T78-11428 NHF

JSC-14246

8.0-10262
NASA CR-160712

"AS-BUILT" DESIGN SPECIFICATION
FOR
THE PATTERSON-PITT-THADANI
MINIMUM LOSS CLASSIFIER

Job Order 71-593

TIRF 77-0073

Prepared By
Lockheed Electronics Company, Inc.
Systems and Services Division
Houston, Texas

Contract NAS 9-15200

For

EARTH OBSERVATIONS DIVISION

SCIENCE AND APPLICATIONS DIRECTORATE

(E80-10262) AS-BUILT DESIGN SPECIFICATION FOR THE PATTERSON-PITT-THADANI MINIMUM LOSS CLASSIFIER (Lockheed Electronics Co.) 56 p HC A04/MF A01 CSCL 05B

N80-30845

Unclas

G3/43 00262

National Aeronautics and Space Administration

LYNDON B. JOHNSON SPACE CENTER

Houston, Texas

May 1978

LEC-12285

JSC- 14246

"AS-BUILT" DESIGN SPECIFICATION
FOR
THE PATTERSON-PITT-THADANI
MINIMUM LOSS CLASSIFIER

Job Order 71-593

TIRF 77-0073

Prepared By

C. W. Ahlers

APPROVED BY

for James A Wilkinson

Philip L. Krumm, Acting Supervisor
Scientific Applications Section

Prepared By

Lockheed Electronics Company, Inc.

For

Earth Observations Division

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

May 1978 LEC- 12285

CONTENTS

1. SCOPE

This document constitutes an "As-Built" Design Specification for the software conversion of the Patterson-Pitt-Thadani Minimum Loss Trainer and Classifier. This program has been implemented on the Purdue-LARS 370/148 computer system as a stand-alone classifier. It was converted from the UNIVAC EXEC 2 system at NASA/JSC.

In addition to the conversion, several enhancements were built into the program. These include the following:

- Both interactive and batch versions are available.
- All floating point computations are done in double precision for increased accuracy.
- Some inputs have default values provided.
- Program organization has been improved.

2. APPLICABLE DOCUMENTS

1. Patterson, J. D.; Pitt, J. M.; and Womack, B. F.: A Sequentialization of the Patterson Classifier. IEE, vol. 54, Dec. 1966, pp. 1987-1988.

2. Aizerman, M. A.; Braverman, E. M.; and Rozonoer, L. I.: The Probability Problem of Pattern Recognition Learning and the Method of Potential Functions. Automation and Remote Control, vol. 25, Mar. 1965, pp. 1175-1190.

3. Blaydon, C. C.; et al.: Recursive Algorithms for Pattern Classification. Technical Report 520, Division of Engineering and Applied Physics, Harvard University (Cambridge, Mass.), Mar. 1967.

4. Wagner, J. J.; Pitt, J. M.; and Womach, B. F.: A Comparison Between Pattern Classification Approaches. IEEE Trans. on Information Theory, Oct. 1967.

5. Nilsson, Nils J.: Learning Machines. McGraw-Hill, 1965, pp.

6. A Non parametric Loss-Optimal Pattern Classification System, February 1978, Job Order 73-743, LEC-11451, Contract NAS 9-15200.

7. TIRF 77-0073 Minimum-Loss Classifier February 2, 1978.

3. SYSTEM DESCRIPTION

3.1 HARDWARE DESCRIPTION

N/A

3.2 SOFTWARE DESCRIPTION

In general, this system is composed of two principal programs. The function of the first program (MPPTA or MPPTAI) is to compute a loss vector matrix using the input data. The second program (MPPTC or MPPTCI) uses the loss vector matrix computed by the first and input data to classify the input data into one of two classes.

The structure of both programs is: A driver, an input subroutine and a computational subroutine. There are separate drivers for the batch and interactive versions of the program, as well as separate input routines, but the computational subroutine is used by both versions.

All floating point calculations have been made to be double precision, thus increasing accuracy.

3.2.1 SOFTWARE COMPONENT NO.1 (MPPTA)

The program MPPTA is the main driver program for the batch version of the first processor. This processor writes a loss vector matrix out to unit no. 7, to be used by the second processor.

3.2.1.1 Linkages

The program MPPTA calls subroutines SPPTA, PPTA, CLOCK, GETIME, GTDATE, and IDNAME. The subroutine PPTA in turn calls READIT, NP, and PHI. The subroutines CLOCK, GETIME, GTDATE, and IDNAME are "system subroutines" and descriptive by name.

3.2.1.2 Interfaces

MPPTA interfaces with other routines through calling sequences, and common blocks UN and FV. The common blocks are initialized in PPTBLK.

3.2.1.3 Inputs

All input to MPPTA comes from subroutines called by it.

3.2.1.4 Outputs

Output to the printout from MPPTA are: date, time, user name, user I.D., and C.P.U. time.

3.2.1.5 Storage

Program size = 398694.

3.2.1.6 Description

The program MPPTA is the first of two processors used in sequence to classify input data using the Patterson-Pitt-Thadani algorithm for minimum loss classification. MPPTA writes a loss vector matrix to a disk data set to be used by the second processor.

3.2.1.7 Flowchart

N/A

3.2.1.8 Listing

FILE MPPTA

```
            C   ADAPTED BY C W AHLERS
            C  THIS PROGRAM (MPPTA) USES THE FOLLOWING SUBROUTINES
            C -SPPTA
            C  PPTA
            C  READIT
            C  GP
            C  PHI
            C THE PATTERSON-PITT-THADANI ALGORITHM.
            C THIS PROGRAM USES UNITS RUNIT AND WUNIT FOR SCRATCH WORK.
            C THE FINAL LOSS VECTOR MATRIX A IS OUTPUT TO UNIT   WUNIT.
            C TRAINING.
            C P(N1)....PHI FUNCTION VECTOR.
            C Q(N1,T)....CLASS PHI SUM MATRIX.
            C R(N1)....PN INVERSE * PHI FUNCTION VECTOR.
            C PNI(N2)....PN INVERSE MATRIX.
0001              INTEGER D,T,CAI,RUNIT,WUNIT
            C     PARAMETER NN1=300,NN2=40000,TT=10,DD=30
0002               COMMON /UN/NRDR1,NRDR2,NPRT,RUNIT,WUNIT
0003              INTEGER FEATVC
0004              COMMON /FV/FEATVC(30),IFMT(20),NDATA
0005              DOUBLE PRECISION PNI(40000),P(300),Q(300,10),R(300),S(300,10)
0006              DOUBLE PRECISION A(300,10),ALPHA,TRACE
0007              DOUBLE PRECISION C(10,10),X(30)
0008               INTEGER USERID(2),NAME(4),TIME(3),DATE(3)
0009              TIKTOK=0.
0010              CALL CLOCK(TIKTOK)
0011              WRITE(NPRT,100)
0012         100  FORMAT(1H1,10X,'THE PATTERSON-PITT-THADANI ALGORITHM PROGRAM')
0013               CALL GETIME(TIME)
0014               CALL GTDATE(DATE)
0015               CALL IDNAME(USERID,NAME)
0016               WRITE(NPRT,200)USERID,NAME,DATE,TIME
0017         200   FORMAT(//,10X,2A4,4X,4A4,4X,3A4,4X,3A4)
0018              CALL SPPTA(D,T,ISGZ,NT,E,C,INDEX,N1,N2)
0019              CALL PPTA(D,T,ISGZ,NT,N1,N2,PNI,P,Q,R,S,A,E,INDEX,C,X)
0020              CALL CLOCK(TIKTOK)
0021              WRITE(NPRT,300) TIKTOK
0022         300   FORMAT(//,10X,'TIME FOR PPTA',F10.3)
0023              STOP
0024              END
```

3.2.2 SOFTWARE COMPONENT NO. 2 (SPPTA)

Subroutine SPPTA reads the input cards and sets option switches for the first processor.

3.2.2.1 Linkages

SPPTA is called by the program MPPTA and uses data initialized in PPTBLK.

3.2.2.2 Interfaces

SPPTA interfaces with MPPTA through a calling sequence and interfaces with MPPTA and PPTBLK through common blocks UN, PF, and FV.

3.2.2.3 Inputs

Calling sequence: Subr. SPPTA(D,T,ISGZ,NT,E,C,INDEX,N1,N2)

Parameter	Dimension	In/Out	Description
D	1	Out	No. of channels
T	1	Out	No. of classes
ISGZ	1	Out	No. of small grain pixels
NT	1	Out	Total no. of samples
E	1	Out	Error Tolerance
C	(10,10)	Out	Cost Matrix
INDEX	1	Out	Index which determines the feature whose interactions with other features are to be ignored.
N1	1	Out	A number that determines certain array sizes

Parameter	Dimension	In/Out	Description
N2	1	Out	A number that determines certain array sizes

Common Blocks:

See PPTBLK for information about the common blocks.

Input cards (unit NRDR1):

	Variables	Format	Function
1.	PFLAG	I5	0- for short printout 1- for printout
2.	DT,ISGZ,NT	4I5	D- no. of channels T- no. of classes (at present T=2) ISGZ- no. of small grain pixels NT- Total no. of samples
3.	E	F10.7	Error tolerance
4.	((C(I,J),J=1,T), I=1,T)	10F5.2	The cost matrix
5.	INDEX	I5	Interaction index
6.	IDEF	A1	Y- use default data vector input format N- input an input format
7.	(use if IDEF=N) NDATA	I5	Number of data points per pixel
8.	(use if IDEF=N) (IFMT(I),I=1,20)	20A4	Input format
9.	IDEF	A1	Y- use default feature index vector N- input a feature index vect

	Variables	Format	Function
10.	(Use if IDEF=N) (FEATVC(I), I=1,D)	30I2	The feature index vector.

3.2.2.4 Outputs

Input information is printed out.

3.2.2.5 Storage

Program size = 2694.

3.2.2.6 Description

SPPTA is the input subroutine for all except the pixel data.

If default options are not used this subroutine inputs the format for the pixel data and the feature index vector.

3.2.2.7 Flowchart

N/A

3.2.2.8 Listing

FILE SPP1A

```
C     WRITTEN BY C [illegible]
      SUBROUTINE SPP1A(D,T,ISSZ,NT,F,C,INDEX,N1,N2)
      INTEGER D,T,C[illegible],RUNIT,WUNIT
      COMMON /[illegible]/ RDR1,RDR2,RPRT,RUNIT,WUNIT
      INTEGER PFLAG
      COMMON /PP/PFLAG
      INTEGER FEATVC,TT,DD
      COMMON /FV/FEATVC(30),IFMT(20),NDATA
C     PARAMETER ND1=300,N2=40000,TT=10,DD=30
C DIMENSIONS CHECKED IN THIS SUBROUTINE
      DATA N1/300/,N2/40000/,TT/10/,DD/30/
      DATA IY/'Y'/
      DOUBLE PRECISION C(10,10)
C READ INPUT DATA.
      READ(RDR1,10) PFLAG
 10   FORMAT(I1)
      WRITE(RPRT,837) PFLAG
 837  FORMAT(//,10X,'PFLAG=',I5)
      WRITE(RPRT,20) RUNIT,WUNIT
 20   FORMAT(//,10X,'RUNIT=',I5,5X,'WUNIT=',I5)
      READ(RDR1,21) D,T,ISSZ,NT
 21   FORMAT(4I5)
      WRITE(RPRT,22) D,T,ISSZ,NT
 22   FORMAT(//,10X,'NO. OF CHANNELS=',I5,//,10X,'NO. OF CLASSES=',
     *   I5,//,10X,'NO. OF SMALL GRAIN PIXELS=',I5,//,10X,
     *   'TOTAL NO. OF SAMPLES=',I5)
      IF(D.GT.DD) WRITE(RPRT,101) DD
      IF(T.GT.TT) WRITE(RPRT,102) TT
 101  FORMAT(//,10X,' ERROR -- THE NO. OF CHANNELS EXCEEDS ',I5)
 102  FORMAT(//,10X,' ERROR -- THE NO. OF CLASSES EXCEEDS ',I5)
      IF(D.GT.DD) STOP
      IF(T.GT.TT) STOP
      READ(RDR1,1) F
    1 FORMAT(F10.7)
      WRITE(RPRT,15) F
 15   FORMAT(//,10X,'F = ',F10.7)
      READ(RDR1,2) ((C(I,J),J=1,T),I=1,T)
    2 FORMAT(10F5.2)
      WRITE(RPRT,35)
 35   FORMAT(//,10X,'THE COST MATRIX')
      DO 150 I=1,T
      WRITE(RPRT,25) (C(I,J),J=1,T)
 150  CONTINUE
 25   FORMAT(//,10X,10F5.2)
C READ INDEX OF FEATURE WHOSE INTERACTIONS WITH
C THE OTHER FEATURES ARE TO BE IGNORED.
      READ(RDR1,778) INDEX
 778  FORMAT(I5)
      WRITE(RPRT,700) INDEX
 700  FORMAT(//,10X,'INDEX=',I5)
      READ(RDR1,506) IOFF
 506  FORMAT(A1)
      IF(IOFF.EQ.IY) GO TO 507
C     READ IN NO. OF DATA POINTS PER PIXEL
      READ (RDR1,21) NDATA
 444  FORMAT(//,10X,'NDATA=',I5)
C     READ IN THE FORMAT FOR THE FEATURE VECTOR
      READ(RDR1,450) IFMT
 450  FORMAT(20A4)
 507  CONTINUE
      WRITE(RPRT,444) NDATA
      WRITE(RPRT,451) IFMT
 451  FORMAT(//,10X,'FEATURE VECTOR FORMAT ',//,10X,20A4)
      READ(RDR1,506) IOFF
      IF(IOFF.EQ.IY) GO TO 509
C     READ IN THE FEATURE INDEX VECTOR
      READ(RDR1,452) (FEATVC(I),I=1,D)
 452  FORMAT(30I2)
 509  CONTINUE
      WRITE(RPRT,453) (FEATVC(I),I=1,D)
 453  FORMAT(//,10X,'THE FEATURE INDEX VECTOR ',//,10X,30I3)
      DO 300 I=1,D
      IF(FEATVC(I).[illegible]) WRITE(RPRT,301)
```

ORIGINAL PAGE IS OF POOR QUALITY

FILE SPPTA

```
                  ** FEATURE INDEX VECTOR IS WRONG')                       SPP00770
0066              N1=N*1+(N*(N+1))/2                                        SPP00780
0067              NN2=N1+(N1*(N1+1))/2                                      SPP00790
0068              WRITE(NPRT,33) N1,N2                                      SPP00800
0069        33    FORMAT(//,10X,'N1=',I5,5X,'N2=',I5)                       SPP00810
0070               IF(N1.GT.NN1) WRITE(NPRT,103) NN1                        SPP00820
0071               IF(N2.GT.NN2) WRITE(NPRT,104) NN2                        SPP00830
0072               IF(N2.NE.NN2) N2=NN2                                     SPP00840
0073              IF(N1.GT.NN1) STOP                                        SPP00850
0074              IF(N2.GT.NN2) STOP                                        SPP00860
0075        103    FORMAT(//,10X,' ERROR -- N1 EXCEEDS ',I10)               SPP00870
           C104    FORMAT(//,10X,' ERROR -- N2 EXCEEDS ',I10)               SPP00880
0076        106    FORMAT(//,10X,' N2 REPLACED BY ',I10)                    SPP00890
0077              RETURN                                                    SPP00900
0078              END                                                       SPP00910
```

3.2.3 SOFTWARE COMPONENT NO. 3 (PPTA)

Subroutine PPTA is the main computational subroutine of the first processor. Input from SPPTA or SPPTAI is passed to PPTA. PPTA with the aid of other subroutines calculates the loss vector matrix and writes it out to unit WUNIT.

3.2.3.1 Linkages

Subroutine PPTA is called by MPPTA or MPPTAI and is passed information from SPPTA or SPPTAI. PPTA calls subroutines READIT, PHI, and NP.

3.2.3.2 Interfaces

PPTA interfaces with other routines through a calling sequence and common blocks UN and PF.

3.2.3.3 Inputs

Calling sequence:

Subr. PPTA(D,T,ISGZ,NT,N1,N2,PNI,P,Q,R,S,A,E,INDEX,C,X)

Parameter	Dimension	In/Out	Description
D	1	In	No. of channels
T	1	In	No. of classes
ISGZ	1	In	No. of small grain pixels
NT	1	In	Total no. of samples
N1	1	In	Dimension for some arrays
N2	1	In	Dimension for some arrays
PNI	N2	In	PN inverse
P	N1	In	Phi function vector
Q	(N1,T)	In	Class phi sum matrix

Parameter	Dimension	In/Out	Description
R	N1	In	PNI*P
S	(N1,T)	In	Working storage
A	(N1,T)	Out	The loss vector matrix
E	1	In	Error Tolerance
INDEX	1	In	Interaction index.
C	(10,10)	In	Cost matrix.
X	D	In	The feature vector.

Common Blocks:

See PPTBLK for information about the common blocks.

3.2.3.4 Outputs

The loss vector matrix is printed out and written to uni. UNIT. Optional information is printed out if PFLAG=1.

3.2.3.5 Storage

Program size = 6184.

3.2.3.6 Description

PPTA uses the input of SPPTA or SPPTAI and READIT as principle input to compute the loss vector matrix and write it to unit WUNIT.

3.2.3.7 Flowchart

N/A

3.2.3.8 Listing

FILE PPTA

```
           C   ADAPTED BY C W AHLERS
           C THE PATTERSON-PITT-THADANI ALGORITHM.
           C THIS PROGRAM USES UNITS RUNIT AND WUNIT FOR SCRATCH WORK.
           C THE FINAL LOSS VECTOR MATRIX A IS OUTPUT TO UNIT   WUNIT.
           C TRAINING.
           C P(N1)....PHI FUNCTION VECTOR.
           C Q(N1,T)....CLASS PHI SUM MATRIX.
           C R(N1)....PN INVERSE * PHI FUNCTION VECTOR.
           C PNI(N2)....PN INVERSE MATRIX.
0001             SUBROUTINE PPTA(D,T,ISGZ,NT,N1,N2,PNI,P,Q,R,S,A,E,
                *  INDEX,C,X)
0002             INTEGER D.T.CAT.RUNIT.WUNIT
0003              COMMON /UN/NRDR1,NRDR2.NPRT.RUNIT,WUNIT
0004             INTEGER PFLAG
0005             COMMON /PF/PFLAG
0006             DOUBLE PRECISION PNI(N2).P(N1).Q(N1,T),R(N1),S(N1,T)
0007             DOUBLE PRECISION A(N1,T).ALPHA,TRACE
0008             DOUBLE PRECISION C(10,10),X(D)
0009             IOUT=RUNIT
           C     WRITE(NPRT.999) D.T,ISGZ,NT.RUNIT,WUNIT,N1,N2,E.INDEX,C
0010        999   FORMAT(//,2X.8I5,F10.7.I5,10(/,10F5.2))
           C INITIALIZE PNI.P.Q,R,A
0011             DO 3 I=1.N2
0012             PNI(N2)=0.0D0
0013           3 CONTINUE
0014             DO 4 I=1.N1
0015             DO 4 J=1,T
0016             Q(I.J)=0.0D0
0017             A(I,J)=0.0D0
0018             S(I.J)=0.0D0
0019           4 CONTINUE
0020             DO 5 I=1.N1
0021             P(I)=0.0D0
0022             R(I)=0.0D0
0023           5 CONTINUE
           C COMPUTE NO. OF PHI FUNCTIONS.
0024             M=D+D+((D*(D-1))/2)+1
           C     M1=(N2/M)
0025             ARG=1+8*N2
0026             M1=(SQRT(ARG) - 1)/2
0027             WRITE(NPRT.800) M1.M
0028         800 FORMAT(/1X.'M1=DISC I/O RATE...LINES/ACCESS',I10,
                1/.1X.'PN INVERSE IS M BY M.........M=',I10)
           C M1= NO. OF LINES OF PN INVERSE THAT CAN BE STORED IN PNI.
           C COMPUTE P0I....WRITE P0I TO DISC IFF M1>M.
0029             IF(PFLAG.EQ.1) WRITE(NPRT,937)
0030        937   FORMAT(//.10X,'THE INPUT DATA',//)
           C     DO 100 L=1.M
0031             K1=1
0032             K2=M1
0033             K3=M1
0034             IF(M1 .GE. M) K2=M
0035           9 DO 6 I=K1.K2
           C     ID=((I-K1)*M)+I
0036             ID=NP(I,I,M)
           C     WRITE(NPRT,806) ID
0037         806 FORMAT(/1X,'ID...P0 INVERSE LOOP ',I10)
           C     WRITE(NPRT.803) ID
0038         803 FORMAT(/1X,'ID IN P0 INVERSE LOOP =',I10)
0039             PNI(ID)=1.0D0/E
0040           6 CONTINUE
0041             IF(M1 .GE. M) GO TO 100
           C     WRITE(NPRT.35) K1,K2.K3
0042          35 FORMAT(/'K1,K2,K3 ','I/O   NO. 1',3I6)
0043             K4=K3*M
0044             WRITE(RUNIT,7) (PNI(I),I=1,K4)
0045           7 FORMAT(4D20.10)
0046             IF(K2 .EQ. M) GO TO 100
0047             K1=K2+1
0048             K2=K2+M1
0049             IF(K2 .LE. M) GO TO 9
0050             K2=M
0051             K3=K2-K1+1
0052             [illegible]
```

FILE PPTA

```
0054          8 ENDFILE RUNIT                                          PPT00770
0055            DO 10 KOUNT=1,NT                                       PPT00780
0056      1000I FORMAT(1X,'KOUNT',6X,I10)                              PPT00790
0057            CALL READIT(X,D)                                       PPT00800
0058            IF(PFLAG.EQ.1) WRITE(NPRT,801) (X(I),I=1,D)            PPT00810
0059       801  FORMAT(1H ,2X,'X',2X,15F6.1)                           PPT00820
0060            IF(KOUNT .GT. ISGZ) GO TO 11                           PPT00830
0061            CAT=1                                                  PPT00840
0062            GO TO 12                                               PPT00850
0063         11 CAT=2                                                  PPT00860
        C COMPUTE PHI FUNCTION VECTOR.                                 PPT00870
0064         12 CALL PHI(X,P,D,N1,INDEX)                               PPT00880
        C       WRITE(NPRT,802) (P(I),I=1,M)                           PPT00890
0065        802 FORMAT(/1X,'PHI VECTOR',/,(1X,3D20.10))                PPT00900
        C UPDATE PHI SUM MATRIX Q.                                     PPT00910
0066            DO 13 I=1,M                                            PPT00920
0067            A(I,CAT)=A(I,CAT)+P(I)                                 PPT00930
0068         13 CONTINUE                                               PPT00940
        C BEGIN PN INVERSE COMPUTATIONS.                               PPT00950
        C DISC I/O REQUIRED VIA UNITS RUNIT AND WUNIT                  PPT00960
        C IFF M1  <  M.                                                PPT00970
        C COMPUTE PN INVERSE * PHI.                                    PPT00980
        C       ENDFILE RUNIT                                          PPT00990
        C       ENDFILE WUNIT                                          PPT01000
0069            REWIND RUNIT                                           PPT01010
0070            REWIND WUNIT                                           PPT01020
        C       DO 101 L=1,M                                           PPT01030
0071            K1=1                                                   PPT01040
0072            K2=M1                                                  PPT01050
0073            K3=M1                                                  PPT01060
0074            IF(M1 .GE. M) K2=M                                     PPT01070
        C    17 WRITE(NPRT,36) K1,K2,K3                                PPT01080
0075         36 FORMAT(/2X,'K1,K2,K3  I/O   NO. 2  ',3I6)              PPT01090
0076            IF(M1 .GE. M) GO TO 140                                PPT01100
0077         17 K4=K3*M                                                PPT01110
0078            READ(RUNIT,7) (PNI(I),I=1,K4)                          PPT01120
0079        140 DO 14 I=K1,K2                                          PPT01130
0080            R(I)=0.0D0                                             PPT01140
0081            DO 15 J=1,M                                            PPT01150
        C       IC=((I-K1)*M)+J                                        PPT01160
0082            IC=NP(I,J,M)                                           PPT01170
        C       WRITE(NPRT,804) I,J,IC                                 PPT01180
0083        804 FORMAT(/1X,'I,J,IC...PNI*PHI...140',3I5)               PPT01190
0084            R(I)=R(I)+PNI(IC)*P(J)                                 PPT01200
0085         15 CONTINUE                                               PPT01210
0086         14 CONTINUE                                               PPT01220
0087            IF(M1 .GE. M) GO TO 101                                PPT01230
0088            WRITE(NPRT,37) K1,K2,K3                                PPT01240
0089         37 FORMAT(/2X,'K1,K2,K3  I/O   NO. 21  ',3I5)             PPT01250
0090            IF(K2 .EQ. M) GO TO 101                                PPT01260
        C       WRITE(NPRT,38) K1,K2,K3                                PPT01270
0091         38 FORMAT(/2X,'K1,K2,K3  I/O   NO. 22  ',3I6)             PPT01280
0092            K1=K2+1                                                PPT01290
0093            K2=K2+M1                                               PPT01300
0094            IF(K2 .LE. M) GO TO 17                                 PPT01310
0095            K2=M                                                   PPT01320
0096            K3=K2-K1+1                                             PPT01330
        C       WRITE(NPRT,39) K1,K2,K3                                PPT01340
0097         39 FORMAT(/2X,'K1,K2,K3  I/O   NO. 23  ',3I6)             PPT01350
0098            GO TO 17                                               PPT01360
0099        101 CONTINUE                                               PPT01370
        C UPDATE PN INVERSE.                                           PPT01380
        C IF M1  <  M, DISC I/O TO UNITS RUNIT AND WUNIT ALTERNATELY.  PPT01390
        C COMPUTE PHI' * PN INVERSE * PHI.                             PPT01400
0100         16 ALPHA=1.0D0                                            PPT01410
        C       WRITE(NPRT,40)                                         PPT01420
0101         40 FORMAT(/2X,'CAME TO 16'/)                              PPT01430
0102            DO 18 I=1,M                                            PPT01440
0103            ALPHA=ALPHA+P(I)*R(I)                                  PPT01450
0104         18 CONTINUE                                               PPT01460
        C       WRITE(NPRT,807) ALPHA                                  PPT01470
0105        807 FORMAT(/1X,'ALPHA ',D20.10)                            PPT01480
        C UPDATE PN INVERSE.                                           PPT01490
```

FILE PPTA

```
 0107              REWIND WUNIT
            C      DO 102 L=1,M
 0108              K1=1
 0109              K2=M1
 0110              K3=M1
 0111              IF(M1 .GE. M) K2=M
 0112              IF(M1 .GE. M) GO TO 190
 0113           22 K4=K3*M
 0114              READ(RUNIT,7) (PNI(I),I=1,K4)
            C      WRITE(NPRT,409) K1,K2,K3
 0115          409 FORMAT(/2X,'K1,K2,K3  I/O   NO. 3  ',3I6)
 0116          190 DO 19 I=K1,K2
 0117              DO 20 J=1,M
            C      IPN=((I-K1)*M)+J
 0118              IF(I .GT. J) GO TO 20
 0119              IPN=NP(I,J,M)
 0120              PNI(IPN)=PNI(IPN)-((R(I)*R(J))/ALPHA)
 0121           20 CONTINUE
 0122           19 CONTINUE
 0123              IF(M1 .GE. M) GO TO 102
 0124              WRITE(WUNIT,7) (PNI(I),I=1,K4)
 0125              IF(M1 .GE. M) GO TO 102
 0126              IF(K2 .EQ. M) GO TO 102
 0127              K1=K2+1
 0128              K2=K2+M1
 0129              IF(K2 .LE. M) GO TO 22
 0130              K2=M
 0131              K3=K2-K1+1
 0132              GO TO 22
 0133          102 CONTINUE
            C NEXT SAMPLE...SWITCH UNITS.
 0134           21 IUNIT=RUNIT
            C      WRITE(NPRT,41)
 0135           41 FORMAT(/2X,'CAME TO 21')
 0136              ENDFILE WUNIT
 0137              RUNIT=WUNIT
 0138              WUNIT=IUNIT
            C      ENDFILE RUNIT
            C      ENDFILE WUNIT
 0139              REWIND RUNIT
 0140              REWIND WUNIT
 0141              IK=(KOUNT/5)*5
 0142              IF(KOUNT .EQ. IK.AND.PFLAG.EQ.1) WRITE(NPRT,10001) KOUNT
 0143           10 CONTINUE
            C PN INVERSE IS NOW SITTING ON RUNIT.
            C COMPUTE LOSS VECTOR MATRIX A.
            C A(J)=(PN INVERSE)*(C(J/1)*Q(1)+...+C(J/T))*Q(T).
            C COMPUTE (PN INVERSE) * Q.
            C      ENDFILE RUNIT
            C      ENDFILE WUNIT
 0144              REWIND RUNIT
 0145              REWIND WUNIT
 0146              DO 50 I=1,T
 0147              DO 60 J=1,T
 0148              DO 70 K=1,M
 0149              S(K,I)=S(K,I) + C(I,J)*A(K,J)
 0150           70 CONTINUE
 0151           60 CONTINUE
 0152           50 CONTINUE
            C      DO 103 L=1,M
 0153              K1=1
 0154              K2=M1
 0155              K3=M1
 0156              IF(M1 .GE. M) K2=M
 0157              IF(M1 .GE. M) GO TO 230
 0158           27 K4=K3*M
 0159              READ(RUNIT,7) (PNI(I),I=1,K4)
 0160          230 DO 23 I=1,T
 0161              DO 24 J=K1,K2
 0162              DO 25 K=1,M
            C      IE=((J-K1)*M)+K
 0163              IE=NP(J,K,M)
 0164              Q(J,I)=Q(J,I)+PNI(IE)*A(K,I)
```

FILE PPTA

```
0168            IF(M1 .GE. M) GO TO 103
0169            IF(K2 .EQ. M) GO TO 103
0170            K1=K2+1
0171            K2=K2+M1
0172            IF(K2 .LE. M) GO TO 27
0173            K2=M
0174            K3=K2-K1+1
0175            GO TO 27
0176        103 CONTINUE
          C COMPUTE A.
0177         26 DO 28 I=1,T
0178            DO 29 K=1,M
0179            A(K,I)=0.0D0
0180            DO 30 J=1,T
0181            A(K,I)=A(K,I)+C(I,J)*Q(K,J)
0182         30 CONTINUE
0183         29 CONTINUE
0184         28 CONTINUE
0185            TRACE=0.0D0
0186            DO 80 K=1,T
0187            DO 90 I=1,M
0188            TRACE=TRACE + A(I,K)*S(I,K)
0189         90 CONTINUE
0190         80 CONTINUE
0191            WUNIT=IOUT
0192            TRACE=(TRACE/NT)
          C WRITE M.D.T. LOSS VECTOR MATRIX A TO WUNIT.
          C     ENDFILE WUNIT
0193            REWIND WUNIT
0194            WRITE(WUNIT,31) M,D,T
0195         31 FORMAT(3I3)
0196            WRITE(WUNIT,32) ((A(I,J),J=1,T),I=1,M)
0197            IF(PFLAG.EQ.1) WRITE(NPRT,237)
0198      237    FORMAT(//,10X,'PN INVERSE',//)
0199            NINV=N2-N1
0200            IF(PFLAG.EQ.1) WRITE(NPRT,238) (PNI(I),I=1,NINV)
0201      238    FORMAT(1H ,5X,3D20.10)
0202            WRITE(NPRT,555)
0203      555    FORMAT(//,10X,'THE LOSS VECTOR MATRIX',/)
0204            WRITE(NPRT,332)((A(I,J),J=1,T),I=1,M)
0205         32 FORMAT(2D20.10)
0206        332 FORMAT(1H ,2D20.10)
0207            ENDFILE WUNIT
          C TRAINING OVER.
          C CLASSIFICATION PROGRAM WILL READ LOSS VECTOR MATRIX A
          C FROM UNIT WUNIT.
0208            WRITE(NPRT,33) WUNIT
0209         33 FORMAT(/1X,'TRAINING OVER',/,
               11X,'LOSS VECTOR MATRIX RESIDES ON UNIT ',I8/)
0210            WRITE(NPRT,34) M,D,T
0211         34 FORMAT(/1X,'M,D,T',3I8/)
0212            WRITE(NPRT,110) TRACE
0213        110 FORMAT(/1X,'UPPER BOUND ON BAYES RISK',6X,D20.10/)
0214            RETURN
0215            END
```

3.2.4 SOFTWARE COMPONENT NO. 4 (READIT)

Subroutine READIT reads in a vector of data about a pixel, using the input format IFMT, and stores it in the feature vector using the feature index vector.

3.2.4.1 Linkages

READIT is called by PPTA and PPTC.

3.2.4.2 Interfaces

READIT interfaces with PPTA and PPTC through a calling sequence and PPTBLK through the common blocks UN, PF, and FV. READIT reads data from unit NRDR2.

3.2.4.3 Inputs

Calling sequence:

Subr. READIT (X,ND)

Parameter	Dimension	In/Out	Description
X	ND	Out	The feature vector
ND	1	In	The number of channels

Common blocks:

COMMON/FV/FEATVC(30,IFMT(20),NDATA

Parameter	Dimension	In/Out	Description
FEATVC	30	In	The feature index vector
IFMT	2	In	The data input format
NDATA	1	In	Number of data points per pixel

See PPTBLK for information on the other common blocks.

Input cards (unit NRDRZ):

Variables	Format	Function
(XX(I),I=1,ND)	IFMT	Input data for a pixel.

3.2.4.4 Outputs

If PFLAG=1 then the vector XX is printed out.

3.2.4.5 Storage

Program size=888.

3.2.4.6 Description

READIT reads in a vector of data (length NDATA) about a pixel using the input format IFMT and stores it in the feature vector using the feature index vector as a set of pointers.

3.2.4.7 Flowchart

N/A

3.2.4.8 Listing

FILE READIT

```
      C   ADAPTED BY C W AHLERS
            SUBROUTINE READIT(X,ND)
      C THIS SUBROUTINE READS DATA FOR SUPER PAT-PIT-THAD.
            DOUBLE PRECISION X(ND),XX(30)
            INTEGER RUNIT,WUNIT
            COMMON /UN/NRDR1,NRDR2,NPRT,RUNIT,WUNIT
            INTEGER PFLAG
            COMMON /PF/PFLAG
            INTEGER FEATVC(30)
            COMMON /FV/FEATVC,IFMT(20),NDATA
            READ(NRDR2,IFMT) (XX(I),I=1,NDATA)
            IF(PFLAG.EQ.1) WRITE(NPRT,3)
        3   FORMAT(1H )
            IF(PFLAG.EQ.1) WRITE(NPRT,2) (XX(I),I=1,NDATA)
        2   FORMAT(1H ,2X,'XX ',15F6.1)
            DO 1 I=1,ND
        1   X(I)=XX(FEATVC(I))
            RETURN
            END
```

3.2.5 SOFTWARE COMPONENT NO. 5 (PHI)

Subroutine PHI computes the quadratic function vector.

3.2.5.1 Linkages

PHI is called by subroutines PPTA and PPTC.

3.2.5.2 Interfaces

PHI interfaces with other routines through a calling sequence.

3.2.5.3 Inputs

Calling sequence

Subr. PHI(X,P,D,NP,Z)

Parameter	Dimension	In/Out	Description
X	D	In	The feature vector.
P	NP	Out	The phi function vector.
D	1	In	Number of channels.
NP	1	In	Number of terms in the phi vector (N1).
Z	1	In	The interaction index.

3.2.5.4 Outputs

N/A

3.2.5.5 Storage

Program size=824.

3.2.5.6 Description

PHI computes the quadratic function vector. This vector consists of squared terms, cross product terms, first order terms, and one.

Cross product terms for the Zth feature are set to zero. If Z is zero all terms are used.

3.2.5.7 Flowchart

N/A

3.2.5.8 Listings

FILE PHI

```
      SUBROUTINE PHI(X,P,D,NP,Z)
C      THIS ROUTINE COMPUTES THE TERMS OF THE QUADRIC XT*A*X + BT*X + C.
C      OR (XI**2, XI*XJ, XI, 1).
C      THE ELEMENTS OF THE P VECTOR FOLLOW THE ABOVE ORDER.
C      Z IS THE INDEX OF THE FEATURE WHOSE INTERACTIONS
C      WITH THE REST OF THE FEATURES ARE TO BE IGNORED.
C      Z = 0 IMPLIES ALL INTERACTIONS ARE CONSIDERED.
      DOUBLE PRECISION P(NP)
      INTEGER D,Z
      DOUBLE PRECISION X(D)
      L = D
      DO 10 I=1,D
      P(I) = X(I)**2
      K = I + 1
      DO 10 J=K,D
      L = L + 1
      P(L) = X(I)*X(J)
      IF(I .EQ. Z .OR. J .EQ. Z) P(L)=0.0D0
   10 CONTINUE
      M = (D*(D-1)/2) + D
      DO 20 I=1,D
      M=M+1
      P(M) = X(I)
   20 CONTINUE
      M = M + 1
      P(M) = 1.0D0
      RETURN
      END
```

3.2.6 SOFTWARE COMPONENT NO. 6 (NP)

Function NP determines the pointer NP to an upper triangular array.

3.2.6.1 Linkages

The function NP is called by the subroutine PPTA.

3.2.6.2 Interfaces

NP interfaces with PPTA through a calling sequence and as a function subprogram.

3.2.6.3 Inputs

Calling sequence

Function. NP(I,J,M)

Parameter	Dimension	In/Out	Description
I	1	In	First rectangular coordinate
J	1	In	Second rectangular coordinate
M	1	In	The size of the PN matrix is M by M.

3.2.6.4 Outputs

N/A

3.2.6.5 Storage

Program size=514.

3.2.6.6 Description

Function NP determines the pointer NP (the function value) to an upper triangular array using the rectangular coordinates I and J.

3.2.6.7 Flowchart

N/A

3.2.6.8 Listings

FILE NP

```
0001            FUNCTION NP(I,J,M)
          C THIS SUBPROGRAM DETERMINES THE POINTER NP
          C TO AN UPPER TRIANGULAR ARRAY USING RECTANGULAR
          C COORDINATES I,J.
          C NP(I,J) IS CALLED BY THE PIT-PAT-THADANI PROGRAM.
0002            II=I
0003            JJ=J
0004            IF(II .GT. JJ) GO TO 1
0005          3 NP=(M*(II-1)) - (((II-1)*(II-2))/2) + (JJ-II+1)
0006            GO TO 2
0007          1 K=II
0008            II=JJ
0009            JJ=K
0010            GO TO 3
0011          2 RETURN
0012            END
```

3.2.7 SOFTWARE COMPONENT NO. 7 (PPTBLK)

PPTBLK is a block data subprogram. It is used to initialize several variables.

3.2.7.1 Linkages

N/A

3.2.7.2 Interfaces

PPTBLK interfaces with almost all the subprograms in this system through the common blocks FV, UN, and TUN.

3.2.7.3 Inputs

N/A

3.2.7.4 Outputs

N/A

3.2.7.5 Storage

Storage = $E4_{16}$ bytes.

3.2.7.6 Description

PPTBLK is a block data subprogram which initializes the common blocks FV, UN, and TUN.

Common blocks:

COMMON/FV/FEATVC(30),IFMT(20),NDATA

Parameter	Dimension	Description
FEATVC	30	The feature index vector.
IFMT	20	The input format for the input data (see READIT)

Parameter	Dimension	Description
NDATA	1	The number of data points per pixel.

The common block UN stores some of the various unit numbers as follows:

NRDR1 - Card reader for the setup cards or the terminal

NRDR2 - Card reader for the pixel data.

NPRT - Line printer (or output) unit number.

RUNIT - Utility data set unit number.

WUNIT - Utility data set unit number. (The loss vector is written to this unit)

The common block TUN stores only the terminal output unit number.

3.2.7.7 Flowchart

N/A

3.2.7.8 Listings

FILE PPTBLK

```
0001      BLOCK DATA                                                          PPT00010
0002      INTEGER FEATVC                                                      PPT00020
0003      INTEGER RUNIT,WUNIT                                                 PPT00030
0004      COMMON /FV/FEATVC(30),IFMT(20),NDATA                                PPT00040
0005      DATA NDATA/22/                                                      PPT00050
0006      DATA FEATVC/[illegible],[illegible],7,[illegible],9,10,11,12,13,10,11,12,5,21,11,9,8,6,2,15,4, PPT00060
         * [illegible]/                                                       PPT00070
0007      DATA IFMT/'(2X,','4F2.','1,2X','4(F','5.1,','F4.1',                 PPT00080
         * ',1X)','F4.','1,1X','F1.','0,1X','4F2','.0,1','X,4F','1.0)'/       PPT00090
0008      COMMON /DS/NRDR1,NRDR2,NPRT,RUNIT,WUNIT                             PPT00100
0009      DATA NRDR1/5/,NRDR2/10/,NPRT/6/                                     PPT00110
0010      DATA RUNIT/3/,WUNIT/7/                                              PPT00120
0011      COMMON /TON/NTRM                                                    PPT00130
0012      DATA NTRM/[illegible]/                                              PPT00140
0013      END                                                                 PPT00150
```

3.2.8 SOFTWARE COMPONENT NO. 8 (MPPTAI)

Program MPPTAI is the interactive version of MPPTA. The only difference is MPPTAI calls SPPTAI instead of SPPTA. For more detail see SOFTWARE COMPONENT NO. 1.

3.2.8.1 Listings

FILE MPPTAI

```
      C   ADAPTED BY C W AHLERS
      C  THIS PROGRAM (MPPTA) USES THE FOLLOWING SUBROUTINES
      C  SPPTA
      C  PPTA
      C  READIT
      C  NP
      C  PHI
      C THE PATTERSON-PITT-THADANI ALGORITHM.
      C THIS PROGRAM USES UNITS RUNIT AND WUNIT FOR SCRATCH WORK.
      C THE FINAL LOSS VECTOR MATRIX A IS OUTPUT TO UNIT   WUNIT.
      C TRAINING.
      C P(N1)....PHI FUNCTION VECTOR.
      C Q(N1,T)....CLASS PHI SUM MATRIX.
      C R(N1)....PN INVERSE * PHI FUNCTION VECTOR.
      C PNI(N2)....PN INVERSE MATRIX.
0001        INTEGER D,T,CAT,RUNIT,WUNIT
      C     PARAMETER NN1=300,NN2=40000,TT=10,DD=30
0002         COMMON /UN/NRDR1,NRDR2,NPRT,RUNIT,WUNIT
0003        INTEGER FEATVC
0004        COMMON /FV/FEATVC(30),IFMT(20),NDATA
0005        DOUBLE PRECISION PNI(40000),P(300),Q(300,10),R(300),S(300,10)
0006        DOUBLE PRECISION A(300,10),ALPHA,TRACE
0007        DOUBLE PRECISION C(10,10),X(30)
0008         INTEGER USERID(2),NAME(4),TIME(3),DATE(3)
0009        TIKTOK=0.
0010        CALL CLOCK(TIKTOK)
0011        WRITE(NPRT,100)
0012   100  FORMAT(1H1,10X,'THE PATTERSON-PITT-THADANI ALGORITHM PROGRAM')
0013         CALL GTTIME(TIME)
0014         CALL GTDATE(DATE)
0015         CALL IDNAME(USERID,NAME)
0016         WRITE(NPRT,200)USERID,NAME,DATE,TIME
0017   200   FORMAT(//,10X,2A4,4X,4A4,4X,3A4,4X,3A4)
0018        CALL SPPTAI(D,T,ISGZ,NT,E,C,INDEX,N1,N2)
0019        CALL PPTA(D,T,ISGZ,NT,N1,N2,PNI,P,Q,R,S,A,E,INDEX,C,X)
0020        CALL CLOCK(TIKTOK)
0021        WRITE(NPRT,300) TIKTOK
0022   300   FORMAT(//,10X,'TIME FOR PPTA',F10.3)
0023        STOP
0024        END
```

3.2.9 SOFTWARE COMPONENT NO. 9 (SPPTAI)

Subroutine SPPTAI is an interactive version of SPPTA. It prompts the user to input set up information.

3.2.9.1 Linkages

SPPTAI is called by the program MPPTAI and uses data initialized in PPTBLK.

3.2.9.2 Interfaces

SPPTAI interfaces with MPPTAI through a calling sequence and interfaces with MPPTAI and PPTBLK through common blocks UN, PF, FV and TUN.

3.2.9.3 Inputs

Calling sequence:

Subr. SPPTAI(D,T,ISGZ,NT,E,C,INDEX,N1,N2)

Parameter	Dimension	In/Out	Description
D	1	Out	No. of channels
T	1	Out	No. of classes
ISGZ	1	Out	No. of small grain pixels.
NT	1	Out	Total no. of samples.
E	1	Out	Error Tolerance
C	(10,10)	Out	Cost matrix
INDEX	1	Out	Index which determines the feature whose interactions with other features are to be ignored.
N1	1	Out	A number that determines certain array sizes.

Parameter	Dimension	In/Out	Description
N2	1	Out	A number that determines certain array sizes.

Common blocks:

See PPTBLK for information about the common blocks.

Input variables:

The user is prompted to input the setup variables from the terminal.

3.2.9.4 Outputs

Input information is printed out and sent to the terminal.

3.2.9.5 Storage

Program size=4404.

3.2.9.6 Description

SPPTAI is the interactive input subroutine for all except the pixel data. If default options are not used this subroutine inputs the format for the pixel data and the feature index vector. A long or short printout is an option.

3.2.9.7 Flowchart

3.2.9.8 Listing

FILE SPPTAI

```
      C     ADAPTED BY C R AHLERS
0001          SUBROUTINE SPPTAI(D,T,ISGZ,NT,F,C,INDEX,N1,N2)
0002          INTEGER D,T,CAT,RUNIT,WUNIT
0003           COMMON /UN/NRDR1,NRDR2,NPRT,RUNIT,WUNIT
0004          INTEGER PFLAG
0005          COMMON /PF/PFLAG
0006          INTEGER FEATVC,TT,DD
0007          COMMON /FV/FEATVC(30),IFMT(20),NDATA
      C       PARAMETER NN1=300,NN2=40000,TT=10,DD=30
      C DIMENSIONS CHECKED IN THIS SUBROUTINE
0008          DATA NN1/300/,NN2/40000/,TT/10/,DD/30/
0009          DATA IY/'Y'/
0010          DOUBLE PRECISION C(10,10)
0011          COMMON /TUN/NTRM
      C READ INPUT DATA.
0012   667    CONTINUE
0013          WRITE(NTRM,520)
0014   520    FORMAT(//,10X,'INPUT THE PRINT FLAG --0 OR 1--  I1')
0015          READ(NRDR1,110) PFLAG
0016    110     FORMAT(I1)
0017    10     FORMAT(3I5)
0018          WRITE(NTRM,837) PFLAG
0019    837    FORMAT(//,10X,'PFLAG=',I5)
0020            WRITE(NTRM,20) RUNIT,WUNIT
0021    20    FORMAT(//,10X,'RUNIT=',I5,5X,'WUNIT=',I5)
0022          WRITE(NTRM,502)
0023    502    FORMAT(//,10X,'INPUT NO. OF CHANNELS   I5')
0024    502    FORMAT(//,10X,'INPUT NO. OF CLASSES   I5')
0025    603    FORMAT(//,10X,'INPUT NO. OF SMALL GRAIN PIXELS   I5')
0026    604    FORMAT(//,10X,'INPUT NO. OF SAMPLES   I5')
0027          READ(NRDR1,21) D
0028    21    FORMAT(I5)
0029          WRITE(NTRM,502)
0030          READ(NRDR1,21) T
0031          WRITE(NTRM,603)
0032          READ(NRDR1,21) ISGZ
0033          WRITE(NTRM,604)
0034          READ(NRDR1,21) NT
0035          WRITE(NTRM,22) D,T,ISGZ,NT
0036    22     FORMAT(//,10X,'NO. OF CHANNELS=',I5,//,10X,'NO. OF CLASSES=',
            *   I5,//,10X,'NO. OF SMALL GRAIN PIXELS=',I5,//,10X,
            *   'TOTAL NO. OF SAMPLES=',I5)
0037           IF(D.GT.DD) WRITE(NTRM,101) DD
0038           IF(T.GT.TT) WRITE(NTRM,102) TT
0039    101      FORMAT(//,10X,' ERROR -- THE NO. OF CHANNELS EXCEEDS ',I5)
0040    102      FORMAT(//,10X,' ERROR -- THE NO. OF CLASSES EXCEEDS ',I5)
0041          IF(D.GT.DD) GO TO 667
0042          IF(T.GT.TT) GO TO 667
0043          WRITE(NTRM,503)
0044    503    FORMAT(//,10X,'INPUT THE ERROR TOLERANCE  E  F10.7')
0045          READ(NRDR1,1) E
0046        1 FORMAT(F10.7)
0047          WRITE(NTRM,15) E
0048    15    FORMAT(//,10X,'E = ',F10.7)
0049          WRITE(NTRM,504)
0050    504    FORMAT(//,10X,'INPUT THE COST MATRIX   F5.2')
0051          READ(NRDR1,2) ((C(I,J),J=1,T),I=1,T)
0052        2 FORMAT(F5.2)
0053          WRITE(NTRM,35)
0054    35     FORMAT(//,10X,'THE COST MATRIX')
0055          DO 150 I=1,T
0056          WRITE(NTRM,25) (C(I,J),J=1,T)
0057    150    CONTINUE
0058    25    FORMAT(//,10X,10F5.2)
      C READ INDEX OF FEATURE WHOSE INTERACTIONS WITH
      C THE OTHER FEATURES ARE TO BE IGNORED.
0059          WRITE(NTRM,777)
0060    777 FORMAT(/6X,'INPUT NON-INTERACTIVE FEATURE INDEX  I1'/)
0061          READ(NRDR1,778) INDEX
0062    778 FORMAT(I1)
0063          WRITE(NTRM,700) INDEX
0064    700    FORMAT(//,10X,'INDEX=',I5)
0065    666     CONTINUE
```

ORIGINAL PAGE IS OF POOR QUALITY

FILE SPPTAI

```
      506  FORMAT(A1)
           IF(IDEF.EQ.IY) GO TO 507
C  READ IN NO. OF DATA POINTS PER PIXEL
           READ (NRDR1,21) NDATA
      444  FORMAT(//,10X,'NDATA=',I5)
C    READ IN THE FORMAT FOR THE FEATURE VECTOR
           WRITE(NTRM,515)
      515  FORMAT(//,10X,'INPUT THE FEATURE VECTOR FORMAT   20A4')
           READ(NRDR1,450) IFMT
      450  FORMAT(20A4)
      507  CONTINUE
           WRITE(NTRM,444) NDATA
           WRITE(NTRM,852)
           WRITE(NTRM,851) IFMT
      451  FORMAT(//,10X,'FEATURE VECTOR FORMAT  ',//,10X,20A4)
      852  FORMAT(//,10X,'FEATURE VECTOR FORMAT  ')
      851  FORMAT(//,10X,10A4)
           WRITE(NTRM,508)
      508  FORMAT(//,10X,'USE DEFAULT FEATURE INDEX VECTOR--Y OR N-- A1')
           READ(NRDR1,506) IDEF
           IF(IDEF.EQ.IY) GO TO 509
C  READ IN THE FEATURE  INDEX VECTOR
           WRITE(NTRM,516)
      516  FORMAT(//,10X,'INPUT THE FEATURE INDEX VECTOR  30I2')
           READ(NRDR1,452) (FEATVC(I),I=1,D)
      452  FORMAT(30I2)
      509  CONTINUE
           WRITE(NTRM,453) (FEATVC(I),I=1,D)
      453  FORMAT(//,10X,'THE FEATURE INDEX VECTOR ',//,10X,30I3)
           DO 300 I=1,D
           IF(FEATVC(I).LE.0) WRITE(NTRM,301)
           IF(FEATVC(I).LE.0) GO TO 666
      300  CONTINUE
      301  FORMAT(//,10X,'ERROR -- FEATURE VECTOR FORMAT - OR -',
          *' FEATURE INDEX VECTOR IS WRONG')
           N1=D+1+(D*(D+1))/2
           N2=N1+(N1*(N1+1))/2
           WRITE(NTRM,33) N1,N2
       33  FORMAT(//,10X,'N1=',I5,5X,'N2=',I5)
           IF(N1.GT.NN1) WRITE(NTRM,103) NN1
           IF(N2.GT.NN2) WRITE(NTRM,104) NN2
           IF(N2.GT.NN2) N2= NN2
           IF(N1.GT.NN1) GO TO 667
      103  FORMAT(//,10X,' ERROR -- N1 EXCEEDS ',I10)
C104  FORMAT(//,10X,' ERROR -- N2 EXCEEDS ',I10)
      104  FORMAT(//,10X,' N2 REPLACED BY ',I10)
           WRITE(NTRM,523)
      523  FORMAT(//,10X,'IS ALL THE INPUT CORRECT? --Y OR N-- A1')
           READ(NRDR1,506) IDEF
           IF(IDEF.NE.IY) GO TO 667
           WRITE(NTRM,639)
      639  FORMAT(//,10X,'EXECUTION BEGINS')
C
C    OUTPUT TO UNIT NPRT
C
           WRITE(NPRT,837) PFLAG
           WRITE(NPRT,20) RUNIT,WUNIT
           WRITE(NPRT,22) D,T,ISG7,NT
           IF(D.GT.DD) WRITE(NPRT,101) DD
           IF(T.GT.TT) WRITE(NPRT,102) TT
           WRITE(NPRT,15) E
           WRITE(NPRT,35)
           DO 350 I=1,T
           WRITE(NPRT,25) (C(I,J),J=1,T)
      350  CONTINUE
           WRITE(NPRT,700) INDEX
           WRITE(NPRT,444) NDATA
           WRITE(NPRT,451) IFMT
           WRITE(NPRT,453) (FEATVC(I),I=1,D)
           IF(FEATVC(I).LE.0) WRITE(NPRT,301)
           WRITE(NPRT,33) N1,N2
           IF(N1.GT.NN1) WRITE(NPRT,103) NN1
           IF(N2.GT.NN2) WRITE(NPRT,104) NN2
           RETURN
```

3.2.10 SOFTWARE COMPONENT NO. 10 (MPPTC)

The program MPPTC is the main driver program for the batch version of the second processor. This processor uses the loss vector matrix (made by the first processor) and the pixel data to compute the minimum loss clasification.

3.2.10.1 Linkages

The program MPPTC calls subroutines SPPTC PPTC, CLOSK GETIME, GTDATE, and IDNAME. The subroutine PPTC in turn calls READIT and PHI. The subroutines CLOCK, GETIME, GTDATE, and IDNAME are "system subroutines" and descriptive by name.

3.2.10.2 Interfaces

MPPTC interfaces with other routines through calling sequences, and common blocks UN and FV. The common blocks are initialized in PPTBLK.

3.2.10.3 Inputs

All input to MPPTC comes from subroutines called by it.

3.2.10.4 Outputs

Output to the printout from MPPTC are the date, time, user name, user I.D., and C.P.U. time.

3.2.10.5 Storage

Program size=27550.

3.2.10.6 Description

The program MPPTC is the second of two processors used in sequence to classify the input data using the Patterson-Pitt-Thadani algorithm for minimum loss classification. MPPTC classifies the data using the loss vector matrix computed by the first processor.

3.2.10.7 <u>Flowchart</u>

N/A

3.2.10.8 <u>Listing</u>

FILE MPPTC

```
                C    CONVERTED BY G W AHLERS
                C  THIS PROGRAM (MPPTC) USES THE FOLLOWING SUBROUTINES
                C  SPPTC
                C  PPTC
                C  READIT
                C  [illegible]I
                C THE PATTERSON-PITT-THADANI CLASSIFIER.
      0001            INTEGER D,T,CAT,UNIT
                C     PARAMETER NN1=300,TT=10,DD=30
      0002            INTEGER RUNIT,WUNIT
      0003            COMMON /US/USER1,N-DD2,NPRT,RUNIT,WUNIT
      0004            INTEGER FEATVC
      0005            COMMON /FV/FEATVC(30),IFMT(20),DATA
      0006            DOUBLE PRECISION A(300,10),L(10),P(300)
      0007            DOUBLE PRECISION X(30)
      0008            INTEGER USERID(2),NAME(4),TIME(3),DATE(3)
      0009            TIKTOK=0.
      0010            CALL CLOCK(TIKTOK)
      0011            WRITE(NPRT,100)
      0012        100 FORMAT(1H1,10X,'THE PATTERSON-PITT-THADANI CLASSIFIER PROGRAM')
      0013            CALL GETIME(TIME)
      0014            CALL GTDATE(DATE)
      0015            CALL IDNAME(USERID,NAME)
      0016            WRITE(NPRT,200)USERID,NAME,DATE,TIME
      0017        200 FORMAT(//,10X,2A4,4X,4A4,4X,3A4,4X,3A4)
      0018            CALL SPPTC(UNIT,N,D,T,INSZ,NT,INDEX,N1,NP)
      0019            CALL PPTC(N,D,T,ISZ,NT,UNIT,N1,A,L,P,INDEX,X,NP)
      0020            CALL CLOCK(TIKTOK)
      0021            WRITE(NPRT,300) TIKTOK
      0022        300 FORMAT(//,10X,'TIME FOR PPTC',F10.3)
      0023            STOP
      0024            END
```

3.2.11 SOFTWARE COMPONENT NO. 11 (SPPTC)

Subroutine SPPTC reads the input cards and sets option switches for the first processor.

3.2.11.1 Linkages

SPPTC is called by the program MPPTC and uses data initialized in PPTBLK.

3.2.11.2 Interfaces

SPPTC interfaces with MPPTA through a calling sequence and interfaces with MPPTC and PPTBLK through common blocks UN, PF, and FV.

3.2.11.3 Inputs

Calling sequence:

Subr. SPPTC(UNIT,M,D,T,ISGZ,NT,INDEX,NP)

Parameter	Dimension	In/Out	Description
UNIT	1	Out	Unit number for the loss vector matrix data set
M	1	Out	First dimension of the loss vector matrix
D	1	Out	Number of channels
T	1	Out	Number of classes
ISGZ	1	Out	Number of small grain pixels
NT	1	Out	Total number of pixels
INDEX	1	Out	Interaction index
N1	1	Out	Array size used in PPTC
NP	1	Out	Same as NT

Common blocks:

See PPTBLK for information about the common blocks.

Input cards (unit NRDR1):

	Variables	Format	Function
1.	PFLAG	I5	0- for short printout 1- for long printout.
2.	ISGZ,NT	2I5	ISGZ- No. of small grain pixels NT- Total number of pixels
3.	INDEX	I5	Interaction index
4.	IDEF	A1	Y- use default data vector input format N- input an input format
5.	(use if IDEF=N) NDATA	I5	Number of data points per pixel
6.	(use if IDEF=N) (IFMT(I),I=1,20)	20AA	Input format
7.	IDEF	A1	Y- use default feature index vector N- input a feature index vector
8.	(use if IDEF=N) (FEATVC(I),I=1,D)	30I2	The feature index vector

3.2.11.4 Outputs

Input information is printed out.

3.2.11.5 Storage

Program size=2694.

3.2.11.6 Description

SPPTC is the input subroutine for all except the pixel data.

If default options are not used this subroutine inputs the format for the pixel data and the feature index vector.

3.2.11.7 Flowchart

N/A

3.2.11.8 Listing

```
      [illegible]
      [illegible] SPPTC( [illegible]T,[illegible]T,ISAZ,[illegible]T,INDEX,NL,NF)
      [illegible]
      [illegible]
      [illegible] NRMT,MUNIT,MUNIT
      [illegible]
      [illegible]
      [illegible]
      [illegible] ,IFAZ(I(20),MDATA
      [illegible]
      [illegible] THIS SUBROUTINE
      [illegible] 1/3[illegible]/,TI/13/,[illegible]/30/
      [illegible] 1X/Y/
      [illegible]
      [illegible] READ
      [illegible]
      [illegible]
      [illegible] 23) [illegible]T
      [illegible] CLASS VECTOR MATRIX UNIT NO. = ',I5)
      [illegible]
      [illegible] ISAZ,[illegible]T
   21 [illegible]
      [illegible]
  375 [illegible]
      [illegible] 376) [illegible]
  376 [illegible] = ',I5)
      [illegible] ISAZ,[illegible]T
   22 [illegible] OF CHANNELS=',I5,//,10X,'NO. OF CLASSES=',
     * I5,//,10X,[illegible] PIXELS=',I5,//,10X,
     * [illegible]=',I5)
      IF([illegible]) GO
      IF([illegible] 102) IT
  101 [illegible] -- THE NO. OF CHANNELS EXCEEDS ',I5)
  102 [illegible] -- THE NO. OF CLASSES EXCEEDS ',I5)
      IF([illegible]) STOP
      IF([illegible]) STOP
C [illegible] INTERACTIONS WITH
C [illegible]
      [illegible] INDEX
  776 [illegible]
      [illegible] INDEX
  7[illegible] [illegible]=',I5)
      [illegible]
  1[illegible]
      [illegible]
      [illegible] 334) [illegible]
  334 [illegible] = ',I5)
      [illegible]
  5[illegible]
      IF([illegible]) GO TO 507
      [illegible] MDATA
  444 [illegible]TA=',I5)
C [illegible] THE FEATURE VECTOR
      [illegible] (446) IFMT
  451 [illegible] (20A4)
  507 [illegible]
      [illegible] (445) [illegible]
      [illegible]
  451 [illegible] VECTOR FORMAT ',20A4)
      [illegible]
      IF([illegible]) GO TO 509
C [illegible] THE FEATURE INDEX VECTOR
      [illegible] (FEATVC(I),I=1,N)
  452 [illegible]
      [illegible]
      [illegible] (FEATVC(I),I=1,N)
  453 [illegible] THE FEATURE INDEX VECTOR ',//,10X,30I3)
      [illegible] I=1,[illegible]
      IF([illegible]) WRITE([illegible],301)
      IF([illegible]) STOP
      [illegible]
  5[illegible] -- FEATURE VECTOR FORMAT - OR -',
     * [illegible] VECTOR IS [illegible]')
      [illegible]
      [illegible]
```


ORIGINAL PAGE IS
OF POOR QUALITY

FILE SPPTC

```
0064          IF(I.GT.N1) WRITE [illegible]                SPP00770
0065          IF( I.GT.N1) STOP                            SPP00780
0066    103   FORMAT(//,13X,' ERROR -- N1 EXCEEDS ',I10)   SPP00790
0067          RETURN                                       SPP00800
0068          END                                          SPP00810
```

3.2.12 SOFTWARE COMPONENT NO. 12 (PPTC)

Subroutine PPTC is the main computational subroutine of the second processor. Input from SPPTC or SPPTCI is passed to PPTC. PPTC with the aid of other subroutines calculates the classification losses to find the minimum loss.

3.2.12.1 Linkages

Subroutine PPTC is called by MPPTC or MPPTCI and is passed information from SPPTC or SPPTCI. PPTC calls subroutines READIT and PHI.

3.2.12.2 Interfaces

PPTC interfaces with other routines through a calling sequence and common blocks UN and PF.

3.2.12.3 Inputs

Calling sequence:

Subr. PPTC(M,D,T,ISGZ,NT,UNIT,N1,A,L,P,INDEX,X,NP)

Parameter	Dimension	In/Out	Description
M	1	In	First dimension of the loss vector matrix
D	1	In	Number of channels
T	1	In	Number of classes
ISGZ	1	In	Number of small grain pixels
NT	1	In	Total number of pixels
UNIT	1	In	Unit number for the loss vector matrix data set
N1	1	In	Array size for A and P
A	(N1,T)	In	The loss vector matrix.

Parameter	Dimension	In/Out	Description
L	T	-	The losses for each class
P	N1	-	The phi function vector
INDEX	1	In	Interactive index
X	D	-	The feature vector
NP	1	In	Same as NT

Common blocks:

See PPTBLK for information about the common blocks.

3.2.12.4 Outputs

Classification information is printed out.

3.2.12.5 Storage

Program size=2550.

3.2.12.6 Description

PPTC takes the interproduct of a loss vector and a phi vector to determine a class loss for a particular feature vector. The minimum of these is used as the classification for a particualr set of input data.

3.2.12.7 Flowchart

N/A

3.2.12.8 Listing

FILE PPTC

```
            C        THE PAT-PIT-THADANI CLASSIFIER.
            C        SUBROUTINES REQUIRED...PHI1,READII.
            C        THIS PROGRAM CLASSIFIES THE TEST PATTERN X.
            C        THE LOSS VECTOR MATRIX IS READ OF A USER SPECIFIED UNIT.
 0001                SUBROUTINE PPTC(M,D,T,ISGZ,NT,UNIT,N1,A,L,P,INDEX,X,NP)
            C        PARAMETER N1=300
 0002                INTEGER CAT,D,T,UNIT
 0003                INTEGER RUNIT,WUNIT
            C        PARAMETER TT=10
 0004                INTEGER P11,P12,P22,P21
 0005                DOUBLE PRECISION A(N1,T),L(T),P(N1),LMIN
 0006                DOUBLE PRECISION X(D)
 0007                COMMON /AA/NRDR1,NRDR2,NPRT,RUNIT,WUNIT
 0008                INTEGER PFLAG
 0009                COMMON /PF/PFLAG
            C         READ LOSS VECTORS OF APPROPRIATE UNIT.
            C         COMPUTE CLASS LOSSES L(J) = A(J)*P.
            C         ASSIGN X TO CLASS WITH MINIMUM LOSS.
            C      READ LOSS VECTOR MATRIX
 0010                INTZ=NT-ISGZ
 0011                WRITE(NPRT,777) PFLAG
 0012          777    FORMAT(//,10X,'PFLAG=',I5)
 0013                READ(UNIT,105) ((A(I,J),J=1,T),I=1,M)
 0014          105   FORMAT(2D20.10)
            C        WRITE(NPRT,700) ((A(I,J),J=1,T),I=1,M)
 0015          700   FORMAT(/1X,'LOSS VECTOR MATRIX  '/,(2D20.10))
 0016                P11=0
 0017                P12=0
 0018                P22=0
 0019                P21=0
 0020                IF(PFLAG.EQ.1) WRITE(NPRT,106)
 0021          106    FORMAT(//,10X,'THE INPUT DATA AND CLASSIFICATION RESULTS',//)
 0022                DO 1 KOUNT=1,NP
 0023                CALL READII(X,D)
 0024                CALL PHI(X,P,D,N1,INDEX)
 0025                DO 10 J=1,T
 0026                L(J) = 0.0D0
 0027                DO 20 I=1,M
 0028                L(J) = L(J) + A(I,J)*P(I)
 0029           20   CONTINUE
 0030           10   CONTINUE
            C         DETERMINE MINIMUM LOSS AND CATEGORY.
 0031                CAT = 1
 0032                LMIN = L(1)
 0033                DO 30 I=1,T
 0034                IF(L(I) .GE. LMIN) GO TO 30
 0035                LMIN=L(I)
 0036                CAT = I
 0037           30   CONTINUE
            C        WRITE(NPRT,3)  (X(II),II=1,D)
            C        WRITE(NPRT,5)  (L(II),II=1,T)
            C        WRITE(NPRT,4)  CAT
 0038                IF(KOUNT .GT. ISGZ) GO TO 200
 0039                IF(CAT .EQ. 2) GO TO 102
 0040                P11=P11+1
 0041                GO TO 111
 0042          102   P12=P12+1
 0043                GO TO 111
 0044          200   IF(CAT .EQ. 1) GO TO 201
 0045                P22=P22+1
 0046                GO TO 111
 0047          201   P21=P21+1
 0048           111    CONTINUE
            C         WRITE CLASSIFICATION RESULT.
 0049                IF(PFLAG.EQ.1) WRITE(NPRT,3)  (X(I),I=1,D)
 0050        3        FORMAT(1H ,2X,'X',2X,15F8.1)
 0051                IF(PFLAG.EQ.1) WRITE(NPRT,4)  CAT
 0052            4   FORMAT(1H ,8X,5HCLASS,I5)
 0053                IF(PFLAG.EQ.1) WRITE(NPRT,5)  (L(I),I=1,T)
 0054            5   FORMAT(1H ,6X,16H  CLASS LOSSES   ,(6X,4D20.10))
 0055                IF(PFLAG.EQ.1) WRITE(NPRT,56) KOUNT
 0056           56     FORMAT(1H ,10X,'KOUNT = ',I5)
 0057                IF(PFLAG.EQ.1) WRITE(NPRT,55)
 0058           55     FORMAT(//)
            C        WRITE(NPRT,4) CAT
 0059            1   CONTINUE
```

FILE PPIC

```
      WRITE(IPRT,605)
  605 FORMAT(/1X,'...OVERALL CLASSIFICATION RESULTS...'/)
      WRITE(IPRT,550) P11,P12,P22,P21
  550 FORMAT(/7X,'P11   P12   P22   P21',/,4X,4I5/)
      P1=FLOAT(P11)/FLOAT(ISSZ)
      P2=FLOAT(P22)/FLOAT(IOTHZ)
      ITLZ = ISSZ + IOTHZ
      PAV=(FLOAT(P11) + FLOAT(P22))/FLOAT(ITLZ)
      WRITE(IPRT,600) P1,P2,PAV
  600 FORMAT(/4X,'   PC1         PC2         PAV   ',/,3F10.3)
      STOP
      END
```

3.2.13 SOFTWARE COMPONENT NO. 13 (MPPTCI)

Program MPPTCI is the interactive version of MPPTC. The only difference is MPPTCI calls SPPTCI instead of SPPTC. For more detail see SOFTWARE COMPONENT NO. 10.

3.2.14.1 <u>Listings</u>

FILE MPPTCI

```
              C   ADAPTED BY C. AHLERS
              C  THIS PROGRAM (MPPTC) USES THE FOLLOWING SUBROUTINES
              C  SPPTC
              C  PPTC
              C  READIT
              C  PHI
              C THE PATTERSON-PITT-THADANI CLASSIFIER.
0001                INTEGER D,T,CAT,UNIT
              C     PARAMETER NN1=300,TT=10,DD=30
0002                INTEGER RUNIT,WUNIT
0003                 COMMON /UN/NRDR1,NRDR2,NPRT,RUNIT,WUNIT
0004                INTEGER FEATVC
0005                COMMON /FV/FEATVC(30),IFMT(20),NDATA
0006                DOUBLE PRECISION A(300,10),L(10),P(300)
0007                DOUBLE PRECISION X(30)
0008                 INTEGER USERID(2),NAME(4),TIME(3),DATE(3)
0009                TIKTOK=0.
0010                CALL CLOCK(TIKTOK)
0011                WRITE(NPRT,100)
0012           100  FORMAT(1H1,10X,'THE PATTERSON-PITT-THADANI CLASSIFIER PROGRAM')
0013                 CALL GETIME(TIME)
0014                 CALL GTDATE(DATE)
0015                 CALL IDNAME(USERID,NAME)
0016                 WRITE(NPRT,200)USERID,NAME,DATE,TIME
0017           200   FORMAT(//,10X,2A4,4X,4A4,4X,3A4,4X,3A4)
0018                CALL SPPTCI(UNIT,M,D,T,ISGZ,NT,INDEX,N1,NP)
0019                CALL PPTC(M,D,T,ISGZ,NT,UNIT,N1,A,L,P,INDEX,X,NP)
0020                CALL CLOCK(TIKTOK)
0021                WRITE(NPRT,300) TIKTOK
0022           300   FORMAT(//,10X,'TIME FOR PPTC',F10.3)
0023                STOP
0024                END
```

ORIGINAL PAGE IS
OF POOR QUALITY

3.2.14 SOFTWARE COMPONENT NO. 14 (SPPTCI)

Subroutine SPPTCI is an interactive version of SPPTC. It prompts the user to input set up information.

3.2.14.1 Linkages

SPPTCI is called by the program MPPTCI and uses data initialized in PPTBLK.

3.2.14.2 Interfaces

SPPTCI interfaces with MPPTCI through a calling sequence and interfaces with MPPTCI and PPTBLK through common blocks UN, PF, FV, and TUN.

3.2.14.3 Inputs

Calling sequence:

Subr. SPPTCI(UNIT,M,D,T,ISGZ,NT,INDEX,NP)

Parameter	Dimension	In/Out	Description
UNIT	1	Out	Unit number for the loss vector matrix data set
M	1	Out	First dimension of the loss vector matrix
D	1	Out	Number of channels
T	1	Out	Number of classes
ISGZ	1	Out	Number of small grain pixels
NT	1	Out	Total number of pixels
INDEX	1	Out	Interaction index
N1	1	Out	Array size used in PPTC
NP	1	Out	Same as NT

Common blocks:

See PPTBLK for information about the common blocks.

Input variables:

The user is prompted to input the setup variables from the terminal.

3.2.14.4 Outputs

Input information is printed out and sent to the terminal.

3.2.14.5 Storage

Program size=3568.

3.2.14.6 Description

SPPTCI is the interactive input subroutine for all except the pixel data. If default options are not used this subroutine inputs the format for the pixel data and the feature index vector. A long or short printout is an option.

3.2.14.7 Flowchart

N/A

3.2.14.8 Listings

```
[illegible]
      SUBROUTINE [illegible] (II,[illegible],I,[illegible],INDEX,[illegible],NP)
      [illegible]
      [illegible] MPRT,[illegible]UNIT,[illegible]UNIT
      [illegible]
      [illegible]
      [illegible],TFMT(20),NDATA
C     [illegible],II=1,[illegible]=30
C DIMENSIONS CONCERN IN THIS SUBROUTINE
      DATA [illegible]/,II/10/,[illegible]/39/
      [illegible]
      [illegible]
  [illegible] CONTINUE
C [illegible]
      [illegible]
  [illegible] [illegible] THE PRINT FLAG -- 0 OR 1 -- I1')
      [illegible]
      [illegible] FLAG
  [illegible] [illegible]FLAG=',I5)
      [illegible]
   10 [illegible]
      [illegible]
   26 [illegible]CLASS VECTOR MATRIX UNIT NO. = ',I5)
      [illegible]
  [illegible] [illegible] IS67     I5')
  [illegible] [illegible]      I5')
      [illegible] IS6Z
   [illegible] [illegible]
      [illegible]
      [illegible]
      [illegible]
   37 [illegible]
      [illegible]
  [illegible] [illegible] = ',I5)
      [illegible]
   22 [illegible] OF CHANNELS=',I5,//,10X,'NO. OF CLASSES=',
     * I5,//,10X,'NO. OF SMALL GRAIN PIXELS=',I5,//,10X,
     * 'TOTAL NO. OF SAMPLES=',I5)
      IF([illegible]) [illegible]
      IF([illegible]) [illegible]
  101 [illegible] -- THE NO. OF CHANNELS EXCEEDS ',I5)
  102 [illegible] ERROR -- THE NO. OF CLASSES EXCEEDS ',I5)
      IF([illegible]) STOP
      IF([illegible]) STOP
C [illegible] FEATURE WHOSE INTERACTIONS WITH
C THE [illegible] FEATURES ARE TO BE IGNORED.
      [illegible],777)
  777 FORMAT(//,5X,'[illegible] NON-INTERACTIVE FEATURE INDEX  I1'/)
      [illegible]
  77[illegible] [illegible]
      [illegible]
  700 [illegible]=',I5)
C     [illegible]
  [illegible] [illegible] NO. OF TEST SAMPLES...I5')
C     [illegible]
      [illegible]
  1[illegible] [illegible](I5)
C     [illegible],333) [illegible]
  333 [illegible] = ',I5)
  [illegible] [illegible]
      [illegible]
  [illegible] [illegible]USE DEFAULT FEATURE VECTOR FORMAT--Y OR N-- A1')
      [illegible]
  [illegible] [illegible](A1)
      IF([illegible].EQ.IY) GO TO 507
      [illegible]
  [illegible] [illegible] NDATA    I5')
      [illegible]
      [illegible] NDATA
  [illegible] [illegible]=',I5)
```

3-49

ORIGINAL PAGE IS
OF POOR QUALITY

FILE SPPTCI

```
0065            READ(PROBL,503) IF
0066      [illegible] FORMAT(A4)
0067      507   CONTINUE
0068            WRITE([illegible],[illegible])
0069            WRITE([illegible],[illegible]) IFM
0070      451   FORMAT(//,1X,'FEATURE VECTOR FORMAT  ',A4)
0071      [illegible] FORMAT(//,15X,'FEATURE VECTOR FORMAT  ')
0072      852   FORMAT(//,10X,1084)
0073            WRITE([illegible],[illegible])
0074      50[illegible]   FORMAT(//,10X,'USE DEFAULT FEATURE INDEX VECTOR--Y OR N-- A1')
0075            READ([illegible],506) IDEF
0076            IF(IDEF.EQ.IY) GO TO 509
0077            WRITE([illegible],516)
0078      516   FORMAT(//,10X,'INPUT THE FEATURE INDEX VECTOR  30I2')
C     READ IN THE FEATURE INDEX VECTOR
0079            READ(PROBL,452) (FEATVC(I),I=1,N)
0080      452   FORMAT(30I2)
0081      509   CONTINUE
0082            WRITE([illegible],453) (FEATVC(I),I=1,N)
0083      453   FORMAT(//,10X,'THE FEATURE INDEX VECTOR ',//,10X,30I3)
0084            DO 300 I=1,N
0085            IF(FEATVC(I).LE.0) WRITE([illegible],301)
0086            IF(FEATVC(I).LE.0) GO TO 666
0087      300   CONTINUE
0088      301   FORMAT(//,10X,'ERROR -- FEATURE VECTOR FORMAT - OR -',
         *'FEATURE INDEX VECTOR IS WRONG')
0089            [illegible]=[illegible]*([illegible]+1))/2
0090            WRITE([illegible],33) [illegible]
0091      33    FORMAT(//,10X,'[illegible]=',I5)
0092            IF([illegible].GT.[illegible]) WRITE([illegible],103) [illegible]
0093            IF([illegible].GT.[illegible]) STOP
0094      103   FORMAT(//,10X,' ERROR -- [illegible] EXCEEDS ',I10)
0095            WRITE([illegible],555)
0096      555   FORMAT(//,10X,'ARE ALL INPUTS CORRECT? -- Y OR N --  A1')
0097            READ([illegible],506) IDEF
0098            IF(IDEF.EQ.IY) GO TO 667
0099            WRITE([illegible],556)
0100      556   FORMAT(//,10X,'BEGINNING EXECUTION')
C
C     OUTPUT TO UNIT NPRT
0101            WRITE(NPRT,20) UNIT
0102            WRITE(NPRT,326) [illegible]
0103            WRITE(NPRT,22) [illegible],[illegible],ISSZ,[illegible]
0104            IF([illegible].GT.[illegible]) WRITE(NPRT,101) GG
0105            IF(I.GT.IT) WRITE(NPRT,102) TT
0106            WRITE(NPRT,700) INDEX
C           WRITE(NPRT,333) [illegible]
0107            WRITE(NPRT,566) [illegible]
0108            WRITE(NPRT,451) IFM
0109            WRITE(NPRT,453) (FEATVC(I),I=1,N)
0110            IF(FEATVC(I).LE.0) WRITE(NPRT,301)
0111            WRITE(NPRT,33) [illegible]
0112            IF([illegible].GT.[illegible]) WRITE(NPRT,103) [illegible]
0113            RETURN
0114            END
```

4. OPERATION

This program has been implemented on the Purdue-LARS 370/148 Computer and runs under CMS370/VM/370. It is callable from dial-up remote terminals or from the directly connected terminals in the LARS terminal area in JSC Bldg 17. For information regarding sign-on, please contact personnel in one of the following:

1. LEC Scientific Applications Section.

2. LEC Techniques Development Section.