교 과 목 414.519 강의록

수 치 선 박 유 체 역 학

- 보 텍 스 방 법 -

COMPUTATIONAL MARINE HYDRODYNAMICS
-VORTEX METHODS-

2019년 6월 27일

Suh, Jung-Chun

서 정 천

Seoul National Univ., Dept. NAOE

서울대학교 공과대학조선해양공학과

C

CODE *PRpan* FOR PANEL METHOS

C.1 Introduction

PRpan code provides the calculation of potential and velocity induced by source and doublet distribution on a planar polygon element with linearly varying density.

As input arguments, the number of sides and vertex positions for specifying the element geometry are needed. The source and doublet strengths should be entered. Note that the doublet strength is defined as the negative value of potential: $\mu = -\phi$.

The number of sides is at least 3 and the vertex positions are specified as a pair of x, y, z coordinates. Multiple field points can be entered in a form of (x_p, y_p, z_p) to avoid duplicate calculations for each side of the element when calculating the influence coefficients in the panel method. By assuming that any combination of 3 points among the vertices form a unique planar element, the vector outward normal to the element surface is calculated by cross-product of two vectors connected between the first, the second and the third vertex points.

C.2 Program Lists of Subroutine *PRpan*

```
C--------------------- Subroutine PRpan.ftn ----------------------------
C                                                                      |
C     PRpan code provides the calculation of potenial and velocity     |
C     induced by source and doublet distributions with constant plus   |
C     linear variation on a planar polygon element, by following the   |
C     formulation of Cantaloube and Rehbach ~(1986) and performing the |
C     analytic evaluations of the associated line integrals            |
C     (Suh ~(1990)).                                                   |
C     Before this subroutine is called, the subroutine PRgeom should   |
C     be called to generate the geometric parameters of the element    |
C     associated with the analytic evaluations of the line integrals.  |
C                                                                      |
C     REFERENCES:                                                      |
C      [1] Cantaloube, B. and Rehbach, C. ~(1986)                      |
C         " Calcul des Integrales de la Methode des Singularites"      |
C           Recherche Aerospatiale, no. 1, pp. 15-22,                  |
C           (English Title: " Calculation of the Integrals of the      |
C           Singularity Method" Aerospace Research, No. 1, pp. 15-22)  |
C      [2] Suh, J. C. ~(1990)                                          |
C         " Review of the Paper; Calculation of the Integrals of the   |
C           Singularity Method by Cantaloube and Rehbach"              |
C           KRISO Propulsor Technology Laboratory Report, 22-90        |
C      [3] Suh, J. C. ~(1990)                                          |
C         " Analytic Evaluations of the Induction-Integrals for        |
C           Distributions of Sources and Doublets over a Planar        |
C           Polygon Element"                                           |
C           KRISO Propulsor Technology Laboratory Report, 23-90        |
C      [4] Lee, C.-S. ~(1990)                                          |
C         " Treatment of non-planar panel"                             |
```

```
C          Technical Notes (unpublished), 11/29/90                 |
C                                                                  |
C     Note: The doublet strength is defined as a negative value of |
C          potential here.                                         |
C                                                                  |
C     Programmer: Suh, Jung-Chun                                   |
C      Dept. Naval Architecture and Ocean Engineering,             |
C      Seoul National University                                   |
C     (Copyright:Korea Research Institute of Ships & Ocean Engineering) |
C                                                                  |
C     Version 1.0, December 15, 1990                               |
C     Version 1.1, December 27, 1990                               |
C     Version 1.2, January   3, 1991                               |
C                                                                  |
C     Input argument descriptions:                                 |
C      NSIDE            = number of sides of a planar element      |
C                         (at least 3)                             |
C      XV(i,N)          = x-, y- and z-coordinates of the          |
C                         NSIDE vertices of the element            |
C                         (two-dimensional arrayed values in a     |
C                          counterclockwise order).                |
C      NFP              = number of field points at which the induced |
C                         potentials and velocities are calculated. |
C      XFP(i,n)         = x-, y- and z-coordinates of the NFP field |
C                         points (two-dimensional arrayed values). |
C      ISELF(n)         = choice of location of each field point to |
C                         specify a self-induction case            |
C                         (NFP arrayed values).                    |
C                         (If it is just    on surface,    ISELF(n)=+1 |
C                          If it is exactly at surface,    ISELF(n)= 0 |
C                          If it is just    below surface, ISELF(n)=-1 |
C                          If it is away from surface, ISELF(n)=others)|
C      SEGL(N)          = lengths of the NSIDE respective sides,   |
C                         computed by the subroutine PRGEOM        |
C                         (NSIDE arrayed values).                  |
C      AN(i)            = x-, y- and z-components of the unit outward |
C                         normal vector of the planar element,     |
C                         computed by the subroutine PRGEOM        |
C                         (3 arrayed values).                      |
C      EL(i,N)          = x-, y- and z-components of the unit      |
C                         orientation vector of the NSIDE respective |
C                         sides, computed by the subroutine PRGEOM |
C                         (two-dimensional arrayed values).        |
```

```
C      EM(i,N)           = x-, y- and z-components of the unit        |
C                          vector lying on the plane and normal to the |
C                          NSIDE respective sides,computed by the      |
C                          subroutine PRGEOM                           |
C                          (two-dimensional arrayed values).           |
C      CG(i)             = x-, y- and z- coordinates of the centroid   |
C                          of the element, computed by the subroutine  |
C                          PRGEOM (3 arrayed values).                  |
C      AREA              = surface area of the element, computed by the |
C                          subroutine PRGEOM.                          |
C      DIAGNL            = longest diagonal of the element, computed by |
C                          the subroutine PRGEOM.                      |
C      RFAR              = reference ratio of field-point distance to  |
C                          longest diagonal to apply far-field         |
C                          approximation.                              |
C      ICOMP             = choices of selected calculations            |
C                          (ICOMP=1: only potential calculations;      |
C                           ICOMP=2: only velocity calculations;       |
C                           ICOMP=3: both potential and velocity calc.) |
C      SIGMAX,SIGMAY,    = coefficients to specify the linear variation |
C      SIGMAZ,SIGMA0       of source distribution as                   |
C                          sigma=SIGMAX*x+SIGMAY*y+SIGMAZ*z+SIGMA0.    |
C      AMUX,AMUY,        = coefficients to specify the linear variation |
C      AMUZ,AMU0           of doublet distribution as                  |
C                          mu=AMUX*x+AMUY*y+AMUZ*z+AMU0.               |
C                                                                      |
C     Output argument descriptions:                                    |
C      SPOT(N)                = induced potentials at NFP field points |
C                               due to the linear-source distribution  |
C                               (NFP arrayed values).                  |
C      SVX(N), SVY(N), SVZ(N) = induced velocity componets at field    |
C                               points due to the linear-source        |
C                               distribution (NFP arrayed values).     |
C      DPOT(N)                = induced potentials at field points due |
C                               to the linear-doublet distribution     |
C                               (NFP arrayed values).                  |
C      DVX(N), DVY(N), DVZ(N) = induced velocity componets at field    |
C                               points due to the linear-doublet       |
C                               distribution (NFP arrayed values).     |
C     Recommendations:                                                 |
C     When one applies this subroutine to the panel method, it may be  |
C     efficient in computing time to separate the subroutine into      |
C     parts for constant and linear distribution cases, and            |
```

```
 C     furthermore into potential and velocity calculation parts.        |
 C                                                                       |
 C-----------------------------------------------------------------------
 C

      SUBROUTINE PRPAN
     &   (NSIDE,XV,NFP,XFP,ISELF,SEGL,AN,EL,EM,CG,AREA,DIAGNL,RFAR,{input}
     &    ICOMP,SIGMAX,SIGMAY,SIGMAZ,SIGMA0,AMUX,AMUY,AMUZ,AMU0,   {input}
     &    SPOT,SVX,SVY,SVZ,DPOT,DVX,DVY,DVZ)                      {output}
      IMPLICIT REAL*8 (A-H,O-Z)            {high precision is recommended}
C     single precision is used only for calling arguments.
      REAL*4 XV,XFP,SEGL,AN,EL,EM,RINTM,CG,AREA,DIAGNL,RFAR,
     &       SIGMAX,SIGMAY,SIGMAZ,SIGMA0,AMUX,AMUY,AMUZ,AMU0,
     &       SPOT,SVX,SVY,SVZ,DPOT,DVX,DVY,DVZ
      PARAMETER (NSDMAX=4)
      {up to a quadrilateral panel; change 4 to N for an N-side polygon}
      DIMENSION AN(3),R(3,NSDMAX),D(3),CG(3),SEGL(NSDMAX),PQ(NSDMAX),
     &          EL(3,NSDMAX),EM(3,NSDMAX),XV(3,NSDMAX),XFP(3,1),
     &          ISELF(1),GSEM(NSDMAX),GDEM(NSDMAX),SIGMA(NSDMAX),
     &          AMU(NSDMAX),ALPHA(NSDMAX),BETA(NSDMAX),
     &          SPOT(1),SVX(1),SVY(1),SVZ(1),
     &          DPOT(1),DVX(1),DVY(1),DVZ(1)
C     CHECK NUMBER OF SIDES OF A POLYGON
      IF (NSIDE.LT.3 .OR. NSIDE.GT.NSDMAX) THEN
      WRITE(*,*) 'ERROR: NUMBER OF SIDE SHOULD BE AT LEAST 3 ',
     &                   'AND AT MOST ',NSDMAX,'.'
      STOP     {terminate process in the case of out of range for NSIDE}
      END IF
C     DEFINE OFTEN-USED CONSTANTS
      PI=3.141592653589793
      R4PI=0.25D0/PI                                          {1/(4*pi)}
      EPS=1.0D-07    {tolerance for checking that field point is on plane}
C     LINEAR VARIATION OF SIGMA & DOUBLET
      A1=SIGMAX
      A2=SIGMAY
      A3=SIGMAZ
      A4=SIGMA0
      B1=AMUX
      B2=AMUY
      B3=AMUZ
      B4=AMU0
      DO 3 J=1,NSIDE
      SIGMA(J)=A1*XV(1,J)+A2*XV(2,J)+A3*XV(3,J)+A4
```

```
C        {source density at vertex}
        AMU(J)   =B1*XV(1,J)+B2*XV(2,J)+B3*XV(3,J)+B4
C        {doublet density at vertex}
        ALPHA(J)=A1*EL(1,J)+A2*EL(2,J)+A3*EL(3,J)
C        {linear variation along side}
        BETA(J)  =B1*EL(1,J)+B2*EL(2,J)+B3*EL(3,J)
C       DOT PRODUCT BETWEEN GRADIENT OF SIGMA (OR DOUBLET) & VECTOR em
        GSEM(J)=A1*EM(1,J)+A2*EM(2,J)+A3*EM(3,J)
      3 GDEM(J)=B1*EM(1,J)+B2*EM(2,J)+B3*EM(3,J)
C
C       CALCULATIONS FOR MULTIPLE FIELD POINTS
C
        DO 100 K=1,NFP                              {loop for field points}
C
        DO 5 I=1,3
      5 D(I)=CG(I)-XFP(I,K)
        ANR= AN(1)*D(1)+AN(2)*D(2)+AN(3)*D(3)            {define constant a}
C       FAR-FIELD APPROXIMATIONS
        RR=D(1)**2+D(2)**2+D(3)**2
        IF(RR/DIAGNL**2 .GT.RFAR**2) THEN                {far-field approx.}
           APPR=DSQRT(RR)                           {representative distance}
           AA=R4PI*AREA/(RR*APPR)
           SIGMAR=A1*CG(1)+A2*CG(2)+A3*CG(3)+A4
C           {representative source density}
           AMUR   =B1*CG(1)+B2*CG(2)+B3*CG(3)+B4
C           {representative doublet density}
           IF(ICOMP.EQ.1 .OR. ICOMP.EQ.3) THEN
           SPOT(K)=-R4PI*AREA/APPR*SIGMAR                   {source-potential}
           DPOT(K)=+AA*ANR*AMUR                            {doublet-potential}
           END IF
           IF(ICOMP.EQ.2 .OR. ICOMP.EQ.3) THEN
           SVX(K)  =-AA*D(1)*SIGMAR
           SVY(K)  =-AA*D(2)*SIGMAR                          {source-velocity}
           SVZ(K)  =-AA*D(3)*SIGMAR
           DVX(K)  =+AA*(3*ANR*D(1)/RR-AN(1))*AMUR
           DVY(K)  =+AA*(3*ANR*D(2)/RR-AN(2))*AMUR          {doublet-velocity}
           DVZ(K)  =+AA*(3*ANR*D(3)/RR-AN(3))*AMUR
           END IF
C
        ELSE                                                 {if not far-field}
C
C       DEFINE VECTOR r FOR EACH SIDE AND DISTANCE
        DO 20 J=1,NSIDE
```

```
      DO 10 I=1,3
   10 R(I,J)=XV(I,J)-XFP(I,K)    {vector r between vertex and field point}
      PQ(J)=DSQRT(R(1,J)**2+R(2,J)**2+R(3,J)**2)            {its distance}
   20 CONTINUE
C     SIGN OF n.e
      IF(ISELF(K).EQ.+1.OR.ISELF(K).EQ.0.OR.ISELF(K).EQ.-1) THEN
        ENR=0.D0
        ISIGN=-ISELF(K)
      ELSE
        ENR=DABS(ANR)                                              {a=e.r}
        IF(ANR.GT.0.0D0) THEN
          ISIGN=+1
        ELSE                                                {sign of n.e}
          ISIGN=-1
        END IF
      END IF
C     CHECK THAT FIELD POINT IS ON EXTENSION PLANE OF PANEL (IANULL=1).
      IANULL=0
      IF(ENR.LT.EPS) IANULL=1
C     INITIAL SET FOR SUMMING UP CONTRIBUTION OF RESPECTIVE SIDE
      SPT=0.0D0
      DPT=0.0D0
      IF(ICOMP.EQ.2 .OR. ICOMP.EQ.3) THEN      {selection of calculations}
      SVEL1 =0.0D0
      SVEL2X=0.0D0
      SVEL2Y=0.0D0
      SVEL2Z=0.0D0
      SVEL3 =0.0D0
      DVEL1X=0.0D0
      DVEL1Y=0.0D0
      DVEL1Z=0.0D0
      DVEL2 =0.0D0
      DVEL3 =0.0D0
C
      END IF
C     FOR CONTRIBUTION OF EACH SIDE BY ANALYTIC EVALUATIONS
      DO 30 J=1,NSIDE                           {loop for sides of element}
C
      J1=J+1
      IF(J.EQ.NSIDE) J1=1                              {cyclic convention}
      AL=SEGL(J)                                          {length of side}
      IF(AL. LT. EPS) GO TO 30 {skip contribution of side of small length}
      B=-(R(1,J)*EM(1,J)+R(2,J)*EM(2,J)+R(3,J)*EM(3,J))      {constant bi}
```

```
C      CALL CROSS(R(1,J),EL(1,J),D)
       D(1)=R(2,J)*EL(3,J)-R(3,J)*EL(2,J)
       D(2)=R(3,J)*EL(1,J)-R(1,J)*EL(3,J)                {vector d=r x el}
       D(3)=R(1,J)*EL(2,J)-R(2,J)*EL(1,J)
C      CALCULATIONS OF LOCAL PLANE COORDINATES (x',z')
       X=-(EL(1,J)*R(1,J)+EL(2,J)*R(2,J)+EL(3,J)*R(3,J))
       Z2=D(1)**2+D(2)**2+D(3)**2
       Z=DSQRT(Z2)
C      CHECK THAT FIELD POINT IS ON EXTENSION LINE OF SIDE (IZNULL=1).
       IZNULL=0
       IF(Z.LT.EPS) IZNULL=1                              {if z' approx. 0}
C      SUPPRESS CALCULATIONS WHEN FIELD POINT IS JUST ON SIDE LINE.
       IF(IZNULL.EQ.1 .AND. (X.GE.0 .AND. X.LE.AL)) THEN
       WRITE(*,2) XFP(1,K),XFP(2,K),XFP(3,K),
      &           XV(1,J),XV(2,J),XV(3,J),XV(1,J1),XV(2,J1),XV(3,J1)
     2 FORMAT('WARNING: NUMERICAL SINGULARITY OCCURS AT FIELD POINT',
      &       ' ( ',3E13.4,' )', /6X,'FOR THE SEGMENT WITH END POINTS',
      &       ' ( ',3E13.4,' ), (', 3E13.4, ' )')
       STOP                     {terminate the process for a singular point}
       END IF
C
C      INTEGRALS WITH INTEGRAND 1/r,1/(r+a),1/r(r+a) AND 1/r**3.
C
C--- EXACT EVALUATIONS ---
C
       ALMX=AL-X                                 {because of often-used one}
       PQ1=PQ(J)                {distances between field point and end points}
       PQ2=PQ(J1)
       IF(IZNULL.EQ.1) THEN
C       {if field pt is on extension line of side, z'=a=0}
          AI1=DLOG(-ALMX/X)
          AI4=0.5D0*(AL*(ALMX-X)/(X*ALMX)**2)
             IF(X.GT.AL) THEN
                AI1=-AI1
                AI4=-AI4
             END IF
          AI3=-AL/(ALMX*X)                        {1/r(r+a) becomes 1/r**2}
          AI2=AI1                                    {1/(r+a) becomes 1/r}
       ELSE                                                {if IZNULL.NE.1}
           IF( X.LT. 0.D0-EPS) THEN
                AI1=DLOG((PQ2+ALMX)/(PQ1-X))               {integral of 1/r}
          ELSE IF( X .GT. AL+EPS) THEN
                AI1=DLOG((PQ1+X)/(PQ2-ALMX))
```

```
          ELSE IF((PQ1-X).GT.EPS .AND. (PQ2-ALMX).GT.EPS) THEN
                AI1=0.5D0*DLOG((PQ2+ALMX)*(PQ1+X)/((PQ1-X)*(PQ2-ALMX)))
C                {mean}
                 ELSE IF((PQ1-X).LE.EPS) THEN
                 AI1=DLOG((PQ2+ALMX)*2.D0*X/Z2)
                 ELSE IF((PQ2-ALMX).LE.EPS) THEN
                 AI1=DLOG((PQ1+X)*2.D0*ALMX/Z2)
                 ELSE
                 WRITE(*,2) XFP(1,K),XFP(2,K),XFP(3,K),
     &             XV(1,J),XV(2,J),XV(3,J),XV(1,J1),XV(2,J1),XV(3,J1)
                 STOP
          END IF
C
          IF(X.GT.0.D0 .AND. X.LT.AL) THEN
          AI4=(AL*PQ1 + X*(PQ2-PQ1))/(Z2*PQ1*PQ2)     {integral of 1/r**3}
          ELSE
          AI4=AL*(ALMX-X)/(PQ1*PQ2*(ALMX*PQ1-X*PQ2))
          END IF
          IF(IANULL.EQ.1) THEN                   {1/r(r+a) becomes 1/r**2}
             AI2=AI1                                 {1/(r+a) becomes 1/r}
             IF(DABS(Z2-X*ALMX).LT.EPS) THEN
              ARG=DATAN(ALMX/Z)+DATAN(X/Z)
              ELSE
              ARG=DATAN(AL*Z/(Z2-X*ALMX))  {combine two inverse functions}
              IF(Z2.LT.X*ALMX .AND. X.LT.AL) ARG=PI+ARG
             END IF
             AI3=ARG/Z
          ELSE
             IF((Z-ENR).LT.EPS) THEN                          {if z'.EQ.a}
                AI3=(AL*(PQ1+ENR)+X*(PQ2-PQ1))/((PQ2+ENR)*(PQ1+ENR)*ENR)
             ELSE                                             {if z'.NE.a}
                AA=(-Z2-ENR*PQ1)/(Z*(PQ1+ENR))
                BB=(-Z2-ENR*PQ2)/(Z*(PQ2+ENR))
                EE=DSQRT(Z2-ENR**2)
                 IF(X.GT.0.D0 .AND. X.LT.AL) THEN
                  CC= EE*(Z2*AL+ENR*(ALMX*PQ1+X*PQ2))
     &                  /(Z2*(PQ1+ENR)*(PQ2+ENR))
                  ELSE
                  CC= EE*AL*(1+ENR*(ALMX-X)/(ALMX*PQ1-X*PQ2))
     &                  /((PQ1+ENR)*(PQ2+ENR))
                 END IF
                IF(DABS(CC).LT.1.0D0) ASINCC=DASIN(CC)
                IF(CC.GE.1.0D0) ASINCC=0.5D0*PI
```

```
                 IF(CC.LE.-1.0D0) ASINCC=-0.5D0*PI
                  IF(X.LE.0. .OR. X.GE.AL .OR. (AA**2+BB**2).GE.1.) THEN
                   AI3=ASINCC/EE
                   ELSE
                   AI3=(PI-ASINCC)/EE
                  END IF
C
               END IF                          {for special case of z'=a}
               AI2=AI1-ENR*AI3 {by partial fraction, a/r(r+a)=1/r - 1/(r+a)}
            END IF                                  {for extension of plane}
      END IF                                    {for extension of each side}
C     BASIC INTEGRALS FOR LINEAR VARIATION
      AJ1=PQ2-PQ1+X*AI1
      AJ4=(PQ2-PQ1)/(PQ1*PQ2)+X*AI4
      AJ5=0.5D0*(AL*PQ2+X*(PQ1-PQ2))+0.5D0*Z2*AI1
      ALOG1=DLOG(PQ1)
      ALOG2=DLOG(PQ2)
      AJ6=AL*ALOG2+X*(ALOG1-ALOG2)-AL
      IF(IZNULL.NE.1) THEN
             IF(IANULL.NE.1) THEN
             IF(DABS(Z2-X*ALMX).LT.EPS) THEN
              ARG=DATAN(ALMX/Z)+DATAN(X/Z)
              ELSE
              ARG=DATAN(AL*Z/(Z2-X*ALMX))  {combine two inverse functions}
              IF(Z2.LT.X*ALMX .AND. X.LT.AL) ARG=PI+ARG
             END IF
             END IF
             AJ6=AJ6+Z*ARG
      END IF
      IF(IANULL.EQ.1) THEN
      AJ2=AJ1
      AJ7=AJ6
      ELSE
      ALOG1=DLOG(PQ1+ENR)
      ALOG2=DLOG(PQ2+ENR)
      AJ2=PQ2-PQ1-ENR*(ALOG2-ALOG1)+X*AI2
      AJ7=AL*ALOG2+X*(ALOG1-ALOG2)-AL+ENR*AI1
             IF((Z-ENR).GE.EPS) THEN                          {if z'.NE.a}
                 IF(X.LE.0. .OR. X.GE.AL .OR. (AA**2+BB**2).GE.1.) THEN
                  AJ7=AJ7+EE*ASINCC
                  ELSE
                  AJ7=AJ7+EE*(PI-ASINCC)
                 END IF
```

```
              END IF
       END IF
C
       AJ3=ALOG2-ALOG1+X*AI3
C      MULTIPLY FACTORS AND SUM UP CONTRIBUTION OF EACH SIDE
       IF(ICOMP.EQ.1 .OR. ICOMP.EQ.3) THEN
       IF(IANULL.EQ.1) THEN
       SPT=SPT+B*(SIGMA(J)*AI2+ALPHA(J)*AJ2)
      &       +GSEM(J)*AJ5
       ELSE
       SPT=SPT+B*(SIGMA(J)*AI2+ALPHA(J)*AJ2)
      &       +GSEM(J)*(AJ5-ENR*AJ7)
       END IF
       DPT=DPT+B*(AMU(J)*AI3 +BETA(J)*AJ3)
      &       +GDEM(J)*AJ7
       END IF
       IF(ICOMP.EQ.2 .OR. ICOMP.EQ.3) THEN      {selection of calculations}
          SVEL1=SVEL1+B*(SIGMA(J)*AI3+ALPHA(J)*AJ3)+GSEM(J)*AJ7
          SVEL2X=SVEL2X+EM(1,J)*(SIGMA(J)*AI1+ALPHA(J)*AJ1)
          SVEL2Y=SVEL2Y+EM(2,J)*(SIGMA(J)*AI1+ALPHA(J)*AJ1)
          SVEL2Z=SVEL2Z+EM(3,J)*(SIGMA(J)*AI1+ALPHA(J)*AJ1)
          SVEL3=SVEL3+B*AI2
          DVEL1X=DVEL1X+D(1)*(AMU(J)*AI4+BETA(J)*AJ4)
          DVEL1Y=DVEL1Y+D(2)*(AMU(J)*AI4+BETA(J)*AJ4)
          DVEL1Z=DVEL1Z+D(3)*(AMU(J)*AI4+BETA(J)*AJ4)
          DVEL2=DVEL2+B*AI3
          DVEL3=DVEL3+GDEM(J)*AI1
       END IF
C
   30 CONTINUE                                         {end of side-loop}
C
       DPT=ISIGN*DPT
       IF(ICOMP.EQ.1 .OR. ICOMP.EQ.3) THEN
          SPOT(K)=-R4PI*SPT                            {source-potential}
          DPOT(K)=+R4PI*DPT                           {doublet-potential}
       END IF
       IF(ICOMP.EQ.2 .OR. ICOMP.EQ.3) THEN
          SVX(K) =-R4PI*(ISIGN*AN(1)*SVEL1+SVEL2X+A1*SVEL3)
          SVY(K) =-R4PI*(ISIGN*AN(2)*SVEL1+SVEL2Y+A2*SVEL3)
          SVZ(K) =-R4PI*(ISIGN*AN(3)*SVEL1+SVEL2Z+A3*SVEL3)
          DVX(K) =+R4PI*(DVEL1X+ISIGN*B1*DVEL2-AN(1)*DVEL3)
          DVY(K) =+R4PI*(DVEL1Y+ISIGN*B2*DVEL2-AN(2)*DVEL3)
          DVZ(K) =+R4PI*(DVEL1Z+ISIGN*B3*DVEL2-AN(3)*DVEL3)
```

```
      END IF
C
      END IF                             {end of control of far-field approx.}
  100 CONTINUE                                       {end of field-point loop}
      RETURN
      END
C

C--------------------- Subroutine PRgeom.ftn ----------------------------
C                                                                         |
C     PRgeom code provides the geometric parameters of a planar or        |
C     non-planar element for the calculation of potential and velocity    |
C     induced by constant plus linear source/doublet distributions on a   |
C     planar polygon element, by adapting the formulation of Lee ~(1990)  |
C     to follow the formulation of Cantaloube and Rehbach ~(1986) and to  |
C     perform the analytic evaluations of the associated line integrals   |
C     (Suh ~(1990)).                                                      |
C     This subroutine may be called before the subroutine PRpan for       |
C     computing the induced potentials and velocities is called, but      |
C     only once for each element.                                         |
C                                                                         |
C     REFERENCES:                                                         |
C      [1] Cantaloube, B. and Rehbach, C. ~(1986)                         |
C         " Calcul des Integrales de la Methode des Singularites"         |
C           Recherche Aerospatiale, no. 1, pp. 15-22,                     |
C           (English Title: " Calculation of the Integrals of the         |
C           Singularity Method" Aerospace Research, No. 1, pp. 15-22)     |
C      [2] Suh, J. C. ~(1990)                                             |
C         " Review of the Paper; Calculation of the Integrals of the      |
C           Singularity Method by Cantaloube and Rehbach"                 |
C           KRISO Propulsor Technology Laboratory Report, 22-90           |
C      [3] Suh, J. C. ~(1990)                                             |
C         " Analytic Evaluations of the Induction-Integrals for           |
C           Distributions of Sources and Doublets over a Planar           |
C           Polygon Element"                                              |
C           KRISO Propulsor Technology Laboratory Report, 23-90           |
C      [4] Lee, C.-S. ~(1990)                                             |
C         " Treatment of non-planar panel"                                |
C           Technical Notes (unpublished), 11/29/90                       |
C                                                                         |
*-------------------------------------------------------------------------|
*                                                                         |
*     Version 1.2, November 16, 1990    by Y.-G. Kim                      |
```

```
*     Version 2.2, November 28, 1990    by C.-S. Lee                 |
C     (add a control for planar-element case on 1/4/91 by J.-C. Suh)    |
C                                                                      |
*----------------------------------------------------------------------|
*                                                                      |
*     INPUT :  Vertex Points:  xv(i,j), where i=1,3 for x,y,z-coords    |
*                                          j=1,...,Nside for sides.    |
*                                                                      |
*     OUTPUT:  Panel geometrical data, including                       |
*              cg, el, em, an, segl,area,diagnl                        |
*                                                                      |
*  ...Panel Geometry...................................................|
*      1)  Counter-clockwise when               4 *-----* 3            |
*            viewed from fluid                    |     |              |
*      2)  Dipole axis pointing                   |     |              |
*            into the fluid                 J = 1 *-----* 2            |
*  ....................................................................|
*                                                                      |
C     Input argument descriptions:                                     |
C      NSIDE              = number of sides of a planar element        |
C                           (at least 3).                              |
C      XV(i,N)            = x-, y- and z-coordinates of the            |
C                           NSIDE vertices of the element              |
C                           (two-dimensional arrayed values in a       |
C                           counterclockwise order).                   |
C      IPLANE             = control index for planar or non-planar     |
C                           element (if planar element, IPLANE=1)      |
C                                                                      |
C     Output argument descriptions:                                    |
C      SEGL(N)            = lengths of the NSIDE respective sides      |
C                           (NSIDE arrayed values).                    |
C      AN(i)              = x-, y- and z-components of the unit outward |
C                           normal vector of the planar element        |
C                           (3 arrayed values).                        |
C      EL(i,N)            = x-, y- and z-components of the unit        |
C                           orientation vector of the NSIDE respective |
C                           sides (two-dimensional arrayed values).    |
C      EM(i,N)            = x-, y- and z-components of the unit        |
C                           vector lying on the plane and normal to the |
C                           NSIDE respective sides                     |
C                           (two-dimensional arrayed values).          |
C      CG(i)              = x-, y- and z- coordinates of the centroid  |
C                           of the element (3 arrayed values).         |
```

```
 C      AREA              = surface area of the element.                 |
 C      DIAGNL            = longest diagonal of the element.             |
 C                                                                       |
 C     Notice: Please contact C.-S. Lee at CNU by a letter or by         |
 C             a phone call at (042) 821-6623 if there are any problems  |
 C             for usage of this subroutine, so that he can give notice  |
 C             of them to other users.                                   |
 C                                                                       |
 C-----------------------------------------------------------------------
 C

      SUBROUTINE PRGEOM (NSIDE,XV,IPLANE,SEGL,AN,EL,EM,CG,AREA,DIAGNL)
                 {input arguments}{output to be used in other subroutines}
      PARAMETER (NSDMAX=4)                   {up to a quadrilateral panel}
      DIMENSION AN(3),CG(3),SEGL(NSDMAX),EL(3,NSDMAX),EM(3,NSDMAX),
     &          XV(3,NSDMAX),Xip(3,4),Xgl(3),UL(3),VL(3)
      DATA QUART / 0.25D+00 /, HALF/ .5D+00/
      DATA ZERO/0.D+00/,ONE/1.D+00/,THREE/3.D+00/,FOUR/4.D+00/
C     CHECK NUMBER OF SIDES OF A POLYGON
      IF (NSIDE.LT.3 .OR. NSIDE.GT.NSDMAX) THEN
      WRITE(*,*) 'ERROR: NUMBER OF SIDE SHOELD BE AT LEAST 3 ',
     &                   'AND AT MOST ',NSDMAX,'.'
      STOP       {terminate process in the case of out of range for NSIDE}
      END IF
C
      TOL=1.0D-07
C      {tolerance for checking that neighboring vertices are near}
C
      IF(NSIDE.EQ.3 .OR. IPLANE.EQ.1) THEN
C                   ----------- FOR A PLANAR ELEMENT ---------
C     UNIT OUTWARD VECTOR NORMAL TO THE ELEMENT
      DO 10 I=1,3
      UL(I)=(XV(I,2)-XV(I,1))
   10 VL(I)=(XV(I,3)-XV(I,1))            {with only 3 vertices of polygon}
      CALL CROSS(UL,VL,AN)                                {cross product}
      ANS=SQRT(PRDOT(AN,AN))                                {dot product}
      IF(ANS.LT.TOL) THEN                           {check small magnitude}
      IER=0
      RETURN
      END IF
      DO 20 I=1,3
   20 AN(I)=AN(I)/ANS                                   {unit normal vector}
C     DEFINE ASSOCIATED UNIT VECTORS
```

```
      DO 50 J=1,NSIDE                            {loop for sides of element}
      J1=J+1
      IF(J.EQ.NSIDE) J1=1
      DO 30 I=1,3
   30 UL(I)=XV(I,J1)-XV(I,J)
      SEGL(J)=SQRT(PRDOT(UL,UL))
      IF(SEGL(J) .GT. TOL) THEN                  {check tiny length of side}
       DO 40 I=1,3
   40  EL(I,J)=UL(I)/SEGL(J)    {el: unit directional vectors along sides}
       CALL CROSS(AN,EL(1,J),EM(1,J))              {unit vector em = n x el}
      ELSE
       IER=1
      END IF
   50 CONTINUE
C     CHARACTERISTIC LENGTH, AREA, CENTROID
      IF(NSIDE.EQ.3) THEN                                {triangular element}
       S=0.5D0*(SEGL(1)+SEGL(2)+SEGL(3))
       AREA=SQRT(S*(S-SEGL(1))*(S-SEGL(2))*(S-SEGL(3)))  {Helon’s formula}
       DO 60 I=1,3
   60  CG(I)=(XV(I,1)+XV(I,2)+XV(I,3))/3.D0
       DIAGNL=AMAX1(SEGL(1),SEGL(2))
       DIAGNL=AMAX1(SEGL(3),DIAGNL)
      ELSE                                            {quadrilateral element}
       DIAG1=(XV(1,3)-XV(1,1))**2+(XV(2,3)-XV(2,1))**2
     &      +(XV(3,3)-XV(3,1))**2
       DIAG2=(XV(1,4)-XV(1,2))**2+(XV(2,4)-XV(2,2))**2
     &      +(XV(3,4)-XV(3,2))**2
       DIAGNL=SQRT(DIAG1)                 {separate it into two triangles}
       S=0.5D0*(DIAGNL+SEGL(1)+SEGL(2))                    {first triangle}
       AREA=SQRT(S*(S-DIAGNL)*(S-SEGL(1))*(S-SEGL(2)))
       DO 70 I=1,3
   70  VL(I)=   AREA*(XV(I,1)+XV(I,2)+XV(I,3))/3.D0            {1st moment}
       S=0.5D0*(DIAGNL+SEGL(3)+SEGL(4))                   {second triangle}
       S=SQRT(S*(S-DIAGNL)*(S-SEGL(3))*(S-SEGL(4)))
       DO 80 I=1,3
   80  VL(I)=VL(I)+S*(XV(I,3)+XV(I,4)+XV(I,1))/3.D0
       AREA=AREA+S
       DO 90 I=1,3
   90  CG(I)=VL(I)/AREA
       IF(DIAG2.GT.DIAG1) DIAGNL=SQRT(DIAG2)
      END IF
C
      ELSE                                          {for non-planar element}
```

```
 C
 C                    ----------- FOR A NON-PLANAR ELEMENT ---------
 C (method: projection of non-planar surface
 C                             onto a mean planar surface)
*----------------------------------------------------------------------
*     EVALUATION OF REFERENCE COORDINATES OF THE ORIGIN OF THE LOCAL
*     COORDINATE SYSTEM.
*     ORIGIN AT AVERAGE POINT; FIRST APPROXIMATION.
*----------------------------------------------------------------------
*
      do 100 i = 1,3
        V12=xv(i,1)+xv(i,2)
        V23=xv(i,2)+xv(i,3)
        V34=xv(i,3)+xv(i,4)
        V41=xv(i,4)+xv(i,1)
        CG(I)=(V12+V34)*QUART
        UL(I)=(V23-V41)*HALF
        VL(I)=(V34-V12)*HALF
  100    continue
      ULS=SQRT(PRdot(UL,UL))
      VLS=SQRT(PRdot(VL,VL))
      IF(ULS.LT.TOL.OR.VLS.LT.TOL) THEN
       IER=0
       RETURN
      END IF
      do 110 i = 1,3
        UL(I)=UL(I)/ULS
        VL(I)=VL(I)/VLS
  110    continue
*----------------------------------------------------------------------
*     EVALUATION OF THE UNIT VECTORS(ul,vl,wl) OF LOCAL FRAME
*----------------------------------------------------------------------
      CALL CROSS(UL,VL,AN)
      ANS = SQRT( PRdot(AN,AN) )
      IF(ANS.LT.TOL) THEN
      IER=0
      RETURN
      END IF
      do 120 i = 1,3
  120      AN(I)=AN(I)/ANS
      CALL CROSS(AN,UL,VL)
*----------------------------------------------------------------------
*     LOCAL COORDINATES OF VERTICES WITH CG AS THE TEMPORARY ORIGIN.
```

```
*     Project onto u-v plane(local planar plane) by dropping Xip(3,j).
*-----------------------------------------------------------------------
      do 140 j = 1,Nside
          do 130 i = 1,3
  130     SEGL(I)=xv(i,j)-CG(I)
        Xip(1,J)=PRdot( SEGL,UL)
        Xip(2,J)=PRdot( SEGL,VL)
  140    continue
*-----------------------------------------------------------------------
*     LOCAL COORDINATES RELATIVE TO CENTROID;
*-----------------------------------------------------------------------
      A1=(Xip(1,2)+Xip(1,3))*HALF
      A2=(Xip(1,3)+Xip(1,4))*HALF
      A3=(Xip(1,3)+Xip(1,1))*HALF
      B1=(Xip(2,2)+Xip(2,3))*HALF
      B2=(Xip(2,3)+Xip(2,4))*HALF
      B3=(Xip(2,3)+Xip(2,1))*HALF
      D0=A1*B2-A2*B1
        area = four * D0
      D1=A1*B3-A3*B1
      D2=A3*B2-A2*B3
      D0I=ONE/(THREE*D0)
*.....Centroid in Temporary Coordinate System.
      Xgl(1)=(D1*A1+D2*A2)*D0I
      Xgl(2)=(D1*B1+D2*B2)*D0I
*.....Xip is now relative to the centroid.
      do 160 j = 1,Nside
          do 150 i = 1,2
  150        Xip(I,J)=Xip(I,J) - Xgl(I)
  160        CONTINUE
*-----------------------------------------------------------------------
*     RE-EVALUATION OF NEW COORDINATES OF CENTROID and VERTICES
*        IN GLOBAL COORDINATE SYSTEM.
*-----------------------------------------------------------------------
      DO 165 I=1,3
  165      cg(I)= CG(I) + UL(I)*Xgl(1) + VL(I)*Xgl(2)
      do 180 j = 1,Nside
        do 170 i = 1,3
  170      xv(i,j)  = cg(i) + UL(i)*Xip(1,j)+VL(i)*Xip(2,j)
  180    continue
C     DEFINE ASSOCIATED UNIT VECTORS
      DO 210 J=1,NSIDE                         {loop for sides of element}
      J1=J+1
```

```
      IF(J.EQ.NSIDE) J1=1
      DO 190 I=1,3
  190 EL(I,J)=XV(I,J1)-XV(I,J)
      SEGL(J)=SQRT(PRDOT(EL(1,J),EL(1,J)))
      IF(SEGL(J) .GT. TOL) THEN
      DO 200 I=1,3
  200 EL(I,J)=EL(I,J)/SEGL(J)   {el: unit directional vectors along sides}
      CALL CROSS(AN,EL(1,J),EM(1,J))              {unit vector em = n x el}
      ELSE
      IER=1
      END IF
  210 CONTINUE
       DIAG1=(XV(1,3)-XV(1,1))**2+(XV(2,3)-XV(2,1))**2
     &      +(XV(3,3)-XV(3,1))**2
       DIAG2=(XV(1,4)-XV(1,2))**2+(XV(2,4)-XV(2,2))**2
     &      +(XV(3,4)-XV(3,2))**2
       DIAGNL=AMAX1(DIAG1,DIAG2)
       DIAGNL=SQRT(DIAGNL)     {longest diagonal as characteristic length}
*.......................
      END IF
C
      RETURN
      END
 C
 C      CALCULATE VECTOR CROSS PRODUCT
      SUBROUTINE CROSS(A,B,C)
       DIMENSION A(1),B(1),C(1)
       C(1)=A(2)*B(3)-A(3)*B(2)
       C(2)=A(3)*B(1)-A(1)*B(3)
       C(3)=A(1)*B(2)-A(2)*B(1)
       RETURN
       END
 C
 C      Calculates vector dot product
      FUNCTION PRdot(A,B)
       DIMENSION A(1),B(1)
       PRdot=A(1)*B(1)+A(2)*B(2)+A(3)*B(3)
       RETURN
       END
C-------------------------------------------------------{end of file}
```