PREDIK

R.E. McFarland

Marcus Phipps

March 1991

NASA

National Aeronautics and Space Administration
**Function:** This program is for use by Computer-Generated Imagery (CGI) drive programs to predict positions and angles, as required, in order to compensate for transport delay in the mainframe-to-CGI presentation path.

**Language:** FORTRAN, Computer Portable.

**Location:**
1. SIMDEV VAX, $DISK1:[STRIKE]PREDIK.FOR.
2. FSD VAX, $LIBRARY:[STRIKE]PREDIK.FOR.
3. ADDEV VAX, $DISK1:[STRIKE]PREDIK.FOR
4. Author, PC Diskette.

**References:**

**Summary:** Subroutine PREDIK standardizes the operation of the prediction algorithm. It predicts both angles and positions for the CGI drives. Since the CGI processors operate in parallel and the information received flows serially through the processors, there is a time delay between the transmittal of information and scene presentation.

Setup for the program is controlled by three BLOCK COMMON variables: ICOMPN, WTUNE, and THEORY. THEORY is the prediction interval in seconds that determines how far into the future the program will predict. WTUNE is the tuned frequency. ICOMPN must be set to unity to enable predictions. Otherwise the output CGI drive vector will be merely the uncompensated positions (they are properly transformed to "runway coordinates").

The prediction is carried out through several phases. The compensation algorithm extrapolation coefficients C0, C1, and C2 are computed in I.C.
(Initial Condition) mode. Local Frame-to-Body Frame derivatives, XPRD, YPRD, ZPRD, not found in STRIKE are generally developed (see Appendix C of Reference (1)). These quantities are currently used only by DIG1 software, although CT5A software may also require them in the future.

The velocity ladder (array CGIV with six rows of three values) contains the present, and two past values of the six (positional and rotational) rates. These values are set equal in I.C. mode. When the model goes into operate mode, the velocity values use a "ladder-down" scheme, including a "smoothed starting gate."

Reference (3) should be consulted for definitions. A brief explanation follows: For the DIG1 system, a value of about 0.0917 seconds for THEORY is appropriate, due to the fact that the three processors of this system each operate at 30 Hz. For the CT5A system, with four processors each operating at 50 Hz, the value for THEORY is appropriately 0.075 seconds. The value for WTUNE is required in rad/sec. As shown in Ref. (3), this value should be the equivalent of about 3 Hz. (18.85 rad/sec).

ICOMP is the flag that enables PREDIK's prediction logic. It may be turned on and off at will, in all modes.

Velocity data is also made available from subroutine PREDIK. This data is currently used only by the DIG1 system, wherein it is utilized by a projection (or smoothing) process within the first pipeline processor. The purpose of the "smoothing operation" is to synchronize asynchronous signals between the mainframe and CGI computer system. The operation extrapolates signals over small and variable intervals. See Ref. (2).

Using the current, past and previous values of velocity along with the current position values, and by using the coefficients computed in the algorithm development, a prediction can be made as to the positions and angles -THEORY- seconds later. This operation is standardized in subroutine PREDIK, wherein all outputs are in "runway coordinates."
Communications with BASIC COMMON

In order to communicate with COMMON arrays standardized for simulation models (BASIC), the following equivalences appear within subroutine PREDIK.

**Inputs:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Array</th>
<th>Description (inputs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHIR</td>
<td>A(4)</td>
<td>Roll Euler angle in local frame (rad)</td>
</tr>
<tr>
<td>THETR</td>
<td>A(5)</td>
<td>Pitch Euler angle in local frame (rad)</td>
</tr>
<tr>
<td>PSIR</td>
<td>A(6)</td>
<td>Yaw Euler angle in local frame (rad)</td>
</tr>
<tr>
<td>PHID</td>
<td>A(7)</td>
<td>Roll Euler rate in local frame (rad/sec)</td>
</tr>
<tr>
<td>THED</td>
<td>A(8)</td>
<td>Pitch Euler rate in local frame (rad/sec)</td>
</tr>
<tr>
<td>PSID</td>
<td>A(9)</td>
<td>Yaw Euler rate in local frame (rad/sec)</td>
</tr>
<tr>
<td>T11</td>
<td>A(16)</td>
<td>(\text{COS(THETR)} \times \text{COS(PSIR)})</td>
</tr>
<tr>
<td>T21</td>
<td>A(17)</td>
<td>(\text{SIN(PHIR)} \times \text{SIN(THETR)} \times \text{SIN(PSIR)} - \text{COS(PHIR)} \times \text{SIN(PSIR)})</td>
</tr>
<tr>
<td>T31</td>
<td>A(18)</td>
<td>(\text{COS(PHIR)} \times \text{SIN(THETR)} \times \text{COS(PSIR)} + \text{SIN(PHIR)} \times \text{SIN(PSIR)})</td>
</tr>
<tr>
<td>T12</td>
<td>A(19)</td>
<td>(\text{COS(THETR)} \times \text{SIN(PSIR)})</td>
</tr>
<tr>
<td>T22</td>
<td>A(20)</td>
<td>(\text{SIN(PHIR)} \times \text{SIN(THETR)} \times \text{SIN(PSIR)} + \text{COS(PHIR)} \times \text{COS(PSIR)})</td>
</tr>
<tr>
<td>T32</td>
<td>A(21)</td>
<td>(\text{COS(PHIR)} \times \text{SIN(THETR)} \times \text{SIN(PSIR)} - \text{SIN(PHIR)} \times \text{COS(PSIR)})</td>
</tr>
<tr>
<td>T13</td>
<td>A(22)</td>
<td>(-\text{SIN(THETR)})</td>
</tr>
<tr>
<td>T23</td>
<td>A(23)</td>
<td>(\text{SIN(PHIR)} \times \text{COS(THETR)})</td>
</tr>
<tr>
<td>T33</td>
<td>A(24)</td>
<td>(\text{COS(PHIR)} \times \text{COS(THETR)})</td>
</tr>
<tr>
<td>PT</td>
<td>A(46)</td>
<td>Total roll rate of the Local-Frame (rad/sec)</td>
</tr>
<tr>
<td>QT</td>
<td>A(47)</td>
<td>Total pitch rate of the Local-Frame (rad/sec)</td>
</tr>
<tr>
<td>RT</td>
<td>A(48)</td>
<td>Total yaw rate of the Local-Frame (rad/sec)</td>
</tr>
<tr>
<td>ALTD</td>
<td>A(80)</td>
<td>Rate of change of altitude (ft/sec)</td>
</tr>
<tr>
<td>XLOND</td>
<td>A(81)</td>
<td>Rate of change of longitude (rad/sec)</td>
</tr>
<tr>
<td>XLATD</td>
<td>A(82)</td>
<td>Rate of change of latitude (rad/sec)</td>
</tr>
<tr>
<td>XPR</td>
<td>A(103)</td>
<td>Distance of pilot eye position down the runway, Runway-Frame (ft)</td>
</tr>
<tr>
<td>YPR</td>
<td>A(104)</td>
<td>Distance of pilot eye position to the right of the runway, Runway-Frame (ft)</td>
</tr>
</tbody>
</table>
HPR  A(105)  Height of the pilot eye position above the runway, Runway-Frame (ft)
RR   A(108)  RR = (RE) radius of the earth + (HR) height of runway (ft)
CLATR A(113) Cosine of the runway latitude (XLATR)
STHETR A(114) Sine of the runway angle (THETRR)
CTHETR A(115) Cosine of the runway angle (THETRR)
DT2  A(168)  Second loop frame time (sec)
XP   A(171)  X position of pilot w/r/t C.G. (ft)
YP   A(172)  Y position of pilot w/r/t C.G. (ft)
ZP   A(173)  Z position of pilot w/r/t C.G. (ft)
ZPE  A(492)  Z Distance from eye point to motion point (+ft)

The following are control inputs to PREDIK.

<table>
<thead>
<tr>
<th>Name</th>
<th>Array</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WTUNE</td>
<td>A(488)</td>
<td>The tuned frequency in RAD/SEC. This parameter is set in BLOCK COMMON and should normally be about 18.85 rad/sec (3 HZ.)</td>
</tr>
<tr>
<td>THEORY</td>
<td>A(489)</td>
<td>The theoretical transport delay of the CGI system in seconds. The prediction interval should be about 0.0917 for the DIG. or about 0.075 for the CT5A.</td>
</tr>
<tr>
<td>IMODE</td>
<td>IA(1)</td>
<td>Mode control integer where (-1) = I.C.; (0) = HOLD; and (+1) = OPERATE.</td>
</tr>
<tr>
<td>ICOMPN</td>
<td>IA(248)</td>
<td>The enable flag for CGI visual delay compensation. The values for ICOMPN are either 0 or 1. If ICOMPN is not set, the output vector will simply consist of the values XPR, YPR, HPR, PHIR, THETR, and PSIR.</td>
</tr>
</tbody>
</table>

Outputs

Subroutine PREDIK's outputs are a specially created COMMON buffer (vector) containing XCGI, YCGI, ZCGI, PHICGI, THTCGI and PSICGI (see descriptions below), beginning in the location A(447). The positional elements of this vector emulate the pilot's position with respect to the runway. The linear derivatives of these quantities, as required by the compensation scheme, are computed in PREDIK.

When enabled (ICOMPN=1), the output vector contains the predicted positions for the transmittal to a CGI drive routine. Associated velocities are also included in this buffer. They are XPRD, YPRD, ZPRD, PHDCCI, THDCGI, and PSDCcgi. The first three derivatives are actually computed in PREDIK (because they are not provided by STRIKE.
or SMART). The last three values are provided elsewhere in the BASIC system, but are replicated within PREDIK for the convenience of the CGI driving subroutine.

<table>
<thead>
<tr>
<th>Name</th>
<th>Array</th>
<th>Description (outputs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCGI</td>
<td>A(447)</td>
<td>X Drive for CGI (ft)</td>
</tr>
<tr>
<td>YCGI</td>
<td>A(448)</td>
<td>Y Drive for CGI (ft)</td>
</tr>
<tr>
<td>ZCGI</td>
<td>A(449)</td>
<td>Z Drive for CGI (ft)</td>
</tr>
<tr>
<td>PHICGI</td>
<td>A(450)</td>
<td>Roll Drive of CGI (rad)</td>
</tr>
<tr>
<td>THTCGI</td>
<td>A(451)</td>
<td>Pitch Drive of CGI (rad)</td>
</tr>
<tr>
<td>PSICGI</td>
<td>A(452)</td>
<td>Yaw Drive of CGI (rad)</td>
</tr>
<tr>
<td>XPRD</td>
<td>A(453)</td>
<td>X velocity drive for CGI (computed in PREDIK)</td>
</tr>
<tr>
<td>YPRD</td>
<td>A(454)</td>
<td>Y velocity drive for CGI (computed in PREDIK)</td>
</tr>
<tr>
<td>ZPRD</td>
<td>A(455)</td>
<td>Z velocity drive for CGI (computed in PREDIK)</td>
</tr>
<tr>
<td>PHDCGI</td>
<td>A(456)</td>
<td>Roll velocity drive for CGI (rad/sec)</td>
</tr>
<tr>
<td>THDCGI</td>
<td>A(457)</td>
<td>Pitch velocity drive for CGI (rad/sec)</td>
</tr>
<tr>
<td>PSDCGI</td>
<td>A(458)</td>
<td>Yaw velocity drive for CGI (rad/sec)</td>
</tr>
</tbody>
</table>
PREDIK.FOR

* STRIKE POSITION PREDICTION ROUTINE FOR CGI DRIVES *

* FOR USE BY CGI DRIVE ROUTINES TO PREDICT POSITIONS AND ANGLES *
* IN THE RUNWAY FRAME. VELOCITIES PROVIDED BY STRIKE. *

SUBROUTINE PREDIK

CREATION AND MODIFICATION

VERSION 1.0 - MARCH 16, 1988 R. E. MCFARLAND - NASA -
VERSION 1.1 - MARCH 28, 1988, ASSIGNED OUTPUT VECTOR
VERSION 1.3 - DEC. 14, 1989, ASSIGNED OUTPUT VELOCITY VECTOR
VERSION 1.4 - FEB. 6, 1990, CHANGED SENSE OF ZCGI AND DERIVATIVE.
ALSO PUT IN ZPE LOGIC.
VERSION 1.5 - JAN. 31, 1991, UPDATED COMMENTS CONCERNING VALUES FOR
THEORY, AND ESTABLISHED RATE NAMES.
VERSION 1.6 - FEB. 13, 1991, PURGED OF INTERNAL DERIVATIVES. SINCE
THESE COMPUTATIONS HAVE BEEN MOVED TO STRIKE. THIS PROGRAM IS NO LONGER
COMPATIBLE WITH SMART...

SIGNIFICANT VARIABLES

VARIABLES ARRAY LOCATION & DEFINITION    UNIT    SOURCE
IMODE IA(1)   =--: I.C., =0: HOLD, =+: OPERATE.
ICOMPNN IA(248) SWITCH. 1=COMPENSATE, 0=DON'T
XPR A(103)   DIST. OF PILOT DOWN RUNWAY    FT      STRIKE
YPR A(104)   DIST. OF PILOT RIGHT OF RUNWAY  FT      ..
HPR A(105)   DIST. OF PILOT ABOVE RUNWAY   FT      ..
PHIR A(4)    ROLL ANGLE L-FRAME         RAD      STRIKE
THETR A(5)   PITCH                        ::      ::
PSIR A(6)    YAW                           ::      ::

THE FOLLOWING ARE OUTPUTS OF STRIKE, NOT OF THIS PROGRAM.
THEY ARE INPUTS TO THIS PROGRAM (AND SMART DOES NOT COMPUTE THEM).

XPRD A(453)   PILOT POSITIONAL RATES    FT/SEC
YPRD A(454)   ..
ZPRD A(455)   ..
PHDCGI A(456) PILOT ANGULAR RATES       RAD/SEC
THDCGI A(457) ..
PSDCGI A(458) ..

DT2 A(168)   CYCLE TIME                   SEC      DATA
WTUNE A(488) EQUIVALENT TO ABOUT 3 HZ. R/S      ..
THEORY A(489) VARIES WITH SYSTEM. SEE NOTES. SEC      ::

OUTPUTS (DESTINATION: BVISUAL)

LINEAR AND ANGULAR POSITIONS ARE PREDICTED
THEORY SECONDS INTO THE FUTURE.

XCGI A(447)   PREDICTED PILOT POSITIONS FT
YCGI A(448)   ..
ZCGI A(449)   ..
PHICGI A(450) PREDICTED PILOT ANGLES RAD
THTCGI A(451) ..

PREDIK - I -
PSICGI A(452) ..

PREDICTED POSITIONS

(1) XCGI - PILOT X POSITION W/R/T RUNWAY (FT) (CGIDR(1))
(2) YCGI - PILOT Y POSITION W/R/T RUNWAY (FT)
(3) ZCGI - PILOT Z POSITION W/R/T RUNWAY (FT) (+ DOWN)
(4) PHICGI - ROLL ATTITUDE (RAD)
(5) THTCGI - PITCH ATTITUDE (RAD)
(6) PSICGI - YAW ATTITUDE (RAD)

THE ABOVE SIX-VECTOR OCCUPIES A(447) TO A(452). IN CELLS A(453) TO
A(458) THE RATES OF CHANGE OF THESE QUANTITIES APPEARS - AS COMPUTED
BY STRIKE. THESE ARE THE RATES THAT SHOULD BE SENT TO THE CGI.
(THES RATES ARE NOT PREDICTED) CURRENTLY, ONLY THE DIG SYSTEM
USES RATES. THE CT5A SYSTEM, IN ITS ATTEMPT TO ACCOMMODATE
ASYNCHRONOUS DELAY, USES DIFFERENCES IN POSITIONS.

-----------------------------------------------
LOCAL
-----------------------------------------------

CGIV(6,3) CURRENT AND TWO PAST VALUES OF 6 VELOCITIES.
KSTART USED INTERNALLY TO AVOID JUMP UPON ENTERING OPERATE MODE.
C0,C1,C2 THE COMPENSATION COEFFICIENTS. THESE ARE FUNCTIONS OF
WTUNE, THEORY AND DT2 (SEE REFERENCES).

-----------------------------------------------
COMMON/XFLOAT/A(500)/IFIXED/IA(250)
-----------------------------------------------

EQUIVALENCE VALUES

EQUIVALENCE (A(4), PHIR )
EQUIVALENCE (A(5), THETR )
EQUIVALENCE (A(6), PSIR )
EQUIVALENCE (A(103), XPR )
EQUIVALENCE (A(104), YPR )
EQUIVALENCE (A(105), HPR )
EQUIVALENCE (A(168), DT2 )

THIS 6-VECTOR IS THE OUTPUT CGI DRIVE VECTOR, ASSIGNED 3/28/88
EQUIVALENCE (A(447), XCGI, CGIDR(1))
EQUIVALENCE (A(448), YCGI)
EQUIVALENCE (A(449), ZCGI)
EQUIVALENCE (A(450), PHICGI)
EQUIVALENCE (A(451), THTCGI)
EQUIVALENCE (A(452), PSICGI)

VELOCITIES ARE NOT 'PREDICTED'. THIS IS IMPORTANT. ONLY THE
CURRENT VALUE OF VELOCITIES ARE USED IN THE DIG SYSTEM, AND THESE
VALUES ARE TO 'PROJECT' THROUGH THE ASYMMETRIC DELAY CAUSED BY
THE DOUBLE-BUFFER SCHEME IN THE FIRST PIPELINE PROCESSOR.
THE ABOVE STATEMENT CONCERNS ONLY THE DIG SYSTEM. PERHAPS LATER
THE CT5A SYSTEM WILL ALSO USE THESE VELOCITY VALUES.

THE BVIS (AL) PROGRAM SHOULD USE THE FOLLOWING VELOCITIES FOR
TRANSMIT TO CGI (DIG):

THOSE ARE COMPUTED IN STRIKE (NOT SMART):
EQUIVALENCE (A(453), XPRD)
EQUIVALENCE (A(454), YPRD)
EQUIVALENCE (A(455), ZPRD)
EQUIVALENCE (A(456), PHDCGI)
EQUIVALENCE (A(457), THDCGI)
EQUIVALENCE (A(458), PSDCGI)

NOTE THAT THE OUTPUTS ARE A LONG, CONVENIENT VECTOR, FOR USE IN
THE VISUAL DRIVE ROUTINE. PREDIK SHOULD BE CALLED BEFORE THE
CALL TO A VISUAL DRIVE ROUTINE. ----
THE VECTOR IS 12 LONG.

CONTROL INPUTS TO PREDIK ARE ICOMPN, W TUNE AND THEORY.

ICOMP N = 1 TO ENABLE PREDICTIONS. IF ICOMP N IS NOT SET THE
OUTPUT VECTOR (ABOVE) WILL SIMPLY CONSIST OF THE
NON-PREDICTED VALUES (HENCE, GENERAL APPLICABILITY).

WTUNE THE TUNED FREQUENCY IN RAD/SEC. THIS IS SET IN BLOCK
COMMON AND SHOULD BE ABOUT 18.8 RAD/SEC (3 HZ).

THEORY THE PREDICTION INTERVAL IN SECONDS. THIS SHOULD BE ABOUT:

THEORY = 0.0917 FOR THE D.I.G. SYSTEM (SEC)
THEORY = 0.0750 FOR THE CT5A SYSTEM (SEC)

THESE NUMBERS COME ABOUT FROM:

THEORY = (NUMBER OF PROCESSORS - 1/4)*(CYCLE TIME OF A PROCESSOR)

A REMINDER: THE STRIKE SYSTEM NEVER SETS COMMON VALUES WITHIN
INDIVIDUAL ROUTINES. THEY ARE ONLY SET WITHIN A
BLOCK DATA PROGRAM, OR RAD FILES, OR MANUALLY, IF YOU
LIKE TO TYPE...

THIS PROGRAM IS NOT COMPATIBLE WITH -SMART- BECAUSE SMART DOES NOT
COMPUTE THE REQUIRED DERIVATIVES.

EQUIVALENCE (A(488), W TUNE)
EQUIVALENCE (A(489), THE ORY)
EQUIVALENCE (IA(241), IMODE)
EQUIVALENCE (IA(248), ICOMP N)

-----------------------------------------------

DECLARATIONS

DIMENSION CGIDR(6), CGIV(6,3)

-----------------------------------------------

NO DATA

-----------------------------------------------

REFERENCES

NASA TM 100084, JAN. 1988 BY R. E. MCFARLAND:
TRANSPORT DELAY COMPENSATION FOR COMPUTER-GENERATED IMAGERY SYSTEMS

NASA TM 86703, JAN. 1986 BY R. E. MCFARLAND:
CGI DELAY COMPENSATION

NASA CR 2497, JAN. 1975 BY R. E. MCFARLAND:
A STANDARD KINEMATIC MODEL FOR FLIGHT SIMULATION AT NASA/AMES
RESEARCH CENTER

FSE PROGRAM SUMMARY #2.01, JAN. 1991, BY MCFARLAND AND PHIPPS

-----------------------------------------------

EXECUTABLE CODE

IF(IMODE.GT.0) GO TO 30
IF(IMODE.EQ.0) RETURN

-----------------------------------------------

I.C. MODE

IF((WTUNE*THEORY).LE.0.0) THEN

PREDIK -3-
C COMPUTE CGI COMPENSATION PARAMETERS
THEX = WTIME*DT2
PSIX = WTIME*THEORY
CT = COS(THEX)
ST = SIN(THEX)
CP = COS(PSIX)
SP = SIN(PSIX)
OMCT = 1.0 - CT
DEN = 1.0/(2.0*WTIME*ST*OMCT)
C0 = DEN*(ST*(PSIX + SP*(1.0 - 2.0*CT))
     + (0.5*THEX*ST - CP*OMCT)*(1.0 + 2.0*CT))
C1 = DEN*ST*(2.0*(ST*CP + CT*SP)
     - 2.0*PSIX*CT - THEX*(1.0 + CT))
C2 = DEN*(ST*(PSIX - SP + 0.5*THEX) - CP*OMCT)

10 CGIDR(1) = XPR
    CGIDR(2) = YPR
    CGIDR(3) = HPR
    CGIDR(4) = PHIR
    CGIDR(5) = THETR
    CGIDR(6) = PSIR

C INITIALIZE VELOCITY LADDER, BUT DO NOT PREDICT
C POSITIONS (USING THESE VELOCITIES) IN I.C. MODE
CGIV(1,1) = XPRD
CGIV(2,1) = YPRD
CGIV(3,1) = ZPRD
CGIV(4,1) = PHIRD
CGIV(5,1) = THETRD
CGIV(6,1) = PSIRD

C DO 20 J=2,3
20 DO 20 I=1,6
    CGIV(I,J) = CGIV(I,1)

C KSTART = 1
RETURN

----------------------------- OPERATE MODE -----------------------------

30 CONTINUE

C LADDER DOWN VELOCITY VALUES
   DO 40 I=1,6
40    CGIV(I,3) = CGIV(I,2)

C CGIV(1,1) = XPRD
CGIV(2,1) = YPRD
CGIV(3,1) = ZPRD
CGIV(4,1) = PHIRD
CGIV(5,1) = THETRD
CGIV(6,1) = PSIRD

C PICKUP NEW BASELINE POSITIONAL VALUES
CGIDR(1) = XPR
CGIDR(2) = YPR
CGIDR(3) = HPR
CGIDR(4) = PHIR
CGIDR(5) = THETR
CGIDR(6) = PSIR

C CHECK FOR OPERATE MODE TURN-OFF OF ICOMP
   IF(ICOMP.NE.1) GO TO 50
   KSTART = 1

PREDIK -4-
50 RETURN
C IF(KSTART.EQ.0) GO TO 70
C SMOOTH STARTING GATE. ONLY ON THE 2ND AND LATER OPERATE-MODE
C PASSES ARE THREE DISTINCT VELOCITY VALUES AVAILABLE.
C KSTART = 0
DO 60 I = 1,6
60 CGIV(I,3) = 2.0*CGIV(I,2) - CGIV(I,1)
70 CONTINUE
C PREDICT VALUES -THEORY- SECONDS LATER (ADD APPROPRIATE INCREMENTS)
DO 80 I=1,6
80 CGIDR(I) = CGIDR(I) + C0*CGIV(I,1) + C1*CGIV(I,2) + C2*CGIV(I,3)
C RETURN
END
SUBROUTINE PRODIK(NPRED,ICOMP,W Cut,PIPE,PVECTC,VVECTC,DR)

CREATION AND MODIFICATION

VERSION 1.0 - FEB. 19, 1991  R. E. MCFARLAND -NASA-

SIGNIFICANT VARIABLES

VARIABLES ARRAY LOCATION & DEFINITION UNIT SOURCE
IMODE IA( 1) = : I.C., = : HOLD, = : OPERATE.
DT2 A(168) CYCLE TIME SEC

IN CALLING SEQUENCE:
NPRED DESIGNATES OBJECT NUMBER FROM 1 TO 5
ICOMP SWITCH, 1 = COMPENSATE, 0 = DON'T
WCUT EQUIVALENT TO ABOUT 3 HZ. R/S
PIPE VARIES WITH SYSTEM. SEE NOTES. SEC
PVECTC VECTOR OF POSITIONS AND ANGLES FT & RAD
VVECTC VECTOR OF RATES OF ABOVE FPS AND RAD/SEC

OUTPUTS (DESTINATION: BVISUAL)

LINEAR AND ANGULAR POSITIONS ARE PREDICTED
PIPE SECONDS INTO THE FUTURE.

DR 6-VECTOR OF PREDICTED POSITIONS AND ANGLES

LOCAL

CGIV(6,3,5) CURRENT AND TWO PAST VALUES OF 6 VELOCITES, UP TO 5 OBJECTS.
KSTART(5) USED INTERNALLY TO AVOID JUMP UPON ENTERING OPERATE MODE.
C0(5) THE COMPENSATION COEFFICIENTS. THESE ARE FUNCTIONS OF
C1(5) WCUT, PIPE AND DT2 (SEE REFERENCES).
C2(5)

COMMONS

COMMON/XFLOAT/A(500)/IFIXED/IA(250)

EQUIVALENCES

EQUIVALENCE (A(168), DT2 )
EQUIVALENCE (IA( 1), IMODE)
VELOCITIES ARE NOT 'PREDICTED'. THIS IS IMPORTANT. ONLY THE CURRENT VALUE OF VELOCITIES ARE USED IN THE DIG SYSTEM, AND THESE VALUES ARE TO 'PROJECT' THROUGH THE ASYMMETRIC DELAY CAUSED BY THE DOUBLE-BUFFER SCHEME IN THE FIRST PIPELINE PROCESSOR.
THE ABOVE STATEMENT CONCERNS ONLY THE DIG SYSTEM. PERHAPS LATER THE CTSA SYSTEM WILL ALSO USE THESE VELOCITY VALUES.

CONTROL INPUTS TO PRODIK ARE ICOMP, WCUT, AND PIPE.

ICOMP = 1 TO ENABLE PREDICTIONS FOR OBJECT NUMBER NPRED.
IF ICOMP IS NOT SET THE OUTPUT VECTOR (ABOVE) WILL SIMPLY CONSIST OF THE NON-PREDICTED VALUES (HENCE, GENERAL APPLICABILITY).

WCUT
THE TUNED FREQUENCY IN RAD/SEC. THIS SHOULD BE ABOUT 18.8 RAD/SEC (3 HZ).

PIPE
THE PREDICTION INTERVAL IN SECONDS. THIS SHOULD BE ABOUT:
PIPE = 0.0917 FOR THE D.I.G. SYSTEM (SEC)
PIPE = 0.0750 FOR THE CTSA SYSTEM (SEC)

THESE NUMBERS COME ABOUT FROM:
PIPE = (NUMBER OF PROCESSORS - 1/4)*(CYCLE TIME OF A PROCESSOR)

A REMINDER: THE STRIKE SYSTEM NEVER SETS COMMON VALUES WITHIN INDIVIDUAL ROUTINES. THEY ARE ONLY SET WITHIN A BLOCK DATA PROGRAM, OR RAD FILES, OR MANUALLY, IF YOU LIKE TO TYPE...

THIS PROGRAM IS NOT COMPATIBLE WITH -SMART- BECAUSE SMART DOES NOT COMPUTE THE REQUIRED DERIVATIVES FOR OBJECT #1 (OWNSHIP).

---------------------------------------------------------------------

DECLARATIONS
---------------------------------------------------------------------

DIMENSION PVECTC(6),VVECTC(6),DR(6),CGIV(6,3,5)
DIMENSION C0(5),C1(5),C2(5),KSTART(5)

---------------------------------------------------------------------

NO DATA
---------------------------------------------------------------------

REFERENCES
---------------------------------------------------------------------

NASA TM 100084, JAN. 1988 BY R. E. MCFARLAND:
TRANSPORT DELAY COMPENSATION FOR COMPUTER-GENERATED IMAGERY SYSTEMS

NASA TM 86703, JAN. 1986 BY R. E. MCFARLAND:
CGI DELAY COMPENSATION

FSE PROGRAM SUMMARY #2.01, JAN. 1991, BY MCFARLAND AND PHIPPS

---------------------------------------------------------------------

EXECUTABLE CODE
---------------------------------------------------------------------

IF(IMODE.GT.0) GO TO 130
IF(IMODE.EQ.0) RETURN

---------------------------------------------------------------------

I.C. MODE

IF((WCUT*PIPE).LE.0.0) THEN
  C0(NPRED) = 0.0

PRODIK -2-
\begin{verbatim}
C1(NPRED) = 0.0
C2(NPRED) = 0.0
GO TO 70
END IF

COMPUTE CGI COMPENSATION PARAMETERS
THEX = WCUT*DT2
PSIX = WCUT*PIPE
CT = COS(THEX)
ST = SIN(THEX)
CP = COS(PSIX)
SP = SIN(PSIX)
OMCT = 1.0 - CT
DEN = 1.0/(2.0*WCUT*ST*OMCT)
C0(NPRED) = DEN*(ST*(PSIX + SP*(1.0 - 2.0*CT))
  + (0.5*THEX*ST - CP*OMCT)*(1.0 + 2.0*CT))
C1(NPRED) = DEN*ST*(2.0*(ST*CP + CT*SP)
  - 2.0*PSIX*CT - THEX*(1.0 + CT))
C2(NPRED) = DEN*(ST*(PSIX - SP + 0.5*THEX) - CP*OMCT)

70 CONTINUE
DO 80 I=1,6
   DR(I) = PVECTC(I)
80 CONTINUE

INITIALIZE VELOCITY LADDER, BUT DO NOT PREDICT
POSITIONS (USING THESE VELOCITIES) IN I.C. MODE

DO 90 I=1,6
   CGIV(I,1,NPRED) = VVECTC(I)
90 CONTINUE

DO 100 J=2,3
   DO 100 I=1,6
100 CONTINUE
   CGIV(I,J,NPRED) = VVECTC(I)

KSTART(NPRED) = 1
RETURN

---------------------------------------------------------------
OPERATE MODE
---------------------------------------------------------------

130 CONTINUE

LADDER DOWN VELOCITY VALUES (NEED CURRENT AND 2 PAST VALUES)
DO 140 I=1,6
   CGIV(I,3,NPRED) = CGIV(I,2,NPRED)
   CGIV(I,2,NPRED) = CGIV(I,1,NPRED)
140 CONTINUE

DO 150 I=1,6
150 CONTINUE
   CGIV(I,1,NPRED) = VVECTC(I)

PICKUP NEW BASELINE POSITIONAL VALUES

DO 160 I=1,6
160 CONTINUE
   DR(I) = PVECTC(I)

CHECK FOR OPERATE MODE TURN-OFF OF ICOMP
IF(ICOMP.EQ.1) GO TO 170
   KSTART(NPRED) = 1
RETURN

170 CONTINUE

IF(KSTART(NPRED).EQ.0) GO TO 190

SMOOTH STARTING GATE, ONLY ON THE 2ND AND LATER OPERATE-MODE
PASSES ARE THREE DISTINCT VELOCITY VALUES AVAILABLE.
KSTART(NPRED) = 0
DO 180 I = 1,6
180 CONTINUE
   CGIV(I,3,NPRED) = 2.0*CGIV(I,2,NPRED) - CGIV(I,1,NPRED)

190 CONTINUE

PREDICT VALUES -PIPE- SECONDS LATER (ADD APPROPRIATE INCREMENTS)
DO 200 I=1,6

PRODIK -3-
\end{verbatim}
200 DR(I) = DR(I) + C0(NPRED)*CGIV(I,1,NPRED)
       + C1(NPRED)*CGIV(I,2,NPRED) + C2(NPRED)*CGIV(I,3,NPRED)
C
RETURN
END