3)
  READ(1,1) N
1  FORMAT(I4)
  READ(1,2) (X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I),X3(I),Y3(I),Z3(I),
  *X4(I),Y4(I),Z4(I),I=1,N)
2  FORMAT(6F10.3,20X)
  DO 3 I=1,N
  WRITE(2,4) X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I)
4  FORMAT(' ',DREPTILE SINJ DETERMINATE DE PUNCTELE'/ ' ',X1=',F10.3
  *,2X,Y1=',F10.3,2X,Y2=',F10.3/' ',X2=',F10.3,2X,Y3=',F10.3,2X,
  *,Z2=',F10.3//)
  WRITE(2,8) X3(I),Y3(I),Z3(I),X4(I),Y4(I),Z4(I)
8  FORMAT(' ',X3=',F10.3,2X,Y3=',F10.3,2X,Z3=',F10.3/' ',X4=',F10
  *.3,2X,Y4=',F10.3,2X,Z4=',F10.3//)
  AL1=X2(I)-X1(I)
  AL2=X4(I)-X3(I)
  AM1=Y2(I)-Y1(I)
  AM2=Y4(I)-Y3(I)
  AN1=Z2(I)-Z1(I)
  AN2=Z4(I)-Z3(I)
  CALL INPLPL(AL1,AM1,AN1,1.0,AL2,AM2,AN2,1.0,X,Y,Z,AL,AM,AN)
  XK=X1(I)
  YK=Y1(I)
  ZK=Z1(I)
  CALL PCTDR(AL,AM,AN,XK,YK,ZK,XT,YT,ZT)
  XX=X2(I)
  YY=Y2(I)
  ZZ=Z2(I)
  CALL PL3P(XK,YK,ZK,XX,YY,ZZ,XT,YT,ZT,A1,B1,C1,D1)
  XV=X3(I)
  YV=Y3(I)
  ZV=Z3(I)
  CALL INDRPL(A1,B1,C1,D1,XV,YV,ZV,AL2,AM2,AN2,XM,YM,ZM)
  CALL PCTDR(AL,AM,AN,XV,YV,ZV,XU,YU,ZU)
  XR=X4(I)
  YR=Y4(I)
  ZR=Z4(I)
  CALL PL3P(XV,YV,ZV,XR,YR,ZR,XU,YU,ZU,A2,B2,C2,D2)
  CALL INDRPL(A2,B2,C2,D2,XK,YK,ZK,AL1,AM1,AN1,XN,YN,ZN)
  D=SQRT((XN-XM)**2+(YM-YN)**2+(ZN-ZM)**2)
  WRITE(2,6)
6  FORMAT(' ',REZULTATE FINALE'/ ' ',EXTREMITATILE PERPENDICULAREI
  *COMUNE SINT PUNCTELE M SI N'/)
  WRITE(2,5) XM,YM,ZM,XN,YN,ZN
5  FORMAT(' ',2X,XM=',F10.3,2X,YM=',F10.3,2X,ZM=',F10.3/' ',2X,

```



```

*XM=F10.3,2X,*YN=F10.3,2X,*ZN=F10.3//)
WRITE(2,7) D
7 FORMAT(' ',DISTANTA DINTRE CELE DOUA DREPTI DISJUNCTE (LUNGIMEA
*PERPENDICULAREI COMUNE)'/ ' ,D=F10.3//)
CALL TEX(10.,-25.,0.,4.,0.,'SUBROUTINE PECO',15)
CALL TEX(10.,-35.,0.,4.,0.,'DETERMINAREA PERPENDICULAREI',28)
CALL TEX(10.,-45.,0.,4.,0.,'COMUNE MN DINTRE DOUA DREPTI',28)
CALL TEX(10.,-55.,0.,4.,0.,'DISJUNCTE AB SI CE',18)
C TRASEAZA AXELE EPURE
CALL LIN(-130.,0.,130.,0.)''
CALL LIN(0.,-95.,0.,95.)
CALL TEX(1.,90.,0.,2.5,0.,'Z',1)
CALL TEX(1.,-95.,0.,2.5,0.,'Y',1)
CALL TEX(130.,1.,0.,2.5,0.,'Y1',2)
CALL TEX(-130.,1.,0.,2.5,0.,'X',1)
CALL TEX(.3.,3.,0.,2.5,0.,'O',1)
T(1)=65
U(1)=66
CALL REORDR(X1,Y1,Z1,X2,Y2,Z2,XP,YP,XQ,YQ,T,U)
T(1)=67
U(1)=69
CALL REORDR(X3,Y3,Z3,X4,Y4,Z4,XP,YP,XQ,YQ,T,U)
T(1)=77
U(1)=78
CALL REORDR(XM,YM,ZM,XN,YN,ZN,XP,YP,XQ,YQ,T,U)
T(1)=77
C LL REORPU(XM,YM,ZM,XP,YP,T)
T(1)=78
CALL REORPU(XN,YN,ZN,XP,YP,T)
CALL CIS(-XM,-YM,1.)
CALL CIS(-XM,ZM,1.)
CALL CIS(YM,ZM,1.)
CALL CIS(-XN,-YN,1.)
CALL CIS(-XN,ZN,1.)
CALL CIS(YN,ZN,1.)
CALL TEX(10.,-65.,0.,4.,0.,'D=',2)
CALL NUM(20.,-65.,0.,4.,0.,D,2)
3 CONTINUE
CALL EOF
STOP
END

SUBROUTINE PL3P(X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3,A,B,C,D)
A=Y1*Z2-Y1*Z3+Y2*Z3-Y2*Z1+Y3*Z1-Y3*Z2
B=X1*Z3-X1*Z2+X2*Z1-X2*Z3-X3*Z1+X3*Z2
C=X1*Y2-X1*Y3+X2*Y3-X2*Y1+X3*Y1-X3*Y2
D=X1*Y2*Z3+X2*Y3*Z1+X3*Y1*Z2-X3*Y2*Z1-X2*Y1*Z3-X1*Y3*Z2
D=-D
RETURN
END

SUBROUTINE INDRPI(A,B,C,D,X1,Y1,Z1,AL,AM,AN,X,Y,Z)
AK=(A*X1+B*Y1+C*Z1+D)/(A*AL+B*AM+C*AN)
X=X1-AL*AK
Y=Y1-AM*AK
Z=Z1-AN*AK
RETURN
END

SUBROUTINE PCTDR(AL,AM,AN,X1,Y1,Z1,X,Y,Z)
D=SQRT(AL**2+AM**2+AN**2)
X=X1+AL/D*2
Y=Y1+AM/D*2
Z=Z1+AN/D*2
RETURN
END

```

```

SUBROUTINE INPLP(A1,B1,C1,D1,A2,B2,C2,D2,X1,Y1,Z1,AL,AM,AN,
AL=B1*C2-B2*C1
AM=-A1*C2+A2*C1
AN=A1*B2-A2*B1
X1=(B1*D2-D1*B2)/(A1*B2-B1*A2)
Y1=(A2*D1-A1*D2)/(A1*B2-B1*A2)
Z1=0
RETURN
END

```

```

SUBROUTINE REORDR(XT,YT,ZT,XR,YR,ZR,XP,YP,XQ,YQ,T,U)
LOGICAL*1 T(1),U(1)
DIMENSION XP(3),YP(3),XQ(3),YQ(3)
CALL REORPU(XT,YT,ZT,XP,YP,T)
CALL REORPU(XR,YR,ZR,XQ,YQ,U)
DO 10 I=1,3
10 CALL LIN(XP(I),YP(I),XQ(I),YQ(I))
RETURN
END

```

```

SUBROUTINE REORPU(X,Y,Z,XP,YP,T)
LOGICAL*1 T(1)
DIMENSION XP(3),Yr(3)
XP(1)=-X
YP(1)=-Y
XP(2)=Z
YP(2)=Z
XP(3)=Y
YP(3)=Z
DO 1 I=1,3
CALL CIS(XP(I),YP(I),.5)
T(2)=48+I
CALL TEX(XP(I)+1.,YP(I)+1.,0.,2.5,0,T,2)
1 CONTINUE
RETURN
END

```

Epura executată integral automat la plotter este dată în figura 2.9. D este lungimea perpendiculară comună dintre cele două drepte disjuncte date.

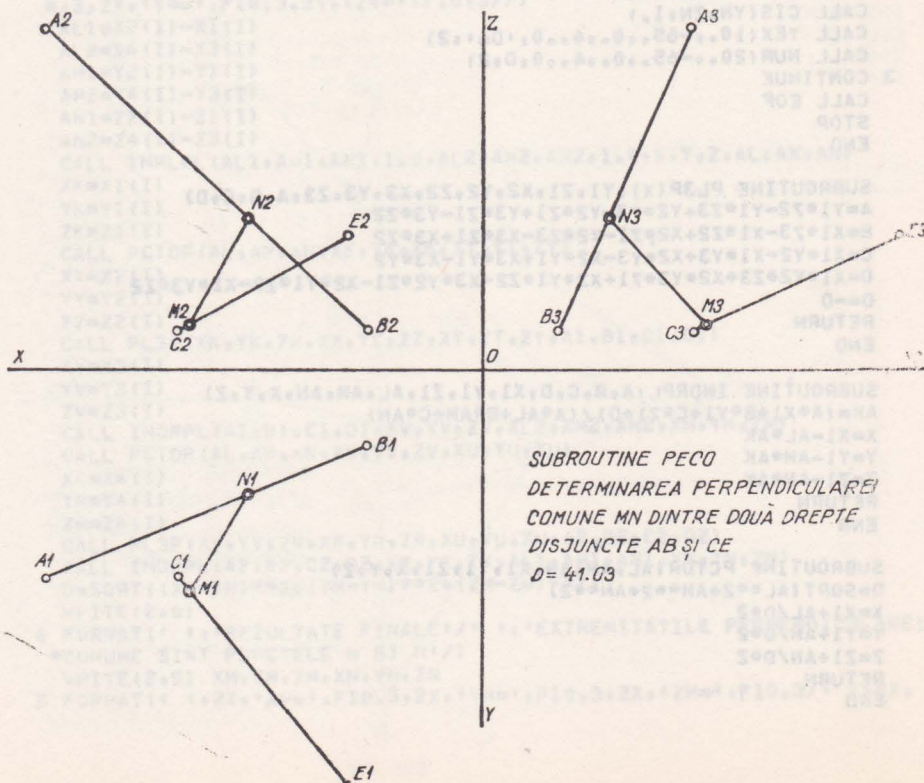


Fig. 2.9.

3.1. Secțiuni plane în poliedre

3.1.1 Generalități

Vom considera secțiunile plane în suprafețele poliedrale convexe sau concave de tip prismă (cub) sau piramidă; precum și combinații ale acestora, în situațiile în care planul de secțiune poate intersecta sau nu poligonul de bază al poliedrului. Dar studiul poate fi ușor extins pentru orice fel de suprafață poliedrală.

Planul de secțiune va fi definit prin parametrii săi directori A , B , C și D sau, eventual, prin alte elemente geometrice care-l pot determina. În acest caz, se va face legătura cu subprogramele corespunzătoare din capitolul II, pentru a se ajunge la calculul parametrilor directori ai planului. Vom nota, în general, parametrii directori ai muchiilor poliedrelor prin ALD , AMD și AND și vom defini poliedrele prin toate coordonatele vîrfurilor. Programul va fi intitulat **SECPOS** de la **SEC**țiuni plane în **POL**iedre și **Supra**fețe poliedrale. Un indicativ IND va specifica tipul poliedrului astfel:

$IND = 0$ poliedrul este o prismă (cub)

$IND = 1$ poliedrul este o piramidă

În acest fel, vom folosi același program pentru ambele tipuri de poliedre. Vîrfurile piramidei va avea coordonatele (V_1 , V_2 , V_3). Informația extrasă la ieșire va consta din coordonatele punctelor care alcătuiesc poligonul plan de secțiune. Mai departe se pune problema reprezentării grafice prin desen automat sau vizualizare în triplă proiecție ortogonală a conturilor aparente ale poliedrului și ale poligonului de secțiune.

3.1.2 Programul SECPOS pentru determinarea secțiunilor plane în suprafețele poliedrale.

Instrucțiunea de apel a programului **SECPOS** este următoarea:

**CALL SECPOS (N, IND, A, B, C, D, VAR1, VAR2, VAR3, X1, Y1, Z1,
X, Y, Z)**

în care parametrii au următoarea semnificație:

N

— Numărul vîrfurilor poligonului de bază la piramidă sau numărul tuturor vîrfurilor celor două baze ale prisme (cub).

- IND* — Indicativul care specifică forma poliedrului $IND = 0$ pentru prismă, $IND = 1$ pentru piramidă
- A, B, C, D* — Parametrii directori ai planului de secțiune
- VAR1* — Variabile care reprezintă parametrii directori ai muchiei prismei dacă $IND = 0$ sau reprezintă coordonatele $V1, V2, V3$, ale vârfului piramidei dacă $IND = 1$
- VAR2*
- VAR3*
- X1, Y1, Z1* — Coordonatele celor N vîrfuri ale poligonului (poligoanelor) de bază ale poliedrelor
- X, Y, Z* — Coordonatele punctelor de intersecție (vîrfurile poligonului de secțiune).

Subprogramele apelate în cadrul programului **SECPOS** sînt următoarele:

1. Subprogramul **CALCUL**: — Calculează coordonatele vîrfurilor poligonului de secțiune prin intersecția planului cu o dreaptă pe care se află segmentul reprezentînd muchia poliedrului.
2. Subprogramul **LIMITE**: — Stabilește limitele inferioare și superioare ale coordonatelor punctelor care determină fiecare muchie a poliedrului.
3. Subprogramul **BAZA**: — Stabilește ordinea în care se chiamă subprogramul **INLAT**.
4. Subprogramul **INLAT**: — Calculează coordonatele punctului de intersecție dintre planul de secțiune și o latură a poligonului de bază.
5. Subprogramul **CLSDR**: — Clasifică muchiile poliedrului în funcție de parametrii directori ai acestora.
6. Subprogramul **INTMU**: — Stabilește dacă fiecare punct de intersecție se află între cele două puncte care definesc muchiile poliedrului.

Aceste subprograme sînt foarte utile și în studiul intersecției de poliedre așa cum vom vedea în Programul **INTPOL**

În comentariile programului principal **SECPOS** este specificat faptul că el poate fi utilizat și în problemele de secțiuni plane în suprafețe cilindrice sau conice asimilînd muchiile poliedrelor cu generatoarele (în număr mult mai mare) ale suprafețelor. De fapt problema se reduce tot la repetitiva intersecție dintre o dreaptă și un plan de secțiune definit în diverse moduri așa cum se va vedea mai departe în capitolul respectiv.

Programul SECPOS

```

C PROGRAMUL PENTRU DETERMINAREA SECȚIUNILOR PLANE IN POLIEDRE DE TIP
C PIRAMIDA-PRISMA SI COMBINATII ALE ACESTORA
C DIMENSION X1(200),Y1(200),Z1(200),AL(200),AM(200),AN(200)
C * X(200),Y(200),Z(200)
C INDICATIVUL IND SPECIFICA FORMA POLIEDRULUI
C DACA IND=0 POLIEDRUL ESTE O PRISMA SAU UN CILINDRU
C DACA IND=1 POLIEDRUL ESTE O PIRAMIDA SAU UN CON
C A,B,C,D SINT PARAMETRII CARE DEFINESC PLANUL DE INTERSECȚIE
C ALD,AMD,AND SINT PARAMETRII DIRECTORI AI DREPTELOR CARE SINT
C MUCHII SAU GENERATOARE ALE POLIEDRELOR
C N REPREZINTA NUMARUL DE PUNCTE DE PE CONTURUL BAZEI
  READ(105,1) N,IND
  1 FORMAT(14,4X,12,70X)
  IF(IND.EQ.0) GO TO 100
  WRITE(108,2)
  2 FORMAT(' ',10X,'PIRAMIDA=CON')
  READ(105,3) A,B,C,D,V1,V2,V3
  3 FORMAT(7F10,3,10X)
  WRITE(108,24)
  24 FORMAT(' ',4X,'DATELE INITIALE FIXE')
  WRITE(108,14) A,B,C,D,V1,V2,V3
  4 FORMAT(' ',4X,'A=',F10,3,2X,'B=',F10,3,2X,'C=',F10,3,2X,'D=',F10,3
  *,2X,'V1=',F10,3,2X,'V2=',F10,3,2X,'V3=',F10,3//)
  NR=1
  GO TO 101
100 WRITE(108,5)
  5 FORMAT(' ',10X,'PRISMA-CILINDRU')
  READ(105,6) A,B,C,D,ALD,AMD,AND
  6 FORMAT(7F10,3,10X)
  WRITE(106,26)
  26 FORMAT(' ',4X,'DATELE INITIALE FIXE')
  WRITE(108,7) A,B,C,D,ALD,AMD,AND
  7 FORMAT(' ',4X,'A=',F10,3,2X,'B=',F10,3,2X,'C=',F10,3,2X,'D=',F10,3
  *,2X,'AL=',F10,3,2X,'AM=',F10,3,'AN=',F10,3//)
  NR=0
101 READ(105,8) (X1(I),Y1(I),Z1(I),I=1,N)
  8 FORMAT(6E10,3,20X)
  IF(IND.EQ.0) GO TO 155
  M=N
  GO TO 156
155 M=N/2
156 DO 107 I=1,M
  WRITE(108,48) I
  48 FORMAT(' ',10X,'CALCULUL PENTRU I=',14,' ',4X,'COORDONATELE PUNCTULUI
  *DE PE CONTURUL BAZEI')

  WRITE(108,25) X1(I),Y1(I),Z1(I)
  25 FORMAT(' ',4X,'X1=',F10,3,2X,'Y1=',F10,3,2X,'Z1=',F10,3//)
  IF(NR.EQ.0) GO TO 102
  AL(I)=V1-X1(I)
  AM(I)=V2-Y1(I)
  AN(I)=V3-Z1(I)
  ALD=AL(I)
  AMD=AM(I)
  AND=AN(I)
102 K=A*ALD+B*AMD+C*AND
  IF(ALD.EQ.0) GO TO 103
  IF(AMD.EQ.0) GO TO 104
  IF(AND.EQ.0) GO TO 105
  WRITE(108,9)
  9 FORMAT(' ',5X,'DREAPTA ESTE OARECARE')
  IF(K.EQ.0) GO TO 106
  XK=X1(I)
  YK=Y1(I)
  ZK=Z1(I)
  CALL CALCUL(A,B,C,D,XK,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,XP2,YP2,
  *XP3,YP3)
  IF(IND.EQ.0) GO TO 136
  CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,XP3,YP3,I
  *ND,XT,YT,ZT,I)
  GO TO 137
136 V1=0
  V2=0

```

```

V3=0
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,X73,YP3,I
*ND,XT,YT,ZT,I)
137 CONTINUE
X(I)=XT
Y(I)=YT
Z(I)=ZT
GO TO 107
106 WRITE(108,10)
10 FORMAT(' ',5X,'DREAPTA NU INTERSECTEAZA PLANUL'/)
GO TO 107
105 WRITE(108,11)
11 FORMAT(' ',5X,'DREAPTA ESTE ORIZONTALA'/)
IF(K.EQ.0) GO TO 108
XK=X1(I)
YK=Y1(I)
ZK=Z1(I)
CALL CALCUL(A,B,C,D,XK,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3)
IF(IND.EQ.0) GO TO 138

CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,XP3,YP3,I
*ND,XT,YT,ZT,I)
GO TO 150
138 V1=0
V2=0
V3=0
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,X73,YP3,I
*ND,XT,YT,ZT,I)
150 CONTINUE
X(I)=XT
Y(I)=YT
Z(I)=ZT
GO TO 107
108 WRITE(108,12)
12 FORMAT(' ',5X,'DREAPTA NU INTERSECTEAZA PLANUL'/)
GO TO 107
104 IF(AND.EQ.0) GO TO 109
WRITE(108,13)
13 FORMAT(' ',5X,'DREAPTA ESTE FRONTALA'/)
IF(K.EQ.0) GO TO 110
XK=X1(I)
YK=Y1(I)
ZK=Z1(I)
CALL CALCUL(A,B,C,D,XK,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3)
IF(IND.EQ.0) GO TO 139
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,XP3,YP3,I
*ND,XT,YT,ZT,I)
GO TO 151
139 V1=0
V2=0
V3=0
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,X73,YP3,I
*ND,XT,YT,ZT,I)
151 CONTINUE
X(I)=XT
Y(I)=YT
Z(I)=ZT
GO TO 107
110 WRITE(108,14)
14 FORMAT(' ',5X,'DREAPTA NU INTERSECTEAZA PLANUL'/)
GO TO 107
109 WRITE(108,15)
15 FORMAT(' ',5X,'DREAPTA ESTE FRONTOORIZONTALA'/)
IF(K.EQ.0) GO TO 111
XK=X1(I)
YK=Y1(I)

```

```

ZK=Z1(I)
CALL CALCUL(A,B,C,D,XK,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3)
IF(IND.EQ.0) GO TO 140
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,XP3,YP3,I
*ND,XT,YT,ZT,I)
GO TO 141
140 V1=C
V2=C
V3=C
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,X73,YP3,I
*ND,X1,Y1,Z1,I)
141 CONTINUE
X(I)=XT
Y(I)=YT
Z(I)=ZT
111 WRITE(108,16)
16 FORMAT(' ',5X,'DREAPTA NU INTERSECTEAZA PLANUL/')
GO TO 107
103 IF(AMD.EQ.0) GO TO 112
IF(AND.EQ.0) GO TO 113
WRITE(108,17)
17 FORMAT(' ',5X,'DREAPTA ESTE DE PROFIL/')
IF(K.EQ.0) GO TO 114
XK=X1(I)
YK=Y1(I)
ZK=Z1(I)
CALL CALCUL(A,B,C,D,XK,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3)
IF(IND.EQ.0) GO TO 142
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,XP3,YP3,I
*ND,XT,YT,ZT,I)
GO TO 143
142 V1=C
V2=C
V3=C
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,X73,YP3,I
*ND,XT,YT,ZT,I)
143 CONTINUE
X(I)=XT
Y(I)=YT
Z(I)=ZT
GO TO 107
110 WRITE(108,18)
18 FORMAT(' ',5X,'DREAPTA NU INTERSECTEAZA PLANUL/')
GO TO 107
113 WRITE(108,19)
19 FORMAT(' ',5X,'DREAPTA ESTE DE CAPAT/')
IF(K.EQ.0) GO TO 115
XK=X1(I)
YK=Y1(I)
ZK=Z1(I)
CALL CALCUL(A,B,C,D,XK,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3)
IF(IND.EQ.0) GO TO 144
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,XP3,YP3,I
*ND,XT,YT,ZT,I)
GO TO 145
144 V1=C
V2=C
V3=C
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,X73,YP3,I
*ND,XT,YT,ZT,I)
145 CONTINUE
X(I)=XT
Y(I)=YT
Z(I)=ZT
GO TO 107
115 WRITE(108,20)
20 FORMAT(' ',5X,'DREAPTA NU INTERSECTEAZA PLANUL/')
GO TO 107
112 IF(AND.EQ.0) GO TO 116
WRITE(108,21)
21 FORMAT(' ',5X,'DREAPTA ESTE VERTICALA/')
IF(K.EQ.0) GO TO 117
XK=X1(I)
YK=Y1(I)
ZK=Z1(I)

```

```

CALL CALCUL(A,B,C,D,XK,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3)
IF(IND.EQ.0) GO TO 146
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,XP3,YP3,I
*ND,XT,YT,ZT,I)
GO TO 147
146 V1=0
V2=0
V3=0
CALL LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,X73,YP3,I
*ND,XT,YT,ZT,I)
147 CONTINUE
X(I)=XT
Y(I)=YT
Z(I)=ZT
GO TO 107

117 WRITE(108,22)
22 FORMAT(' ',5X,'DREAPTA NU INTERSECTEAZA PLANUL')
GO TO 107
116 WRITE(108,23)
23 FORMAT(' ',5X,'DREAPTA ARE AL=0 AM=0 AN=0')
107 CONTINUE
IF(IND.EQ.0) GO TO 210
WRITE(108,211)
211 FORMAT('///PUNCTELE DE INTERSECȚIE ALE PLANULUI CU LATURILE POLIGO
*NU LUI DE LA BAZA PIRAMIDEI-CONULUI')
L=1
K=N-1
CALL BAZA(N,X1,Y1,Z1,A,B,C,D,L,K,IND)
GO TO 220
210 WRITE(108,212)
212 FORMAT('///PUNCTELE DE INTERSECȚIE ALE PLANULUI CU LATURILE POLIGO
*NELOR DE LA BAZA ALE PRISMEI-CILINDRULUI')
L=N/2+1
K=N-1
CALL BAZA(N,X1,Y1,Z1,A,B,C,D,L,K,IND)
220 CONTINUE
STOP
END

```

3.1.3 Subprogramul CALCUL

Acest subprogram calculează coordonatele vîrfurilor poligonului de secțiune prin intersecția planului cu o dreaptă pe care se află segmentul reprezentînd muchia poliedrului.

Instrucțiunea de apel a subprogramului **CALCUL** este
CALL CALCUL(A, B, C, D, XK, YK, ZK, ALD, AMD, AND, XS, YS, ZS,
XP1, YP1, XP2, YP2, XP3, YP3)

unde semnificația parametrilor este următoarea:

- | | |
|-------------------|--|
| <i>A, B, C, D</i> | — Parametrii directori ai planului de secțiune. |
| <i>XK, YK, ZK</i> | — Coordonatele punctului prin care trece muchia poliedrului (suportul muchiei). |
| <i>ALD</i> | — Parametrii directori ai muchiei considerate a poliedrului. |
| <i>AMD</i> | |
| <i>AND</i> | |
| <i>XS, YS, ZS</i> | — Coordonatele spațiale ale punctului de intersecție dintre suportul muchiei și planul de secțiune (nu este încă vîrf al poligonului de secțiune). |
| <i>XP1, YP1</i> | — Coordonatele proiecțiilor <i>MONGE</i> orizontală, verticală și laterală ale punctului <i>XS, YS, ZS</i> . |
| <i>XP2, YP2</i> | |
| <i>XP3, YP3</i> | |


```

SUBROUTINE CALCUL(A,B,C,D,XX,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,
C CALCULEAZA COORDONATELE PUNCTULUI DE INTERSECȚIE DINTRE PLANUL DAT DE
C A,B,C,D SI O DREAPTA PE CARE SE AFLA SEGMENTUL REPREZENTIND MUCHIA
C POLIEDRULUI
*XP2,YP2,XP3,YP3)
AK=(A*XX+B*YK+C*ZK+D)/(A*ALD+B*AMD+C*AND)
XS=AK*ALD
YS=YK-AMD*AK
ZS=ZK-AND*AK
CALL FLT(XS)
CALL FLT(YS)
CALL FLT(ZS)
XP1==XS
YP1==YS
XP2==XS
YP2==ZS
XP3==YS
YP3==ZS
RETURN
END

```

3.1.4 Subprogramul LIMITE

Acest subprogram stabilește limitele inferioare și superioare ale coordonatelor punctelor care determină fiecare muchie a poliedrului.

Instrucțiunea de apel a subprogramului **LIMITE** este:

CALL LIMITE (N, V1, V2, V3, X1, Y1, Z1, XS, YS, ZS, XP1, YP1, XP2, YP2, XP3, YP3, IND, XT, YT, ZT, I)

unde semnificația parametrilor este următoarea:

- N** — Numărul vîrfurilor poligonului de bază la piramidă sau numărul tuturor vîrfurilor celor două baze ale prisme (cub).
- V1, V2, V3** — Coordonatele vîrfului piramidei
- X1, Y1, Z1** — Coordonatele celor N vîrfuri ale poligonului (poligoanelor) de bază ale poliedrelor.
- XS, YS, ZS** — Coordonatele de intrare aflate prin subprogramul **CALCUL**.
- XP1, YP1** — Coordonatele proiecțiilor *Monge* orizontală, verticală și laterală ale punctului XS, YS, ZS (sau XT, YT, ZT) dacă este vîrf al poligonului de secțiune).
- XP2, YP2**
- XP3, YP3**
- IND** — Indicativul care specifică forma poliedrului.
- XT, YT, ZT** — Coordonatele punctelor rezultate din **INTMU** deci coordonatele vîrfurilor poligonului de secțiune.
- I** — Indice de ordine al muchiei apelate.

În cadrul acestui subprogram este apelată **SUBROUTINE INTMU**.

```

SUBROUTINE LIMITE(N,V1,V2,V3,X1,Y1,Z1,XS,YS,ZS,XP1,YP1,XP2,YP2,XP3
C STABILESTE LIMITELE INFERIOARE SI SUPERIOARE ALE COORDONATELOR
PUNCTELOR DETERMINA FIECARE MUCHIE
* ,YP3,IND,XI,YT,ZT,I)
DIMENSION XI(200),YI(200)
M=N/2
IF(IND.EQ.0) GO TO 56
IF(X1(I).LE.V1) GO TO 57
XINF=V1
XSUP=X1(I)
GO TO 58
57 XINF=X1(I)
XSUP=V1
58 CONTINUE
IF(Y1(I).LE.V2) GO TO 59
YINF=V2
YSUP=Y1(I)
GO TO 70
59 YINF=Y1(I)
YSUP=V2
70 CONTINUE
IF(Z1(I).LE.V3) GO TO 71
ZINF=V3
ZSUP=Z1(I)
GO TO 72
71 ZINF=Z1(I)
ZSUP=V3
72 CONTINUE
CALL INTMU(XINF,YINF,XSUP,YSUP,ZINF,ZSUP,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3,XT,YT,ZT)
GO TO 73
56 IF(I.GT.M) GO TO 73
IF(X1(I).LE.X1(M+I)) GO TO 74
XINF=X1(M+I)
XSUP=X1(I)
GO TO 75
74 XINF=X1(I)
XSUP=X1(M+I)
75 CONTINUE
IF(Y1(I).LE.Y1(M+I)) GO TO 76
YINF=Y1(M+I)
YSUP=Y1(I)
GO TO 77
76 YINF=Y1(I)
YSUP=Y1(M+I)
77 CONTINUE
IF(Z1(I).LE.Z1(M+I)) GO TO 78

ZINF=Z1(M+I)
ZSUP=Z1(I)
GO TO 79
78 ZINF=Z1(I)
ZSUP=Z1(M+I)
79 CONTINUE
CALL INTMU(XINF,YINF,XSUP,YSUP,ZINF,ZSUP,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3,XT,YT,ZT)
73 CONTINUE
RETURN
END

```

3.1.5 Subprogramul BAZA

Acest subprogram stabilește ordinea în care se chiamă subprogramul INLAT.

Instrucțiunea de apel a subprogramului BAZA este:

CALL BAZA(N, X1, Y1, Z1, A, B, C, D, L, K, IND)

unde semnificația parametrilor este următoarea:

- N — Numărul vîrfurilor poligonului de bază la piramidă sau numărul tuturor vîrfurilor celor două baze ale prisme (cub).

- $X1, Y1, Z1$ — Coordonatele celor N vîrfuri ale poligonului (poligoanelor) de bază ale poliedrelor
- A, B, C, D — Parametrii directori ai planului.
- L — $L = \frac{N}{2} + 1$
- K — $K = N - 1$
- IND — Indicativul care specifică forma poliedrului.

În cadrul acestui subprogram este apealtă SUBROUTINA INLAT.

```

SUBROUTINE BAZA(N,X1,Y1,Z1,A,B,C,D,L,K,IND)
C STABILESTE ORDINEA IN CARE TREBUIE CHEMATA SUBROUTINA INLAT
DIMENSION X1(200),Y1(200),Z1(200)
IF(IND.EQ.C) GO TO 2
M=N-1
DO 1 I=1,M
  J=1
  1 CALL INLAT(I,X1,Y1,Z1,A,B,C,D,M,J)
  CONTINUE
GO TO 10
2 M=L-2
DO 3 I=1,M
  J=1
  3 CALL INLAT(I,X1,Y1,Z1,A,B,C,D,M,J)
  CONTINUE
M=K
DO 4 I=L,M
  J=L
  4 CALL INLAT(I,X1,Y1,Z1,A,B,C,D,M,J)
  CONTINUE
10 RETURN
END

```

3.1.6 Subprogramul INLAT

Acest subprogram calculează coordonatele punctului de intersecție dintre planul de secțiune și o latură a poligonului de bază.

Instrucțiunea de apel a subprogramului **INLAT** este:

CALL INLAT(I, X1, Y1, Z1, A, B, C, D, M, J)

unde semnificația parametrilor este următoarea:

- I — Indice de ordine al muchiei apelate (stabilește muchia).
- $X1, Y1, Z1$ — Coordonatele celor N vîrfuri ale poligonului (poligoanele) de bază ale poliedrelor.
- A, B, C, D — Parametrii directori ai planului.
- M — $M = N - 1$ pentru piramidă. Pentru prismă $M = L - 2$ sau $M = K$ în funcție de baza studiată.
- J — $J = 1$ pentru piramidă. Pentru prismă dacă $M = L - 2$ atunci $J = 1$, dacă $M = K$ atunci $J = L$.

În cadrul acestui subprogram sînt apelate următoarele subrutine:

SUBROUTINE CLSDR SUBROUTINE CALCUL SUBROUTINE INTMU

```

SUBROUTINE INLAT(I,X1,Y1,Z1,A,B,C,D,M,J)
C  CALCULEAZA COORDONATELE PUNCIULUI DE INTERSECȚIE DINTRE PLANUL DAT
DE A,B,C,D ȘI D LATURA A BAZEI POLIEDRULUI
DIMENSION X1(200),Y1(200),Z1(200)
XK=X1(I)
YK=Y1(I)
ZK=Z1(I)
ALD=X1(I)-X1(I+1)
AMD=Y1(I)-Y1(I+1)
AND=Z1(I)-Z1(I+1)
WRITE(108,14) X1(I),Y1(I),Z1(I),X1(I+1),Y1(I+1),Z1(I+1)
14  FORMAI(' ',2X,'PENTRU LATURA DETERMINATA DE PUNCTELE',/ ' ',4X,'X1='
*,F10.3,2X,'Y1=',F10.3,2X,'Z1=',F10.3/ ' ',4X,'X2=',F10.3,2X,'Y2=',F
*10.3,2X,'Z2=',F10.3/)
CALL CLSDR(ALD,AMD,AND)
*CALL CALCUL(A,B,C,D,XK,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3)
IF(X1(I).LE.X1(I+1)) GO TO 1
XSUP=X1(I)
XINF=X1(I+1)
GO TO 2
1  XINF=X1(I)
XSUP=X1(I+1)
2  CONTINUE
IF(Y1(I).LE.Y1(I+1)) GO TO 3
YSUP=Y1(I)
YINF=Y1(I+1)
GO TO 4
3  YINF=Y1(I)
YSUP=Y1(I+1)
4  CONTINUE
IF(Z1(I).LE.Z1(I+1)) GO TO 5
ZSUP=Z1(I)
ZINF=Z1(I+1)
GO TO 6
5  ZINF=Z1(I)
ZSUP=Z1(I+1)
6  CONTINUE
CALL INTMU(XINF,YINF,XSUP,YSUP,ZINF,ZSUP,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3,XT,YT,ZT)
20  IF(1.NE.M) GO TO 10
I=M+1
XK=X1(J)
YK=Y1(J)
ZK=Z1(J)
ALD=X1(J)-X1(I)
AMD=Y1(J)-Y1(I)
AND=Z1(J)-Z1(I)
WRITE(108,15) X1(J),Y1(J),Z1(J),X1(I),Y1(I),Z1(I)
15  FORMAI(' ',2X,'PENTRU LATURA DETERMINATA DE PUNCTELE',/ ' ',4X,'X1='
*,F10.3,2X,'Y1=',F10.3,2X,'Z1=',F10.3/ ' ',4X,'X2=',F10.3,2X,'Y2=',F
*10.3,2X,'Z2=',F10.3/)
CALL CLSDR(ALD,AMD,AND)
CALL CALCUL(A,B,C,D,XK,YK,ZK,ALD,AMD,AND,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3)
IF(X1(J).LE.X1(I)) GO TO 7
XSUP=X1(J)
XINF=X1(I)
GO TO 8
7  XINF=X1(J)
XSUP=X1(I)
8  CONTINUE
IF(Y1(J).LE.Y1(I)) GO TO 9
YSUP=Y1(J)
YINF=Y1(I)
GO TO 11
9  YINF=Y1(J)
YSUP=Y1(I)
11  CONTINUE
IF(Z1(J).LE.Z1(I)) GO TO 12
ZSUP=Z1(J)
ZINF=Z1(I)
GO TO 13
12  ZINF=Z1(J)
ZSUP=Z1(I)
13  CONTINUE
CALL INTMU(XINF,YINF,XSUP,YSUP,ZINF,ZSUP,XS,YS,ZS,XP1,YP1,XP2,YP2,
*XP3,YP3,XT,YT,ZT)
10  RETURN
END

```

3.1.7 Subprogramul CLSDR

Acest subprogram clasifică muchiile poliedrului în funcție de parametrii directori *ALD*, *AMD*, *AND* ai acestora.

Instrucțiunea de apel a subprogramului **CLSDR** este:

CALL CLSDR (ALD, AMD, AND)

```

SUBROUTINE CLSDR(ALD,AMD,AND)
IF(ALD.EQ.0) GO TO 1
IF(AMD.EQ.0) GO TO 6
IF(AND.EQ.0) GO TO 8
WRITE(108,20)
20 FORMAT(' ',2X,'DREAPTA ESTE DARECARE')
GO TO 5
8 WRITE(108,21)
21 FORMAT(' ',2X,'DREAPTA ESTE ORIZONTALA')
GO TO 5
6 IF(AND.EQ.0) GO TO 7
WRITE(108,22)
22 FORMAT(' ',2X,'DREAPTA ESTE FRONTALA')
GO TO 5
7 WRITE(108,23)
23 FORMAT(' ',2X,'DREAPTA ESTE FRONTOORIZONTALA')
GO TO 5
1 IF(AMD.EQ.0) GO TO 2
IF(AND.EQ.0) GO TO 4
WRITE(108,24)
24 FORMAT(' ',2X,'DREAPTA ESTE DE PROFIL')
GO TO 5
4 WRITE(108,25)
25 FORMAT(' ',2X,'DREAPTA ESTE DE CAPAT')
GO TO 5
2 IF(AND.EQ.0) GO TO 3
WRITE(108,26)
26 FORMAT(' ',2X,'DREAPTA ESTE VERTICALA')
GO TO 5
3 WRITE(108,27)
27 FORMAT(' ',2X,'EROARE')
5 RETURN
END

```

3.1.8 Subprogramul INTMU

Acest subprogram stabilește dacă fiecare punct de intersecție se află între cele două puncte care definesc muchiile poliedrului.

Instrucțiunea de apel a subprogramului **INTMU** este:

CALL INTMU (XINF, YINF, ZINF, XSUP, YSUP, ZSUP, XS, YS, ZS, XP1, YP1, XP2, YP2, XP3, YP3, XT, YT, ZT)

unde semnificația parametrilor este următoarea:

XINF, YINF, ZINF — Valorile minime ale coordonatelor care definesc

prin extremități muchia poliedrului. La fel pentru valorile maxime.

XSUP, YSUP, ZSUP

XS, YS, ZS

— Coordonatele de intrare aflate prin subprogramul **CALCUL**.

XP1, YP1

XP2, YP2,

XP3, YP3

— Coordonatele proiecțiilor *Monge* orizontală, verticală și laterală ale punctului *XS, YS, ZS* (sau *XT, YT, ZT* dacă este vîrf al poligonului de secțiune).

XT, YT, ZT

— Coordonatele punctelor rezultate din **INTMU** deci coordonatele vîrfurilor poligonului de secțiune dacă există.

```

SUBROUTINE INTNU(XINF,YINF,XSUP,YSUP,ZINF,ZSUP,XS,YS,ZS,XP1,YP1,XP
*2,YP2,XP3,YP3,XT,YT,ZT)
C STABILESTE DACA FIECARE PUNCT DE INTERSECȚIE SE AFLA ÎNTR-UNEA
PUNCTE CARE DETERMINA MUCHIA POLIEDRULUI
IF(XINF.LE.XS) GO TO 1
GO TO 10
1 IF(XS.LE.XSUP) GO TO 2
GO TO 10
2 XT=XS
IF(YINF.LE.YS) GO TO 3
GO TO 10
3 IF(YS.LE.YSUP) GO TO 4
GO TO 10
4 YT=YS
IF(ZINF.LE.ZS) GO TO 5
GO TO 10
5 IF(ZS.LE.ZSUP) GO TO 6
GO TO 10
6 ZT=ZS
GO TO 11
10 XT=9999
YT=9999
ZT=9999
11 IF(XT.EQ.9999) GO TO 12
IF(YT.EQ.9999) GO TO 12
IF(ZT.EQ.9999) GO TO 12
GO TO 13
12 WRITE(108,14)
14 FORMAT(108,4X,'PLANUL NU INTERSECȚEAZA MUCHIA'///)
GO TO 16
13 WRITE(108,15) XT,YT,ZT
15 FORMAT(108,4X,'PUNCTUL DE INTERSECȚIE ARE COORDONATELE'/' ',4X,
'X=',F10.3/' ',4X,'Y=',F10.3/' ',4X,'Z=',F10.3/)
WRITE(108,40) XP1,YP1,XP2,YP2,XP3,YP3
40 FORMAT(108,4X,'XP1=',F10.3,2X,'YP1=',F10.3,2X,'XP2=',F10.3,2X,'YP2
=',F10.3,2X,'XP3=',F10.3,2X,'YP3=',F10.3///)
16 RETURN
END

```

3.2. Programele de desen automat sau de vizualizare pentru poliedre și pentru poligonul de secțiune

3.2.1 Generalități

Desenul automat al poliedrelor și al poligoanelor de secțiune sînt concepute în reprezentarea triplu ortogonală *Monge*.

Desenele pot fi mai mult sau mai puțin complete în funcție de ceea ce se cere în sensul că unele puncte pot fi sau nu balustrate, pot fi sau nu notate, liniile pot fi sau nu trasate cu aceeași culoare sau cu culori diferite, pot fi scrise sau nu texte pe epure și așa mai departe. Deci desenul rezultat va reflecta exact ceea ce îi vom cere echipamentului periferic automat să execute cu mijloacele și cu caracteristicile pe care le are din fabricație dar și cu un material software pe care trebuie să-l gîndim bine în corelația cu caracteristicile echipamentului respectiv.

Programele (subprogramele) de desen pe care în general le vom utiliza în desenul automat al poliedrelor vor fi următoarele:

3.2.2 Subprogramul DR1P

Subrutină de desen pentru un segment de dreaptă cu balustrarea și notarea primei extremități a segmentului.

Instrucțiunea de apel este:

CALL DR1P(XC1, YC1, ZC1, XC2, YC2, ZC2, I)

unde parametrii sînt coordonatele extremităților segmentului.

```

0001      SUBROUTINE DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2)
0002      XP1=-XC1
0003      YP1=-YC1
0004      XP2=-XC1
0005      YP2=ZC1
0006      XP3=YC1
0007      YP3=ZC1
0008      XQ1=-XC2
0009      YQ1=-YC2
0010      XQ2=-XC2
0011      YQ2=ZC2
0012      XQ3=YC2
0013      YQ3=ZC2
0014      CALL PLOT(XP1-0.1,YP1,3)
0015      CALL CIRCLE(XP1,YP1,360.)
0016      CALL PLOT(XP1,YP1,3)
0017      CALL PLOT(XQ1,YQ1,2)
0018      CALL PLOT(XP2-0.1,YP2,3)
0019      CALL CIRCLE(XP2,YP2,360.)
0020      CALL PLOT(XP2,YP2,3)
0021      CALL PLOT(XQ2,YQ2,2)
0022      CALL PLOT(XP3-0.1,YP3,3)
0023      CALL CIRCLE(XP3,YP3,360.)
0024      CALL PLOT(XP3,YP3,3)
0025      CALL PLOT(XQ3,YQ3,2)
0026      RETURN
0027      END

```

3.2.3 Subprogramul DRFP

Subrutină de desen pentru un segment de dreaptă simplu fără balustrări și notații.

Instrucțiunea de apel este:

CALL DRFP(XC1, YC1, ZC1, XC2, YC2, ZC2)

unde parametrii sînt coordonatele extremităților segmentului.

```

SUBROUTINE DRFP(XC1,YC1,ZC1,XC2,YC2,ZC2)
XP1=-XC1
YP1=-YC1
XP2=-XC1
YP2=ZC1
XP3=YC1
YP3=ZC1
XQ1=-XC2
YQ1=-YC2
XQ2=-XC2
YQ2=ZC2
XQ3=YC2
YQ3=ZC2
CALL PLOT(XP1,YP1,3)
CALL PLOT(XQ1,YQ1,2)
CALL PLOT(XP2,YP2,3)
CALL PLOT(XQ2,YQ2,2)
CALL PLOT(XP3,YP3,3)
CALL PLOT(XQ3,YQ3,2)
RETURN
END

```

3.2.4 Programul POLIGO(N)

Program principal pentru desenul secțiunii plane în poliedru (poligon cu N laturi) în triplă proiecție ortogonală.

```

DIMENSION X1(100),Y1(100),Z1(100)
LOGICAL*1 V1(10),V2(10),V3(10),S1(2),S2(2),S3(2),LF(32)
DATA V1/'1','1','2','1','3','1','4','1','5','1','6','1','7','1',
*'8','1','9','1','10'/
DATA V2/'1','2','2','3','2','4','2','5','2','6','2','7','2',
*'8','2','9','2','10'/
DATA V3/'1','3','2','3','3','3','4','3','5','3','6','3','7','3',
*'8','3','9','3','10'/
TYPE 1
1 FORMAT(' NUME FISIER DATE=? ',*)
ACCEPT 2,10,(LF(I),I=1,10)
2 FORMAT(3,32A1)
CALL ASSIGN(2,LF,10)
READ(2,*) N
DO 13 I=1,N
13 READ(2,*) X1(I),Y1(I),Z1(I)
CALL CLOSE(2)
CALL PLOTS(3,1,7)
CALL NEWPE(3)
J=1
DO 10 I=1,N-1
S1(1)=V1(J)
S1(2)=V1(J+1)
S2(1)=V2(J)
S2(2)=V2(J+1)
S3(1)=V3(J)
S3(2)=V3(J+1)
XP1=-X1(I)
YP1=-Y1(I)
XP2=-X1(I)
YP2=Z1(I)
XP3=Y1(I)
YP3=Z1(I)
X+1=-X1(I+1)
Y+1=-Y1(I+1)
Z+1=Z1(I+1)
X+3=Y1(I+1)
Y+3=Z1(I+1)

```

Felul în care se desenează sau se vizualizează poligonul secțiunii plane în poliedru depinde, desigur, de intențiile utilizatorului. De aceea, este bine ca în acest program care oferă un minim de date, să se mai reflecte și intențiile despre care am amintit mai înainte. Astfel, grosimile de linii, invizibilitatea liniilor, ignorarea unora dintre ele, hașura secțiunii, culorile utilizate etc., sînt bine venite în amplificarea programului POLIGO (N), dacă se dorește o epură cu adevărat și artistică.


```

CALL PLOT(XP1=0.1,YP1,3)
CALL CIRCLE(XP1,YP1,360.)
CALL SYMBOL(XP1+0.2,YP1+0.2,0.4,S1,0.,2,1.,0.)
CALL PLOT(XP1,YP1,3)
CALL PLOT(XQ1,YQ1,2)
CALL PLOT(XP2=0.1,YP2,3)
CALL CIRCLE(XP2,YP2,360.)
CALL SYMBOL(XP2+0.2,YP2+0.2,0.4,S2,0.,2,1.,0.)
CALL PLOT(XP2,YP2,3)
CALL PLOT(XQ2,YQ2,2)
CALL PLOT(XP3=0.1,YP3,3)
CALL CIRCLE(XP3,YP3,360.)
CALL SYMBOL(XP3+0.2,YP3+0.2,0.4,S3,0.,2,1.,0.)
CALL PLOT(XP3,YP3,3)
CALL PLOT(XQ3,YQ3,2)
J=J+2

```

```

10 CONTINUE
XP1=-X1(N)
YP1=-Y1(N)
XP2=-X1(N)
YP2=Z1(N)
XP3=Y1(N)
YP3=Z1(N)
XQ1=-X1(1)
YQ1=-Y1(1)
XQ2=-X1(1)
YQ2=Z1(1)
XQ3=Y1(1)
YQ3=Z1(1)
S1(1)=V1(J)
S1(2)=V1(J+1)
S2(1)=V2(J)
S2(2)=V2(J+1)
S3(1)=V3(J)
S3(2)=V3(J+1)
CALL PLOT(XP1=0.1,YP1,3)
CALL CIRCLE(XP1,YP1,360.)
CALL SYMBOL(XP1+0.2,YP1+0.2,0.4,S1,0.,2,1.,0.)
CALL PLOT(XP1,YP1,3)
CALL PLOT(XQ1,YQ1,2)
CALL PLOT(XP2=0.1,YP2,3)
CALL CIRCLE(XP2,YP2,360.)
CALL SYMBOL(XP2+0.2,YP2+0.2,0.4,S2,0.,2,1.,0.)
CALL PLOT(XP2,YP2,3)
CALL PLOT(XQ2,YQ2,2)
CALL PLOT(XP3=0.1,YP3,3)
CALL CIRCLE(XP3,YP3,360.)
CALL SYMBOL(XP3+0.2,YP3+0.2,0.4,S3,0.,2,1.,0.)
CALL PLOT(XP3,YP3,3)
CALL PLOT(XQ3,YQ3,2)
CALL PLOT(0.,0.,999)
STOP
END

```

3.2.5 Programul PIR(AMIDA)

Program principal pentru desen al piramidei în triplă proiecție ortogonală cu balustrări, notații, axe și cu (sau fără) urmele planului de secțiune.

```

DIMENSION X1(100),Y1(100),Z1(100)
LOGICAL*1 LF(32)
TYPE 1
1 FORMAT('I NUME FISIER DATE=? ',)
ACCEPT 2,10,(LF(I),I=1,10)
2 FORMAT(2,3PA1)
CALL ASSIGN(2,LF,10)
READ(2,*) N
DO 13 I=1,N
13 READ(2,*) X1(I),Y1(I),Z1(I)
   READ(2,*) T1,T2,T3,T4
   CALL CLOSE(2)
   CALL PLOTS(0,1,7)
   CALL PLOT(0.,-25.,3)
   CALL PLOT(0.,25.,2)
   CALL PLOT(-25.,0.,3)
   CALL PLOT(25.,0.,2)
   CALL PLOT(0.,T1,3)
   CALL PLOT(T2,0.,2)
   CALL PLOT(0.,T3,2)
   CALL PLOT(T4,0.,2)
   DO 250 I=2,N-1
     XC1=X1(I)
     YC1=Y1(I)
     ZC1=Z1(I)
     XC2=X1(I+1)
     YC2=Y1(I+1)
     ZC2=Z1(I+1)
     CALL DRIP(XC1,YC1,ZC1,XC2,YC2,ZC2,I)
250 CONTINUE
     XC1=X1(1)
     YC1=Y1(1)
     ZC1=Z1(1)
     XC2=X1(2)
     YC2=Y1(2)
     ZC2=Z1(2)
     CALL DRIP(XC1,YC1,ZC1,XC2,YC2,ZC2,I)
     DO 251 I=3,N
       XC1=X1(I)
       YC1=Y1(I)
       ZC1=Z1(I)
       XC2=X1(I)
       YC2=Y1(I)
       ZC2=Z1(I)
       CALL DRFP(XC1,YC1,ZC1,XC2,YC2,ZC2)
251 CONTINUE
   CALL PLOT(0.,0.,999)
STOP
END

```

3.2.6 Programul CUB (PRISMA)

Program principal pentru desen al prisme (cubului) în triplă proiecție ortogonală cu balustrări, notații, axe și cu (sau fără) urmele planului de secțiune.

```

      DIMENSION X1(100) ,Y1(100),Z1(100)
      LOGICAL*1 LF(32)
      TYPE 1
1  FORMAT(' NUME FINIER DATE='F ',#)
      ACCEPT 2,IQ,(LF(I),I=1,IQ),
2  FORMAT(Q,32A1)
      CALL ASSIGN(2,LF,IQ)
      READ(2,*) N
      DO 13 I=1,N
13  READ(2,*) X1(I),Y1(I),Z1(I)
      CALL CLOSE(2)
      CALL PLOTS(0,1,7)
      CALL PLOT(0.,-25.,3)
      CALL PLOT(0.,25.,2)
      CALL PLOT(-25.,0.,3)
      CALL PLOT(25.,0.,2)
      DO 250 I=1,N/2-1
      XC1=X1(I)
      YC1=Y1(I)
      ZC1=Z1(I)
      XC2=X1(I+1)
      YC2=Y1(I+1)
      ZC2=Z1(I+1)
      CALL DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2,I)
250  CONTINUE
      XC1=X1(N/2)
      YC1=Y1(N/2)
      ZC1=Z1(N/2)
      XC2=X1(1)
      YC2=Y1(1)
      ZC2=Z1(1)
      CALL DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2,I)
      DO 251 I=N/2+1,N-1
      XC1=X1(I)
      YC1=Y1(I)
      ZC1=Z1(I)
      XC2=X1(I+1)
      YC2=Y1(I+1)
      ZC2=Z1(I+1)
      CALL DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2,I)
251  CONTINUE
      XC1=X1(N)
      YC1=Y1(N)
      ZC1=Z1(N)
      XC2=X1(N/2+1)
      YC2=Y1(N/2+1)
      ZC2=Z1(N/2+1)
      CALL DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2,I)
      DO 252 I=1,N/2
      XC1=X1(I)
      YC1=Y1(I)
      ZC1=Z1(I)
      XC2=X1(N/2+I)
      YC2=Y1(N/2+I)
      ZC2=Z1(N/2+I)
      CALL DRFF(XC1,YC1,ZC1,XC2,YC2,ZC2)
252  CONTINUE
      CALL PLOT(0.,0.,999)
      STOP
      END

```

3.3. Aplicații la secțiuni plane în poliedre

3.3.1 Aplicația 1

Secțiunea plană în piramida concavă octogonală efectuată cu un plan oarecare care nu întâlnește baza piramidei

Exemplul a fost astfel ales încît piramida să fie concavă și să prezinte cît mai multe muchii (laterale și ale poligonului de bază) în poziții particulare în raport cu planele de proiecție (fig. 3.1). Astfel dintre orizontalele poligonului de bază 4 sînt fronto-orizontale, 2 drepte de capăt și 2 orizontale oarecare. Dintre muchiile laterale ale piramidei 3 sînt frontale, una este de profil, una este verticală, iar 3 sînt drepte oarecare.

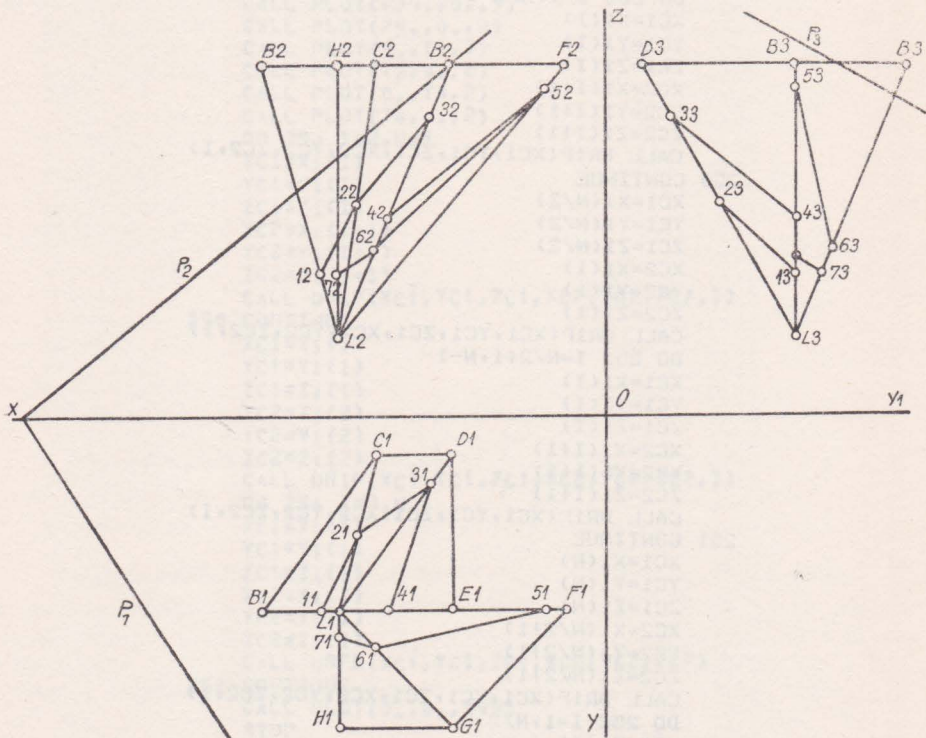


Fig. 3.1

Datele numerice pentru desenul automat al piramidei și datele rezultate pentru poligonul de secțiune sînt următoarele:

DESEN PIRAMIDA:

Date inițiale pentru piramida octogonală și pentru urmele planului de secțiune

9

7., 5., 2. Vîrfurile piramidei

9., 5., 9.

6., 1., 9.

4., 1., 9.

4., 5., 9.

1., 5., 9.

4., 8., 9.

7., 8., 9.

7., 5., 9.

-23., -15.34, 12.26, 23.

Planul de secțiune

(T₁, T₂, T₃, T₄)

Am utilizat planul de secțiune

$$12x + 8y + 15z - 184 = 0$$

cu tăieturile pe axele triedrului ortogonal (în epură):

$$P_x = S_1 (-15.34, 0, 0) = T_2 \text{ pentru plotare; } P_y = S_2 (0, -23, 0) = T_1$$

$$P_z = S_3 (0, 0, 12.26) = T_3 \quad P_{y1} = S_4 (23, 0, 0) = T_4$$

Pentru desenul automat al piramidei în triplă proiecție ortogonală am folosit Programul **SECPOS** (3.1.2), programul (3.2.5) **PIR(AMIDA)** iar pentru poligonul de secțiune am folosit programul 3.2.4 **POLIGO(N)**.

3.3.2 Aplicația 2

Secțiunea plană în piramida concavă octogonală efectuată cu un plan oarecare, care intersectează și baza piramidei.

Alegem aceeași piramidă din exemplul precedent (fig. 3.2). Vom modifica planul de secțiune lăsînd neschimbată urma orizontală a planului. În aceste condiții tăieturile planului pe axele triedrului ortogonal (în epură) vor fi:

$$P_x = S_1 (-15.34, 0, 0) = T_2 \text{ pentru plottare; } P_y = S_2 (0, -23, 0) = T_1;$$

$$P_z = S_3 (0, 0, 14.7) \text{ (modificat)} = T_3; \quad P_{y1} = S_4 (0, 23, 0) = T_4$$

Planul de secțiune cu aceste tăieturi va avea ecuația:

$$338.10x + 225.49y + 352.82z - 5186.45 = 0$$

Epura secțiunii plane automate în această piramidă în care este întilnită și baza piramidei este ilustrată în figura 3.2.

SECȚIUNEA REZULTATA:

Date rezultate pentru poligonul de secțiune în piramida cu planul care nu intersectează baza

8

7.46, 5., 3.62

6.5, 3.03, 5.44

4.56, 1.75, 7.67

5.69, 5., 5.04

1.54, 5., 8.36

6.03, 5.96, 4.25

7., 5.69, 3.62

7., 5., 4.

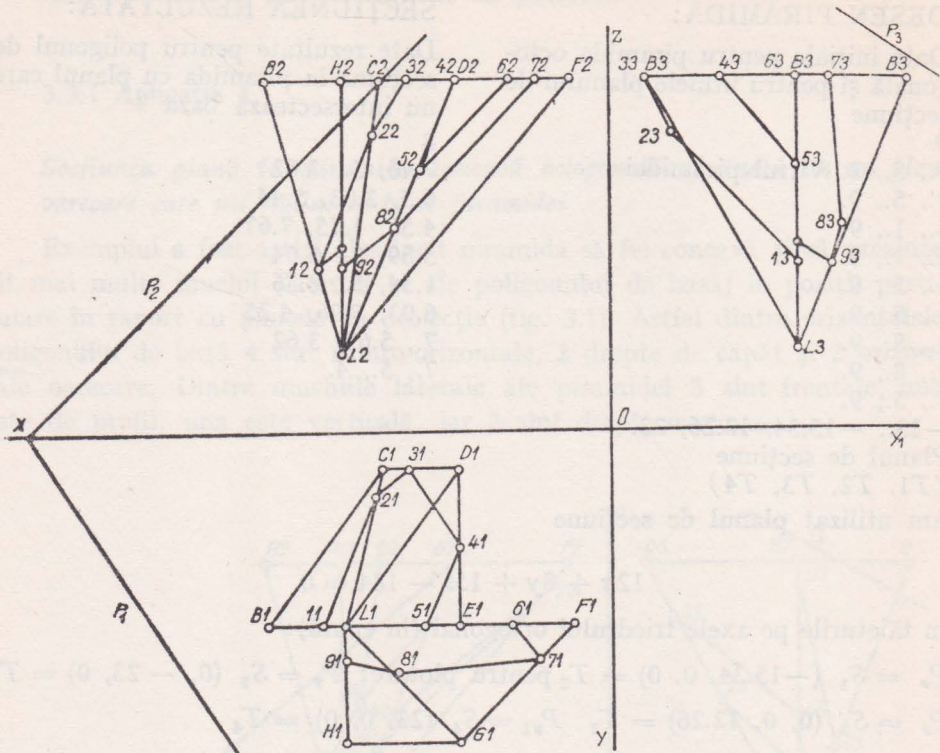


Fig. 3.2

3.3.3 Aplicația 3

Secțiunea plană în prisma oblică concavă octogonală efectuată cu un plan oarecare care nu intersectează baza prisme.

Vom folosi prisma oblică concavă octogonală (fig. 3.3) cu același poligon de bază ca formă și mărime și poziție ca în exemplele precedente la cotele +2, respectiv +9, parametrii directori ai muchiilor rezultând

$$ALD = 16 - 9 = 7; \quad AMD = 7.4 - 5 = 2.4; \quad AND = 9 - 2 = 7$$

Planul de secțiune este și el același din (3.3.1), deci are ecuația:

$$12x + 8y + 15z - 184 = 0$$

cu tăieturile pe axele triedrului ortogonal (în epură)

$$P_x = S_1(-15.33, 0, 0) = T_2 \text{ pentru plotare}$$

$$P_y = S_2(0, -23, 0) = T_1$$

$$P_z = S_3(0, 0, 12.26) = T_3$$

$$P_{y1} = S_4(23, 0, 0) = T_4$$

Datele numerice pentru desenul automat al prisme și datele rezultate pentru poligonul de secțiune sînt următoarele:

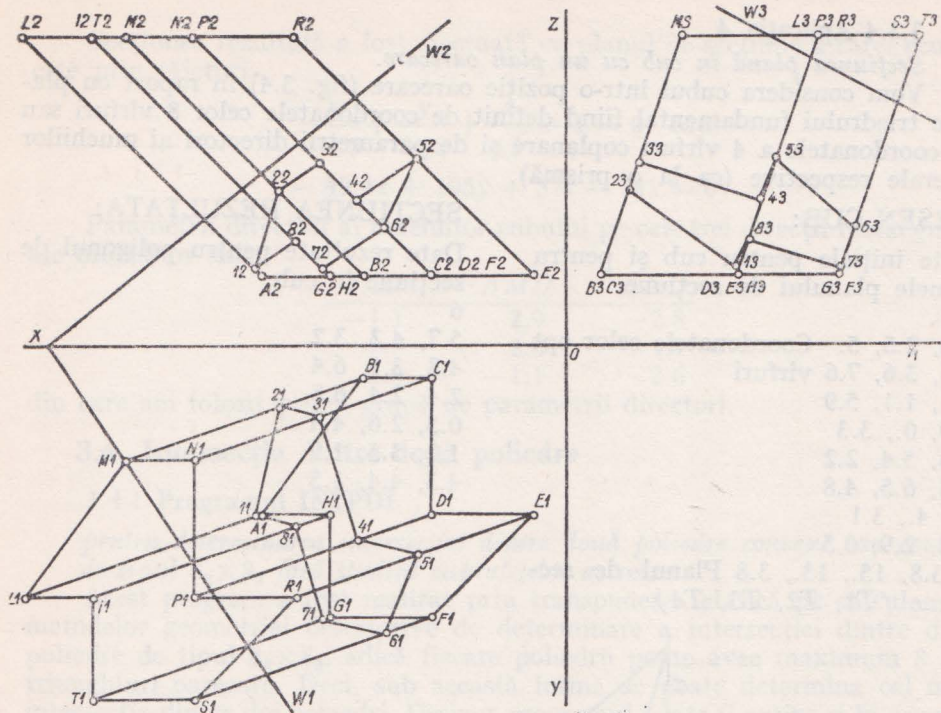


Fig. 3.3

DESEN PRISMA:

Date inițiale pentru prisma octogonală și pentru urmele planului de secțiune.

16

9., 5., 2. Baza inferioară

6., 1., 2

4., 1., 2.

4., 5., 2.

1., 5., 2.

4., 8., 2.

7., 8., 2.

7., 5., 2.

16., 7.4, 9. Baza superioară

13., 3.4, 9.

11., 3.4, 9.

11., 7.4, 9.

8., 7.4, 9.

11., 10.4, 9.

14., 10.4, 9.

14., 7.4, 9.

-23., -15.33, 12.26, 23.

Planul de secțiune
(T_1, T_2, T_3, T_4)**SECȚIUNEA REZULTATA:**

Date rezultate pentru poligonul de secțiune în prismă cu planul care nu intersectează baza.

8

9.2, 5., 2.2

8.48, 1.85, 4.48

7.29, 2.13, 5.29

6.2, 5.7, 4.2

4.42, 6.17, 5.42

5.4, 8.48, 3.4

7.2, 8.06, 2.2

8., 5.34, 3.

Pentru desenul automat al prisme în triplă proiecție ortogonală am folosit programul **SECPOS** (3.1.2), programul (3.2.6) **CUB (PRISMA)** iar pentru poligonul de secțiune am folosit programul (3.2.4) **POLIGO(N)**. Epura secțiunii plane automate în prismă este ilustrată în fig. 3.3.

3.3.4 Aplicația 4

Secțiunea plană în cub cu un plan oarecare.

Vom considera cubul într-o poziție oarecare (fig. 3.4) în raport cu planele triedrului fundamental fiind definit de coordonatele celor 8 vîrfuri sau de coordonatele a 4 vîrfuri coplanare și de parametrii directori ai muchiilor laterale respective (ca la o prismă).

DESEN CUB:

Date inițiale pentru cub și pentru urmele planului de secțiune.

8

6.4, 2.5, 5. Coordonatele celor opt

3.6, 3.6, 7.6 vîrfuri

1.1, 1.1, 5.9

3.9, 0., 3.3

5.3, 5.4, 2.2

2.5, 6.5, 4.8

0., 4., 3.1

2.8, 2.9, 0.5

—3.8, 15., 13., 3.8 Planul de secțiune (T_1, T_2, T_3, T_4)

SECȚIUNEA REZULTATA:

Date rezultate pentru poligonul de secțiune în cub.

6

5.7, 4.2, 3.2

4.8, 3.1, 6.4

2.4, 2.4, 6.8

0.5, 2.6, 4.4

1.2, 3.5, 1.9

4.3, 4.4, 1.5

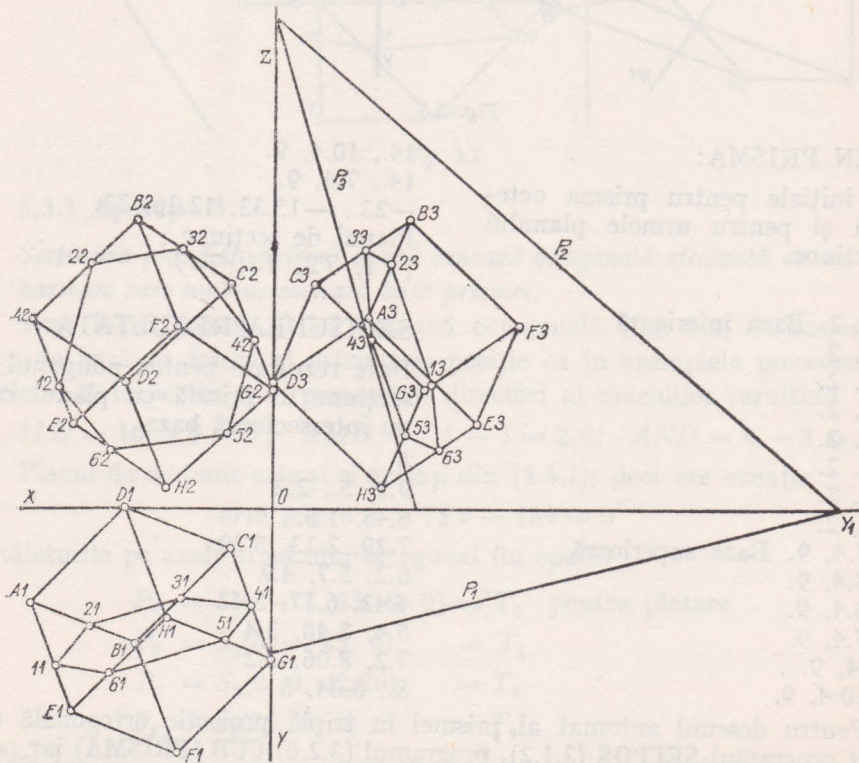


Fig. 3.4

Secțiunea rezultată a fost efectuată cu planul de secțiune a cărei ecuație este prin tăieturi:

$$\frac{x}{-15} + \frac{y}{3.8} + \frac{z}{13} - 1 = 0 \quad \text{sau}$$

$$-49.4x + 195y + 57z - 741 = 0$$

Parametrii directori ai muchiilor cubului pe cele trei direcții rectangulare ale muchiilor sînt:

<i>ALD</i>	<i>AMD</i>	<i>AND</i>
-1.1	2.9	-2.8
2.5	2.5	1.7
2.8	-1.1	2.6

din care am folosit prima grupă de parametrii directori.

3.4. Intersecția dintre două poliedre

3.4.1 Programul INTPOL

pentru determinarea intersecției dintre două poliedre convexe sau concave de tipul $8_3 \times 8_3$ fără studiul suprafețelor ascunse.

Acest program a fost realizat prin transpunerea clasică pe calculator a metodelor geometriei descriptive de determinare a intersecției dintre două poliedre de tipul $8_3 \times 8_3$, adică fiecare poliedru poate avea maximum 8 fețe triunghiuri oarecare. Deci, sub această formă se poate determina cel mult intersecția dintre doi octaedri. Desigur programul poate fi extins și în cazurile în care poliedrele conțin mai mult de opt fețe fiecare și în care laturile poligoanelor fețe sînt în număr de $n > 3$.

Extinderea studiului acestui program **INTPOL** poate cuprinde și elaborarea unui algoritm diferit de subprogramul **VIZLIN** sau adaptarea lui pentru determinarea suprafețelor ascunse ale poliedrelor.

În acest program **INTPOL** determinarea punctelor de intersecție dintre muchiile unui poliedru și fețele celui alt poliedru se realizează cu ajutorul subprogramului **PTINT**, iar unirea punctelor de intersecție, deci poligonul strămb de intersecție dintre cele două poliedre, se obține prin subprogramul **UPIP** (Unirea Punctelor din Intersecția Poliedrelor).

3.4.2 Subprogramul PTINT

Acest subprogram determină punctele de intersecție utile dintre muchiile unui poliedru și fețele triunghiulare ale celui alt poliedru. Subprogramul este original și diferă de metodele clasice prin felul în care se testează apartenența unui punct de intersecție în interiorul conturului triunghiular al feței poliedrului. (O generalizare a subprogramului **PTINT** va fi prezentată prin subprogramul **POLIG** pentru $n > 3$).

Prin urmare acest subprogram **PTINT** stabilește dacă punctul de intersecție dintre o muchie limitată de două vîrfuri a unui poliedru se găsește în interiorul feței limitate de laturi a celui alt poliedru. De exemplu, dacă punctul de intersecție I al muchiei M_1M_2 se găsește în planul triunghiului față a celui de al doilea poliedru, deci în interiorul conturului $u_1u_2u_3$. (fig. 3.5). Cu alte cuvinte acest subprogram selectează punctele utile rezultate din intersecția celor două poliedre, puncte care alcătuiesc în final poligonul strămb de intersecție.

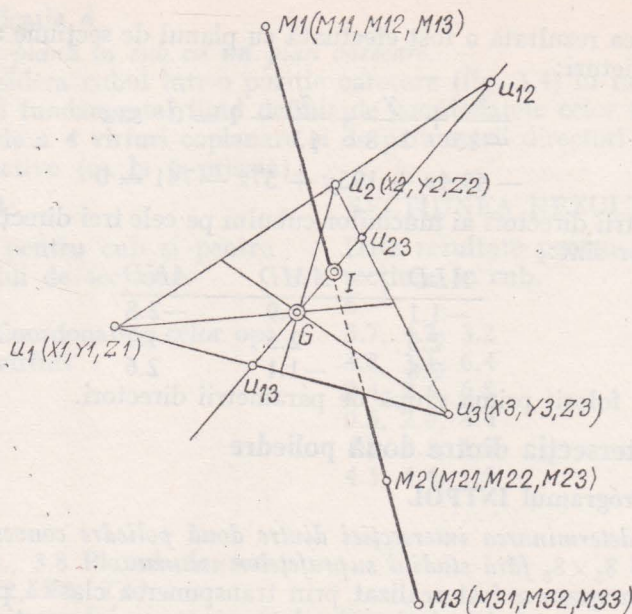


Fig. 3.5

Acest lucru este necesar deoarece în intersecția dintre două poliedre apar muchii care intersecțiază fețele celuiilalt poliedru numai prin prelungirea dreptei suport ce conține muchia limitată de două vîrfuri a poliedrului respectiv.

Mai departe este necesar ca punctul de intersecție I al acestei muchii M_1M_2 cu fața respectivă $u_1u_2u_3$ a celuiilalt poliedru să fie inclus în interiorul feței limitată de laturi și nu numai în planul ei, deci în afara conturului feței respective.

Așa dar acest subprogram **PCTINT** realizează un test original de apartenență al unui punct de pe o dreaptă arbitrară M_1M_3 , interiorului unei figuri plane.

Dintre cele mai cunoscute teste de acest fel amintim testul în care se calculează suma unghiurilor făcute de dreptele duse din punctul respectiv (interior) spre colțurile (vîrfurile) figurii — sumă care trebuie să fie egală cu 360° ca punctul I să aparțină planului figurii —, și testul în care dreapta dusă arbitrar într-un sens din punctul I întîlnește un număr impar de puncte la intersecțiile cu laturile figurii, dacă punctul testat I este situat în planul și interiorul figurii.

Aceste teste sînt utilizate în cadrul altor programe de intersecție a poliedrelor incluse în această lucrare.

În cadrul subprogramului **PCTINT** punctele de intersecție I se calculează cu subprogramul **INDRPL** (Intersecția dreaptă-plan) apelată de cîte ori este necesar.

Dacă punctul I îndeplinește condiția $M_1 < I < M_2$ testul continuă și se calculează — pentru comoditate — centrul de greutate G al triunghiului $u_1u_2u_3$ (În locul lui G poate fi luat oricare alt punct interior triunghiului).

Se intersectează apoi dreapta IG cu laturile triunghiului $u_1u_2u_3$ obținându-se punctele de intersecție u_{12} , u_{23} și u_{13} pe laturi sau pe prelungirile lor.

Se testează dacă aceste puncte îndeplinesc condițiile de situare între vîrfurile triunghiului, deci dacă

$$u_1 < u_{12} < u_2$$

$$u_2 < u_{23} < u_3$$

$$u_1 < u_{13} < u_3$$

În cazul figurii este eliminat punctul u_{12} care este situat pe prelungirea laturii u_1u_2 .

Se rețin punctele u_{13} și u_{23} și se testează dacă punctul I aparține interiorului segmentului u_{13} și u_{23} deci dacă

$$u_{13} < I < u_{23}$$

Dacă da, punctul I este bun și va fi luat în considerare ca punct util în intersecția dintre cele două poliedre.

Dacă în caz particular dreapta IG este paralelă cu una dintre laturile triunghiului, se testează numai punctele de intersecție situate pe laturile triunghiului la distanță finită.

În cazul figurii 3.5, punctul I rezultat din intersecția dreptei M_2M_3 cu planul $u_1u_2u_3$ nu este util deoarece nu este îndeplinită prima condiție, adică I nu este cuprins între punctele M_2M_3 care definesc o muchie a unui poliedru.

În afară de subprogramul **INDRPL** se mai folosesc subprogramele **CLASPL** și **LIMITE**.

Primul subprogram realizează o clasificare a planului respectiv în raport cu triedrul de referință $OXYZ$ iar al doilea subprogram realizează efectiv teste de incluziune enumerate mai sus.

Instrucțiunea de apel a subprogramului **PCTINT** este:

CALL PCTINT (X, Y, Z, N1, N2, V3, XT, YT, ZT, ID)

unde semnificația parametrilor este următoarea:

- X, Y, Z — Coordonatele punctelor care definesc cele două poliedre (vectori cu N componente fiecare).
- $N1, N2$ — Indicii care definesc capetele unei muchii $X(N1)$, $Y(N1)$, $Z(N1)$ este prima extremitate. $X(N2)$, $Y(N2)$, $Z(N2)$ este a doua extremitate a muchiei din primul poliedru.
- $V3$ — Vector format din linia care cuprinde definiția unei fețe a celui de al doilea poliedru plus parametrii directori ai planului acestei fețe.
- XT, YT, ZT — Coordonatele punctului prezumptiv al poligonului de intersecție.
- ID — Constantă care indică dacă punctul XT, YT, ZT este interior feței testate ($ID = 1$), deci dacă este punct util al poligonului de intersecție. Dacă nu $ID = 0$

Instrucțiunea de apel a subprogramului **CLASPL**

CALL CLASPL (A, B, C)

se referă la planul feței poliedrului care poate fi în funcție de parametrii directori A, B, C plan oarecare, vertical, de capăt, de profil, paralel cu axa OX sau plan de nivel.

Instrucțiunea de apel a subprogramului **LIMITE** este:

CALL LIMITE(X1, Y1, Z1, X2, Y2, Z2, XC, YC, ZC, XT, YT, ZT, IND) și realizează testele de incluziune cu semnificația parametrilor rezultată din analiza figurii 3.5 adică testează dacă punctul XC, YC, ZC este situat între punctele $X1, Y1, Z1 - X2, Y2, Z2$.

Semnificația parametrilor este următoarea:

- $X1, Y1, Z1$ — Coordonatele extremităților laturei $u_i u_j (i, j = 1, 3)$ a triunghiului testat sau extremitățile segmentelor de forma (de exemplu) $u_{13} u_{23}$.
- $X2, Y2, Z2$ — Coordonatele punctelor testate care pot fi de tipul u_{ij} sau I .
- XC, YC, ZC — Coordonatele XC, YC, ZC în ipoteza că îndeplinesc testul de incluziune.
- XT, YT, ZT — Coordonatele XC, YC, ZC în ipoteza că îndeplinesc testul de incluziune.
- IND — Constantă care indică dacă punctul XT, YT, ZT este interior segmentului testat ($IND = 1$)
Dacă nu $IND = 0$.

Listările subprogramelor **PCTINT**, **CLASPL** și **LIMITE** pot fi urmărite în cadrul programului **INTPOL**.

3.4.3. Subprogramul UPIP

Acest subprogram este elaborat pentru stabilirea ordinii de unire a punctelor rezultate din intersecția a două poliedre convexe sau concave de tipul $8_3 \times 8_3$.

Subprogramul este fundamentat pe transpunerea pe calculator a matricelor incluse în tabelul comunității de apartenență a punctelor de intersecție la fețele poliedrelor.

În acest subprogram sînt admise nu numai puncte de intersecție rezultate din intersecția a trei plane. Așa dar, poliedrele se pot intersecta în vîrfuri sau pe muchii.

De asemenea, fiecare poliedru trebuie să fie convex sau concav și, să posede maximum 8 fețe laterale care pot fi poligoane cu $n > 3$ laturi. Numărul de puncte de intersecție admise este 30 matricile avînd dimensiunea $8/30$ (8 coloane corespunzător planelor fețelor pentru fiecare poliedru și 30 de linii corespunzătoare punctelor de intersecție luate în orice ordine. Numerotarea, deci definirea unui punct de intersecție este dată de linia în care se află. Apartenența punctelor la diferitele plane ale fețelor se marchează în matricile respective prin cifra 1, iar neapartența prin zero, în felul următor:

Poliedrul I: Matricea A

	Fața 1	Fața 2	Fața 3	Fața 8
Primul punct de intersecție	1	1	0 0 0 0 0	0	

Poliedrul II: Matricea B

	Fața 1	Fața 2	Fața 3	Fața 8
Primul punct de intersecție	0	1	0	0	0	0

PROGRAM UPIP

```

C      *****
C      PROGRAM PENTRU UNIREA PUNCTELOR
C      REZULTATE DIN INTERSECȚIA A DOUA POLIEDRE
C      *****
C      INTEGER A(30,8),B(30,8),C(30),D(30),Q
10     FORMAT (8I5)
11     FORMAT (' SE VOR UNI URMATOARELE PUNCTE: ')
12     FORMAT (31X,' PUNCTUL ',I5,' CU ', ' PUNCTUL ',I5)
13     FORMAT (' MATRICEA PUNCTELOR DE INTERSECȚIE ')
14     FORMAT (' PENTRU POLIEDRUL A ESTE ')
15     FORMAT (' PENTRU POLIEDRUL B ESTE ')
16     FORMAT (' ***** ')
17     FORMAT (' ')
18     FORMAT (' DETERMINAREA ORDINII DE UNIRE ')
19     FORMAT (' PENTRU PUNCTELE DE INTERSECȚIE ')
20     FORMAT (I2)
21     FORMAT (' A DOUA POLIEDRE A SI B ')
30     READ (105,20) K
      M=1
40     DO 50 M=1,K
C/    WRITE (108,20) M
50     READ (105,10) (A(M,J),J=1,8)
60     READ (105,20) K
70     DO 80 M=1,K
C/    WRITE (108,20) M
80     READ (105,10) (B(M,J),J=1,8)
82     FORMAT (8(2X,I5))
C/ 83     WRITE (108,82) ((A(J,M),J=1,8),M=1,12)
      WRITE (108,17)
      WRITE (108,13)
      WRITE (108,14)
      WRITE (108,16)
      WRITE (108,17)
      WRITE (108,82) ((A(M,J),J=1,8),M=1,K)
      WRITE (108,17)
      WRITE (108,13)
      WRITE (108,15)
      WRITE (108,16)
      WRITE (108,17)
      WRITE (108,82) ((B(M,J),J=1,8),M=1,K)
C/ 84     WRITE (108,82) ((B(M,J),J=1,12),M=1,8)
C/    WRITE (108,82) ((A(M,J),J=1,12),M=1,8)
90     J=0
C/    WRITE (108,20) M
100    L=K
      K=1
      P=0

```

```

110 CONTINUE
    J=J+1
    IF (J.GE.L) GO TO 320
120 CONTINUE
    I=1
130 CONTINUE
    M=1
    N=J
C/   WRITE (108,20) M
C/   WRITE (108,20) I
140 GO TO 160
150 IF (I.GE.8) GO TO 110
C/   WRITE (108,20) I
    I=I+1
160 IF (A(J,I).NE.1) GO TO 150
170 GO TO 220
180 IF (P-1)200,190,200
190 P=0
    GO TO 110
    I=I+1
C/   WRITE (108,20) I
    P=1
    GO TO 130
210 M=M+1
    IF (M.GT.8) GO TO 180
C/   WRITE (108,20) M
220 IF (B(N,M).NE.1) GO TO 210
    C(K)=N
230 GO TO 250
240 IF (N.GE.L) GO TO 270
250 N=N+1
    IF (A(N,I).NE.1.OR.B(N,M).NE.1) GO TO 240
    D(K)=M
    K=K+1
260 GO TO 280
270 N=J
C/   WRITE (108,20) M
    GO TO 210
280 IF (M.LE.8) GO TO 270
290 GO TO 310
300 I=I+1
    P=1
    GO TO 120
310 IF (P.NE.1) GO TO 300
    P=0
C/   WRITE (108,20) M
320 IF (J.LT.L) GO TO 110

C/321 WRITE (108,322) (C(I),I=1,30)
C/322 FORMAT (5X,15(2X,I5))
C/323 WRITE (108,322) (D(I),I=1,30)
    I=1
    WRITE (108,17)
    WRITE (108,17)
    WRITE (108,18)
    WRITE (108,19)
    WRITE (108,21)
    WRITE (108,16)
    WRITE (108,17)
    WRITE (108,11)
    G=K-1
    DO 400 I=1,G
    WRITE (108,12) C(I),D(I)
400 CONTINUE
500 STOP
    END

```

Ordinea de unire a punctelor este dată de algoritm și afișată în rezultat. Dacă în intersecția dintre două poliedre apare evident apriori faptul că unele fețe nu pot lua parte la intersecție este indicat pentru simplificarea calculului să nu mai fie trecute în matrice aceste fețe. Problema liniilor ascunse nu este inclusă ca rezolvare în acest algoritm, deși din tabelul comunității de apartenență poate fi dedusă vizibilitatea sau invizibilitatea laturilor poligonului strîmb de intersecție dintre cele două poliedre.

3.4.4. Aplicație la programul UPIP

Să considerăm intersecția dintre doi tetraedri regulați $ABCD$ și $EFGI$ definiți prin următoarele coordonate:

$$\begin{aligned} A(13.9, 7.2, 4) & \quad E(9.7, 7.2, 0) \\ B(6, 11.7, 4) & \quad F(8.3, 11.7, 7.8) \\ C(6, 2.7, 4) & \quad G(8.3, 2.7, 7.8) \\ D(8.5, 7.2, 11.4) & \quad I(1.4, 7.2, 3.8) \end{aligned}$$

Cele 12 puncte de intersecție care vor fi obținute vor avea următoarele coordonate și vor aparține fiecarei celor trei plane de intersecție corespunzătoare;

PUNCT DE INTERSECȚIE	PLANE / FEȚE /
1 (9 ; 9.5 ; 4)	1 = $ABC \cap EFG \cap IEF$
2 (9 ; 4.9 ; 4)	2 = $ABC \cap EFG \cap IEG$
3 (8.9 ; 9.8 ; 4.7)	3 = $DAB \cap EFG \cap IEF$
4 (8.9 ; 4.6 ; 4.7)	4 = $DCA \cap EFG \cap IEG$
5 (8.3 ; 8.9 ; 7.8)	5 = $DAB \cap EFG \cap IFG$
6 (8.3 ; 5.5 ; 7.8)	6 = $DCA \cap EFG \cap IFG$
7 (7.1 ; 9.7 ; 7.1)	7 = $DBC \cap DAB \cap IFG$
8 (7.1 ; 4.7 ; 7.1)	8 = $DBC \cap DCA \cap IFG$
9 (6.8 ; 10.4 ; 6.2)	9 = $DAB \cap DBC \cap IEF$
10 (6.8 ; 4 ; 6.2)	10 = $DBC \cap DAC \cap IEF$
11 (6 ; 8.6 ; 4)	11 = $ABC \cap DBC \cap IEF$
12 (6 ; 5.7 ; 4)	12 = $ABC \cap DBC \cap IEG$

Cele două tabele ale comunității de apartenență ale punctelor de intersecție pentru fiecare poliedru în parte vor avea matricele următoare (4×12) care vor fi unificate pe matrice de dimensiunea 8×12 :

Fața								
Punct de intersecție	DCA	DBC	DAB	ABC	IEG	IFG	IEF	EFG
1				1			1	1
2				2	2			2
3			3				3	3
4	4				4			4
5			5			5		5
6	6					6		6
7		7	7			7		
8	8	8				8		
9		9	9				9	
10	10	10			10			
11		11		11			11	
12		12		12	12			

sau:

1	0	0	0	1	0	0	1	1
2	0	0	0	1	1	0	0	1
3	0	0	1	0	0	0	1	1
4	1	0	0	0	1	0	0	1
5	0	0	1	0	0	1	0	1
6	1	0	0	0	0	1	0	1
7	0	1	1	0	0	1	0	0
8	1	1	0	0	0	1	0	0
9	0	1	1	0	0	0	1	0
10	1	1	0	0	1	0	0	0
11	0	1	0	1	0	0	1	0
12	0	1	0	1	1	0	0	0

MATRICEA PUNCTELOR DE INTERSECȚIE PENTRU POLIEDRUL A ESTE

```

0 0 0 1 0 0 0 0 0
0 0 0 1 0 0 0 0 0
0 0 1 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0
0 1 1 0 0 0 0 0 0
1 1 0 0 0 0 0 0 0
1 1 0 0 0 0 0 0 0
0 1 0 1 0 0 0 0 0
0 1 0 1 0 0 0 0 0

```

MATRICEA PUNCTELOR DE INTERSECȚIE PENTRU POLIEDRUL B ESTE

```

0 0 1 1 0 0 0 0 0
1 0 0 1 0 0 0 0 0
0 0 1 1 0 0 0 0 0
1 0 0 1 0 0 0 0 0
0 1 0 1 0 0 0 0 0
0 1 0 1 0 0 0 0 0
0 1 0 1 0 0 0 0 0
0 1 0 1 0 0 0 0 0
0 1 0 1 0 0 0 0 0
1 0 0 1 0 0 0 0 0
0 0 1 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0

```

DETERMINAREA ORDINII DE UNIRE PENTRU PUNCTELE DE INTERSECȚIE A DOUĂ POLIEDRE A ȘI B

SE VOR UNI URMĂTOARELE PUNCTE:

PUNCTUL 1 CU PUNCTUL 11	PUNCTUL 4 CU PUNCTUL 6
PUNCTUL 1 CU PUNCTUL 2	PUNCTUL 5 CU PUNCTUL 7
PUNCTUL 2 CU PUNCTUL 12	PUNCTUL 6 CU PUNCTUL 8
PUNCTUL 3 CU PUNCTUL 9	PUNCTUL 7 CU PUNCTUL 8
PUNCTUL 3 CU PUNCTUL 5	PUNCTUL 9 CU PUNCTUL 11
PUNCTUL 4 CU PUNCTUL 10	PUNCTUL 10 CU PUNCTUL 12

Observație importantă: Notarea punctelor de intersecție dela 1 la 12 reprezintă ordinea în care au fost determinate punctele de intersecție prin program. Pentru unirea cursivă este necesară ordonarea punctelor care este efectuată tot prin program. Această ordonare convine echipa mentului de desen automat.

3.4.5. Descrierea programului INTPOL

În acest program se determină așa dar coordonatele și unirea vîrfurilor poligonului de intersecție dintre două poliedre date. Cele două poliedre constituie datele de intrare care sînt organizate în felul următor:

- N — numărul de vîrfuri ale celor două poliedre.
 $(X(I), Y(I), Z(I), I = 1, N)$ — coordonatele celor N vîrfuri.
 M_1, M_2 — numărul de fețe ale primului poliedru, respectiv ale celui de al doilea poliedru.
 $C(I, J)_{\substack{J=1,4 \\ I=1, M_1}}$ — matrice care cuprinde pe fiecare linie un număr de ordine pe coloana 1 iar pe celelalte 3 coloane indicii ($I = 1, \dots, N$) care stabilesc coordonatele punctelor care formează o față a primului poliedru.
 $H(I, J)_{\substack{J=1,4 \\ I=1, M_2}}$ — matricea analogă cu matricea $C(I, J)$, dar pentru al doilea poliedru.

Din matricea $C(I, J)_{\substack{J=1,4 \\ I=1, M_1}}$ se formează o nouă matrice $E(I, J)_{\substack{J=1,8 \\ I=1, M_1}}$, care cuprinde exact elementele matricei $C(I, J)$ completate pe fiecare linie cu încă patru elemente A, B, C, D care reprezintă parametrii directori ai planului determinat de față dată de punctele stabilite de indicii de pe coloanele 2, 3, 4 ale aceleiași linii,

Analog se formează matricea $F(I, J)_{\substack{J=1,8 \\ I=1, M_2}}$ din matricea $H(I, J)_{\substack{J=1,4 \\ I=1, M_2}}$

Cu aceste matrici se va lucra mai departe în felul următor: se caută în prima matrice $E(I, J)$ elementele egale dintre două linii, linia $V1$, și linia $V2$ și se păstrează aceste elemente, ele fiind chiar indicii care stabilesc coordonatele extremităților unei muchii a primului poliedru. Această muchie se va testa dacă se intersectează sau nu, cu o față a celui de al doilea poliedru, față definită prin cei 3 indici din fiecare linie a celei de a doua matrice $F(I, J)$, linie notată $V3$.

Intersecția dintre muchie și față se rezolvă în subrutina **PCTINT**, subrutină din care rezultă coordonatele punctului de intersecție, dacă punctul de intersecție dintre muchie și planul feței testate este interior feței, iar dacă nu, se atribuie unei constante ID , valoarea $ID = 0$.

După ce se iau în considerație toate muchiile primului poliedru, rezultate din matricea $F(I, J)_{\substack{J=1,8 \\ I=1, M_1}}$ și toate fețele celui de al doilea poliedru rezultate din toate liniile matricei $F(I, J)_{\substack{J=1,8 \\ I=1, M_2}}$, procedeul se inversează luîndu-se în locul matricei $F(I, J)_{\substack{J=1,8 \\ I=1, M_1}}$ matricea $F(I, J)_{\substack{J=1,8 \\ I=1, M_2}}$ și invers, obținîndu-se acum toate punctele utile de intersecție dintre muchiile celui de al doilea poliedru și fețele primului poliedru.

Toate aceste puncte se păstrează într-un vector prin coordonatele lor care se afișează ca rezultat, acesta fiind un prim obiectiv al programului **INTPOL**.

Un al doilea obiectiv este determinarea ordinii de unire a acestor puncte pentru a defini poligonul ce reprezintă intersecția dintre cele două poliedre.

În vederea obținerii acestui rezultat se formează două matrici A și B , care au atîtea linii cîte puncte utile de intersecție există și atîtea coloane cîte fețe au cele două poliedre împreună. Elementele acestor matrici sînt 0 sau

1. Elementul egal cu 1 este cel care indică intersecția dintre o față a unui poliedru și o muchie a celuilalt poliedru. Elementul egal cu 1 se poziționează în coloana care reprezintă fața ce cuprinde punctul util de intersecție în matricea A (de exemplu) în același timp, în matricea B , 1 poziționându-se în ambele coloane care reprezintă fețele care au ca intersecție muchia ce intersectează fața de mai sus (din matricea A).

Urmează apoi un raționament care determină ordinea de unire a punctelor inclus în subprogramul **UPIP**, așa cum s-a arătat la 3.4.3.

Datele de ieșire ale acestui program sînt:

- (1) — $A(I, J), B(I, J)$ — matricele care indică prin poziția elementului egal cu 1 poziția fiecărui punct de intersecție în fețele poliedrelor.
- (2) — Mesaje care ne indică ordinea de unire a celor NR puncte de intersecție.
- (3) — $XI(I), YI(I), ZI(I), I = 1, NR$ = Coordonatele celor NR puncte de intersecție, în ordinea unirii lor.

Coordonatele obținute la punctul (3) formează un fișier de date ordonate pentru un program de desen, care împreună cu programele de desen ale poliedrelor studiate dau o imagine finală a problemei de intersecție dintre cele două poliedre.

3.4.6 Program principal INTPOL

```

PROGRAM INTPOL PENTRU DETERMINAREA INTERSECȚIEI DINTRE DOUA
POLIEDRE CONVEXE A SI B CU MAXIMUM OPT FETE PENTRU FIECARE
POLIEDRU
COORDONATELE VIRFURILOR SINT X(I),Y(I),Z(I) NUMEROTATE
CRONOLOGIC PENTRU FIECARE POLIEDRU (IN CONTINUARE)
NUMARUL DE FETE LA POLIEDRUL A ESTE M1
NUMARUL DE FETE LA POLIEDRUL B ESTE M2
DEFINIREA FETELOR POLIEDRELOR SE FACE CRONOLOGIC SEPARAT
PENTRU FIECARE POLIEDRU INCEPIND CU CIFRA 1 PINA LA M1
SI 1 PINA LA M2
PUNCTELE DE INTERSECȚIE SE DETERMINA CU SUBPROGRAMUL PCTINT
UNIREA PUNCTELOR REZULTATE DIN INTERSECȚIE SE REALIZEAZA CU
SUBPROGRAMUL UPIP
DIMENSION D(30),G(30)
DIMENSION C(20,4),H(20,4),E(20,8),F(20,8),A(100,20),B(100,20),X(20
1),Y(20),Z(20)
INTEGER Q
INTEGER C,H,CONT,CONT1
READ(105,1) N
1 FORMAT(I4)
READ(105,2) (X(I),Y(I),Z(I),I=1,N)
2 FORMAT(3F10.3,50X)
READ(105,3) M1,M2
3 FORMAT(2I4)
READ(105,4) ((C(I,J),J=1,4),I=1,M1)
4 FORMAT(4(I7,3X),40X)
READ(105,5) ((H(I,J),J=1,4),I=1,M2)
5 FORMAT(4(I7,3X),40X)
DO 22 I=1,M1
WRITE(108,23) (C(I,J),J=1,4)
23 FORMAT(' ',4I2/)
22 CONTINUE
DO 24 I=1,M2
WRITE(108,25) (H(I,J),J=1,4)
25 FORMAT(' ',4I2/)
24 CONTINUE
DO 6 I=1,100
DO 7 J=1,20
A(I,J)=0
B(I,J)=0
7 CONTINUE
6 CONTINUE
DO 8 I=1, M1)
DO 9 J=1, 4
E(I, J)=C(I, J
9 CONTINUE
8 CONTINUE

```

```

DO 10 I=1,M2
DO 11 J=1,4
F(I,J)=H(I,J)
11 CONTINUE
10 CONTINUE
DO 12 I=1,M1
K1=C(I,2)
K2=C(I,3)
K3=C(I,4)
20 WRITE(108,20) K1,K2,K3
FORMAT(' ',3I5/)
CALL PARAM(X,Y,Z,K1,K2,K3,A1,B1,C1,D1)
E(I,5)=A1
E(I,6)=B1
E(I,7)=C1
E(I,8)=D1
12 CONTINUE
DO 13 I=1,M2
K4=H(I,2)
K5=H(I,3)
K6=H(I,4)
WRITE(108,20) K4,K5,K6
CALL PARAM(X,Y,Z,K4,K5,K6,A1,B1,C1,D1)
F(I,5)=A1
F(I,6)=B1
F(I,7)=C1
F(I,8)=D1

WRITE(108,21) A1,B1,C1,F(I,5),F(I,6),F(I,7)
13 FORMAT(' ',20X,6(F10.3,2X)/)
CONTINUE
CONT=0
DO 26 I=1,4
WRITE(108,27) (E(I,J),J=1,8)
27 FORMAT(' ',8F10.3/)
26 CONTINUE
NR=1
CALL CALCUL(E,F,X,Y,Z,CONT,NR,M1,M2,A,B)
NR=2
CALL CALCUL(F,E,X,Y,Z,CONT,NR,M1,M2,A,B)
M12=M1+M2
WRITE(108,28)
28 *UL A ESTE '/', ' ', 'PENTRU POLIEDR
*UL A ESTE /)
DO 15 I=1,CONT
WRITE(108,14) (A(I,J),J=1,M12)
14 FORMAT(' ',20F10.3/)
15 CONTINUE

WRITE(108,29)
29 *UL B ESTE '/', ' ', 'PENTRU POLIEDR
*UL B ESTE /)
DO 16 I=1,CONT
WRITE(108,17) (B(I,J),J=1,M12)
17 FORMAT(' ',20F10.3/)
16 CONTINUE
K=CONT
30 FORMAT(' ', 'SE VOR UNI URMATOARELE PUNCTE: '/')
31 FORMAT(31X, 'PUNCTUL ',I5, ' CU PUNCTUL ',I5/)
32 FORMAT(' ', '*****')
33 FORMAT(' ', '*****')
18 FORMAT(' ', 'DETERMINAREA ORDINII DE UNIRE '/')
19 FORMAT(' ', 'PENTRU PUNCTELE DE INTERSECȚIE '/')
1111 FORMAT(' ', 'A DOUA POLIEDRE A SI B '/')
J=0
L=K
100 K=1
P=0
110 CONTINUE
J=J+1
IF(J.GE.L) GO TO 320
120 CONTINUE
I=1
C/ WRITE(108,20) M
C/ WRITE(108,20) I
C/ WRITE(108,20) M
130 CONTINUE
M=1
N=J
140 GO TO 160
150 IF(I.GE.8) GO TO 110
C/ WRITE(108,20) I
I=I+1
160 IF(A(J,I).NE.1) GO TO 150
170 GO TO 220
180 IF(P-1) 200,190,200
190 P=0
GO TO 110
200 I=I+1
C/ WRITE(108,20) I
P=1
GO TO 130
210 M=M+1
IF(M.GT.8) GO TO 180
C WRITE(108,20) M
220 IF(B(N,M).NE.1) GO TO 210

```

```

G(K)=
230 GO TO 250
240 IF(N.GE.L) GO TO 270
250 N=N+1
    IF(A(N,I).NE.1.OR.B(N,M).NE.1) GO TO 240
    D(K)=N
    K=K+1
260 GO TO 280
270 N=J
    WRITE(108,20) M
    GO TO 210
280 IF(M.LE.8) GO TO 270
290 GO TO 310
300 I=I+1
    P=1
    GO TO 120
310 IF(P.NE.1) GO TO 300
    P=0
C/ WRITE(108,20) M
    IF(J.LT.L) GO TO 110
C/321 WRITE(108,322) (G(I),I=1,30)
C/322 FORMAT(5X,15(2X,15))
C/323 WRITE(108,322) (D(I),I=1,30)
    I=1
    WRITE(108,33)
    WRITE(108,33)
    WRITE(108,18)
    WRITE(108,19)
    WRITE(108,1111)
    WRITE(108,32)
    WRITE(108,33)
    WRITE(108,30)
    Q=K-1
    DO 400 I=1,Q
    WRITE(108,31) G(I),D(I)
400 CONTINUE
500 STOP
    END

```

3.4.7 Subrutina CALCUL

```

SUBROUTINE CALCUL(D,G,X,Y,Z,CONT,NR,M1,M2,A,8)
DIMENSION D(20,8),G(20,8),A(100,20),B(100,20),V1(8),V2(8),V3(8),L(
*4),X(20),Y(20),Z(20)
WRITE(108,25)
25 FORMAT(' ', 'CALCUL')
WRITE(108,27) (X(I),Y(I),Z(I),I=1,10)
27 FORMAT(' ', 3F10.3/)
WRITE(108,28) NR
28 FORMAT(' ', 'NR=', I2/)
INTEGER L,CONT,CONT1
I=1
1 M=1
2 DO 3 J=1,8
    V1(J)=D(I,J)
3 CONTINUE
    DO 4 J=1,8
    V2(J)=D(I+M,J)
4 CONTINUE
    IF(NR.EQ.1) GO TO 37
    MF=M1
    GO TO 35
37 MF=M2
35 DO 5 K=1,MF
    DO 6 J=1,8
    V3(J)=G(K,J)
6 CONTINUE
    DO 7 J=1,4
    L(J)=0
7 CONTINUE
    WRITE(108,18) (V1(J),J=1,4)
18 FORMAT(' ', 'V1=', 4F10.3/)
    WRITE(108,19) (V2(J),J=1,4)
19 FORMAT(' ', 'V2=', 4F10.3/)
    DO 9 J1=2,4
    DO 8 J2=2,4
    IF(V1(J1).EQ.V2(J2)) GO TO 10
    GO TO 8
10 L(J1)=V1(J1)
8 CONTINUE
9 CONTINUE
    WRITE(108,20) (L(J),J=1,4)
20 FORMAT(' ', 'L=', 4I2/)
    IF(L(1).EQ.0) GO TO 11
    N1=L(1)
    IF(L(2).EQ.0) GO TO 15
    N2=L(2)
    GO TO 17

```

```

15 IF(L(3).EQ.0) GO TO 16
   N2=L(3)
   GO TO 17
16 N2=L(4)
   GO TO 17
11 IF(L(2).EQ.0) GO TO 12
   N1=L(2)
   IF(L(3).EQ.0) GO TO 13
   N2=L(3)
   GO TO 17
13 N2=L(4)
   GO TO 17
12 N1=L(3)
   N2=L(4)
17 IF(N1.EQ.0) GO TO 5
   IF(N2.EQ.0) GO TO 5
   CALL PCTINT(X,Y,Z,N1,N2,V3,XT,YT,ZT,TD)
   IF(ID.EQ.0) GO TO 5
   CONT=CONT+1
   IF(NK.EQ.1) GO TO 21
   IF(NR.EQ.2) GO TO 22
   GO TO 5
22 B(CONT,I)=1
   B(CONT,I+M)=1
   A(CONT,K)=1
   GO TO 5
21 A(CONT,I)=1
   A(CONT,I+M)=1
   B(CONT,K)=1
   DO 31 II=1,12
   WRITE(108,32) (B(II,JJ),JJ=1,8)
32 FORMAT(' ',8F10.3/)
31 CONTINUE
   DO 30 II=1,12
   WRITE(108,29) (A(II,JJ),JJ=1,8)
29 FORMAT(' ',8F10.3/)
30 CONTINUE
5 CONTINUE
   M=M+1
   IF(NR.EQ.1) GO TO 33
   IF((I+M).LE.M2) GO TO 2
   I=I+1
   IF(I.NE.M2) GO TO 1
   GO TO 34
33 IF((I+M).LE.M1) GO TO 2
   I=I+1
   IF(I.NE.M1) GO TO 1
34 RETURN
   END

```

3.4.8 Subrutina PARAM

```

SUBROUTINE PARAM(X,Y,Z,K1,K2,K3,A1,B1,C1,D1)
DIMENSION X(20),Y(20),Z(20)
X1=X(K1)
Y1=Y(K1)
Z1=Z(K1)
X2=X(K2)
Y2=Y(K2)
Z2=Z(K2)
X3=X(K3)
Y3=Y(K3)
Z3=Z(K3)
A1=Y1*Z2-Y1*Z3+Y2*Z3-Y2*Z1+Y3*Z1-Y3*Z2
B1=X1*Z3-X1*Z2+X2*Z1-X2*Z3-X3*Z1+X3*Z2
C1=X1*Y2-X1*Y3+X2*Y3-X2*Y1+X3*Y1-X3*Y2
D1=X1*Y2*Z3+X2*Y3*Z1+X3*Y1*Z2-X3*Y2*Z1-X2*Y1*Z3-X1*Y3*Z2
D1=-D1
RETURN
END

```



```

57 RK2=(X2-X1)*AM1-AL1*(YY-Y1)/RN2
X23=RK2*AL23+X2
Y23=RK2*AM23+Y2
Z23=RK2*AN23+Z2
60 RN3=AL13*AM1-AM13*AL1
IF(RN3.NE.0.0) GO TO 58
GO TO 52
58 RK3=(X3-X1)*AM1-AL1*(YY-Y1)/RN3
X13=RK3*AL13+X3
Y13=RK3*AM13+Y3
Z13=RK3*AN13+Z3
WRITE(108,72) RN1,RN2,RN3
72 FORMAT(40X,'RN1=',F10.3,2X,'RN2=',F10.3,2X,'RN3=',F10.3/)
WRITE(108,79) X12,Y12,Z12,X23,Y23,Z23,X13,Y13,Z13

79 FORMAT(' ', 'U12=',3F10.3/' ', 'U23=',3F10.3/' ', 'U13=',3F10.3/)
62 IF(RN1.EQ.0.0) GO TO 1
IF(RN2.EQ.0.0) GO TO 5
IF(RN3.EQ.0.0) GO TO 7
GO TO 61
7 XP=X12
YP=Y12
ZP=Z12
XU=X23
YU=Y23
ZU=Z23
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF(IND.EQ.0) GO TO 8
WRITE(108,140) XT,YT,ZT
ID=1
GO TO 4
8 ID=0
GO TO 4
5 IF(RN3.EQ.0.0) GO TO 6
XP=X12
YP=Y12
ZP=Z12
XU=X13
YU=Y13
ZU=Z13
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF(IND.EQ.0) GO TO 9
WRITE(108,140) XT,YT,ZT
ID=1
GO TO 4
9 ID=0
GO TO 4
6 WRITE(108,10)
10 FORMAT(' ', 'EROARE:RN2=0,RN3=0'//)
GO TO 4
1 IF(RN2.EQ.0.0) GO TO 2
IF(RN3.EQ.0.0) GO TO 3
XP=X23
YP=Y23
ZP=Z23
XU=X13
YU=Y13
ZU=Z13
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF(IND.EQ.0) GO TO 11
WRITE(108,140) XT,YT,ZT
140 FORMAT(' ', 'COORDONATELE PUNCTULUI DE INTERS',3F10.3/)
ID=1
GO TO 4
11 ID=0
GO TO 4
3 WRITE(108,12)
12 FORMAT(' ', 'EROARE:RN1=0,RN3=0'//)
GO TO 4
2 WRITE(108,13)
13 FORMAT(' ', 'EROARE:RN1=0,RN2=0'//)
61 XP=X1
YP=Y1
ZP=Z1

```

```

XU=X2
YU=Y2
ZU=Z2
XC=X12
YC=Y12
ZC=Z12
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
XB1=XT
YB1=YT
ZB1=ZT
XP=X1
YP=Y1
ZP=Z1
XU=X3
YU=Y3
ZU=Z3
XC=X13
YC=Y13
ZC=Z13
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
XB3=XT
YB3=YT
ZB3=ZT
XP=X2
YP=Y2
ZP=Z2

XU=X3
YU=Y3
ZU=Z3
XC=X23
YC=Y23
ZC=Z23
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
XB2=XT
YB2=YT
ZB2=ZT
IF(XB1.NE.9999) GO TO 40
WRITE(108,82) XB1,YB1,ZB1,XB2,YB2,ZB2,XB3,YB3,ZB3
82 FORMAT(' ',XB1,YB1,ZB1',3F10.3/',' ,', B2,YB2,ZB2',3F10.3/',' ,',XB3,
*YB3,ZB3',3F10.3/)
XP=XB2
YP=YB2
ZP=ZB2
XU=XB3
YU=YB3
ZU=ZB3
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF(IND.EQ.0) GO TO 37
WRITE(108,140) XT,YT,ZT
10=1
GO TO 4
37 10=0
GO TO 36
40 IF(XB2.NE.9999) GO TO 32
XP=XB1
YP=YB1
ZP=ZB1
XU=XB3
YU=YB3
ZU=ZB3
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF(IND.EQ.0) GO TO 30
WRITE(108,140) XT,YT,ZT
10=1
GO TO 4
30 10=0
GO TO 36

32 IF(XB3.NE.9999) GO TO 50
XP=XB1
YP=YB1
ZP=ZB1
XU=XB2
YU=YB2
ZU=ZB2
XC=XX
YC=YY
ZC=ZZ

```



```

CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF(IND.EQ.0) GO TO 33
WRITE(108,140) XT,YT,ZT
ID=1
GO TO 4
33 ID=0
GO TO 36
50 WRITE(108,135)
35 FORMAT(' ',ERDARE:DREAPTA IG TAIE LATURILE TRIUNGHULUI'//)
36 CONTINUE
4 RETURN
END

```

3.4.10 Subrutina LIMITE

```

SUBROUTINE LIMITE(X1,Y1,Z1,X2,Y2,Z2,XC,YC,ZC,XT,YT,ZT,IND)
IF(X1.LE.X2) GO TO 1
XINF=X2
XSUP=X1
GO TO 2
1 XINF=X1
XSUP=X2
2 CONTINUE
IF(Y1.LE.Y2) GO TO 3
YINF=Y2
YSUP=Y1
GO TO 4
3 YINF=Y1
YSUP=Y2
4 CONTINUE
IF(Z1.LE.Z2) GO TO 5
ZINF=Z2
ZSUP=Z1
GO TO 6
5 ZINF=Z1
ZSUP=Z2
6 CONTINUE
IF(XINF.LE.XC) GO TO 7
GO TO 8
7 IF(XC.LE.XSUP) GO TO 9
GO TO 8
9 XT=XC
IF(YINF.LE.YC) GO TO 10
GO TO 8
10 IF(YC.LE.YSUP) GO TO 11
GO TO 8
11 YT=YC
IF(ZINF.LE.ZC) GO TO 12
GO TO 8
12 IF(ZC.LE.ZSUP) GO TO 13
GO TO 8
13 ZT=ZC
GO TO 14
8 XT=9999
YT=9999
ZT=9999
14 IF(XT.EQ.9999) GO TO 15
IF(YT.EQ.9999) GO TO 15
IF(ZT.EQ.9999) GO TO 15
GO TO 16
15 IND=0
GO TO 17
16 IND=1
17 RETURN
END

```

3.4.11 Program principal desen INTPOL

```

0001      DIMENSION X1(100),Y1(100),Z1(100),X(30),Y(30),Z(30)
0002      LOGICAL*1 V1(18),V2(18),V3(18),S1(2),S2(2),S3(2),LF(32)
0003      DATA V1/'1','1','2','2','1','3','1','4','1','5','1','6','1','7','1',
      *'8','1','9','1'/
0004      DATA V2/'1','2','2','2','3','2','4','2','5','2','6','2','7','2',
      *'8','2','9','2'/
0005      DATA V3/'1','3','2','3','3','3','3','4','3','5','3','6','3','7','3',
      *'8','3','9','3'/
0006100    TYPE 1000
00071000  FORMAT(' NUME FISIER DATE=? ',#)
0008      ACCEPT 2,IQ,(LF(I),I=1,IQ)
00092     FORMAT(Q,32A1)
0010      ISW=0
0011      IF (IQ.LE.0)STOP
0012      CALL ASSIGN(2,LF,IQ)
0013      CALL PLOTS(0,1,7)
0014304   READ(2,*,END=500)ITIP,N
0015      GOTO(301,302,303)ITIP
0016      TYPE *,'EROARE TIP FRELUCRARE',ITIP
0017500   CALL CLOSE(2)
0018      CALL PLOT(0.,0.,999)
0019      GOTO 100
0020303   DO 130 I=1,N
0021130   READ(2,*) X1(I),Y1(I),Z1(I)
0022      CALL NEWPEN(2)
0023      J=1
0024      DO 110 I=1,N-1
0025      S1(1)=V1(J)
0026      S1(2)=V1(J+1)
0027      S2(1)=V2(J)
0028      S2(2)=V2(J+1)
0029      S3(1)=V3(J)
0030      S3(2)=V3(J+1)
0031      XP1=-X1(I)
0032      YP1=-Y1(I)
0033      XP2=-X1(I)
0034      YP2=Z1(I)
0035      XP3=Y1(I)
0036      YP3=Z1(I)
0037      XQ1=-X1(I+1)
0038      YQ1=-Y1(I+1)
0039      XQ2=-X1(I+1)
0040      YQ2=Z1(I+1)
0041      XQ3=Y1(I+1)
0042      YQ3=Z1(I+1)
0043      CALL PLOT(XP1-0.1,YP1,3)
0044      CALL CIRCLE(XP1,YP1,360.)
      CALL SYMBOL(XP1+0.2,YP1+0.2,0.4,S1,0.,2,1.,0.)
0045      CALL PLOT(XP1,YP1,3)
0046      CALL PLOT(XQ1,YQ1,2)
0047      CALL PLOT(XP2-0.1,YP2,3)
0048      CALL CIRCLE(XP2,YP2,360.)
      CALL SYMBOL(XP2+0.2,YP2+0.2,0.4,S2,0.,2,1.,0.)
0049      CALL PLOT(XP2,YP2,3)
0050      CALL PLOT(XQ2,YQ2,2)
0051      CALL PLOT(XP3-0.1,YP3,3)

```

1 = octaedru
2 = piramida
3 = poligon de intersec-
ție.

```

0052          CALL CIRCLE(XP3,YP3,360.)
          C    CALL SYMBOL(XP3+0.2,YP3+0.2,0.4,S3,0.,2,1.,0.)
0053          CALL PLOT(XP3,YP3,3)
0054          CALL PLOT(XQ3,YQ3,2)
0055          J=J+2
0056          110 CONTINUE
0057          XP1=-X1(N)
0058          YP1=-Y1(N)
0059          XP2=-X1(N)
0060          YP2=Z1(N)
0061          XP3=Y1(N)
0062          YP3=Z1(N)
0063          XQ1=-X1(1)
0064          YQ1=-Y1(1)
0065          XQ2=-X1(1)
0066          YQ2=Z1(1)
0067          XQ3=Y1(1)
0068          YQ3=Z1(1)
0069          S1(1)=V1(J)
0070          S1(2)=V1(J+1)
0071          S2(1)=V2(J)
0072          S2(2)=V2(J+1)
0073          S3(1)=V3(J)
0074          S3(2)=V3(J+1)
0075          CALL PLOT(XP1-0.1,YP1,3)
0076          CALL CIRCLE(XP1,YP1,360.)
          C    CALL SYMBOL(XP1+0.2,YP1+0.2,0.4,S1,0.,2,1.,0.)
0077          CALL PLOT(XP1,YP1,3)
0078          CALL PLOT(XQ1,YQ1,2)
0079          CALL PLOT(XP2-0.1,YP2,3)
0080          CALL CIRCLE(XP2,YP2,360.)
          C    CALL SYMBOL(XP2+0.2,YP2+0.2,0.4,S2,0.,2,1.,0.)
0081          CALL PLOT(XP2,YP2,3)
0082          CALL PLOT(XQ2,YQ2,2)
0083          CALL PLOT(XP3-0.1,YP3,3)
0084          CALL CIRCLE(XP3,YP3,360.)
          C    CALL SYMBOL(XP3+0.2,YP3+0.2,0.4,S3,0.,2,1.,0.)
0085          CALL PLOT(XP3,YP3,3)
0086          CALL PLOT(XQ3,YQ3,2)
0087          GOTO 304
0088          302 DO 413 I=1,N
0089          413 READ(2,*) X1(I),Y1(I),Z1(I)
0090          CALL NEWPEN(1)
0091          IF(ISW.NE.0)GOTO 414
0092          CALL PLOT(0.,-25.,3)
0093          CALL PLOT(0.,25.,2)
0094          CALL PLOT(-25.,0.,3)
0095          CALL PLOT(25.,0.,2)
0096          ISW=1
0097          414 DO 450 I=2,N-1
0098          XC1=X1(I)
0099          YC1=Y1(I)
0100          ZC1=Z1(I)
0101          XC2=X1(I+1)
0102          YC2=Y1(I+1)
0103          ZC2=Z1(I+1)

```

```

0104          CALL DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2)
0105 450      CONTINUE
0106          CALL DR1P(X1(N),Y1(N),Z1(N),X1(2),Y1(2),Z1(2))
0107          XC1=X1(1)
0108          YC1=Y1(1)
0109          ZC1=Z1(1)
0110          XC2=X1(2)
0111          YC2=Y1(2)
0112          ZC2=Z1(2)
0113          CALL DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2)
0114          DO 451 I=3,N
0115          XC1=X1(1)
0116          YC1=Y1(1)
0117          ZC1=Z1(1)
0118          XC2=X1(I)
0119          YC2=Y1(I)
0120          ZC2=Z1(I)
0121          CALL DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2)
0122 451      CONTINUE
0123          GOTO 304
0124 301      DO 13 I=1,N
0125 13        READ(2,*) X(I),Y(I),Z(I)
0126          CALL NEWPEN(1)
0127          IF(1SW.NE.0)GOTO 415
0128          1SW=1
0129          CALL PLOT(0.,-25.,3)
0130          CALL PLOT(0.,25.,2)
0131          CALL PLOT(-25.,0.,3)
0132          CALL PLOT(25.,0.,2)
0133 415      DO 250 I=2,N-1
0134          XC1=X(I)
0135          YC1=Y(I)
0136          ZC1=Z(I)
0137          XC2=X(I+1)
0138          YC2=Y(I+1)
0139          ZC2=Z(I+1)
0140          CALL DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2)
0141 250      CONTINUE
0142          XC2=X(2)
0143          YC2=Y(2)
0144          ZC2=Z(2)
0145          XC1=X(N-1)
0146          YC1=Y(N-1)
0147          ZC1=Z(N-1)
0148          CALL DR1P(XC1,YC1,ZC1,XC2,YC2,ZC2)
0149          XP1=-X(1)
0150          YP1=-Y(1)
0151          XP2=-X(1)
0152          YP2=Z(1)
0153          XP3=Y(1)
0154          YP3=Z(1)
0155          CALL PLOT(XP1-0.1,YP1,3)
0156          CALL CIRCLE(XP1,YP1,360.)
0157          DO 251 I=1,N-1
0158          XC1=X(I)
0159          YC1=Y(I)

```

```

0160          ZC1=Z(1)
0161          XC2=X(I+1)
0162          YC2=Y(I+1)
0163          ZC2=Z(I+1)
0164          CALL DRFP(XC1,YC1,ZC1,XC2,YC2,ZC2)
0165          251 CONTINUE
0166          XP1=-X(N)
0167          YP1=-Y(N)
0168          XP2=-X(N)
0169          YP2=Z(N)
0170          XP3=Y(N)
0171          YP3=Z(N)
0172          CALL PLOT(XP1-0.1,YP1,3)
0173          CALL CIRCLE(XP1,YP1,360.)
0174          DO 252 I=2,N-1
0175             XC1=X(I)
0176             YC1=Y(I)
0177             ZC1=Z(I)
0178             XC2=X(I)
0179             YC2=Y(I)
0180             ZC2=Z(I)
0181             CALL DRFP(XC1,YC1,ZC1,XC2,YC2,ZC2)
0182             252 CONTINUE
0183             GOTO 304
0184             END

```

3.5. Aplicații rezolvate prin programul INTPOL

3.5.1 Intersecția dintre doi tetraedri regulați (Fig. 3.6.)

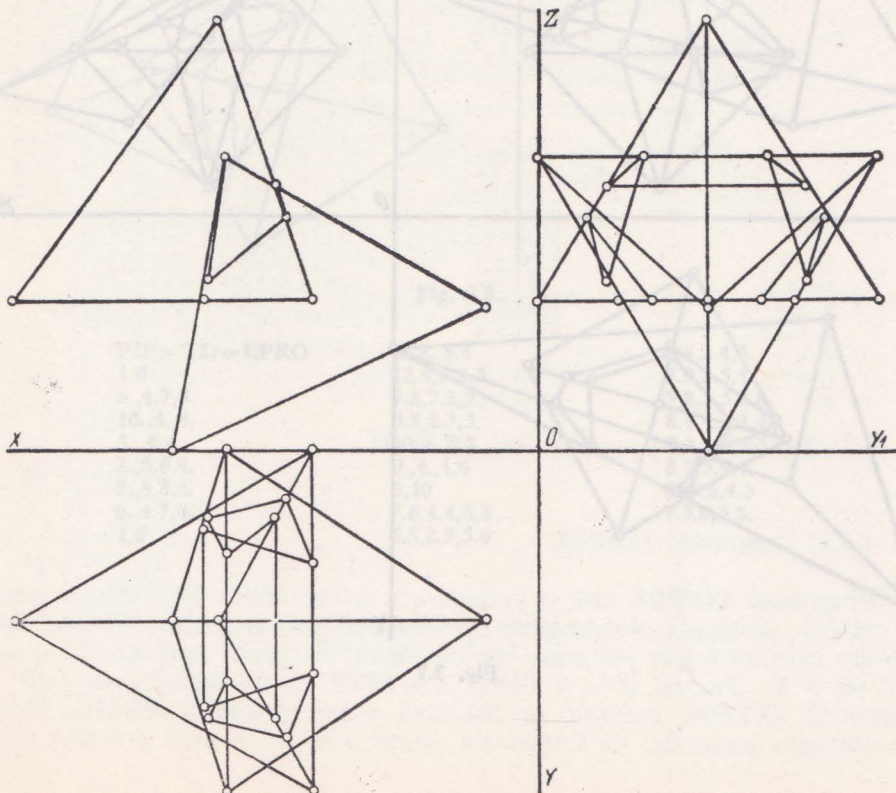


Fig. 3.6

PIP>TI:=21.PRO	8.3,11.7,7.8	6.7,4.,6.2
2,4	8.3,2.7,7.8	8.8,4.5,4.5
13.9,7.2,4.	1.4,7.2,3.8	8.3,5.5,7.8
6.,11.7,4.	3,12	7.,4.5,7.
6.,2.7,4.	6.,8.6,4.	7.,9.8,7
8.5,7.2,11.4	8.9,9.5,4.	8.3,8.8,7.8
2,4	8.9,4.8,4.	8.8,9.8,4.5
9.7,7.2,0.	6.,5.7,4.	6.7,10.3,6.2

3.5.2 Intersecția dintre un tetraedru și un octaedru (Fig. 3.7.)

3.5.3 Intersecția dintre doi octaedri (Fig. 3.8)

3.5.4 Idem. Date schimbate (Fig. 3.9)

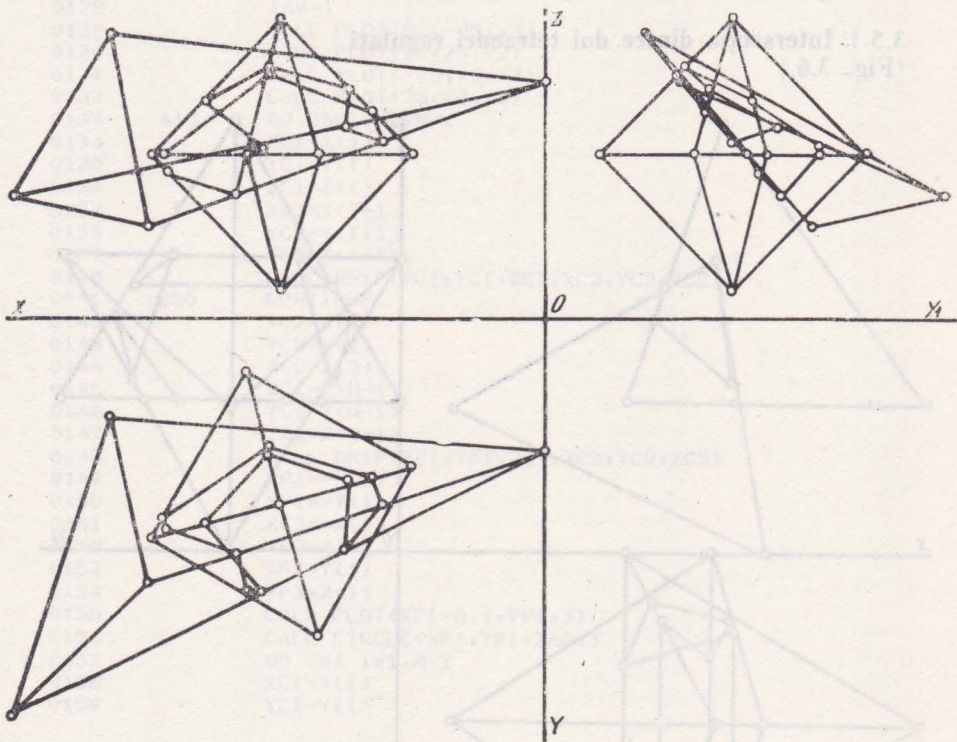


Fig. 3.7

PIP > TI := 11.PRO

1,6
7.,4,9,7,9
10.4,5,8,4,3
7,9,1,4,4,3
3.5,3,9,4,3
6.,8,4,4,3
7.,4,9.,7
2,4

0.,3,5,6,2
11.5,2,6,7,5
10.5.7.,2.4
14.1,10.5,3.2
3,12
4.3,4,9,4,6
4.6,4,2,5,4
7.3,3,4,6,4
10.1,5,3,4,3

10.,5,6,3,8
8.2,6,2,3,2
7.9,7,2,4,3
7.5,7,2,4,5
9.,5,4,5,7
7.3,3,6,6,6
5.2,4,3,6.
5.3,6,1,5.

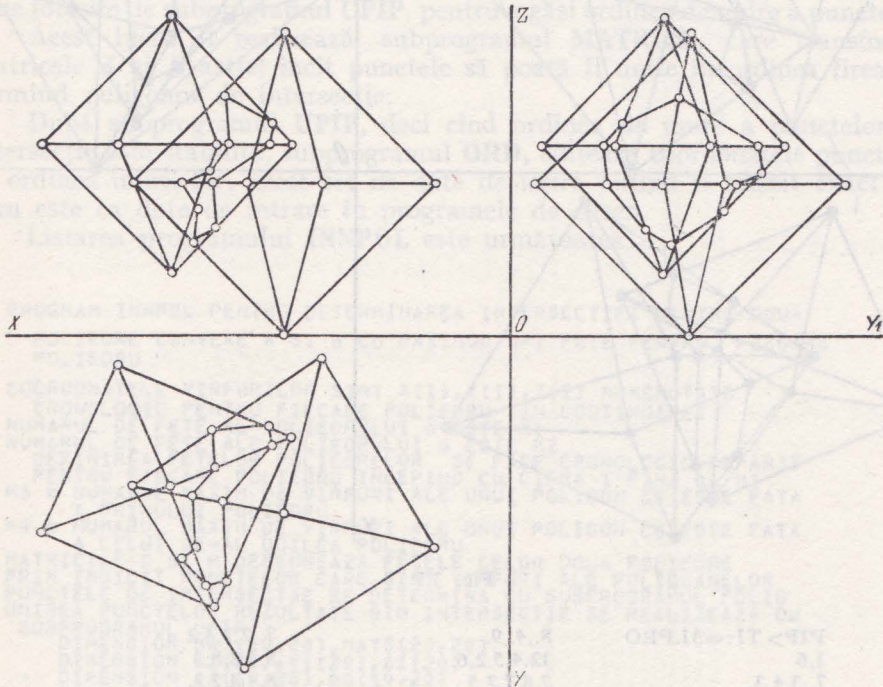


Fig. 3.8

PIP > TI := 1.PRO

1,6
6.,4,7,8.
10.,4.,4.
5.,,6,4.
2.,,5,6,4.
7.,8,8,4.
6.,4,7,0.
1,6

9.,4.,8,4
12.4,5,2,5.
7.8,7,2,5.
5.8,2,7,5.
10.4.,8,5.
9.,4.,1.6
3,10
7.6,4,4,6,3
6.3,2,9,5,6

6.4,2,4,5.
7.9,2,5,4.
7.8,3,5,2,8
8.4,4,2,2,4
8.3,5,6,3,3
8.7,5,9,4.
8.,6,6,4,3
7.5,6,5,5.

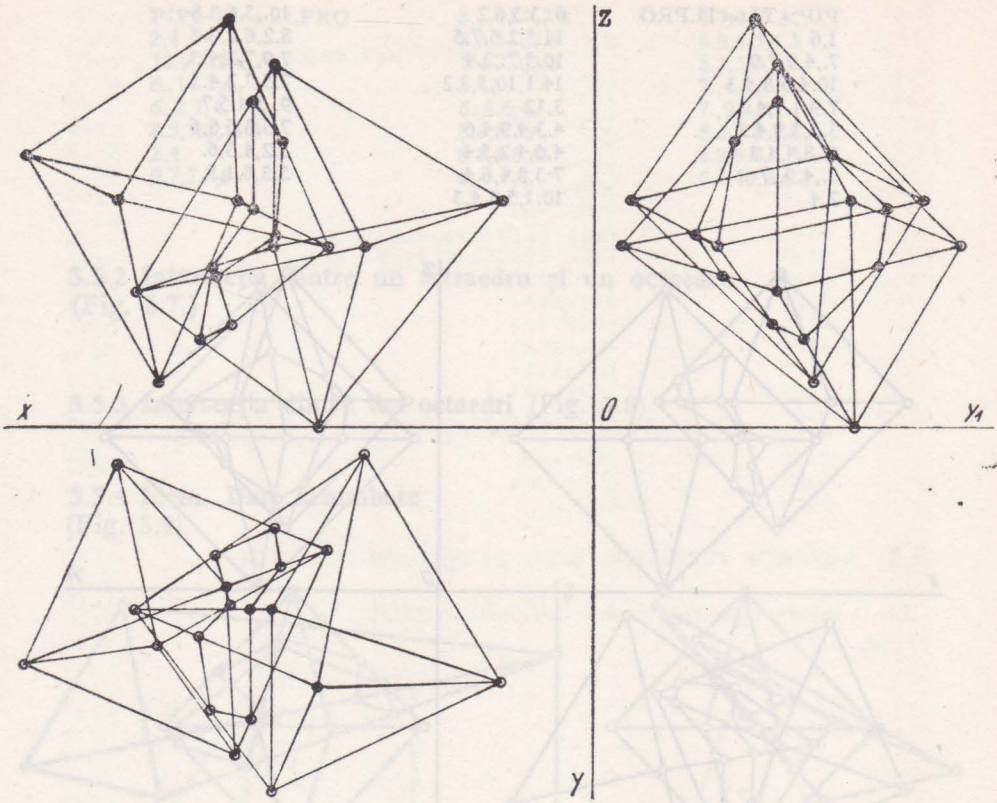


Fig. 3.9

PIP>TI:=51.PRO
 1,6
 7.,3.4.,3.
 10.,4.,3.
 5.,.6.4.
 2.,.5.6.5.
 7.,8.,4.
 6.,4.7.0.
 1,6

8.,4.,9.
 12.4.5.2.6.
 7.8.7.2.5.
 5.8.2.7.4.
 10.4.,8.5.
 9.5.4.8.1.
 3.8
 7.5.4.3.7.2
 6.8.3.3.6.4

7.,2.2.4.2
 8.3.2.8.3.
 8.4.4.2.2.
 8.5.4.3.1.9
 7.2.4.6.3.7
 6.7.4.6.4.4
 PIP> Z
 >

3.6. Extinderea programului INTPOL

3.6.1. Programul INNPOL

Programul **INNPOL** este o extindere a programului **INTPOL** în sensul că rezolvă problema determinării poligonului de intersecție dintre două poliedre care au fețele poligoane cu un număr de laturi mai mare sau egal cu 3 ($m \geq 3$). Pentru $m = 3$ **INNPOL** \equiv **INTPOL**. Îmbunătățirile față de programul **INTPOL** constau în folosirea subprogramului **POLIG** pentru determinarea punctului de intersecție dintre o latură a unui poliedru și un

poligon plan, ce reprezintă o față a celui de-al doilea poliedru în cazul că $m > 3$ laturi.

Din utilizarea programului **INNPOL** în diferite cazuri mai speciale de intersecții ale unor poliedre, au decurs câteva adăugiri răspunzând necesității de a alege datele cu o precizie maximă (problema coplanarității punctelor). Dar cum această precizie este greu de realizat s-a stabilit folosirea unui coeficient de eroare de 0.01 cm.

De asemenea din motivul că multe puncte de intersecție se suprapun ca valori, iar acestea pot fi identice sau nu, matricele A și B nu mai erau bine folosite de subprogramul **UPIP**, pentru a găsi ordinea de unire a punctelor.

Acest lucru îl realizează subprogramul **MATRICE**, care transformă matricele A și B astfel încît punctele să poată fi unite în ordinea firească, formînd poligonul de intersecție.

După subprogramul **UPIP**, deci cînd ordinea de unire a punctelor de intersecție este stabilită, subprogramul **ORD**, editează coordonatele punctelor în ordinea unirii lor, acest set de date de ieșire putînd fi folosit exact așa cum este ca date de intrare în programele de desen.

Listarea programului **INNPOL** este următoarea:

```

C PROGRAM INNPOL PENTRU DETERMINAREA INTERSECȚIEI DINTRE DOUA
C POLIEDRE CONVEXE A SI B CU MAXIMUM OPT FETE PENTRU FIECARE
C POLIEDRU
C COORDONATELE VIRFURILOR SINT X(I),Y(I),Z(I) NUMEROTATE
C CRONOLOGIC PENTRU FIECARE POLIEDRU (IN CONTINUARE)
C NUMARUL DE FETE AL POLIEDRULUI A ESTE M1
C NUMARUL DE FETE AL POLIEDRULUI B ESTE M2
C DEFINIREA FETELOR POLIEDRELOR SE FACE CRONOLOGIC SEPARAT
C PENTRU FIECARE POLIEDRU INCEPIND CU CIFRA 1 PINA LA M1
M3 = NUMARUL MAXIM DE VIRFURI ALE UNUI POLIGON CE ESTE FATA
C A PRIMULUI POLIEDRU
M4 = NUMARUL MAXIM DE VIRFURI ALE UNUI POLIGON CE ESTE FATA
C A CELUI DE-AL DOILEA POLIEDRU
C MATRICILE C SI H DESENEAZA FETELE CELOR DOUA POLIEDRE
C PRIN INDICI I PUNCTELOR CARE SINT VIRFURI ALE POLIGONELOR
C PUNCTELE DE INTERSECȚIE SE DETERMINA CU SUBPROGRAMUL POLIG
C UNIREA PUNCTELOR REZULTATE DIN INTERSECȚIE SE REALIZEAZA CU
C SUBPROGRAMUL UPIP
DIMENSION MAT(20,20),MATS(20,20)
DIMENSION XI(20),YI(20),ZI(20)
DIMENSION AA(30,20),BB(30,20)
DIMENSION D(30),G(30)
DIMENSION C(30,20),H(30,20),F(30,20),A(30,20),B(30,20),
* X(30),Y(30),Z(30),K(20),KL(20),KB(20)
INTEGER G,G,D
INTEGER C,H,CONT,CONT1,A,B
DO 55 I=1,30
DO 36 J=1,20

AA(I,J)=0
BB(I,J)=0
B(I,J)=0
A(I,J)=0
36 CONTINUE
35 CONTINUE
DO 7 I=1,20
DO 60 J=1,20
MAT(I,J)=0
60 CONTINUE
7 CONTINUE
READ(105,1) N
1 FORMAT(I4)
C N = NUMARUL DE VIRFURI ALE CELOR DOUA POLIEDRE IN TOTAL
READ(105,2) (X(I),Y(I),Z(I),I=1,N)
2 FORMAT(3F10,3,50X)
3 READ(105,3) M1,M2
3 FORMAT(2I4)
4 READ(105,4) M3,M4

```

```

4 FORMAT(2I4)
M31=M3+1
READ(105,5) ((C(I,J),J=1,M31),I=1,M1)
5 FORMAT(8(I7,3X))
M41=M4+1
READ(105,6) ((H(I,J),J=1,M41),I=1,M2)
6 FORMAT(8(I7,3X))
DO 22 I=1,M1
WRITE(108,23) (C(I,J),J=1,M31)
22 FORMAT(' ',12I2/)
CONTINUE
DO 24 I=1,M2
WRITE(108,25) (H(I,J),J=1,M41)
25 FORMAT(' ',12I2/)
CONTINUE
DO 8 I=1,M1
DO 9 J=1,M31
E(I,J)=C(I,J)
9 CONTINUE
8 CONTINUE
DO 10 I=1,M2
DO 11 J=1,M41
F(I,J)=H(I,J)
11 CONTINUE
10 CONTINUE
DO 12 I=1,M1
DO 13 J=1,M31
KL(J)=C(I,J)
61 CONTINUE
20 WRITE(108,20) (KL(J),J=1,M31)
20 FORMAT(' ',10I5/)
K1=KL(2)
K2=KL(3)
K3=KL(4)
CALL PARAM(X,Y,Z,K1,K2,K3,A1,B1,C1,D1)
E(I,M31+1)=A1
E(I,M31+2)=B1
E(I,M31+3)=C1
E(I,M31+4)=D1
J2 CONTINUE
DO 13 J=1,M2
DO 62 I=1,M1
K8(J)=H(I,J)
62 CONTINUE
WRITE(108,20) (K8(J),J=1,M4)
K1=K8(2)
K2=K8(3)
K3=K8(4)
CALL PARAM(X,Y,Z,K1,K2,K3,A1,B1,C1,D1)
F(I,M41+1)=A1
F(I,M41+2)=B1
F(I,M41+3)=C1
F(I,M41+4)=D1
WRITE(108,21) A1,B1,C1,D1,(F(I,J),J=1,M41)
21 FORMAT(' ',20X,20(F10.3,2X)/)
13 CONTINUE
CONT=0
DO 26 I=1,M1
M34=M31+4
WRITE(108,27) (E(I,J),J=1,M34)
27 FORMAT(' ',20F10.3/)
26 CONTINUE

```

3.6. Programul INTPOL

3.6.1. Programul INNPOL

Programul INNPOL este o extindere a programului INTPOL. Acesta este conceput pentru a calcula intersecția dintre două poliedre în spațiul tridimensional. Programul INNPOL este conceput pentru a calcula intersecția dintre două poliedre în spațiul tridimensional. Programul INNPOL este conceput pentru a calcula intersecția dintre două poliedre în spațiul tridimensional.

```

IN=1
NR=1
CALL CALCUL(E,F,X,Y,Z,CONT,N,NR,M1,M2,M3,M4,XI,YI,ZI,IN,A,B,NRP)
NR=2
CONT=CONT+1
CALL CALCUL(F,E,X,Y,Z,CONT,N,NR,M1,M2,M3,M4,XI,YI,ZI,IN,A,B,NRP)
M12=M1+M2
DO 63 I=1,CONT
DO 64 J=1,M1
MAT(I,J)=A(I,J)
64 CONTINUE
63 CONTINUE
DO 65 I=1,CONT
DO 66 J=1,M2
M1J=M1+J
MAT(I,M1J)=B(I,J)
66 CONTINUE
65 CONTINUE
CALL MATRICE(MAT,CONT,M12,MATS,I2)
DO 80 I=1,I2
DO 81 J=1,M1
AA(I,J)=MATS(I,J)
81 CONTINUE
80 CONTINUE
DO 82 I=1,I2
DO 83 J=1,M2
M1J=M1+J
BB(I,J)=MATS(I,M1J)
83 CONTINUE
82 CONTINUE
28 WRITE(108,28)
28 FORMAT(' ',MATRICEA PUNCTELOR DE INTERSECTIE'/' ', 'PENTRU POLIEDR

*UL A ESTE'/)
DO 15 I=1,I2
WRITE(108,14) (MATS(I,J),J=1,M1)
14 FORMAT(' ',16I3/)
15 CONTINUE
WRITE(108,29)
29 FORMAT(' ',MATRICEA PUNCTELOR DE INTERSECTIE'/' ', 'PENTRU POLIEDR'
*UL B ESTE'/)
DO 16 I=1,I2
M11=M1+1
WRITE(108,17) (MATS(I,J),J=M11,M12)
17 FORMAT(' ',16I3/)
16 CONTINUE
DO 67 I=1,I2
WRITE(108,58) (MATS(I,J),J=1,M12)
68 FORMAT(' ',MAT'/' ',16I3/)
67 CONTINUE
WRITE(108,51)
51 FORMAT(' ',COORDONATELE VIRFURILOR'/' ', 'POLIEDRELOR A SI B SINT
*URMATOARELE'/)
WRITE(108,52) (X(I),Y(I),Z(I),I=1,N)
52 FORMAT(' ',3F10,3/)
KK=12
30 FORMAT(' ',SE VOR UNI URMATOARELE PUNCTE: '/')
31 FORMAT(31X,'PUNCTUL ',I5,' CU PUNCTUL ',I5/)
32 FORMAT(' ', '*****')
33 FORMAT(' ')
18 FORMAT(' ',DETERMINEAREA ORDINII DE UMIERE'/)
19 FORMAT(' ',PENTRU PUNCTELE DE INTERSECTIE'/)
1111 FORMAT(' ',A DOUA POLIEDRE A SI B'/)
J=0
L=KK
100 KK=1
P=0
110 CONTINUE
J=J+1
IF(J,GE,L) GO TO 320
120 CONTINUE
I=1
C/ WRITE(108,20) M
C/ WRITE(108,20) I
C/ WRITE(108,20) M
130 CONTINUE
M=1
N=J
140 GO TO 160
150 IF(I,GE,8) GO TO 110

```

```

C/   WRITE(108,20) I
      I=I+1
160  IF(AA(J,I),NE,1) GO TO 150
170  GO TO 220
180  IF(P=1) 200,190,200
190  P=0
      GO TO 110
200  I=I+1
C/   WRITE(108,20) I
      P=1
      GO TO 130
210  M=M+1
      IF(M.GT,8) GO TO 180
C/   WRITE(108,20) M
220  IF(BB(N,M),NE,1) GO TO 210
      G(KK)=N
230  GO TO 250
240  IF(N,GE,L) GO TO 270
250  N=N+1
      IF(AA(N,I),NE,1,OR,BB(N,M),NE,1) GO TO 240
      D(KK)=N
      KK=KK+1
260  GO TO 280
270  N=J
C/   WRITE(108,20) M
      GO TO 210
280  IF(M,LE,8) GO TO 270
290  GO TO 310
300  I=I+1
      P=1
      GO TO 120
310  IF(P,NE,1) GO TO 300
      P=0
C/   WRITE(108,20) M
320  IF(J,LT,L) GO TO 110
      I=1
      WRITE(108,33)
      WRITE(108,33)
      WRITE(108,18)
      WRITE(108,19)
      WRITE(108,11,1)
      WRITE(108,32)
      WRITE(108,33)
      WRITE(108,30)
      Q=KK-1
      DO 400 I=1,Q
      WRITE(108,31) G(I),D(I)

400  CONTINUE
      CALL ORD(XI,YI,ZI,G,D,Q)
500  STOP
      END

```

Așa cum rezultă din comentariile de început ale programului INNPOL, problema se încheie cu specificarea ordinii de unire a punctelor de intersecție dată de subprogramul UPIP. Programul INNPOL poate fi să zicem „curățat“ într-o etapă ulterioară înțelegerii lui, prin scoaterea numeroaselor instrucțiuni WRITE și prin amplificarea lui în sensul cerinței ca rezultatul scos de subprogramul UPIP să fie automat desenat sau vizualizat și, bine înțeles, studiat din punctul de vedere al liniilor și suprafețelor ascunse. Acest fapt este absolut obligatoriu pentru utilizator.

3.6.2. Subprogramul MATRICE

```

SUBROUTINE MATRICE(MAT,CONT,M12,MATS,I2)
DIMENSION MAT(20,20),MATS(20,20)
INTEGER MAT,MATS,CONT,CONT1
DO 14 I=1,CONT
WRITE(108,15) (MAT(I,J),J=1,M12)
15 FORMAT(' ', 'MAT INT',16I3/)
14 CONTINUE
DO 3 I=1,20
DO 4 J=1,20
MATS(I,J)=0
4 CONTINUE
3 CONTINUE
DO 6 I=1,CONT1
CONT1=CONT-1
L=1
8 I=I+L
WRITE(108,18) I1
18 FORMAT(' ', 'I1',I3/)
K=0
DO 7 J=1,M12
WRITE(108,19) I,J,I1
19 FORMAT(' ', 'I,J,I1',3I3/)
IF (MAT(I,J).EQ.1) GO TO 9
GO TO 7
9 IF (MAT(I,J).EQ.1) GO TO 11
GO TO 7
11 K=K+1
CONTINUE
WRITE(108,20) K
20 FORMAT(' ', 'K=',I3/)
IF (K.GE.3) GO TO 12
WRITE(108,21) (MAT(I1,J),J=1,M12)
21 FORMAT(' ', 'MAT',16I3/)
GO TO 22
DO 13 J=1,M12
MAT(I1,J)=0
13 CONTINUE
WRITE(108,21) (MAT(I1,J),J=1,M12)
22 L=L+1
WRITE(108,23) L,I
23 FORMAT(' ', 'L,I',2I3/)
IF ((1+L).LE.CONT1) GO TO 8
6 CONTINUE
DO 16 I=1,CONT
WRITE(108,17) (MAT(I,J),J=1,M12)
17 FORMAT(' ', 'MAT',16I3/)
16 CONTINUE

I2=0
DO 25 I=1,CONT
J=1
27 IF (MAT(I,J).NE.0) GO TO 26
J=J+1
IF (J.LE.M12) GO TO 27
GO TO 25
26 I2=I2+1
DO 28 J=1,M12
MATS(I2,J)=MAT(I,J)
28 CONTINUE
25 CONTINUE
DO 30 I=1,I2
WRITE(108,29) (MATS(I,J),J=1,M12)
29 FORMAT(' ', 'MATS',16I3/)
30 CONTINUE
RETURN
END

```

3.6.3 Subprogramul ORD

```

SUBROUTINE ORD(XI,YI,ZI,G,D,Q)
  INTEGER Q,G,D
  DIMENSION K(20),XI(20),YI(20),ZI(20),G(30),D(30),AUX(20)
  LF=Q
  DO 1 I=1,20
1  K(I)=0
   K(I)=G(I)
   K(I)=D(I)
   M=I
   N=I
   V=D(I)
2  J=1
3  IF(V.EQ.G(J).AND.M.NE.J) GO TO 4
   J=J+1
   IF(J.LE.Q) GO TO 3
   I=1
5  IF(V.EQ.D(I).AND.I.NE.N) GO TO 6
   I=I+1
   IF(I.LE.Q) GO TO 5
7  WRITE(108,7)
4  K(M)=D(J)
   M=M+1
   V=D(J)
   N=J
   GO TO 8
6  K(M)=G(I)
   M=M+1
   V=G(I)
   N=I
   DO 12 JJ=1,LF
   AUX(JJ)=G(JJ)
   G(JJ)=D(JJ)
   D(JJ)=AUX(JJ)
12 CONTINUE
8  IF(M.LE.(Q+1)) GO TO 2
   WRITE(108,107)
407 FORMAT(1,'COORDONATELE VIRFURILOR POLIGONULUI DE INTERSECȚIE'/)
   DO 9 M=1,LF
   L=K(M)
10  WRITE(108,10) M,XI(L),YI(L),ZI(L)
   FORMAT(1,'10X,12,3(2X,F8.3)/')
9  CONTINUE
11 RETURN
END

```

3.6.4 Subprogramul PARAM

```

SUBROUTINE PARAM(X,Y,Z,K1,K2,K3,A1,B1,C1,D1)
  DIMENSION X(30),Y(30),Z(30)
  X1=X(K1)
  Y1=Y(K1)
  Z1=Z(K1)
  X2=X(K2)
  Y2=Y(K2)
  Z2=Z(K2)
  X3=X(K3)
  Y3=Y(K3)
  Z3=Z(K3)
  A1=Y1*Z2-Y1*Z3+Y2*Z3-Y2*Z1+Y3*Z1-Y3*Z2
  B1=X1*Z3-X1*Z2+X2*Z1-X2*Z3-X3*Z1+X3*Z2
  C1=X1*Y2-X1*Y3+X2*Y3-X2*Y1+X3*Y1-X3*Y2
  D1=X1*Y2*Z3+X2*Y3*Z1+X3*Y1*Z2-X3*Y2*Z1-X2*Y1*Z3-X1*Y3*Z2
  D1=D1
  RETURN
END

```

3.6.5 Subprogramul CALCUL

```

SUBROUTINE CALCUL(D,G,X,Y,Z,CONT,N,NR,M1,M2,M3,M4,XI,YI,ZI,IN,A,B,
*NRP)
DIMENSION XI(20),YI(20),ZI(20)
DIMENSION D(30,20),G(30,20),A(30,20),B(30,20),V1(20),V2(20),V3(20)
*L(20),X(30),Y(30),Z(30),NN(20)
INTEGER L,CONT,CONT1,A,B
25 FORMAT(' ', 'CALCUL' /)
1 I=1
1 M1=M4+1
M31=M3+1
M35=M3+5
2 DO 3 J=1,M35
V1(J)=D(I,J)
3 CONTINUE
DO 4 J=1,M35
V2(J)=D(I+M,J)
4 CONTINUE
IF(NR.EQ.1) GO TO 37
MF=M1
GO TO 38
37 MF=M2
38 DO 5 K=1,MF
M45=M4+5
DO 6 J=1,M45
V3(J)=G(K,J)
6 CONTINUE
DO 7 J3=1,20
L(J3)=0
7 CONTINUE
18 WRITE(108,18) (V1(J),J=1,M31)
FORMAT(' ', 'V1=',4F10.3/)
19 WRITE(108,19) (V2(J),J=1,M31)
FORMAT(' ', 'V2=',4F10.3/)
MNR=M31
IF(NR.EQ.1) GO TO 42
MNR=M41
DO 9 J1=2,MNR
DO 8 J2=2,MNR
IF(V1(J1).EQ.0) GO TO 8
IF(V2(J2).EQ.0) GO TO 8
IF(V1(J1).EQ.V2(J2)) GO TO 10
GO TO 8
10 L(J1)=V1(J1)
WRITE(108,40) V1(J1),V2(J2),L(J1)
40 FORMAT(' ', 'V1,V2,L',3F10.3/)

8 CONTINUE
9 CONTINUE
DO 15 J4=1,M31
NN(J4)=0
15 CONTINUE
MA=MAX0(M31,M41)
WRITE(108,20) (L(J),J=1,MA)
20 FORMAT(' ', 'L=',10I2/)
J6=1
MR=M31
IF(NR.EQ.1) GO TO 43
MR=M41
43 DO 16 JS=2,MR
IF(L(JS).EQ.0) GO TO 16
NN(J6)=L(JS)
J6=J6+1
16 CONTINUE
N1=NN(1)
N2=NN(2)
IF(N1.EQ.0) GO TO 5
IF(N2.EQ.0) GO TO 5
IF(NR.EQ.1) GO TO 35
MS=M3
GO TO 36

```

```

35 MS=M4
36 CALL POLIG(X,Y,Z,N,N1,N2,V3,MS,XT,YT,ZT,XI,YI,ZI,IN,HRP,IDP)
IF(TD.EQ.0) GO TO 5
IF(NRP.EQ.1) GO TO 61
NRF=IN-N
DO 304 C=1,NRF
IF(XI.LE.XI(IC)+0.01) GO TO 305
GO TO 304
305 IF(XI.GE.XI(IC)=0.01) GO TO 307
GO TO 304
307 IF(YI.LE.YI(IC)+0.01) GO TO 308
GO TO 304
308 IF(YI.GE.YI(IC)=0.01) GO TO 309
GO TO 304
309 IF(ZI.LE.ZI(IC)+0.01) GO TO 310
GO TO 304
310 IF(ZI.GE.ZI(IC)=0.01) GO TO 303
304 CONTINUE
303 IF(XI.LE.XI(IN-2)+0.01) GO TO 313
GO TO 61
313 IF(XI.GE.XI(IN-2)-0.01) GO TO 314
GO TO 61
314 IF(YI.LE.YI(IN-2)+0.01) GO TO 315
GO TO 61
315 IF(YI.GE.YI(IN-2)=0.01) GO TO 316
GO TO 61
316 IF(ZI.LE.ZI(IN-2)+0.01) GO TO 317
GO TO 61
317 IF(ZI.GE.ZI(IN-2)-0.01) GO TO 318
GO TO 61
318 CONT=CONT
GO TO 60
61 CONT=CONT+1
60 IF(HR.EQ.1) GO TO 21
IF(HR.EQ.2) GO TO 22
GO TO 5
22 B(CONT,I)=1
B(CONT,I+M)=1
A(CONT,K)=1
GO TO 44
21 A(CONT,I)=1
A(CONT,I+M)=1
B(CONT,K)=1
44 DO 31 I=1,CONT
WRITE(108,32) (B(II,JJ),JJ=1,20)
32 FORMAT(' ',16I3/)
31 CONTINUE
5 CONTINUE
M=M+1
WRITE(108,311) M,I
311 FORMAT(' ',M='I2',I='I2/')
IF(HR.EQ.1) GO TO 33
IM=I+M
WRITE(108,312) IM,M1,M2
312 FORMAT(' ',IM='I2',M1='I2',M2='I2/')
IF((I+M).LE.M2) GO TO 2
I=I+1
IF(I.NE.M2) GO TO 1
GO TO 34
33 IM=I+M
WRITE(108,312) IM,M1,M2
IF((I+M).LE.M1) GO TO 2
I=I+1
IF(I.NE.M1) GO TO 1
34 RETURN
END

```


3.6.6 Subprogramul LIMITE

```

SUBROUTINE LIMITE(X1,Y1,Z1,X2,Y2,Z2,XC,YC,ZC,XT,YT,ZT,IND)
  IF(X1.LE.X2) GO TO 1
  XSUP=X1+0.01
  XINF=X2-0.01
  GO TO 2
1  XINF=X1-0.01
  XSUP=X2+0.01
2  CONTINUE
  IF(Y1.LE.Y2) GO TO 3
  YINF=Y2-0.01
  YSUP=Y1+0.01
  GO TO 4
3  YINF=Y1-0.01
  YSUP=Y2+0.01
4  CONTINUE
  IF(Z1.LE.Z2) GO TO 5
  ZINF=Z2-0.01
  ZSUP=Z1+0.01
  GO TO 6
5  ZINF=Z1-0.01
  ZSUP=Z2+0.01
6  CONTINUE
  IF(XINF.LE.XC) GOTO 7
  GO TO 8
7  IF(XC.LE.XSUP) GO TO 9
  GO TO 8
9  XT=XC
  IF(YINF.LE.YC) GO TO 10
  GO TO 8
10 IF(YC.LE.YSUP) GO TO 11
  GO TO 8
11 YT=YC
  IF(ZINF.LE.ZC) GO TO 12
  GO TO 8
12 IF(ZC.LE.ZSUP) GO TO 13
  GO TO 8
13 ZT=ZC
  GO TO 14
8  XT=9999
  YT=9999
  ZT=9999
14 IF(XT.EQ.9999) GO TO 15
  IF(YT.EQ.9999) GO TO 15
  IF(ZT.EQ.9999) GO TO 15
  GO TO 16
15 IND=0.
  GO TO 17

16 IND=1.
17 RETURN
  END

```

Observație. Având în vedere că în cuprinsul lucrării există mai multe subprograme cu aceeași denumire LIMITE este necesar ca întotdeauna să nu se confunde introducerea lor în programele principale respective. Acest lucru este de altfel valabil și în cazul altor programe.

3.6.7 Subprogramul POLIG

```

SUBROUTINE POLIG(X,Y,Z,N,N1,N2,V3,MS,XT,YT,ZT,XI,YI,ZI,IN,NRP,IDP)
DIMENSION XI(20),YI(20),ZI(20)
DIMENSION X(30),Y(30),Z(30),V3(20),K(20)
DO 141 I=1,20
K(I)=0
141 CONTINUE
M4S=MS+1
I=0
DO 142 J=2,M4S
I=I+1
IF(V3(J).EQ.0) GO TO 142
K(I)=V3(J)
142 CONTINUE
WRITE(108,143) (K(J),J=1,I)
143 FORMAT(' ',20F10.3/)
RX=X(N1)
RY=Y(N1)
RZ=Z(N1)
PX=X(N2)
PY=Y(N2)
PZ=Z(N2)
144 WRITE(108,144) RX,RY,RZ,PX,PY,PZ
FORMAT(' ',6F10.3/)
K1=K(1)
K2=K(2)
K3=K(3)
CALL PARAM(X,Y,Z,K1,K2,K3,A1,B1,C1,D1)
A=A1
B=B1
C=C1
D=D1
X1=X(K1)
Y1=Y(K1)
Z1=Z(K1)
X2=X(K2)
Y2=Y(K2)
Z2=Z(K2)
X3=X(K3)
Y3=Y(K3)
Z3=Z(K3)
10 WRITE(108,10) X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3
FORMAT(' ',3F10.3/,' ',3F10.3/,' ',3F10.3/)
* CALL PCTINT(A,B,C,D,RX,RY,RZ,PX,PY,PZ,X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3,X
* T,YT,ZT,ID)
IF(ID.EQ.1) GO TO 4
M=I-1
IF(M.GT.2) GO TO 301
M=3
301 DO 5 J=3,M
K2=K(J)
K3=K(J+1)
IF(K3.NE.0) GO TO 300
300 K3=K(M)
X2=X(K2)
Y2=Y(K2)
Z2=Z(K2)
X3=X(K3)
Y3=Y(K3)
Z3=Z(K3)
WRITE(108,10) X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3
CALL PCTINT(A,B,C,D,RX,RY,RZ,PX,PY,PZ,X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3,X
* T,YT,ZT,ID)
IF(ID.EQ.1) GO TO 4
5 CONTINUE
WRITE(108,6)
6 FORMAT(' ',1,'PUNCTUL I NU APARTINE SUPRAFETEI POLIGONULUI'//)
IDP=0
GO TO 7
4 WRITE(108,8) XT,YT,ZT
8 FORMAT(' ',1,'PUNCTUL ESTE INTERIOR SUPRAFETEI POLIGONULUI'/' ',1,'COO
* RDONATELE PUNCTULUI DE INTERSECȚIE SINT',3F10.3//)
306 XI(IN)=XT
YI(IN)=YT
ZI(IN)=ZT
WRITE(108,302) XI(IN),YI(IN),ZI(IN),IN,NRP
302 FORMAT(' ',3F10.3/,' ',1,'IN,NRO',2I3/)
IN=IN+1
NRP=NRP-1
I=I-1
7 RETURN
END

```

3.6.8 Subprogramul PCTINT

```

SUBROUTINE PCTINT(A,B,C,D,RX,RY,RZ,PX,PY,PZ,X1,Y1,Z1,X2,Y2,Z2,X3,Y
*3,Z3,XT,YT,ZT, ID)
AL=RX-PX
AM=RY-PY
AN=RZ-PZ
R=A*AL+B*AM+C*AN
IF(R,NE,0) GO TO 81
ID=0
GO TO 4
81 AK=(A*RX+B*RY+C*RZ+D)/R
XC=RX-AL*AK
YC=RY-AM*AK
ZC=RZ-AN*AK
CALL LIMITE(RX,RY,RZ,PX,PY,PZ,XC,YC,ZC,XT,YT,ZT,IND)
IF(IND,EQ,0) GO TO 73
GO TO 74
73 ID=0
GO TO 4
74 XX=XT
YY=YT
ZZ=ZT
XG=(X1+X2+X3)/3.
YG=(Y1+Y2+Y3)/3.
ZG=(Z1+Z2+Z3)/3.
AL1=XX-XG
AM1=YY-YG
AN1=ZZ-ZG
AL12=X2-X1
AM12=Y2-Y1
AN12=Z2-Z1
AL13=X3-X1
AM13=Y3-Y1
AN13=Z3-Z1
AL23=X3-X2
AM23=Y3-Y2
AN23=Z3-Z2
RN1=AL12*AM1-AM12*AL1
IF(RN1,NE,0.0) GO TO 56
GO TO 55
56 RK1=((XX-X1)*AM1-AL1*(YY-Y1))/RN1
X12=RK1*AL12+X1
Y12=RK1*AM12+Y1
Z12=RK1*AN12+Z1
55 RN2=AL23*AM1-AM23*AL1
IF(RN2,NE,0.0) GO TO 57
GO TO 60
57 RK2=((XX-X2)*AM1-AL1*(YY-Y2))/RN2
X23=RK2*AL23+X2
Y23=RK2*AM23+Y2
Z23=RK2*AN23+Z2
60 RN3=AL13*AM1-AM13*AL1
IF(RN3,NE,0.0) GO TO 58
GO TO 62
58 RK3=((XX-X3)*AM1-AL1*(YY-Y3))/RN3
X13=RK3*AL13+X3
Y13=RK3*AM13+Y3
Z13=RK3*AN13+Z3
62 WRITE(108,200) RN1,RN2,RN3,X12,Y12,Z12,X13,Y13,Z13,X23,Y23,Z23
200 FORMAT('1',RN1,RN2,RN3',3F10.3/)
IF(RN1,EQ,0.0) GO TO 1
IF(RN2,EQ,0.0) GO TO 5
IF(RN3,EQ,0.0) GO TO 7
GO TO 81
7 XP=X12
YP=Y12
ZP=Z12
XU=X13
YU=Y13
ZU=Z13
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF(IND,EQ,0) GO TO 8
WRITE(108,140) XT,YT,ZT

```

```

ID=1
GO TO 4
8 ID=0
GO TO 4
5 IF (RN3.EQ.0.0) GO TO 6
XP=X3
YP=Y3
ZP=Z3
XU=X1
YU=Y1
ZU=Z1
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF (IND.EQ.0) GO TO 9
WRITE (108,140) XT,YT,ZT
ID=1
GO TO 4

9 ID=0
GO TO 4
6 WRITE (108,10)
10 FORNAT (' ', 'EROARE:RN=0,RN3=0'//)
GO TO 4
1 IF (RN2.EQ.0.0) GO TO 2
IF (RN3.EQ.0.0) GO TO 3
XP=X23
YP=Y23
ZP=Z23
XU=X13
YU=Y13
ZU=Z13
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF (IND.EQ.0) GO TO 11
WRITE (108,140) XT,YT,ZT
140 FORNAT (' ', 'COORDONATELE PUNCTULUI DE INTERSECȚIE',3F10,3//)
ID=1
GO TO 4
11 ID=0
GO TO 4
12 WRITE (108,12)
FORNAT (' ', 'EROARE:RN1=0,RN=0'//)
GO TO 4
12 WRITE (108,13)
FORNAT (' ', 'EROARE:RN1=0,RN=2'//).
61 XP=X1
YP=Y1
ZP=Z1
XU=X2
YU=Y2
ZU=Z2
XC=X12
YC=Y12
ZC=Z12
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
XB1=XT
YB1=YT
ZB1=ZT
XP=X1
YP=Y1
ZP=Z1
XU=X3
YU=Y3

```

```

ZU=Z3
XC=X13
YC=Y13
ZC=Z13
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
XB3=XT
YB3=YT
ZB3=ZT
XP=X2
YP=Y2
ZP=Z2
XU=X3
YU=Y3
ZU=Z3
XC=X3
YC=Y3
ZC=Z3
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
XB2=XT
YB2=YT
ZB2=ZT
IF (XB1.NE.9999) GO TO 40
XP=XB2
YP=YB2
ZP=ZB2
XU=XB3
YU=YB3
ZU=ZB3
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF (IND.EQ.0) GO TO 37
WRITE (108,140) XT,YT,ZT
ID=1
GO TO 4
37 ID=0
GO TO 36
40 IF (XB2.NE.9999) GO TO 32
XP=XB1
YP=YB1
ZP=ZB1
XU=XB3
YU=YB3
ZU=ZB3
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
ID=1
IF (IND.EQ.0) GO TO 30
WRITE (108,140) XT,YT,ZT
GO TO 4
30 ID=0
GO TO 36
32 IF (XB3.NE.9999) GO TO 50
XP=XB1
YP=YB1
ZP=ZB1
XU=XB2
YU=YB2
ZU=ZB2
XC=XX
YC=YY
ZC=ZZ
CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
IF (IND.EQ.0) GO TO 33
WRITE (108,140) XT,YT,ZT
ID=1
GO TO 4
33 ID=0
GO TO 36
50 IF (XB1.EQ.XB2.AND.YB1.EQ.YB2.AND.ZB1.EQ.ZB2) GO TO 105
IF (XB1.EQ.XB3.AND.YB1.EQ.YB3.AND.ZB1.EQ.ZB3) GO TO 107
IF (XB2.EQ.XB3.AND.YB2.EQ.YB3.AND.ZB2.EQ.ZB3) GO TO 107
WRITE (108,35)
ID=0
GO TO 36

```

```

107 XP=XB1
    YP=YB1
    ZP=ZB1
    XU=XB2
    YU=YB2
    ZU=ZB2
    XC=XX
    YC=YY
    ZC=ZZ
    CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
    IF(IND.EQ.0) GO TO 108
    ID=1
    WRITE(108,140) XT,YT,ZT
    GO TO 36
108 ID=0
    GO TO 36
105 XP=XB1
    YP=YB1
    ZP=ZB1
    XU=XB2
    YU=YB2
    ZU=ZB2
    XC=XX
    YC=YY
    ZC=ZZ
    CALL LIMITE(XP,YP,ZP,XU,YU,ZU,XC,YC,ZC,XT,YT,ZT,IND)
    IF(IND.EQ.0) GO TO 106
    WRITE(108,140) XT,YT,ZT
    GO TO 36
106 ID=0
    GO TO 36
35 FORMAT(' ','EROARE,DREAPTA IG TAIE TOATE LATURILE TRIUNGHIULUI',/)
36 CONTINUE
4 RETURN
END

```

3.6.9 Aplicație pentru programul INTPOL Intersecția dintre două cuburi P1 și P2

COORDONATELE VÂRFURILOR
POLIEDRELOR A ȘI B SÎNT URMĂTOARELE

					DEFINIȚIA FEȚELOR
1	67.000	62.000	100.000	A	
2	46.500	26.500	72.000	B	POLIEDRELOR
3	87.500	26.500	44.000	C	
4	107.000	62.000	72.000	D	1 1 2 3 4
5	87.500	97.500	44.000	E	2 5 6 7 8
6	46.500	97.500	72.000	F	3 1 4 5 6 P1
7	26.000	62.000	44.000	G	4 1 6 7 2
8	67.000	62.000	16.000	H	5 3 8 7 2
9	67.000	62.000	86.000	K	6 3 4 5 8
10	103.500	82.500	57.300	L	1 9 10 11 12
11	67.000	103.000	28.600	M	2 13 14 15 16
12	31.500	82.500	57.300	N	3 9 10 14 13 P2
13	67.000	21.000	57.300	O	4 10 14 15 11
14	103.500	41.500	28.600	P	5 11 12 16 15
15	67.000	62.000	.000	Q	6 9 12 16 13
16	31.500	41.500	28.600	R	

MATRICEA PUNCTELOR DE INTER-
SECȚIE PENTRU POLIEDRUL A ESTE

1	0	0	0	1	0
1	0	0	0	1	0
0	1	0	1	0	0
0	1	0	1	1	0
0	1	0	0	0	1
0	0	1	0	0	1
0	0	1	0	0	1
0	0	0	0	1	1

MATRICEA PUNCTELOR DE INTER-
SECȚIE PENTRU POLIEDRUL B ESTE

0	0	1	0	0	0
0	0	0	0	0	1
1	0	0	0	0	0
0	0	0	0	0	1
0	0	0	1	0	0
1	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0

0	0	1	0	0	0	1	0	0	0	0	0
0	1	0	0	0	1	1	0	0	1	0	0
0	0	0	1	0	0	1	0	0	0	0	1
0	0	0	0	1	0	0	1	1	0	0	0
0	0	0	0	0	1	0	1	1	0	0	0
1	0	0	0	0	0	0	0	1	0	0	1
0	1	0	0	0	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	0	0	1	1

DETERMINAREA ORDINII DE UNIRE
PENTRU PUNCTELE DE INTERSECȚIE
A DOUĂ POLIEDRE P1 ȘI P2

SE VOR UNI URMĂTOARELE PUNCTE:

PUNCTUL 1 CU PUNCTUL 16
PUNCTUL 1 CU PUNCTUL 14
PUNCTUL 2 CU PUNCTUL 16
PUNCTUL 2 CU PUNCTUL 18
PUNCTUL 3 CU PUNCTUL 11
PUNCTUL 3 CU PUNCTUL 13
PUNCTUL 4 CU PUNCTUL 17
PUNCTUL 4 CU PUNCTUL 13
PUNCTUL 5 CU PUNCTUL 17

PUNCTUL 5 CU PUNCTUL 18
PUNCTUL 6 CU PUNCTUL 11
PUNCTUL 6 CU PUNCTUL 12
PUNCTUL 7 CU PUNCTUL 10
PUNCTUL 7 CU PUNCTUL 12
PUNCTUL 8 CU PUNCTUL 10
PUNCTUL 8 CU PUNCTUL 15
PUNCTUL 9 CU PUNCTUL 14
PUNCTUL 9 CU PUNCTUL 15

Această numerotare a punctelor de intersecție este în ordinea găsirii punctelor prin program. După ordonare obținem punctele de intersecție în succesiunea necesară pentru poligonul de intersecție convenabilă echipamentului de trasare sub forma:

COORDONATELE VÎRFURILOR POLIGONULUI DE INTERSECȚIE

1	73.345	26.500	53.667	10	83.828	93.548	41.832
2	67.000	33.020	65.714	11	82.639	89.083	37.361
3	65.338	26.500	59.135	12	88.067	90.864	45.590
4	31.500	60.563	41.944	13	94.419	84.903	53.936
5	36.608	62.000	36.755	14	91.448	75.731	66.776
6	31.500	67.602	46.871	15	98.470	77.528	59.752
7	33.290	74.623	53.957	16	86.231	31.801	42.178
8	41.884	76.504	65.695	17	82.196	35.686	36.755
9	37.081	81.189	59.135	18	79.844	28.214	47.200

Această ordine a numerotării corespunde epurei din figura 3.10 pentru punctele poligonului strîmb de intersecție. Intersecția dintre cele două poliedre / cuburi / P1 și P2 este o rupere.

În figura 3.11 este dată o axonometrie izometrică a intersecției dintre cele două cuburi.

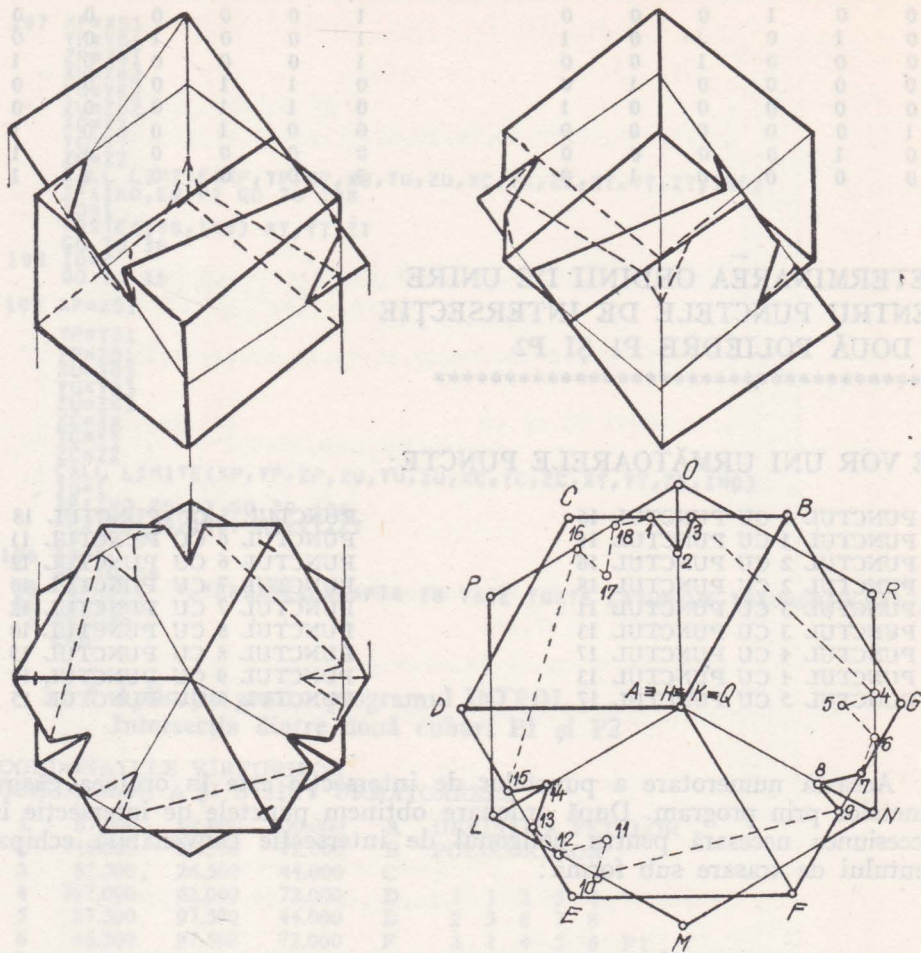


Fig. 3.10

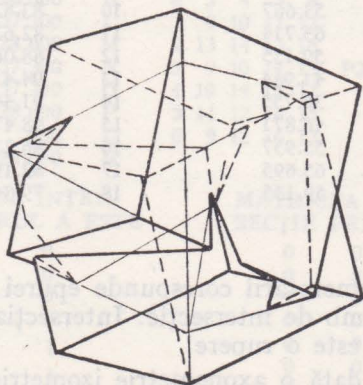


Fig. 3.11

**Conice. Cuadrice. Suprafețe de rotație.
Suprafețe de translație. Reprezentare.
Secțiuni plane. Intersecții mixte de
suprafețe.**

4.1 Secțiuni plane în sferă. Intersecția dintre două sfere.

4.1.1 Generalități. Baza matematică

Vom considera în triplă proiecție ortogonală secțiunea în sferă efectuată cu un plan oarecare.

Astfel fie planul de secțiune

$$Ax + By + Cz + SD = 0; \quad S = \pm 1$$

și sfera definită de centrul $\Omega(x_0, y_0, z_0)$ și raza R .

Centrul $E(E_1, E_2, E_3)$ al cercului de secțiune se obține din intersecția cu planul de secțiune a perpendicului dusă pe acest plan din centrul sferei.

Astfel fie

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = R^2$$

ecuația sferei și

$$\frac{x - x_0}{A} = \frac{y - y_0}{B} = \frac{z - z_0}{C} = RK$$

perpendiculara dusă din centrul sferei pe plan.

Intersecția perpendicului cu planul de secțiune este

$$\begin{cases} x = A \cdot RK + x_0 = E_1 \\ y = B \cdot RK + y_0 = E_2 \\ z = C \cdot RK + z_0 = E_3 \end{cases}$$

unde

$$RK = -\frac{Ax_0 + By_0 + Cz_0 + SD}{A^2 + B^2 + C^2}$$

Fie R_2 raza cercului de secțiune în sferă unde

$$R_2 = \sqrt{R^2 - \overline{\Omega E}^2}$$

Distanța $\overline{\Omega E}$ este

$$\overline{\Omega E} = \frac{Ax_0 + By_0 + Cz_0 + SD}{-S\sqrt{A^2 + B^2 + C^2}}$$

Dacă $\overline{\Omega E} < R$ Planul secționează sfera

Dacă $\overline{\Omega E} = R$ Planul este tangent sferei

Dacă $\overline{\Omega E} > R$ Eroare: Planul nu secționează sfera.

În aceste condițiuni cele trei elipse proiecții ale cercului de secțiune vor avea următoarele centre și semiaxe:

Elipsa proiecție orizontală are centrul $(E_1, E_2, 0)$

Semiaxa mare R_2

Semiaxa mică $R_2 \cos \varphi_1$

Elipsa proiecție verticală are centrul $(E_1, 0, E_3)$

Semiaxa mare R_2

Semiaxa mică $R_2 \cos \varphi_2$

Elipsa proiecție laterală are centrul $(0, E_2, E_3)$

Semiaxa mare R_2

Semiaxa mică $R_2 \cos \varphi_3$

Unghiurile diedre φ_1, φ_2 și φ_3 formate respectiv de planul de secțiune și planele de proiecție pot fi calculate cu următoarele relații:

$$\text{Pentru } z = 0 \quad \cos \varphi_1 = \frac{C}{-S\sqrt{A^2 + B^2 + C^2}}$$

$$\text{Pentru } y = 0 \quad \cos \varphi_2 = \frac{B}{-S\sqrt{A^2 + B^2 + C^2}}$$

$$\text{Pentru } x = 0 \quad \cos \varphi_3 = \frac{A}{-S\sqrt{A^2 + B^2 + C^2}}$$

4.1.2 Plotarea elipselor

Pentru plotarea elipselor este necesar să fie calculate unghiurile axelor mari ale elipselor cu axa absciselor ox :

$$BB = \arctg \left(-\frac{A_1}{B_1} \right)$$

$$BB = \arctg \left(-\frac{A_1}{C_1} \right)$$

$$BB = \arctg \left(-\frac{B_1}{C_1} \right) \text{ Instrucțiunea de apel este}$$

CALL ELIPSA (XC, YC, J, R1, R2, AA, DD, BB) unde (fig. 4.1a) (Instruirea face parte din (*Biblioteca Aristo*)
XC, YC Coordonatele centrului \bar{O} față de sistemul de axe xOy , în cm,
J = 3 Poziția peniței pe sus de la centrul \bar{O} la punctul de început i al trasării elipsei (Tipul deplasării)

R1, R2 În cm. semiaxele elipsei
AA 0° sau α° (vezi semnificația unghiului pe figură)

DD 360° pentru elipsa completă sau mai puțin pentru un arc de elipsă

BB Unghiul în grade zecimale, calculat cu relațiile date, dintre axa absciselor ox și axa $\bar{o}\bar{x}$ a elipsei după rotație și translație.

Așadar adaptarea instrucțiunilor pentru plotter la cele trei proiecții ortogonale pentru trasarea celor trei elipse proiectii este următoarea:

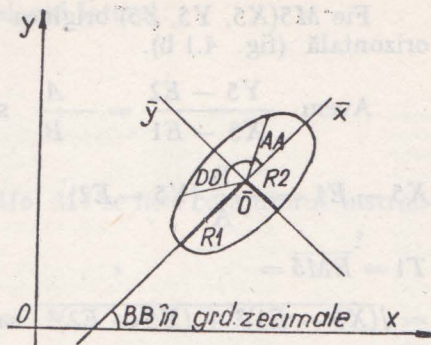


Fig. 4.1 a

$$\text{CALL ELIPSA} \left(E1, E2, 3, R2, R2 \cos \varphi_1, 0., 360., \text{arc tg} \left(-\frac{A_1}{B_1} \right) \right)$$

$$\text{CALL ELIPSA} \left(E1, E3, 3, R2, R2 \cos \varphi_2, 0., 360., \text{arc tg} \left(-\frac{A_1}{C_1} \right) \right)$$

$$\text{CALL ELIPSA} \left(E2, E3, 3, R2, R2 \cos \varphi_3, 0., 360., \text{arc tg} \left(-\frac{B_1}{C_1} \right) \right)$$

Centrul sferei în triplă proiecție ortogonală va fi plotat cu ajutorul subrutinei **REORPU** (vezi CAP. II).

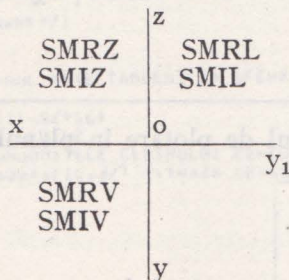
Cele trei contururi aparente ale sferei în triplă proiecție ortogonală vor fi plotate cu ajutorul instrucțiunilor (vezi CAP. I sau *Biblioteca ARISTO*):

CALL PLOT (X, Y, J)

CALL CIRCLE (X, Y, ARC)

În continuare este necesară determinarea virfurilor de plotare „i” pentru cele trei elipse proiectii.

Să considerăm următoarea notare a semiaxelor elipselor proiecții în program:



unde semiaxele mari $R2 = T1 = SMRZ = SMRV = SMRL$

Fie $M5(X5, Y5, Z5)$ originea vârfului de plotare „2” a elipsei în proiecția orizontală (fig. 4.1 b).

$$\text{Avem } \frac{Y5 - E2}{X5 - E1} = -\frac{A}{B} \quad \text{sau}$$

$$X5 - E1 = -\frac{B}{A} (Y5 - E2)$$

$$T1 = \overline{EM5} = \sqrt{(X5 - E1)^2 + (Y5 - E2)^2} \quad \text{sau}$$

$$T1 = \sqrt{(Y5 - E2)^2 \left(\frac{A^2 + B^2}{A^2} \right)}$$

$$Y5 - E2 = A \frac{T1}{\sqrt{A^2 + B^2}} = AV$$

unde am notat

$$V = \frac{T1}{\sqrt{A^2 + B^2}} \quad \text{deci prin analogie: } V1 = \frac{\text{SMRZ}}{\sqrt{A^2 + B^2}}$$

$$V2 = \frac{\text{SMRV}}{\sqrt{A^2 + B^2}}$$

$$V3 = \frac{\text{SMRL}}{\sqrt{A^2 + B^2}}$$

obținem, așa dar, coordonatele vârfului de plotare

$$X5 = E1 - BV$$

$$Y5 = E2 + AV$$

Particularizînd acest rezultat pentru fiecare vîrf de plotare obținem $M5(X5, Y5, Z5)$ vîrf de plotare în planul orizontal:

$$\begin{aligned} X5 &= E1 - B(I)V1 = E1 - B * \text{SMRZ} / \text{SQRT}(A * A + B * B) \\ Y5 &= E2 + A(I) * V1 \\ Z5 &= 0 \end{aligned}$$

$M6(X6, Y6, Z6)$ vîrf de plotare în planul vertical:

$$\begin{aligned} X6 &= E1 - C(I) * V2 \\ Y6 &= 0 \\ Z6 &= E3 + A(I) * V2 \end{aligned}$$

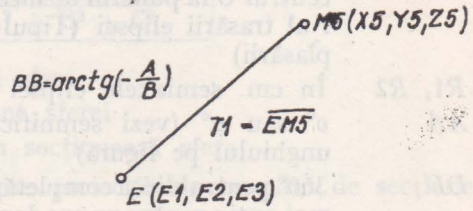


Fig. 4.1 b

$M7(X7, Y7, Z7)$ virful de plotare în planul lateral:

$$\begin{aligned} X7 &= 0 \\ Y7 &= E2 - C(I) * V3 \\ Z7 &= E3 + B(I) * V3 \end{aligned}$$

Evident accesul la aceste puncte $M5, M6, M7$ se face cu ajutorul instrucțiunii:

CALL PLOT(X,Y,J)

4.1.3 Programul SFERA

```

C SECTIUNE PLANA IN SFERA
C PARAMETRI PLANULUI DE SECTIUNE SINT A,B,C,D,S
C CENTRUL SFEREI ESTE X0,Y0,Z0 CU RAZA R
C CENTRUL CERCIULUI DE SECTIUNE ESTE E1,E2,E3
0001      DIMENSION A(100),B(100),C(100),D(100),X0(100),Y0(100),Z0(100),R(10)
          *C),S(100)
0002      CALL ASSIGN(1,'C0:')
0003      CALL ASSIGN(2,'LP:')
0004      CALL ASSIGN(3,'PP:')
0005      CALL INI(3)
0006      READ(1,3) N
0007      3 FORMAT(14)
0008      READ(1,1) (A(I),B(I),C(I),D(I),S(I),I=1,N)
0009      1 FORMAT(5F10.3,30X)
0010      READ(1,2) (X0(I),Y0(I),Z0(I),R(I),I=1,N)
0011      2 FORMAT(4F10.3,40X)
0012      DO 7 I=1,N
0013          WRITE(2,102) A(I),B(I),C(I),D(I)
0014      102 FORMAT(' ',COEFICIENTII PLANULUI,'/ ',A='F7.2,2X,
          *B='F7.2,2X,*C='F7.2,2X,*D='F10.3/)
0015          WRITE(2,103) X0(I),Y0(I),Z0(I),R(I)
0016      103 FORMAT(' ',CENTRUL SFEREI',ZX,'X0('F7.2,2X,*YU='F7.2,2X,' 0='
          *F7.2/' ',RAZA SFEREI'F7.2/)
0017          WRITE(2,100)
0018      100 FORMAT(//,30(1X,*)//)
0019          WRITE(2,101) I
0020      101 FORMAT(' ',SITUATIA NR:',I2//)
0021          RNUM1=A(I)*X0(I)+B(I)*Y0(I)+C(I)*Z0(I)+S(I)*D(I)
0022          KNUM2=A(I)**2+B(I)**2+C(I)**2
0023          KR=-RNUM1/RNUM2
0024          E1=A(I)*KR+X0(I)
0025          E2=B(I)*KR+Y0(I)
0026          E3=C(I)*KR+Z0(I)
0027          RNUM3=A(I)*X0(I)+A(I)*Y0(I)+C(I)*Z0(I)+S(I)*D(I)
0028          KNUM4=S(I)*SQRT(A(I)**2+B(I)**2+C(I)**2)
0029          SE=-RNUM3/KNUM4
0030          IF(SE.GE.R(I)) GO TO 9
0031          IF(SE.EW.R(I)) GO TO 4
0032      GO TO 4
0033      9 WRITE(2,6)
0034      6 FORMAT(' ',EPOURF//)
0035      GO TO 7
0036      5 WRITE(2,8)
0037      8 FORMAT(' ',PLANUL ESTE TANGENT LA SFERA//)
0038      GO TO 7
0039      4 RZ=SQRT(R(I)**2-SE*SE)
0040          WRITE(2,10)E1,E2,F3,RZ
0041      10 FORMAT(' ',COORDONATELE CENTRULUI CERCIULUI,' ',E1='F10.3,2X,'E
          I2='F10.3,2X,*E3='F10.3/' ',RAZA CERCIULUI DE SECTIUNE,'/ ',R='
          2F10.3//)
0042          S1=D(I)/A(I)
0043          S2=D(I)/B(I)
0044          S3=D(I)/C(I)
0045          XURZ=E1
0046          YURZ=E2
0047          ZURZ=0

```

```

0048 SMRZ=H2
0049 COS1=-C(I)/(S(I)*SQRT(A(I)**2+B(I)**2+C(I)**2))
0050 SMIZ=R2*COS1
0051 ARG1=-A(I)/B(I)
0052 BBZ=ATAN(ARG1)
0053 XVER=E1
0054 YVER=0
0055 ZVER=E3
0056 SMRV=H2
0057 COS2=-B(I)/(S(I)*SQRT(A(I)**2+B(I)**2+C(I)**2))
0058 SMIV=R2*COS2
0059 ARG2=A(I)/C(I)
0060 BBV=ATAN(ARG2)
0061 XLAT=0
0062 YLAT=E2
0063 ZLAT=E3
0064 SMRL=R2
0065 COS3=-A(I)/(S(I)*SQRT(A(I)**2+B(I)**2+C(I)**2))
0066 SMIL=R2*COS3
0067 ARG3=-B(I)/C(I)
0068 BBL=ATAN(ARG3)
0069 WRITE(2,11)SMRZ,SMI17

0070 11 FORMAT(' ',SEMIAXA MARE A ELIPSEI IN PROIECTIE ORIZONTALA='F10.3/
1/ ' ',SEMIAXA MICA A ELIPSEI IN PROIECTIE ORIZONTALA='F10.3/)
0071 WRITE(2,12)SMRV,SMIV
0072 12 FORMAT(' ',SEMIAXA MARE A ELIPSEI IN PROIECTIE VERTICALA='F10.3/
1/ ' ',SEMIAXA MICA A ELIPSEI IN PROIECTIE VERTICALA='F10.3/)
0073 WRITE(2,13)SMRL,SMIL
0074 13 FORMAT(' ',SEMIAXA MARE A ELIPSEI IN PROIECTIE LATERALA='F10.3/
1/ ' ',SEMIAXA MICA A ELIPSEI IN PROIECTIE LATERALA='F10.3////)
0075 UBZ=(360*RBZ)/6.28
0076 UBV=(360*RBV)/6.28
0077 UBL=(360*RBL)/6.28
0078 WRITE(2,14)UBZ,UHV,UBL
0079 14 FORMAT(' ',PANTA SEMIAXEI MARI IN PROIECTIE ORIZONTALA='F10.3/
' ',PANTA SEMIAXEI MARI IN PROIECTIE VERTICALA='F10.3/ ' ',PANTA S
*EMIAXEI MARI IN PROIECTIE LATERALA='F10.3//)
C CALCULUL COORDONATELOR CENTRELOR DE PLUATARE M5,M6,M7
0080 V1=SMRZ/SQRT(A(I)**2+B(I)**2)
0081 V2=SMRV/SQRT(A(I)**2+C(I)**2)
0082 V3=SMRL/SQRT(B(I)**2+C(I)**2)
0083 X5=E1-B(I)*V1
0084 Y5=E2+A(I)*V1
0085 Z5=0
0086 X6=E1-C(I)*V2
0087 Y6=0
0088 Z6=E3+A(I)*V2
0089 X7=0
0090 Y7=E2-C(I)*V3
0091 Z7=E3+B(I)*V3
0092 WRITE(2,15)X5,Y5,Z5,X6,Y6,Z6,X7,Y7,Z7
0093 15 FORMAT(' ',COORDONATELE CELOR 3 ORIGINI PENTRU PLUATARE// ' ',X5='
1*F10.3,2X,'Y5='F10.3,2X,'Z5='F10.3/ ' ',X6='F10.3,2X,'Y6='F10.
23,2X,'Z6='F10.3/ ' ',X7='F10.3,2X,'Y7='F10.3,2X,'Z7='F10.3//)
0094 CALL CIS(E2,E3,0.5)
0095 CALL TEX(F2+1.,E3+1.,0.,0.,0.,E3,2)

0096 CALL CIS(-E1,E3,0.5)
0097 CALL TEX(-E1+1.,E3+1.,0.,0.,0.,E2,2)
0098 CALL CIS(-E1,-E2,0.5)
0099 CALL TEX(-E1+1.,-E2+1.,0.,0.,0.,E1,2)
0100 CALL TEX(Y0(1)+1.,Z0(1)+1.,0.,0.,0.,03,2)
0101 CALL CIS(Y0(1),Z0(1),0.5)
0102 CALL CIS(Y0(1),Z0(1),50.)
0103 CALL TEX(-X0(1)+1.,Z0(1)+1.,0.,0.,0.,02,2)
0104 CALL CIS(-X0(1),Z0(1),0.5)
0105 CALL CIS(-X0(1),Z0(1),50.)

```

```

0106      CALL TEX(-X0(1)+J.,-Y0(1)+1..0.,4..0,'01',2)
0107      CALL CIS(-X0(1),-Y0(1),0.5)
0108      CALL CIS(-X0(1),-Y0(1),50.)
0109      CALL TEX(-140.,2..0.,4..0,'X',1)
0110      CALL LIN(-140..0..250..0.)
0111      CALL TEX(250.,2..0.,4..0,'Y1',2)
0112      CALL TEX(2.,-250..0.,4..0,'Y',1)
0113      CALL LIN(0.,-250..0..150.)
0114      CALL TEX(2.,150..0.,4..0,'Z',1)
0115      CALL PLOT(0.,-S2,0)
0116      CALL TEX(1.,-S2,0.,4..0,'S2',2)
0117      CALL PLOT(-S1,0.,1)
0118      CALL TEX(-S1,1.,0.,4..0,'S1',2)
0119      CALL PLOT(0.,S3,1)
0120      CALL TEX(1.,S3,0.,4..0,'S3',2)
0121      CALL PLOT(S2,0.,1)
0122      CALL TEX(S2,1.,0.,4..0,'S2',2)
0123      CALL TEX(20.,-40..0.,8..0,'SUBROUTINE SFERA',16)
0124      CALL TEX(20.,-60..0.,6..0,'SECTIUNE PLANA IN SFERA',23)
0125      CALL CIS(S2,0.,.5)
0126      CALL CIS(0.,-S2,.5)
0127      CALL CIS(0.,S3,.5)
0128      CALL CIS(-S1,0.,.5)
0129      CALL TRA(E2,E3)
0130      CALL DEG
0131      CALL ANG(UBL)
0132      CALL DESEN(SMRL,SMIL)
0133      CALL TRA(-E2,-E3)
0134      CALL TRA(-E1,E3)
0135      CALL DEG
0136      CALL ANG(UBV)
0137      CALL DESEN(SMRV,SMIV)
0138      CALL TRA(E1,-E3)
0139      CALL TRA(-E1,-E2)
0140      CALL DEG
0141      CALL ANG(UBZ)
0142      CALL DESEN(SMRZ,SMIZ)
0143      CALL TRA(E1,E2)
0144      7 CONTINUE
0145      STOP
0146      END

0001      SUBROUTINE DESEN(A,B)
0002      DIMENSION X(40),Y(40)
0003      N=37
0004      PI=3.1416
0005      DU=2.*PI/36
0006      U=0.
0007      DO 10 I=1,N
0008      X(I)=A*COS(U)
0009      Y(I)=B*SIN(U)
0010      10 U=U+DU
0011      N1=N-1
0012      CALL PLOT(X(1),Y(1),0)
0013      DO 11 I=1,N1
0014      CALL PLOT(X(I+1),Y(I+1),1)
0015      11 CONTINUE
0016      RETURN
0017      END

```

COEFICIENTII PLANULUI

A= 336.00 B= 182.00 C= 312.00 D= 43680.000

CENTRUL SFEREI X0= 70.00 Y0= 80.00 Z0= 70.00

RAZA SFEREI 50.00

SITUATIA NR: 1

COORDONATELE CENTRULUI CERCLUI

E1= 47.578 E2= 67.855 F3= 49.180

RAZA CERCLUI DE SECTIUNE

R= 37.634

SEMIAXA MARE A ELIPSEI IN PROIECTIE ORIZONTALA= 37.634

SEMIAXA MICA A ELIPSEI IN PROIECTIE ORIZONTALA= 23.801

SEMIAXA MARE A ELIPSEI IN PROIECTIE VERTICALA= 37.634

SEMIAXA MICA A ELIPSEI IN PROIECTIE VERTICALA= 13.884

SEMIAXA MARE A ELIPSEI IN PROIECTIE LATERALA= 37.634

SEMIAXA MICA A ELIPSEI IN PROIECTIE LATERALA= 25.632

PANTA SEMIAXEI MARI IN PROIECTIF ORIZONTALA= -61.588

PANTA SEMIAXEI MARI IN PROIECTIE VERTICALA= 47.145

PANTA SEMIAXEI MARI IN PROIECTIF LATERALA= -30.272

COORDONATELE CELOR 3 ORIGINI PENTRU PLOTARE

X5= 29.654 Y5= 100.946 Z5= 0.000

X6= 21.970 Y6= 0.000 Z6= 70.757

X7= 0.000 Y7= 35.348 Z7= 68.142

```

0001 PROGRAM ELIPSA
0002 DIMENSION X(40),Y(40)
0003 CALL ASSIGN(1,'CR:')
0004 CALL ASSIGN(2,'LP:')
0005 CALL ASSIGN(3,'PP:')
0006 CALL INI(3)
0007 READ(1,4) A,B
0008 4 FORMAT(2F5.2,70X)
0009 CALL LIN(-100.,0.,100.,0.)
0010 CALL LIN(0.,-100.,0.,100)
0011 CALL DESEN(A,B)
0012 CALL TRA(20.,20.)
0013 CALL DEG
0014 CALL ROT(30.)
0015 CALL DESEN(A,B)
0016 STOP
0017 END

```

Programul principal ELIPSA

```

PROGRAM ELIPSA
DIMENSION X(40),Y(40)
CALL ASSIGN(1,'CR:')
CALL ASSIGN(2,'LP:')
CALL ASSIGN(3,'PP:')
CALL INI(3)
READ(1,4) A,B
4 FORMAT(2F5.2,70X)
PI=3.1416
N=36
DU=2.*PI/36
U=0.
DO 10 I=1,N
X(I)=A*COS(U)
Y(I)=B*XIN(U)
10 U=U+DU
N1=N-1
CALL PLOT(X(1),Y(1),0)
DO 11 I=1,N1
CALL PLOT(X(I+1),Y(I+1),1)
11 CONTINUE
CALL LIN(-100.,0.,100.,0.)
CALL LIN(0.,-100.,0.,100)
STOP
END

```

Subprogramul DESEN pentru desenarea elipsei

```

0001 SUBROUTINE DESEN(A,B)
0002 DIMENSION X(40),Y(40)
0003 N=37
0004 PI=3.1416
0005 DU=2.*PI/36
0006 U=0.
0007 DO 10 I=1,N
0008 X(I)=A*COS(U)
0009 Y(I)=B*SIN(U)
0010 10 U=U+DU
0011 N1=N-1
0012 CALL PLOT(X(1),Y(1),0)
0013 DO 11 I=1,N1
0014 CALL PLOT(X(I+1),Y(I+1),1)
0015 11 CONTINUE
0016 RETURN
0017 END

```


4.1.4 Intersecția dintre două sfere. Programul principal.

```

PROGRAM IN2SF
C INTERSECȚIA DINTRE DOUA SFERE
C PARAMETRI PLANULUI RADICAL DE SECȚIUNE SIN A,B,C,D
C CENTRUL PRIMEI SFERE ESTE X0,Y0,Z0 CU RAZA R
C CENTRUL SFEREI A DOUA ESTE X1,Y1,Z1 CU RAZA R1
C CENTRUL CERCULUI DE SECȚIUNE ESTE CU PLANUL RADICAL ESTE E1,E2,E3
  CALL ASSIGN(1,'CR:')
  CALL ASSIGN(2,'LP:')
  CALL ASSIGN(3,'PP:')
  CALL INI(3)
  ACCEPT 2, X0,Y0,Z0,R
  ACCEPT 2, X1,Y1,Z1,R1
  2 FORMAT(4F10.3,40X)
  A=-2*(X1-X0)
  B=-2*(Y1-Y0)
  C=-2*(Z1-Z0)
  D=-R1**2+R**2-X0**2-Y0**2-Z0**2+(X1**2+Y1**2+Z1**2)
  TYPE 102, A,B,C,D
102 FORMAT(' ', 'COEFICIENTII PLANULUI', /' ', 'A=', F7.2, '2X', 'B=', F7.2, '2X',
* 'C=', F7.2, '2X', 'D=', F10.3 /)
  KNUM1=A*X0+B*Y0+C*Z0+D
  RNUM2=A**2+B**2+C**2
  RK=-RNUM1/RNUM2
  E1=A*RK+X0
  E2=B*RK+Y0
  E3=C*RK+Z0
  KNUM3=A*X0+B*Y0+C*Z0+D
  RNUM4=SQRT(A**2+B**2+C**2)
  SE=-RNUM3/RNUM4
  IF(SE.GE.R) GO TO 9
  IF(SE.LE.-R) GO TO 5
  GO TO 4
  9 TYPE 6
  6 FORMAT(' ', 'ERORARE /')
  GO TO 7
  5 TYPE 8
  8 FORMAT(' ', 'PLANUL ESTE TANGENT LA SFERA /')
  GO TO 7
  4 R2=SQRT(R**2-SE*SE)
  TYPE 10, E1,E2,E3,R2
10 FORMAT(' ', 'COORDONATELE CENTRULUI CERCULUI', /' ', 'E1,E2,E3', '2X',
* '3F10.3', /' ', 'R2=', F10.3 /)
  S1=-D/A
  S2=-D/B
  S3=-D/C
  XOKZ=E1
  YOKZ=E2
  ZOKZ=0
  SMKZ=R2
  COS1=-C/(SQRT(A**2+B**2+C**2))
  SMIZ=R2*COS1
  ARG1=-A/J
  EBZ=ATAN(ARG1)
  XVER=E1
  YVER=0
  ZVER=E3

```

```

SMKV=RL
COS2=-0.7/(SQRT(A**2+B**2+C**2))
SMIV=R2*COS2
ARG2=A/C
BBV=ATAN(ARG2)
XLAT=0
YLAT=FZ
SMKL=RZ
COS3=-A7/(SQRT(A**2+B**2+C**2))
SMIL=R2*COS3
ARG3=-B/C
BBL=ATAN(ARG3)
UBZ=(360*BBZ)/0.28
UBV=(360*BBV)/0.28
URL=(360*BBL)/0.28
TYPE 14, UBZ,UBV,URL
FORMAT(' ', 'PANTELE SERIAXEI MARI IN CELE TREI PROIECTII', ' ', ' ', 'UBZ
,UBV', '3F10.3')
CALL CIS(E2,E3,0.5)
CALL TEX(E2+1.,E3+1.,0.,4.,0,'E3',2)
CALL CIS(-E1,E3,0.5)
CALL TEX(-E1+1.,E3+1.,0.,4.,0,'E2',2)
CALL CIS(-E1,-E2,0.5)
CALL TEX(-E1+1.,-E2+1.,0.,4.,0,'E1',2)

CALL TEX(Y0+1.,Z0+1.,0.,4.,0,'U13',3)
CALL CIS(Y0,Z0,0.5)
CALL CIS(Y0,Z0,R)
CALL TEX(-X0+1.,Z0+1.,0.,4.,0,'O12',3)
CALL CIS(-X0,Z0,0.5)
CALL CIS(-X0,Z0,R)
CALL TEX(-X0+1.,-Y0+1.,0.,4.,0,'U11',3)
CALL TEX(Y1+1.,Z1+1.,0.,4.,0,'U23',3)
CALL CIS(-X1,Z1,0.5)
CALL CIS(-X1,Z1,R1)
CALL TEX(-X1+1.,Z1+1.,0.,4.,0,'O22',3)
CALL CIS(Y1,Z1,0.5)
CALL CIS(Y1,Z1,R1)
CALL TEX(-X1+1.,-Y1+1.,0.,4.,0,'U21',3)
CALL CIS(-X1,-Y1,0.5)
CALL CIS(-X1,-Y1,R1)
CALL CIS(-X0,-Y0,0.5)
CALL CIS(-X0,-Y0,R)
CALL TEX(-100.,2.,0.,4.,0,'X',1)
CALL LIN(-100.,0.,150.,0.)
CALL TEX(150.,2.,0.,4.,0,'Y1',2)
CALL TEX(2.,-150.,0.,4.,0,'Y',1)
CALL LIN(0.,-150.,0.,150.)
CALL TEX(2.,150.,0.,4.,0,'Z',1)
CALL PLOT(0.,-32,0)
CALL TEX(1.,-32,0.,4.,0,'S2',2)
CALL PLOT(-31,0.,1)
CALL TEX(-31,1.,0.,4.,0,'S1',2)
CALL PLOT(0.,33,1)
CALL TEX(1.,33,0.,4.,0,'S3',2)
CALL PLOT(32,0.,1)

```

```

CALL TEX(S2,1.,0.,4.,0,'S2',2)
CALL TEX(20.,-40.,0.,9.,0,'SUBROUTINE IN2SF',10)
CALL TEX(20.,-60.,0.,6.,0,'INTERSECȚIA DINTRE DOUA SFERE',29)
CALL CIS(S2,0.,.5)
CALL CIS(0.,-S2,.5)
CALL CIS(0.,S3,.5)
CALL CIS(-S1,0.,.5)
CALL TRA(E2,E3)
CALL DEG
CALL ANG(UBL)
CALL DESEN(SMRL,SMIL)
CALL TRA(-E2,-E3)
CALL TRA(-E1,E3)
CALL DEG
CALL ANG(UBV)
CALL DESEN(SMRY,SMIV)
CALL TRA(E1,-E3)
CALL TRA(-E1,-E2)
CALL DEG
CALL ANG(UBZ)
CALL DESEN(SMRZ,SMIZ)
CALL TRA(E1,L2)
7 STOP
END

```

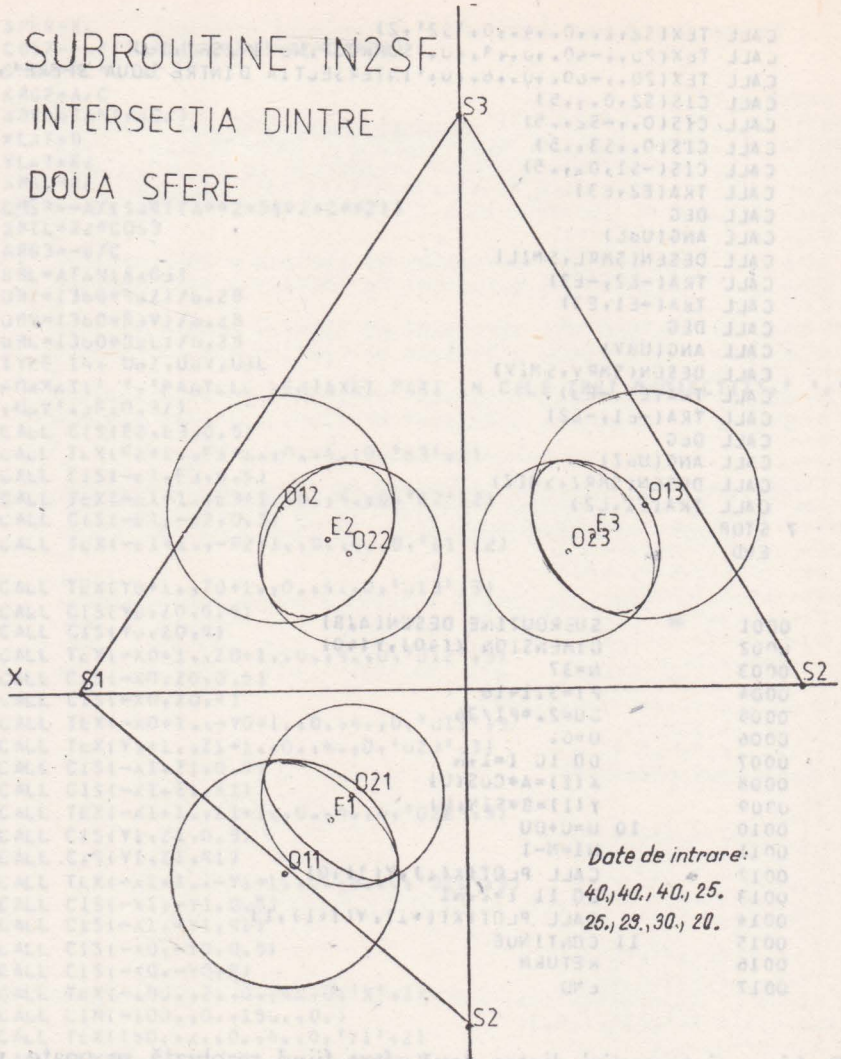
```

0001 * SUBROUTINE DESEN(A,B)
0002 DIMENSION X(40),Y(40)
0003 N=37
0004 PI=3.1416
0005 DU=2.*PI/36
0006 U=0.
0007 DO 10 I=1,N
0008 X(I)=A*COS(U)
0009 Y(I)=B*SIN(U)
0010 10 U=U+DU
0011 N1=N-1
0012 CALL PLOT(X(1),Y(1),0)
0013 DO 11 I=1,N1
0014 CALL PLOT(X(I+1),Y(I+1),1)
0015 11 CONTINUE
0016 RETURN
0017 END

```

Problema intersecției dintre două sfere fiind rezolvată se poate pune problema mai departe, de a combina această intersecție între mai multe sfere, în anumite condiții. Astfel, acest program principal IN2SF va deveni un subprogram într-un program mai general determinat de anumite rațiuni geometrice, impuse de rezolvarea unor probleme puse de practică sau de matematică. Grafica pe calculator va consta și ea din desenarea sau vizualizarea diferitelor succesiuni de elipse sau a rezultatului final.

SUBROUTINE IN2SF
INTERSECȚIA DINTRE
DOUA SFERE



Date de intrare:
40, 40, 40, 25.
25, 29, 30, 20.

Fig. 4.2 a

Problema intersecției dintre două sfere fiind rezolvată se poate pune problema mai departe de a construi intersecția între mai multe sfere. În anumite condiții, astfel, acest program principal IN2SF va deveni un subprogram într-un program mai general de calcul al intersecțiilor matematice. Datele pe calculator se conțin și în documentul din anexa la această lucrare.

4.1.5 Reprezentarea sferei în izometrie.

```

PROGRAM SFERIZ
C REPREZENTAREA SFEREI IN IZOMETRIE CU CONSTRUCTIA
C CERCURILOR PARALEL SI MERIDIAN
C
  DIMENSION XT(80),YT(80),ZT(80),XS(80),YS(80),ZS(80),XR(80),YR(80),
  *ZR(80)
  COMMON /BLUCD/AL,BE,GA,X2,Y2,Z2,AL1,BE1,GA1,
  *AL2,BE2,GA2,A,B,C,D,XD,YD
  X2=0.
  Y2=0.
  Z2=0.
  AL1=0.707
  BE1=-AL1
  GA1=0.
  AL2=-0.408
  BE2=AL2
  GA2=0.82
  AL=0.578
  BE=AL
  GA=AL
  A=0.002
  B=A
  C=A
  D=-10.
  CALL ASSIGN(1,'CR:')
  CALL INI(3)
  ACCEPT 2,XC,YC,ZC,R
2  FCRMAT(4F10.3,40X)
  PAS=5.
  ALFA=0
  ALF1=0
  I=1
5  XT(I)=XC+R*COS(ALFA)
  YT(I)=YC+R*SIN(ALFA)
  ZT(I)=ZC
  XS(I)=XT(I)
  YS(I)=YC
  ZS(I)=ZC+R*SIN(ALFA)
  XR(I)=XC
  YR(I)=YT(I)
  ZR(I)=ZC+R*COS(ALFA)
  I=I+1
  ALF1=ALF1+PAS
  ALFA=ALF1*3.1459/180.
  IF(ALF1.LE.300.) GO TO 5
  DO 20 I=1,73
  XK=XT(I)
  YK=YT(I)
  ZK=ZT(I)
  CALL PRPLOT(I,XK,YK,ZK)
20 CONTINUE
  DO 21 I=1,73
  XK=XS(I)
  YK=YS(I)
  ZK=ZS(I)
  CALL PRPLOT(I,XK,YK,ZK)
  SUBROUTINE PRPLOT(I,X,Y,Z)
  COMMON /BLUCD/AL,BE,GA,X2,Y2,Z2,AL1,BE1,GA1,
  *AL2,BE2,GA2,A,B,C,D,X1,Y1
  LAMBDA=(A*X+B*Y+C*Z+D)/(A*AL+B*BE+C*GA)
  XP=X-AL*LAMBDA
  YP=Y-BE*LAMBDA
  ZP=Z-GA*LAMBDA
  X1=AL1*(XP-X2)+BE1*(YP-Y2)+GA1*(ZP-Z2)
  Y1=AL2*(XP-X2)+BE2*(YP-Y2)+GA2*(ZP-Z2)
  IF(I.EQ.1) CALL PLOT(X1,Y1,0)
  CALL PLOT(X1,Y1,1)
  RETURN
  END
  21 CONTINUE
  DO 22 I=1,73
  XK=XR(I)
  YK=YR(I)
  ZK=ZR(I)
  CALL PRPLOT(I,XK,YK,ZK)
22 CONTINUE
  CALL DEG
  CALL PRPLOT(1,XC,YC,ZC)
  CALL CIS(XD,YD,R)
  CALL PRPLOT(1,0.,0.,0.)
  CALL CIS(XD,YD,0.5)
  CALL LIN(XD,YD,XD,YC+100.)
  CALL LIN(XD,YD,XD-86.5,YD-50.)
  CALL LIN(XD,YD,XD+86.5,YD-50.)
  CALL EOF
  STOP
  END

```

Aplicații la programul SFERIZ

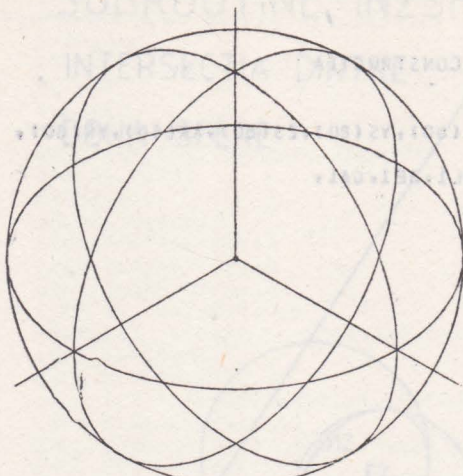


Fig. 4.2 c

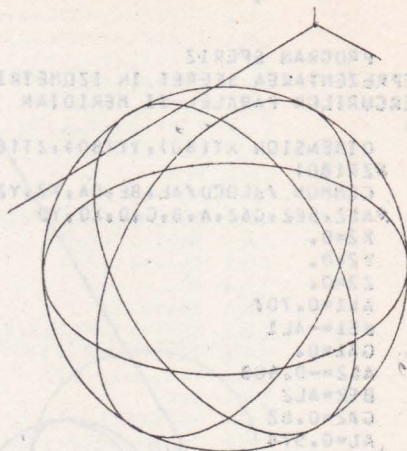


Fig 4.2 b

4.2. Secțiuni plane în suprafețele conice

4.2.1 Secțiunea eliptică în conul circular oblic.

Vom considera, în primul rând, secțiunea în conul circular oblic care nu conține ramuri infinite, deci secțiunea eliptică, pe care o vom determina prin semiaxe.

Astfel fie planul de secțiune

$$Ax + By + Cz + SD = 0; \quad S = \pm 1$$

iar

$$S(x_0, y_0, z_0) \text{ vârful conului.}$$

Directoarea conului este cercul din planul orizontal de proiecție cu centrul $M(x_1, y_1, z_1)$ și raza R . Se transformă planul de secțiune în plan de capăt.

Condiția pentru definirea elipsei de secțiune prin semiaxe se subînțelege:

$$Bx_0 - Ay_0 = 0 \quad (\text{fig. 4.3})$$

Intersecția generatoarelor SM_2 și SM_3 cu planul de secțiune conduce la punctele M_4 și M_5 . Deci

$$\overline{SM}_2 \cap P = M_4(x_4, y_4, z_4)$$

$$\overline{SM}_3 \cap P = M_5(x_5, y_5, z_5)$$

Punctele $M_2(x_2, y_2, z_2)$ și $M_3(x_3, y_3, z_3)$ sînt urmele horizontale ale generatoarelor SM_2 și SM_3 și se obțin din rezolvarea sistemului format de ecuația cercului director și dreapta S_0M_1 din planul horizontal:

$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 = R^2 \\ y - y_1 = \frac{y_0 - y_1}{x_0 - x_1} (x - x_1) \end{cases}$$

Notăm panta dreptei:

$$S_1 = \frac{y_0 - y_1}{x_0 - x_1}$$

Avem

$$(x - x_1)^2 + S_1^2 (x - x_1)^2 = R^2$$

$$(x - x_1)^2 (1 + S_1^2) = R^2$$

obținem

$$M_2 \begin{cases} x_2 = x_1 - \frac{R}{\sqrt{1 + S_1^2}} \\ y_2 = y_1 - S_1(x_2 - x_1) \end{cases}$$

$$M_3 \begin{cases} x_3 = x_1 + \frac{R}{\sqrt{1 + S_1^2}} \\ y_3 = y_1 + S_1(x_3 - x_1) \end{cases}$$

Determinarea coordonatelor punctelor M_4 și M_5 se face rezolvînd ecuațiile:

$$(SM_2) \frac{x - x_2}{x_0 - x_2} = \frac{y - y_2}{y_0 - y_2} = \frac{z - z_2}{z_0 - z_2} = S_3$$

$$S_3 = - \frac{Ax_2 + By_2 + Cz_2 + D}{A(x_0 - x_2) + B(y_0 - y_2) + C(z_0 - z_2)}$$

Deci

$$\left. \begin{aligned} x = x_4 &= S_3(x_0 - x_2) + x_2 \\ y = y_4 &= S_3(y_0 - y_2) + y_2 \\ z = z_4 &= S_3(z_0 - z_2) + z_2 \end{aligned} \right\} M_4(x_4, y_4, z_4)$$

Analog rezolvînd sistemul:

$$Ax + By + Cz + SD = 0$$

$$(SM_3) \frac{x - x_3}{x_0 - x_3} = \frac{y - y_3}{y_0 - y_3} = \frac{z - z_3}{z_0 - z_3} = S_4$$

obținem:

$$\left. \begin{aligned} x_5 &= S_4(x_0 - x_3) + x_3 \\ y_5 &= S_4(y_0 - y_3) + y_3 \\ z_5 &= S_4(z_0 - z_3) + z_3 \end{aligned} \right\} M_5(x_5, y_5, z_5)$$

Centrul elipsei de secțiune în proiecția triplu ortogonală este:

$$M_6 \begin{cases} x_6 = \frac{x_4 + x_5}{2} \\ y_6 = \frac{y_4 + y_5}{2} \\ z_6 = \frac{z_4 + z_5}{2} \end{cases}$$

deci centrul axei mici spațiale $\overline{M}_4 \overline{M}_5$.

Cercul de nivel care definește axa mare a elipsei în spațiu și în proiecția orizontală are centrul M_7 unde $M_7(x_7, y_7, z_6)$ deoarece $Z_7 = Z_6$ al punctului M_6 . Raza acestui cerc este R_1 .

Astfel

$$(SM_1) \frac{x - x_1}{x_0 - x_1} = \frac{y - y_1}{y_0 - y_1} = \frac{z - z_1}{z_0 - z_1} = RK$$

$$Z = Z_6$$

Rezultă:

$$\begin{aligned} x &= x_7 = RK(x_0 - x_1) + x_1 \\ y &= y_7 = RK(y_0 - y_1) + y_1 \\ z &= z_6 = RK(z_0 - z_1) + z_1 \end{aligned} \quad \text{unde } RK = \frac{z_6 - z_1}{z_0 - z_1}$$

sau

$$\left. \begin{aligned} x_7 &= x_1 + RK(x_0 - x_1) \\ y_7 &= y_1 + RK(y_0 - y_1) \\ z_7 &= z_6 \end{aligned} \right\} M_7(x_7, y_7, z_6)$$

Determinarea razei cercului de nivel R_1 :

$$\frac{R_1}{R} = \frac{SM_7}{SM_1} \quad R_1 = \frac{R \cdot SM_7}{SM_1} = \frac{R \sqrt{(x_0 - x_7)^2 + (y_0 - y_7)^2 + (z_0 - z_7)^2}}{\sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2 + (z_0 - z_1)^2}}$$

Distanța centrului M_7 al acestui cerc față de semi-axa mare a elipsei în proiecția orizontală este $\overline{M}_6 \overline{M}_7$

$$\overline{M}_6 \overline{M}_7 = \sqrt{(x_7 - x_6)^2 + (y_7 - y_6)^2}$$

Semi-axa mare a elipsei în proiecția orizontală este T_1

$$T_1 = \sqrt{R_1^2 - \overline{M}_6 \overline{M}_7^2}; \quad T_1 = \overline{M}_6 \overline{M}_8$$

Semiaxa mică a elipsei în proiecția orizontală este T_2

$$T_2 = \overline{M_5 M_6} \cos \varphi_1 \text{ unde}$$

$$\cos \varphi_1 = \frac{C}{-S\sqrt{A^2 + B^2 + C^2}} \text{ iar}$$

$$\overline{M_5 M_6} = \sqrt{(x_6 - x_5)^2 + (y_6 - y_5)^2 + (z_6 - z_5)^2}$$

Pentru plottarea elipsei proiecție orizontală:

$$BB = \arctg\left(-\frac{A}{B}\right)$$

Dacă vârful de plottare este

$$M_8(x_8, y_8, z_8); \quad z_8 = 0$$

obținem:

$$T_1 = \overline{M_6 M_8} = F_1$$

unde F_1 este semiaxa mare reală a elipsei de secțiune iar $F_2 = R56$ este semiaxa mică reală a elipsei.

Avem

$$V = \frac{T_1}{\sqrt{A^2 + B^2}}$$

Deci

$$M_8 \begin{cases} x_8 = x_6 - BV \\ y_8 = y_6 + AV \\ z_8 = 0 \end{cases}$$

coordonatele vârfului de plottare în proiecția orizontală pentru elipsa proiecție.

Instrucțiunea pentru plottare este după relațiile anterioare:

$$\text{CALL ELIPSA } (X_6, Y_6, 3, T_1, T_2, 0., 360., \arctg\left(-\frac{A}{B}\right))$$

În mod analog se obține determinarea vîrfurilor de plottare ale elipselor proiecției verticală și laterală.

4.2.2 Aplicație:

Vîrfurile conului $S(x_0, y_0, z_0)$ $x_0 = 19; y_0 = 16.5; z_0 = 10.5$

Centrul cercului director $x_1 = 6; y_1 = 5; z_1 = 0$

Raza cercului director: $R = 4$

Planul de secțiune:

$$\frac{x}{19} + \frac{y}{21.6} + \frac{z}{5.2} - 1 = 0 \text{ sau}$$

$$112.32 x + 98.8 y + 410.4 z - 2134.08 = 0 \quad S = -1$$

X2 =	3.004	Y2 =	2.350	Z2 =	.000
X3 =	8.996	Y3 =	7.650	Z3 =	.000
X4 =	6.339	Y4 =	5.300	Z4 =	2.189
X5 =	9.579	Y5 =	8.166	Z5 =	.612
X6 =	7.959	Y6 =	6.733	Z6 =	1.401
X7 =	7.734	Y7 =	6.534	Z7 =	1.401
T1 =	3.453	T2 =	2.163		
UBB =	-48.689				
X8 =	5.678	Y8 =	9.326	Z8 =	.000
SEMIAXA MARE REALĂ =			3.453		
SEMIAXA MICĂ REALĂ =			2.302		

Aplicații propuse:

a) Planul de secțiune:

$$\frac{x}{22} + \frac{y}{16} + \frac{z}{7} - 1 = 0 \text{ sau}$$

$$112x + 22y + 16z - 246 = 0$$

virful conului: $S(x = 26; y_0 = 19; z_0 = 14)$ cercul director: $x_c = 6; y_c = 6; z_c = 0; R = 5$

b) Planul de secțiune comun

$$\frac{x}{16,5} + \frac{y}{15} + \frac{z}{10} - 1 = 0 \text{ sau}$$

$$A1 = 150$$

$$B1 = 165$$

$$C1 = 247,5$$

$$D1 = -2475$$

Cercul director al cilindrilor (comun)

$$x = 4$$

$$y = 5 \text{ RAZA } R = 3$$

$$z = 0$$

CILINDRUL OBLIC $AL1 = 13; BE1 = 6; GA1 = 12$ CILINDRUL VERTICAL $AL1 = 0; BE1 = 0; GA1 = 10$ CILINDRUL OBLIC FRONTAL $AL1 = 11; BE1 = 0; GA1 = 8$

4.2.3 Program principal con

```

C SECTIUNEA ELIPTICA IN CONUL CIRCULAR OBLIC SAU DREPT
C VIRFUL CONULUI ESTE XO,YO,ZO
C RAZA CERCULUI DIKECTOR AL CONULUI ESTE R IAR
C CENTRUL ACESTUI CERC ESTE X1,Y1,Z1
C PARAMETRI PLANULUI DE SECTIUNE SINT A,B,C,U,S
C DIMENSION A(100),B(100),C(100),D(100),S(100),XO(100),YO(100),ZO(10
  *U),X1(100),Y1(100),Z1(100),R(100)
  READ(105,1) N
  1 FORMAT(I4)
  2 READ(105,2) (A(I),B(I),C(I),D(I),S(I),I=1,N)
  3 FORMAT(5F10.3,30X)
  4 READ(105,3) (XO(I),YO(I),ZO(I),X1(I),Y1(I),Z1(I),R(I),I=1,N)
  5 FORMAT(7F10.3,10X)
  6 DO 4 I=1,N
    WRITE(108,100)
  100 FORMAT(/,30(1X,'*')//)
    WRITE(108,101) I
  101 FORMAT(' ',SITUATIA NR: ',I2//)
    S1=(YO(I)-Y1(I))/(XO(I)-X1(I))
    X3=X1(I)+R(I)/SQRT(1+S1*S1)
    Y3=Y1(I)+S1*(X3-X1(I))
    X2=X1(I)-R(I)/SQRT(1+S1*S1)
    Y2=Y1(I)+S1*(X2-X1(I))
    Z3=0
    Z2=0
    WRITE(108,13) X2,Y2,Z2,X3,Y3,Z3
  13 FORMAT(' ',X2=',',F10.3,2X,',',Y2=',',F10.3,2X,',',Z2=',',F10.3/' ',X3=',',F10
    *,3,2X,',',Y3=',',F10.3,2X,',',Z3=',',F10.3//)
    RNUM1=A(I)*X2+B(I)*Y2+C(I)*Z2+D(I)
    RNUM2=A(I)*(XO(I)-X2)+B(I)*(YO(I)-Y2)+C(I)*(ZO(I)-Z2)
    S3=-RNUM1/RNUM2
    X4=S3*(XO(I)-X2)+X2
    Y4=S3*(YO(I)-Y2)+Y2
    Z4=S3*(ZO(I)-Z2)+Z2
    WRITE(108,6) X4,Y4,Z4
  6 FORMAT(' ',X4=',',F10.3,2X,',',Y4=',',F10.3,2X,',',Z4=',',F10.3//)
    RNU3=A(I)*X3+B(I)*Y3+C(I)*Z3+D(I)
    RNU4=A(I)*(XO(I)-X3)+B(I)*(YO(I)-Y3)+C(I)*(ZO(I)-Z3)
    S4=-RNU3/RNU4
    X5=S4*(XO(I)-X3)+X3
    Y5=S4*(YO(I)-Y3)+Y3
    Z5=S4*(ZO(I)-Z3)+Z3
    WRITE(108,7) X5,Y5,Z5
  7 FORMAT(' ',X5=',',F10.3,2X,',',Y5=',',F10.3,2X,',',Z5=',',F10.3//)
    X6=(X4+X5)/2
    Y6=(Y4+Y5)/2
    Z6=(Z4+Z5)/2
    WRITE(108,8) X6,Y6,Z6
  8 FORMAT(' ',X6=',',F10.3,2X,',',Y6=',',F10.3,2X,',',Z6=',',F10.3//)
    RK=(Z6-Z1(I))/(ZO(I)-Z1(I))
    X7=RK*(XO(I)-X1(I))+X1(I)
    Y7=RK*(YO(I)-Y1(I))+Y1(I)
    Z7=Z6
    WRITE(108,9) X7,Y7,Z7
  9 FORMAT(' ',X7=',',F10.3,2X,',',Y7=',',F10.3,2X,',',Z7=',',F10.3//)
    RRI=R(I)*SQRT((XO(I)-X7)**2+(YO(I)-Y7)**2+(ZO(I)-Z6)**2)
    RR2=SQRT((XO(I)-X1(I))**2+(YO(I)-Y1(I))**2+(ZO(I)-Z1(I))**2)
    K1=RR1/RR2
    R67=SQRT((X7-X6)**2+(Y7-Y6)**2)
    T1=SQRT(K1**2-R67**2)
    COS=C(I)/(-S(I)*SQRT(A(I)**2+B(I)**2+C(I)**2))
    K56=SQRT((X6-X5)**2+(Y6-Y5)**2+(Z6-Z5)**2)
    T2=R56*COS
    WRITE(108,10) T1,T2
  10 FORMAT(' ',T1=',',F10.3,2X,',',T2=',',F10.3//)
    ARG=-A(I)/B(I)
    BB=ATAN(ARG)
    UBB=(360*BB)/6.28
    WRITE(108,12) UBB
  12 FORMAT(' ',UBB=',',F10.3//)
    V=T1/SQRT(A(I)**2+B(I)**2)
    X8=X6-B(I)*V
    Y8=Y6+A(I)*V
    Z8=0
    WRITE(108,11) X8,Y8,Z8
  11 FORMAT(' ',X8=',',F10.3,2X,',',Y8=',',F10.3,2X,',',Z8=',',F10.3//)
C MARIMEA REALA A ELIPSEI DE SECTIUNE IN CON
C F1=SEMIAXA MARE REALA
C F2=SEMIAXA MICA REALA
  F1=T1
  F2=R56
  WRITE(108,20) F1,F2
  20 FORMAT(' ',SEMIAXA MARE REALA=',',F10.3/' ',SEMIAXA MICA REALA=',',F
    *10.3//)
  4 CONTINUE
  STOP
  END

```

4.2.4 Secțiuni plane punctuale în suprafețele conice sau cilindrice.

În cazul general al secțiunilor plane în suprafețele conice, conform teoremelor lui Dandelin, putem obține printr-un număr suficient de puncte secțiunile eliptică, hiperbolică sau parabolică. Alegînd, astfel, un număr suficient de mare de generatoare, obținem punctele curbei de secțiune, intersectînd aceste generatoare cu planul de secțiune. Putem utiliza în acest scop „SUBROUTINE INPLDR” studiată în capitolul II.

Astfel, fie planul de secțiune

$$Ax + By + Cz + D = 0$$

(x_0, y_0, z_0) vîrfurile conului și (x_c, y_c, z_c) coordonatele centrului cercului director al conului, cerc de rază R în ipoteza că este situat într-unul din planele de proiecție (sau în plane paralele cu acestea). De fapt, curba directoare a suprafeței conice sau cilindrice poate fi arbitrară în spațiu. În acest caz, generatoarele ce vor fi considerate în secțiunea plană în con sau cilindru vor fi definite de vîrfurile conului parametri directori și de punctul curent al curbei directoare ce va fi determinat printr-un subprogram special.

Această mulțime de puncte împreună cu vîrfurile conului vor determina mulțimea de generatoare, care vor fi intersectate cu planul de secțiune prin subprogramul **INPLDR**.

Vom considera, de exemplu, 72 puncte pe circumferința cercului director al conului sau cilindrului ca urme orizontale ale generatoarelor alese din 5° în 5° sau din 9° în 9° .

Astfel, pentru întocmirea programului, vom putea scrie:

$$\text{PAS} = 5 \text{ sau } 9$$

$$\text{COSA} = \text{COS}(\text{ALFA})$$

$$\text{SINA} = \text{SIN}(\text{ALFA})$$

$$\text{XT} = \text{XC} + \text{R} * \text{COS}(\text{ALFA})$$

$$\text{YT} = \text{YC} + \text{R} * \text{SIN}(\text{ALFA})$$

$$\text{ZT} = \text{ZC}$$

$$\text{ALF1} = \text{ALFI} + \text{PAS}$$

$$\text{ALFA} = \text{ALF1} * 3.14159/180$$

Construcția tangentelor la cercul director al suprafeței conice (pentru generatoarele de contur aparent)

Ecuția cercului este:

$$(x - a)^2 + (y - b)^2 = R^2$$

Ecuția dedublată a tangentei este:

$$(x - a)(x_1 - a) + (y - b)(y_1 - b) = R^2$$

în punctul x_1, y_1 de pe cerc.

Această tangentă trebuie să treacă prin vârful (v_1, v_2) al conului. Deci rezolvarea sistemului:

$$\begin{cases} (v_1 - a)(x_1 - a) + (v_2 - b)(y_1 - b) = R^2 \\ (x_1 - a)^2 + (y_1 - b)^2 = R^2 \end{cases}$$

conduce la determinarea celor două puncte de tangentă ale generatoarelor de contur aparent (orizontal, vertical, lateral).

Dezvoltînd, obținem:

$$\frac{v_1 - a}{v_2 - b} = u_1$$

$$\frac{R^2 + av_1 + bv_2 - a^2 - b^2}{v_2 - b} = u_2$$

$$y_1 = -x_1 u_1 + u_2$$

Înlocuind această valoare în cea de a doua ecuație a sistemului, rezultă:

$$T_1 x_1^2 + T_2 x_1 + T_3 = 0$$

unde:

$$+ T_1 = 1 + u_1^2$$

$$+ T_2 = 2bu_1 - 2a - 2u_1 u_2$$

$$+ T_3 = a^2 + b^2 + u_2^2 - 2u_2 b - R^2$$

Rezolvînd ecuația de gradul doi, obținem coordonatele punctelor de tangentă pe care le notăm (în forma programabilă) prin X_{11} , X_{12} și Y_{11} , Y_{12} , unde:

$$X_{11}, X_{12} = \frac{-T_2 \pm \sqrt{T_2^2 - 4T_1 T_3}}{2T_1}$$

$$Y_{11} = -X_{11} \cdot U_1 + U_2$$

$$Y_{12} = -X_{12} \cdot U_1 + U_2$$

4.2.5 Program INTCIL

```

PROGRAM INICIL
C COORDONATELE CERCULUI DE BAZA ALE CILINDRULUI SINT AC,YA,ZC IAR RAZA R
C PARAMETRII DIRECTORI AI GENERATOARELOR CILINDRULUI SINT AL,BE,GA
C PARAMETRII CARE DEFINESC PLANUL DE SECTIUNE SINT A,B,C,D
  DIMENSION XC(100),YC(100),ZC(100),R(100),A(100),B(100),C(100),D(10
  *0),AL(100),BE(100),GA(100),X2(100),Y2(100),Z2(100)
  CALL ASSIGN(1,'CR:')
  CALL ASSIGN(2,'LP:')
  CALL ASSIGN(3,'PP:')
  CALL INI(3)
  READ(1,3) N
  3 FORMAT(14)
  READ(1,1) (XC(I),YC(I),ZC(I),R(I),I=1,N)
  1 FORMAT(4F10.3,40X)
  READ(1,2) (AL(I),BE(I),GA(I),I=1,N)
  2 FORMAT(3F10.3,50X)
  READ(1,10) (A(I),B(I),C(I),D(I),I=1,N)
  10 FORMAT(4F10.3,40X)
  PAS=9
  DO 4 I=1,N
  WRITE(2,101) I
  101 FORMAT(' ','SITUATIA NR:',I2//)
  K=0
  WRITE(2,21) XC(I),YC(I),ZC(I),R(I)
  21 FORMAT(' ','COORDONATELE CENTRULUI CERCULUI SI RAZA',I',',4F10.3//)
  ALF1=0
  J=1
  5 COSA=COS(ALFA)
  SINA=SIN(ALFA)
  XT=XC(I)+R(I)*COSA
  YT=YC(I)+R(I)*SINA
  ZT=ZC(I)
  WRITE(2,8) K
  8 FORMAT(' ','K=',I3//)
  WRITE(2,7) ALF1
  7 FORMAT(' ','CALCULUL PENTRU UNghiUL ALFA=',F6.2//)
  A1=A(I)
  B1=B(I)
  C1=C(I)
  D1=D(I)
  AL1=AL(I)
  BE1=BE(I)
  GA1=GA(I)
  CALL INPLDR(A1,B1,C1,D1,XT,YT,ZT,AL1,BE1,GA1,X,Y,Z)
  X2(J)=X
  Y2(J)=Y
  Z2(J)=Z
  WRITE(2,6) XT,YT,ZT,X2(J),Y2(J),Z2(J)
  6 FORMAT(' ','PUNCTUL DE PE CONTURUL BAZEI',3F10.3/ ' ','PUNCTUL DE
  *INTERSECTIE',3F10.3//)
  K=J
  J=J+1
  ALF1=ALF1+PAS
  ALFA=ALF1*3.14159/180.
  IF(ALF1.LT.360) GO TO 5
  CALL LIN(-130.,0.,130.,0.)

```

```

CALL LIN(0.,-95.,0.,95.)
CALL TEX(-130.,1.,0.,2.5,0.,'X',1)
CALL TEX(130.,1.,0.,2.5,0.,'Y',2)
CALL TEX(1.,95.,0.,2.5,0.,'Z',3)
CALL TEX(1.,-95.,0.,2.5,0.,'Y',1)
CALL TEX(.3.,3,0.,2.5,0.,'U',1)
CALL TEX(10.,-20.,0.,4.,0.,'SUBROUTINE INTCIL',17)
CALL TEX(10.,-30.,0.,4.,0.,'SECTIUNE PLANA IN CILINDRU',26)
S1=-D1/A1
S2=-D1/B1
S3=-D1/C1
CALL PLOT(0.,-S2,0)
CALL PLOT(-S1,0.,1)
CALL PLOT(0.,S3,1)
CALL PLOT(S2,0.,1)
CALL TEX(1.,-S2,0.,2.5,0.,'S2',2)
CALL TEX(-S1,1.,0.,2.5,2.,'S1',2)
CALL TEX(1.,S3,0.,2.5,0.,'S3',2)
CALL TEX(S2,1.,0.,2.5,0.,'S2',2)
CALL CIS(S2,0.,.5)
CALL CIS(0.,-S2,.5)
CALL CIS(0.,S3,.5)
CALL CIS(-S1,0.,.5)
CALL CIS(-80.,-40.,R)
CALL CIS(-80.,-20.,R)
XX1=-20.+R(1)
XY1=-20.-R(1)
XX2=-80.+R(1)
XY2=-80.-R(1)
CALL LIN(XX1,0.,XX2,50.)
CALL LIN(XX2,50.,XY2,50.)
CALL LIN(XY2,50.,XY1,0.)
XX3=20.+R(1)
XY3=20.-R(1)
XX4=40.+R(1)
XY4=40.-R(1)
CALL LIN(XY3,0.,XY4,50.)
CALL LIN(XY4,50.,XX4,50.)
CALL LIN(XX4,50.,XX3,0.)
CALL LIN(-80.,-40.,-20.,-20.)
CALL LIN(-20.,0.,-80.,50.)
CALL LIN(20.,0.,40.,50.)
CALL PLG(Y2,Z2,1,39,1)
DO 30 J=1,39
X2(J)=-X2(J)
Y2(J)=-Y2(J)
30 CONTINUE
CALL PLG(X2,Z2,1,39,1)
CALL PLG(X2,Y2,1,39,1)
4 CONTINUE
DO 31 J=1,39
X2(J)=-X2(J)
Y2(J)=-Y2(J)
31 CONTINUE
RAD=SQRT(4000.)
BC=RAD/3.

```

Secțiunea plană punctuală într-o suprafață cilindrică este prezentată în triplă proiecție ortogonală în figura 4.4 a epura fiind executată integral la plotter.


```

BE2=BC**2-...
BT=SQRT(BE2)
PM=15.*BT/20.
MC=15.*BC/20.
H1=PM*15./MC
X11=-20.+M1
YS2=15.**2-H1**2
YS=SQRT(YS2)
Y11=-20.-YS
AD=2.*MAD/3.
AF2=AD**2-1600.
AF=SQRT(AF2)
KN=15.*AF/40.
NU=15.*AD/40.
HM=KN*15./NU
X12=-80.-HM
Y1S2=15.**2-HM**2
Y1S=SQRT(Y1S2)
Y12=-40.+Y1S
X21=-20.-M1
Y21=-20.+YS
X22=-80.+HM
Y22=-40.-Y1S
WRITE(2,40) X11,Y11,X21,Y21,X12,Y12,X22,Y22
40 FORMAT(' ',X11,Y11',2F10.3',X21,Y21',2F10.3',
',X12,Y12',2F10.3'
',X22,Y22',2F10.3//)
CALL LIN(X11,Y11,X22,Y22)
CALL LIN(X12,Y12,X21,Y21)
CALL EOF
STOP
END

SUBROUTINE INPLDR(A1,B1,C1,D1,XT,YT,ZT,AL1,BE1,GA1,X,Y,Z)
AK=(A1*XT+B1*YT+C1*ZT+D1)/(A1*AL1+B1*BE1+C1*GA1)
X=XT-AL1*AK
Y=YT-BE1*AK
Z=ZT-GA1*AK
RETURN
END

```

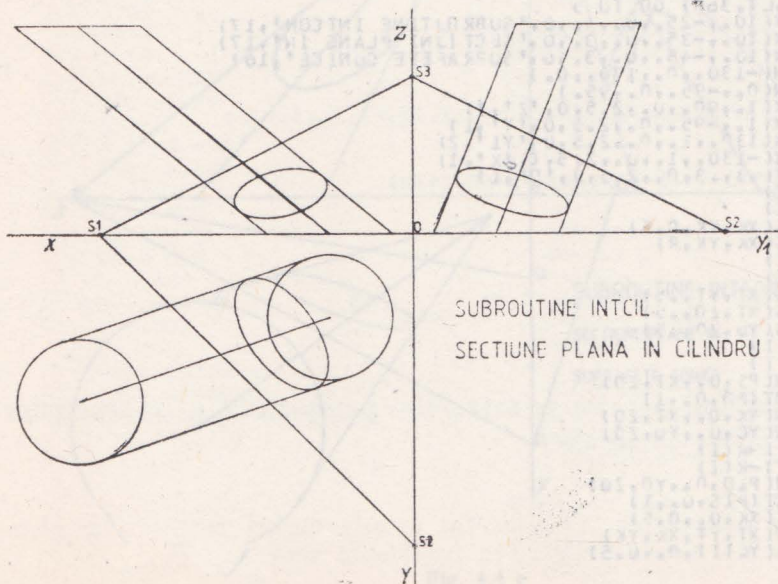


Fig 4.4 a

```

C SECTIONI PLANE PUNCTUALE IN SUPRAFETELE
C CONICE SI CILINDRICE
DIMENSIUN X0(100),Y0(100),Z0(100),XC(100),YC(100),ZC(100),R(100)
DIMENSIUN X2(100),Y2(100),Z2(100)
X0,Y0,Z0 SINT COORDONATELE VIRFULUI CONULUI
XC,YC,ZC SINT COORDONATELE CENTRULUI CERCLUI DE BAZA
K ESTE RAZA CERCLUI DE BAZA
A,B,C,J SINT PARAMETRII CARE DEFINESC PLANUL DE SECTIUNE
CALL ASSIGN(1,'CK: ')
CALL ASSIGN(2,'LP: ')
CALL ASSIGN(3,'PP: ')
CALL INI(3)
READ(1,3) N
3 FORMAT(14)
READ(1,1) (X0(I),Y0(I),Z0(I),XC(I),YC(I),ZC(I),R(I),I=1,N)
1 FORMAT(7F10.3,10X)
READ(1,2) X01,Y01,Z01,X02,Y02,Z02,X03,Y03,Z03
2 FORMAT(9F5.2,35X)
PAS=10.
DO 4 I=1,N
K=0
WRITE(2,21)X0(I),Y0(I),Z0(I),XC(I),YC(I),ZC(I)
21 FORMAT(' ','COORDONATELE VIRFULUI CONULUI',3F10.3/' ','COORDONATELE
* CENTRULUI CERCLUI',3F10.3//)
ALF1=0.
J=1
5 COSA=COS(ALFA)
SINA=SIN(ALFA)
XT=XC(I)+R(I)*COSA
YT=YC(I)+R(I)*SINA
ZT=ZC(I)
X1=X0(I)
Y1=Y0(I)
Z1=Z0(I)
A1=Y01*Z02-Y02*Z03+Y03*Z02-Y02*Z01+Y03*Z01-Y03*Z02
B1=X01*Z03-X02*Z02+X02*Z01-X02*Z03-X03*Z01+X03*Z02
C1=X01*Y02-X01*Y03+X02*Y03-Y03*Z01+X03*Y01-X03*Y02
D1=X01*Y02*Z03+X02*Y03*Z01+X03*Y01*Z02-X03*Y02*Z01-X02*Y01*Z03-X
01*Y03*Z02
D1=-D1
CALL INDRPL(A1,B1,C1,D1,XT,YT,ZT,X1,Y1,Z1,X,Y,Z)
X2(J)=X
Y2(J)=Y
Z2(J)=Z
K=J
J=J+1
ALF1=ALF1+PAS
ALFA=ALF1*3.14159/180.
IF(ALF1.LT.360) GO TO 5
CALL TEX(10.,-25.,0.,4.,0.,'SUBROJITINE INTCON',17)
CALL TEX(10.,-35.,0.,3.,0.,'SECTIUNI PLANE IN',17)
CALL TEX(10.,-45.,0.,3.,0.,'SUPRAFETE CONICE',16)
CALL LIN(-130.,0.,130.,0.)
CALL LIN(0.,-95.,0.,95.)
CALL TEX(1.,90.,0.,2.5,0,'Z',1)
CALL TEX(1.,-95.,0.,2.5,0,'Y',1)
CALL TEX(130.,1.,0.,2.5,0,'Y1',2)
CALL TEX(-130.,1.,0.,2.5,0,'X',1)
CALL TEX(.3,.3,0.,2.5,0,'D',1)
XK=-XC(I)
YK=-YC(I)
CALL CIS(XK,YK,0.5)
CALL CIS(XK,YK,R)
XT=-X0(I)
YT=-Y0(I)
CALL CIS(XT,YT,.5)
CALL CIS(XT,Z0,.5)
CALL CIS(XT,Y0,Z0,.5)
PD=XK+R(I)
PS=XK-R(I)
CALL LIN(PS,0.,XT,Z0)
CALL PLOT(PD,0.,1)
CALL LIN(XK,0.,XT,Z0)
CALL LIN(YC,0.,Y0,Z0)
P1D=YC(I)+R(I)
P1S=YC(I)-R(I)
CALL LIN(P1D,0.,Y0,Z0)
CALL PLOT(P1S,0.,1)
CALL CIS(XK,0.,0.5)
CALL LIN(XT,YT,XK,YK)
CALL CIS(YC(I),0.,0.5)

```

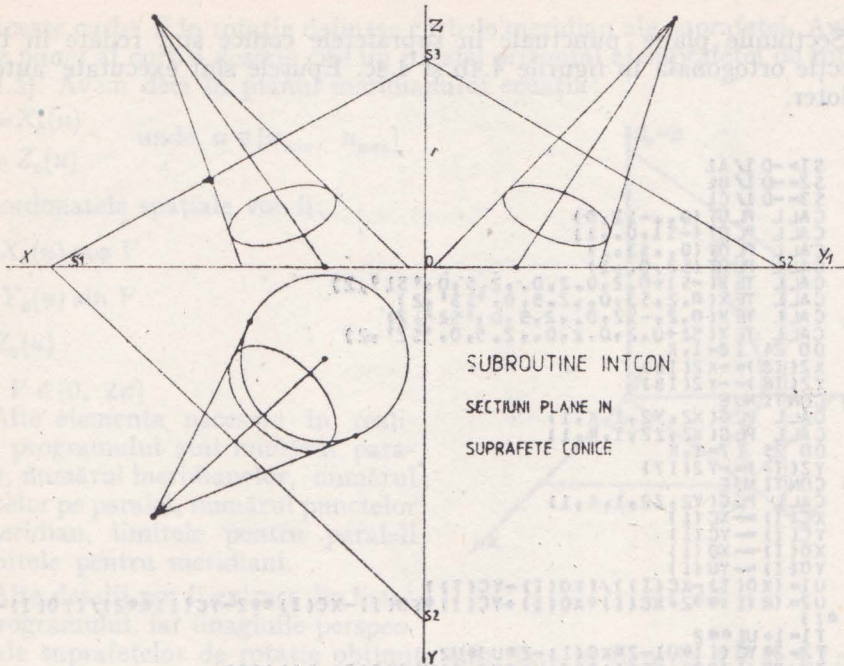


Fig. 4.4 b

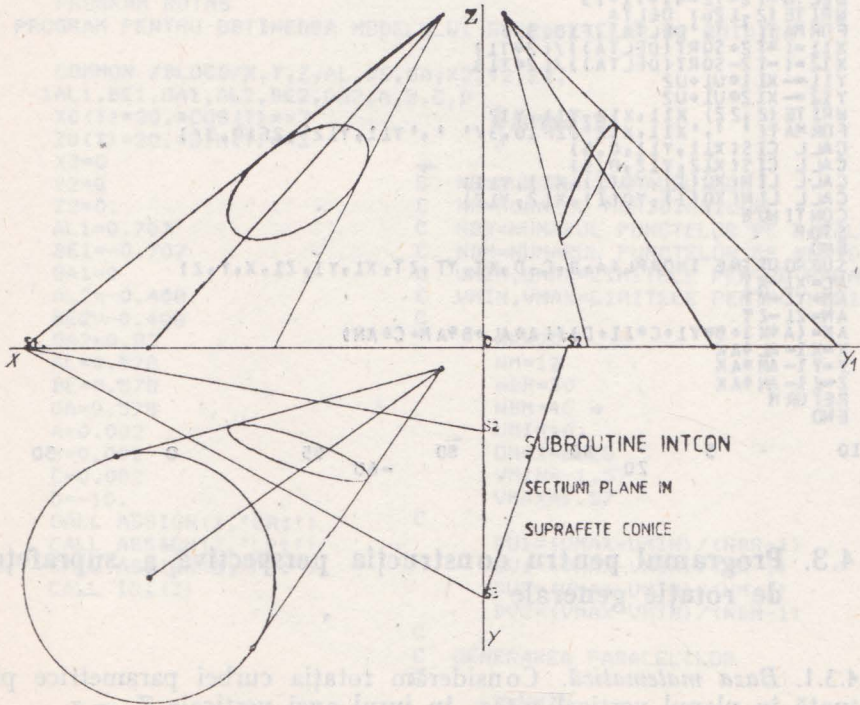


Fig. 4.4 c

Secțiunile plane punctuale în suprafețele conice sînt redată în triplă proiecție ortogonală în figurile 4.4b și 4.4c. Epurele sînt executate automat la ploter.

```

S1=-0 / AL
S2=-0 / BL
S3=-0 / CL
CALL PLOT (0, -S2, 0)
CALL PLOT (-S1, 0, 1)
CALL PLOT (0, S3, 1)
CALL PLOT (S2, 0, 1)
CALL TEX (-S1, 0, 2, 0, 2, 0, 2, 5, 0, S1, 2)
CALL TEX (0, 2, S3, 0, 2, 5, 0, S3, 2)
CALL TEX (0, 2, -S2, 0, 2, 5, 0, -S2, 2)
CALL TEX (S2, 0, 2, 0, 2, 0, 2, 5, 0, S2, 2)
DO 24 I8=1, K
X2(I8)=-X2(I8)
Y2(I8)=-Y2(I8)
24 CONTINUE
CALL PLG(X2, Y2, 1, K, 1)
CALL PLG(X2, Z2, 1, K, 1)
DO 25 I7=1, K
Y2(I7)=-Y2(I7)
25 CONTINUE
CALL PLG(Y2, Z2, 1, K, 1)
XC(I)=-XC(I)
YC(I)=-YC(I)
XO(I)=-XO(I)
YO(I)=-YO(I)
U1=(XO(I)-XC(I))/(YO(I)-YC(I))
U2=(R(I)**2*XC(I)*XO(I)+YC(I)*YO(I)-XC(I)**2-YC(I)**2)/(YO(I)-YC(I)
*)
T1=1+U1**2
T2=2*YC(I)*U1-2*XC(I)-2*U1*U2
T3=XC(I)**2+YC(I)**2+U2**2-2*U2*YC(I)-R(I)**2
WRITE (2, 124) T1, T2, T3
124 FORMAT (2, 124) T1, T2, T3, 3F10.3/)
DELTA=T2*T2-4*T1*T3
126 FORMAT (2, 126) DELTA, F10.3/)
X11=-T2+SQRT(DELTA)/(2*T1)
X12=-T2-SQRT(DELTA)/(2*T1)
Y11=-X11*U1+U2
Y12=-X12*U1+U2
22 FORMAT (2, 22) X11, X12, Y11, Y12
CALL CIS(X11, Y11, 0.5)
CALL CIS(X12, Y12, 0.5)
CALL LIN(XO(I), YO(I), X11, Y11)
CALL LIN(XO(I), YO(I), X12, Y12)
4 CONTINUE
STOP
END
SUBROUTINE INDRPL(A, B, C, D, XT, YT, ZT, X1, Y1, Z1, X, Y, Z)
AL=X1-XT
AM=Y1-YT
AN=Z1-ZT
AK=(A*X1+B*Y1+C*Z1+D)/(A*AL+B*AM+C*AN)
X=X1-AL*AK
Y=Y1-AM*AK
Z=Z1-AN*AK
RETURN
END
1
110 10 5 20 80 80 -60 55 0 30

```

4.3. Programul pentru construcția perspectivă a suprafețelor de rotație generale

4.3.1. *Baza matematică.* Considerăm rotația curbei parametrice plane Γ situată în planul vertical x_0Oz_0 în jurul axei verticale $Z_0 = z$.

Aceste curbe Γ în rotație definesc curbele meridian ale suprafeței. Analog fiecare punct al curbei descrie câte un paralel orizontal al suprafeței de rotație (fig. 4.5). Avem deci în planul meridianului ecuația:

$$\begin{cases} X_0 = X_0(u) \\ Z_0 = Z_0(u) \end{cases} \quad \text{unde } u \in [u_{\min}, u_{\max}]$$

iar coordonatele spațiale vor fi:

$$X = X_0(u) \cos V$$

$$Y = Y_0(u) \sin V$$

$$Z = Z_0(u)$$

unde $V \in [0, 2\pi]$

Alte elemente necesare în realizarea programului sînt numărul paralelilor, numărul meridianelor, numărul punctelor pe paralel, numărul punctelor pe meridian, limitele pentru paraleli și limitele pentru meridiani.

Alte detalii pot fi extrase din listarea programului, iar imaginile perspective ale suprafețelor de rotație obținute sînt date în figurile 4.6 și 4.7 a, b c, d, e și f.

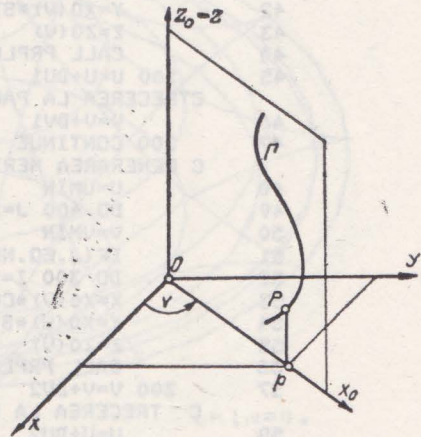


Fig. 4.5

```

PROGRAM ROTAS
C PROGRAM PENTRU OBTINEREA MODELULUI SUPRAFETELOR DE ROTATIE
C
COMMON /BLOC1/X,Y,Z,AL,BE,GA,X2,Y2,Z2,
1AL1,BE1,GA1,AL2,BE2,GA2,A,B,C,D
X0(T)=20.*COS(T)**3
Z0(T)=20.*SIN(T)**3
X2=0
Y2=0
Z2=0.
AL1=0.707
BE1=-0.707
GA1=0
AL2=-0.408
BE2=-0.408
GA2=0.82
AL=0.578
BE=0.578
GA=0.578
A=0.002
B=0.002
C=0.002
D=-10.
CALL ASSIGN(1,'CR:')
CALL ASSIGN(2,'LP:')
CALL ASSIGN(3,'PP:')
CALL INI(Z)
C
C } ASTROIDA
C NR=NUMARUL PARALELILOR
C NM=NUMARUL MERIDIANILOR
C NBR=NUMARUL PUNCTELOR PE PARALEL
C NBM=NUMARUL PUNCTELOR PE MERIDIAN
C UMIN,UMAX=LIMITELE PENTRU PARALELI
C VMIN,VMAX=LIMITELE PENTRU MERIDIAN
C
NR=20
NM=12
NBR=30
NBM=40
UMIN=0.
UMAX=6.28
VMIN=-1.57
VMAX=1.57
C
DU1=(UMAX-UMIN)/(NBR-1)
DV1=(VMAX-VMIN)/(NR-1)
DU2=(UMAX-UMIN)/(NM-1)
DV2=(VMAX-VMIN)/(NBM-1)
C
C GENERAREA PARALELILOR
C
V=VMIN
DO 200 J=1,NR
U=UMIN
C
DO 100 I=1,NBR

```

```

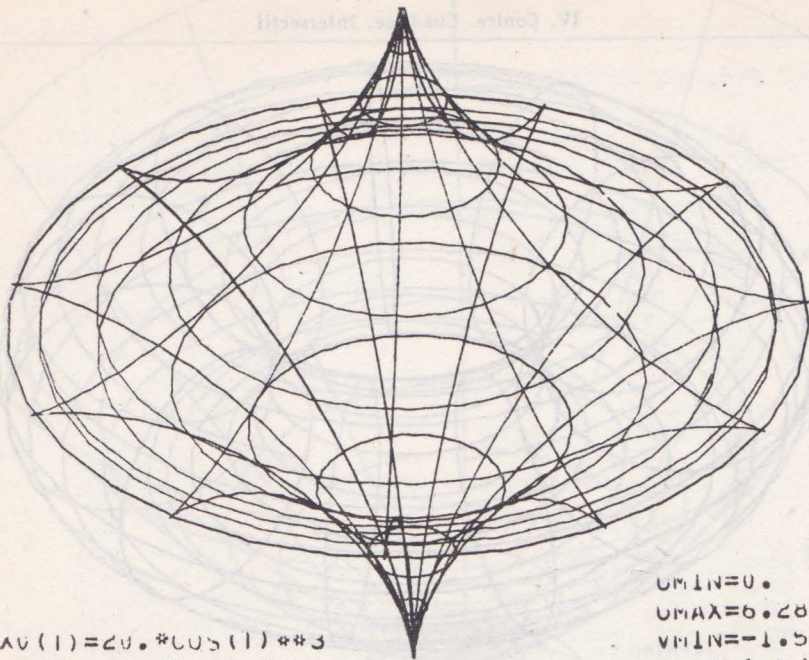
+1      X=X0(V)*COS(U)
42      Y=X0(V)*SIN(U)
43      Z=Z0(V)
44      CALL PRPLOT(I)
45      100 U=U+DU1
        CTRECEREA LA PARALELUL URMATOR
46      V=V+DV1
47      200 CONTINUE
        C GENERAREA MERIDIANILOR
48      U=UMIN
49      DO 400 J=1,NM
50      V=VMIN
51      IF(J.EQ.NM) U=UMAX
52      DO 300 I=1,NBM
53      X=X0(V)*COS(U)
54      Y=X0(V)*SIN(U)
55      Z=Z0(V)
56      CALL PRPLOT(I)
57      300 V=V+DV2
        C TRECEREA LA MERIDIANUL URMATOR
58      U=U+DU2
59      400 CONTINUE
60      CALL EOF
61      STOP
62      END

```

```

01      SUBROUTINE PRPLOT(I)
02      COMMON /BLOC/D/X,Y,Z,AL,BE,GA,X2,Y2,Z2,
        1AL1,BE1,GA1,AL2,BE2,GA2,A,B,C,D
03      LAMBDA=(A*X+B*Y+C*Z+D)/(A*AL+B*BE+C*GA)
04      XP=X-AL*LAMBDA
05      YP=Y-BE*LAMBDA
06      ZP=Z-GA*LAMBDA
07      X1=AL1*(XP-X2)+BE1*(YP-Y2)+GA1*(ZP-Z2)
08      Y1=AL2*(XP-X2)+BE2*(YP-Y2)+GA2*(ZP-Z2)
09      IF(I.EQ.1) CALL PLOT(X1,Y1,0)
10      CALL PLOT(X1,Y1,1)
11      RETURN
12      END

```



$XU(T) = 20. * \cos(T) ** 3$
 $ZU(T) = 20. * \sin(T) ** 3$

$UMIN = 0.$
 $UMAX = 6.28$
 $VMIN = -1.57$
 $VMAX = 1.57$

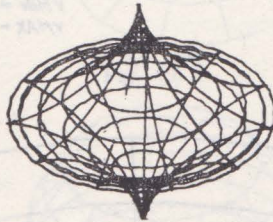
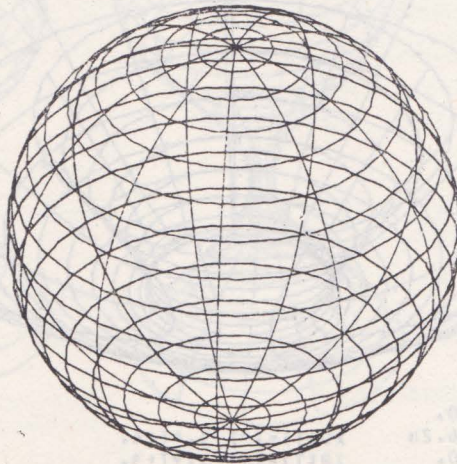


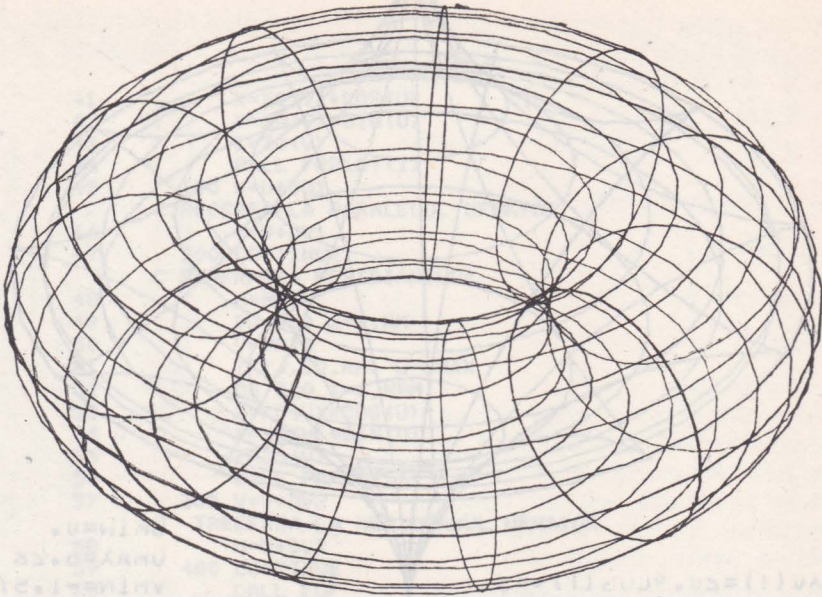
Fig. 4.6 a



$XO(T) = 50. * \cos(T)$
 $ZO(T) = 50. * \sin(T)$

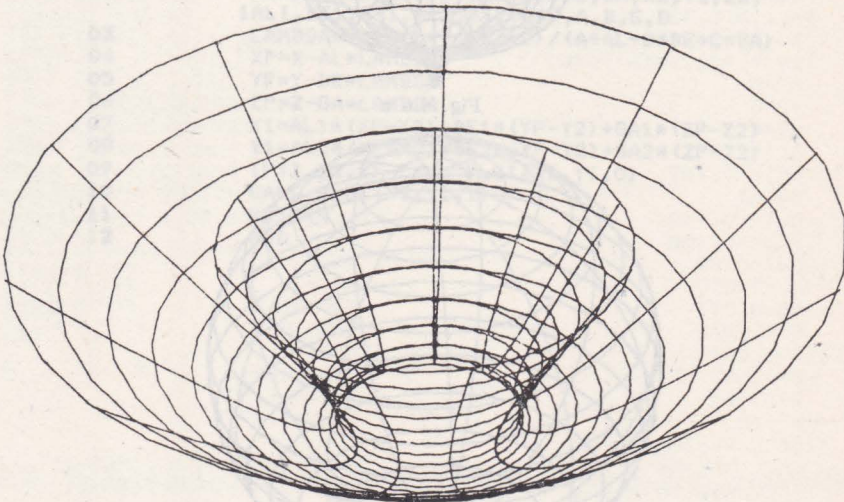
$UMIN = 0.$
 $UMAX = 6.28$
 $VMIN = -1.57$
 $VMAX = 1.57$

Fig. 4.6 b



$$\begin{aligned} X0(T) &= 60. + 30. * \cos(T) & UMIN &= 0. \\ Z0(T) &= 30. + 30. * \sin(T) & UMAX &= 6.28 \\ & & VMIN &= 0. \\ & & VMAX &= 6.28 \end{aligned}$$

Fig. 4.6 c



$$\begin{aligned} UMIN &= 0. & X0(T) &= T * T - 3. * T + 4. \\ UMAX &= 6.28 & Z0(T) &= T * T - 4. * T + 4. \\ VMIN &= 0. \\ VMAX &= 4. \end{aligned}$$

Fig. 4.6 d

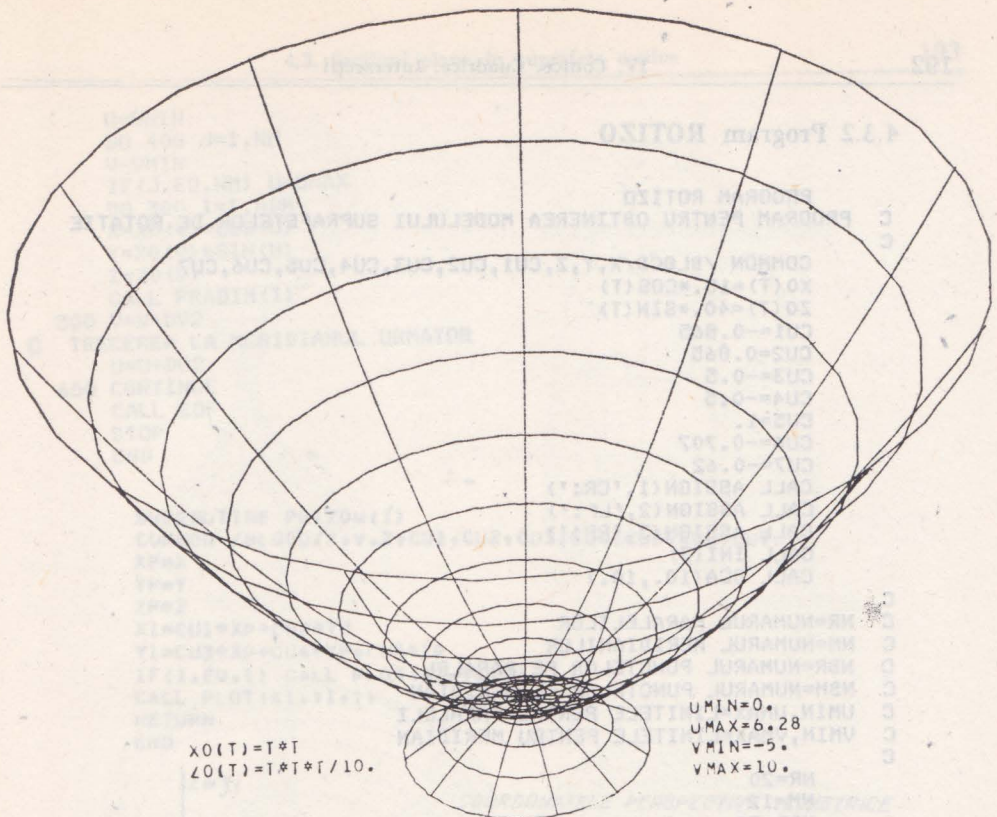
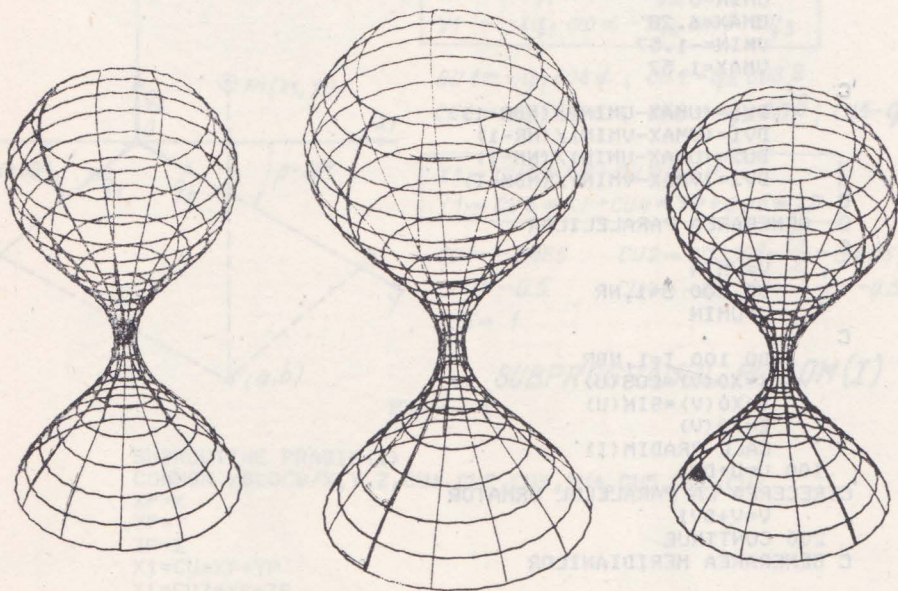


Fig. 4.6 e



TEST ROTIZO

Fig. 4.6 f

4.3.2 Program ROTIZO

```

PROGRAM ROTIZO
C PROGRAM PENTRU OBTINEREA MODELULUI SUPRAFETELOR DE ROTATIE
C
COMMON /BL0CD/X,Y,Z,CU1,CU2,CU3,CU4,CU5,CU6,CU7
XO(T)=15.*COS(T)
ZO(T)=40.*SIN(T)
CU1=-0.865
CU2=0.865
CU3=-0.5
CU4=-0.5
CU5=1.
CU6=-0.707
CU7=-0.62
CALL ASSIGN(1,'CR:')
CALL ASSIGN(2,'LP:')
CALL ASSIGN(3,'PF:')
CALL INI(3)
CALL SCA(10.,10.)

C
C NR=NUMARUL PARALELILOR
C NM=NUMARUL MERIDIANILOR
C NBR=NUMARUL PUNCTELOR PE PARALEL
C NBM=NUMARUL PUNCTELOR PE MERIDIAN
C UMIN,UMAX=LIMITELE PENTRU PARALELI
C VMIN,VMAX=LIMITELE PENTRU MERIDIAN
C
NR=20
NM=12
NBR=30
NBM=40
UMIN=0
UMAX=6.28
VMIN=-1.57
VMAX=1.57

C
DU1=(UMAX-UMIN)/(NBR-1)
DV1=(VMAX-VMIN)/(NBR-1)
DU2=(UMAX-UMIN)/(NM-1)
DV2=(VMAX-VMIN)/(NBM-1)

C
C GENERAREA PARALELILOR
C
V=VMIN
DO 200 J=1,NR
U=UMIN

C
DO 100 I=1,NBR
X=XO(V)*COS(U)
Y=XO(V)*SIN(U)
Z=ZO(V)
CALL PRADIM(I)
100 U=U+DU1
CTRECEREA LA PARALELUL URMATOR
V=V+DV1
200 CONTINUE
C GENERAREA MERIDIANILOR

```

```

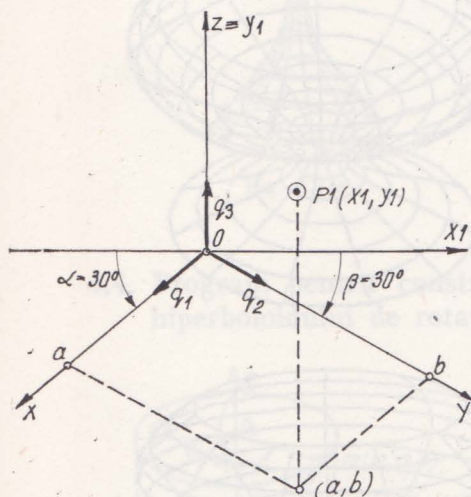
U=UMIN
DO 400 J=1,NM
V=UMIN
IF (J.EQ.NM) U=UMAX
DO 300 I=1,NBM
X=XO(V)*COS(U)
Y=XO(V)*SIN(U)
Z=ZO(V)
CALL PRADIM(I)
300 V=V+DV2
C TRECERE LA MERIDIANUL URMATOR
U=U+DU2
400 CONTINUE
CALL EOF
STOP
END

```

```

SUBROUTINE PRIZOM(I)
COMMON /BLOC/D/X,Y,Z,CU1,CU2,CU3,CU4,CU5,CU6,CU7
XP=X
YP=Y
ZP=Z
X1=CU1*XP+CU2*YP
Y1=CU3*XP+CU4*YP+CU5*ZP
IF (I.EQ.1) CALL PLOT(X1,Y1,0)
CALL PLOT(X1,Y1,1)
RETURN
END

```



COORDONATELE PERSPECTIVE IZOMETRICE

$$\begin{aligned} X1 &= -XQ_1 \cos \alpha + YQ_2 \cos \beta \\ Y1 &= -XQ_1 \sin \alpha - YQ_2 \sin \beta + ZQ_3 \end{aligned}$$

$$\begin{aligned} CU1 &= -Q_1 \cos \alpha; & CU2 &= Q_2 \cos \beta \\ CU3 &= -Q_1 \sin \alpha; & CU4 &= -Q_2 \sin \beta; & CU5 &= Q_3 \end{aligned}$$

$$\begin{aligned} X1 &= CU1 * XP + CU2 * YP \\ Y1 &= CU3 * XP + CU4 * YP + CU5 * ZP \end{aligned}$$

$$\begin{aligned} CU1 &= -0.865 & CU2 &= \cos 30^\circ = \frac{\sqrt{3}}{2} = 0.865 \\ CU3 &= -0.5 & CU4 &= -\sin 30^\circ = -\frac{1}{2} = -0.5 \\ CU5 &= 1 \end{aligned}$$

SUBPROGRAMUL PRIZOM(I)

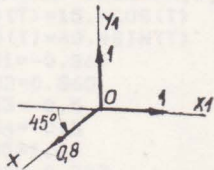
Fig. 4.7 a

```

01 SUBROUTINE PRADIM(I)
02 COMMON /BLOC/D/X,Y,Z,CU1,CU2,CU3,CU4,CU5,CU6,CU7
03 XP=X
04 YP=Y
05 ZP=Z
06 X1=CU*XP+YP
07 Y1=CU7*XP+ZP
08 IF (I.EQ.1) CALL PLOT(X1,Y1,0)
09 CALL PLOT(X1,Y1,1)
10 RETURN
11 END

```

COORDONATELE PERSPECTIVE
DIMETRICE



$$X1 = CU6 * XP + YP$$

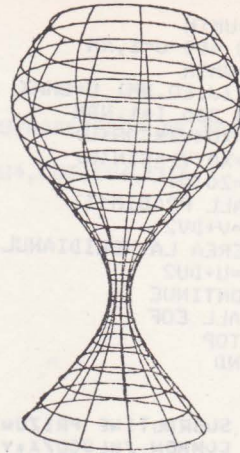
$$Y1 = CU7 * XP$$

$$CU6 = -0,70 = -\frac{\sqrt{2}}{2}$$

$$CU7 = -0,62 = -0,8 \cdot \frac{\sqrt{2}}{2}$$

SUBPROGRAMUL PRADIM(I)

Fig. 4.7 b



PRIZOM

Fig. 4.7 c

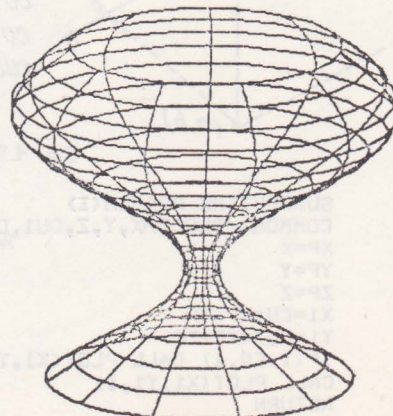
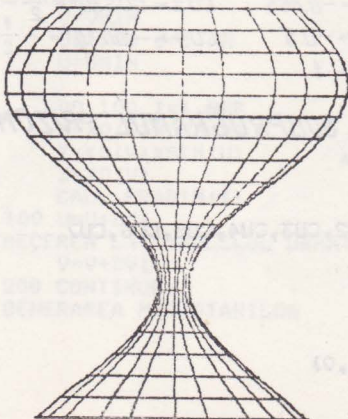
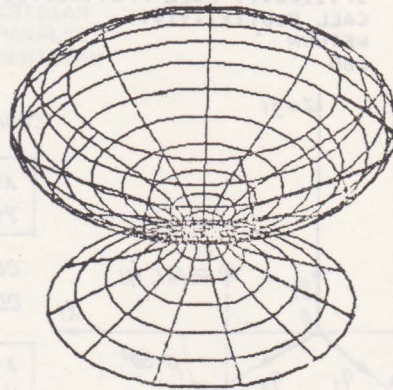
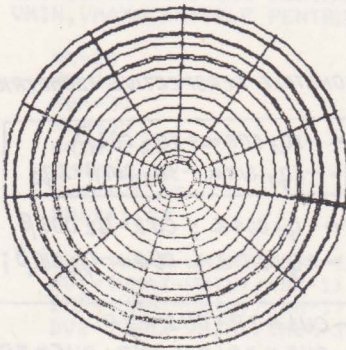


Fig. 4.7 d

Aplicațiile programelor ROTIZO și ale subprogramelor PRIZOM (I) și PRADIM (I) sînt ilustrate în figurile 4.7.b, c, d, e, f.

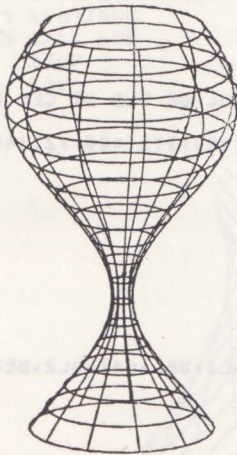
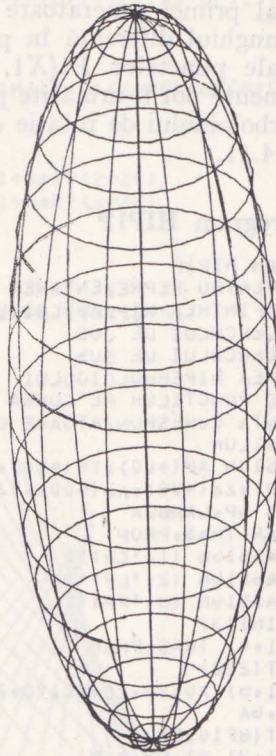


Fig. 4.7 e



PRADIM

Fig. 4.7 f

4.4. Program pentru construcția (reprezentarea perspectivă a) hiperboloidului de rotație cu o pînză

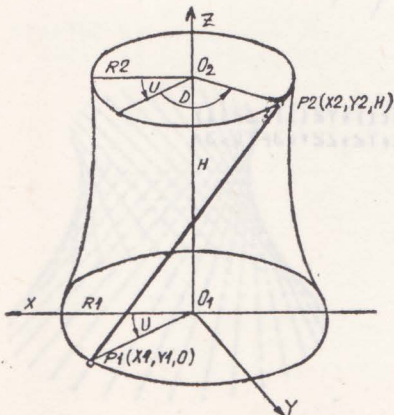


Fig. 4.8

4.4.1 Baza matematică

Să considerăm reprezentarea schematică în perspectivă paralelă a hiperboloidului de rotație cu o pînză din fig. 4.8.

Elementele care sînt necesare pentru întocmirea programului vor fi următoarele:

- R_1 — raza cercului paralel inferior
- R_2 — raza cercului paralel superior
- H — înălțimea segmentului de hiperboloid cuprins între planele celor doi paraleli

- N — numărul de generatoare din primul sau al doilea sistem
 U — unghiul care măsoară poziția urmei $P1(X1, Y1, 0)$ de început al primei generatoare pe cercul inferior față de originea dată
 D — unghiul diferență în plan orizontal dintre proiecțiile orizontale ale punctelor $P1(X1, Y1, 0)$ și $P2(X2, Y2, H)$.

Alte elemente pot fi urmărite pe listarea programului. Imaginile perspective ale hiperboloidului de rotație cu o pinză obținute sint redade pe figurile 4.9, 4.10, și 4.11.

4.4.2. Program HIP1P

```

PROGRAM HIP1P
C PROGRAM PENTRU REPREZENTAREA HIPERBOLOIDULUI CU O PINZA
C PENTRU DEFINIREA HIPERBOLOIDULUI SE DAU
C K1=RAZA CERCULUI DE JOS
C K2=RAZA CERCULUI DE SUS
C H=INALTIMEA HIPERBOLOIDULUI
C N=NUMARUL PUNCTELOR PE CURBA
C D=DIFERENTA CORESPUNZATOARE DINTRE PUNCTUL DE SUS SI CEL DE JOS ALE
C *CERCURILOR
  DIMENSION XP(400),YP(400),ZP(400),X1(400),Y1(400),Z1(400),X2(400),
  *Y2(400),Z2(400),X3(400),Y3(400)
  REAL*4 NP,LAMBDA
  INTEGER TRAS,PROP
  CALL ASSIGN (1,'CR:')
  CALL ASSIGN (2,'LP:')
  CALL ASSIGN (3,'PP:')
  CALL INI(3)
  READ(1,4) TRAS,PROP
  4 FORMAT(2I2)
  READ(1,5) XU,YU,ZU,XC,YC,ZC,A,B,C,D,AL1,BE1,GA1,AL2,BE2,GA2,
  *AL,BE,GA
  5 FORMAT(8F10.5)
  READ(1,2) R1,R2,H,N1
  2 FORMAT(3F10.3,I5,45X)
  PI=3.1416
  DU=2.*PI/FLOAT(N1)
  I=0
  U=0.
100 IF(U.GE.2.*PI) GO TO 200
  I=I+1
  X1(I)=R1*COS(U)
  Y1(I)=R1*SIN(U)
  X2(I)=R2*COS(U+DU)
  Y2(I)=R2*SIN(U+DU)
  Z1(I)=0.
  Z2(I)=H
  U=U+DU
  WRITE(2,203) I,X1(I),Y1(I),Z1(I),X2(I),Y2(I),Z2(I)
203 FORMAT(' ',I3,'X1,Y1,Z1',3F10.3,'X2,Y2,Z2',3F10.3/
GO TO 100
200 N=N1
  DO 1 I1=1,N
  XP(I1)=X1(I1)
  YP(I1)=Y1(I1)
  ZP(I1)=Z1(I1)
  1 CONTINUE
  N1=N+1
  NC=2*N
  DO 21 I2=N1,NC
  XP(I2)=X2(I2-N)
  YP(I2)=Y2(I2-N)
  ZP(I2)=Z2(I2-N)
21 CONTINUE
  DO 20 I=1,NC
  IF (PROP.NE.1) GO TO 15
  LAMBDA=(A*XP(I)+R*YP(I)+C+ZP(I)+D)/(A*AL+B*BE+G*GA)

```

```

X=XP(I)-AL*LAMBDA
Y=YP(I)-BE*LAMBDA
Z=ZP(I)-GA*LAMBDA
GO TO 13
15 DVP=SQRT((X0-XP(I))**2+(Y0-YP(I))**2+(Z0-ZP(I))**2)
CL=(X0-XP(I))/DVP
CM=(Y0-YP(I))/DVP
CN=(Z0-ZP(I))/DVP
LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*CL+B*CM+C*CN)
X=XP(I)-CL*LAMBDA
Y=YP(I)-CM*LAMBDA
Z=ZP(I)-CN*LAMBDA
13 X3(I)=AL1*(X-XC)+BE1*(Y-YC)+GA1*(Z-ZC)
Y3(I)=AL2*(X-XC)+BE2*(Y-YC)+GA2*(Z-ZC)
20 WRITE(2,202) I,X3(I),Y3(I)
202 FORMAT(' ',I4,2(14X,F7.1))
IF(TRAS.NE.1) GO TO 35
DO 9 I3=1,N
CALL LIN(X3(I3),Y3(I3),X3(I3+N),Y3(I3+N))
9 CONTINUE
35 CALL EOF
STOP
END

```

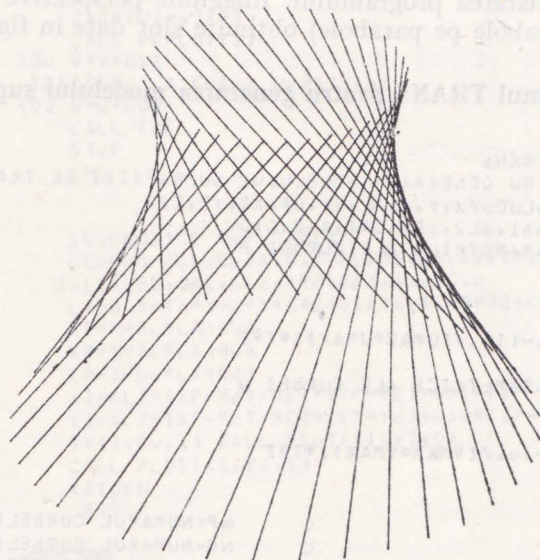


Fig. 4.9

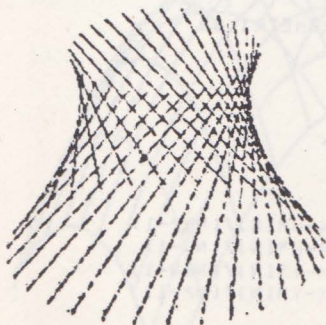


Fig. 4.10

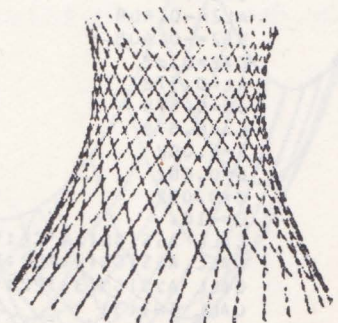


Fig. 4.11

4.5. Program pentru construcția perspectivă a suprafețelor de translație (I)

4.5.1 Baza matematică

Considerăm suprafața de translație definită de curba generatoare $p = p(u)$ care se deplasează pe curba directoare $q = q(v)$ sau reciproc.

Ecuția parametrică a suprafeței este de forma:

$$\begin{aligned} \bar{r} &= \bar{r}(u, v) = \bar{p}(u) + \bar{q}(v) \quad \text{sau} \\ x &= \bar{p}_1(u) + \bar{q}_1(v) \\ y &= \bar{p}_2(u) + \bar{q}_2(v); \quad \text{unde} \quad \begin{cases} U \in [U_{\min}, U_{\max}] \\ V \in [V_{\min}, V_{\max}] \end{cases} \\ z &= \bar{p}_3(u) + \bar{q}_3(v) \end{aligned}$$

Sînt necesare ecuațiile parametrice ale curbelor P și Q , limitele parametrilor, numărul de curbe în translație (10), numărul de puncte pe fiecare astfel de curbă (30), precum și pasul sau distanțele dintre curbe. Alte elemente pot fi citite pe listarea programului. Imaginile perspective ale suprafețelor de translație (parabole pe parabole) obținute sînt date în fig. 4.12 a, b, c, d.

4.5.2 Programul TRANS pentru generarea modelului suprafeței de translație

```

PRUGKAM TRANS
C PROGRAM PENTRU GENERAREA MODELULUI SUPRAFETEI DE TRANSLATIE
COMMON /BLJCU/X,Y,Z,AL,BE,GA,X2,Y2,Z2,
1AL1,BE1,GA1,AL2,BE2,GA2,A,B,C,D
C ECUATIILE PARAMETRICE ALE CURBEI P
C
P1(T)=0.
P2(T)=T
P3(T)=10.-(10./(UMAX*UMAX))*T*T
C
C ECUATIILE PARAMETRICE ALE CURBEI Q
Q1(T)=T
Q2(T)=0.
Q3(T)=0.-(0./(VMAX*VMAX))*T*T
C
X2=0.
Y2=0.
Z2=0.
AL1=0.707
BE1=-0.707
GA1=0
AL2=-0.408
BE2=-0.408
GA2=0.d2
AL=0.578
BE=0.578
GA=0.578
A=0.002
B=0.002
C=0.002
D=-10.
CALL ASSIGN(1,'CR:')
CALL ASSIGN(2,'LP:')
CALL ASSIGN(3,'PP:')
CALL INI(3)
CALL SCA(5,5)
C NP=NUMARUL CURBELOR P
C NQ=NUMARUL CURBELOR Q
C NBP=NUMARUL PUNCTELOR PE CURBA P
C NBQ=NUMARUL PUNCTELOR PE CURBA Q
C
C LIMITELE PARAMETRILOR
NP=10
NQ=10
NBP=30
NBQ=30
UMIN=-10.
UMAX=10.
VMIN=-5.
VMAX=+5.
DU1=(UMAX-UMIN)/(NBP-1)
DU2=(UMAX-UMIN)/(NQ-1)
DV1=(VMAX-VMIN)/(NBQ-1)
DV2=(VMAX-VMIN)/(NP-1)

```


C CURBA GENERATOARE=P, CURBA DIRECTOARE =Q

C

```
V=VMIN
DC 250 J=1,NP
U=UMIN
JC 200 I=1,NbP
X=P1(U)+Q1(V)
Y=P2(U)+Q2(V)
Z=P3(U)+Q3(V)
CALL PRPLOT(I)
```

200 U=U+DU1

C TRECEREA LA CURBA URMATOARE

250 V=V+DV2

C

C CURBA GENERATOARE=Q, CURBA DIRECTOARE=P

C

```
U=UMIN
DC 400 J=1,NU
V=VMIN
JC 300 I=1,NbQ
X=P1(U)+Q1(V)
Y=P2(U)+Q2(V)
Z=P3(U)+Q3(V)
CALL PRPLOT(I)
```

300 V=V+DV1

C TRECEREA LA CURBA URMATOARE

400 U=U+DU2

CALL EOF

STOP

END

SUBROUTINE PRPLOT(I)

COMMON /BLSCU/X,Y,Z,AL,BE,GA,X2,YZ,ZZ,

LA1,BE1,GA1,AL2,BE2,GA2,A,B,C,U

LAMBDA=(A*X+B*Y+C*Z+D)/(A*AL+B*BE+C*GA)

XP=X-AL*LAMBDA

YP=Y-BE*LAMBDA

ZP=Z-GA*LAMBDA

X1=AL1*(XP-X2)+BE1*(YP-Y2)+GA1*(ZP-Z2)

Y1=AL2*(XP-X2)+BE2*(YP-Y2)+GA2*(ZP-Z2)

IF(I.EQ.1) CALL PLOT(X1,Y1,0)

CALL PLOT(X1,Y1,1)

RETURN

END

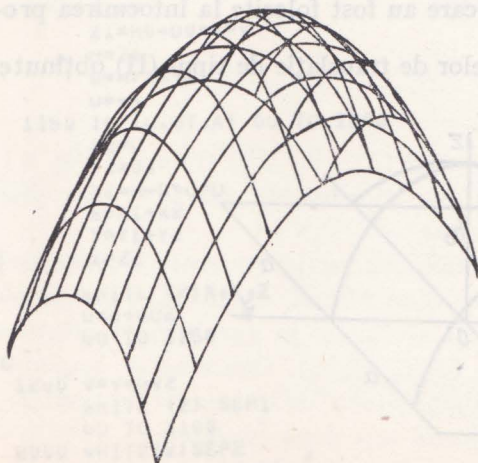


Fig. 4.12 a

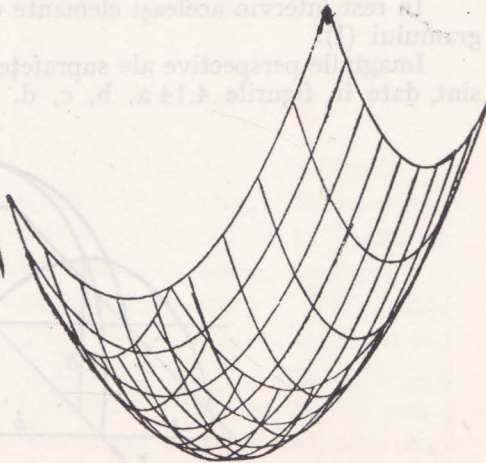


Fig. 4.12 b

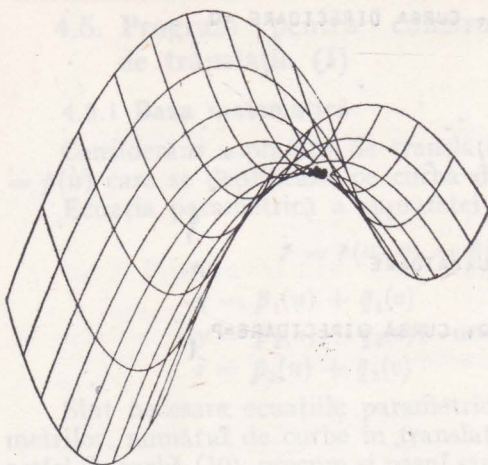


Fig. 4.12 c

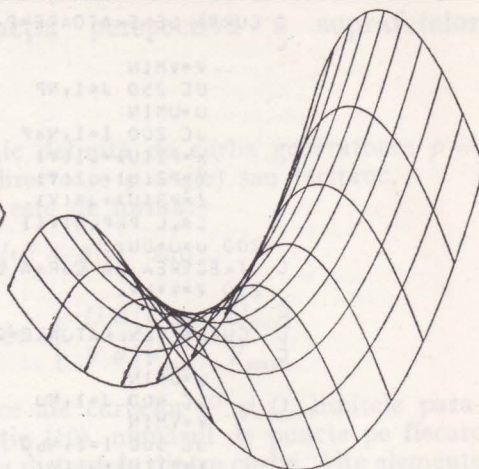


Fig. 4.12 d

4.6. Program pentru construcția perspectivă a suprafețelor de translație (II) speciale care trec prin frontierele domeniului

4.6.1. Baza matematică

Generalizînd programul suprafețelor de translație de tipul (I) putem impune condiția ca suprafața de translație să treacă prin frontierele domeniului. Analizînd fig. 4.13 putem scrie curba (parabola) directoare:

$$\begin{aligned} q_1(v) &= v & -a \leq v \leq +a \\ q_2(v) &= 0 & \text{unde } d_0 = \frac{h_0}{a^2} \\ q_3(v) &= h_0 - d_0 v^2 \end{aligned}$$

De asemenea, pentru curba (parabola) generatoare avem:

$$\begin{aligned} p_1(u) &= \text{const.} & -b \leq u \leq +b \\ p_2(u) &= u & \text{unde } h = q_3(v); \quad d = \frac{h}{b^2} \\ p_3(u) &= h - d \cdot u^2 \end{aligned}$$

În rest intervin aceleași elemente care au fost folosite la întocmirea programului (I).

Imaginile perspective ale suprafețelor de translație de tipul (II) obținute sînt date în figurile 4.14 a, b, c, d.

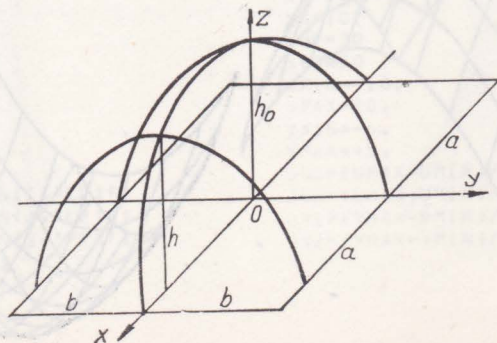


Fig. 4.13

4.6.2. Program TRAN 2

```

PROGRAM TRAN2
C PROGRAM PENTRU CONSTRUCTIA SUPRAFETELOR DE TRANSLATIE SPECIALE
C CAKE TREC PRIN FRONTIERELE DOMENIULUI
  REAL SEP1(3),SEP2(3),X,Y,Z
  DATA SEP1,SEP2/-5555.,-5555.,-5555.,-9999.,-9999.,-9999./
  CALL ASSIGN(2,'TRAN2.DAT')
  NP=10
  E=5.
  DV1=0.2
  TYPE 1
  1 FORMAT(' ', 'DATI DOMENIUL SI PARABOLA DIRECTOARE:A,B,H0!/')
  ACCEPT *,A,B,H0
  DU=H0/(A*A)
  DU1=2*A/(NP-1)
  DU2=0.4
  DV2=2*B/(NP-1)
C GENERAREA CURBEI P
  U=-A
  100 IF(U.GT.A+E)GO TO 1000
  X1=U
  Y1=0.
  Z1=H0-DU*U*U
  H=Z1
  D=H/(B*B)
  V=(-B)
  150 IF (V.GT.B) GO TO 200
  X2=0
  Y2=V
  Z2=H-U*V*V
  X=X1+X2
  Y=Y1+Y2
  Z=Z2
  WRITE (2) X,Y,Z
  V=V+DV1
  GO TO 150
  200 U=U+DV1
  WRITE (2) SEP1
  GO TO 100
C
C GENERAREA CURBEI Q
  1000 DU=H0/(B*B)
  V=-B
  1100 IF(V.GT.B+E)GO TO 2000
  X1=0
  Y1=V
  Z1=H0-DU*V*V
  H=Z1
  U=H/(A*A)
  V=-A
  1150 IF (U.GT.A) GO TO 1200
  X2=U
  Y2=0.
  Z2=H-U*U*U
  X=X1+X2
  Y=Y1+Y2
  Z=Z2
  WRITE (2) X,Y,Z
  U=U+DU2
  GO TO 1150
C
  1200 V=V+DV2
  WRITE (2) SEP1
  GO TO 1100.
  2000 *WRITE(2)SEP2
  CALL CLOSE(2)
  STOP
  END

```

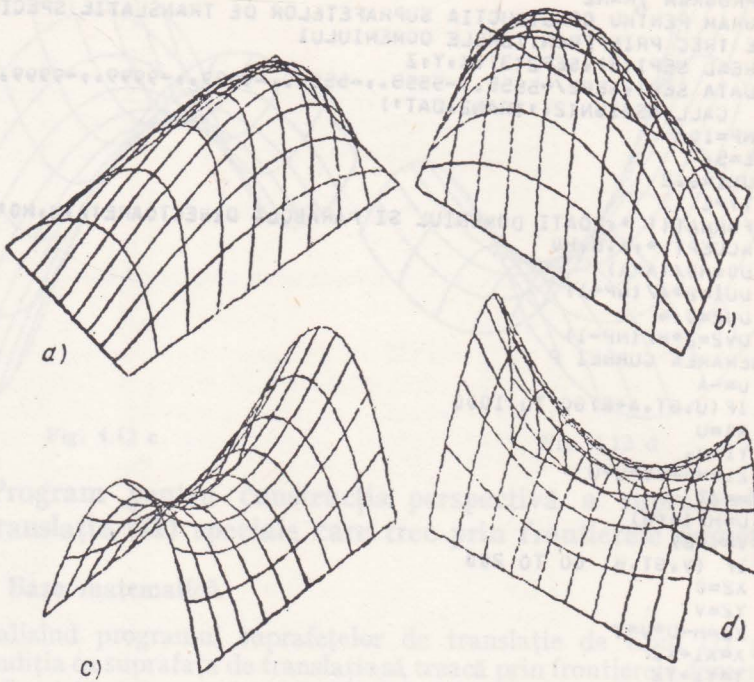


Fig. 4.14

4.7. Program pentru determinarea intersecției dintre o sferă și un paraboloid hiperbolic

4.7.1. Baza matematică

Considerăm sfera definită prin centrul (XQ, YQ, ZQ) și raza R (fig. 4.15). Paraboloidul hiperbolic este definit prin directoarele AB și CD iar planul director Γ al paraboloidului este frontal.

Pentru determinarea liniei de intersecție secționăm ambele suprafețe cu planele $\Gamma' \parallel \Gamma$, pasul secțiunilor plane fiind $2R/GK$. Secțiunile plane în sferă vor fi cercurile de diametre $P_3P'_3$, iar secțiunile în paraboloidul hiperbolic pot fi generatoare de tipul (M_1N_1, M_2N_2, M_3N_3) . Cercurile de secțiune și aceste generatoare vor avea (dacă există) puncte de intersecție de tipul $K_2K'_2$ care constituie linia de intersecție dintre cele două suprafețe.

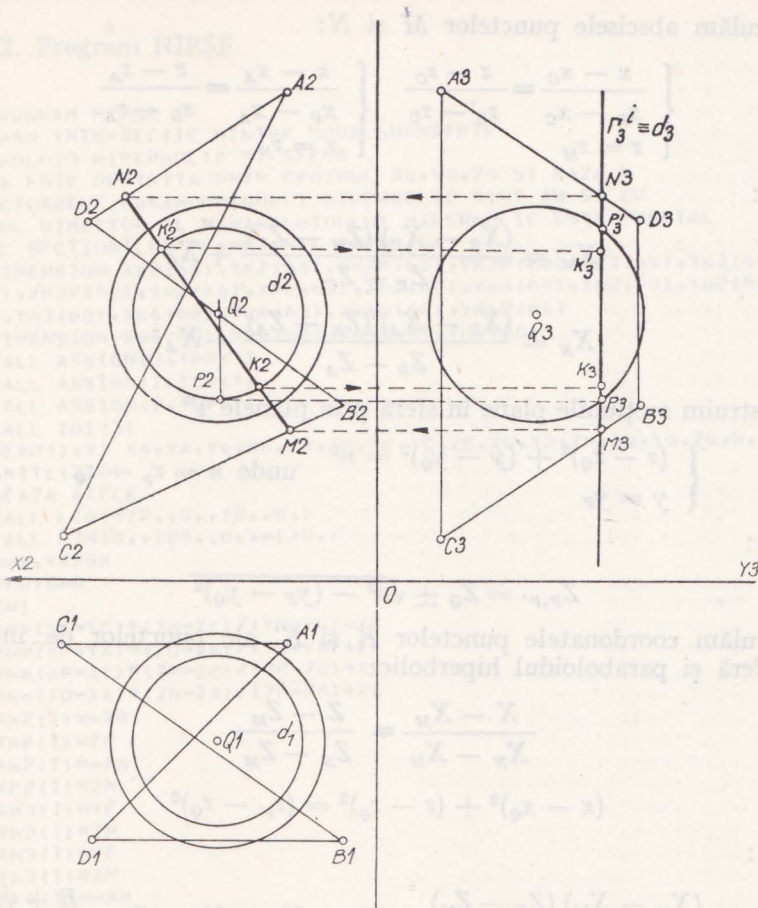


Fig. 4.15

Astfel coordonatele punctelor M și N se obțin rezolvând sistemele de ecuații:

$$\begin{cases} \frac{y - y_C}{y_B - y_C} = \frac{z - z_C}{z_B - z_C} \\ y = y_F \end{cases} \quad \begin{cases} \frac{y - y_A}{y_D - y_A} = \frac{z - z_A}{z_D - z_A} \\ y = y_F \end{cases}$$

Avem:

$$\begin{cases} y_M = y_F \\ z_M = \frac{(y_F - y_C)(z_B - z_C)}{y_B - y_C} + z_C \end{cases}$$

$$\begin{cases} y_N = y_K \\ z_N = \frac{(y_F - y_A)(z_D - z_A)}{y_D - y_A} + z_A \end{cases}$$

Calculăm abscisele punctelor M și N :

$$\begin{cases} \frac{x - x_C}{x_B - x_C} = \frac{z - z_C}{z_B - z_C} \\ z = z_M \end{cases} \quad \begin{cases} \frac{x - x_A}{x_D - x_A} = \frac{z - z_A}{z_D - z_A} \\ z = z_N \end{cases}$$

obținem:

$$X_M = \frac{(X_B - X_C)(Z_M - Z_C)}{Z_B - Z_C} + X_C$$

$$X_N = \frac{(X_D - X_A)(Z_N - Z_A)}{Z_D - Z_A} + X_A$$

Construim secțiunile plane în sferă prin planele Γ^t

$$\begin{cases} (z - z_Q)^2 + (y - y_Q)^2 = n^2 \\ y = y_F \end{cases} \quad \text{unde } n = z_p - z_Q$$

Obținem:

$$Z_{P,P'} = Z_Q \pm \sqrt{n^2 - (y_F - y_Q)^2}$$

Calculăm coordonatele punctelor K și K' ale punctelor de intersecție dintre sferă și paraboloidul hiperbolic:

$$\frac{X - X_M}{X_N - X_M} = \frac{Z - Z_M}{Z_N - Z_M}$$

$$(x - x_Q)^2 + (z - z_Q)^2 = (z_{P'} - z_Q)^2$$

Obținem:

$$X_K = \frac{(X_N - X_M)(Z_K - Z_M)}{Z_N - Z_M} + X_M; \quad Y_K = Y_F; \quad Z_K = \frac{B + \sqrt{D}}{A}$$

$$X_{K'} = \frac{(X_N - X_M)(Z_{K'} - Z_M)}{Z_N - Z_M} + X_M; \quad Y_{K'} = Y_F; \quad Z_{K'} = \frac{B - D}{A}$$

unde:

$$A = (X_N - X_M)^2 + (Z_N - Z_M)^2$$

$$B = -2(X_N - X_M)[Z_M(X_N - X_M) - (X_Q - X_M)(Z_N - Z_M)] + Z_Q(Z_N - Z_M)^2$$

$$C = (Z_N - Z_M)^2[Z_Q^2 - n^2 + (Y_F - Y_Q)^2] + [Z_M(X_N - X_M) - (X_Q - X_M)(Z_N - Z_M)]^2; \quad D = B^2/4 - AC$$

Cu aceste elemente matematice putem întocmi programul. Rezultatele intersecției în triplă proiecție ortogonală sint ilustrate în fig. 4.16 executată la plotter integral.

4.7.2. Program HIPSEF

```

PROGRAM HIPSEF
C PROGRAM INTERSECȚIE ÎNTRU DOUĂ SUPRAFEȚE
C PARABOLOID HIPERBOLIC ȘI SFERĂ
C SFERĂ ESTE DEFINITĂ PRIN CENTRUL X0,Y0,Z0 ȘI RAZA R
C DIRECTOARELE PARABOLOIDULUI HIPERBOLIC SÎNT AB ȘI CD
C PLANUL DIRECTOR AL PARABOLOIDULUI HIPERBOLIC ESTE FRONTAL
C PASUL SECȚIUNILOR PLANE ESTE H=2*R/GK
    DIMENSION XK2(60),YK2(60),XK2P(60),YK2P(60),XK3(60),YK3(60),XK3P(60),
    *YK3P(60),XM2(60),YM2(60),XM3(60),YM3(60),XN2(60),YN2(60),XN3(60),
    *YN3(60),XK4(60),YK4(60),YK4P(60),XK4P(60)
    DIMENSION XM4(60),YM4(60),XN4(60),YN4(60)
    CALL ASSIGN(3,'PP:')
    CALL ASSIGN(1,'CW:')
    CALL ASSIGN(2,'LP:')
    CALL INI(3)
    HEAD(1,9) XA,YA,ZA,XB,YB,ZB,XC,YC,ZC,XD,YD,ZD,X0,Y0,Z0,R,GK
    WRITE(2,8)
C TRASEAZĂ AXELE
    CALL LIN(-70.,0.,70.,0.)
    CALL LIN(0.,120.,0.,-120.)
    H=2.*R/GK
    YF=Y0-R
    I=1
    ZM=(YF-YC)*(ZB-ZC)/(YB-YC)+ZC
    ZN=(YF-YA)*(ZD-ZA)/(YD-YA)+ZA
    XM=(XB-XC)*(ZM-ZC)/(ZB-ZC)+XC
    XN=(XD-XA)*(ZN-ZA)/(ZD-ZA)+XA
    XM2(I)=-XM
    YM2(I)=ZM
    XN2(I)=-XN
    YN2(I)=ZN
    XM3(I)=YF
    YM3(I)=ZM
    XN3(I)=YF
    YN3(I)=ZN
    XM4(I)=-XM
    YM4(I)=-YF
    XN4(I)=-XN
    YN4(I)=-YF
    D=((XN-XM)*(ZM*(XN-XM)+(XQ-XM)*(ZN-ZM))+ZQ*(ZN-ZM)**2)**2-((XN-XM)
    1**2+(ZN-ZM)**2)*((7N-7M)**2+(7Q**2-R**2+(YF-Y0)**2)+
    2(ZM*(XN-XM)+(XQ-XM)*(7N-ZM)**2)
    IF(D) 2,3,3
    2 WRITE(2,10)
    10 FORMAT(15X,'NU EXISTA REZOLVARE!')
    GO TO 1
    3 Z1=((XN-XM)*(ZM*(XN-XM)+(XQ-XM)*(ZN-ZM))+ZQ*(ZN-ZM)**2+SQRT(D))/
    3*((XN-XM)**2+(ZN-ZM)**2)
    X1=(XN-XM)*(Z1-ZM)/(7N-7M)+XM
    Z2=((XN-XM)*(ZM*(XN-XM)+(XQ-XM)*(ZN-ZM))+ZQ*(ZN-ZM)**2-SQRT(D))/
    4*((XN-XM)**2+(ZN-ZM)**2)
    X2=(XN-XM)*(Z2-ZM)/(7N-7M)+XM
    XK2(I)=-X1
    YK2(I)=Z1
    XK2P(I)=-X2
    YK2P(I)=Z2

```

```

XK3(I)=YF
YK3(I)=Z2
XK3P(I)=YF
YK3P(I)=Z1
XK4(I)=-X1
YK4(I)=-YF
XK4P(I)=-X2
YK4P(I)=-YF
I=I+1
WRITE(2,7) YF,X1,X2,Z1,Z2
1 YF=YF+H
IF(YF-(YQ+K)) 4,4,5
9 FORMAT(8F10,3)
8 FORMAT(15X,'PUNCT',4X,'INTERSECȚIE',/9X,'Y',10X,'X1',10X,'X2',10X,'
5'Z1',10X,'Z2')
7 FORMAT(1X,5F12,1)
5 CALL TEX(-71..1..0..2.5.0,'X2',2)
CALL TEX(71..1..0..2.5.0,'Y3',2)
CALL TFX(1..101..0..2.5.0,'Z23',3)
CALL TEX(30..-10..0..2.5.0,'PROGRAM HIPSF',13)
CALL TEX(10..-20..0..2.5.0,'INTERSECȚIA DINTRE',18)
CALL TEX(30..-30..0..2.5.0,'SFERA S1',8)
CALL TEX(10..-40..0..2.5.0,'PARABOLOID HIPERBOLIC',21)
N=N-1
CALL PLOT(XM3(I),YM3(I),0)
CALL PLOT(XN3(I),YN3(I),1)
CALL PLOT(XN3(N),YN3(N),1)
CALL PLOT(XM3(N),YM3(N),1)
CALL PLOT(XM3(I),YM3(I),1)
CALL PLOT(XM2(I),YM2(I),0)
CALL PLOT(XN2(I),YN2(I),1)
CALL PLOT(XN2(N),YN2(N),1)
CALL PLOT(XM2(N),YM2(N),1)
CALL PLOT(XM2(I),YM2(I),1)
CALL PLOT(XM4(I),YM4(I),0)
CALL PLOT(XN4(I),YN4(I),1)
CALL PLOT(XN4(N),YN4(N),1)
CALL PLOT(XM4(N),YM4(N),1)
CALL PLOT(XM4(I),YM4(I),1)
DO 20 J=1,I,5
CALL PLOT(XM3(J),YM3(J),0)
CALL PLOT(XN3(J),YN3(J),1)
CALL PLOT(XM2(J),YM2(J),0)
CALL PLOT(XN2(J),YN2(J),1)
CALL PLOT(XM4(J),YM4(J),0)
CALL PLOT(XN4(J),YN4(J),1)
20 CONTINUE
CALL CIS(-XQ,-YQ,0)
CALL CIS(-XQ,-YQ,.5)
CALL PLOT(XK2(1),YK2(1),0)
CALL PLG(XK2,YK2,1,N,0)
CALL PLOT(XK3(1),YK3(1),0)
CALL PLG(XK3,YK3,1,N,0)
CALL PLOT(XK2P(1),YK2P(1),0)
CALL PLG(XK2P,YK2P,1,N,0)
CALL PLOT(XK3P(1),YK3P(1),0)
CALL PLG(XK3P,YK3P,1,N,0)
CALL PLOT(XK4(1),YK4(1),0)
CALL PLG(XK4,YK4,1,N,0)
CALL PLOT(XK4P(1),YK4P(1),0)
CALL PLG(XK4P,YK4P,1,N,0)
CALL CIS(-XQ,ZQ,0)
CALL CIS(YQ,ZQ,R)
CALL CIS(-XQ,ZQ,.5)
CALL CIS(YQ,ZQ,.5)
CALL FUF
STOP
END)

```

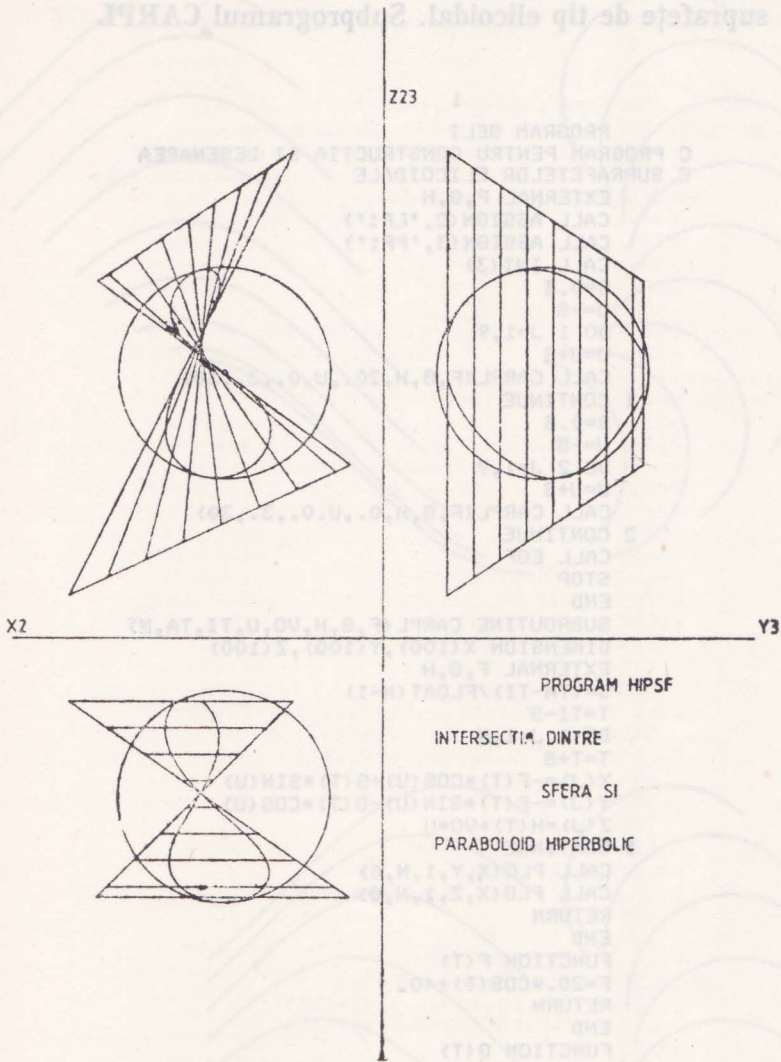



Fig. 4.16

4.8. Program SELI pentru construcția și desenarea anumitor suprafețe de tip elicoidal. Subprogramul CARPL

```

PROGRAM SELI
C PROGRAM PENTRU CONSTRUCTIA SI DESENAREA
C SUPRAFETELOR ELICOIDALE
EXTERNAL F,G,H
CALL ASSIGN(2,'LF:')
CALL ASSIGN(3,'PP:')
CALL INI(3)
S=0.3
U=-S
DO 1 J=1,9
U=U+S
CALL CARPL(F,G,H,20.,U,0.,3.,30)
1 CONTINUE
S=0.3
U=-S
DO 2 J=1,9
U=U+S
CALL CARPL(F,G,H,0.,U,0.,3.,30)
2 CONTINUE
CALL EOF
STOP
END
SUBROUTINE CARPL(F,G,H,VO,U,TI,TA,N)
DIMENSION X(100),Y(100),Z(100)
EXTERNAL F,G,H
S=(TA-TI)/FLOAT(N-1)
T=TI-S
DO 1 J=1,N
T=T+S
X(J)=-F(T)*COS(U)+G(T)*SIN(U)
Y(J)=-F(T)*SIN(U)-G(T)*COS(U)
Z(J)=H(T)+VO*U
1 CONTINUE
CALL PLG(X,Y,1,N,0)
CALL PLG(X,Z,1,N,0)
RETURN
END
FUNCTION F(T)
F=20.*COS(T)+40.
RETURN
END
FUNCTION G(T)
G=20.*SIN(T)
RETURN
END
FUNCTION H(T)
H=20.*T
RETURN
END

```

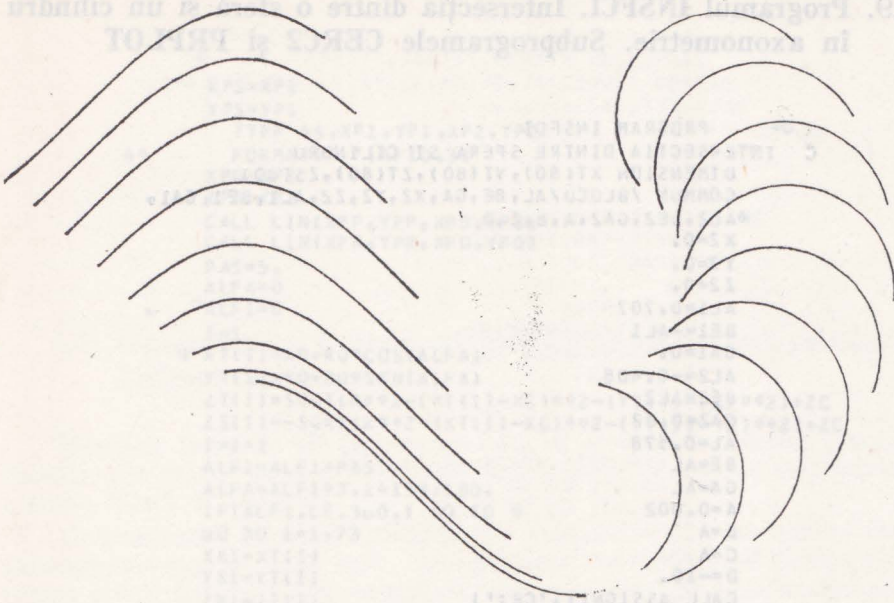


Fig. 4.17

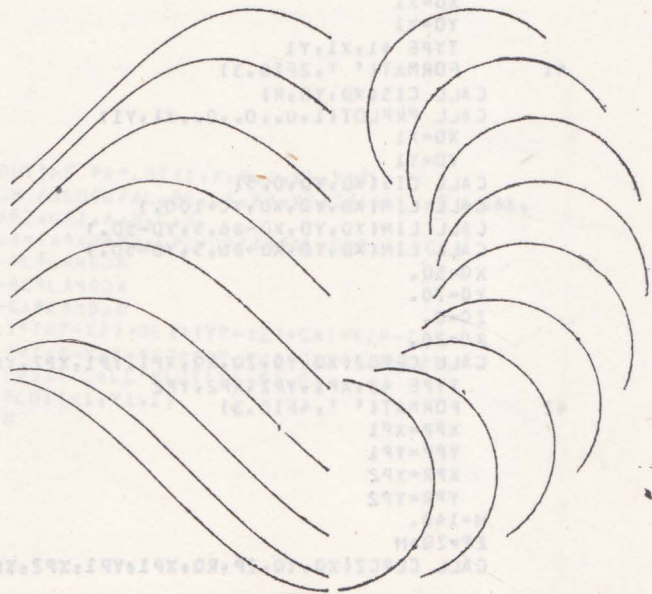


Fig. 4.18

4.9. Programul INSFCI. Intersecția dintre o sferă și un cilindru în axonometrie. Subprogramele CERC2 și PRPLOT

```

PROGRAM INSFCI
C INTERSECȚIA DINTRE SFERA ȘI CILINDRU
DIMENSION XT(80),YT(80),ZT(80),ZS(80)
COMMON /BLOCU/AL, BE, GA, XZ, YZ, ZZ, AL1, BE1, GA1,
*AL2, BE2, GA2, A, B, C, D
XZ=0.
YZ=0.
ZZ=0.
AL1=0.707
BE1=-AL1
GA1=0.
AL2=-0.408
BE2=AL2
GA2=0.82
AL=0.578
BE=AL
GA=AL
A=0.002
B=A
C=A
D=-10.
CALL ASSIGN(1, 'CR:')
CALL INI(3)
ACCEPT 2, XC, YC, ZC, R
2 FORMAT(4F10.3, 40X)
TYPE `40, XC, YC, ZC, R
40 FORMAT(' ', 4F10.3)
CALL CERC2(XC, YC, ZC, R, XP1, YP1, XP2, YP2)
CALL DEG
CALL PRPLOT(1, XC, YC, ZC, X1, Y1)
XD=X1
YD=Y1
TYPE 41, X1, Y1
41 FORMAT(' ', 2F10.3)
CALL CIS(XD, YD, R)
CALL PRPLOT(1, 0., 0., 0., X1, Y1)
XD=X1
YD=Y1
CALL CIS(XD, YD, 0.5)
CALL LIN(XD, YD, XD, YD+100.)
CALL LIN(XD, YD, XD-86.5, YD-50.)
CALL LIN(XD, YD, XD+86.5, YD-50.)
XQ=50.
YQ=70.
ZQ=0.
RQ=20.
CALL CERC2(XQ, YQ, ZQ, RQ, XP1, YP1, XP2, YP2)
TYPE 42, XP1, YP1, XP2, YP2
42 FORMAT(' ', 4F10.3)
XPP=XP1
YPP=YP1
XPR=XP2
YPR=YP2
H=140.
ZP=ZQ+H
CALL CERC2(XQ, YQ, ZP, RQ, XP1, YP1, XP2, YP2)

```

```

XPS=XP1
YPS=YP1
  TYPE 44,XP1,YP1,XP2,YP2
44  FORMAT(' ',4F10.3)
XPQ=XP2
YPQ=YP2
CALL LIN(XPP, YPP, XPS, YPS)
CALL LIN(XPR, YPR, XPQ, YPQ)
PAS=5.
ALFA=0
ALF1=0
I=1
9  XT(I)=XQ+RQ*COS(ALFA)
  YT(I)=YQ+RQ*SIN(ALFA)
  ZT(I)=SQRT(R**2-(XT(I)-XC)**2-(YT(I)-YC)**2)+ZC
  ZS(I)=-SQRT(R**2-(XT(I)-XC)**2-(YT(I)-YC)**2)+ZC
  I=I+1
  ALF1=ALF1+PAS
  ALFA=ALF1*3.14159/180.
  IF(ALF1.LE.360.) GO TO 9
  J0 30 I=1,73
  XK1=XT(I)
  YK1=YT(I)
  ZK1=ZT(I)
  CALL PRPLOT(I, XK1, YK1, ZK1, X1, Y1)
30 CONTINUE
  J0 31 I=1,73
  XK2=XT(I)
  YK2=YT(I)
  ZK2=ZS(I)
  CALL PRPLOT(I, XK2, YK2, ZK2, X1, Y1)
31 CONTINUE
  STOP
  END

```

```

SUBROUTINE PRPLOT(I, X, Y, Z, X1, Y1)
COMMON /BLOCU/AL, BE, GA, XZ, YZ, ZZ, AL1, BE1, GA1,
*AL2, BE2, GA2, A, B, C, D
LAMBDA=(A*X+B*Y+C*Z+D)/(A*AL+B*BE+C*GA)
XP=X-AL*LAMBDA
YP=Y-BE*LAMBDA
ZP=Z-GA*LAMBDA
X1=AL1*(XP-XZ)+BE1*(YP-YZ)+GA1*(ZP-ZZ)
Y1=AL2*(XP-XZ)+BE2*(YP-YZ)+GA2*(ZP-ZZ)
IF(I.EQ.1) CALL PLOT(X1, Y1, 0)
CALL PLOT(X1, Y1, 1)
RETURN
END

```

```

SUBROUTINE CERC2(XC,YC,ZC,R,XP1,YP1,XP2,YP2)
  DIMENSION XT(80),YT(80),ZT(80)
  COMMON /BLUCD/AL,BE,GA,X2,Y2,Z2,AL1,BE1,GA1,
  *AL2,BE2,GA2,A,B,C,D
  PAS=5.
  ALFA=0
  ALF1=0
  I=1
5  XT(I)=XC+R*COS(ALFA)
   YT(I)=YC+R*SIN(ALFA)
   ZT(I)=ZC
   I=I+1
   ALF1=ALF1+PAS
   ALFA=ALF1*3.1459/180.
   IF(ALF1.LE.360.) GO TO 5
  DO 8 I=1,73
   XK=XT(I)
   YK=YT(I)
   ZK=ZT(I)
   CALL PRPLOT(I,XK,YK-ZK,X1,Y1)
   IF(I.EQ.27) GO TO 6
   IF(I.EQ.63) GO TO 7
  GO TO 8
6  XP1=X1
   YP1=Y1
  GO TO 8
7  XP2=X1
   YP2=Y1
8  CONTINUE
   RETURN
   END

```

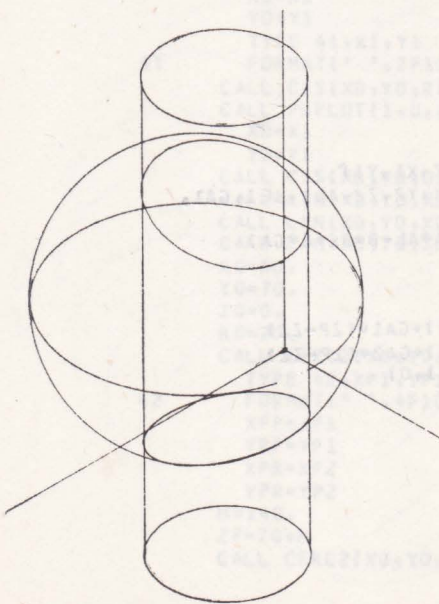


Fig. 4.19 a

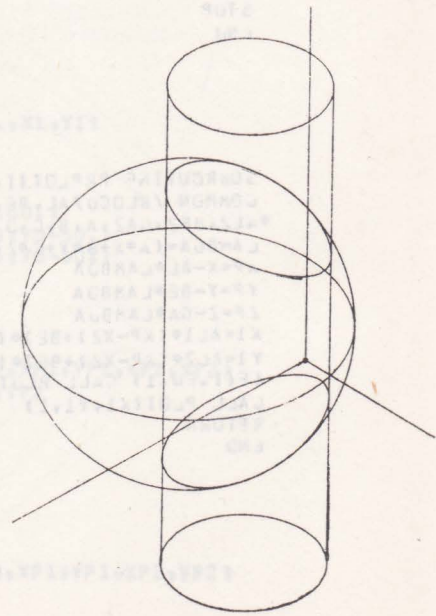


Fig. 4.19 b

5.1. Generalizarea metodei coordonatelor perspective pe un tablou înclinat oarecare

Metodele teoretice ale perspectivei au un pronunțat caracter practic, iar operațiunile grafice implicate în construcția unei perspective sînt într-o succesiune logică și naturală.

Generalizarea metodei coordonatelor perspective pe un tablou înclinat oarecare presupune, mai întii, determinarea coordonatelor perspective, în raport cu un triedru tridreptunghiular fix, apoi determinarea acelorași coordonate în raport cu reperul de referință asociat tabloului de perspectivă.

5.1.1. Determinarea coordonatelor perspective absolute

În raport cu triedrul tridreptunghic fix $OXYZ$ se consideră punctul de vedere $\Omega(X_0, Y_0, Z_0)$, planul oarecare reprezentat prin ecuația

$$AX + BY + CZ + D = 0$$

ales ca tablou înclinat de perspectivă și punctele

$$P_i(X_i, Y_i, Z_i) \quad i = 1, 2, 3, \dots, n$$

ale spațiului real sau intermediar (fig. 5.1).

Ecuția razei vizuale care trece prin punctul de vedere $\Omega(X_0, Y_0, Z_0)$ și punctele $P_i(X_i, Y_i, Z_i)$ este reprezentată prin ecuația:

$$\frac{X - X_0}{l} = \frac{Y - Y_0}{m} = \frac{Z - Z_0}{n}$$

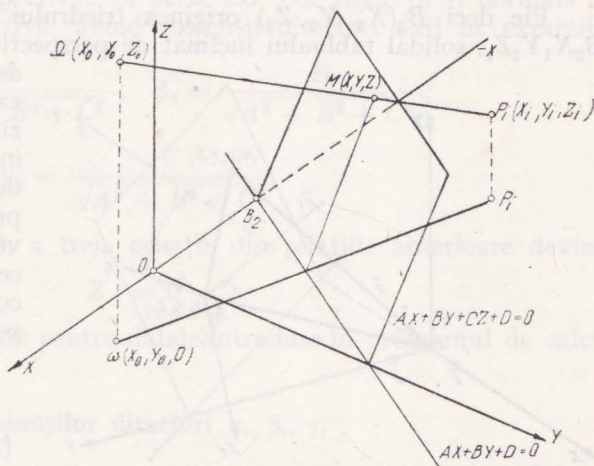


Fig. 5.1

unde cantitățile l, m, n sînt parametrii directori ai dreptei reprezentînd raza vizuală și se calculează cu relațiile:

$$l = \frac{X_i - X_0}{d\Omega P_i}; \quad m = \frac{Y_i - Y_0}{d\Omega P_i}; \quad n = \frac{Z_i - Z_0}{d\Omega P_i}$$

unde s-a notat prin $d\Omega P_i$ distanța dintre punctele în spațiu Ω și P_i determinată prin relația

$$d\Omega P_i = \sqrt{(X_i - X_0)^2 + (Y_i - Y_0)^2 + (Z_i - Z_0)^2}$$

Pentru a găsi coordonatele perspective absolute $M(X, Y, Z)$, trebuie rezolvat sistemul format din ecuația planului și ecuațiile razelor vizuale. După unele transformări elementare se obțin expresiile coordonatelor absolute sub forma:

$$\begin{cases} X = X_0 - \lambda l \\ Y = Y_0 - \lambda m \\ Z = Z_0 - \lambda n \end{cases}$$

unde cantitatea λ se determină cu expresia:

$$\lambda = \frac{AX_0 + BY_0 + CZ_0 + D}{Al + Bm + Cn}$$

Reprezentarea perspectivei pe tabloul de perspectivă prin aceste coordonate este greoaie. De aceea, este mai util și chiar necesar, să fie exprimate coordonatele perspectivei $M(X, Y, Z)$, în raport cu un triedru tridreptunghic, de referință, asociat tabloului înclinat de perspectivă reprezentat prin ecuația planului dat.

5.1.2. Determinarea coordonatelor perspective relative

Fie deci $B_2(X_2, Y_2, Z_2)$ originea triedrului tridreptunghic de referință $B_2X_1Y_1Z_1$, solidar tabloului înclinat de perspectivă (fig. 5.2). Pentru a trece de la sistemul cartezian oarecare $OXYZ$ la sistemul cartezian $B_2X_1Y_1Z_1$ se pot utiliza mai multe posibilități oferite de geometria analitică. Se preferă proiectarea ecuației vectoriale a vectorilor ce leagă cele două origini ale sistemelor considerate și punctul de perspectivă:

$$\begin{aligned} \vec{r} &= \vec{R} - \vec{r}_2 \\ X_1\vec{i}_1 + Y_1\vec{j}_1 + Z_1\vec{k}_1 &= \\ (X - X_2)\vec{i} + (Y - Y_2)\vec{j} + \\ &+ (Z - Z_2)\vec{k} \end{aligned}$$

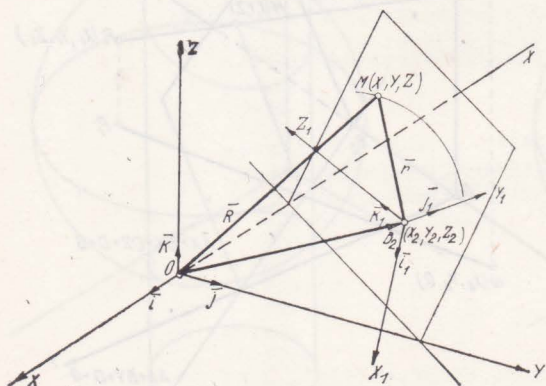


Fig. 5.2

Proiectînd vectorii sistemului $OXYZ$ pe triedrul $B_2X_1Y_1Z_1$ avem

$$\begin{aligned}\vec{i} &= \vec{i}_1 \cos(\vec{i}, \vec{i}_1) + \vec{j}_1 \cos(\vec{i}, \vec{j}_1) + \vec{k}_1 \cos(\vec{i}, \vec{k}_1) = \\ &= \alpha_1 \vec{i}_1 + \beta_1 \vec{j}_1 + \gamma_1 \vec{k}_1\end{aligned}$$

Înlocuind și identificînd, se obțin coordonatele perspectivei $M(X_1, Y_1, Z_1)$ în raport cu noul triedru tridreptunghic de referință $B_2X_1Y_1Z_1$ solidar cu tabloul înclinat de perspectivă:

$$\begin{aligned}X_1 &= (X - X_2) \alpha_1 + (Y - Y_2) \beta_1 + (Z - Z_2) \gamma_1 \\ Y_1 &= (X - X_2) \alpha_2 + (Y - Y_2) \beta_2 + (Z - Z_2) \gamma_2 \\ Z_1 &= (X - X_2) \alpha_3 + (Y - Y_2) \beta_3 + (Z - Z_2) \gamma_3\end{aligned}$$

unde

$$\alpha_1 = \cos(X_1, X); \quad \beta_1 = \cos(X_1, Y);$$

$$\alpha_2 = \cos(Y_1, X); \quad \beta_2 = \cos(Y_1, Y)$$

$$\alpha_3 = \cos(Z_1, X); \quad \beta_3 = \cos(Z_1, Y)$$

$$\gamma_1 = \cos(X_1, Z)$$

$$\gamma_2 = \cos(Y_1, Z)$$

$$\gamma_3 = \cos(Z_1, Z)$$

Aceste relații generalizează metoda coordonatelor perspective, în perspectiva pe un tablou înclinat oarecare.

În aceste relații trebuie remarcat faptul că, deocamdată, numai originea B_2 a noului triedru tridreptunghic aparține tabloului de perspectivă. De aceea, se vor determina mai departe condițiile ca planul $X_1B_2Y_1$ să coincidă cu tabloul înclinat de perspectivă. În acest caz axa B_2Z_1 va fi normala la planul (tabloul) de perspectivă, avînd cosinușii directori dați în expresiile

$$\alpha_3 = \frac{A}{\sqrt{A^2 + B^2 + C^2}}; \quad \beta_3 = \frac{B}{\sqrt{A^2 + B^2 + C^2}};$$

$$\gamma_3 = \frac{C}{\sqrt{A^2 + B^2 + C^2}}$$

Ca urmare acestui fapt, a treia ecuație din relațiile anterioare devine.

$$Z_1 = 0$$

și ea poate servi ca o verificare pentru datele introduse în programul de calcul

5.1.3. Determinarea cosinușilor directori $\alpha_i, \beta_i, \gamma_i$

Pentru comoditatea scrierii programului de calcul, este necesară determinarea în prealabil a cosinușilor directori $\alpha_i, \beta_i, \gamma_i$ ($i = 1, 2$).

Urma tabloului înclinat de perspectivă pe planul XOY se obține pentru $Z = 0$ și ea poate fi aleasă ca axă B_2X_1 a absciselor perspective în tablou. Această alegere conduce, însă, la necesitatea precizării sensurilor pozitive astfel încât, de exemplu $\sphericalangle B_2X_1, OX) = \sphericalangle$ ascuțit și deci $\alpha_1 > 0$ (fig. 5.3 și 5.4).

Discuția cu privire la acest semn al cosinurilor directori poate fi concentrată în tabela următoare:

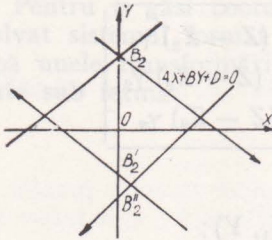


Fig. 5.3

Cadranul	I	II	III	IV
α_1	>0	>0	<0	<0
β_1	<0	>0	>0	<0
γ_1	$=0$	$=0$	$=0$	$=0$

unde

$$\alpha_1 = \frac{|B|}{\sqrt{A^2 + B^2}}$$

$$\beta_1 = \frac{|A|}{\sqrt{A^2 + B^2}}$$

Pentru determinarea analitică a cosinurilor directori ai axei B_2Y_1 se procedează în felul următor (fig. 5.4).

Axa B_2Y_1 rezultă ca dreapta de intersecție a tabloului înclinat de perspectivă cu un plan vertical trecând prin punctul $B_2(0, b, 0)$ și în consecință ecuațiile celor două plane vor fi ecuațiile axei B_2Y_1 . Considerând tăieturile planului înclinat de perspectivă pe axele de coordonate ca fiind lungimile a, b , și c , ecuațiile celor două plane sînt:

$$aX - bY + b^2 = 0$$

$$bcX + acY + baZ - abc = 0$$

din care rezultă cu ușurință ecuațiile reduse ale dreptei

$$X = -\frac{a}{c} \frac{b^2}{a^2 + b^2} Z; X = -\lambda bz$$

sau

$$Y = -\frac{b}{c} \frac{a^2}{a^2 + b^2} Z + b;$$

$$Y = -\lambda az + b$$

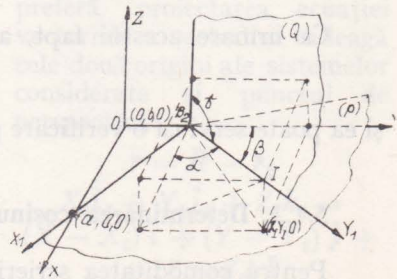


Fig. 5.4

Pentru a determina cosinuşii directori $\alpha_2, \beta_2, \gamma_2$ se intersectează axa B_2Y_1 cu un plan orizontal de cotă $Z = 1/\lambda b$, rezultând din ecuaţiile reduse, abscisa şi respectiv depărtarea punctului de intersecţie I (fig. 5.4).

$$X_1 = -I; \quad Y_1 = -\frac{a}{b} + b$$

şi deci

$$\alpha_2 = \frac{|a|b^2}{\sqrt{(a^2 + b^2)(a^2b^2 + a^2c^2 + b^2c^2)}}$$

$$\gamma_2 = \sin \theta$$

$$\beta_2 = \frac{|b|a^2}{\sqrt{(a^2 + b^2)(a^2b^2 + a^2c^2 + b^2c^2)}}$$

iar pentru cazul $a = b$:

$$\alpha_2 = \beta_2 = \frac{a}{\sqrt{2(a^2 + 2c^2)}}$$

unde $\alpha_2, \beta_2, \gamma_2$ sînt definiţi prin relaţiile de mai sus iar θ este unghiul cuprins între tabloul înclinat de perspectivă şi planul XOY al triedrului tridreptunghic fix iniţial sau în termenii geometriei descriptive, unghiul liniei de cea mai mare pantă al tabloului faţă de planul XOY .

În privinţa semnului, acesta se stabileşte conform tabeli de mai jos:

Cosinuşii directori	$\gamma_3 > 0$				$\gamma_3 < 0$			
	I	II	III	IV	I	II	II	IV
α_2	>0	<0	<0	>0	<0	>0	>0	<0
β_2	>0	>0	<0	<0	<0	<0	>0	>0

Cu aceasta studiul generalizării metodei coordonatelor perspective în perspectiva pe un tablou înclinat oarecare poate fi aplicat în programul de calcul al unui calculator electronic.

5.1.4 Observaţia 1. Aceste valori pentru ultimele relaţii pot fi, evident mai puţin precis, determinate şi grafic pe epură (fig. 5.5). Pentru unghiurile $\sphericalangle (X_1, X)$; $\sphericalangle (X_1, Y)$; $\sphericalangle (X_1, Z)$ care conduc la cosinuşii directori $\alpha_1; \beta_1; \gamma_1$ măsurătoare se poate face direct pe epură, ţinându-se seama de semnul dat.

Pentru unghiurile $\sphericalangle (Y_1, X)$; $\sphericalangle (Y_1, Y)$; $\sphericalangle (Y_1, Z) = \frac{\pi}{2} - \theta$, sînt necesare două rabateri pe planul XOY ale punctului (I, i) situat pe o paralelă la linia de cea mai mare pantă a tabloului înclinat de perspectivă dusă prin originea O . Se observă că $O_1 \perp X_1$ şi că cota $iI = -c$, deci egală în valoare absolută cu tăietura tabloului de perspectivă pe axa OZ . Construind cele două rabateri i_0 şi i'_0 ale punctului I , axele de rabateri fiind $-XOX$ şi respectiv $-YOY$ rezultă grafic unghiurile care conduc la valorile de calcul ale cosinuşilor directori α_2, β_2 şi γ_2 .

ghiul dintre tablou și planul XOY . Aceste relații generalizează și ele metoda coordonatelor perspective în perspectiva pe un tablou înclinat oarecare. Ele pot înlocui relațiile generale și pot totodată constitui un mijloc util de verificare.

Pentru anumite valori particulare, se pot obține expresii simplificate. De exemplu, pentru $\varphi = 45^\circ$; $\theta = 60^\circ$ rezultă:

$$\begin{cases} X_1 = (Y - b - X) \frac{\sqrt{2}}{2} + \frac{2}{\sqrt{2}} X \\ Y_1 = \frac{2}{\sqrt{3}} Z \end{cases}$$

5.1.6 Observația 3. Relațiile care generalizează metoda coordonatelor perspective, în perspectiva pe un tablou înclinat oarecare, pot conduce la două particularizări importante și anume la utilizarea aceluiași relații pentru determinarea perspectivei pe un tablou vertical oarecare sau pe un tablou vertical frontal sau orizontal.

5.2. Proiecția centrală sau paralelă a spațiului $S^{(3)}$

5.2.1. Proiecția centrală a spațiului $S^{(3)}$ pe planul P

Se observă că ecuațiile razelor vizuale, ce unesc punctul de vedere $\Omega(X_0, Y_0, Z_0)$ cu punctele $P_i(X_i, Y_i, Z_i)$ ale spațiului $S^{(3)}$ (vezi fig. 5.1), pot avea și forma:

$$\frac{X - X_i}{l} = \frac{Y - Y_i}{m} = \frac{Z - Z_i}{n},$$

care este echivalentă. În mod similar, se pot transforma și celelalte expresii. avem:

$$d\Omega P_i = \sqrt{(X_0 - X_i)^2 + (Y_0 - Y_i)^2 + (Z_0 - Z_i)^2}$$

$$l = \frac{X_0 - X_i}{d\Omega P_i}; \quad m = \frac{Y_0 - Y_i}{d\Omega P_i}; \quad n = \frac{Z_0 - Z_i}{d\Omega P_i}$$

$$\lambda = \frac{AX_i + BY_i + CZ_i + D}{Al + Bm + Cn}$$

$$X = X_i - l\lambda; \quad Y = Y_i - m\lambda; \quad Z = Z_i - n\lambda$$

Coordonatele perspective relative rămân nemodificate, iar determinarea cosinuşilor directori ai axelor solidare planului P , în raport cu triedrul fundamental (față de care s-au referit punctele P_i), se face conform 5.1.3.

Relațiile generale stabilesc corespondența univocă între punctele spațiului tridimensional $S^{(3)}$ și punctele planului dat P .

Punctele definite prin coordonatele generale se numesc perspectivele punctelor corespunzătoare din spațiu, iar figura obținută prin unirea lor organizată se numește perspectiva liniară a figurii din spațiu definită de punctele $P_i(X_i, Y_i, Z_i)$.

Dacă pe planul P se proiectează triedrul de referință al punctelor P_i , se obțin pe acest plan trei axe concurente.

Dacă în raport cu acest nou sistem de axe se reprezintă punctele P_i dar la scările obținute prin proiectarea centrală pe același plan dat P , a scării comune axelor triedrului fundamental, se obține reprezentarea axonometrică a spațiului tridimensional dat $S^{(3)}$.

5.2.2. Proiecția paralelă a spațiului $S^{(3)}$ pe planul P

Punctul de vedere $\Omega(X_0, Y_0, Z_0)$ poate fi înlocuit prin direcția Δ , definită prin cosinușii directori α, β , și γ , în raport cu triedrul tridreptunghic fundamental al spațiului $S^{(3)}$.

În condițiile acestor date ecuațiile proiectantelor punctelor P_i și paralele cu direcția Δ (fig. 5.1) sînt date tot de ecuația în care cosinușii directori l, m și n sînt înlocuiți prin α, β și γ ale căror valori sînt date prin enunțul problemei:

$$\frac{X - X_i}{\alpha} = \frac{Y - Y_i}{\beta} = \frac{Z - Z_i}{\gamma}$$

În continuare, din relații se iau în considerare numai ultimele două expresii, în care, de asemenea, cosinușii directori l, m, n , sînt înlocuiți prin cosinușii directori ai direcției date α, β și γ :

$$\lambda = \frac{AX_i + BY_i + CZ_i + D}{\alpha A + \beta B + \gamma C}$$

$$X = X_i + \alpha\lambda; \quad Y = Y_i + \beta\lambda; \quad Z = Z_i + \gamma\lambda$$

Coordonatele perspective relative rămîn și în acest caz neschimbate, iar determinarea cosinușilor directori ai axelor solidale planului P în raport cu triedrul fundamental, se face analog. Și în acest caz relațiile stabilesc corespondența univocă între punctele spațiului $S^{(3)}$ și punctele planului dat P .

Punctele definite prin coordonatele generale se numesc perspectivele paralele ale punctelor corespunzătoare din spațiu, iar figura obținută prin unirea lor organizată se numește perspectiva paralelă liniară a figurii din spațiu definită de punctele $P_i(X_i, Y_i, Z_i)$.

În funcție de poziția planului P față de triedrul fundamental și a direcției Δ față de planul P , rezultă proiecțiile ortogonală și respectiv paralelă.

Astfel, dacă $(P) \in (H), (V)$ sau (L) și $\bar{\Delta} \times \bar{k} = 0, \bar{\Delta} \times \bar{j} = 0$ sau $\bar{\Delta} \times \bar{i} = 0$ se obține proiecția paralelă ortogonală iar dacă (P) este oarecare iar $\bar{\Delta} \times \bar{N} \neq 0$ (unde N este normala planului P) se obține proiecția paralelă oblică.

Dacă pe planul P se proiectează triedrul de referință al punctelor P_i se obțin pe acest plan trei axe concurente și coplanare. Dacă în raport cu acest nou sistem de axe se reprezintă punctele P_i , dar la scările obținute prin proiectarea paralelă pe același plan dat P , a scării comune axelor triedrului principal se obține reprezentarea axonometrică a spațiului tridimensional dat $S^{(3)}$.

În funcție de poziția relativă plan — triedrul principal și plan — direcția dată Δ se pot obține axonometria ortogonală și axonometria oblică. Astfel, dacă (P) este oarecare și $\bar{\Delta} \times \bar{N} = 0$ se obține reprezentarea axonometrică ortogonală, care în funcție de tăieturile planului P pe axele triedrului principal poate fi izometrică, dimetrică sau anizometrică. Dacă (P) este oa-

recare, iar $\bar{\Delta} \times \bar{N} \neq 0$ se obține reprezentarea axonometrică oblică care pentru poziții particulare ale planului P poate fi cavalieră — orizontală sau cavalieră — frontală.

Este util ca pozițiile posibile ale planului de proiecție P împreună cu cosinuşii directori ai triedrului de referință solidar acestui plan și respectiv ai direcției de proiecție Δ (proiecția paralelă) să fie sistematizate pe un tablou.

5.2.3. Perspectiva pe tablou înclinat

După poziția tabloului înclinat de perspectivă în raport cu punctul de vedere și cu obiectul dat se deosebesc: perspectiva ascendentă (cu punctul de fugă al verticalelor situat deasupra liniei orizontului) și perspectiva descendentă (cu punctul de fugă al verticalelor situat sub linia orizontului).

Perspectiva ascendentă a unui obiect foarte înalt apare în situația în care observatorul privește construcția de jos și destul de aproape.

Dimpotrivă, perspectiva descendentă a unui obiect apare în situația în care observatorul privește construcția de sus, sub raze vizuale destul de oblice.

În figurile 5.7 și 5.8 sînt recapitulate construcțiile de bază care conduc la perspectiva Aa descendentă a unei verticale A_1a_1 .

Pe aceste figuri pot fi definite următoarele elemente:

Ω	punctul de vedere din spațiu
ω	proiecția orizontală a punctului de vedere (poziția observatorului)
XX	baza tabloului înclinat de perspectivă (descendentă)
hh	linia orizontului
P	punctul principal (proiecția ortogonală a punctului de vedere pe tablou)
(P)	punctul pseudoprincipal
F_z	punctul de fugă al verticalelor
M_z	punctul de măsură al verticalelor
U	unghiul dintre tabloul înclinat de perspectivă și planul orizontal
Ω_0	rabaterea pe tablou a punctului de vedere.
a_i	urma verticalei A_1a_1 pe tablou
a_x	abscisa punctului în raport cu P_x sau cu altă origine de pe xx .
$a_x a_1$	profundimea punctului
$A_1 a_1$	cota punctului.

În aceste condițiuni în tablou (fig. 5.8) dreapta $A_x(P)$ este o pseudo-principală, iar dreptele PA' sau Pa' sînt principale. Dreapta $F_z a_i$ este o verticală. Perspectivele A a punctului A_1 din spațiu și a a proiecției sale orizontale pot fi determinate din intersecțiile respective combinînd cîte două drepte citate mai sus.

Pentru determinarea perspectivei B a fost utilizat punctul de măsură al verticalelor M_z . Cota $b_x K$ a acestui punct este proiectată din M_z în K_1 pe verticala lui b_x . Perspectiva B se găsește pe verticala dusă prin perspectiva b și pe pseudoprincipala $(P)K_1$.

Perspectiva pe tablou înclinat a unui obiect nu necesită alte variante de construcții geometrice grafice, în afara celor reamintite aici pentru o dreaptă verticală.

Problema se pune, de exemplu, în felul următor:

În raport cu triedrul tridreptunghic fix $OXYZ$ se dă obiectul prismatic cu vîrfurile P_i notate de la 1 la N , punctul de vedere $\Omega(140; 84; 130)$ și planul

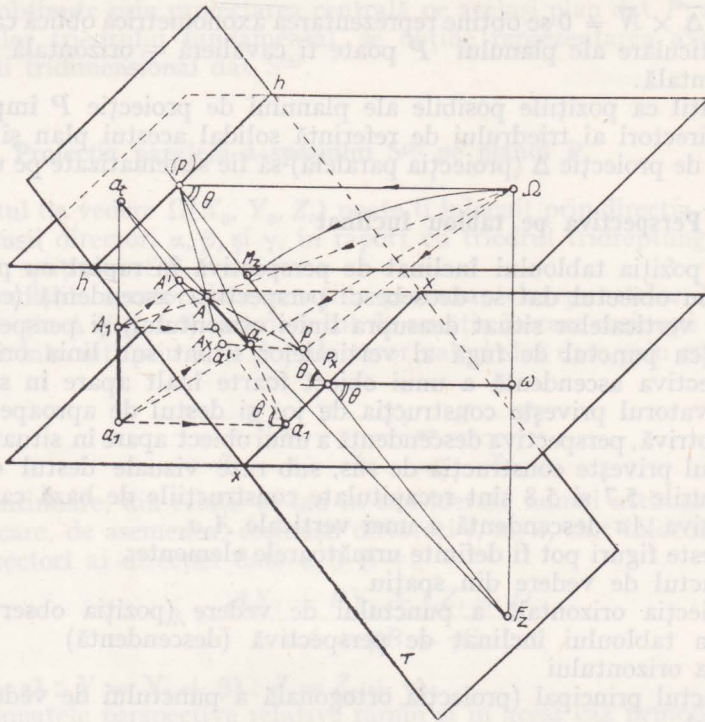


Fig. 5.7

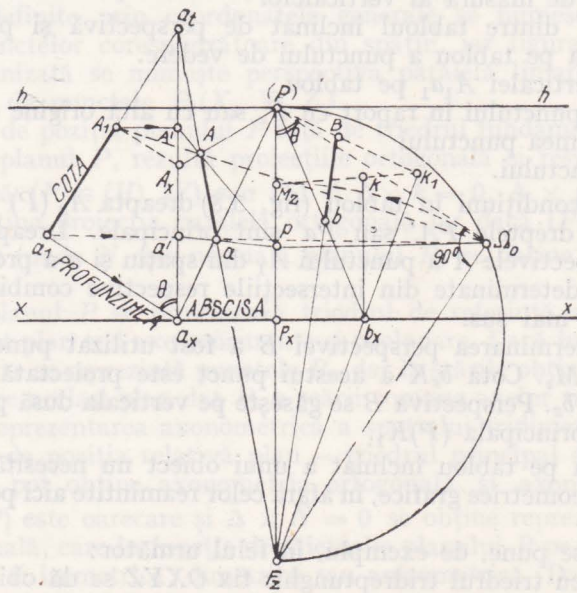


Fig. 5.8

înclinat al tabloului de perspectivă descendentă materializat prin unghiul $\theta = 60^\circ$ față de planul orizontal și prin urma sa orizontală X_1X_1' aleasă perpendiculară pe direcția ωP_{x1} .

Coordonatele punctelor P_i ($i = 1, \dots, N$) adică $XP(I), YP(I)$, pot fi cuprinse într-o tabelă de forma următoare:

Punctul P_i $i = 1, N$	Abcisa	Profundimea	Cota
-----------------------------	--------	-------------	------

Baza X_1X_1' a tabloului înclinat de perspectivă intersectează axa Y a profunzimilor (ordonatelor) în punctul $O_1 = B_2(0; 50; 0)$ și face unghiuri de 45° cu sensurile pozitive ale axelor OX și OY . Cu alte cuvinte, tăieturile tabloului înclinat de perspectivă pe muchiile triedrului tridreptunghic fix $OXYZ$ sînt $a = -50$; $b = 50$;

$$c = \frac{ab}{\sqrt{a^2 + b^2}} \operatorname{tg} \theta = -0,01635$$

În aceste condițiuni, valorile unghiurilor dintre axele triedrului fix și axele triedrului solidar cu tabloul înclinat de perspectivă, precum și valorile cosinuşilor directori respectivi, sînt următoarele:

$$\begin{aligned} \sphericalangle (X_1, X) &= 45^\circ & \alpha_1 &= \cos (X_1, X) = \frac{\sqrt{2}}{2} \\ \sphericalangle (X_1, Y) &= 45^\circ & \beta_1 &= \cos (X_1, Y) = \frac{\sqrt{2}}{2} \\ \sphericalangle (X_1, Z) &= 90^\circ & \gamma_1 &= \cos (X_1, Z) = 0 \\ \sphericalangle (Y_1, X) &= 111^\circ & \alpha_2 &= \cos (Y_1, X) = -0,358 \\ \sphericalangle (Y_1, Y) &= 60^\circ & \beta_2 &= \cos (Y_1, Y) = 0,358 \\ \sphericalangle (Y_1, Z) &= \frac{\pi}{2} - \theta = 30^\circ & \gamma_2 &= \cos (Y_1, Z) = \frac{\sqrt{3}}{2} \end{aligned}$$

Valorile unghiurilor, care conduc la cosinuşii directori α_2 și β_2 pot fi determinate grafic, pe epură, sau prin calcul, așa cum s-a arătat.

Aceste valori ale cosinuşilor directori, odată calculate, sînt valabile pentru orice program de calcul, în orice altă aplicație, în care tabloul înclinat de perspectivă își păstrează caracteristicile sale de poziție date de axa X_1X_1' și unghiul θ al liniei sale de cea mai mare pantă în raport cu planul orizontal.

5.2.4. Perspectiva pe tablou vertical (frontal)

Relațiile care generalizează metoda coordonatelor perspective în perspectiva pe un tablou înclinat oarecare, pot conduce la două particularizări importante și anume la utilizarea aceluiași relații pentru determinarea perspectivei pe un tablou vertical oarecare sau pe un tablou vertical frontal.

În cazul tabloului de perspectivă vertical oarecare planul $X_1B_2Z_1$ se confundă cu planul orizontal XOY , originea $B_2(X_2, Y_2, Z_2)$ devine (fig. 5.2) $B_2(0, Y_2, 0)$, iar cosinuşii directori:

$$\alpha_1 \neq 0 \quad \alpha_2 = 0 \quad \alpha_3 \neq 0$$

$$\beta_1 \neq 0 \quad \beta_2 = 0 \quad \beta_3 \neq 0$$

$$\gamma_1 = 1 \quad \gamma_2 = 1 \quad \gamma_3 = 0$$

Cu aceste precizări relațiile

$$\begin{cases} X_1 = X\alpha_1 + (Y - Y_2)\beta_1 \\ Y_1 = Z \\ Z_1 = \alpha_3 X + \beta_3(Y - Y_2) \end{cases}$$

generalizează metoda coordonatelor perspective în perspectiva pe tablou vertical oarecare.

În cazul cînd tabloul de perspectivă este vertical frontal, deci paralel cu planul XOZ al triedrului tridreptunghic fix $OXYZ$, de asemenea planul $X_1B_2Z_1$ se confundă cu planul orizontal XOY , originea $B_2(X_2, Y_2, Z_2)$ devine $B_2(O, Y_2, 0)$, iar cosinuşii directori

$$\alpha_1 = 1 \quad \alpha_2 = \beta_2 = 0 \quad \alpha_3 = \gamma_3 = 0$$

$$\beta_1 = \gamma_1 = 0 \quad \gamma_2 = 1 \quad \beta_3 = -1$$

cu aceste precizări relațiile devin

$$\begin{cases} X_1 = X \\ Y_1 = Z \\ Z_1 = -(Y - Y_2) = 0 \end{cases}$$

Aceste relații generalizează metoda coordonatelor perspective în perspectiva pe tablou vertical frontal.

La relațiile echivalente se poate ajunge și pe altă cale. Astfel, fie sistemul perspectiv de proiecție și punctul A_1a_1 din spațiu (fig. 5.9). Se observă că planul tabloului este divizat în patru cadrane I—IV de linia orizontului hh și de verticala principală P_xP , care este dreapta de intersecție dintre tablou și planul principal de vedere (PPV). Pentru perspectivele A sau a se pot introduce coordonatele perspective X_A și Y_A .

Se introduc următoarele notații (fig. 5.10)

d — distanța principală,

h — înălțimea orizontului,

ρ — distanța punctului A_1a_1 dat la planul neutru,

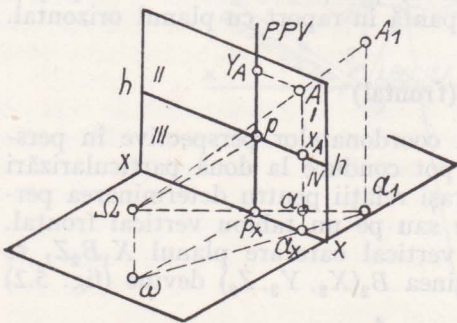


Fig. 5.9

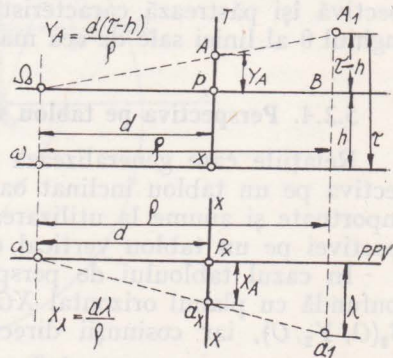


Fig. 5.10

λ — distanța punctului A_1a_1 dat față de planul principal de vedere,
 τ — cota punctului A_1a_1 dat față de planul orizontal.

În aceste condițiuni, din asemănarea triunghiurilor $\Delta\Omega AP \sim \Delta\Omega A_1B$
 și $\Delta\omega P_x a_x \sim \Delta\omega b a_1$ rezultă

$$\frac{\tau - h}{Y_A} = \frac{\rho}{d}; \quad \frac{\lambda}{X_A} = \frac{\rho}{d} \text{ de unde } \boxed{\begin{array}{l} Y_A = \frac{d(\tau - h)}{\rho} \\ X_A = \frac{d\lambda}{\rho} \end{array}}$$

și respectiv

Evident, pentru punctele planului orizontal (ca a_1) cota Z este zero.

Calculînd cu semnele respective coordonatele perspective X_A și Y_A ale unui punct din spațiu se obține în tablou poziția perspectivei acestui punct.

Aceste elemente vor fi reconsiderate în paragraful 5.8.1. referitor la vizualizarea obiectelor spațiale unde tabloul de perspectivă T va fi asimilat cu ecranul display.

Întrucît trasarea perspectivei pe un tablou vertical presupune numai o poziție particulară a tabloului de perspectivă în raport cu triedrul tridreptrunghic fix, restul datelor rămînînd neschimbate, caracterul de generalitate al relațiilor perspective se păstrează.

Avem însă

$$GAMA1 = 0; ALFA2 = 0; BETA2 = 0;$$

$$GAMA2 = 1; C = 0$$

5.2.5. Perspectiva pe un tablou orizontal

Relațiile care generalizează metoda coordonatelor perspective în perspectiva pe un tablou înclinat oarecare pot conduce, de asemenea, la o particularizare importantă pentru determinarea perspectivei pe un tablou orizontal plan.

În acest caz, planul $X_1B_2Y_1$ devine paralel cu planul XOY (fig. 5.11), iar cosinuzii directori au valorile următoare:

$$\alpha_1 = 1 \quad \alpha_2 = 0$$

$$\beta_1 = 0 \quad \beta_2 = 1$$

$$\gamma_1 = 0 \quad \gamma_2 = 0$$

De asemenea, coordonatele originii B_2 sînt $X_2 = Y_2 = 0; Z_2 = k$. Pentru tabloul de perspectivă orizontal

$$A = B = 0; \quad C = \frac{1}{k}; \quad D = -1$$

Deoarece

$$X_1 = \alpha_1(X - X_2) + \beta_1(Y - Y_2) + \gamma_1(Z - Z_2)$$

$$Y_1 = \alpha_2(X - X_2) + \beta_2(Y - Y_2) + \gamma_2(Z - Z_2)$$

și

$$X = X_0 - \lambda l$$

Se vor modifica însă parametrii care fixează poziția, tabloului de perspectivă în felul următor:

$$A = 0; \quad B = 0; \quad C = \frac{1}{k}; \quad D = -1$$

$$\begin{aligned} ALFA1 &= 1; & BETA1 &= 0; & GAMA1 &= 0 \\ ALFA2 &= 0; & BETA2 &= 1; \\ GAMA2 &= 0. \end{aligned}$$

5.3. Formularea problemei trasării perspectivei

Trasarea perspectivei unuia sau a mai multor obiecte poate fi considerată conștind dintr-un set de linii drepte sau curbe unind un set de puncte date sau calculate. Datele pot fi considerate conștind din:

— O listă L a coordonatelor tridimensionale în raport cu un sistem tri-dreptunghiular fix și care reprezintă punctele prin care trec liniile sau curbele ce delimitează conturul obiectului sau obiectelor a căror perspectivă urmează a fi desenată.

— O listă a liniilor ce unesc coordonatele din lista anterioară, linia fiind specificată prin numerele de referință corespunzătoare punctelor sale extreme.

— Coordonatele, punctului sau punctelor de vedere (poziția ochiului observatorului) în raport cu sistemul de referință tridreptunghiular fix.

— Coeficienții A, B, C, D , ai ecuației planului pe care urmează a se trasa perspectiva, împreună cu cosinușii directori ai axelor triedrului solidal cu planul perspectivei în raport cu axele triedrului fix.

Cunoscându-se aceste date se cere trasarea perspectivei unui sau mai multor obiecte (poliedre, curbe, suprafețe, etc.) ale căror coordonate sînt date în ista L pe un plan dat prin coeficienții A, B, C, D , de mai sus.

5.3.1. Notaii cu caracter general pentru întocmirea programului de calculator

$XP(I), YP(I), ZP(I)$	Coordonatele punctelor obiectului vizat.
$X0, Y0, Z0$	Coordonatele observatorului
$X2, Y2, Z2$	Coordonatele originii triedrului solidal cu planul tabloului
A, B, C, D	Coeficienții ecuației planului tablou de perspectivă
$ALFA1,$	Cosinușii directori ai axelor triedrului solidal cu planul perspectivei
$ALFA2,$	
$BETA1,$	
$BETA2,$	
$GAMA1,$	
$GAMA2,$	
N	Numărul total de puncte vizate
$M = \frac{N}{2}$	Jumătatea numărului de puncte vizate
DVP	Distanța de la punctul vizat la observator
CL, CM, CN	Cosinușii directori ai lui DVP
$LAMBDA$	Raport variabil cu punctul vizat
X, Y, Z	Coordonatele perspectivei punctului în raport cu triedrul fix

$X1(I), Y1(I), Z1(I)$
TRAS

Idem în raport cu triedrul solidar planului perspectivei
Este variabilă întregă ce poate lua valorile „1” sau „0” după cum se cheamă sau nu subrutina TRASXY.
(Acest subprogram TRASXY va fi prezentat în paragraful 5.4.2).

AL, BE, GA

Cosinușii directori ai direcției de proiectare Δ în raport cu axele triedrului principal pentru proiecția paralelă

PROP
sau
IZOM

Variabilă întregă ce poate lua valorile „1” sau „0” cum se calculează proiecția paralelă sau proiecția centrală (în ambele cazuri perspectiva liniară sau convențională).

În cazul trasării perspectivelor convenționale punctelor $XP(I), YP(I), ZP(I)$, ce definesc corpul geometric considerat, — li se adaugă originea triedrului fundamental și punctele ce marchează tăieturile planului P pe acest triedru.

5.3.2. Program principal PROP CENT sau IZOM pentru perspectiva paralelă oarecare, centrală sau izometrică

```

DETERMINAREA SI TRASAREA PERSPECTIVEI CENTRALA, A PERSPECTIVEI
PARALELE OARECARE SAU A PERSPECTIVEI AXONOMETRICE IZOMETRICE
* DEFINE FILE *1=1, *2=3
DIMENSION XP(200), YP(200), ZP(200), X1(200), Y1(200)
REAL *4 NP, LAMBDA
INTEGER N, PROP
READ(1, 101) N, PROP
101 FORMAT(I4, I2)
C PENTRU PERSPECTIVA CENTRALA PROP=0 IAR AL=BE=GA=0.0
C PENTRU PERSPECTIVA PARALELA OARECARE PROP=1 IAR
C      AL = 0.85      BE = 0.500      GA = 0.450
C PENTRU AXONOMETRIA IZOMETRICA PROP=1 IAR AL=BE=GA=0.578
READ(1, 102) XO, YO, ZO, X2, Y2, Z2, A, B, C, D, AL1, BE1, GA1, AL2, BE2, GA2,
*AL, BE, GA
102 FORMAT(8F10.4)
WRITE(3, 201) N, XO, YO, ZO, X2, Y2, Z2, A, B, C, D, AL1, BE1, GA1, AL2, BE2, GA2,
*AL, BE, GA
201 FORMAT('0', 'DETERMINAREA PROIECTIEI CUNOSCIND'//',', 'N=', I4, 'X0=',
1', F9.3, '1X', 'Y0=', F9.3, '1X', 'Z0=', F9.3, '1X', 'X2=', F9.3, '1X', 'Y2=',
2', F9.3, '1X', 'A=', F9.3, '1X', 'B=', F9.3, '1X', 'C=', F9.3, '1X', 'D=',
3X, 'AL1=', F9.3, '1X', 'BE1=', F9.3, '1X', 'GA1=', F9.3, '1X', 'AL2=',
42', F9.3, '1X', 'GA2=', F9.3, '1X', 'AL=', F9.3, '1X', 'BE=', F9.3, '1X', 'GA=',
57//', '1X', '17X', 'X1(I)', '16X', 'Y1(I)')//')
M=Y/Z
DO 10 I=1, N
10 READ 11, XP(I), YP(I), ZP(I)
11 FORMAT(3F6.1)
DO 20 I=1, N
IF(PROP.NE.1) GO TO 15
C      PERSPECTIVA PARALELA OARECARE SAU IZOMETRIA
LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*AL+B*BE+C*GA)
X=XP(I)-AL*LAMBDA
Y=YP(I)-BE*LAMBDA
Z=ZP(I)-GA*LAMBDA
GO TO 13
C
C      PERSPECTIVA CENTRALA
15 DVP=SQRT((XO-XP(I))**2+(YO-YP(I))**2+(ZO-ZP(I))**2)
CL=(XO-XP(I))/DVP
CM=(YO-YP(I))/DVP
CN=(ZO-ZP(I))/DVP
LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*CL+B*CM+C*CN)
X=XP(I)-CL*LAMBDA
Y=YP(I)-CM*LAMBDA
Z=ZP(I)-CN*LAMBDA
C
C      COORDONATELE PERSPECTIVE ALE PUNCTULUI
13 X1(I)=AL1*(X-X2)+BE1*(Y-Y2)+GA1*(Z-Z2)
Y1(I)=AL2*(X-X2)+BE2*(Y-Y2)+GA2*(Z-Z2)
20 WRITE(3, 202) I, X1(I), Y1(I)
202 FORMAT('0', I4, 2(14X, F7.1))
STOP
&ND

```

Din acest program vor fi derivate programe pentru desenarea automată la plotter sau la imprimanta grafică a curbelor și suprafețelor exprimate prin ecuații explicit sau parametric.

5.4. Reprezentarea perspectivă pe imprimantă a curbelor și suprafețelor pentru o primă aproximare

5.4.1. Subprogramul CALCF

Acest subprogram calculează coordonatele punctelor situate pe o curbă sau pe o familie de curbe de pe o suprafață considerînd secțiunile în suprafața definită de o funcție de 2 variabile, cu 1000 de puncte în domeniul de definiție $A1, A2$, pentru x și $B1, B2$ pentru y . Pasul CW de creștere pentru x poate fi diferit de pasul DW de creștere pentru y . Acestea sînt de fapt datele de intrare la care se adaugă funcția $ZP(M)$ unde am notat $XP(M) = G$ și $YP(M) = H$.

Forma apelului este:

CALL CALCF(A1, A2, B1, B2, CW, DW, XP, YP, ZP)

unde datele de ieșire (valorile funcției) sînt XP, YP, ZP . Listarea subprogramului **CALCF** este următoarea:

```

SUBROUTINE CALCF(A1,A2,B1,B2,CW,DW,XP,YP,ZP)
DIMENSION XP(1000),YP(1000),ZP(1000)
ELEMENTELE INTRODUSE SINT:
C  - A1,A2 = DOMENIUL DE DEFINITIE PT X
C  - B1,B2 = DOMENIUL DE DEFINITIE PT Y
C  - CW   = PASUL DE CREȘTERE PT X
C  - DW   = PASUL DE CREȘTERE PT Y
PC=(A2-A1)/CW
PD=(B2-B1)/DW
MC=IFIX(PC)
ND=IFIX(PD)
G=A1
H=B1
M=0
DO 131 I=1,MC
DO 130 J=1,ND
H=M+1
E1=6-8*G+3*G*G
E2=1-8*G+12*G*G-8*G*G*G+G*G*G*G
E3=-1+16*G-42*G*G+44*G*G*G-14*G*G*G*G
ZP(M)=G*G*E1+2*H*E2+H*H*E3
XP(M)=G
YP(M)=H
H=H+DW
130 CONTINUE
H=A1
G=G+CW
131 CONTINUE
WRITE(3,140)
140 FORMAT(1H,10X,'VALORILE FUNCTIEI')
WRITE(3,150)
150 FORMAT(1H,5X,' X      Y      Z  ')
DO 171 J=1,M
WRITE(3,170) XP(J),YP(J),ZP(J)
170 FORMAT(5X,3F7.2)
171 CONTINUE
WRITE(3,180)
180 FORMAT(//1H0,'GATA')
RETURN
END

```

Pentru utilizarea subprogramului **CALCF** se introduce acesta în programul principal **PROP**, **CENT** sau **IZOM** care determină și trasează perspectiva paralelă oarecare, perspectiva centrală sau perspectiva axonometrică a șirului de puncte ce definește o curbă de secțiune în suprafață.

Pentru subprogramul de trasare **TRASXY** se pot folosi numai valorile pozitive ale domeniului. Valorile negative pot fi înlăturate printr-o translație.

5.4.2. Subprogramul **TRASXY**

Acest subprogram determină calculatorul să realizeze trasarea în trepte, a oricăreia dintre perspective: pe tablou oarecare, vertical, frontal, sau cilindric.

Valorile coordonatelor perspective, obținute pe calculator în baza instrucțiunilor cuprinse în programul principal urmează a fi reprezentate grafic, de unde rezultă și formularea problemei:

Având masivele $X1(I)$ și $Y1(I)$ de dimensiuni ce corespund numărului de puncte vizate N , să se efectueze reprezentarea lor grafică (pe imprimantă).

Notațiile admise sînt următoarele:

<i>IMPR</i>	Masiv de lungime 101 semicuvinte (100 puncte plus originea) cu ajutorul căruia se imprimă o linie din reprezentarea grafică.
<i>BLANK, SEMN, ILIT, LIN, PLUS.</i>	Sînt nume de variabile întregi cu lungimea 2 (ocupă două locații în memorie) și se inițializează prin caracterele sau semnele: &, *, I, -, +, Astfel:
<i>BLANK „&“</i>	Servește la inițializarea masivului <i>IMPR</i> .
<i>SEMN „*“</i>	Este un semn special ce servește la reprezentarea punctelor.
<i>ILIT „I“</i>	Este caracterul cu care se reprezintă axa ordonatelor.
<i>LIN „-“</i>	Este semnul ce reprezintă axa abciselor.
<i>PLUS „+“</i>	Este semnul ce reprezintă originea axelor.
<i>XMIN, YMIN</i>	Reprezintă punctele de intrare într-un sistem de axe rectangulare ($X1, Y1$) trasat.
<i>N</i>	Numărul punctelor reprezentate în perspectivă. —
<i>X, Y</i>	Sînt argumente fictive care, vor căpăta prin enunțul CALL valorile masivelor $X1$ și $Y1$ calculate în programul chemător.
<i>XMAX, XMIN, YMAX, YMIN</i>	Reprezintă valorile maxime respectiv minime pe care le pot lua variabilele fictive X și Y .
<i>AX, AY</i>	Reprezintă amplitudinea variațiilor lui X și respectiv Y .
<i>SC</i>	Reprezintă valoarea maximă a uneia dintre amplitudinile de mai sus.
<i>IX, IY</i>	Reprezintă masivele în care se trec coordonatele punctelor de reprezentat în sistemul de axe în care se va face reprezentarea. Variabilele indicate din acest masiv au lungimea 2 și sînt întregi.
<i>SCX, SCY</i>	Sînt scările pentru variabilele X și respectiv Y și se ține seama de faptul că pentru X sînt prevăzute 100 de intervale, iar pentru Y numai 80.

MAXX, MAXY Reprezintă amplitudinele maxime ale variabilelor IX și respectiv IY .

SUBROUTINE ORD Este un subprogram cu ajutorul căruia se ordonează punctele după valorile crescătoare ale variabilei Y și în care se trec cu coordonatele celor N , puncte sub forma de 2 masive.

DATA IMPR /101' /Este un enunț utilizat pentru inițializarea masivului **IMPR** cu caracterul vid.

INTER Este o variabilă care indică modul în care se parcurge programul prin enunțul **GOTO** calculat. Are rolul unui comutator ce poate fi trecut într-una din cele două poziții controlînd parcurgerea programului.

VMAX și VMIN Sînt funcțiuni ce calculează valoarea maximă și respectiv minimă dintr-un șir de valori date într-un masiv. Ordinea de introducere a datelor este

$X1, Y1, N$

iar informația extrasă la ieșire este $XMIN, XMAX, YMAX, YMIN$, împreună cu trasarea perspectivă prin puncte.

Listarea subprogramului **TRASXY** este următoarea:

```

SUBROUTINE TRASXY (X,Y,N)
INTEGER*2 BLANK,SEMN,ILIT,LIN,IX(1000),IY(1000),IMPR(101),PLUS
DIMENSION X(N),Y(N)
DATA BLANK,SEMN,ILIT,LIN,PLUS/' ','*','I','-',',','/'
DATA IMPR/101*' '/
XMAX=VMAX(X,N)
XMIN=VMIN(X,N)
YMAX=VMAX(Y,N)
YMIN=VMIN(Y,N)
AX=XMAX-XMIN
AY=YMAX-YMIN
SC=AMAX1(AX,AY)
SCX=SC/100.
SCY=SC/80.
MAXY=AY/SCY+1.5
MAXX=AX/SCX+1.5
DO 10 I=1,N
10 IX(I)=(X(I)-XMIN)/SCX+1.5
IY(I)=(Y(I)-YMIN)/SCY+1.5
CALL ORD(IY,IX,N)
NV=N
WRITE (3,101) YMAX
MY=MAXY
17 DO 11 J=2,MAXX
11 IMPR(J)=BLANK
IMPR(1)=ILIT
IF(IY(NV).NE.MY) GO TO 13
12 IND=IX(NV)
IMPR(IND)=SEMN
NV=NV-1
IF(IY(NV).EQ.IY(NV+1)) GO TO 12
13 WRITE (3,102) (IMPR(J),J=1,MAXX)
MY=MY-1
IF(MY.NE.1) GO TO 17
NV=NV+1
18 DO 14 J=2,MAXX
14 IMPR(J)=LIN
IMPR(1)=PLUS
15 NV=NV-1
IND=IX(NV)
IMPR(IND)=SEMN
IF(NV.GT.1) GO TO 15
16 WRITE (3,103) YMIN, (IMPR(J),J=1,MAXX)
WRITE (3,104) XMIN,XMAX
101 FORMAT('1','YMAX=',F12.5)
102 FORMAT(' ','18X,101A1)
103 FORMAT(' ','YMIN=',F12.5,1X,101A1)
104 FORMAT('0',12X,'XMIN=',F12.5,30X,'XMAX=',F12.5)
RETURN
END

```

```

SUBROUTINE ORD (X,Y,N)
INTEGER*2 X(N),Y(N),SALV
WRITE(3,5) (X(I),I=1,N)
5 FORMAT(' ',8F10.3)
INTER=1
N1=N-1
DO 3 K=1,N1
GO TO (1,4),INTER
1 INTER=2
MAX=N-K
DO 3 J=1,MAX
IF(X(J)-X(J+1)) 3,3,2
2 SALV=X(J)
X(J)=X(J+1)
X(J+1)=SALV
SALV=Y(J)
Y(J)=Y(J+1)
Y(J+1)=SALV
INTER=1
3 CONTINUE
WRITE(3,5) (X(I),I=1,N)
4 RETURN
END

```

```

FUNCTION VMIN (A,N)
DIMENSION A(N)
VMIN=A(1)
DO 10 L=2,N
IF(A(L).LT.VMIN) VMIN=A(L)
10 CONTINUE
RETURN
END

```

```

FUNCTION VMAX (A,N)
DIMENSION A(N)
VMAX=A(1)
DO 10 L=2,N
IF(A(L).GT.VMAX) VMAX=A(L)
10 CONTINUE
RETURN
END

```

5.4.3 Subprogramul CALCFU

Acest subprogram este o particularizare a programului CALCF și are ca scop reprezentarea în ortogonal a unei funcții $y = f(x)$. El este foarte util în aplicații și de aceea a fost luat în considerare în cadrul acestui capitol.

Listarea programului CALCFU este următoarea:

```

C
C PROGRAM PENTRU REPREZENTAREA
C FUNCTIEI Y=F(X)
DIMENSION XP(1000),YP(1000)
READ(105,50) A1,A2,CW
50 PC=(A2-A1)/CW
FORMAT(9X,3F7.2)
WRITE(108,55) A1,A2,CW
55 FORMAT(1X,'123456789',3(/,F7.2))
MC=IFIX(PC)
MC=MC+1
G=A1
M=0
DO 131 I=1,MC
M=M+1
YP(M)=GM*2+2*G+1
XP(M)=G
G=G+CW
131 CONTINUE
WRITE(108,140)
140 FORMAT(1H,10X,'VALORILE FUNCTIEI')
WRITE(108,150)
150 FORMAT(1H,5X, ' X Y ')
DO 171 J=1,M
WRITE(108,170) XP(J),YP(J)
170 FORMAT(5X,2F7.2)
171 CONTINUE
STOP
END

```

5.4.4 Aplicații. Suprafețe desenate pe imprimanta obișnuită. Figurile 5,12a,b,c,d

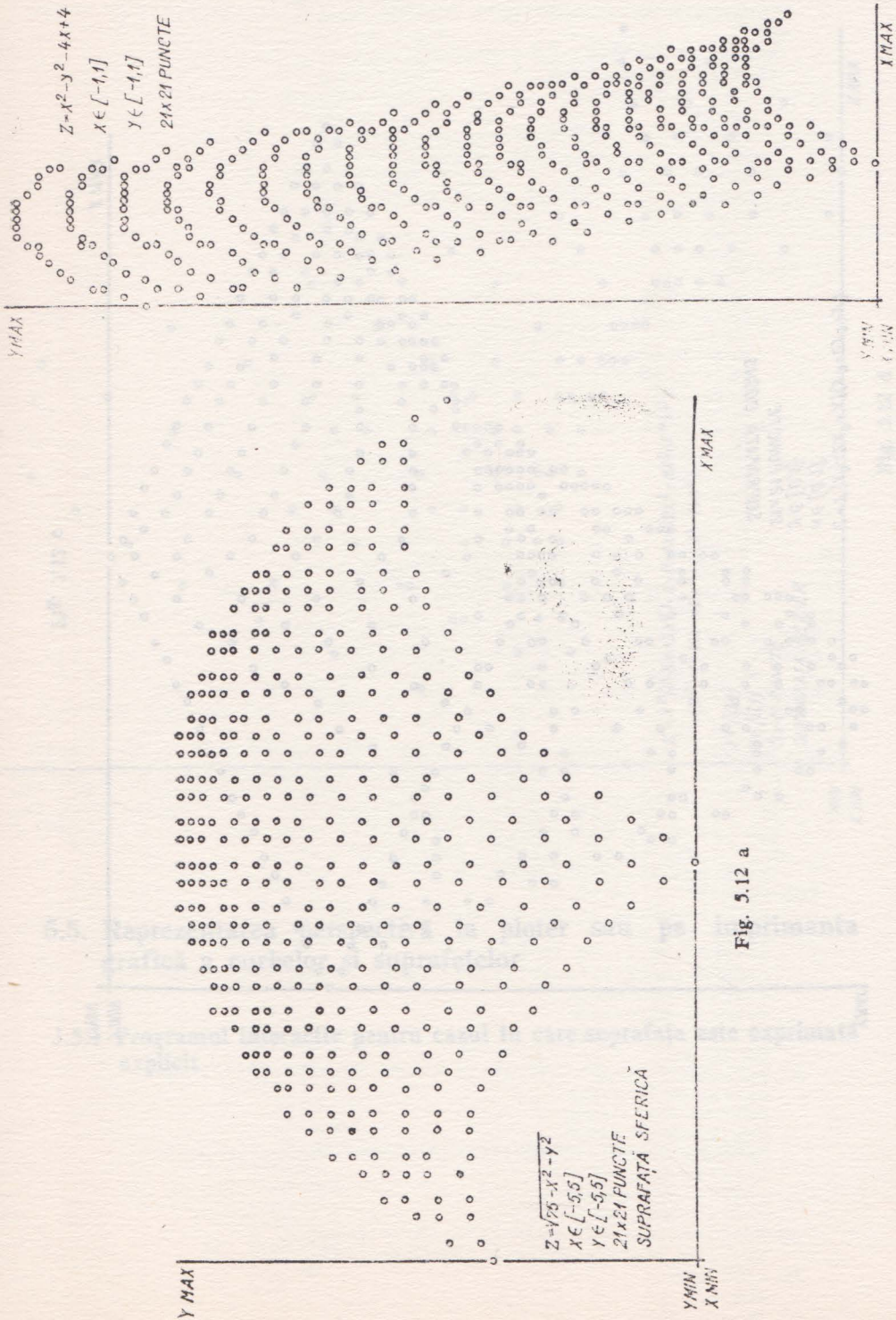


Fig. 5.12 a

Fig. 5.12 b

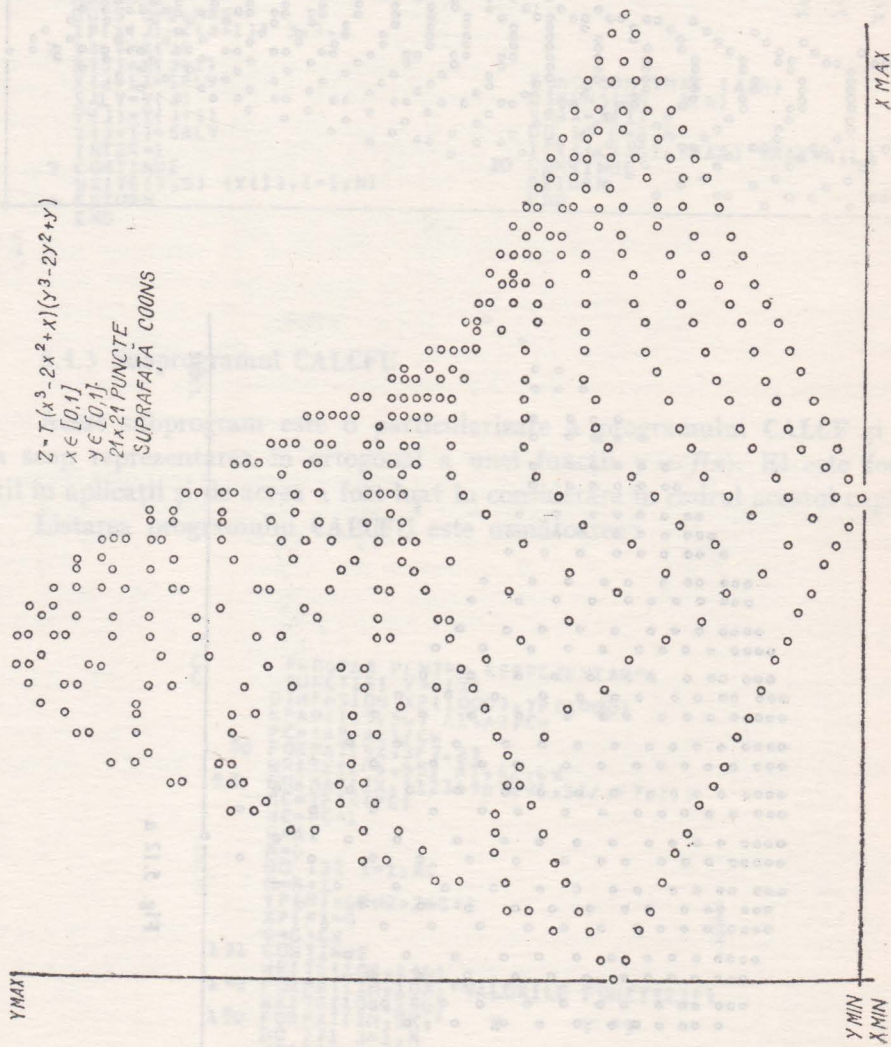


Fig. 5.12 c

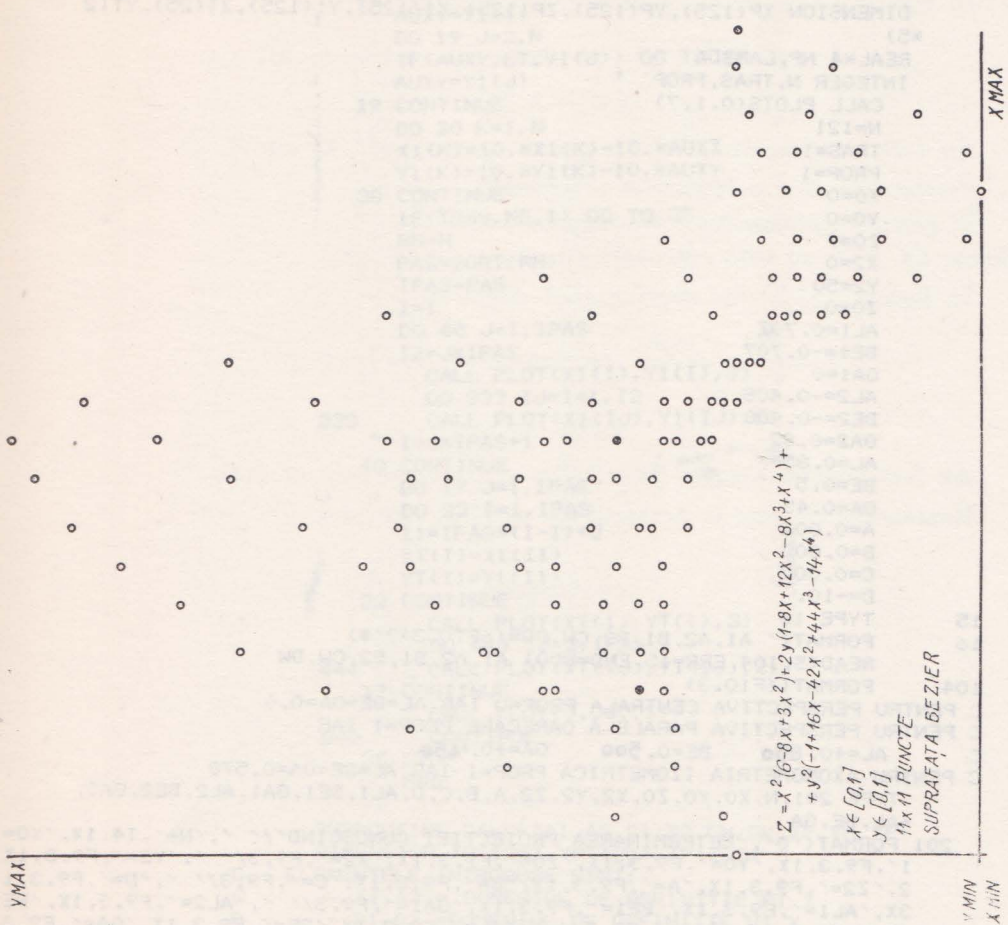


Fig. 5.12 d

5.5. Reprezentarea perspectivă la ploter sau pe imprimanta grafică a curbelor și suprafețelor

5.5.1 Programul interactiv pentru cazul în care suprafața este exprimată explicit

```

DIMENSION XP(125),YP(125),ZP(125),X1(125),Y1(125),XT(25),YT(2
*5)
REAL*4 NP,LAMBDA
INTEGER N,TRAS,PROP
CALL PLOTS(0,1,7)
N=121
TRAS=1
PROP=1
X0=0
Y0=0
Z0=0
X2=0
Y2=50
Z0=0
AL1=0.707
BE1=-0.707
GA1=0
AL2=-0.408
BE2=-0.408
GA2=0.82
AL=0.85
BE=0.5
GA=0.45
A=0.002
B=0.002
C=0.002
D=-10.
15 TYPE 16
16 FORMAT(' A1,A2,B1,B2,CW,DW=(6F10.3)?'*)
READ(5,104,ERR=15,END=800) A1,A2,B1,B2,CW,DW
104 FORMAT(6F10.3)
C PENTRU PERSPECTIVA CENTRALA PROP=0 IAR AL=BE=GA=0.0
C PENTRU PERSPECTIVA PARALELA OARECARE PROP=1 IAR
C AL=+0.850 BE=0.500 GA=+0.450
C PENTRU AXONOMETRIA IZOMETRICA PROP=1 IAR AL=BE=GA=0.578
TYPE 201,N,X0,Y0,Z0,X2,Y2,Z2,A,B,C,D,AL1,BE1,GA1,AL2,BE2,GA2,
*AL,BE,GA
201 FORMAT('0',DETERMINAREA PROIECTIEI CUNOSCIND'//',N=',14,1X,'X0= X0= =
1',F9.3,1X,'Y0=',F9.3,1X,'Z0=',F9.3,1X,'X2=',F9.3,1X,'Y2=',F9.3,1X
2',F9.3,1X,'A=',F9.3,1X,'B=',F9.3,1X,'C=',F9.3,1X,'D=',F9.3,1
3X,'AL1=',F9.3,1X,'BE1=',F9.3,1X,'GA1=',F9.3,1X,'AL2=',F9.3,1X,'BE
42=',F9.3,1X,'GA2=',F9.3,1X,'AL=',F9.3,1X,'BE=',F9.3,1X,'GA=',F9.3
5//',2X,'I',17X,'X1(I)',16X,'Y1(I)')
TYPE 203,A1,A2,B1,B2,CW,DW
203 FORMAT(' ',6F10.3/)
CALL CALCF(A1,A2,B1,B2,CW,DW,XP,YP,ZP)
DO 20 I=1,N
IF (PROP.NE.1) GO TO 25
LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*AL+B*BE+C*GA)
X=XP(I)-AL*LAMBDA
Y=YP(I)-BE*LAMBDA
Z=ZP(I)-GA*LAMBDA
GO TO 23
25 DVP=SQRT((X0-XP(I))**2+(Y0-YP(I))**2+(Z0-ZP(I))**2)
CL=(X0-XP(I))/DVP
CM=(Y0-YP(I))/DVP
CN=(Z0-ZP(I))/DVP
LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*CL+B*CM+C*CN)
X=XP(I)-CL*LAMBDA+8
Y=YP(I)-CM*LAMBDA+8
Z=ZP(I)-CN*LAMBDA+10
23 X1(I)=AL1*(X-X2)+BE1*(Y-Y2)+GA1*(Z-Z2)
Y1(I)=AL2*(X-X2)+BE2*(Y-Y2)+GA2*(Z-Z2)
20 CONTINUE
AUX=X1(1)
DO 18 J=2,N

```

```

      IF(AUXX.LT.X1(J)) GO TO 18
      AUXX=X1(J)
18  CONTINUE
      AUXY=Y1(1)
      DO 19 J=2,N
      IF(AUXY.LT.Y1(J)) GO TO 19
      AUXY=Y1(J)
19  CONTINUE
      DO 30 K=1,N
      X1(K)=10.*X1(K)-10.*AUXX
      Y1(K)=10.*Y1(K)-10.*AUXY
30  CONTINUE
      IF(TRAS.NE.1) GO TO 35
      RN=N
      PAS=300/RN
      IPAS=PAS
      I=1
      DO 40 J=1,IPAS
      I2=J*IPAS
      CALL PLOT(X1(I),Y1(I),3)
      DO 333 IJ=I+1,I2
333  CALL PLOT(X1(IJ),Y1(IJ),2)
      I=J*IPAS+1
40  CONTINUE
      DO 17 J=1,IPAS
      DO 22 I=1,IPAS
      I1=IPAS*(I-1)+J
      XT(I)=X1(I1)
      YT(I)=Y1(I1)
22  CONTINUE
      CALL PLOT(XT(1),YT(1),3)
      DO 444 IJ=2,IPAS
444  CALL PLOT(XT(IJ),YT(IJ),2)
17  CONTINUE
      CALL PLOT(0.,0.,999)
800  CONTINUE
35  STOP
      END

```

SUBROUTINE CALCF(A1,A2,B1,B2,CW,DW,XP,YP,ZP)

DIMENSION XP(125),YP(125),ZP(125)

C ELEMENTELE INTRODUSE SINT:

C - A1,A2 = DOMENIUL DE DEFINITIE PT X

C - B1,B2 =DOMENIUL DE DEFINITIE PT Y

C - CW = PASUL DE CRESTERE PT X

C - DW = PASUL DE CRESTERE PT Y

PI=3.1416

PC=(A2-A1)/CW

PD=(B2-B1)/DW

MC=PC

ND=PD

G=A1

H=B1

M=0

DO 131 I=1,MC

DO 130 J=1,ND

M=M+1

XP(M)=G

YP(M)=H

ZP(M)=64.*(1.-G)*(1.-G)*(1.-H)*(1.-H)*G*H

H=H+DW

130 CONTINUE

H=A1

G=G+CW

131 CONTINUE

RETURN

END

5.5.2 Aplicații.

În figura 5.13 a este reprezentată suprafața:

$$Z = \sin \sqrt{x^2 + y^2} \quad \text{pe domeniul } x \in [-5, 5]; \\ y \in [-5, 5]$$

Au fost considerate 21×21 puncte pentru cele 42 curbe.

Perspectiva este o izometrie ortogonală.

În general aproape toate reprezentările din capitolele IX și X au fost executate cu aceste programe./

În figura 5.13 b este reprezentată suprafața:

$$Z = \frac{x^3 - 3xy^2}{25} \quad \text{pe domeniul: } x \in [-5, 5]; \\ y \in [-5, 5]$$

Au fost considerate tot 21×21 puncte pentru cele 42 curbe.

Reprezentarea este o perspectivă paralelă oarecare.

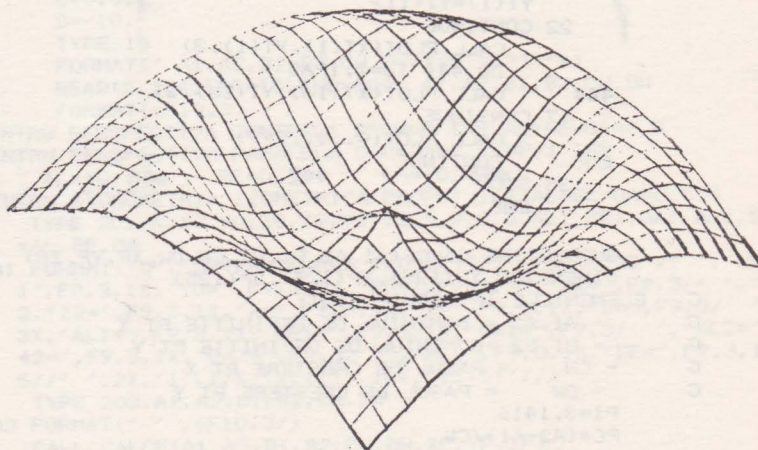
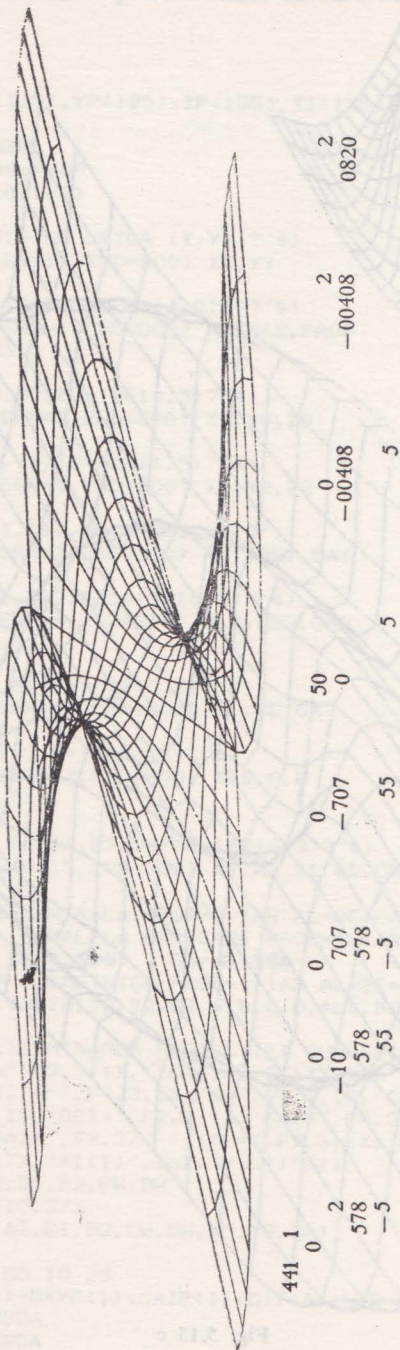


Fig. 5.13 a,



441	1	0	2	578	-5	0	-10	578	55	0	707	578	-5	0	707	55	0	-707	55	50	0	0	-00408	5	0	-00408	5	2	0820
-----	---	---	---	-----	----	---	-----	-----	----	---	-----	-----	----	---	-----	----	---	------	----	----	---	---	--------	---	---	--------	---	---	------

Fig. 5.13 b

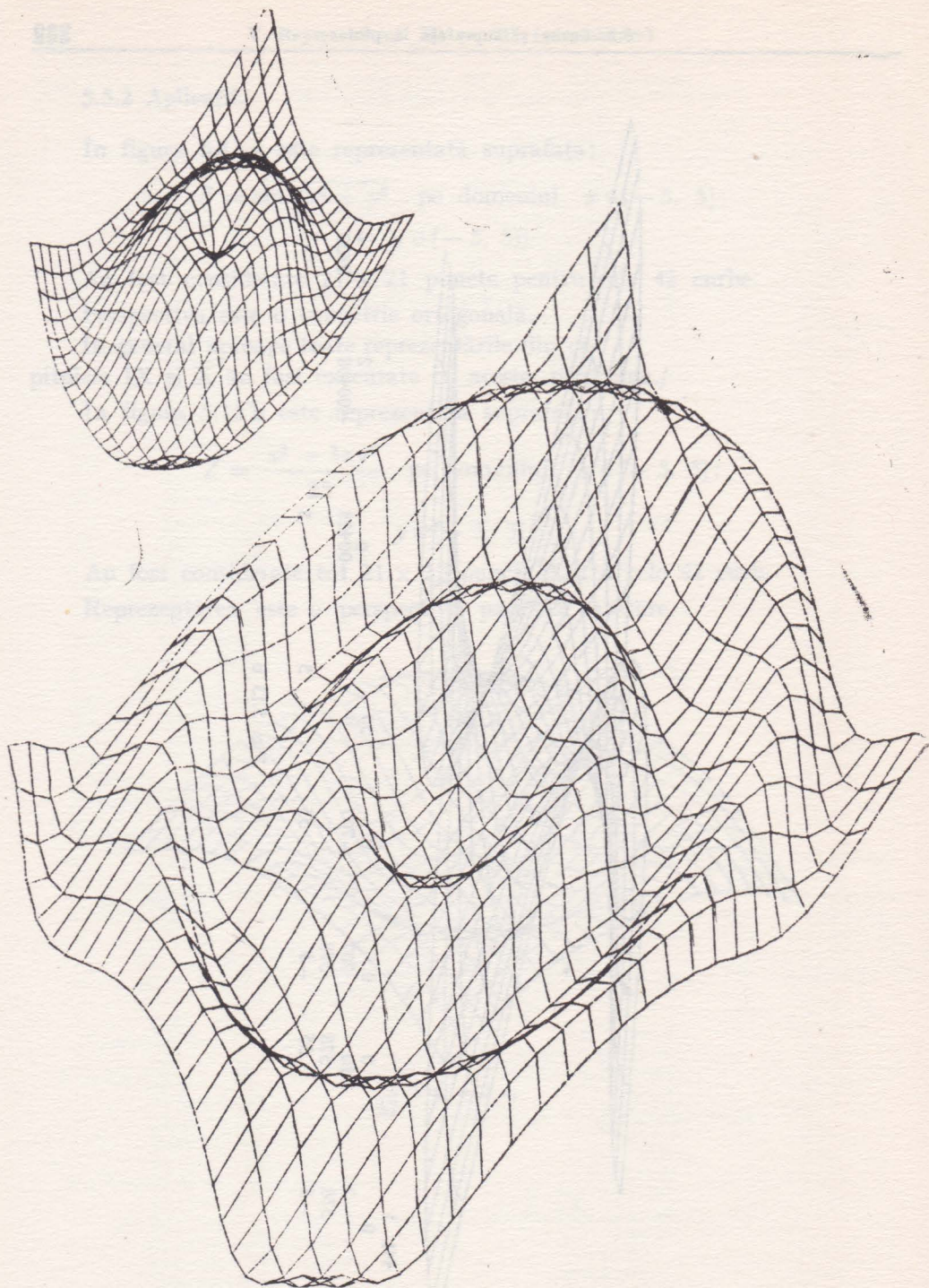


Fig. 5.13 c

Alte reprezentări anizometrice ale suprafeței considerind 42 sau 50 curbe pe domeniul $-12; 13; -12; 13$; sint date în fig. 5.13 c.

5.5.3 Programul interactiv pentru cazul în care suprafața este exprimată parametric

```

DIMENSION XP(125),YP(125),ZP(125),X1(125),Y1(125),XT(25),YT(2
*5)
REAL*4 NP,LAMBDA
INTEGER N,TRAS,PROP
CALL PLOTS(0,1,7)
111 TYPE 222
222 FORMAT(' ORIGINE GRILA (X,Y) ?'%)
READ(5,*,ERR=111,END=800) XX,YY
1 TYPE 2
2 FORMAT(' N,TRAS,PROP=(I4,2I2)?'%)
READ(5,101,ERR=1,END=800) N,TRAS,PROP
101 FORMAT(I4,2I2)
3 TYPE 4
4 FORMAT(' X0,Y0,Z0=(3F10.5)?'%)
READ(5,102,ERR=3,END=800) X0,Y0,Z0.
5 TYPE 6
6 FORMAT(' X2,Y2,Z2=(3F10.5)?'%)
READ(5,102,ERR=5,END=800) X2,Y2,Z2
7 TYPE 8
8 FORMAT(' AL1,BE1,GA1=(3F10.5)?'%)
READ(5,102,ERR=7,END=800) AL1,BE1,GA1
9 TYPE 10
10 FORMAT(' AL2,BE2,GA2=(3F10.5)?'%)
READ(5,102,ERR=9,END=800) AL2,BE2,GA2
102 FORMAT(3F10.5)
11 TYPE 12
12 FORMAT(' AL,BE,GA=(3F10.5)?'%)
READ(5,102,ERR=11,END=800) AL,BE,GA
13 TYPE 14
14 FORMAT(' A,B,C,D=(4F10.5)?'%)
READ(5,103,ERR=13,END=800) A,B,C,D
103 FORMAT(4F10.5)
15 TYPE 16
16 FORMAT(' A1,A2,B1,B2,CW,DW=(6F10.3)?'%)
READ(5,104,ERR=15,END=800) A1,A2,B1,B2,CW,DW
104 FORMAT(6F10.3)
C PENTRU PERSPECTIVA CENTRALA PROP=0 IAR AL=BE=GA=0.0
C PENTRU PERSPECTIVA PARALELA OARECARE PROP=1 IAR
C AL=0.850 BE=0.500 GA=0.450
C PENTRU AXONOMETRIA IZOMETRICA PROP=1 IAR AL=BE=GA=0.578
TYPE 201,N,X0,Y0,Z0,X2,Y2,Z2,A,B,C,D,AL1,BE1,GA1,AL2,BE2,GA2,
*AL,BE,GA
201 FORMAT('0','DETERMINAREA PROIECTIEI CUNOSCIND'/' ','N=',I4,1X,'X0='
1',F9.3,1X,'Y0=',F9.3,1X,'Z0=',F9.3,1X,'X2=',F9.3/' ','Y2=',F9.3,1X
2',Z2=',F9.3,1X,'A=',F9.3,1X,'B=',F9.3,1X,'C=',F9.3/' ','D=',F9.3,1
3X,'AL1=',F9.3,1X,'BE1=',F9.3,1X,'GA1=',F9.3/' ','AL2=',F9.3,1X,'BE
42=',F9.3,1X,'GA2=',F9.3/' ','AL=',F9.3,1X,'BE=',F9.3,1X,'GA=',F9.3
5/'/'',2X,'I',17X,'X1(I)',16X,'Y1(I)')
TYPE 203,A1,A2,B1,B2,CW,DW
203 FORMAT(' ','6F10.3/')
CALL CALCF(A1,A2,B1,B2,CW,DW,XP,YP,ZP)
DO 20 I=1,N
IF(PROP.NE.1) GO TO 25
LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*AL+B*BE+C*GA)
X=XP(I)-AL*LAMBDA
Y=YP(I)-BE*LAMBDA
Z=ZP(I)-GA*LAMBDA
GO TO 23

```

```

25 DVP=SQRT((X0-XP(I))**2+(Y0-YP(I))**2+(Z0-ZP(I))**2)
   CL=(X0-XP(I))/DVP
   CM=(Y0-YP(I))/DVP
   CN=(Z0-ZP(I))/DVP
   LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*CL+B*CM+C*CN)
   X=XP(I)-CL*LAMBDA+8
   Y=YP(I)-CM*LAMBDA+8
   Z=ZP(I)-CN*LAMBDA+10
23 X1(I)=AL1*(X-X2)+BE1*(Y-Y2)+GA1*(Z-Z2)
   Y1(I)=AL2*(X-X2)+BE2*(Y-Y2)+GA2*(Z-Z2)
20 CONTINUE
   AUXX=X1(1)
   DO 18 J=2,N
     IF(AUXX.LT.X1(J)) GO TO 18
     AUXX=X1(J)
18 CONTINUE
   AUXY=Y1(1)
   DO 19 J=2,N
     IF(AUXY.LT.Y1(J)) GO TO 19
     AUXY=Y1(J)
19 CONTINUE
   DO 30 K=1,N
     X1(K)=10.*X1(K)-10.*AUXX
     Y1(K)=10.*Y1(K)-10.*AUXY
30 CONTINUE
   IF(TRAS.NE.1) GO TO 35
   RN=N
   PAS=SQRT(RN)
   IPAS=PAS
   I=1
   DO 40 J=1,IPAS
     I2=J*IPAS
     CALL PLOT(X1(I),Y1(I),3)
     DO 333 IJ=I+1,I2
       CALL PLOT(X1(IJ),Y1(IJ),2)
       I=J*IPAS+1
40 CONTINUE
   DO 17 J=1,IPAS
     DO 22 I=1,IPAS
       I1=IPAS*(I-1)+J
       XT(I)=X1(I1)
       YT(I)=Y1(I1)
22 CONTINUE
     CALL PLOT(XT(1),YT(1),3)
     DO 444 IJ=2,IPAS
       CALL PLOT(XT(IJ),YT(IJ),2)
444 CONTINUE
   CALL PLOT(0.,0.,999)
800 CONTINUE
35 STOP
   END

```

```

SUBROUTINE CALCF(A1,A2,B1,B2,CW,DW,XP,YP,ZP)
DIMENSION XP(125),YP(125),ZP(125)
C   ELEMENTELE INTRODUSE SINT:
C   - A1,A2 = DOMENIUL DE DEFINITIE PT X
C   - B1,B2 =DOMENIUL DE DEFINITIE PT Y
C   - CW   = PASUL DE CREȘTERE PT X
C   - DW   = PASUL DE CREȘTERE PT Y
PI=3.1416
PC=(A2-A1)/CW
PD=(B2-B1)/DW
MC=PC
ND=PD
G=A1
H=B1
M=0
DO 131 I=1,MC
DO 130 J=1,ND
M=M+1
ANG=PI*G
XP(M)=(2.*H**3-3.*H**2+2.)*COS(ANG)
YP(M)=2.*H**3-3.*H**2+1.+(2.*H**3-3.*H**2+2.)*SIN(ANG)
ZP(M)=8.*H**3-8.*H**2+2.*H
H=H+DW
130 CONTINUE
H=A1
G=G+CW
131 CONTINUE
RETURN
END

```

5.5.4 Programul interactiv pentru cazul în care curba este exprimată parametric.

```

DIMENSION XP(125),YP(125),ZP(125),X1(125),Y1(125)
REAL*4 NP,LAMBDA
INTEGER PROP
CALL PLOTS(0,1,7)
TYPE 302
302 FORMAT(' ORIGINE GRILA (X,Y)?')
READ(5,*,ERR=301,END=800) XX,YY
TYPE 304
304 FORMAT(' PROP (I4)?')
READ(5,101,ERR=303,END=800) PROP
101 FORMAT(I4)
C   PENTRU PERSPECTIVA CENTRALA PROP=0 IAR AL=BE=GA=0.0
C   PENTRU PERSPECTIVA PARALELA OARECARE PROP=1 IAR
C   AL=-0.614 BE=0.614 GA=-0.502
C   PENTRU AXONOMETRIA IZOMETRIA PROP=1 IAR AL=BE=GA=0.578
401 TYPE 402
402 FORMAT(' N (I4)?')
READ(5,101,ERR=401,END=800) N
403 TYPE 404
404 FORMAT(' X0,Y0,Z0(3F10.5)?')
READ(5,102,ERR=403,END=800) X0,Y0,Z0
405 TYPE 406
406 FORMAT(' X2,Y2,Z2(3F10.5)?')
READ(5,102,ERR=405,END=800) X2,Y2,Z2
407 TYPE 408
408 FORMAT(' AL1,BE1,GA1(3F10.5)?')
READ(5,102,ERR=407,END=800) AL1,BE1,GA1

```

```

407 TYPE 410
410 FORMAT(' AL2, BE2, GA2(3F10.5)?'*)
READ(5, 102, ERR=409, END=800) AL2, BE2, GA2
102 FORMAT(3F10.5)
411 TYPE 412
412 FORMAT(' AL, BE, GA(3F10.5)?'*)
READ(5, 102, ERR=411, END=800) AL, BE, GA
413 TYPE 414
414 FORMAT(' A, B, C, D(4F10.5)?'*)
READ(5, 103, ERR=413, END=800) A, B, C, D
103 FORMAT(4F10.5)
415 TYPE 416
416 FORMAT(' A1, A2, B1, B2, CW, DW(6F10.3)?'*)
READ(5, 104, ERR=415, END=800) A1, A2, B1, B2, CW, DW
104 FORMAT(6F10.3)
TYPE 201, N, X0, Y0, Z0, X2, Y2, Z2, A, B, C, D, AL1, BE1, GA1, AL2, BE2, GA2,
*AL, BE, GA
201 FORMAT('O', 'DETERMINAREA PROIECTIEI CUNOSCIND'// ' ', 'N=', I4, 1X, 'X0=
1', F9.3, 1X, 'Y0=', F9.3, 1X, 'Z0=', F9.3, 1X, 'X2=', F9.3// ' ', 'Y2=', F9.3, 1X
2', 'Z2=', F9.3, 1X, 'A=', F9.3, 1X, 'B=', F9.3, 1X, 'C=', F9.3// ' ', 'D=', F9.3, 1
3X, 'AL1=', F9.3, 1X, 'BE1=', F9.3, 1X, 'GA1=', F9.3// ' ', 'AL2=', F9.3, 1X, 'BE
42=', F9.3, 1X, 'GA2=', F9.3// ' ', 'AL=', F9.3, 1X, 'BE=', F9.3, 1X, 'GA=', F9.3
5// ' ', 2X, 'I', 17X, 'X1(I)', 16X, 'Y1(I)')//)
TYPE 203, A1, A2, B1, B2, CW, DW
203 FORMAT(' ', 6F10.3/)
CALL CALCS(A1, A2, B1, B2, CW, DW, XP, YP, ZP, MC)
DO 20 I=1, MC
IF (PROP.NE.1) GO TO 15
LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*AL+B*BE+G*GA)
X=XP(I)-AL*LAMBDA
Y=YP(I)-BE*LAMBDA
Z=ZP(I)-GA*LAMBDA
GO TO 13
15 DVP=SQRT((X0-XP(I))**2+(Y0-YP(I))**2+(Z0-ZP(I))**2)
CL=(X0-XP(I))/DVP
CN=(Z0-ZP(I))/DVP
CM=(Y0-YP(I))/DVP
LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*CL+B*CM+C*CN)
X=XP(I)-CL*LAMBDA+S
Y=YP(I)-CM*LAMBDA+S
Z=ZP(I)-CN*LAMBDA+10
13 X1(I)=AL1*(X-X2)+BE1*(Y-Y2)+GA1*(Z-Z2)
Y1(I)=AL2*(X-X2)+BE2*(Y-Y2)+GA2*(Z-Z2)
20 CONTINUE
AUXX=X1(1)
DO 18 J=2, MC
IF (AUXX.LT. X1(J)) GO TO 13
AUXX=X1(J)
18 CONTINUE
AUXY=Y1(1)
DO 19 J=2, MC
IF (AUXY.LT. Y1(J)) GO TO 19
AUXY=Y1(J)

```

```

19 CONTINUE
   DO 300 K=1,MC
     X1(K)=10.*(X1(K)-AUXX)
     Y1(K)=10.*(Y1(K)-AUXY)
     X1(K)=X1(K)*4.
     Y1(K)=Y1(K)*4.
300  CONTINUE
     CALL PLOT(X1(1),Y1(1),3)
     DO 666 IJ=2,MC
       TYPE *,X1(IJ),Y1(IJ)
666  CALL PLOT(X1(IJ),Y1(IJ),2)
     DO 22 M=1,MC
       ZP(M)=0
22  CONTINUE
     DO 21 I=1,MC
       IF (PROP.NE.1) GO TO 30
       LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*AL+B*BE+C*GA)
       X=XP(I)-AL*LAMBDA
       Y=YP(I)-BE*LAMBDA
       Z=ZP(I)-GA*LAMBDA
       GO TO 36
30  DVP=SQRT((X0-XP(I))**2+(Y0-YP(I))**2+(Z0-ZP(I))**2)
       CL=(X0-XP(I))/DVP
       CM=(Y0-YP(I))/DVP
       CN=(Z0-ZP(I))/DVP
       LAMBDA=(A*XP(I)+B*YP(I)+C*ZP(I)+D)/(A*CL+B*CM+C*CN)
       X=XP(I)-CL*LAMBDA+8
       Y=YP(I)-CM*LAMBDA+8
       Z=ZP(I)-CN*LAMBDA+10
36  X1(I)=AL1*(X-X2)+BE1*(Y-Y2)+GA1*(Z-Z2)
     Y1(I)=AL2*(X-X2)+BE2*(Y-Y2)+GA2*(Z-Z2)
21  CONTINUE
     DO 3000 K=1,MC
       X1(K)=10.*(X1(K)-AUXX)
       Y1(K)=10.*(Y1(K)-AUXY)
       X1(K)=X1(K)*4.
       Y1(K)=Y1(K)*4.
3000 CONTINUE
     CALL PLOT(X1(1),Y1(1),3)
     DO 777 IJ=2,MC
       TYPE *,X1(IJ),Y1(IJ)
777  CALL PLOT(X1(IJ),Y1(IJ),2)
     CALL PLOT(X1(1),Y1(1),3)
     XT=SQRT(10800.)
     XI=XT+X1(1)
     XS=X1(1)-XT
     YI=Y1(1)-60.
     YS=Y1(1)+120.
     CALL PLOT(X1(1),YS,2)
     CALL PLOT(XS,YI,3)
     CALL PLOT(X1(1),Y1(1),2)
     CALL PLOT(XI,YI,2)
800  CONTINUE
     CALL PLOT(0.,0.,999)
     STOP
END

```

```

SUBROUTINE CALCS(A1,A2,B1,B2,CW,DW,XP,YP,ZP,MC)
C SUBROUTINE CALCS PENTRU EXPRIMAREA PARAMETRICA A CURBELOR SPATIALE
DIMENSION XP(125),YP(125),ZP(125)
PC=(A2-A1)/CW
MC=PC
T=A1
M=0
DO 130 I=1,MC
M=M+1
XP(M)=13.*T*T*T-17.*T*T+5.*T
YP(M)=-2.*T*T*T+3.*T*T
ZP(M)=5.*T*T*T-10.*T*T+5.*T
T=T+CW
130 CONTINUE
RETURN
END

```

Altă variantă în FORTRAN 77 a acestor programe este următoarea:

```

C Program --- SUPEXP ---
C Vizualizarea functiilor exprimate parametric sau explicit
C Functiile pot fi de forma :
C 1. Parametric - X = X ( U , V )
C Y = Y ( U , V )
C Z = Z ( U , V )
C 2. Explicit -- Z = Z ( X , Y )
C Functiile se introduc in locul specificat in procedura CALCF
C Alti parametri ceruti pentru reprezentare :
C A1 , A2 - Domeniul de variatie pentru primul parametru
C B1 , B2 - Domeniul de variatie pentru al doilea parametru
C NU - Numar de curbe U = ct.
C NV - Numar de curbe V = ct.
C Parametri ceruti pentru proiectie :
C XV , YV , ZV - Coordonate pentru punctul de vedere
C IPR - Tipul proiectiei IPR=0 - paralela IPR #0 - centrala
C -----
C DIMENSION X(1000),Y(1000),Z(1000),X1(1000),Y1(1000),Z1(1000)
C CALL ASSIGN(1,'II:~')
C setarea unor parametri legati de proiectie
C CALL INISP
C CALL SETIR(10.)
C CALL SETOIM(S.,S.)
C CALL INI(1,0,-5.,-5.,5.,5.,100,100,600)
C introducerea datelor despre functie
C TYPE 1
1 FORMAT(' Introduceti A1,A2,B1,B2 [4F10.3] : ',*)
ACCEPT 2,A1,A2,B1,B2
2 FORMAT(4F10.3)
C TYPE 3
3 FORMAT(' Introduceti NU , NV : ',*)
ACCEPT 4,NU,NV
4 FORMAT(2I4)
C calculul valorilor functiei si a valorilor extreme
* CALL CALCF(A1,A2,B1,B2,NU,NV,X,Y,Z,
* XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)
C setarea punctului spre care se priveste
CALL SETC((XMIN+XMAX)/2.,(YMIN+YMAX)/2.,(ZMIN+ZMAX)/2.)
C introducerea datelor despre proiectie
1000 TYPE 5
5 FORMAT(' Introduceti datele despre proiectie',/,
* ' XV,YV,ZV,IPR [3F10.3,I2] : ',*)
ACCEPT 6,XV,YV,ZV,IPR
6 FORMAT(3F10.3,I2)
C traseea functiei
CALL DESEN(NU,NV,X,Y,Z,X1,Y1,Z1,IPR)
C alte date de proiectie ?
C TYPE 7
7 FORMAT(' Doriti alta proiectie a functiei [NU=0] : ',*)
ACCEPT 8,J
8 FORMAT(I1)
IF(J.NE.0) GO TO 1000
CALL EOF
STOP
END

```



```

C      Procedura -- CALCF --
C      Calculul valorilor functiei si a valorilor extreme pe X,Y si Z
C
C      SUBROUTINE CALCF(A1,A2,B1,B2,NU,NV,XP,YP,ZP,
*      XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)
C      DIMENSION XP(NU,NV),YP(NU,NV),ZP(NU,NV)
C setarea valorilor extreme
      XMIN=10.*10
      XMAX=-XMIN
      YMIN=XMIN
      YMAX=XMAX
      ZMIN=XMIN
      ZMAX=XMAX
C pasul de discretizare
      PU=(A2-A1)/FLOAT(NU-1)
      PV=(B2-B1)/FLOAT(NV-1)
C calculul valorilor din matrice
      DO 2 J=1,NV
      V=B1+PV*(J-1)
      DO 2 I=1,NU
      U=A1+PU*(I-1)
C aici se introduce forma functiei (parametrica sau explicita)
C -----
      XP(I,J)=U
      YP(I,J)=V
      ZP(I,J)=(U**3-3.*V*U**2)/25.
C modificarea valorilor extreme
      IF(XP(I,J).GT.XMAX)XMAX=XP(I,J)
      IF(XP(I,J).LT.XMIN)XMIN=XP(I,J)
      IF(YP(I,J).GT.YMAX)YMAX=YP(I,J)
      IF(YP(I,J).LT.YMIN)YMIN=YP(I,J)
      IF(ZP(I,J).GT.ZMAX)ZMAX=ZP(I,J)
      IF(ZP(I,J).LT.ZMIN)ZMIN=ZP(I,J)
2      CONTINUE
      RETURN
      END
C
C

```

```

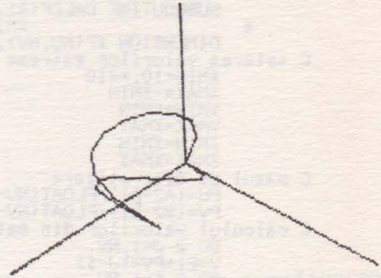
C      Procedura DESEN
C      Trasarea suprafeței prin curba U= ct. și V= ct.
C
C      SUBROUTINE DESEN(NU,NV,XP,YP,ZP,XP1,YP1,ZP1,IPR)
*      DIMENSION XP(NU,NV),YP(NU,NV),ZP(NU,NV),XP1(NU,NV),YP1(NU,NV),
      ZP1(NU,NV)
C calculul noilor coordonate atasate planului de proiectia
      DO 1 I=1,NU
      DO 1 J=1,NV
1      CALL NEWC(XP(I,J),YP(I,J),ZP(I,J),XP1(I,J),YP1(I,J),ZP1(I,J))
C trasarea curbelor U = ct.
      DO 3 I=1,NU
      DO 3 J=1,NV-1
      CALL CLIPS(XP1(I,J),YP1(I,J),ZP1(I,J),XP1(I,J+1),YP1(I,J+1),
*      ZP1(I,J+1),IPR,XP1,YP1,XPR2,YPR2,NP)
      IF(NP.EQ.0)CALL LIN(XPR1,YPR1,XPR2,YPR2)
3      CONTINUE
C trasarea curbelor V = ct.
      DO 4 J=1,NV
      DO 4 I=1,NU-1
      CALL CLIPS(XP1(I,J),YP1(I,J),ZP1(I,J),XP1(I+1,J),YP1(I+1,J),
*      ZP1(I+1,J),IPR,XP1,YP1,XPR2,YPR2,NP)
      IF(NP.EQ.0)CALL LIN(XPR1,YPR1,XPR2,YPR2)
4      CONTINUE
      RETURN
      END

```

5.5.5 Aplicații.

În figura 5.14 a este reprezentată în izometrie curba

$$\begin{cases} x(t) = \frac{9}{2}t - \frac{9}{2}t^2 \\ y(t) = -\frac{7}{2}t + \frac{27}{2}t^2 - 9t^3; \quad t \in [0,1] \\ z(t) = 9t - \frac{45}{2}t^2 + \frac{27}{2}t^3 \end{cases}$$

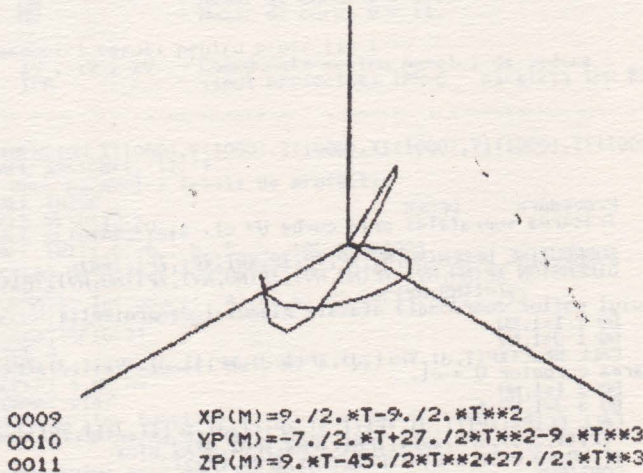


precum și proiecția sa orizontală. Au fost considerate 11 puncte pe curbă.

În figura 5.14 b este reprezentată tot în izometrie curba:

$$\begin{cases} x(t) = 13t^2 - 17t^2 + 5t \\ y(t) = -2t^3 + 3t^2 \\ z(t) = 5t^3 - 10t^2 + 5t \end{cases}$$

pentru $t \in [0, 1]$



5.6. Generalități asupra vizualizării obiectelor spațiale

5.6.1 Influența echipamentelor electronice utilizate

Este de la sine înțeles faptul că natura reprezentării este profund influențată de calitatea echipamentelor electronice utilizate.

Ecranul de vizualizare cu baleiaj de televiziune sau cu tub de memorie permite reprezentarea obiectelor în semi-tonuri cu condiția de a dispune de un echipament capabil să memorizeze un număr important de informații și să le transmită în același timp foarte rapid.

De exemplu o imagine de $256 + 256$ puncte utilizând 32 nivele de gri necesită o memorie de 2^{21} biți și o viteză de transmitere de ordinul a 60.000.000 biți pe secundă pentru a fi regenerată la fiecare 1/30 dintr-o secundă.

Astfel pot fi folosiți — în funcție de terminalul de ieșire — fie algoritmi de eliminare a liniilor ascunse, fie algoritmi de studiu ai suprafețelor ascunse (sau vizibile). În primul caz ne vom mulțumi cu reprezentarea obiectelor fie prin scheletul lor, fie prin conturul lor aparent, eliminând părțile ascunse și folosind calculul în funcție de precizia terminalului de ieșire.

În cel de al doilea caz reprezentarea obiectelor se face în funcție de calculul gradului de luminozitate al fiecărei porțiuni de pe suprafața obiectului considerând una sau mai multe surse de lumină date. Vor rezulta astfel porțiunile de suprafață vizibile.

În ultimul timp **eliminarea liniilor ascunse** este considerată din ce în ce mai mult ca un subprodus al căutării suprafețelor vizibile și astfel majoritatea celor mai noi algoritmi sînt răsolați pe reprezentarea obiectelor **prin suprafețele lor vizibile**.

5.7. Transformări în S^2 și S^3

5.7.1 Coordonate omogene. Utilizarea coordonatelor omogene revine la a reprezenta obiectele din spațiul cu N dimensiuni în spațiul cu $N + 1$ dimensiuni, astfel încît o proiecție perspectivă particulară să redea spațiul cu N dimensiuni. Se poate de asemenea considera că în fapt se adaugă cîte o coordonată suplimentară fiecărui vector, un factor de omotetie, astfel încît vectorul nu se schimbă după multiplicarea cu o constantă. De exemplu, un punct din spațiul bidimensional va fi reprezentat prin vectorul (X, Y, W) și va fi manipulat ca un punct aparținînd spațiului tridimensional. Pentru a găsi punctul inițial se va calcula

$$X^1 = \frac{X}{W}$$

$$Y^1 = \frac{Y}{W}$$

Spațiul astfel creat este format de drepte și de plane trecînd toate prin origine. Acest fapt permite reprezentarea tuturor transformărilor printr-o matrice 3×3 și reduce totul la operațiuni matriciale. Aceste operațiuni fiind ușor cablate, se înțelege interesul deosebit al utilizării coordonatelor omogene pentru producerea desenelor cu ajutorul unei console de vizualizare. Este deasemenea de notat faptul că punctele situate la infinit nu constituie cazuri particulare deoarece sînt simple reprezentări printr-o coordonată suplimentară nulă. Acest lucru permite de exemplu, calculul punctului de intersecție dintre două drepte în toate cazurile, chiar și cînd există riscul de a fi paralele.

Coordonatele omogene au fost utilizate pentru prima dată în infografia interactivă de L.G. Roberts.

5.7.2 Puncte și drepte (forma implicită)

- un punct va fi reprezentat prin

$$V = (X, Y, W)$$

- o dreaptă va fi reprezentată prin

$$\gamma = \begin{vmatrix} a \\ b \\ c \end{vmatrix}$$

- Ecuația unei drepte va fi

$$V\gamma = 0$$

Dacă produsul scalar al celor doi vectori este nul, punctul este situat pe dreaptă. În caz contrar, semnul indică în ce parte a dreptei este situat punctul.

- Ecuația dreptei definită de punctele V_0 și V_1 :

$$\text{Fie } \begin{cases} aX_0 + bY_0 + cW_0 = 0 \\ aX_1 + bY_1 + cW_1 = 0 \end{cases}$$

Rezultă

$$a = c \frac{W_0Y_1 - Y_0W_1}{X_1Y_0 - X_0Y_1}$$

$$b = c \frac{X_0W_1 - X_1W_0}{X_1Y_0 - X_0Y_1}$$

Alegînd convenabil valoarea c obținem ecuația implicită a dreptei

$$F(X, Y, W) = (Y_1W_0 - Y_0W_1)X + (X_0W_1 - X_1W_0)Y + (X_1Y_0 - X_0Y_1)W = 0$$

$$\gamma' = |(Y_1W_0 - Y_0W_1), (X_0W_1 - X_1W_0), (X_1Y_0 - X_0Y_1)|$$

- Intersecția dintre două drepte:

$$\gamma'_0 = (a_0, b_0, c_0); \quad \gamma'_1 = (a_1, b_1, c_1)$$

Rezultă

$$V = |(b_1c_0 - b_0c_1), (a_0c_1 - a_1c_0), (a_1b_0 - a_0b_1)|$$

5.7.3 Transformări în S^2

Fie H o matrice de transformare 3×3 . Transformarea punctului V se obține prin produsul VH iar transformarea dreptei se obține prin produsul $H^{-1}\gamma$.

● Translația:

Pentru a aduce punctul $V(X, Y, 1)$ în origine

$$T_1(V) = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -X & -Y & 1 \end{vmatrix}$$

Transformarea opusă este obținută prin

$$T_2(V) = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ X & Y & 1 \end{vmatrix}$$

● Rotația în jurul originii de unghiul α :

$$R(\alpha) = \begin{vmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

● Schimbarea scării:

Fie S_x, S_y factorii de scară ai fiecărei coordonate

$$S = \begin{vmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Dacă:

- $0 < \text{factorul de scară} < 1$ imaginea se micșorează
- factorul de scară = 1 imaginea nu se schimbă
- $1 < \text{factorul de scară} < \infty$ imaginea se mărește

5.7.4 Reprezentarea conicelor. Forma implicită

Ecuția unei conice este de forma

$$VCV' = 0 \quad \text{unde } C: (3 \times 3) \text{ aleasă simetrică}$$

● Cercul de centru $C(X_c, Y_c, 1)$ care trece prin punctul $P(X_0, Y_0, 1)$

$$C = \begin{vmatrix} 1 & 0 & -X_c \\ 0 & 1 & -Y_c \\ -X_c & -Y_c & -(X_0^2 + Y_0^2) + 2X_0X_c + 2Y_0Y_c \end{vmatrix}$$

● Elipsa de centru $C(X, Y, 1)$ cu semiaxele a și b (Fig. 5.15) a cărei axă principală face unghiul α cu axa ox , are ecuația

$$E = T_1(c) R(\alpha) \begin{vmatrix} b^2 & 0 & 0 \\ 0 & a^2 & 0 \\ 0 & 0 & -a^2b^2 \end{vmatrix} R(-\alpha) T_2(c)$$

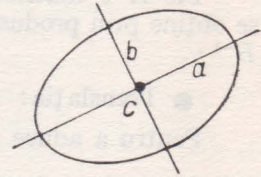


Fig. 5.15

● Hiperbola (fig. 5.16) va avea ecuația

$$H = T_1(c) R(\alpha) \begin{vmatrix} b^2 & 0 & 0 \\ 0 & -a^2 & 0 \\ 0 & 0 & -a^2b^2 \end{vmatrix} R(-\alpha) T_2(c)$$

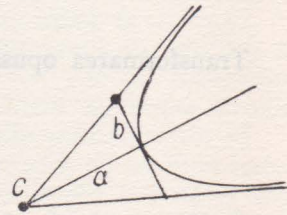


Fig. 5.16

● Parabola (fig. 5.17) va avea ecuația

$$P = T_1(c) R(\alpha) \begin{vmatrix} 1 & 0 & 0 \\ 0 & 0 & -p \\ 0 & -p & 0 \end{vmatrix} R(-\alpha) T_2(c)$$

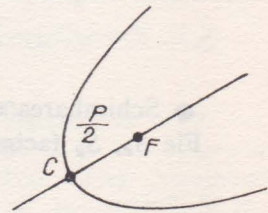


Fig. 5.17

5.7.5 Ecuațiile parametrice

Utilizarea formei implicite nu este întotdeauna utilă în cazul tehnicilor grafice, deoarece matricele obținute descriu o curbă întreagă atunci când în general nu avem nevoie decât de un segment de dreaptă sau de un arc de curbă. Dimpotrivă utilizarea reprezentării parametrice permite exprimarea variabilelor independente unele de altele în funcție de o altă variabilă numită variabilă parametrică. Această variabilă poate fi interpretată și ca fiind timpul necesar de a merge de la un punct la altul al curbei. În acest caz reprezentarea unei mișcări devine foarte ușoară.

● Ecuația parametrică a unui segment de dreapta:

Fie punctele (X_0, Y_0) și (X_1, Y_1)

Avem

$$X = f(t) = (X_1 - X_0)t + X_0$$

$$Y = g(t) = (Y_1 - Y_0)t + Y_0$$

$$W = h(t) = (W_1 - W_0)t + W_0$$

de unde

$$(XYW) = t(1) \begin{vmatrix} X_1 - X_0 & Y_1 - Y_0 & W_1 - W_0 \\ X_0 & Y_0 & W_0 \end{vmatrix}$$

Referitor la intervalul de variație pentru variabila t , se poate considera că ecuațiile parametrice descriu mișcarea unui punct în lungul unei curbe. În fiecare moment t coordonatele sale (X, Y, W) sînt date de

$$X = f(t)$$

$$Y = g(t)$$

$$W = h(t)$$

Mulțimea valorilor luate de t conduce la mulțimea pozițiilor ocupate de punct, ceea ce definește în general un arc de curbă. În majoritatea timpului acest arc este descris între momentele t_1 și t_2 . Adeseori se ia ca interval de referință intervalul $[0, 1]$. Astfel pornind de la matricea A se construiește o matrice A' unde $A' = PA$.

$$P(s) = t_1 + (t_2 - t_1)S \text{ unde } \begin{cases} P(0) = t_1 \\ P(1) = t_2 \end{cases}$$

În cazul dreptelor

$$P = \begin{vmatrix} t_2 - t_1 & 0 \\ t_1 & 1 \end{vmatrix}$$

În cazul conicelor

$$P = \begin{vmatrix} (t_2 - t_1)^2 & 0 & 0 \\ 2t_1(t_2 - t_1) & (t_2 - t_1) & 0 \\ t_1^2 & t_1 & 1 \end{vmatrix}$$

5.7.5 Transformări în S^3 (V. [17])

Formularea matricială a transformărilor din spațiul bidimensional S^2 poate fi extinsă și la spațiul tridimensional S^3 . De fapt transformarea este o singură entitate, o matrice, secvențele de transformări putînd fi compuse într-o singură transformare care are același efect ca și aplicarea întregii secvențe. Vom utiliza de asemenea coordonatele omogene pentru a reprezenta matricele de transformare.

● Translația

Transformarea care translatează un punct (X, Y, Z) spre un nou punct (X_1, Y_1, Z_1) este:

$$|X_1 Y_1 Z_1 \ 1| = |XYZ \ 1| \cdot \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{vmatrix}$$

unde T_x , T_y și T_z sînt componente ale translației în direcțiile X , Y și Z .

● Rotația

Transformările rotației în spațiul tridimensional sînt mai complexe în comparație cu spațiul bidimensional prin faptul că este necesară determinarea unei axe de rotație. Specificarea axei de rotație include atît direcția cît și localizarea. Este necesar să se definească rotațiile în raport cu cele trei axe ale reperului O_x , O_y , O_z .

● Rotația în jurul axei O_x (fig. 5.18) prin punctul $(0, 0, 0)$ este

$$|X_1Y_1Z_1| = |XYZ| \cdot \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

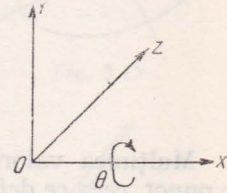


Fig 5. 18

Unghiul de rotație θ este măsurat în sens orar în jurul originii, privind originea dintr-un punct de pe axa X_+ .

Transformarea matricei afectează numai valorile lui Y și Z și este considerată pentru un sistem de coordonate șurub drept.

● Rotația în jurul axei OY (fig. 5.19), prin punctul $(0, 0, 0)$ este

$$|X_1Y_1Z_1| = |XYZ| \cdot \begin{vmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

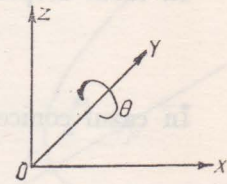


Fig 5.19

● Rotația în jurul axei OZ (fig. 5.20), prin punctul $(0, 0, 0)$ este

$$|X_1Y_1Z_1| = |XYZ| \cdot \begin{vmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

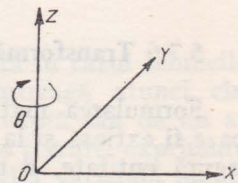


Fig 5.20

● Schimbarea scării este

$$|X_1Y_1Z_1| = |XYZ| \cdot \begin{vmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Transformarea inversă

Fiecare din transformările de mai sus are o inversă care reprezintă transformarea simetrică opusă.

De exemplu matricea inversă a translației este

$$\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -T_x & -T_y & -T_z & 1 \end{vmatrix}$$

care anulează astfel translația

Sau, alt exemplu, inversa rotației în jurul axei OX este

$$\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & \sin \theta & 0 \\ 0 & -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-\theta) & -\sin(-\theta) & 0 \\ 0 & \sin(\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

reprezentând o rotație de același unghi, în jurul aceleiași axe dar în sens opus.

De obicei inversa transformării T se notează prin T^{-1} .

Matricea care reprezintă inversa oricărei matrici de transformare poate fi determinată cu ușurință printr-un program computer.

● Juxtapunerea (compunerea) transformărilor

Aplicarea succesivă a două transformări T_1 și T_2 poate fi simplificată prin aplicarea unei singure transformări T_3 .

Matricea T_3 este produsul matricelor T_1 și T_2 .

Astfel dacă T_1 transformă punctul (X, Y, Z) în (X_1, Y_1, Z_1) și dacă T_2 generează punctul (X_2, Y_2, Z_2) , obținem

$$\begin{aligned} |X_1 Y_1 Z_1 \ 1| &= |XYZ \ 1| T_1 \\ |X_2 Y_2 Z_2 \ 1| &= |X_1 Y_1 Z_1 \ 1| T_2 \end{aligned}$$

sau

$$|X_2 Y_2 Z_2 \ 1| = (|XYZ \ 1| T_1) T_2 = |XYZ \ 1| (T_1 T_2)$$

În înmulțirea matricelor de transformare se va păstra ordinea de aplicare a transformărilor.

5.7.7 Proiecția ortogonală. Fie punctul $I(X, Y, Z, 1)$

— Vederea frontală (proiecția pe planul Ox, O_y) este

$$|XYZ \ 1| \cdot \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = |XYO1|$$

— Vederea de sus: se face o rotație în jurul axei O_x și se proiectează pe planul O_x, O_y .

$$|XYZ1| \cdot \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \cdot \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = |XZO1|$$

— Vedere laterală: se face o rotație în jurul axei O și se proiectează pe planul O_x, O_y .

$$|XYZ1| \cdot \begin{vmatrix} 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \cdot \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = |ZY01|$$

5.8. Proiecția perspectivivă

Generarea unei imagini în perspectivă revine la a diviza coordonatele X și Y ale punctului prin profunzimea punctului, care de fapt este cota sa Z .

Punctul $P(X_0, Y_0, Z_0)$ va avea ca perspectivă $P_1(X_T, Y_T)$.

5.8.1 Sistemul de coordonate al ochiului (observatorului) $O X_0 Y_0 Z_0$ considerat în fig. 5.21 și fig. 5.22 este folosit pentru a converti punctele din spațiul obiect (X, Y, Z) în puncte din sistemul de coordonate (X_0, Y_0, Z_0) al ochiului.

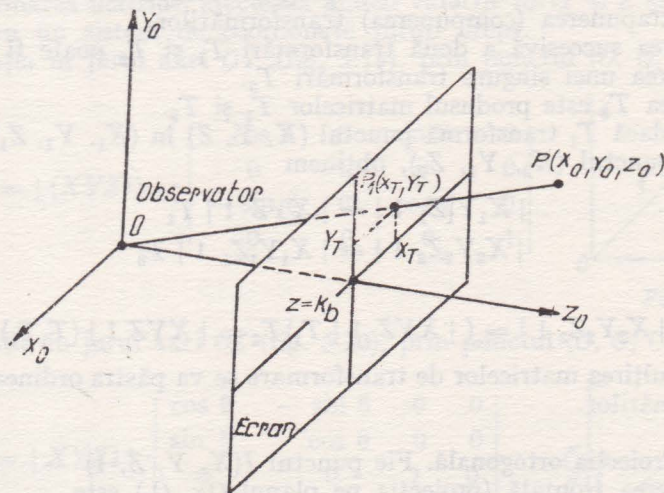


Fig. 5.21

Această transformare poate fi exprimată prin relația

$$|X_0 Y_0 Z_0 1| = |XYZ 1| V$$

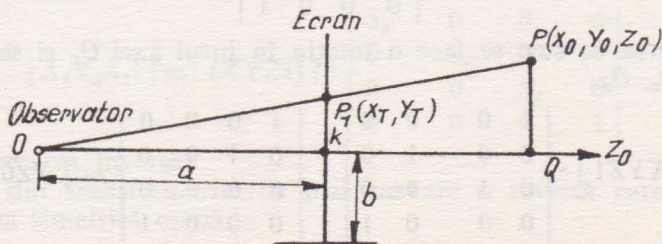


Fig. 5.22

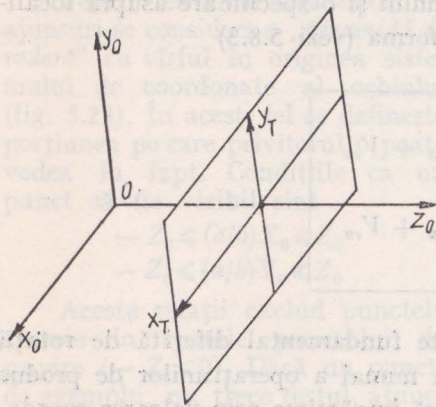


Fig. 5.23

Așadar pentru a calcula poziția unui punct din spațiul obiect pe ecranul display-ului este necesar să transformăm (transpunem) punctul în sistemul de coordonate al ochiului care are originea fixată în punctul de privire și axa Z_0 îndreptată în direcția privirii. Acest sistem de coordonate se deplasează (se mișcă și se rotește) solidar cu direcția de privire Z_0 . Parametrii acestei transformări determină în mod complet perspectiva iar matricea transformării poate fi obținută prin juxtaponerea mai multor rotații și translații.

Sistemul de coordonate al ochiului este de tipul șurub stingă axa Z fiind îndreptată de la punctul de privire înainte. Axa X_0 este îndreptată spre dreapta iar axa Y_0 este îndreptată în sus. În acest fel axele X_0 și Y_0 se aliniază cu axele X și Y de pe ecranul display-ului (fig. 5.23). Din fericire imaginea perspectivă a unei linii poate fi ușor generată transformând numai punctul de capăt și desenând linia care unește aceste două puncte de capăt transformate.

Procesul generării perspective începe prin transformarea coordonatelor punctelor din spațiul obiect în sistemul de coordonate al ochiului pentru fiecare punct. Un generator de vectori poate genera linia între punctele considerate unde X_T și Y_T sînt coordonatele perspective. Ele pot fi deduse din asemănarea triunghiurilor (vezi 5.2.4).

$$\Delta OKP_1 \sim \Delta OQP$$

Avem

$$\frac{Y_T}{a} = \frac{Y_0}{Z_0} \quad \text{și} \quad \frac{X_T}{a} = \frac{X_0}{Z_0}$$

Coordonatele perspective X_T și Y_T sînt

$$X_T = \frac{aX_0}{Z_0}; \quad Y_T = \frac{aY_0}{Z_0}$$

sau dacă împărțim prin diviziunea b a laturei ecranului

$$X_T = \frac{aX_0}{bZ_0}; \quad Y_T = \frac{aY_0}{bZ_0}$$

Dacă se include în coordonatele ecranului și o specificare asupra localizării imaginii, relațiile pot fi scrise sub forma (vezi 5.8.3)

$$\begin{cases} X_T = \left(\frac{aX_0}{bZ_0} \right) V_{sx} + V_{cx} \\ Y_T = \left(\frac{aY_0}{bZ_0} \right) V_{sy} + V_{cy} \end{cases}$$

Această transformare perspectivă este fundamental diferită de rotații sau de translații care presupun utilizarea numai a operațiilor de produs și sumă. Transformarea perspectivă implică împărțirea prin valoarea coordonatei Z_0 .

Așadar a genera o imagine perspectivă implică divizarea prin adîncimea fiecărui punct.

Matricea de transformare va avea forma

$$P = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & Z_0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Relația exprimă în plus și independența sistemului de coordonate al ochiului față de sistemul de coordonate al ecranului. În general valorile parametrilor utilizați vor fi în concordanță cu sistemul de coordonate folosit de hardware-ul display-ului. În relație valorile coordonatelor punctelor în sistemul de coordonate al ochiului apar sub forme diferite fiind folosit numai raportul $\frac{a}{b}$ pentru caracterizarea imaginii. Dacă acest raport este mic, deschiderea va fi mare iar imaginea produsă va fi supra-unghiulară. O valoare mare a raportului $\frac{a}{b}$ înseamnă o deschidere mică, deci o imagine corespunzînd unei vederi telefoto. Efectul de perspectivă va fi exact și acceptabil pentru vederi de la alte distanțe care oricum întii trebuiesc testate.

5.8.2 Decuparea tridimensională a imaginii

Simpla aplicare a relațiilor pentru producerea unei imagini perspective poate avea două efecte nedorite:

- depășirea limitelor prescrise pentru aria imaginii;
- apariția pe ecran a unor puncte nedorite.

Pentru eliminarea acestor neajunsuri se consideră o „piramidă de vedere” cu vârful în originea sistemului de coordonate al ochiului (fig. 5.24). În acest fel se definește porțiunea pe care privitorul o poate vedea în fapt. Condițiile ca un punct să fie vizibil sînt

$$\begin{aligned} -Z_0 &\leq (a/b)X_0 \leq Z_0 \\ -Z_0 &\leq (a/b)Y_0 \leq Z_0 \end{aligned}$$

Aceste relații exclud punctele situate în spatele punctului de privire ($-Z_0 \leq 0$). Dacă un punct, de exemplu, nu trece testul, atunci nu va fi afișat. Liniile nu pot fi examinate atît de ușor ca punctele și de aceea este necesar să fie „decupate” în limitele „piramidei de vedere”. O linie sau o dreaptă poate fi respinsă ca invizibilă dacă nu intersectează piramida. Punctele de capăt sau extremitățile porțiunii vizibile ale dreptei sînt calculate în sistemul de coordonate al ochiului.

Ca structură algoritmul pentru decuparea liniilor față de piramida de vedere este similar cu algoritmul bidimensional. Este de asemenea necesară definirea unui nou sistem de coordonate al decupajului (cu indicele c) dar în termenii sistemului de coordonate al ochiului.

$$|X_c Y_c Z_c 1| = |X_0 Y_0 Z_0 1| N$$

unde

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

Condițiile din relația anterioară devin:

$$\begin{aligned} -Z_c &\leq X_c \leq Z_c \\ -Z_c &\leq Y_c \leq Z_c \end{aligned}$$

Algoritmul de decupare (de exemplu *Sutherland-Cohen*) are un cod de patru biți după cum urmează:

Primul bit:	Y_c este deasupra piramidei :	$Y_c > Z_c$
Al doilea bit:	Y_c este sub piramidă;	$Y_c < -Z_c$
Al treilea bit:	X_c este la dreapta piramidei	$X_c > Z_c$
Al patrulea bit:	X_c este la stînga piramidei	$X_c < -Z_c$

Utilizarea acestor coduri determină dacă linia testată poate fi respinsă sau acceptată. Se intersectează mai întii dreapta pentru a fi „decupată” cu unul din planele $X_c = Z_c$, $X_c = -Z_c$, $Y_c = Z_c$ și $Y_c = -Z_c$. Punctul de intersecție obținut divide dreapta în două segmente pentru care se aplică separat testele de acceptare sau de respingere și așa mai departe. Elaborarea unui algoritm de decupare poate constitui o aplicație utilă pentru cititor. În prezenta lucrare există asemenea algoritmi.

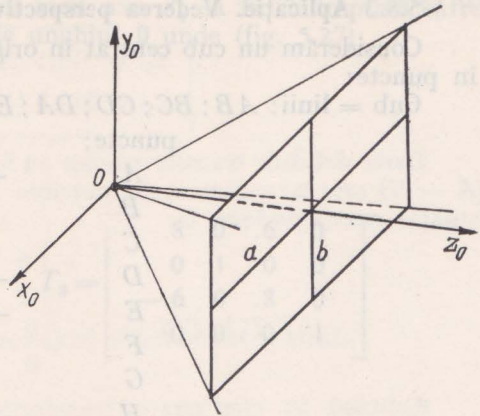
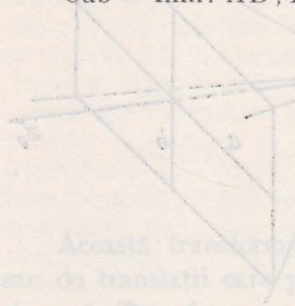


Fig. 5.24

5.8.3 Aplicație. Vederea perspectivă a unui cub.

Considerăm un cub centrat în originea spațiului obiect, definit de linii în puncte:

Cub = linii: $AB; BC; CD; DA; EF; FG; GH; HE; AE; BF; CG; DH$.

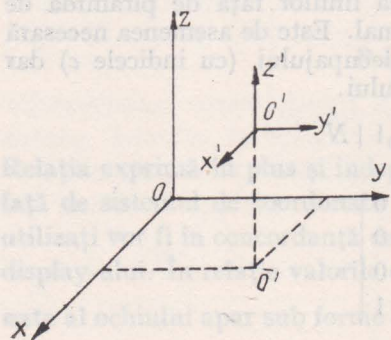


puncte:	X	Y	Z
A	-1	1	-1
B	1	1	-1
C	1	-1	-1
D	-1	-1	-1
E	-1	1	1
F	1	1	1
G	1	-1	1
H	-1	-1	1

Punctul de vedere (ochiul) are coordonatele $(6; 8; 7,5)$, iar axa de vedere Z_0 va fi îndreptată spre originea sistemului de coordonate. A mai rămas un grad de libertate, și anume o rotație arbitrară în jurul axei Z_0 : vom considera că axa X_0 aparține planului $Z = 7,5$.

Transformarea punctului de vedere poate fi stabilită după cum urmează:

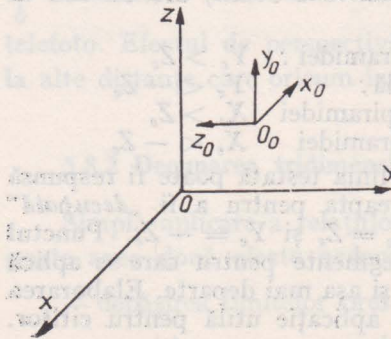
- Translația punctului $(6; 8; 7,5)$ în origine (fig. 5.25)



$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -6 & -8 & -7,5 & 1 \end{bmatrix}$$

Fig. 5.25

- Rearanjarea axelor astfel încât să avem un sistem de coordonate șurub stâng (fig. 5.26)



$$T_2 = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Fig. 5.26

● Rotația în jurul axei Y_0 astfel încât axa Z_0 să fie îndreptată către punctul $(0; 0; 7.5)$, adică o rotație de unghiul θ unde (fig. 5.27)

$$\cos \theta = 8/10$$

$$\sin \theta = 6/10$$

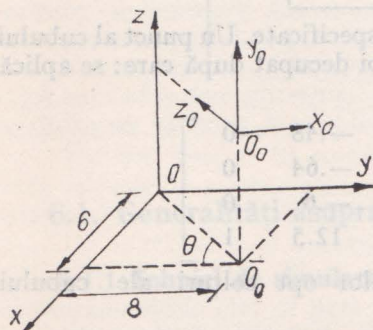


Fig. 5.27

$$T_3 = \begin{bmatrix} .8 & 0 & .6 & 0 \\ 0 & 1 & 0 & 0 \\ -.6 & 0 & .8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

● Rotația în jurul axei X_0 astfel încât axa Z_0 să fie îndreptată către originea spațiului obiect, adică o rotație de unghiul Φ unde (fig. 5.28)

$$\cos \Phi = 4/5$$

$$\sin \Phi = 3/5$$

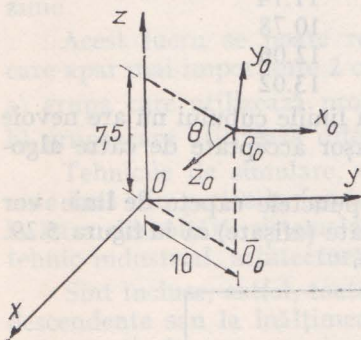


Fig. 5.28

$$T_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & .8 & -.6 & 0 \\ 0 & .6 & .8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Acest fapt completează cele 4 transformări simple necesare pentru transformarea unghiului de vedere $V = T_1 T_2 T_3 T_4$

Să presupunem că dorim să umplem cu perspectiva un ecran de 10/10 inches, cubul fiind văzut de la 10 inches, 25,4 cm distanță; și că sistemul de coordonate al ecranului fuge de la 0 la -1023. Astfel:

$$a = 10 \text{ inch}; \quad b = 5 \text{ inch.}$$

$$V_{sx} = V_{ex} = V_{sy} = V_{cy} = 1023/2$$

Transformarea devine:

$$N = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

iar ecuațiile vor fi

$$\begin{aligned} X_T &= 511.5 (X_0/Z_0) + 511.5 \\ Y_T &= 511.5 (Y_0/Z_0) + 511.5 \end{aligned}$$

Toate detaliile transformărilor au fost deci specificate. Un punct al cubului ($A - H$) este transformat de matricea VN , apoi decupat după care: se aplică ecuația care conduce la

$$T_1 T_2 T_3 T_4 N = \begin{bmatrix} -1.6 & -.72 & -.48 & -0 \\ 1.2 & -.96 & -.64 & 0 \\ 0 & 1.6 & -.6 & 0 \\ 0 & 0 & 12.5 & 1 \end{bmatrix}$$

Aplicînd în continuare transformarea celor opt colțuri ale cubului obținem:

	X_0	Y_0	Z_0
A	2.8	-1.89	12.94
B	-.4	-3.28	11.98
C	-2.8	-1.36	13.26
D	.4	.08	14.22
E	2.8	1.36	11.74
F	-.4	-.08	10.78
G	-2.8	1.84	12.06
H	.4	3.28	13.02

Analizînd acest tabel rezultă că nici una din liniile cubului nu are nevoie să fie „decupată“ adică toate muchiile vor fi ușor acceptate de către algoritmul de decupaj.

Coordonatele perspectivei în ecran pentru punctele capete de linie vor fi calculate cu ecuația dată și liniile vor fi desenate (afișate) ca în figura 5.29. În condițiile date perspectiva este descendentă.

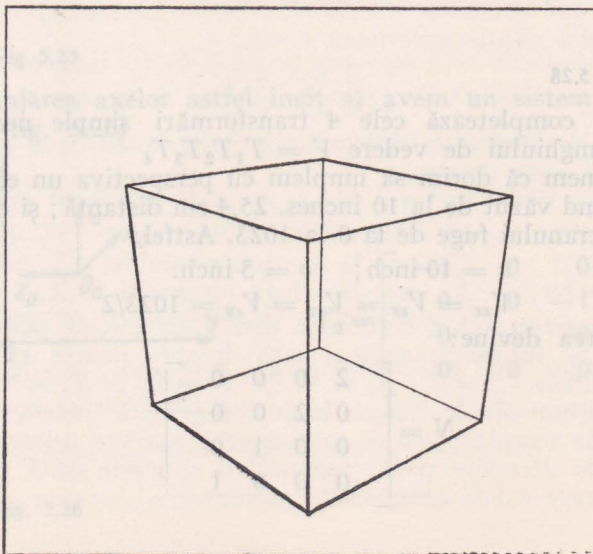


Fig. 5.29

6.1. Generalități asupra reprezentării obiectelor spațiale

6.1.1 Tehnici de simulare

Dispozitivele convenționale de afișaj permit obținerea unor reprezentări plane pentru obiecte, care, în mod normal, aparțin spațiului tridimensional. O astfel de reprezentare a spațiului tridimensional în spațiul bidimensional trebuie să nu conducă la pierderi de informații sau caracteristici ale obiectului de reprezentat (cel puțin din punct de vedere intuitiv). În practică, iluzia celei de a treia dimensiuni este obținută prin crearea unei senzații de profunzime.

Acest lucru se poate rezolva prin diferite tehnici de simulare, dintre care apar mai importante 2 categorii și anume:

- a) grupa care utilizează proiecții perspective particulare,
- b) grupa care utilizează diferite artificii optice.

Tehnicile de simulare, care utilizează proiecții particulare, sînt cunoscute încă din perioada fundamentării noțiunilor de perspectivă, care au stat la baza dezvoltării sistemelor perspective de proiecție cu aplicare în desenul tehnic industrial, arhitectură sau în fotografie.

Sînt incluse, astfel, toate tipurile de reprezentări perspective ascendente, descendente sau la înălțimea normală a orizontului pe tablouri plane înclinate, verticale, precum și perspectivele cavaliere frontale sau orizontale (în special acestea!)

Reprezentările obținute, de cele mai multe ori prin proiectarea tuturor muchiilor care compun corpul, nu sînt suficiente, deoarece induc probleme de interpretare, legate de adîncimea relativă a muchiilor unele față de altele.

De aceea, proiecțiile perspective apar numai ca niște faze în cadrul unor algoritmi mai complecși, care urmăresc rezolvarea nedeterminărilor enunțate anterior.

Acești algoritmi, care vor fi descriși în continuare mai profund, poartă numele, în literatura de specialitate, de algoritmi pentru rezolvarea problemei „liniilor ascunse” sau a „suprafețelor ascunse”.

Tehnicile de lucru pot fi mai mult sau mai puțin sofisticate, în funcție de natura finală a reprezentării dorite, cît și a dispozitivului de afișare utilizat.

În final, se pot obține, spre exemplu, simpla reprezentare a conturilor obiectelor sau, mai complex, porțiunile de suprafață vizibile cărora li se pot aloca culori, texturi, umbriri (în funcție de poziția unor surse luminoase), creînd, în final, imagini extrem de realiste ale scenei.

A doua grupă a tehnicilor de simulare cuprinde, în primul rînd, vederile stereoscopice prezentate pe același ecran, care creează impresia de profunzime prin mijloace relativ simple. În al doilea rînd, pot fi menționate metodele care, cu ajutorul unui joc de oglinzi, pun obiectul în trei dimensiuni, prin formarea unei iluzii optice. Pot fi, de asemenea, amintite metodele care folosesc imagini dinamice, a treia dimensiune fiind creată succesiv cu ajutorul variabilei timp.

6.1.2 Descrierea obiectelor

Una din cele mai importante sarcini, care stau în fața programatorului unei aplicații ce utilizează obiecte spațiale tridimensionale, este alegerea modului de descriere a acestor obiecte. Desigur, există o multitudine de posibilități în scopul obținerii acestei descrieri, dar problema este de a alege o descriere care să satisfacă, într-o măsură cît mai mare, necesitățile, aplicațiile și restricțiile sistemelor de calcul.

În cadrul algoritmilor ce vor fi discutați în continuare se operează cu obiecte alcătuite numai din fețe plane.

Metoda cea mai directă pare a fi descrierea fiecărei fețe poligonale a obiectului, utilizînd coordonatele vîrfurilor și topologia conectării acestor vîrfuri.

Deși datele cerute de diverși algoritmi pot diferi de această formă, conversia la acest tip de descriere este evidentă.

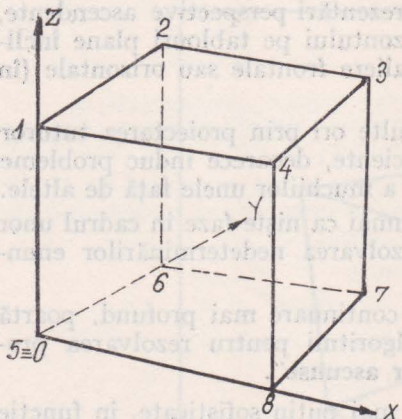
Fiecare vîrf este descris de coordonatele sale în trei dimensiuni alese față de un sistem convenabil de referință (sistem de coordonate obiect).

Fiecare față este descrisă apoi ca un poligon prin testarea vîrfurilor sale (fig. 6.1).

Dacă fața are mai mult de trei vîrfuri, uneori apare și problema planeității suprafeței descrise.

În plus, pentru fiecare față, în funcție de complexitatea aplicației, pot fi asigurate culori, transparențe, reflectante, texturi sau alte proprietăți.

Deoarece, în general, se reprezintă grupuri de obiecte și nu obiecte singulare apare convenabilă construirea de structuri de tip referință. În acest fel, o singură descriere a unui obiect poate fi utilizată de mai multe ori. Fiecare referire a obiectului capătă, însă, carac-



Descrierea unui cub

Fig. 6.1.

teristici particulare de poziție maximă, de orientare și chiar culori sau texturi.

Un mediu structurat de genul prezentat mai sus necesită program în stare să-i interpreteze structura.

Astfel programele care interpretează structura mediului trebuie să poată calcula poziția oricărui vîrf și, astfel, poziția oricărei suprafețe, așa cum apar ele într-o așezare finală.

Mai mult, de obicei mediul este tratat el însuși ca un obiect singular în termenii cunoscuți de „sistem de coordonate obiect“, cînd o exprimare mai exactă ar fi „sistem de coordonate al mediului“.

Trebuie insistat în continuare și spus că generarea mediului și descrierea obiectelor pentru utilizarea, spre exemplu, într-un algoritm de „linii ascunse“ este o sarcină majoră.

De exactitatea și calitatea acestei descrieri depinde, în mod hotărîtor, atît rezultatul final, cît și efortul depus pentru obținerea acestui rezultat (memorie, timp).

Pentru algoritmi urmăriți în continuare mai întii obiectele trebuie approximate prin seturi de fețe plane.

Modul economic, în care se face această operație, care să îmbine o necesitate de minim de fețe cu una de precizie de reprezentare, este o problemă destul de dificilă, care rămîne în sarcina utilizatorului.

Apoi, urmează estimarea coordonatelor spațiale ale obiectelor. Acest proces este lăsat, în multe cazuri, tot în sarcina utilizatorului. Precizia măsurată depinde, evident, de calitatea echipamentelor folosite.

Odată obținute coordonatele, urmează conectarea lor în fețe. În această fază, orice omisiune sau inversiune în ordinea de definire produce deprecierea imaginii și alte efecte laterale care duc la irosire de timp și muncă umană.

O altă latură importantă este dictată de diferențele între diversele proprietăți topologice ale mediului cerute de algoritmi, spre exemplu:

- ce suprafețe se întîlnesc într-o anumită muchie;
- ce fețe fac parte dintr-un anumit corp;
- corpurile care fac parte din mediu să fie convexe.

Apare evident că dificultatea construcției modelului mediului crește odată cu mărirea cantității de informații topologice sau restricții impuse de diversii algoritmi.

Totuși, și acest lucru merită subliniat, algoritmi pot profita enorm de pe urma unor astfel de informații în termenii scurtării timpului de prelucrare.

De exemplu, pentru un mediu cu considerații staționare, în care puține obiecte își schimbă poziția de la un moment la altul, apare drept corespunzător, ca o mare cantitate de calcule preliminare, să fie efectuate înaintea intrării algoritmului propriu zis. Spre exemplu, se pot calcula toate intersecțiile dintre diverse corpuri și modifica structura în mod corespunzător, introducînd liniile de penetrație sau spărgînd corpurile în corpuri neintersectabile. Se pot, de asemenea, face diverse calcule de „prioritate“ pentru fețele unui același corp, care să fie valabile pentru orice poziție a punctului de observare (vedere). Toate aceste informații pot fi stocate și utilizate la reprezentări succesive ale scenei respective.

Dacă, în schimb, mediul este în mișcare, sau se dorește o singură reprezentare, asemenea calcule nu-și mai găsesc rostul.

Ca o concluzie la acest paragraf se impune constatarea că o structurare bine gândită a mediului și o sumă de programe și dispozitive, care să permită măsurători exacte și manipulări cât mai facile ale obiectelor sînt la fel de importante ca algoritmi în sine.

De cele mai multe ori, munca legată de descrierea obiectelor este mult mai laborioasă și mai puțin plăcută decît execuția algoritmilor.

6.2. Problema „suprafețelor ascunse“

6.2.1 O descriere a problemei

În vreme ce producerea unei imagini perspective pentru un obiect transparent alcătuit numai din linii este o problemă relativ simplă, redarea realistă a unui obiect opac este cu mult mai anevoioasă. Obiectul opac este mai greu de reprezentat, deoarece trebuie decisă nu numai poziția fiecărei părți a sa în imagine, dar mai ales ce părți ale sale trebuie să apară în imaginea finală. Unele părți ale obiectului opac sînt acoperite considerînd un punct din care acesta este privit.

Un program care realizează desene pentru obiecte opace trebuie, deci, să fie în stare să decidă ce părți sînt vizibile în vederea aleasă și ce părți sînt ascunse, deci trebuie omise în reprezentare.

Problema acestei decizii a fost inițial cunoscută sub numele de „**Problema liniilor ascunse**“ (**Hidden-line Problem**), cîtă vreme se urmărea eliminarea sau trasarea specială a tuturor liniilor care erau ascunse de alte obiecte din scenă.

Apariția imaginilor cu umbre realizabile pe calculator a făcut ca o variantă a acestei probleme să devină importantă: „**Problema suprafețelor ascunse**“ (**Hidden — Surface Problem**).

Într-o imagine cu umbre trebuie incluse sau omise părți întregi cu suprafețe și nu numai liniile lor de contur.

Cele două probleme au însă numeroase puncte comune, astfel încît o tratare unitară apare drept judicioasă.

Imaginile umbrite sînt produse prin înregistrarea nuanței de gris sau de culoare în fiecare punct acoperit de perspectiva scenei și dependent de dimensiunile ecranului de vizualizare.

Generarea imaginilor cu umbre este, însă, o problemă costisitoare, atît din punct de vedere al memoriei, cît și al timpului de execuție. Chiar și în cazul unor dispozitive cu suficientă memorie de reîmprospătare, generarea unor asemenea imagini în timp real este practic imposibilă în lipsa unui hardware specializat.

O definiție formală pentru un algoritm de suprafețe ascunse poate fi următoarea:

$$\text{ASA (algoritm suprafețe ascunse)} = (\mathbf{O}, \mathbf{S}, \mathbf{I}, \varphi, \sigma)$$

în care:

- O este un set de obiecte în spațiul $3D$
 S este un set de segmente vizibile în $2D$
 I este un set de „reprezentări intermediare“
 φ este un set de „funcții de tranziții“ $\{TP, IS, TI, TA, TV\}$
 σ este o „strategie“

Numele funcțiilor din φ au următoarele sensuri:

- TP este o funcție care produce transformarea perspectivă

$$TP : 3D \rightarrow 2D$$

- IS este o funcție ce calculează punctul de intersecție a 2 segmente. Aici domeniul și codomeniul sînt identice ($2D$ sau $3D$)
- TI este o funcție care reavizează în $2D$ un test de incluziune, adică verifică dacă un punct este sau nu în interiorul unei suprafețe;
- TA este o funcție care realizează un test de adîncime, adică compară adîncimile a 2 puncte față de punctul de vedere.
- VT este o funcție care realizează testul de vizibilitate pentru o suprafață dată, adică testează dacă suprafața este potențial vizibilă sau total invizibilă.

Strategia σ specifică ordinea în care funcțiile cuprinse în φ trebuie aplicate pentru obținerea rezultatului dorit.

În general diversitatea algoritmilor este dată de strategia de abordare a problemei care induce la rîndul ei folosirea unora sau altora dintre funcțiile incluse în φ .

6.2.2 Calcule geometrice

Acest paragraf urmărește să pună la îndemina cititorului cîteva din calculele de rutină, care apar în majoritatea algoritmilor de **suprafețe ascunse** și care, de cele mai multe ori, sînt trecute sub tăcere de autori. Lucrul acesta apare drept foarte folositor deoarece, în ultimă instanță, de viteza și tehnica cu care se execută din punct de vedere procedural aceste calcule depinde eficiența diverselor implementări.

Metodele descrise în continuare sînt rezolvări pentru problemele puse de funcțiile cuprinse în φ (vezi 6.2.1).

a) Teste Minimax

Aceste teste sînt foarte utile în diverse momente ale algoritmilor

- $a1$ — calculul suprapunerii a 2 poligoane în planul de proiecție;
 $a2$ — calculul interiorității unui punct față de un poligon în planul de proiecție;
 $a3$ — calculul intersecție a 2 segmente;
 $a4$ — calcule de adîncime a unui punct față de un poligon sau chiar a 2 poligoane între ele în spațiul obiect.

Calculul minimax este o unealtă deosebit de simplă și de rapidă, implicînd numai comparații. Ele pot duce la rejecția a numeroase cazuri nefavorabile, fiind utilizat înaintea calculelor propriu-zis.

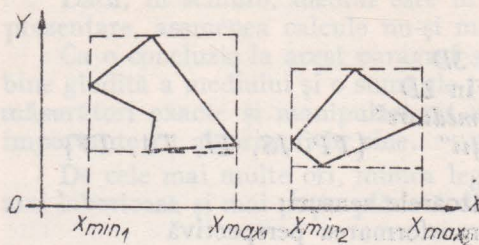


Fig. 6.2.

De exemplu, cazul a 2 poligoane în planul de proiecție. Cele 2 poligoane nu au sigur nici un punct de intersecție dacă $x_{min2} > x_{max1}$ (Fig. 6.2.)

Acest tip de calcul prezent în figura 6.2 se poate extrapola cu ușurință spre exemplu și la coordonata Y sau Z, rezolvînd foarte

simplu numeroase cazuri pentru care, altfel, ar fi fost necesare calcule laborioase.

b) Calcule de intersecții

Aceste calcule apar în numeroase momente al algoritmilor și cu diverse scopuri:

- b1 — găsirea unui punct pe un segment de dreaptă unde apare o schimbare de vizibilitate;
- b2 — aflarea intersecției între 2 poligoane în planul de proiecție;
- b3 — intersecția dintre diverse segmente și linia de explorare
- b4 — ca o funcție în cadrul testului de interioritate
- b5 — determinarea intersecției între fețele obiectului și o linie specială, obținută prin legarea punctului de vedere cu un punct de test.

Iată, în continuare, câteva din calculele necesare pentru diverse intersecții:

Intersecția dintre 2 drepte

Fie ecuațiile a 2 drepte în coordonate normale în 2D

$$A_1x + B_1y + C_1 = 0$$

$$A_2x + B_2y + C_2 = 0$$

Dacă

$$\begin{vmatrix} A_1 & B_1 \\ A_2 & B_2 \end{vmatrix} = 0$$

cele 2 drepte sînt paralele adică nu există punct finit de intersecție. Dacă $A_1/A_2 = B_1/B_2 = C_1/C_2$, cele 2 drepte sînt confundate. Astfel dacă punctul de intersecție există el este dat de relațiile

$$x_1 = \frac{B_1C_2 - B_2C_1}{A_1B_2 - A_2B_1}$$

$$y_1 = \frac{C_1A_2 - C_2A_1}{A_1B_2 - A_2B_1}$$

unde $A_1B_2 \neq A_2B_1$.

Intersecția dintre 2 segmente

Fie 2 drepte în spațiu G și G' . Fie $P_1, P_2 \in G$, $P_1 = (X_1, Y_1)$ și $P_2 = (x_2, y_2)$ și $P'_1, P'_2 \in G'$, $P'_1 = (x'_1, y'_1)$, $P'_2 = (x'_2, y'_2)$

Cele două segmente P_1P_2 și $P'_1P'_2$ se intersectează în punctul $I = (x_I, y_I)$ dacă:

$$\min[\min(x_1, x_2), \min(x'_1, x'_2)] \leq x_I \leq \max[\max(x_1, x_2), \max(x'_1, x'_2)]$$

și

$$\min[\min(y_1, y_2), \min(y'_1, y'_2)] \leq y_I \leq \max[\max(y_1, y_2), \max(y'_1, y'_2)]$$

Este de remarcat faptul că este util să se execute înaintea calculului pentru (x_I, y_I) testul minimax care poate elimina numeroase calcule.

Intersecția dintre un plan și o dreaptă

Fie un plan trecînd prin punctele $P_i = (x_i, y_i, z_i)$,

$$P_j = (x_j, y_j, z_j); P_k = (x_k, y_k, z_k)$$

avînd ecuația

$$Ax + By + Cz + D = 0$$

Fie o dreaptă trecînd prin punctele $P_1 = (x_1, y_1, z_1)$ și $P_2 = (x_2, y_2, z_2)$ care poate fi reprezentată prin ecuațiile:

$$\frac{x - x_1}{\cos \alpha} = \frac{y - y_1}{\cos \beta} = \frac{z - z_1}{\cos \gamma}$$

unde

$$\cos \alpha = \frac{x_2 - x_1}{d}, \quad \cos \beta = \frac{y_2 - y_1}{d}, \quad \cos \gamma = \frac{z_2 - z_1}{d}$$

$$\text{cu } d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

Dacă:

- 1) $A \cos \alpha + B \cos \beta + C \cos \gamma = 0$ dreapta este paralelă cu planul
- 2) $A \cos \alpha + B \cos \beta + C \cos \gamma = 0 \wedge A x_1 + B y_1 + C z_1 + D = 0$ dreapta este conținută în plan.

Astfel, există punctul de intersecție $I = (x_i, y_i, z_i)$ și este dat de relațiile:

$$\begin{cases} x_i = x_1 - t \cos \alpha \\ y_i = y_1 - t \cos \beta \\ z_i = z_1 - t \cos \gamma \end{cases}$$

$$t = \frac{A x_1 + B y_1 + C z_1 + D}{A \cos \alpha + B \cos \beta + C \cos \gamma}$$

Suprapunerea a 2 poligoane în planul de proiecție

Date fiind două poligoane închise în planul de proiecție (care pot fi proiecțiile a 2 fețe), se pune problema existenței unei suprapunerii între cele 2 contururi. Suprapunerea poate fi detectată fie gășind un punct de intersecție între cele 2 contururi caz în care calculul se încheie, sau verificând incluziunea fiecărui vîrf al unui poligon față de celălalt poligon. Figura 6.3.a prezintă cazurile posibile ale poziției relative dintre cele 2 poligoane, iar figura 6.3.b ilustrează cazurile posibile de acoperire a unei muchii de aria unui triunghi.

Este cu mult mai economic să se înceapă testul prin calculul intersecțiilor și, în caz de insucces, să se treacă la testul de incluziune, deoarece:

- probabilitatea intersecției este mai mare decît cea a conținerii,
- relația de incidență este simetrică în vreme ce relația de conținere este antisimetrică. De aceea, ambele poligoane trebuie testate pentru conțineri, pe cînd numai unul trebuie testat pentru intersecție;
- relația de incluziune implică calculele mult mai laborioase (include calcule cu intersecție).

Din nou se poate remarca utilizarea unui test minimax înaintea oricărui calcul. Numai în cazul cînd testul minimax eșuează, se poate trece la calculul intersecțiilor.

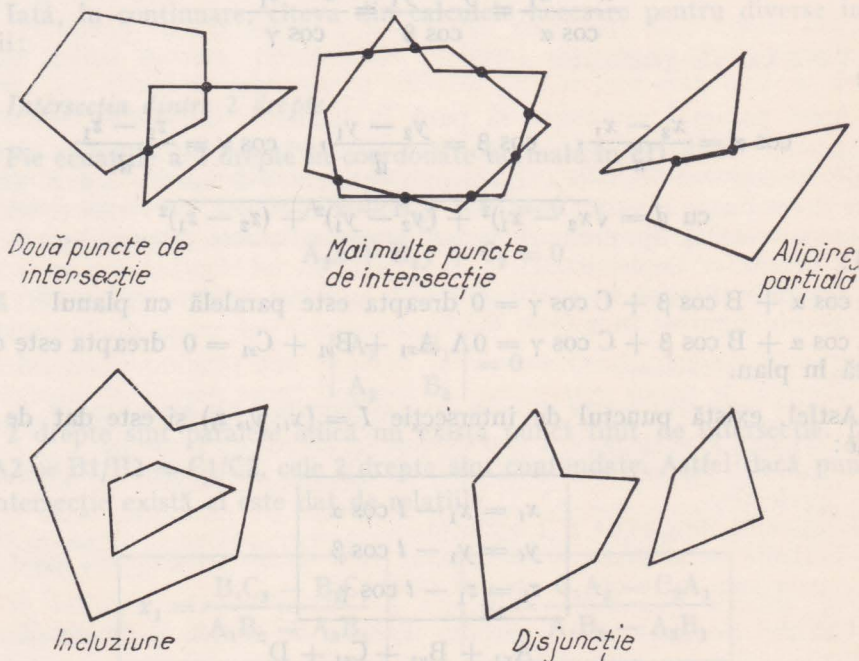
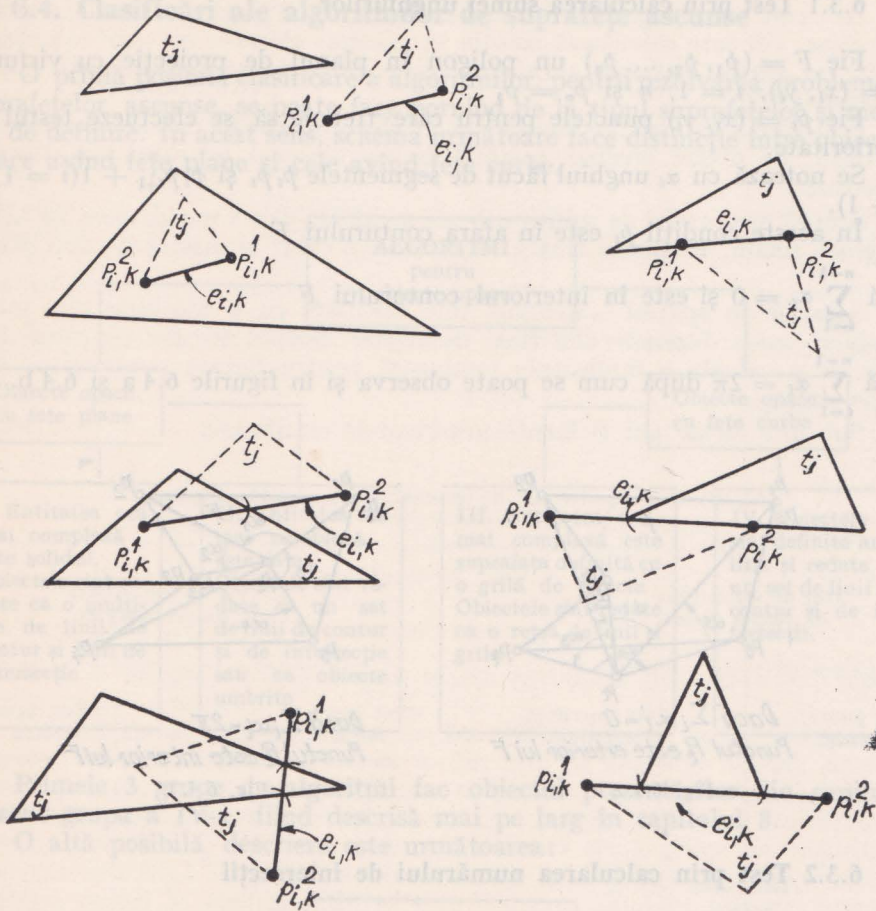


Fig. 6.3. a



Ilustrarea cazurilor posibile de acoperire a unei muchii

$e_{i,k}$ e t_j de triunghiul t_j

Punctele $P_{i,k}^1$ și $P_{i,k}^2$ sînt capetele
segmentului $e_{i,k}$

Fig. 6.3. b

6.3. Teste de interioritate

Una din problemele care apare curent în cadrul algoritmilor de suprafețe este verificarea interiorității unui punct față de un contur în planul de proiecție.

În geometria elementară există 2 proceduri bine cunoscute pentru realizarea acestui test.

6.3.1 Test prin calcularea sumei unghiurilor

Fie $F = (p_1, p_2, \dots, p_n)$ un poligon în planul de proiecție cu virfurile $p_i = (x_i, y_i)$, $i = 1, n$ și $p_n = p_1$

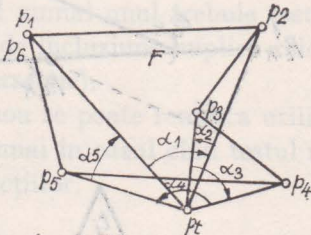
Fie $p_i = (x_i, y_i)$ punctele pentru care trebuie să se efectueze testul de interioritate.

Se notează cu α_i unghiul făcut de segmentele $p_i p_i$ și $p_i p_{i+1} + 1 (i = 1, \dots, n - 1)$.

În aceste condiții p_i este în afara conturului F

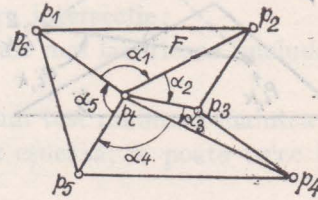
dacă $\sum_{i=1}^{n-1} \alpha_i = 0$ și este în interiorul conturului F

dacă $\sum_{i=1}^{n-1} \alpha_i = 2\pi$ după cum se poate observa și în figurile 6.4 a și 6.4 b.



*Dacă $\sum_i \alpha_i = 0$
Punctul P_i este exterior lui F*

Fig. 6.4. a



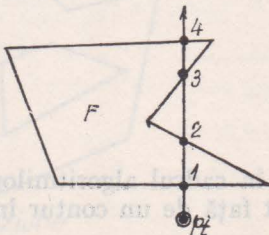
*Dacă $\sum_i \alpha_i = 2\pi$
Punctul P_i este interior lui F*

Fig. 6.4. b

6.3.2 Test prin calcularea numărului de intersecții

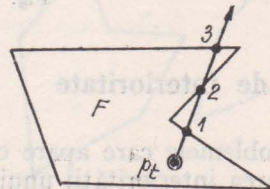
Condițiile problemei sînt aceleași cu cele prezentate anterior. Se consideră o semidreaptă, care pornește din punctul P_i și nu trece prin nici unul din virfurile poligonului F .

Atunci P_i este exterior lui F , dacă numărul de intersecții ale lui R cu laturile lui F este par, și este interior lui F dacă acest număr este impar (fig. 6.5 a și b).



Număr par = P_i exterior lui F

Fig. 6.5 a

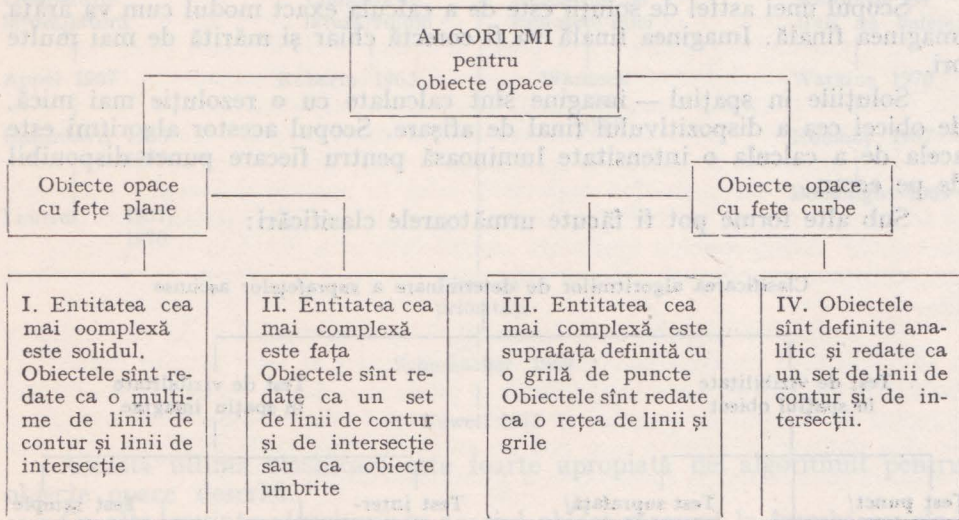


Număr impar = P_i interior lui F

Fig. 6.5. b

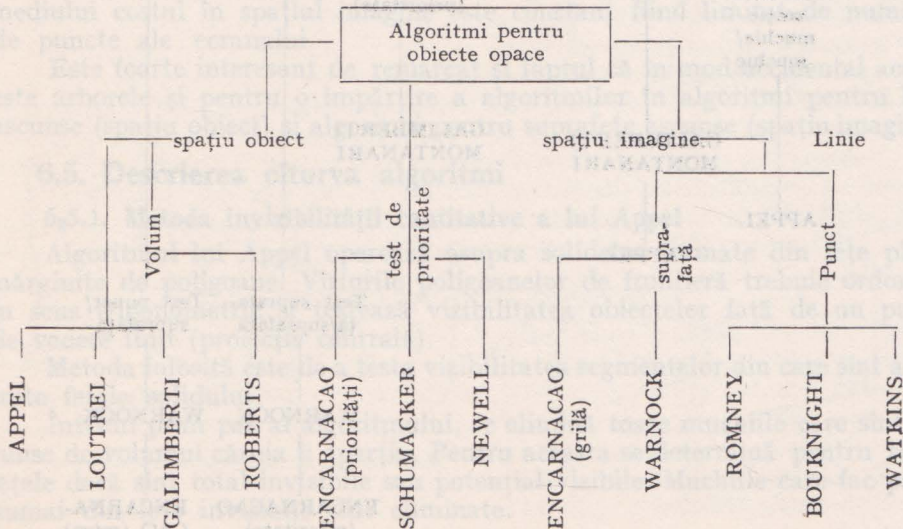
6.4. Clasificări ale algoritmilor de suprafețe ascunde

O primă posibilă clasificare a algoritmilor, pentru rezolvarea problemelor suprafețelor ascunde, se poate face pornind de la tipul suprafețelor și modul lor de definire. În acest sens, schema următoare face distincție între obiectele opace având fețe plane și cele având fețe curbe.



Primele 3 grupe de algoritmi fac obiectul prezentărilor din capitolul prezent grupa a IV-a, fiind descrisă mai pe larg în capitolul 8.

O altă posibilă descriere este următoarea:



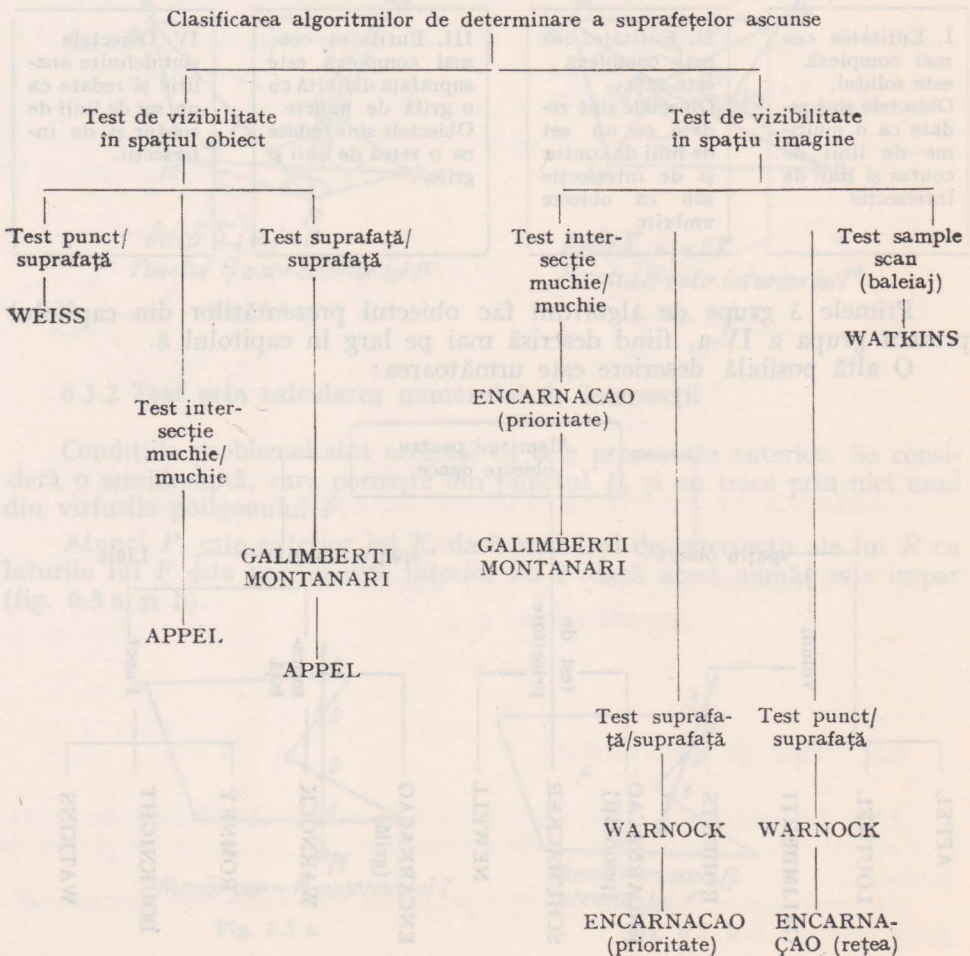
În cazul acestei descrieri nodul rădăcină al arborelui divide algoritmi în 3 clase: cei care calculează soluția în „spațiul obiect”; cei care calculează soluția în „spațiul imagine” și cei care lucrează parțial în ambele spații („lista de priorități”).

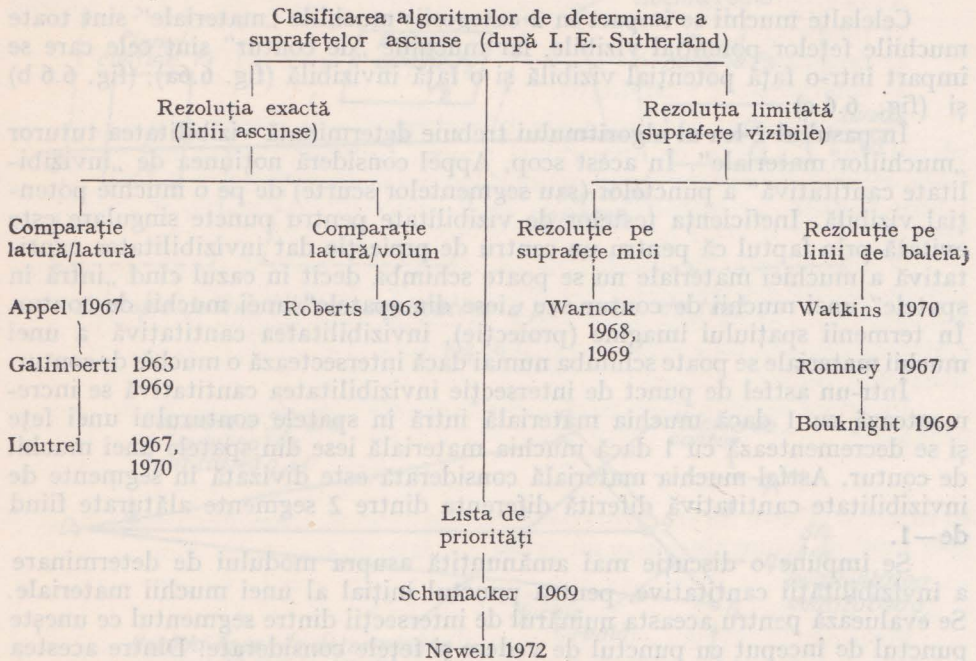
Prin calcul în „spațiul obiect” se înțelege un calcul realizat cu precizie arbitrară, de obicei precizia posibilă în calculatorul pe care, se execută algoritmul.

Scopul unei astfel de soluții este de a calcula exact modul cum va arăta imaginea finală. Imaginea finală va fi corectă chiar și mărită de mai multe ori.

Soluțiile în spațiul — imagine sînt calculate cu o rezoluție mai mică, de obicei cea a dispozitivului final de afișare. Scopul acestor algoritmi este acela de a calcula o intensitate luminoasă pentru fiecare punct disponibil de pe ecran.

Sub alte forme pot fi făcute următoarele clasificări:





Această ultimă clasificare este foarte apropiată de algoritmul pentru obiecte opace descriși.

Cu alte cuvinte algoritmi în spațiul obiect răspund la întrebarea cînd fiecare element potențial vizibil în mediu este cu adevărat vizibil, pe cînd algoritmi în spațiul imagine arată cea ce este vizibil într-un punct al ecranului.

Iată și unul dintre motivele pentru care apare o diferență corespunzătoare a costurilor și performanțelor între aceste două grupe de algoritmi; în vreme ce costul în spațiul — obiect crește ca o funcție de complexitatea mediului costul în spațiul imagine este constant fiind limitat de numărul de puncte ale ecranului.

Este foarte interesant de remarcat și faptul că în mod accidental acesta este arborele și pentru o împărțire a algoritmilor în algoritmi pentru linii ascunse (spațiu obiect) și algoritmi pentru suprafețe ascunse (spațiu imagine).

6.5. Descrierea citorva algoritmi

6.5.1. Metoda invizibilității cantitative a lui Appel

Algoritmul lui Appel operează asupra solidelor formate din fețe plane mărginite de poligoane. Virfurile poligoanelor de frontieră trebuie ordonate în sens trigonometric și testează vizibilitatea obiectelor față de un punct de vedere finit (proiecție centrală).

Metoda folosită este de a testa vizibilitatea segmentelor din care sînt alcătuite fețele solidului.

Într-un prim pas al algoritmului, se elimină toate muchiile care sînt ascunse de volumul căruia îi aparțin. Pentru aceasta se determină pentru toate fețele dacă sînt total invizibile sau potențial vizibile. Muchiile care fac parte numai din fețe invizibile sînt eliminate.

Celelalte muchii se împart în 2 categorii: muchiile „materiale“ sînt toate muchiile fețelor potențial vizibile, iar muchiile „de contur“ sînt cele care se împart într-o față potențial vizibilă și o față invizibilă (fig. 6.6a), (fig. 6.6 b) și (fig. 6.6 c).

În pasul al 2-lea al algoritmului trebuie determinată vizibilitatea tuturor „muchii materiale“. În acest scop, Appel consideră noțiunea de „invizibilitate cantitativă“ a punctelor (sau segmentelor scurte) de pe o muchie potențial vizibilă. Ineficiența testelor de vizibilitate pentru puncte singulare este evitată prin faptul că pentru un centru de proiecție dat invizibilitatea cantitativă a muchiei materiale nu se poate schimba decît în cazul cînd „intră în spatele“ unei muchii de contur sau „iese din spatele“ unei muchii de contur. În termenii spațiului imagine (proiecție), invizibilitatea cantitativă a unei muchii materiale se poate schimba numai dacă intersecțiază o muchie de contur.

Într-un astfel de punct de intersecție invizibilitatea cantitativă se incrementează cu 1 dacă muchia materială intră în spatele conturului unei fețe și se decrementează cu 1 dacă muchia materială iese din spatele unei muchii de contur. Astfel muchia materială considerată este divizată în segmente de invizibilitate cantitativă diferită diferența dintre 2 segmente alăturate fiind de -1.

Se impune o discuție mai amănunțită asupra modului de determinare a invizibilității cantitative, pentru punctul inițial al unei muchii materiale. Se evaluează pentru aceasta numărul de intersecții dintre segmentul ce unește punctul de început cu punctul de vedere și fețele considerate. Dintre acestea se aleg doar acele puncte care cad pe segment. Numărul astfel calculat este valoarea invizibilității cantitative a punctului inițial (fig. 6.7).

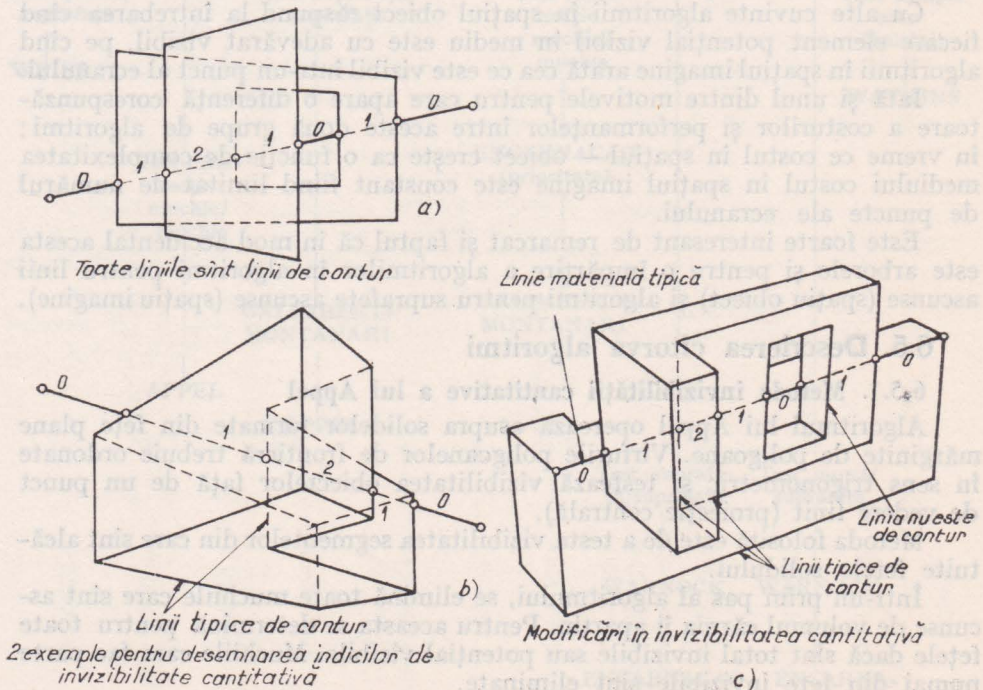


Fig 6.6. a, b și c

Algoritmul propriu-zis al suprafeței ascunse constă din 3 faze majore:

- 1) presortarea
- 2) desemnarea priorității
- 3) determinarea vizibilității.

(1) *Presortarea*: Elimină toate triunghiurile care sînt perpendiculare pe planul xoy din lista de triunghiuri supuse testului ulterior de prioritate, deoarece o astfel de față perpendiculară nu poate acoperi altă față. O față este perpendiculară pe planul imagine dacă proiecțiile virfurilor lor sînt colineare.

(2) *Desemnarea priorității*: Se consideră un obiect în spațiu descompus într-un set de fețe triunghiulare $S = S_1, \dots, S_n$. Presupunem că fețele triunghiulare din spațiul obiectului, cit și proiecțiile lor corespunzătoare din spațiul imaginii, sînt numerotate pentru identificare și specificate de tipurile coordonatelor celor 3 virfuri.

În prima etapă a metodei de prioritate este format un set de M perechi ordonate:

$$(t_i, T_i); \quad i = 1, \dots, M,$$

în care $t_i \in T$ este un triunghi (proiecția unei fețe S_i triunghiulare) și $T_i \subseteq T$ este setul de triunghiuri care au prioritate față de t_i

$$T_i^* = \{t_j \in T; \quad t_i \leq t_j\}$$

Perechile (t_i, T_i) , $i = 1, \dots, M$ reprezintă ordinea flexibilă în care t_i este primul component din toate perechile și elementele lui T_i care sînt componente secundare. Acestea sînt obținute prin aplicarea testului de adîncime. Primul test minimal se aplică pentru a determina care din obiectele t_i au o șansă de intersecție cu celelalte obiecte t_j . Pentru obiectele rămase $L \leq M$, trebuie stabilită prioritatea, solicitînd repetițiile $L(L-1)/2$ ale testului de adîncime.

Punctele test din intersecția a 2 triunghiuri necesare pentru determinarea priorității sînt obținute prin încercarea de a spori intersecțiile dintre muchiile celor 2 triunghiuri și dacă nu reușește, testarea pentru conținut. Suprapunerea ciclică nu poate apare din cauza triangulației. Totuși, fețele triunghiulare se pot împătrunde. Penetrația poate apare numai în cazul cînd proiecțiile celor 2 fețe au o intersecție plină. Apariția ei poate fi stabilită prin executarea testului de prioritate pentru toate punctele de intersecție între muchiile celor 2 proiecții. Dacă testul de prioritate produce același rezultat pentru toate aceste puncte, nu există prioritate.

(3) *Determinarea vizibilității*: În fiecare pereche (t_i, T_i) T_i este setul de triunghiuri care acoperă, într-o anumită măsură, triunghiul t_i . Dacă T_i este vid, atunci t_i este vizibil în întregime (cu excepția muchiilor auxiliare). Pentru un set nevid T_i , trebuie să se determine gradul de acoperire a lui t_i pe triunghiurile din T_i . Aceasta necesită determinarea vizibilității muchiilor lui t_i care nu sînt muchii auxiliare. De aceea, este suficient să se descrie cum poate fi investigată vizibilitatea unei muchii.

Fie $e_{i,k} < t_j = 1.2.3$ o muchie a lui t_i . Vizibilitatea lui $e_{i,k}$ cu privire la un triunghi $t_j > t_i$ poate fi obținută prin stabilirea punctelor de intersecție pe care $e_{i,k}$ poate să le aibă cu t_j și testînd dacă punctele de capăt ale lui $e_{i,k}$ se află în interiorul sau exteriorul lui t_j .

Un segment al lui $e_{i,k}$ care intersecțiază t_j este invizibil; un segment al lui $e_{i,k}$ care nu se intersecțiază cu t_j este vizibil. Ori de cîte ori întregul triun-

ghi t_i este recunoscut ca fiind total acoperit, prelucrarea perechii (t_i, T_i) poate fi terminată. Rezultatul testelor de vizibilitate este constituit nu de triunghiuri ci de muchiile individuale ale triunghiurilor. (vezi și figura 6.3b)

6.5.3. Algoritmul lui WARNOCK

Pentru a reprezenta un obiect în spațiu, Warnock a ales pentru lucru coordonatele din spațiul imagine. Punctul de vedere este plasat la infinit spre Z -uri negative, iar proiecția razelor care trec prin toate punctele obiectului va da semnul dorit. Acest sistem de coordonate permite un calcul destul de simplu pentru a determina dacă un punct P este ascuns sau nu de un poligon: Se vede, într-un prim timp, dacă P este în interiorul poligonului (în proiecție pe xOy), apoi se calculează dacă profunzimea lui P este mai mare sau nu față de cea a intersecției razei care trece prin P cu planul xOy .

Warnock consideră porțiuni de ecran pe care le numește *ferestre* și caută să vadă dacă poate stabili una din cele trei situații următoare:

- nu este nimic de văzut;
- ce se vede este ușor de desenat;
- ce se vede este prea complex.

În primele două cazuri, se consideră că examinarea ferestrei a reușit și se afișeze conținutul său. În ultimul caz, este un eșec și se divide fereastra în patru părți care vor fi examinate la rîndul lor. Acest algoritm va putea fi scris în manieră recursivă iar testul de oprire este interesant, deoarece este bazat pe limita de rezoluție a perifericului. Întrădevăr, dacă la un moment dat conținutul unei ferestre este foarte complex, dar talia (mărimea acestor ferestre este egală cu limita taliei (mărimii) unui punct pe terminal, vom decide să nu mergem mai departe în studii și se va afișa un punct.

Diferitele părți care compun algoritmul lui Warnock sînt următoarele:

- procedura de decupare și de exploatare a ferestrelor;
- procedura care verifică relațiile dintre proiecțiile feței și fereastra intrată în parametrii (*LOOKER*, denumirea de origine);
- procedura care decide complexitatea conținutului unei ferestre pornind de la informațiile stabilite prin procedura precedentă (*THINKER*);
- procedura de afișaj capabilă să descopere ceea ce trebuie desenat în funcție de fereastră și de informațiile calculate prin celelalte două proceduri.

6.5.4. Algoritmul lui WATKINS sau algoritmul liniei de explorare operează în spațiul imaginii pe baza unui raster cu linii de explorare a unei mulțimi de poligoane care sînt proiecțiile fețelor plane ale obiectelor tridimensionale.

Acest algoritm este destinat să producă imagini în semitente, adică să caute suprafețele vizibile ale unui obiect. Perifericul de ieșire este un ecran cu baleiaj de televiziune, ceea ce duce la analiza imaginii linie cu linie de sus în jos și de la stînga la dreapta. Calculele se vor face în coordonate ecran, planul ecranului fiind considerat ca un plan xOy , observatorul fiind la infinit spre Z -urile negative. Se taie, atunci, scena prin planuri paralele cu planul xOy , numite *planuri de baleiaj*. Se calculează intersecția fiecărui plan de baleiaj cu poligoanele definind obiectul de reprezentat, ceea ce dă,

într-un plan de baleiaj, un ansamblu de segmente ce vor fi examinate pentru a ști ceea ce trebuie afișat. Sîntem astfel în fața unei probleme cu două dimensiuni; deoarece vom privi pur și simplu, în planul de baleiaj, care sînt segmentele care le ascund pe celelalte și care sînt deci, vizibile. Testele vor fi relativ simple, deoarece nu vom avea de comparat decît segmentele între ele, din punctul de vedere al poziției lor față de observator. Să notăm, în trecere, asemănarea cu algoritmul lui Warnock: — împărțirea problemei în subprobleme mai simple (linii de baleiaj și ferestre; — folosirea coerenței pentru a reduce numărul de calcule (istoria liniilor de baleiaj și istoria ferestrelor).

Structura este analogă algoritmului lui Warnock în ceea ce privește părțile principale ale programului:

- O procedură care controlează progresele planului de baleiaj.
- O procedură de calcul a relațiilor diferitelor segmente ale unui plan de baleiaj numită de asemenea *LOCKER*.
- O procedură de decizie a complexității unei situații date (*THINKER*)
- O procedură de afișaj, generînd de data aceasta suprafețele și nu puncte sau linii.

Acest algoritm este așa dar potrivit pentru o redare a tablourilor umbrite în tente de gri. În acest caz, segmentele vizibile nu sînt redade cu o strălucire uniformă, ci cu tentă de gri care derivă din valoarea z a feței corespunzătoare.

6.5.5. Metoda rețelei de explorare a lui ENCARNACAO

Acest algoritm poate fi aplicat la suprafețele curbe care sînt definite ca o rețea de linii.

Fragmentele (carourile) suprafețelor curbe sînt specificate numai prin cele 4 vîrfuri ale lor, iar liniile drepte sînt folosite pentru a defini muchiile lor. Suprafața curbă a fragmentului (caroului) este aproximată prin patru plane formate de diagonalele intercalate între vîrfurile opuse ale fragmentului (descompunerea în triunghiuri).

Numele algoritmului derivă din faptul că un raster rectangular bidimensional de linii numit „rețea de explorare“ este suprapus pe proiecția planului imagine a suprafețelor. O listă de fragmente (elemente) de suprafață este construită pentru fiecare suprafață a rețelei de explorare. Această listă dezvăluie care fragmente suprapun fiecare din zonele separate ale rețelei de explorare. Această prezentare de fragmente în zone de rețele de explorare permite stabilirea vizibilității unui punct de test pentru a fi determinate numai acele fragmente de suprafețe care se află în aceeași zonă a rețelei de explorare ca punct de test.

Pentru o imagine tipică cu fragmente 100—400 s-a constatat că o rețea de explorare de 11×11 ar fi de dimensiune rezonabilă. Testul minimax este folosit pentru a stabili care suprafețe de explorare vor fi suprapuse de un fragment specific de suprafață. Imediat ce s-a terminat preselecția, se execută algoritmul principal de vizibilitate. Toate liniile u și v sînt testate pentru vizibilitate în felul următor: Un segment de linie u sau v este rupt într-un șir de puncte de test, spațiate în mod egal între vîrfurile fragmentelor de suprafață. Fiecare punct este testat pentru vizibilitate față de fragmentele de suprafață care se află în interiorul aceleiași zone de explorare cu punctul de testare.

Ori nici un fragment din suprafață nu acoperă punctul de testare, în care caz este vizibil, ori un fragment acoperă punctul de testare, în care caz este invizibil. Dacă toate punctele care se află de-a lungul muchiei fragmentului sînt vizibile întreaga muchie se trasează. Dacă se constată că unele puncte sînt invizibile, sînt trasate numai acele secțiuni ale muchiei fragmentului care se află între punctele vizibile de test.

Pe scurt algoritmul poate fi schițat în felul următor:

- (1) Se face proiecția pe planul imaginii (planul xy);
- (2) Se face determinarea componentelor minimale și maximale x și y ale suprafețelor de reprezentat;
- (3) Din aceste valori x , y minimale și maximale rezultă mărimea rasterului (rețelei de explorare);
- (4) Un raster ($n \times n$) în care n este liber ales, se aplică pe suprafețele proiectate, prin aceasta fiind așezat în raster domeniul de existență al tuturor suprafețelor.
- (5) Vizibilitatea tuturor punctelor nodale u , v , se determină și anume în așa fel încît, cînd un punct nodal se testează cu privire la vizibilitate, se testează numai caroul de raster aferent și nu se mai testează toată suprafața imaginii.

Pentru toate carourile de raster care conțin un punct nodal, se execută următorul proces:

- a) Dacă există în caroul de raster considerat elemente de suprafață, se examinează să se vadă dacă numărul elementelor de suprafață existente este mai mic sau egal cu numărul de elemente de suprafață.
- b) Dacă în caroul de raster se află prea multe elemente de suprafață, astfel încît o nouă examinare ar fi prea cheluitoare de timp, se descompune caroul de raster în 4 subcarouri de raster și se merge cu pasul a) înainte dacă nu, cu c);
- c) Se consemnează apartenența elementelor de suprafață la caroul de raster.
- d) Elementele de suprafață în caroul de raster se descompun în triunghiuri.
- e) Triunghiurile care conțin punctul nodal se examinează mai îndeaproape fiind tratate ca plane: coordonatele x , y ale punctului nodal se așază în ecuațiile de plan ale triunghiurilor putînd determina astfel coordonatele z ale planurilor triunghiurilor.

f) Coordonata z a punctului nodal se compară cu coordonata z calculată a planurilor; punctul nodal se declară în cazul acesta ca invizibil, cînd o coordonată z a ecuațiilor planurilor existentă, pentru care este valabil faptul că este mai mare sau egală cu coordonata z a punctului nodal. În caz contrar, punctul nodal este vizibil.

(6) După ce s-a stabilit vizibilitatea tuturor punctelor nodale, se examinează vizibilitatea liniilor de legătură ale punctelor nodale. În scopul acesta se ia pentru fiecare linie de legătură un test de puncte, distanța punctelor de test rezultînd ca funcțiune a distanței punctelor nodale. Cu fiecare punct de test se procedează analog procesului pentru determinarea vizibilității punctelor nodale u și v . Se parcurg pașii 5(a) și (b). În locul punctului nodal u , v de testat, se consideră acum punctul de testat de-a lungul unei linii de legătură a două puncte nodale u , v .

ANEXA A

Formule fundamentale necesare înțelegerii reprezentărilor geometrice în grafica pe calculator.

A.1. Coordonate rectangulare în plan

A.1.1. Distanța dintre două puncte $P_1(X_1, Y_1)$ și $P_2(X_2, Y_2)$ este:

$$d = \overline{P_1P_2} = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$$

A.1.2. Panta unei drepte AB unde $A(X_1, Y_1)$ și $B(X_2, Y_2)$ este:

$$\operatorname{tg} \theta = m = \frac{Y_2 - Y_1}{X_2 - X_1} = \frac{m_2}{m_1}$$

Există relațiile

$$\alpha_1 = \pm \frac{m_1}{\sqrt{m_1^2 + m_2^2}} ; \quad \alpha_2 = \pm \frac{m_2}{\sqrt{m_1^2 + m_2^2}}$$

$$\alpha_1^2 + \alpha_2^2 = 1$$

A.1.3. Suprafața triunghiului ABC , unde $A(X_1, Y_1)$, $B(X_2, Y_2)$, $C(X_3, Y_3)$ este:

$$S = \frac{1}{2} \begin{vmatrix} 1 & 1 & 1 \\ X_1 & X_2 & X_3 \\ Y_1 & Y_2 & Y_3 \end{vmatrix}$$

A.1.4. Condiția de colinearitate a trei puncte $A(x_1, y_1)$, $B(x_2, y_2)$, $C(x_3, y_3)$ este:

$$\begin{vmatrix} 1 & 1 & 1 \\ X_1 & X_2 & X_3 \\ Y_1 & Y_2 & Y_3 \end{vmatrix} = 0$$

A.2. Coordonate oblice în plan

Unghiul dintre direcția pozitivă a axei X cu direcția pozitivă a axei Y nu mai este 90° , ci un unghi oarecare ω .

A.2.1. Distanța dintre două puncte $P_1(x_1, y_1)$, $P_2(x_2, y_2)$:

$$\overline{P_1P_2} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + 2(x_2 - x_1)(y_2 - y_1) \cos \omega}$$

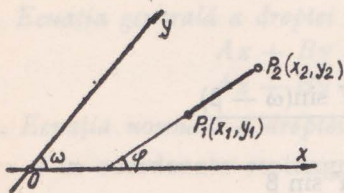


Fig. 1 - A

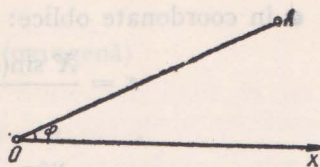


Fig. 2 - A

A.2.2. Panta unei drepte P_1P_2 este (fig. 1.-A)

$$\operatorname{tg} \varphi = \frac{(y_2 - y_1) \sin \omega}{(x_2 - x_1) + (y_2 - y_1) \cos \omega}$$

A.2.3. Suprafața triunghiului ABC , unde $A(x_1, y_1)$, $B(x_2, y_2)$ și $C(x_3, y_3)$ este :

$$S = \frac{\sin \omega}{2} \begin{vmatrix} 1 & 1 & 1 \\ X_1 & X_2 & X_3 \\ Y_1 & Y_2 & Y_3 \end{vmatrix}$$

A.2.4. Condiția de colinearitate a trei puncte $A(x_1, y_1)$; $B(x_2, y_2)$; $C(x_3, y_3)$ este :

$$\begin{vmatrix} 1 & 1 & 1 \\ X_1 & X_2 & X_3 \\ Y_1 & Y_2 & Y_3 \end{vmatrix} = 0$$

A.3. Coordonate polare

Elementele principale sînt: un punct și o rază care pornește din acel punct: Polul O și axa polară OX (fig. 2-A)

Poziția lui A este determinată în plan de distanța $\overline{OA} = \rho$, raza vectoroasă și de mărimea unghiului φ .

A.3.1. Distanța dintre două puncte $P_1(\rho_1, \varphi_1)$; $P_2(\rho_2, \varphi_2)$ este :

$$\overline{P_1P_2} = \sqrt{\rho_1^2 + \rho_2^2 - 2\rho_1\rho_2 \cos(\varphi_2 - \varphi_1)}$$

A.4. Transformarea coordonatelor în plan

A.4.1. Relațiile dintre coordonatele rectangulare și cele polare :

$$x = \rho \cos \varphi \qquad y = \rho \sin \varphi$$

și invers:

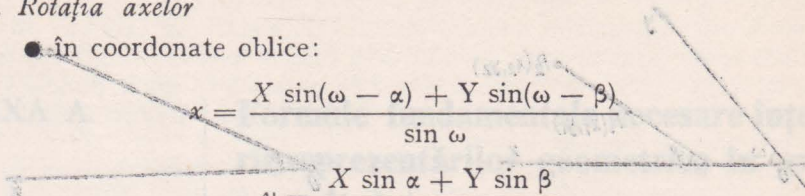
$$\rho = \sqrt{x^2 + y^2} \qquad \varphi = \operatorname{arc} \operatorname{tg} \frac{y}{x}$$

A.4.2. Translația axelor din O în O' (a, b)

$$x = X' + a, \qquad y = Y' + b$$

A.4.3. *rotația axelor*

- în coordonate oblice:



$$x = \frac{X \sin(\omega - \alpha) + Y \sin(\omega - \beta)}{\sin \omega}$$

$$y = \frac{X \sin \alpha + Y \sin \beta}{\sin \omega}$$

α și β fiind noile unghiuri formate de Ox_1 și Oy_1 cu OX .

- în coordonate rectangulare

$$x = X \cos \alpha - Y \sin \alpha$$

$$y = X \sin \alpha + Y \cos \alpha$$

$$X = x \cos \alpha + y \sin \alpha$$

$$Y = -x \sin \alpha + y \cos \alpha$$

A.5. *Dreapta în plan*A.5.1. *Ecuația dreptei*

- Dreapta care trece prin două puncte $P_1(x_1, y_1)$, $P_2(x_2, y_2)$:

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1} (x - x_1)$$

$$\begin{vmatrix} x & y & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{vmatrix} = 0$$

- Dreapta cu coeficientul unghiular m și care trece prin punctul $P_0(x_0, y_0)$:

$$y - y_0 = m(x - x_0)$$

- Dreapta cu coeficientul unghiular m și ordonata la origine n :

$$y = mx + n$$

- Dreapta cu tăieturile a și b pe axele de coordonate:

$$\frac{x}{a} + \frac{y}{b} = 1$$

A.5.2. *Ecuația canonică a dreptei*

- în coordonate rectangulare:

$$\frac{x - x_1}{A} = \frac{y - y_1}{B};$$

- în coordonate oblice:

$$\frac{x - x_1}{A - B \cos \omega} = \frac{y - y_1}{B - A \cos \omega}$$

A.5.3. Ecuația generală a dreptei

$$Ax + By + C = 0$$

$$Ax + By + Cz = 0 \quad (\text{omogenă})$$

A.5.4. Ecuația normală a dreptei

- în coordonate rectangulare (fig. 3-A)

$$x \cos \alpha + y \sin \alpha - p = 0$$

- în coordonate oblice (fig. 4-A)

$$x \cos \alpha + y \cos \beta - p = 0$$

Ecuația generală poate fi adusă la forma normală înmulțind-o cu factorul K

- în coordonate rectangulare:

$$K = \pm \frac{1}{\sqrt{A^2 + B^2}}$$

- în coordonate oblice:

$$K = \pm \frac{\sin \omega}{\sqrt{A^2 + B^2 - 2AB \cos \omega}}$$

A.5.5. Distanța de la un punct $M(x_1, y_1)$ la o dreaptă

- în coordonate rectangulare:

$$d = |x_1 \cos \alpha + y_1 \sin \alpha - p| = \left| \frac{Ax_1 + By_1 + C}{\pm \sqrt{A^2 + B^2}} \right|$$

- în coordonate oblice:

$$d = |x_1 \cos \alpha + y_1 \cos \beta - p| = \left| \frac{(Ax_1 + By_1 + C) \sin \omega}{\pm \sqrt{A^2 + B^2 - 2AB \cos \omega}} \right|$$

A.5.6. Bisectoarele unghiului dintre două drepte

$Ax + By + C = 0$ (ecuațiile a două drepte care fac

$A_1x + B_1y + C_1 = 0$ (între ele unghiul θ)

Ecuația bisectoarei unghiului θ este:

- în coordonate rectangulare:

$$\frac{Ax + By + C}{\sqrt{A^2 + B^2}} = \pm \frac{A_1x + B_1y + C_1}{\sqrt{A_1^2 + B_1^2}}$$

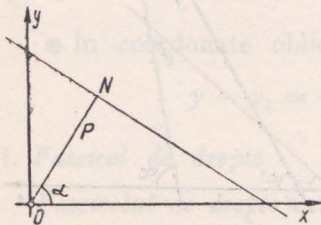


Fig. 3 - A

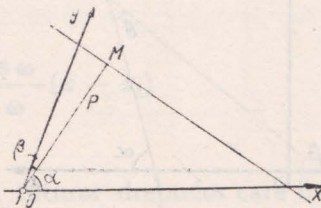


Fig. 4 - A

sau

$$\frac{\mu - m_1}{1 + m_1 \mu} + \frac{\mu - m_2}{1 + m_2 \mu} = 0,$$

m_1, m_2 fiind coeficienții unghiulari ai laturilor unghiului, iar μ fiind coeficientul unghiular al bisectoarei interioare sau exterioare

● în coordonate oblice:

$$\frac{Ax + By + C}{\sqrt{A^2 + B^2 - 2AB \cos \omega}} = \pm \frac{A_1x + B_1y + C_1}{\sqrt{A_1^2 + B_1^2 - 2A_1B_1 \cos \omega}}$$

sau

$$\frac{\mu - m_1}{1 + m_1 \mu + (m_1 + \mu) \cos \omega} + \frac{\mu - m_2}{1 + m_2 \mu + (m_2 + \mu) \cos \omega} = 0$$

A.5.7. Unghiul dintre două drepte

● în coordonate rectangulare (fig. 5 - A)

$$\operatorname{tg} \theta = \frac{m_1 - m_2}{1 + m_1 m_2} = \frac{\operatorname{tg} \alpha - \operatorname{tg} \beta}{1 + \operatorname{tg} \alpha \operatorname{tg} \beta} = \frac{A_1 B_2 - A_2 B_1}{A_1 A_2 + B_1 B_2}$$

● în coordonate oblice: (fig. 6-A)

$$\operatorname{tg} \theta = \frac{(m_1 - m_2) \sin \omega}{1 + m_1 m_2 + (m_1 + m_2) \cos \omega} = \frac{(A_1 B_2 - A_2 B_1) \sin \omega}{A_1 A_2 + B_1 B_2 - (A_1 B_2 + B_1 A_2) \cos \omega}$$

A.5.8. Fie două drepte

$$(D_1) = A_1 x + B_1 y + C_1 = 0$$

$$(D_2) = A_2 x + B_2 y + C_2 = 0$$

(D_1) și (D_2) reprezintă aceeași dreaptă dacă:

$$\frac{A_1}{A_2} = \frac{B_1}{B_2} = \frac{C_1}{C_2}$$

● Condiția de paralelism a dreptelor (D_1) și (D_2) :

$$\frac{A_1}{A_2} = \frac{B_1}{B_2} \neq \frac{C_1}{C_2} \text{ sau } m_1 = m_2$$

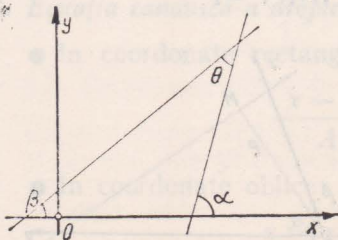


Fig. 5 - A

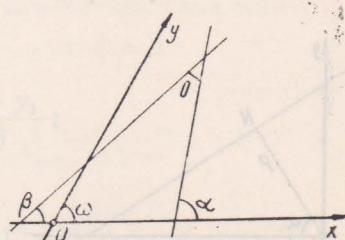


Fig. 6 - A

Condiția de perpendicularitate

- în coordonate rectangulare:

$$A_1A_2 + B_1B_2 = 0$$

sau

$$m_1 \cdot m_2 = -1; \quad m_1 = -\frac{1}{m_2}$$

- în coordonate oblice:

$$A_1A_2 + B_1B_2 - (A_1B_2 + A_2B_1) \cos \omega = 0$$

A.5.9. Discuția poziției relative dintre două drepte

Dacă

$$D = \begin{vmatrix} A_1 & B_1 \\ A_2 & B_2 \end{vmatrix} \neq 0$$

Dreptele (D_1) și (D_2) se întâlnesc în punctul

- $$x = \frac{B_1C_2 - C_1B_2}{D}$$

$$y = \frac{C_1A_2 - A_1C_2}{D}$$

- Dacă $D = 0$ și $B_1C_2 - C_1B_2 \neq 0$ și $C_1A_2 - A_1C_2 \neq 0$

Dreptele (D_1) și (D_2) sînt paralele;

- Dacă $D = 0$ și $B_1C_2 - C_1B_2 = 0$ atunci

$$\frac{A_1}{A_2} = \frac{B_1}{B_2} = \frac{C_1}{C_2}; \quad C_1A_2 - A_1C_2 = 0$$

Dreptele (D_1) și (D_2) sînt confundate;**A.5.10. Perpendiculara dusă din punctul $P(x_1, y_1)$ pe o dreaptă de coeficient unghiular m**

- în coordonate rectangulare:

$$y - y_1 = -\frac{1}{m}(x - x_1)$$

- în coordonate oblice:

$$y - y_1 = -\frac{1 + m \cos \omega}{m + \cos \omega}(x - x_1)$$

A.5.11. Fascicol de drepte

Fascicolul de drepte este ansamblul tuturor dreptelor care trec printr-un punct numit centrul fascicolului, de coordonate x_1 și y_1 .

- Ecuația $A(x - x_1) + B(y - y_1) = 0$ reprezintă orice dreaptă a fascicolului.
- Centrul poate fi determinat și prin două drepte oarecare ale fascicolului:

$$Ax + By + C = 0; \quad A_1x + B_1y + C_1 = 0$$

- Orice dreaptă care trece prin punctul lor de intersecție este dată de ecuația:

$$(Ax + By + C) + \lambda(A_1x + B_1y + C_1) = 0$$

A6. Coordonate și transformări de coordonate în spațiu

A.6.1. *Sistemul de coordonate rectangulare* este format din trei axe Ox , Oy , Oz , perpendiculare două câte două: deci Oz este perpendiculară în O pe planul horizontal xOy , $Oy \perp xOz$ și $Ox \perp yOz$ (fig. 7 - A)

A.6.2. *Coordonate cilindrice* (fig. 8 - A)

$$x = \rho \cos \varphi$$

$$y = \rho \sin \varphi$$

$$z = z$$

A.6.3. *Coordonatele sferice* (fig. 9 - A)

$$x = r \sin \theta \cos \varphi$$

$$y = r \sin \theta \sin \varphi$$

$$z = r \cos \theta$$

A.6.4. *Relațiile între coordonatele carteziene și cele sferice*

$$r = \sqrt{x^2 + y^2 + z^2}$$

$$\operatorname{tg} \varphi = \frac{y}{x}$$

$$\cos \theta = \frac{z}{\sqrt{x^2 + y^2 + z^2}}$$

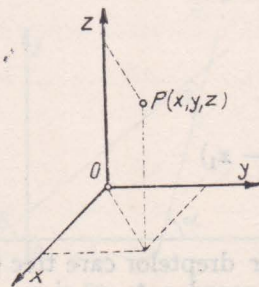


Fig. 7 - A

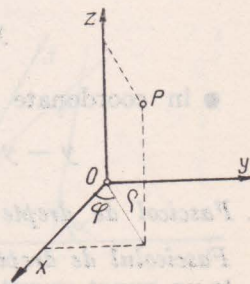


Fig. 8 - A

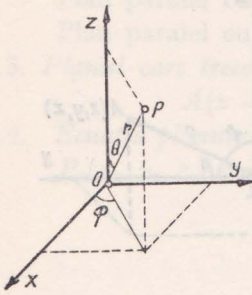


Fig. 9 - A

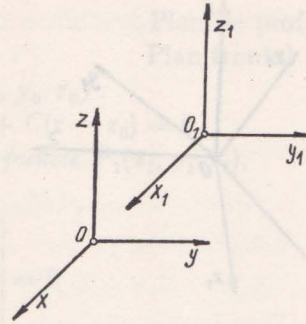


Fig. 10 - A

A.6.5. *Translația* (fig. 10—A)

Trecerea de la sistemul $Oxyz$ la sistemul $O_1x_1y_1z_1$, care este definit în raport cu primul prin origine și cosinușii directori ai axelor:

$$O_1x_1(a_1, b_1, c_1), O_1y_1(a_2, b_2, c_2), O_1z_1(a_3, b_3, c_3).$$

$$x = x_0 + x_1$$

$$y = y_0 + y_1$$

$$z = z_0 + z_1$$

A.6.6. *Rotația* (fig. 11—A)

$$x = a_1x_1 + a_2y_1 + a_3z_1$$

$$y = b_1x_1 + b_2y_1 + b_3z_1$$

$$z = c_1x_1 + c_2y_1 + c_3z_1$$

și invers:

$$x_1 = a_1x + b_1y + c_1z$$

$$y_1 = a_2x + b_2y + c_2z$$

$$z_1 = a_3x + b_3y + c_3z$$

Formule generale de transformare se obțin prin compunerea translației și rotației.

A.6.7. *Raza vectoroare* este distanța ρ de la origine la punctul M :

$$\rho^2 = x^2 + y^2 + z^2$$

A.6.8. *Coordonatele punctului* sînt proiecțiile razei sale vectoroare pe axele de coordonate (fig. 12—A)

$$x = \rho \cos \alpha$$

$$y = \rho \cos \beta$$

$$z = \rho \cos \gamma$$

Există relația :

$$\cos^2 \alpha + \cos^2 \beta + \cos^2 \gamma = 1$$

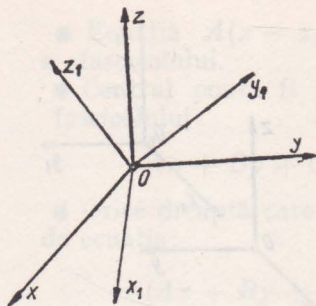


Fig. 11 - A

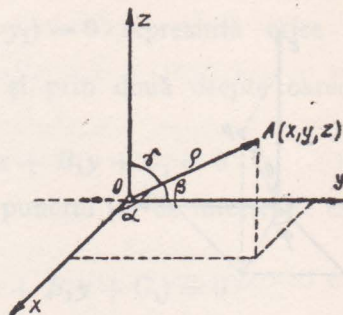


Fig. 12 - A

A.6.9. Direcția segmentului \overline{AB} este determinată de cosinșii lui directori

$$\cos \alpha = \frac{x_2 - x_1}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}$$

$$\cos \beta = \frac{y_2 - y_1}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}$$

$$\cos \gamma = \frac{z_2 - z_1}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}$$

A.6.10. Distanța dintre două puncte $A(x_1, y_1, z_1)$ și $B(x_2, y_2, z_2)$:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

A.6.11. Unghiul dintre două drepte date prin cosinșii lor directori este:

$$\cos \varphi = \cos \alpha \cos \alpha_1 + \cos \beta \cos \beta_1 + \cos \gamma \cos \gamma_1$$

A.6.12. Condiția de perpendicularitate dintre două drepte este:

$$\cos \alpha \cdot \cos \alpha_1 + \cos \beta \cdot \cos \beta_1 + \cos \gamma \cdot \cos \gamma_1 = 0$$

A.6.13. Punctul care împarte segmentul \overline{AB} , $A(x_1, y_1, z_1)$ și $B(x_2, y_2, z_2)$ în raportul λ are coordonatele

$$x = \frac{x_1 + \lambda x_2}{1 + \lambda}; \quad y = \frac{y_1 + \lambda y_2}{1 + \lambda}; \quad z = \frac{z_1 + \lambda z_2}{1 + \lambda}$$

A7. Planul

A.7.1. Ecuația generală a planului este: $Ax + By + Cz + D = 0$

A.7.2. Planele particulare sînt:

Plan care trece prin O : $Ax + By + Cz = 0$;

Plan paralel cu Ox : $By + Cz + D = 0$;

Plan paralel cu Oy : $Ax + Cz + D = 0$; Plan de capăt

Plan paralel cu Oz : $Ax + By + D = 0$; Plan vertical

Plan paralel cu yo_x : $z = c$ Plan de nivel

Plan paralel cu $yo z$: $x = a$; Plan de profilPlan paralel cu xoz : $y = b$. Plan frontalA.7.3. Planul care trece prin punctul $O(x_0, y_0, z_0)$:

$$A(x - x_0) + B(y - y_0) + C(z - z_0) = 0$$

A.7.4. Ecuația planului care trece prin trei puncte $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$ și $P_3(x_3, y_3, z_3)$:

$$\begin{vmatrix} x & y & z & 1 \\ x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \end{vmatrix} = 0$$

A.7.5. Ecuația planului dat prin tăieturile pe axe este :

$$\frac{x}{a} + \frac{y}{b} + \frac{z}{c} = 1$$

A.7.6. Ecuația normală a planului:

$$x \cos \alpha + y \cos \beta + z \cos \gamma - p = 0$$

Ecuația generală a planului poate fi transformată în ecuația normală înmulțind-o cu factorul K

$$K = \pm \frac{1}{\sqrt{A^2 + B^2 + C^2}}$$

A.7.7. Distanța de la un punct (x', y', z') la un plan este :

$$d = \left| \frac{Ax' + By' + Cz' + D}{\pm \sqrt{A^2 + B^2 + C^2}} \right|$$

sau

$$d = |x' \cos \alpha + y' \cos \beta + z' \cos \gamma - p|$$

Semnul radicalului este contrar semnului lui D

A.7.8. Colinearitatea a patru puncte :

Patru puncte (x_1, y_1, z_1) , (x_2, y_2, z_2) , (x_3, y_3, z_3) , și (x_4, y_4, z_4)

sînt colineare dacă:

$$\begin{vmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_4 & y_4 & z_4 & 1 \end{vmatrix} = 0$$

A.7.9. Condiția ca două plane să fie confundate :

Fie două plane:

$$A_1x + B_1y + C_1z + D_1 = 0$$

$$A_2x + B_2y + C_2z + D_2 = 0$$

avem

$$\frac{A_1}{A_2} = \frac{B_1}{B_2} = \frac{C_1}{C_2} = \frac{D_1}{D_2}$$

A.7.10. Condiția de paralelism dintre două plane este :

$$\frac{A_1}{A_2} = \frac{B_1}{B_2} = \frac{C_1}{C_2} \neq \frac{D_1}{D_2}$$

A.7.11. Condiția de ortogonalitate dintre două plane este :

$$A_1A_2 + B_1B_2 + C_1C_2 = 0$$

A.7.12. Ecuația unui fascicol de plane este :

$$A_1x + B_1y + C_1z + D_1 + \lambda(A_2x + B_2y + C_2z + D_2) = 0$$

Planele din fascicol se obțin dând diferite valori parametrului λ .

A.7.13. Unghiul dintre două plane:

$$\cos V = \pm \frac{A_1A_2 + B_1B_2 + C_1C_2}{\sqrt{A_1^2 + B_1^2 + C_1^2} \cdot \sqrt{A_2^2 + B_2^2 + C_2^2}}$$

sau:

$$\sin^2 V = \frac{(A_1B_2 - A_2B_1)^2 + (B_1C_2 - B_2C_1)^2 + (A_1C_2 - A_2C_1)^2}{(A_1^2 + B_1^2 + C_1^2)(A_2^2 + B_2^2 + C_2^2)}$$

A.7.14. Planele bisectoare ale diedrului format de două plane:

$$\frac{A_1x + B_1y + C_1z + D_1}{\sqrt{A_1^2 + B_1^2 + C_1^2}} \pm \frac{A_2x + B_2y + C_2z + D_2}{\sqrt{A_2^2 + B_2^2 + C_2^2}} = 0$$

1.7.15. Poziția relativă dintre trei plane

Fie trei plane:

$$A_1x + B_1y + C_1z + D_1 = 0$$

$$A_2x + B_2y + C_2z + D_2 = 0$$

$$A_3x + B_3y + C_3z + D_3 = 0$$

● Dacă

$$\Delta = \begin{vmatrix} A_1 & B_1 & C_1 \\ A_2 & B_2 & C_2 \\ A_3 & B_3 & C_3 \end{vmatrix} = 0, \text{ planele au un punct comun.}$$

● Dacă $\Delta = 0$ și un minor de ordinul al doilea diferit de zero, de exemplu:

$$\begin{vmatrix} A_1 & B_1 \\ A_2 & B_2 \end{vmatrix} \neq 0$$

$$\text{I } \begin{vmatrix} A_1 & B_1 & D_1 \\ A_2 & B_2 & D_2 \\ A_3 & B_3 & D_3 \end{vmatrix} \neq 0 \text{ planele sînt paralele cu aceeași dreaptă.}$$

$$\text{II } \begin{vmatrix} A_1 & B_1 & D_1 \\ A_2 & B_2 & D_2 \\ A_3 & B_3 & D_3 \end{vmatrix} = 0 \text{ planele trec printr-o aceeași dreaptă.}$$

● Dacă toți minorii de ordinul întâi ai lui Δ sînt nuli și au cel puțin un coeficient nenul, de exemplu $A_1 \neq 0$.

Planele sînt paralele

dacă

$$\begin{vmatrix} A & D \\ A_1 & D_1 \end{vmatrix} \text{ și } \begin{vmatrix} A & D \\ A_2 & D_2 \end{vmatrix} \text{ nu sînt nuli în același timp}$$

Planele sînt confundate

dacă

$$\begin{vmatrix} A & D \\ A_1 & D_1 \end{vmatrix} \text{ și } \begin{vmatrix} A & D \\ A_2 & D_2 \end{vmatrix} \text{ sînt nuli în același timp.}$$

A.7.16. Ecuația generală a planelor care trec prin punctul comun a trei plane:

$$P_1 = 0, \quad P_2 = 0, \quad P_3 = 0:$$

$$\lambda_1 P_1 + \lambda_2 P_2 + \lambda_3 P_3 = 0$$

A.7.17. Condiția ca trei plane să treacă prin aceeași dreaptă:

$$P_1 = 0, \quad P_2 = 0, \quad P_3 = 0$$

$$\begin{vmatrix} A_1 & B_1 & C_1 \\ A_2 & B_2 & C_2 \\ A_3 & B_3 & C_3 \end{vmatrix} = 0$$

A.7.18. Condiția ca patru plane să aibă un punct comun:

$$P_1 = 0, \quad P_2 = 0, \quad P_3 = 0, \quad P_4 = 0$$

$$\begin{vmatrix} A_1 & B_1 & C_1 & D_1 \\ A_2 & B_2 & C_2 & D_2 \\ A_3 & B_3 & C_3 & D_3 \\ A_4 & B_4 & C_4 & D_4 \end{vmatrix} = 0$$

A.7.19. Volumul unui tetraedru cu coordonatele vîrfurilor

(x_1, y_1, z_1) , (x_2, y_2, z_2) , (x_3, y_3, z_3) , (x_4, y_4, z_4) este:

$$V = \pm \frac{1}{6} \begin{vmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_4 & y_4 & z_4 & 1 \end{vmatrix}$$

A.7.20. Centrul de greutate al triunghiului $P_1P_2P_3$:

$$x = \frac{x_1 + x_2 + x_3}{3}, \quad y = \frac{y_1 + y_2 + y_3}{3}, \quad z = \frac{z_1 + z_2 + z_3}{3}$$

A.7.21. Centrul de greutate al tetraedrului $P_1P_2P_3P_4$:

$$x = \frac{x_1 + x_2 + x_3 + x_4}{4} = \frac{\Sigma x_i}{4}, \quad y = \frac{\Sigma y_i}{4}; \quad z = \frac{\Sigma z_i}{4}$$

A8. Dreapta în spațiu

A.8.1. Dreapta obținută prin intersecția dintre două plane:

$$A_1x + B_1y + C_1z + D_1 = 0$$

$$A_2x + B_2y + C_2z + D_2 = 0$$

sau

$$x = mz + n$$

$$y = pz + q$$

A.8.2. Dreapta care trece prin punctul $P_0(x_0, y_0, z_0)$ și are parametrii directori (a, b, c) are ecuațiile:

$$\frac{x - x_0}{a} = \frac{y - y_0}{b} = \frac{z - z_0}{c} = t$$

sau

$$x = x_0 + at$$

$$y = y_0 + bt$$

$$z = z_0 + ct \quad (\text{forma parametrică})$$

Există proporționalitatea:

$$a : b : c = \cos\alpha : \cos\beta : \cos\gamma$$

unde cosinuşii directori sînt dați de formulele:

$$\cos\alpha = \frac{a}{\sqrt{a^2 + b^2 + c^2}}$$

$$\cos\beta = \frac{b}{\sqrt{a^2 + b^2 + c^2}}$$

$$\cos\gamma = \frac{c}{\sqrt{a^2 + b^2 + c^2}}$$

A.8.3. Dreapta care trece prin două puncte $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$ este:

$$\frac{x - x_1}{x_2 - x_1} = \frac{y - y_1}{y_2 - y_1} = \frac{z - z_1}{z_2 - z_1}$$

A.8.4. Condiția ca trei puncte să fie colineare:

$$\frac{x_1 - x_3}{x_1 - x_2} = \frac{y_1 - y_3}{y_1 - y_2} = \frac{z_1 - z_3}{z_1 - z_2}$$

A.8.5. Distanța de la punctul $P_0(x_0, y_0, z_0)$ la dreapta

$$\frac{x - p}{\alpha} = \frac{y - q}{\beta} = \frac{z - r}{\gamma}$$

este:

$$d^2 = (x_0 - p)^2 + (y_0 - q)^2 + (z_0 - r)^2 - \frac{\alpha(x_0 - p) + \beta(y_0 - q) + \gamma(z_0 - r)}{\alpha^2 + \beta^2 + \gamma^2}$$

A.8.6. Distanța de la punctul $M(x, y, z)$ la dreapta care trece prin punctul $A(a, b, c)$ este:

$$d = \frac{[\bar{r}\bar{P}]}{[\bar{P}]}$$

unde $\bar{r} = \overline{AM}$, iar \bar{P} este orice vector dirijat după această dreaptă.

A.8.7. Distanța de la punctul $M(x_1, y_1, z_1)$ pînă la dreapta

$$\frac{x - a}{l} = \frac{y - b}{m} = \frac{z - c}{n}$$

este dată de relația:

$$d^2 = \frac{[m(x_1 - a) - l(y_1 - b)]^2 + [n(x_1 - a) - l(z_1 - c)]^2 + [n(y_1 - b) - m(z_1 - c)]^2}{l^2 + m^2 + n^2}$$

A.8.8. Condiția ca două drepte

$$\frac{x - p}{\alpha} = \frac{y - q}{\beta} = \frac{z - r}{\gamma}$$

$$\frac{x - p'}{\alpha_1} = \frac{y - q'}{\beta_1} = \frac{z - r'}{\gamma_1}$$

să fie concurente este

$$\begin{vmatrix} p - p' & \alpha & \alpha_1 \\ q - q' & \beta & \beta_1 \\ r - r' & \gamma & \gamma_1 \end{vmatrix} = 0$$

A.8.9. Condiția de paralelism:

$$\frac{\alpha}{\alpha_1} = \frac{\beta}{\beta_1} = \frac{\gamma}{\gamma_1}$$

A.8.10. Condiția de ortogonalitate dintre două drepte:

$$\alpha\alpha_1 + \beta\beta_1 + \gamma\gamma_1 = 0$$

A.8.11. Unghiul dintre două drepte :

$$\cos V = \pm \frac{\alpha\alpha_1 + \beta\beta_1 + \gamma\gamma_1}{\sqrt{\alpha^2 + \beta^2 + \gamma^2} \sqrt{\alpha_1^2 + \beta_1^2 + \gamma_1^2}}$$

$$\sin V = \pm \sqrt{\frac{(\beta\gamma_1 - \gamma\beta_1)^2 + (\gamma\alpha_1 - \alpha\gamma_1)^2 + (\alpha\beta_1 - \beta\alpha_1)^2}{(\alpha^2 + \beta^2 + \gamma^2) (\alpha_1^2 + \beta_1^2 + \gamma_1^2)}}$$

A.8.12. Unghiul unei drepte

$$\frac{x - p}{\alpha} = \frac{y - q}{\beta} = \frac{z - r}{\gamma}$$

cu planul $AX + BY + CZ + D = 0$:

$$\sin V = \frac{A\alpha + B\beta + C\gamma}{\pm \sqrt{(A^2 + B^2 + C^2) (\alpha^2 + \beta^2 + \gamma^2)}}$$

$$\cos V = \pm \sqrt{\frac{(B\gamma - C\beta)^2 + (C\alpha - A\gamma)^2 + (A\beta - B\alpha)^2}{(A^2 + B^2 + C^2) (\alpha^2 + \beta^2 + \gamma^2)}}$$

A.8.13. Intersecția dreptei $\frac{x - p}{\alpha} = \frac{y - q}{\beta} = \frac{z - r}{\gamma}$ cu planul $Ax + By +$

$+ Cz + D = 0$:

1. $A\alpha + B\beta + C\gamma \neq 0$, un punct de intersecție

2. $A\alpha + B\beta + C\gamma = 0$ $\begin{cases} A p + B q + C r + D \neq 0, \text{ dreapta paralelă cu} \\ \text{un plan} \\ A p + B q + C r + D = 0, \text{ dreapta situată în} \\ \text{plan} \end{cases}$

A.8.14. Condiția ca dreapta $\frac{x - p}{\alpha} = \frac{y - q}{\beta} = \frac{z - r}{\gamma}$ să fie paralelă cu

planul $Ax + By + Cz + D = 0$

este: $A\alpha + B\beta + C\gamma = 0$

A.8.15. Condiția ca o dreaptă să fie perpendiculară pe un plan este :

$$\frac{A}{\alpha} = \frac{B}{\beta} = \frac{C}{\gamma}$$

A.8.16. Ecuația planului care trece prin două drepte concurente este :

$$\begin{vmatrix} x - p & y - q & z - r \\ \alpha & \beta & \gamma \\ \alpha_1 & \beta_1 & \gamma_1 \end{vmatrix} = 0$$

A.8.17. Ecuația planului care trece prin două drepte paralele este :

$$\begin{vmatrix} x - p & y - q & z - r \\ x - p' & y - q' & z - r' \\ \alpha & \beta & \gamma \end{vmatrix} = 0$$

A.8.18. Ecuația perpendicularei comune dintre două drepte disjuncte este :

$$\begin{vmatrix} x - p & y - q & z - r \\ \alpha & \beta & \gamma \\ \beta\gamma_1 - \gamma\beta_1 & \gamma\alpha_1 - \gamma\alpha_1 & \alpha\beta_1 - \beta\alpha_1 \end{vmatrix} = 0$$

sau

$$\begin{vmatrix} y - p' & y - q' & z - r' \\ \alpha_1 & \beta_1 & \gamma_1 \\ \beta\gamma_1 - \gamma\beta_1 & \gamma\alpha_1 - \gamma\alpha_1 & \alpha\beta_1 - \beta\alpha_1 \end{vmatrix} = 0$$

A.8.19. Lungimea perpendicularei comune sau cea mai scurtă distanță dintre două drepte disjuncte este :

$$d = \frac{\begin{vmatrix} p - p' & \alpha & \alpha_1 \\ q - q' & \beta & \beta_1 \\ r - r' & \gamma & \gamma_1 \end{vmatrix}}{\pm \sqrt{(\beta\gamma_1 - \gamma\beta_1)^2 + (\gamma\alpha_1 - \alpha\gamma_1)^2 + (\alpha\beta_1 - \beta\alpha_1)^2}}$$

sau

$$d = \frac{(\beta\gamma_1 - \gamma\beta_1)(p - p') + (\gamma\alpha_1 - \alpha\gamma_1)(q - q') + (\alpha\beta_1 - \beta\alpha_1)(r - r')}{\pm \sqrt{(\beta\gamma_1 - \gamma\beta_1)^2 + (\gamma\alpha_1 - \alpha\gamma_1)^2 + (\alpha\beta_1 - \beta\alpha_1)^2}}$$

A.8.20. Lungimea d a perpendicularei comune dintre dreptele disjuncte

$$\frac{x - a}{l} = \frac{y - b}{m} = \frac{z - c}{n}$$

$$\frac{x - a_1}{l_1} = \frac{y - b_1}{m_1} = \frac{z - c_1}{n_1}$$

este exprimată de relația

$$d = \frac{\begin{vmatrix} a_1 - a & b_1 - b & c_1 - c \\ l & m & n \\ l_1 & m_1 & n_1 \end{vmatrix}}{\sqrt{(lm_1 - l_1m)^2 + (mn_1 - m_1n)^2 + (ln_1 - l_1n)^2}}$$

A.8.21. Distanța dintre două drepte disjuncte este :

$$d = \frac{|(\vec{r}_1 \vec{r}_2 \vec{r}_3)|}{|[\vec{r}_1 \vec{r}_2]|}$$

unde r_1 și r_2 sînt doi vectori arbitrari dirijați de-a lungul acestor drepte iar r_3 este un vector care unește un punct de pe o dreaptă cu un punct de pe cealaltă dreaptă.

A.8.22. Distanța d dintre două drepte paralele este :

$$d = \frac{|[\bar{r}\bar{P}]|}{|\bar{P}|}$$

unde \bar{r} este un vector, care unește un punct de pe o dreaptă cu un punct de pe cealaltă dreaptă, iar \bar{P} este un vector paralel cu dreptele date.

A.8.23. Distanța d dintre dreptele paralele

$$\frac{x - a}{l} = \frac{y - b}{m} = \frac{z - c}{n}$$

$$\frac{x - a_1}{l} = \frac{y - b_1}{m} = \frac{z - c_1}{n}$$

este dată de relația

$$d^2 = \frac{[m(a_1 - a) - l(b_1 - b)]^2 + [n(b_1 - b) - m(c_1 - c)]^2 + [l(c_1 - c) - n(a_1 - a)]^2}{l^2 + m^2 + n^2}$$

A.8.24. Bisectoarele unghiurilor dintre dreptele concurente

$$\frac{x - a}{l} = \frac{y - b}{m} = \frac{z - c}{n}$$

$$\frac{x - a}{l_1} = \frac{y - b}{m_1} = \frac{z - c}{n_1}$$

sînt date de relațiile

$$x = a + \frac{l}{\Delta} \pm \frac{l_1}{\Delta_1} \cdot t$$

$$y = b + \frac{m}{\Delta} \pm \frac{m_1}{\Delta_1} \cdot t$$

$$z = c + \frac{n}{\Delta} \pm \frac{n_1}{\Delta_1} \cdot t$$

unde

$$\Delta = \sqrt{l^2 + m^2 + n^2}$$

$$\Delta_1 = \sqrt{l_1^2 + m_1^2 + n_1^2}$$

A9. Cercul

A.9.1. Ecuația cercului este

$$(x - a)^2 + (y - b)^2 = r^2 \quad (\text{axe rectangulare; fig. 13 - A})$$

$$x^2 + y^2 - 2ax - 2by + c = 0;$$

$$c = a^2 + b^2 - r^2$$

sau în general,

$$x^2 + y^2 + mx + ny + p = 0$$

coordonatele centrului sînt:

$$x = -\frac{m}{2}; \quad y = -\frac{n}{2}$$

iar raza cercului este:

$$r = \sqrt{\frac{m^2}{4} + \frac{n^2}{4} - p}$$

Dacă centrul este în origine, ecuația cercului este:

$$x^2 + y^2 = R^2 \quad (\text{axe rectangulare (fig. 13 - A)})$$

$$(x - a)^2 + 2(x - a)(y - b) \cos \omega + (y - b)^2 = R^2 \quad (\text{axe oblice; fig. 14 - A})$$

A.9.2. Ecuația cercului care trece prin trei puncte

$$A(x_1, y_1), \quad B(x_2, y_2), \quad C(x_3, y_3):$$

$$\begin{vmatrix} x^2 + y^2 & x & y & 1 \\ x_1^2 + y_1^2 & x_1 & y_1 & 1 \\ x_2^2 + y_2^2 & x_2 & y_2 & 1 \\ x_3^2 + y_3^2 & x_3 & y_3 & 1 \end{vmatrix} = 0 \quad (\text{axe rectangulare})$$

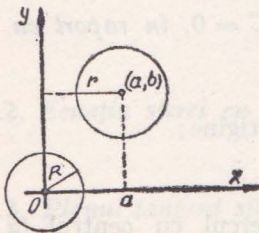


Fig. 13 - A

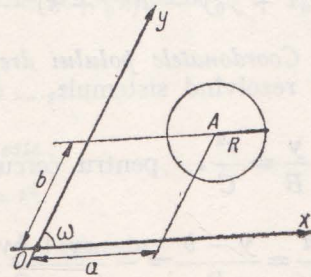


Fig. 14 - A

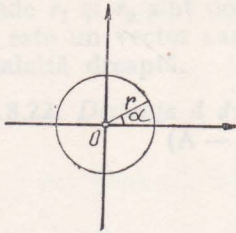


Fig. 15 - A

A.9.3. Ecuația cercului în coordonate polare

$$r^2 - 2rr_0 \cos(\varphi - \varphi_0) + r_0^2 = R^2$$

unde: r_0 este raza vectoare a centrului;
 φ_0 — unghiul polar al centrului;
 R — raza cercului.

A.9.4. Reprezentarea parametrică a cercului

$$\begin{aligned} x &= r \cos \alpha \\ y &= r \sin \alpha \end{aligned} \quad \text{cu centrul în origine (fig. 15 - A)}$$

$$\begin{aligned} x &= a + r \cos \alpha \\ y &= b + r \sin \alpha \end{aligned} \quad \text{cu centrul în } (a, b)$$

A.9.5. Ecuația tangentei în $P_0(x_0, y_0)$ se obține prin polarizarea (dedublarea) ecuației cercului: $x_0x + y_0y - r^2 = 0$, la cercul cu centrul în origine:

$$x_0x + y_0y + a(x_0 + x) + b(y_0 + y) + c = 0$$

la cercul cu centrul în (a, b) .

A.9.6. Ecuația tangentei de direcție dată m :

$$y = mx \pm r\sqrt{1 + m^2}, \quad \text{la cercul cu centrul în origine;}$$

$$y = m(x - a) + b \pm r\sqrt{1 + m^2}, \quad \text{la cercul cu centrul } (a, b).$$

A.9.7. Ecuația normalei în $P_0(x_0, y_0)$:

$$y - y_0 = \frac{y_0}{x_0} (x - x_0), \quad \text{la cercul cu centrul în origine;}$$

$$y - y_0 = \frac{y_0 + b}{x_0 + a} (x - x_0), \quad \text{la cercul cu centrul } (a, b).$$

A.9.8. Ecuația polarei punctului $P_0(x_0, y_0)$:

$$x_0x + y_0y - r^2 = 0, \quad \text{centrul în origine}$$

$$x_0x + y_0y - a(x_0 + x) - b(y_0 + y) + c = 0, \quad \text{centrul } (a, b).$$

A.9.10. Coordonatele polului dreptei $Ax + By + C = 0$, în raport cu cercul se află rezolvând sistemul:

$$\frac{x}{A} = \frac{y}{B} = \frac{r^2}{C}, \quad \text{pentru cercul cu centrul în origine;}$$

$$\frac{x - a}{A} = \frac{y - b}{B} = -\frac{ax + by - c}{C}, \quad \text{pentru cercul cu centrul } (a, b).$$

A.9.11. *Axa radicală a două cercuri* (locul punctelor de puteri egale în raport cu două cercuri)

Ecuatiile cercurilor:

$$(C_1) = x^2 + y^2 - 2a_1x - 2b_1y + C_1 = 0$$

$$(C_2) = x^2 + y^2 - 2a_2x - 2b_2y + C_2 = 0$$

Ecuatia axei radicale:

$$(C_1) - (C_2) = 2(a_1 - a_2)x + 2(b_1 - b_2)y + C_2 - C_1 = 0$$

A.9.12. *Condiția ca două cercuri să fie ortogonale este:*

$$d^2 = R^2 + R'^2 \quad d = \text{distanța centrelor}$$

A.9.13. *Fascicolul de cercuri determinat de cercurile de bază*

$$C_1 = 0, \quad C_2 = 0$$

Are ecuația:

$$C_1 + \lambda C_2 = 0$$

Prin variația parametrului λ se obțin toate cercurile din fascicol.

A.9.14. *Puterea punctului $P(x_0, y_0)$ în raport cu cercul*

$$f(x, y) = A(x^2 + 2xycos\theta + y^2) + 2Dx + 2Ey + F = 0$$

este

$$\frac{f(x_0, y_0)}{A}$$

A. 10 Sfera

A.10.1. *Ecuatia sferei cu centrul în punctul (a, b, c) este:*

$$(x - a)^2 + (y - b)^2 + (z - c)^2 = r^2$$

sau

$$x^2 + y^2 + z^2 - 2ax - 2by - 2cz + d = 0$$

unde

$$d = a^2 + b^2 + c^2 - r^2$$

A.10.2. *Ecuatia sferei cu centrul în origine este:*

$$x^2 + y^2 + z^2 = r^2$$

A.10.3. *Planul tangent sferei în punctul (x_1, y_1, z_1) este:*

$$(x_1 - a)(x - a) + (y_1 - b)(y - b) + (z_1 - c)(z - c) = r^2$$

A.10.4. Puterea punctului (x_0, y_0, z_0) față de sferă este :

$$(x - a)(x_0 - a) + (y - b)(y_0 - b) + (z - c)(z_0 - c) = r^2$$

sau :

$$x_0^2 + y_0^2 + z_0^2 - 2ax_0 - 2by_0 - 2cz_0 + d = 0$$

A.10.5. Planul radical a două sfere C_1, C_2 este:

$$C_1 - C_2 = 0$$

$$2(a_1 - a_2)x + 2(b_1 - b_2)y + 2(c_1 - c_2)z = d_1 - d_2$$

A.10.6. Considerînd sfera

$$(x - a)^2 + (y - b)^2 + (z - c)^2 - R^2 = 0$$

extremitățile diametrului ale cărui cosinusuri directoare sînt m, n, p , sînt :

$$A_1(a + mR, b + nR, c + pR)$$

$$A_2(a - mR, b - nR, c - pR)$$

A.10.7. Considerînd sfera

$$(x - a)^2 + (y - b)^2 + (z - c)^2 - R^2 = 0$$

și planul $\alpha x + \beta y + \gamma z = 0$

ecuațiile planelor tangente la sferă paralele cu acest plan sînt :

$$\alpha(x - a) + \beta(y - b) + \gamma(z - c) - R\sqrt{\alpha^2 + \beta^2 + \gamma^2} = 0$$

$$\alpha(x - a) + \beta(y - b) + \gamma(z - c) + R\sqrt{\alpha^2 + \beta^2 + \gamma^2} = 0$$

A.10.8. Considerînd sfera

$$(x - a)^2 + (y - b)^2 + (z - c)^2 - R^2 = 0$$

și planul $\alpha x + \beta y + \gamma z + d = 0$ coordonatele punctului situat în plan a cărui putere față de sferă este minimă sînt :

$$x = a - \frac{\alpha(\alpha a + \beta b + \gamma c + d)}{\alpha^2 + \beta^2 + \gamma^2}$$

$$y = b - \frac{\beta(\alpha a + \beta b + \gamma c + d)}{\alpha^2 + \beta^2 + \gamma^2}$$

$$z = c - \frac{\gamma(\alpha a + \beta b + \gamma c + d)}{\alpha^2 + \beta^2 + \gamma^2}$$

A.10.9. Considerînd sfera

$$(x - a)^2 + (y - b)^2 + (z - c)^2 - R^2 = 0$$

și dreapta

$$\frac{x - x_1}{\alpha} = \frac{y - y_1}{\beta} = \frac{z - z_1}{\gamma}$$

coordonatele punctului situat pe dreaptă a cărui putere față de sferă este minimă sînt :

$$x = x_1 - \alpha \frac{\alpha(x_1 - a) + \beta(y_1 - b) + \gamma(z_1 - c)}{\alpha^2 + \beta^2 + \gamma^2} = x_1 - \alpha K$$

$$y = y_1 - \beta K$$

$$z = z_1 - \gamma K$$

A.11. Conul

A.11.1. Ecuația conului cu vîrful la intersecția planelor $P = 0$, $Q = 0$, $R = 0$, este :

$$f\left(\frac{P}{R}, \frac{Q}{R}\right) = 0$$

A.11.2. Planul tangent conului în punctul $M_0(x_0, y_0, z_0)$ este :

$$Pf'_{P_0} + Qf'_{Q_0} + Rf'_{R_0} = 0$$

A.11.3. Ecuația omogenă a conului cu vîrful în punctul $P_0(z_0, y_0, z_0)$ este:

$$f(x - x_0, y - y_0, z - z_0) = 0$$

A.11.4. Ecuația conului avînd vîrful în punctul (a, b, c) și curba directoare $f(x, y) = 0$; $z = 0$ este:

$$f\left(\frac{cz - az}{c - z}, \frac{cy - bz}{c - z}\right) = 0$$

A.11.5. Ecuația unui con de gradul al doilea raportat la axele sale de simetrie (cu vîrful în origine) este:

$$ax^2 + by^2 + cz^2 = 0$$

A.11.6. Vîrful conului de gradul al doilea (sau de orice grad) se determină rezolvînd sistemul

$$f'_x = 0; \quad f'_y = 0; \quad f'_z = 0$$

A.11.7. Ecuația conului de gradul doi cu vîrful în punctul $P(x_0, y_0, z_0)$ și tangent sferei

$$f(x, y, z) \equiv x^2 + y^2 + z^2 - 2ax - 2by - 2cz + d = 0 \text{ este :}$$

$$[(x - x_0)(x_0 - a) + (y - y_0)(y_0 - b) + (z - z_0)(z_0 - c)]^2 -$$

$$- [(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2]f(x_0, y_0, z_0) = 0$$

A.11.8. Ecuația conului de gradul doi cu vîrful în punctul $P(x_0, y_0, z_0)$ și care trece prin cercul de intersecție al planului

$$\rho(x, y, z) \equiv Ax + By + Cz + d = 0 \text{ cu sfera}$$

$$f(x, y, z) = x^2 + y^2 + z^2 - 2ax - 2by - 2cz + d = 0$$

este

$$[(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2] \rho^2(x_0, y_0, z_0) - 2[A(x - x_0) + B(y - y_0) + C(z - z_0)] [x - x_0)(x_0 - a) + (y - y_0)(y_0 - b) + (z - z_0)(z_0 - c)] \rho(x_0, y_0, z_0) + [A(x - x_0) + B(y - y_0) + C(z - z_0)]^2 f(x_0, y_0, z_0) = 0$$

A.12. Cilindrul

A.12.1. Ecuația cilindrului cu generatoarea paralelă cu dreapta (D) este :

$$(D) \quad \begin{aligned} P &= ax + by + cz + d = 0, \\ Q &= a'x + b'y + c'z + d' = 0, \quad f(P, Q) = 0 \end{aligned}$$

A.12.2. Planul tangent cilindrului în punctul $M_0(x_0, y_0, z_0)$, axa fiind paralelă cu generatoarea este :

$$(x - x_0) f'_x(x_0, y_0) + (y - y_0) f'_y(x_0, y_0) = 0$$

A.12.3. Ecuația cilindrului a cărui curbă directoare este $f(x, y) = 0$, $z = 0$ și ale cărui generatoare au parametrii directori (a, b, c) este :

$$f\left(\frac{cx - az}{c}, \frac{cy - bz}{c}\right) = 0$$

A.12.4. Considerînd suprafața $f(x, y, z) = 0$ și planul $Ax + By + Cz + D = 0$ proiecțiile curbei de intersecție dintre acestea pe planele de coordonate sînt :

$$z = 0, f\left(x, y, \frac{Ax + By + D}{-C}\right) = 0$$

$$y = 0, f\left(x, z, \frac{Ax + Cz + D}{-B}\right) = 0$$

$$x = 0, f\left(y, z, \frac{By + Cz + D}{-A}\right) = 0$$

A.12.5. Ecuația cilindrului de gradul doi care trece printr-un cerc fix de intersecție dintre un plan și o sferă este :

$$\rho^2(x, y, z) (l^2 + m^2 + n^2) - 2(Al + Bm + Cn) [l(x - a) + m(y - b) + n(z - c)] \rho(x, y, z) + (Al + Bm + Cn)^2 f(x, y, z) = 0$$

A.13. Curbe și suprafețe

A.13.1. Ecuațiile parametrice ale unei curbe strîmbe sînt :

$$x = x(t); \quad y = y(t); \quad z = z(t)$$

A.13.2. Ecuațiile tangentei în punctul $M(x_0, y_0, z_0)$ sînt :

$$\frac{x - x_0}{x'} = \frac{y - y_0}{y'} = \frac{z - z_0}{z'}$$

A.13.3. Ecuația planului osculator al curbei în punctul $M(x_0, y_0, z_0)$ este :

$$\begin{vmatrix} x - x_0 & y - y_0 & z - z_0 \\ x' & y' & z' \\ x'' & y'' & z'' \end{vmatrix} = 0$$

A.13.4. Ecuația planului tangent suprafeței $f(x, y, z) = 0$ în punctul $M(x_0, y_0, z_0)$ este :

$$(x - x_0) \frac{\partial f}{\partial x} + (y - y_0) \frac{\partial f}{\partial y} + (z - z_0) \frac{\partial f}{\partial z} = 0$$

unde $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$, $\frac{\partial f}{\partial z}$ sînt parametrii directori ai normalei la suprafață.

A.13.5. Ecuația planului tangent suprafeței $z = z(x, y)$ (forma explicită) în punctul $M(x_0, y_0, z_0)$ este :

$$z - z_0 = p(x - x_0) + q(y - y_0) \text{ unde}$$

$$p = \frac{\partial z}{\partial x}, \quad q = \frac{\partial z}{\partial y} \quad (\text{Notația lui Monge})$$

deasemenea se notează cu $r = \frac{\partial^2 z}{\partial x^2}$, $s = \frac{\partial^2 z}{\partial x \partial y}$, $t = \frac{\partial^2 z}{\partial y^2}$

A.13.6. Ecuațiile parametrice ale unei suprafețe sînt :

$$x = x(u, v); \quad y = y(u, v); \quad z = z(u, v)$$

iar ecuația planului tangent suprafeței în punctul $M(u, v)$ este

$$\begin{vmatrix} x - x_0 & y - y_0 & z - z_0 \\ x_u & y_u & z_u \\ x_v & y_v & z_v \end{vmatrix} = 0$$

unde x_u, x_v, \dots sînt derivatele parțiale în raport cu u , respectiv v .

A.13.7. Ecuația unei suprafețe de rotație este de forma $f(P, S) = 0$ unde $P = 0$ reprezintă un plan iar $S = 0$ reprezintă o sferă.

A.13.8. Ecuațiile parametrice ale unei suprafețe riglate sînt :

$$x = x_0 + t a \quad y = y_0 + t b \quad z = z_0 + t c$$

unde un parametru este t iar x_0, y_0, z_0, a, b și c depind de un al doilea parametru.

A.13.9. Ecuațiile parametrice ale unei suprafețe desfășurabile sînt:

$$x = x_0 + tx'; \quad y = y_0 + ty'; \quad z = z_0 + tz'$$

unde dreptele mobile sînt tangente unei curbe

$$x = x(t), \quad y = y(t), \quad z = z(t)$$

A.13.10. Ecuația înfășurătoarei unei familii de suprafețe

$f(x, y, z, \lambda) = 0$ se obține eliminînd parametrul λ între ecuațiile

$$f(x, y, z, \lambda) = 0 \quad \text{și}$$

$$\frac{\partial}{\partial \lambda} f(x, y, z, \lambda) = 0$$

Subprogramul HAȘURA utilizat pe mesele de desen de tip ARISTO.

Programul pentru produsul a două matrici.

B. 1. Subprogramul HAȘURA*)

Apelul FORTRAN al acestui subprogram este:

CALL HAȘURA (N, LGP, XV, YV, TETA, NPAS, PAS, LIMIT, RUTINA, XF, YF, DELTA, IER, NINT, XINT, JINT, NTAB)

Cu ajutorul acestui subprogram se poate hașura interiorul unei mulțimi P pe poligoane pline evitându-se interiorul altei mulțimi G de poligoane goale. Poligoanele celor două mulțimi sînt situate arbitrar unul față de altul. Domeniul mărginit de fiecare poligon trebuie să fie conex (laturile neadiacente să nu se întretaie).

De asemenea subprogramul HAȘURA asigură suplimentar:

- un model de hașură;
- apartenența unui punct stabilit din plan la una din liniile modelului de hașură.

Un model de hașură este alcătuit din n linii de hașură paralele care pot să nu fie echidistante. Fiecare linie este caracterizată prin distanța față de linia următoare a modelului și eventual printr-o anumită culoare de desen și model de linie întreruptă. Distanța asociată ultimei linii este cea față de prima linie a modelului. Modelul astfel definit se aplică ciclic.

A. Parametri de intrare conțin :

- descrierea domeniului;
- descrierea hașurii.

1. Descrierea domeniului:

N — Numărul total de poligoane (pline și goale) al figurii (maxim 255 de poligoane);

LGP — Vectorul lungimii poligoanelor. Pentru poligonul I avem:

LGP(I) < 0 — poligon gol

LGP(I) > 0 — poligon plin

IABS (LGP(I)) — număr de laturi (vîrfuri) ale poligonului

XV, YV — Vectorii absciselor/ordonatelor vîrfurilor poligoanelor în cm.

Toate vîrfurile poligoanelor I sînt consecutive, ordinea poligoanelor fiind cea definită de vectorul LGP. În cadrul unui poligon ordinea vîrfurilor corespunde unei ordine arbitrare de parcurgere a conturului acestuia.

2. Descrierea hașurii:

TETA — Unghiul direcției suportului liniilor de hașură în convenție trigonometrică. Unghiul se dă în grade sexazecimale.

NPAS — Numărul de linii ale modelului de hașură

PAS — Vectorul distanțelor dintre două linii consecutive ale modelului. Distanțele se dau în cm. (PAS(I) este distanța liniei I față de linia I + 1, dacă I < NPAS sau față de linia 1 dacă I = NPAS)

LIMIT — Cuvînt de stare avînd forma 16* NRLIN + IBIT

IBIT reprezintă configurația primilor 4 biți ai cuvîntului LIMIT care au semnificația următoare

IBIT = 1 hașurile suprapuse peste o latură a poligoanelor pline se hașurează (alfel se ignoră)

IBIT = 2 — hașurile suprapuse peste o latură a poligoanelor goale (aflate în int. unui polig. plin)

* Subprogramul a fost experimentat la CÎNOR—București

- IBIT = 4 — se apelează rutina RUTINA înaintea trasării unei noi linii a modelului de hașură
 IBIT = 8 — linia modelului de hașură cu numărul NRLIN trebuie să treacă prin punctul de coordonate XF, YF

Observații :

Dacă NRLIN = 0 se consideră prima linie a modelului (ca și pentru NRLIN = 1) se impune $0 < = \text{NRLIN} < = \text{NPAS}$.

Au sens toate cele 16 valori posibile pentru IBIT (0 — 15)
 RUTINA — Subrutină scrisă de utilizator (nume arbitrar) și apelată de subrutina HASURA (dacă IBIT = 4) înainte de trasarea unei noi linii a modelului de hașură.
 Apelul este de forma:

CALL RUTINA (JHAS)

unde JHAS este numărul liniei curente (care urmează să fie trasată) a modelului (JHAS = 1, 2, ..., NPAS). Prin acest apel utilizatorul poate controla unele și modelul de linie întreruptă) pentru următoarea linie a modelului de hașură.

Programul apelant trebuie să declare acest nume într-o declarație EXTERNAL. Dacă bitul al 3-lea din LIMIT este 0 argumentul RUTINA este ignorat (poate lipsi). Pentru masa ARISTO se poate folosi rutina HASPEN descrisă în continuare.

XF, YF — coordonatele date în cm ale unui punct arbitrat prin care se dorește să treacă linia NRLIN a modelului de hașură (dacă IBIT = 8)

Punctul XF, YF poate fi exterior poligoanelor figurii; în acest caz apartenența punctului la linia de hașură este virtuală adică s-ar realiza dacă am considera modelul de hașură extins la nesfârșit în plan.

Dacă bitul al 4-lea din LIMIT este 0 argumentele XF, YF sint ignorate (pot lipsi).

DELTA — Toleranța cu care se consideră că un punct aparține la o dreaptă. Se dă în cm și trebuie să fie pozitivă sau 0. Toate intersecțiile liniilor de hașură cu laturile poligoanelor se calculează luând în considerare această toleranță. În limitele acestei toleranțe se poate vorbi de suprapunerea între o linie de hașură și o latură a poligonului (IBIT = 1,2).

Orientativ DELTA = 0.005

B. Parametri de ieșire :

IER — Răspunsul subrutinei HASURA
 = 1 — hașura a decurs normal
 = 0 — nu s-a trasat nici o hașură (domeniu vid sau distanța între hașuri prea mare față de dimensiunile figurii)
 = -1 — depășirea zonelor de manevră furnizate de utilizator (vezi C)

C. Parametri de manevră

Pentru a nu limita aplicabilitatea subrutinei HASURA utilizatorului va trebui să furnizeze ca parametri un nr. de zone de manevră

NINT — dimensiunea vectorilor XINT și JINT. NINT trebuie să fie cel puțin egal cu nr. max de intersecții ale unei linii de hașură cu laturile tuturor poligoanelor figurii
 XINT — vector REAL * 4 de dimensiune NINT
 JINT — vector INTEGER * 2 de dimensiune NINT
 NTAB — vector INTEGER * 2 de dimensiune N (nr. total de poligoane)

Observație :

Subrutina aplică la început figurii (vectorilor XV, YV) o rotație de unghi TETA. La ieșire se aplică o rotație inversă). De aceea valorile XV, YV pot fi ușor diferite față de intrare

Subrutina nu face nici un control asupra valorilor de intrare.

Dacă sint eronate apar erori.

— Subrutina HASURA apelează subrutina PLOT din GDV6

— Subrutina HASURA nu trasează conturul domeniului hașurat

D. Subrutina de schimbare a peniței și control a modelului de linie întreruptă de hașurare

Este scrisă pentru masa ARISTO și folosește subrutina NEWPEN și INSERT din pachetul GDV6.

- utilizare: Cuvîntul LIMIT trebuie să aibă $IBIT = 4$ la apelul subrutinei HAȘURA
- inițializare: se recomandă ca înainte de apelul subrutinei HAȘURA programul apelant să execute o instrucțiune

CALL HASPEO (fără parametri)

pentru stabilirea condițiilor inițiale.

E. Subrutina HASPEN

Controlul peniței și al modelului de linie întreruptă se realizează prin specificarea numelui HASPEN pe poziția argumentului RUTINA la apelul subrutinei HAȘURA:

CALL HASURA (...NPAS, PAS, 4 + LIM, HASPEN, ...)

Subrutina HASPEN va fi apelată în cadrul subrutinei HAȘURA înainte de trasarea unei noi linii a modelului de hașură prin instrucțiunea:

CALL HASPEN (JHAS)

unde JHAS este nr. liniei ce urmează să fie trasată în cadrul modelului de hașură (JHAS = 1, ..., NPAS)

Informațiile privitoare la unealtă și modelul liniei vor fi comunicate de utilizator prin 2 COMMON-uri:

COMMON /HASPE 1 /LPEN (2, NPAS)

COMMON /HASPE 2 /LDEF (3, NPAS)

INTEGER * 2 LDEF

LOGICAL * 1 LPEN

cele două matrici conțin câte o coloană pentru fiecare linie a modelului de hașurare. În programul apelant dimensiunea NPAS se va înlocui cu nr. maxim de linii utilizabile de vreun model ce poate fi specificat de program.

Altfel subrutina HASPEN nu pune nici o limită asupra acestui număr. Pentru linia I a modelului semnificația informațiilor este următoarea:

- LPEN (1, I) = 0 — se păstrează penița liniei anterioare (eventual cea de la intrarea în subrutina HASURA)
- ≠ 0 — numărul peniței pentru linia I este LPEN (1, I) (fără alt control). Este necesar ca $1 \leq LPEN (1, I) \leq 4$.
- LPEN (2, I) < 0 — linia I se va trasa cu linie continuă (decuplare model linie)
- > 0 — numărul modelului de linie întreruptă pentru masa ARISTO. Este necesar ca $1 \leq LPEN (2, I) \leq 9$
- = 0 — ignorarea modelului de linie. Se păstrează modelul de linie al liniei anterioare (sau de la intrarea în HASURA)

Coloana I a tabelului LDEF se interpretează numai dacă LPEN (2, I) > 0.

Dacă LDEF (1, I) < 0 dimensiunile modelului de linie întreruptă sînt cele ale liniei anterioare (sau de la intrare în HASURA) putîndu-se schimba modeul propriu-zis al liniei întrerupte (1—9). Altfel LDEF (1, I), LDEF (2, I), LDEF (3, I) reprezintă valorile parametrilor I, J, K ai comenzii D 32, respectiv lungimile celor două segmente ale modelului în lungimea pauzei. Aceste valori se vor da direct în sutimi de mm.

F. Sfirșit de hașură

În cazul schimbării peniței și folosirii modelului de linie, starea acestor parametri la masa ARISTO, la ieșirea din HAȘURA este impredictibilă utilizatorul va trebui să forțeze starea dorită. De exemplu:

CALL NEWPEN (1) — restabilirea peniței 1

CALL INSERT ('D 10', 3) — decuplare model de linie întreruptă.

Pentru scrierea programului apelant sînt utile următoarele recomandări:

INTEGER * 2 LDEF

LOGICAL * 1 LPEN

COMMON HASPE 1/LPEN (2, 4) } maximum 4 linii în

COMMON HASPE 2/LDEF (3, 4) } modelul de hașură

EXTERNAL HASPEN

Pregătire parametri LPEN, LDEF

CALL HASPEO

CALL HASURA (... , NPAS, PAS, 4 + LIM, HASPEN, ...)

CALL NEWPEN (1)

CALL INSERT ('D 10', 3)

END.

B.2. Program pentru efectuarea produsului dintre două matrici

```

PROGRAM PRMAT1
C PROGRAM PENTRU PRODUSUL A DOUA MATRICI
  DIMENSION A(10,10),B(10,10),C(10,10)
  DATA BLANK/' '/
  CALL ASSIGN(1,'CR:')
  CALL ASSIGN(2,'LP:')
  DO 1 I=1,10
  DO 1 J=1,10
  A(I,J)=BLANK
  B(I,J)=BLANK
  C(I,J)=BLANK
  1 CONTINUE
  READ(1,2) N,M,L
  2 FORMAT(3I2)
  DO 3 I=1,N
  3 READ(1,4) (A(I,J),J=1,M)
  4 FORMAT(8F10.3)
  DO 5 J=1,M
  5 READ(1,4) (B(J,K),K=1,L)
  WRITE(2,10)
  10 FORMAT(' ','DATELE DE INTRARE SINT')
  DO 11 I=1,N
  WRITE(2,4) (A(I,J),J=1,M)
  11 CONTINUE
  DO 12 I=1,M
  WRITE(2,4) (B(I,J),J=1,L)
  12 CONTINUE
  CALL PRMAT(A,B,C,N,M,L)
  WRITE(2,8)
  8 FORMAT(' ','PRODUSUL CELOR DOUA MATRICI ESTE')
  DO 7 I=1,N
  7 WRITE(2,9) (C(I,K),K=1,L)
  9 FORMAT(' ','10F10.3/')
  STOP
  END

```

```

SUBROUTINE PRMAT(X,Y,Z,N,M,L)
  DIMENSION X(10,10),Y(10,10),Z(10,10)
  DO 5 I=1,N
  DO 5 K=1,L
  Z(I,K)=0
  DO 5 J=1,M
  5 Z(I,K)=Z(I,K)+X(I,J)*Y(J,K)
  RETURN
  END

```

DATELE DE INTRARE SINT

```

*
  2.000    -1.000    -3.000
  0.000     2.000    -1.000
  0.000     3.000
  -1.000     1.000
  3.000     4.000

```

PRODUSUL CELOR DOUA MATRICI ESTE

```

*
  -8.000    -7.000
  -2.000    -2.000

```


**CĂRȚI APĂRUTE ÎN 1986-89 CARE SE MAI POT AFLA ÎN
REȚEAUA DE LIBRĂRII**

1. Adrian Davidoviciu
Boldur Bărbat
 2. Marius Guran
Florin Filip
 - 3-4. Cr. Giumale, D. Preoteșcu,
L. D. Șerbănași, D. Tufiș,
Gh. Tecuci, D. Cristea
 5. M. Suțiu, D. Popescu
Tr. Ionescu
 - 6-7. T. Baron, Al. Isaic-Maniu,
L. Tovissi, D. Niculescu,
C. Baron, V. Antonescu,
I. Roman
 - 8-9. Gh. Turbuș, I. Boicu,
E. Spirea, M. Huțanu,
I. Tomescu și colectiv 100 spe-
cialiști din MTTc, ITCI, IPA
 - 10-13. Colective largi
 14. T. Geber, V. Cristea,
V. Săvescu, I. Miu,
R. Bulgakov, M. Vuici
 - 15-16. Gh. Sabău, Al. Sotir și
colectiv ASE, CSP, ITCI,
Centrul de calcul Constanța
 17. Dan Teodorescu
 18. Th. Boranglu, R Dobrescu,
A. Hossu, S. Molin
 19. N. Patrubani
- Limbaje de programare pentru sisteme în
real, 224 pag., 23 lei
- Sisteme ierarhizate în timp real, cu pre-
lucrare distribuită a datelor, 296 pag., 29 lei
- LISP, 2. vol., 64 lei.
- Microprocesoare, microcalculatoare și ro-
boți în automatizări industriale, 284 pag.,
28 lei
- Calitate și fiabilitate, manual practic, 2 vol.,
1100 pag. (cu 300 aplicații, exemple, studii,
de caz, 4 rigle de calcul, 5 standarde,
2 000 teste-intrebări rezolvate) 139 lei.
- Inginerie de sistem, automatizări și in-
formatică în transporturi feroviare, na-
vale, aeriene, rutiere, vol. 1, 758 pag.
75 lei, vol. 2, 1000 pag., 99 lei.
- Automatică, management, calculatoare
(AMC) volumele 52-55, 2000 pag.,
180 lei (a se căuta și volumele ante-
rioare, în curs de epuizare).
- Echipamente periferice vol. 3, 260 pag.,
19 lei.
- Practica bazelor de date. Totul despre...
SOCRATE și SOCRATE-MINI pe Felix C,
CORAL, INDEPENDENT. Volumele 1 și 2,
768, pagini, Seria Practică, Lei 62.
- Automatizări microelectronice, 256 pag.,
20 lei.
- Conducerea multiprocesor în timp real
a structurilor flexibile de fabricație
- Totul despre... microprocesorul Z80, vol. 1
și vol. 2, 700 pagini, (o parte a tirajului
însoțită de o casetă pentru un simulator
al funcționării microprocesorului pe calcu-
latoarele personale PRAE și aMIC), 65 lei
fără casetă, cu casetă 140 lei.

Vol. 1 + Vol. 2, Lei 56

● Grafica asistată, indisolubil legată de tehnica electronică de calcul, are numeroase aplicații în toate domeniile economiei și va deveni — în scurt timp — indispensabilă tot mai multor procese de producție, proiectare-cercetare, învățămînt.

● Cartea de față atacă, fundamentat științific, subiecte esențiale: reprezentările geometrice, pentru care — prin cantitatea și calitatea algoritmilor, rutinelor și programelor — devine un manual practic, un instrument de lucru.

● Totalitatea programelor sursă din carte constituie un „pachet de programe” inclus, după testări și experimentări, în Biblioteca Națională de Programe (ITCI), la solicitarea — în calitate de elaboratori — a redacției de calculatoare — management — informatică a editurii și a autorilor lucrării.



EDITURA TEHNICĂ

ISBN 973-31-0017-X
ISBN 973-31-0016-1