International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163
Volume 1 Issue 12 (December 2014) www.ijirae.com

BALLISTIC TARGET FLIGHT TRAJECTORY SIMULATION (A MATLAB PROGRAM)

ILAPAVULURI UMAMAHESHWARRAO,
SCIENTIST,PROGRAMME-AD/RCI,
KANCHANBAGH(PO),HYDERABAD-500058,A.P,INDIA

Abstract: ***This article describes the algorithm of ballistic flight trajectory simulation that includes the Mat lab programs for the following software modules.1.Ballistic missile working model simulation2.Ballistic missile motion calculation3.Calculation of Approximation polynomial coefficients4.calculation of position, velocity and acceleration vectors on polynomial5.Air-object motion prediction6.Prediction in the atmosphere7.Outer atmosphere prediction8.Atmosphere density calculation9.a2,a3,a4 calculation. The matlab program is written for all the algorithms mentioned above. This program main feature is that if we have a launch point and impact point are given in terms of longitude and latitude , then what ever may be the launch point and impact point the matlab program gives the trajectory initially in 5 points in space then the co-ordinartes between the 5 points are all derived and the entire trajectory is derived .Also the CEP(CIRCULAR ERROR PROBABLE) is of the order of 0.0001 meter(1millimeter).The graphical plots of RANGE,VELOCITY,ACCELERATION AND ALTITUDE are included.***

Indexing terms: polynomial coefficients,position,velocity,acceleration,range,altitude,a2,a3,a4 etc.

INTRODUCTION:

The task of ballistic missile (BM)trajectory simulation is intended to obtain BM sample trajectories necessary for simulation of the process of target detection ,tracking,firing in tasks of combat action training and planning.

The basic purposes of task solution are

- Computation of the boost path of launch of the given type for the given fire distance according to a particular trajectory type(high and low trajectory).
- Obtaining BM sample trajectory at which the missile payload reaches the given fall point to a rather great accuracy.
- DESCRIPTION: The task includes the following basic subtasks:
- Simulation according to source data of a motion working model(MWM) of the BM of the given type;
- Computation of the given BM motion with its MWM received earlier.
- The Solution of a BM trajectory simulation task consists of realization of the following basic functions
- Formation of a polynomial in a launch coordinate system that approximates the BM boost path in case the source data is given as time dependence of missile flight distance and height changes.
- Formation of BM sample trajectory by fire azimuth computation;
- Determination of BM sample trajectory parameters (vectors of coordinates, velocity and acceleration in various coordinate systems) on any simulated missile flight time.

1. BM MWM SIMULATION:

The algorithm is intended to determine BM motion parameters at the end of a BP of an unknown trajectory,which is based on the specified time dependence upon distance and height of boundary(low trajectory with a minimum throw angle ,high trajectory with av maximum throw angle) andoptimum trajectories as well as on missile flight time according to specified coordinates of launch and fall points by BM trajectory with adjustment in compliance with BP end.

2. CALCULATION OF APPROXIMATION POLYNOMIAL COEFFICIENTS

The algorithm is intended to calculate approximation polynomial coefficients by coordinates of five equally spaced in time points of an observation object trajectory.(OO).

3. CALCULATION OF POSITION ,VELOCITY ANED ACCELERATION VECTORS ON PLYNOMIAL:

The algorithm is intended to predict the space object motion parameters on the trajectory fitted space object 4-th degree polynomial.

4. BM MOTION CALCULATION:

The algorithm is intended to define the BM motion parameters on the boost or passive path of trajectory and its stages on any simulated flight time.

5.AO(AIR OBJECT) MOTION PREDICTION:

The algiorithm is AO motion parameters prediction basing on space object motion polynomial at a specified time, makes it possible to get the motion polynomial for the time predicted.

6.ALGORITHM OF PREDICTION IN THE ATMOSPHERE:

The function of algorithm is as follows: Prediction algorithms of AO motion parameters on polynomial in atmosphere allow receiving AO motion parameters taking into account the influence of atmosphere resistance aerodynamic force. The algorithm nis formed so,that the integration step of motion equation is adjusted depending on value of deceleration.

Forecasting is conducted or up to achievement of time of anticipation, or up to achievement of height Hzem of an arrangement TO,or upto the moment of an output(exit) AO from an atmosphere.

7. OUTER ATMOSPHERIC PREDICTION ALGORITHM:

Space object motion parameters prediction algorithm by polynomial outer the atmosphere makes it possible to get the air object motion parameters with outer atmospheric motion equation integration step.Prediction is performed before or after the prediction time,or before the space object enterd the atmosphere.

8.ALGORITHM OF a2,a3,a4 CALCULATION:

The algorithm of calculation of the heighest coefficients of a motion polynomial allows calculating a 2,a3 and a4 polynomial coefficients on the basis of values of coord inates of a0 position and a1 velocity components ,using the equation of an AO ballistic motion equation.

9.ATMOSPHERE DENSITY CALCULATION ALGORITHM:

The algorithm of calculation of standard atmosphere density at height H makes it possible to calculate according to GOST 4401-

81:"THE STANDARD ATMOSPHERE.PARAMETERS"

All the above algorithms are coded into a matlab program.

APPENDIX-A is the matlab program for the above algorithms

```
  %MCC SOFTWARE PROJECT DESCRIPTION

%-------------------------------------------------------------------------

%STE COMPUTER SOFTWARE

%-------------------------------------------------

%ALGORITHM OF BALLISTIC FLIGHT TRAJECTORY SIMULATION

%THE FOLLOWING ARE THE ALGORITHMS

%1.BM WORKING MODEL SIMULATION:.MBM.MBR04

%2.BM MOTION CALCULATION:.MBM.MBR03

%3.CALCULATION OF APPROXIMATION POLYNOMIAL COEFFICIENTS:.MBM.SAP06

%4.CALCULATION OF POSITION,VELOCITY&ACCN.VECTORS ON POLYNOMIAL:.MBM.SAP03

%5.AIR-OBJECT MOTION PREDICTION.MBM.SAP01

%6.PREDICTION IN THE ATMOSPHERE

%7.OUTER ATMOSPHERE PREDICTION

%8.ATMOSPHERE DENSITY CALCULATION

%9.a2,a3,a4-CALCULATION

%-----------------------------------------------------------------------------------------------

%MBR04:BALLISTIC MISSILE-MISSILE WORKING MODEL SIMULATION

%THE ALGORITHM IS INTENDED TO DETERMINE BM MOTION PARAMETERS AT THE

%END OF A BP OF AN UNKNOWN TRAJECTORY, WHICH IS BASED ON THE SPECIFIED
```


%TIME DEPENDENCE UPON DISTANCE AND HEIGHT OF BOUNDARY(LOW TRAJECTORY

%WITH A MINIMUM THROW ANGLE , HIGH TRAJECTORY WITH A MAXIMUM THROW ANGLE)

%AND OPTIMUM TRAJECTORIES AS WELL AS ON MISSILE FLIGHT TIME

%ACCORDING TO SPECIFIED CO-ORDINATES OF LAUNCH AND POINTS OF FALL BY BM

%TRAJECTORY ADJUSTMENT IN COMPLIANCE WITH BP END

APPENDIX-A:MATLAB PROGRAM

```
%target p2trajectory data
% %-----------------------------------------------------
% load('target1.m');
% xtar=target1(:,5);
% ytar=target1(:,6);
% ztar=target1(:,7);
% %XTAR=xtar(4721:-1:1);
% %YTAR=ytar(4721:-1:1);
% %ZTAR=ztar(4721:-1:1);
% XTAR=xtar(1:653);
% YTAR=ytar(1:653);
% ZTAR=ztar(1:483);
% RTAR=sqrt(XTAR.^2+YTAR.^2+ZTAR.^2);
% rant=RTAR-6378000;
%-------------------------------------------------------------------------------------------------------------

%INPUT INFORMATION STRUCTURE

%BM LAUNCH CO-ORDINATES

phi2=-pi/2:0.0025*pi:pi/2;%lattitude in radians
phi1=83;%lattitude in degrees
lam2=0:0.005*pi:2*pi;%longitude in radians
lam1=76;%longitude in degrees
%h1=185000;%altitude in meters
h1=50000;
%-------------------------------------------------------------

%BM FALL CO-ORDINATES

phif2=-pi/2:0.0025*pi:pi/2;%lattitude in radians
%phif=12;%lattitude in degrees
phif=83;
lamf2=0:0.005*pi:2*pi;%longitude in radians
lamf=77;%longitude in degrees
hf=60000;%altitude in meters
%radius of BM fall (impact )area
rn=0.1:50000;%in meters
gamma=0:0.0001;%bm ballistic coefficient in mt.^2/kg;
%-----------------------------------------------------------------------
hai=37000000;%BM minimum apogee altitude in meters
hao=298200000;%BM maximum apogee altitude in meters
vkayi=1080;%minimum BM velocity in mt/sec at the BP end
vkayo=1680;%maximum BM velocity in mt/sec at the BP end
Di=0;% MINIMUM EARTH SURFACE SPHERICAL DISTANCE OF BM FLIGHT IN METERS
%Do=34150000;%MAXIMUM EARTH SURFACE SPHERICAL DISTANCE OF BM FLIGHT IN METERS
Do=30000;
ti=0:20:215;%BM time in seconds
to=215;
```



```
%di=1:60000:300000;%BM minimum distance in meters in 5 points of BP trajectory
%do=300000;% maximum in meters
di=1:2000:25000;
do=25000;
hi=1:7200:36000;%BM  minimum height in meters
ho=36000;%maximum in meters
vt=1;%BM LOW OR HIGH CALCULATED TRAJECTORY FORM
Req=6378131;%equatorial radius of the earth in meters
Rav=6371000;%average radius of the earth in meters
alfa=1/298.3;%squeeging of the earth
omegae=0.7292115085e-4;%angular velocity of earth rotation
v1k=7900;%satellite velocity in meters/sec
hatm=150000;%atmospheric altitude in meters
dAa=pi/180;%in radians
dAinc=dAa*57.3;%azimuth increment in degrees
tfli=216;%initial flight time in seconds
dt=10;%integration stepsize outside the atmosphere
dha=1000;%apogee altitude  accuracy in meters
dvkay=0.01;%velocity accuracy at BP end
dh=10000;%altitude increment in meters at the BP end
dD=1000;%Range increment in meters at BP end
hho=0.001;%accuracy of reaching surface of the earth in meters
dhai=0;%BM apogee altitude increment in meters
%-------------------------------------------------------------------------------
%AO.01

%CONVERSION  OF LAUNCH CO-ORDINATES AND BM ,BP,, APOGEE
%ALTITUDE AND VELOCITY INTERPOLATION
if(lamf>lam1)|((lamf-lam1)<180)|((lam1-lamf)>180)
d=1;
else  d=-1;
end

for i=1:401
Ddd=sin(phi1).*sin(phif);
end
for i=1:401
fdd=Ddd+cos(phi1).*cos(phif);
end
for i=1:401
DD=fdd.*cos(lamf)-lam1+d*omegae*tfli;
D=inv(cos(DD));
%D-is preliminary angular firing range
DR=D.*Rav;
end
for i=1:5
ha=hai+(hao-hai)/(Do-D).*(DR-D);
vkay=vkayi+(vkayo-vkayi)/(Do-D).*(DR-D);
t(i)=ti(i)+(to-ti(i))/(Do-D).*(DR-D);
d(i)=di(i)+(do-di(i))/(Do-D).*(DR-D);
h(i)=hi(i)+(ho-hi(i))/(Do-D).*(DR-D);
end
%------------------ ----------------------------------------------------
%AO.02

%DETERMINATION OF BM FALL (IMPACT) ACCURACY
for i=1:7
eps=rn./((Req).*(1-(alfa.*sin(phif).*sin(phif))));
eps1=eps/2;%BM IMPACT ACCURACY
end
```



```
%-------------------------------------------------------------
%AO.03
%DETERMINATION OF FIRING AZIMUTH
for i=1:401
Astar=inv(cos((sin(phif)-sin(phi1)).*cos(D)))/(cos(phi1).*sin(D));
end
if(d==-1)
AA=2*(180-57.3*Astar);
else
AA=57.3*Astar;
end
for i=1:5
x(i)=d(i);
y(i)=h(i);
z(i)=0;
end
%-------------------------------------------------------------------------------------
%AO.05:
%CALCULATION  OF  BM BP POLYNOMIAL IN SCS
dA=dAinc;
tkay=t(5);
xxx=min(x):10:max(x);
yyy=min(y):10:max(y);
zzz=zeros(1,168);
for i=1:5
   A(i)=sqrt(xxx(i).^2+yyy(i).^2+zzz(i).^2);
end
for i=1:5
   AX(i)=xxx(i);
   AY(i)=yyy(i);
   AZ(i)=zzz(i);
end
%xt=max(x):-1:min(x);
%yt=max(y):-0.1855:min(y);
%TR=min(t):0.1855:max(t);
%R=sqrt(xt.^2+yt.^2);
%for i=1:5
%       AAZ(:,i)=A(:,i);
% end
%-----------------------------------------------------
% Ax=xxx(i);
% Ay=yyy(i);
% Az=AZ(i);
% ii=0.;
% for t=1:5
%    ii=ii+1.;
%    numx=Ax;
% end
% jj=0.;
% for t=1:5
%    jj=jj+1.;
%    numy=Ay;
% end
% kk=0.;
% for t=1:5
%    kk=kk+1.;
%    numz=Az;
% end
% Amod(ii,jj,kk)=abs(numx)*abs(numy)*abs(numz);
% maxval=max(max(Amod));
```



```
% Amod=(Amod)/(maxval);
% A=Amod;
% % cwd = pwd;
% %   cd(tempdir);
% %  pack
% % cd(cwd)
%
%
%-------------------------------------------------------------

%-------------------------------------------------------------
%AO.01%polynomial factors cdaculation
%A(5)=A(4);
deltaT=t(3)-t(1);
%for i=1:4
axx(1)=AX(3);
axx(2)=(1/(12*deltaT)).*(AX(1)-AX(4))+8*AX(4)-AX(2);
axx(3)=(1/(24*(deltaT.^2))).*16.*(AX(4)+AX(2))-AX(4)-AX(1)-30*AX(3);
axx(4)=(1/(12*(deltaT.^3))).*2*(AX(2)-AX(4))+AX(5)-AX(1);
axx(5)=(1/(24*(deltaT.^4))).*-4.*(AX(4)+AX(2))+AX(4)+AX(1)+6*AX(3);
tp=t(3);
ayy(1)=AY(3);
ayy(2)=(1/(12*deltaT)).*(AY(1)-AY(4))+8*AY(4)-AY(2);
ayy(3)=(1/(24*(deltaT.^2))).*16.*(AY(4)+AY(2))-AY(4)-AY(1)-30*AY(3);
ayy(4)=(1/(12*(deltaT.^3))).*2*(AY(2)-AY(4))+AY(5)-AY(1);
ayy(5)=(1/(24*(deltaT.^4))).*-4.*(AY(4)+AY(2))+AY(4)+AY(1)+6*AY(3);
azz(1)=AZ(3);
azz(2)=(1/(12*deltaT)).*(AZ(1)-AZ(4))+8*AZ(4)-AZ(2);
azz(3)=(1/(24*(deltaT.^2))).*16.*(AZ(4)+AZ(2))-AZ(4)-AZ(1)-30*AZ(3);
azz(4)=(1/(12*(deltaT.^3))).*2*(AZ(2)-AZ(4))+AZ(5)-AZ(1);
azz(5)=(1/(24*(deltaT.^4))).*-4.*(AZ(4)+AZ(2))+AZ(4)+AZ(1)+6*AZ(3);
%-----------------------------------------------------------------------------------------
%SAP-03;
%CALCULATION OF POSITION ,VELOCITY&ACCELERATION VECTORS ON POLYNOMIAL
%INPUT INFORMATION
T0=0;%POLYNOMIAL START TIME IN SECONDS
T=200;%SPACE OBJECT MOTION PARAMETER  PREDICTION TIME IN SECONDS
 i=1:5
%a=-100000:100000;
%
% for i=1
%     ax1(i)=axx(1);
%     ay1(i)=ayy(1);
%     az1(i)=azz(1);
% end
% for i=2
%     ax2(i)=axx(2);
%     ay2(i)=ayy(2);
%     az2(i)=azz(2);
% end
% for i=3
%     ax3(i)=axx(3);
%     ay3(i)=ayy(3);
%     az3(i)=azz(3);
% end
% for i=4
%     ax4(i)=axx(4);
%     ay4(i)=ayy(4);
%     az4(i)=azz(4);
%
```



```
% end
% for i=5
%     ax5(i)=axx(5);
%     ay5(i)=ayy(5);
%     az5(i)=azz(5);
% end
for i=1:5
    adx(i)=axx(i);
ady(i)=ayy(i);
adz(i)=azz(i);
end
    for i=1:5
    aox(i)=adx(i).*(T-T0).^i;

     aoy(i)=ady(i).*(T-T0).^i;
     aoz(i)=adz(i).*(T-T0).^i;

end

for i=2:5

    a1x(i)=i.*adx(i).*(T-T0).^(i-1);
    a1y(i)=i.*ady(i).*(T-T0).^(i-1);
    a1z(i)=i.*adz(i).*(T-T0).^(i-1);
       end
 for i=3:5
a2x(i)=i.*(i-1).*adx(i).*(T-T0).^(i-2);
a2y(i)=i.*(i-1).*ady(i).*(T-T0).^(i-2);
a2z(i)=i.*(i-1).*adz(i).*(T-T0).^(i-2);
end
aoX=min(aox):100000000000000:max(aox);%X-POSITION IN T.C.S
aoY=min(aoy):100000000000000:max(aoy);%Y-POSITION IN T.C.S
aoZ=min(aoz):100000000000000000000000000:max(aoz);%Z-POSITION IN T.C.S
a1X=min(a1x):100000000000000:max(a1x);%VX-VELOCITY IN T.C.S
a1Y=min(a1y):100000000000000:max(a1y);%VY-VELOCITY IN T.C.S
a1Z=min(a1z):100000000000000:max(a1z);%VZ-VELOCITY IN T.C.S
a2X=min(a2x):100000000000000:max(a2x);%AX-ACCELERATION IN T.C.S
a2Y=min(a2y):100000000000000:max(a2y);%AY-ACCELERATION IN T.C.S
a2Z=min(a2z):100000000000000000000000000000:max(a2z);%AZ-ACCELERATION IN T.C.S
%for i=1:902
    %posx(:,i)=aoX(:,i);
 %   posy(:,i)=aoY(:,i);
%    posz(:,i)=aoZ(:,i);
  %  velx(:,i)=a1X(:,i);
   % vely(:,i)=a1Y(:,i);

%   velz(:,i)=a1Z(:,i);
   % acnx(:,i)=a2X(:,i);
    %acny(:,i)=a2Y(:,i);
  %  acnz(:,i)=a2Z(:,i);
  %end
%a(i) and aoy(i) are in start co-ordinate system
%-------------------------------------------------------------------------
%AO.07
%CONVERSION OF BM MOTION PARAMETERS  FROM SCS TO LAUNCH TCS
%TAKING EARTH ROTATION INTO ACCOUNT DURING PREDICTION ON BP
%co=[0 1 0;0 0 1;1 0 0];
%co=[zeros(1,401)  ones(1,401) zeros(1,401);zeros(1,401) zeros(1,401) ones(1,401);ones(1,401) zeros(1,401) zeros(1,401)];
%c1=[cos(lam1)  -sin(lam1)  0;sin(lam1)  cos(lam1) 0 ;0 0 1];
%c2=[sin(phi1)   0  -cos(phi1);0  1 0;cos(phi1) 0 1];
```



```
%coo=[(1:co);ones(4:21)];
%c22=[(1:c2);ones(13:21)];
%for i=1:168
 %   ct=(c1).*(c2).*(co);
 %end
%ctt=[ct;ones(4:401)];
%for i=1:5
%a2x2(i)=(ct(i)').*(ct(i)).*(a2x(i)');
%a2y2(i)=(ct(i)').*(ct(i)).*(a2y(i)');
%a2z2(i)=(ct(i)').*(ct(i)).*(a2z(i)');
%end
%for i=1:5
%a1x1(i)=(ct(i)').*(ct(i)).*(a1x(i));%velocity co-ordinates in T.C.S
%a1y1(i)=(ct(i)').*(ct(i)).*(a1y(i));%velocity co-ordinates in T.C.S
%a1z1(i)=(ct(i)').*(ct(i)).*(a1z(i));%velocity co-ordinates in T.C.S
%end
%for i=1:5
%aox1(i)=(ct(i)').*(ct(i)).*(aox(i));%position  co-ordinates in T.C.S
%aoy1(i)=(ct(i)').*(ct(i)).*(aoy(i));%position  co-ordinates in T.C.S
%aoz1(i)=(ct(i)').*(ct(i)).*(aoz(i));%position  co-ordinates in T.C.S
%end
%a1v=a1n';
%vx=a1v(:,1);
%vy=a1v(:,2);
%vz=a1v(:,3);
%-------------------------------------------------------------------------------------------
%AO.08
%ES TABLISHING INITIAL CONDITIONS  OF THE PREDICTION
tf=tkay;
ty=tfli;
ph=0;
%aoN=[aoX aoY];%POSITION
%a1N=[a1X a1Y];%VELOCITY
%-----------------------------------------------------------------------------------------------------------
%AO.09
%PREDICTION OF BM MOTION POINT IN A LAUNCH T.C.S
%DATA CONSTANTS
aeq=6378137;%semi-major ellipsoid axis  WGS-84(in meters)
beq=6356752314200;%semi minor ellipsoid axis WGS-84(IN METERS)
Rzu=6356767;%nominal earth radius used for geo-potential altitude calculation(in meters)
ee=0.00669437999013;%ellipsoid square eccentricity WGS-84
cz=0.0033528107;%earth contraction (squeezing )on WGS-84
w=0.7292115085e-4;%earth rotation angular velocity(1/sec)
kw=0.5317494234e-10;%earth rotation square angular velocity(1/sec.^2)
mu=0.3986013e+15;%gravity constant to earth mass product(in m^3*sec^-2)
c=-2.194466e+10;%earth compression ratio factor in earth gravity potential decomposition(in m^2)
go=9.80665;%equator gravity force acceleration(in m/sec^2)
Ru=287.05287;%specific gas constant(in j*(kg^-1)*(degreek.^-1)
Hg=71.8020;%geometrical height of the boundary of the layer( in meters)
FM=117.7770;%geo-potential height of the boundary of the lower layer(in meters)
TM=380.60;%LOWER LAYER BOUNDARY MOLAR TEMPERATURE( IN DEGREE KELVIN)
GM=0.0011259;%molar temperature gradient in layer 11 on height in( degreekelvin/mt)
PM=0.00266618;%pressure on the lower boundary of the layer( in pa)
T0w=5.0;%numerical integration step  at extra atmospheric track path (in seconds)
Tow1=5.0;%numerical   integration step in atmosphere from the enter point up to the point when the object
%has the deceleration value equal to Q1
Tow2=1.0;%numerical integration step in atmosphere at the point  when the object has the deceleration
%value in range  from Q1 to Q2  in seconds
Tow3=0.5;%numerical integration step in atmosphere at deceleration value more than Q2(in seconds)
Q1=2.0e-2;%deceleration boundary value for shift to integration  step(Tow2)(in m/sec^2)
```



```
Q2=2.0e-1;%deceleration boundary value at which the shift to intergration step Tow3 is made(in m/sec^2)
Hatm=150000;%height of the atmosphere in meters
DelT=1.0e-8;%polynomial prediction completion parameter for the time predicted(in seconds)
Hzem=100000;%polynomial prediction completion parameter for the time predicted after the earth surface
%has been achieved (in meters)
%-----------------------------------------------------------------------------------------------------------------------------
%AO.09
%this algorithm consists of  prediction block in the atmosphere and prediction block outside the  atmosphere
%AO.09 is based on space object motion  polynomial at a specified time makes it possible t6o get the motion polynomial
%for the time predicted
%INPUT INFORMATION
tt=0;%(assumed value)input polynomial reference time(in seconds)
%mp=   ;%input polynomial up to the fourth degree in T.C.S
sti=1:5;%input polynomial degree
st=2:4;%polynomial degree used when predicting
gamma=0:0.01;%ballistic coefficient( in m^2/kg)
tu=200;%prediction time
stu=5;%output polynomial degree
phi=((pi/2)-phi1);%geodesic T.C.S. reference lattitude to the earth surface
hh=ho;%perpendicular height over the earth ellipsoid surface
%-----------------------------------------------------------------------------------------------------------------------------
%Begining of the AO.09  algorithm
%GLOBAL CALCULATIONS  CONECTED WITH T.C.S
cf=cos(phi);
sf=sin(phi);
%for i=1:5
Ap=beq.*sin(inv(tan(beq/aeq.*tan(phi))))+h.*sf;
Bp=aeq.*cos(inv(tan(beq/aeq.*tan(phi))))+h.*cf;
bx=Ap.*sf-Bp.*cf;
by=Ap.*cf+Bp.*sf;
bz=0;
%end
%-----------------------------------------------------------------------------------------------------------------------
%algorithm of a2,a3,a4 calculation
%the algorithm of calculation of the highest coefficients of a motion polynomial allows calculating a2,a3,a4 polynomial coefficients
%on the basis of  values of co-ordinates of ao position and a1 velocity components using the equation of an A.O. ballistic motion
%equation
a(3)=0;
a(4)=0;
%-----------------------------------------------------------------------------------------------------------------------------
for i=1:168
%xx(i)=x(i);
%yy(i)=y(i);
%zz(i)=z(i);
%xx(i)=A(;,1);
%yy(i)=A(:,2);
%zz(i)=A(:,3);
r(i)=sqrt(xxx(i).^2+yyy(i).^2+zzz(i).^2);
%vv=r/tfli;
%VVX=vv';
%VVY=vv';
%VVZ=vv';
%VVX(i)=vv(:,1);
%VVY(i)=vv(:,2);
%VVZ(i)=vv(:,3);
%vvv=max(vv)-min(vv);
mubyr(i)=(mu/r(i));
end
sa=sin(phi);
as=cos(phi);
```



```
ass=as';
saa=sa';
lx=length(xxx);
lk=ones(168,1);
saz=[saa;lk];
asz=[ass;lk];
for i=1:168
xphiyphi(i)=xxx(i).*saz(i)+yyy(i).*asz(i)+zzz(i).*saz(i);
xphyphs(i)=xphiyphi(i)^2;
xbyrs(i)=3*((xphyphs(i))./(r(i).^2));
ex1(i)=(1-xbyrs(i));
ex2(i)=(c/(r(i).^2));
ex3(i)=(1-ex2(i).*ex1(i));
uph(i)=mubyr(i).*ex3(i);%potential of terrestrial gravitation
end
for i=1:168
sixc(i)=6*c*((xphiyphi(i))/(r(i).^2));
fiftc(i)=15*c*((xphyphs(i))/(r(i).^4));
trc(i)=3*c*(r(i).^2);
mubyrq(i)=(-mu./(r(i).^3));
fact1(i)=(1-trc(i)+fiftc(i)-sixc(i));
end
for i=1:168
exyz(i)=(xxx(i).*fact1(i));
eyz(i)=(yyy(i).*fact1(i));
ezz(i)=(zzz(i).*fact1(i));
end
ln=length(yyy);
nc=ones(size(yyy));
TBG=[cos(phi);sin(phi);0];
%agx=zeros(1,168);
tg=[TBG:ones(4,168)];
for i=1:168
   agx(i)=tg.*mubyrq(i).*exyz(i);
agy(i)=mubyrq(i).*eyz(i).*tg;
agz(i)=mubyrq(i).*ezz(i).*tg;
end
AGX=agx';
AGY=agy';
AGZ=agz';
AG=sqrt(AGX.^2+AGY.^2+AGZ.^2);%accn.vector component in T.C.S. stipulated by terrestrial gravitation
%-----------------------------------------------------------------------------------------------------------------------------------------------------------

%calculation of  acceleration component stipulated by inertia centrifugal force
%algorithm
for i=1:168
   sicos(i)=(1-(xphiyphi(i))/(r(i).^2));
end
for i=1:168
   c(i)=0.5*(omegae.^2).*(r(i).^2-xphiyphi(i));%potential corresponding to inertia
end
for i=1:168
acx(i)=(omegae.^2.*(xxx(i)-xphiyphi(i)).*tg);
acy(i)=(omegae.^2.*(yyy(i)-xphiyphi(i)).*tg);
acz(i)=(omegae.^2.*(zzz(i)-xphiyphi(i)).*tg);
end
ACX=acx';
ACY=acy';
ACZ=acz';
AC=sqrt(ACX.^2+ACY.^2+ACZ.^2);
```



```
%-------------------------------------------------------------------------------------------------------------------------------------------------------
A1n1=a1X(:,1);%VX
A1n2=a1Y(:,1);%VY
%A1n3=a1Z(:,3);%VZ
A1=sqrt(A1n1.^2+A1n2.^2);
%A1--THE FIRST POLYNOMIAL COEFFICIENTS AND VELOCITY COMPONENTS
%-------------------------------------------------------------------------------------------------------------------------------------------------------
nf=ones(size(as));
TMG=[nf nf  -as;nf nf sa;as -sa nf];
om=[omegae:ones(2,168)];
tm=[TMG:ones(4,168)];
for i=1:168
omega=om.*tm;
end
for i=1:168
akx(i)=2*om.*a1X(i);
end
for i=1:168
aky(i)=2*om.*a1Y(i);
%akz(i)=2*omega(i).*A1n3;
end
AKX=akx';
AKY=aky';
%AKZ=akz';
for i=1:168
AK(i)=sqrt(AKX(i).^2+AKY(i).^2);
end
%AK IS THE ACCN . COMPONENT STIPULATED BY CORIOLIS FORCE
%-------------------------------------------------------------------------------------------------------------------------------------------------------
for i=1:168
AF(i)=sqrt(AG(i).^2+AC(i).^2+AK(i).^2);
end
a1x=AF(:,1);%X-ACCN
a1y=AF(:,2);%Y-ACCN
a1z=AF(:,3);%Z-ACCN
 A1X=a1x';
 A1Y=a1y';
 A1Z=a1z';
 %-------------------------------------------------------------------------------------------------------------------------------------------------------
%velocity components
%for i=1:168
 %   Vx=diff(A1X);
  %  Vy=diff(A1Y);
   % Vz=diff(A1Z);
 %THE SECOND POLYNOMIAL COEFFICIENTS
for i=1:168
A2(i)=0.5*AF(i);
end
a2x=A2(:,1);
a2y=A2(:,2);
a2z=A2(:,3);
%-------------------------------------------------------------------------------------------------------------------------------------------------------
%THE THIRD POLYNOMIAL COEFFICIENTS
%for i=1:168
A3=(1/16).*diff(AF);
%end
a3x=A3(:,1);
a3y=A3(:,2);
```



```
a3z=A3(:,3);
%-------------------------------------------------------------------------------------------------------------------
%THE FOURTH POLYNOMIAL COEFFICIENTS
%for i=1:7
A4=(1/24).*6.*diff(A3);
%end
a4x=A4(:,1);
a4y=A4(:,2);
a4z=A4(:,3);

%Aon1=aoN(:,1);
%Aon2=aoN(:,2);
%Aon3=aoN(:,3);
%---------------------------------
%ALGORITHM OF AO  LOCATION HEIGHT H,AS AO LOCATION CO-ORDINATES FUNCTION ao
for i=1:5
   rt(i)=sqrt((aoX(i)+bx(i)).^2+(aoY(i)+by(i)).^2);
end
for i=1:5
   rz=Req.*(1-(cz.*cf.*sf));
   H(i)=rt(i)-rz;
end
if(H<=Hatm)
   pr=1;
else
   pr=2;
end
XX=min(aoX):(mean(aoX)/11):max(aoX);
 YY=min(aoY):(mean(aoY)/11):max(aoY);
ZZ=min(aoZ):(mean(aoZ)/11):max(aoZ);
  VVX=min(a1X):(mean(a1X)/11):max(a1X);
 VVY=min(a1Y):(mean(a1Y)/11):max(a1Y);
 VVZ=min(a1Z):(mean(a1Z)/11):max(a1Z);
 %conversion from ecef to geographic
 lam=77;
 phi=12;
 beta=pi/180;
 hs=100024.207503066;

 HS=hs.*[cos(lam*beta)*cos(phi*beta);sin(lam*beta)*cos(phi*beta);sin(phi*beta)];
 TEG=[-cos(lam*beta)*sin(phi*beta)-sin(lam*beta)*sin(phi*beta)cos(phi*beta);sin(lam*beta)-cos(lam*beta)
0;cos(lam*beta)*cos(phi*beta) sin(lam*beta)*cos(phi*beta) sin(phi*beta)];
 a=6378.137;
 eps=0.08181919;
 q=tan(phi*beta);
 del=atan(q)./(1+(eps.^2));
 rs=a/sqrt(1+eps.^2).*sin(del*beta).*sin(del*beta);
 RS=rs*[cos(lam*beta)*cos(del*beta);sin(lam*beta)*cos(del*beta);sin(del*beta)];
 BG=(HS)+(RS);

 xc=[XX' YY' ZZ'];
 s=xc(1:23);
 ab=TEG(1,3):ones(4,23);
 for i=1:11
   xg(:,i)=ab*(s(:,i)-BG);
 %xg=ab.*(s-BG);
   XG=xg';
    %XXG=(XG:ones(6,11));
  VZ=[VVX' VVY' VVZ'];
ss=VZ(1:11);
```



```
%ab=TEG(1,3):ones(4,11);
    vg=ab*ss;
 end
    VG=vg';
 %conversion from geographic to antenna coordinates
   azp=42.12;
 tr=10;
 rla=15;
 ela=20;
 tp=13;
 c11=-sin(azp*beta)*cos((tr+rla)*beta)+cos(azp*beta)*sin(tp*beta)*sin((tr+rla)*beta);
 c12=-sin(azp*beta)*sin((tr+rla)*beta)*sin(ela*beta)+cos(azp*beta)*(cos(tp*beta))*cos(ela*beta)-
sin(tp*beta)*cos((tr+rla)*beta)*sin(rla*beta);
 c13=-sin(azp*beta)*sin((tr+rla)*beta)*cos(ela*beta)-
cos(azp*beta)*(cos(tp+beta))*sin(ela*beta)+sin(tp*beta)*cos((tr+rla)*beta)*cos(ela*beta);
 c21=-sin(azp*beta)*sin(tp*beta)*sin((tr+rla)*beta)-cos(azp*beta)*cos((tr+rla)*beta);
 c22=-sin(azp*beta)*(cos(tp*beta)*cos(ela*beta)-sin(tp*beta)*cos((tr+rla)*beta)*sin(ela*beta))-
cos(azp*beta)*sin((tr+rla)*beta)*sin(ela*beta);
 c23=sin(azp*beta)*(cos(tp*beta)*sin(ela*beta)+sin(tp*beta)*cos((tr+rla)*beta)*cos(ela*beta))-
cos(azp*beta)*sin((tr+rla)*beta)*cos(ela*beta);
 c31=-cos(tp*beta)*sin((tr+rla)*beta);
 c32=sin(tp*beta)*cos(ela*beta)+cos(tp*beta)*cos((tr+rla)*beta)*sin(ela*beta);
 c33=-sin(tp*beta)*sin(ela*beta)+cos(tp*beta)*cos((tr+rla)*beta)*cos(ela*beta);
 TGA=[c11  c21  c31;c12 c22 c32;c13 c23 c33];
 bb=TGA(1,3):ones(4,11);
    xa=TGA*xg;
     XA=xa';
%bbb=[TGA(1,1):ones(2,11)];
      va=(bb(1,2):ones(3,11)).*vg;
  VA=va';
 %conversion from rectangular co-ordinates to sphrical co-ordinates
 Xxx=XA(:,1);
 Yyy=XA(:,2);
 Zzz=XA(:,3);
 for i=1:11
    rX(i)=sqrt(Xxx(i)^2+Yyy(i)^2+Zzz(i)^2);
 end
 RX=rX';
 for i=1:11
    alpha(i)=(sqrt(Xxx(i)^2+Yyy(i)^2))/Zzz(i);
    theta(i)=inv(tan(alpha(i)));
 end
Theta=theta';
 for i=1:11
    PHIP(i)=inv(tan((Yyy(i)/Xxx(i))));
 end
 phiP=PHIP';
 disp(RX)
disp(Theta)
disp(phiP)
```


FIGURE 1 SHIOWS THE TRAJECTORY DIAGRAM