ENG 4350 6.0 FW16/17- Space Hardware

Lab P5 – Integrating high level subroutine/functions into the main program and debugging the software

Submitted by:

Keith Menezes & James Brook

Submitted to:

Prof. Hugh Chesser

Contents

Purpose

The purpose of this report is to demonstrate the integration of all high-level functions into the main to perform an end-to-end test.

Fixes – Software/Code

The following functions have been altered for enhanced performance and compatibility with the rest of our main code.

- THETAN()

Customized the THETAN function such that it takes two parameters, the TLE epoch and the starting date of our tracking schedule. Inside the function we utilize an internal function in the DateAndTimeCalculations.c to calculated the Julian date at epoch.

Subsequently the header file was changed to include the extra input.

```
29      - double THETAN(double TLEepoch){
30      -     double num = TLEepoch;
31      -     num = TLEepoch/1000;
32      -     double year = num - frac(num);
33      -     double day = TLEepoch - year*1000;
34      -     double yearf = year + 2000;
35      -     double JDy = jdaty(yearf);
36      -     double JD = JDy + day -1;
37      -     double rads = THETAJ(JD);
        29  + double THETAN(double TLEepoch, double JDstart){
        30  +     double JD;
        31  +     JD = jdatep(TLEepoch);
        32  +     double rads = THETAJ(JD, JDstart);
38      33      return rads;
```

- THETAJ()

```
41      - double THETAJ (double JulianDate){
        36  + double THETAJ (double JulianDate, double JulianDateStart){
42      37      double JDm;
43      -     if(JulianDate>=floor(JulianDate) + 0.5){JDm = floor(JulianDate) + 0.5;}
44      -     else{JDm = floor(JulianDate) - 0.5;}
45      -     double Du = JulianDate - 2451545.0;
        38  +     if(JulianDateStart>=floor(JulianDateStart) + 0.5){JDm = floor(JulianDateStart) + 0.5;}
        39  +     else{JDm = floor(JulianDateStart) - 0.5;}
        40  +     double Du = JDm - 2451545.0;
46      41      double Tu = Du / 36525.0;
47      42      double GMST = 24110.54841 + 8640184.812866*Tu + 0.093104*Tu*Tu - 0.0000062*Tu*Tu*Tu;
48      43      for (;GMST > 86400;){
```

The THETAJ function had to be changed to accommodate the tracking schedule start date as an input. In the if-statement the tracking date is used as the comparative, the rest of the code remains the same, but now the function returns the GMST angle in radians for the specific Julian date.

- Added link signal strength to the main()

The signal strength has been added to the AOS/LOS table where an array is created to hold the signal strength of each satellite in the table. The formatting of the table was altered so that it displays as specified in the Tracking Specifications and on the user console.

```
350     -
        350 +     double ss[31];
351     351       for(int j=1; j<32; j++){//Run through each satellite
352     352
353     353           double currentTime;
...     ...     @@ -377,7 +377,7 @@ int main(void){
395     -             double ss[31];
396     -             //ss[num] = linkstrength(rtPos->mag);
        395 +
        396 +             ss[num] = linkstrength(rtPos->mag);
397     397               if (LA->elevation <= stn->az_el_lim.elmax && LA->elevation >= stn->az_el_lim.elmin && acquired == 0){//Go in to this loop if the
                          satellite is acquired.
398     398                   //If the satellite has already been acquired then don't add it again
399     399                   NUM[num] = j;
...     ...     @@ -418,10 +418,11 @@ int main(void){
418     418       fprintf(filepoint, "Sat No.           Name                AOS                 LOS                 Min. ExpectedLevel (dBm)\n");
419     419       for(int i=0;i<num;i++){
420     420           printf("%d    %s %s   ", NUM[i], NAME[i], jd2dat(AOS[i]));
421     -           printf("%s\n", jd2dat(LOS[i]));
        421 +           printf("%s", jd2dat(LOS[i]));
        422 +           printf("\t%f\n",ss[i]);
422     423           fprintf(filepoint, "%d        %s   %s   ", NUM[i], NAME[i], jd2dat(AOS[i]));
423     424           fprintf(filepoint, "%s", jd2dat(LOS[i]));
424     -       //  fprintf(filepoint,"%f\n",linkstrength(rtPos->mag));
        425 +       fprintf(filepoint,"   %f\n",ss[i]);
425     426       }
426     427       fclose(filepoint);
427     428       return 0;
```

Software Advancements

This section shows the advancements we made with the fixes in our code and confirmed with STK to approve.

AOS/LOS Data

Here is what the AOS/LOS table looks like with the expected signal strength:

Sat No.	Name	AOS	LOS	Min. ExpectedLevel (dBm)
0	GPS BIIR-2 (PRN 13)	2017-03-19 00:00:00	2017-03-19 00:59:59	-194.751455
2	GPS BIIR-4 (PRN 20)	2017-03-19 00:00:00	2017-03-19 00:59:59	-191.842836
3	GPS BIIR-5 (PRN 28)	2017-03-19 00:00:00	2017-03-19 00:59:59	-209.598671
4	GPS BIIR-6 (PRN 14)	2017-03-19 00:00:00	2017-03-19 00:59:59	-194.866402
7	GPS BIIR-9 (PRN 21)	2017-03-19 00:49:59	2017-03-19 00:59:59	-191.745841
11	GPS BIIR-13 (PRN 02)	2017-03-19 00:00:00	2017-03-19 00:59:59	-191.760244
12	GPS BIIRM-1 (PRN 17)	2017-03-19 00:00:00	2017-03-19 00:59:59	-204.384352
16	GPS BIIRM-5 (PRN 29)	2017-03-19 00:00:00	2017-03-19 00:59:59	-203.887205
19	GPS BIIF-1 (PRN 25)	2017-03-19 00:00:00	2017-03-19 00:59:59	-208.525202
20	GPS BIIF-2 (PRN 01)	2017-03-19 00:00:00	2017-03-19 00:59:59	-191.225996
22	GPS BIIF-4 (PRN 27)	2017-03-19 00:00:00	2017-03-19 00:59:59	-205.369155
23	GPS BIIF-5 (PRN 30)	2017-03-19 00:00:00	2017-03-19 00:59:59	-195.516778
24	GPS BIIF-6 (PRN 06)	2017-03-19 00:54:59	2017-03-19 00:59:59	-190.791992
25	GPS BIIF-7 (PRN 09)	2017-03-19 00:00:00	2017-03-19 00:59:59	-195.258373
27	GPS BIIF-9 (PRN 26)	2017-03-19 00:00:00	2017-03-19 00:59:59	-208.562454
29	GPS BIIF-11 (PRN 10)	2017-03-19 00:00:00	2017-03-19 00:59:59	-191.842144

To confirm the output is correct here is the Access from STK is featured below in the form of a picture.

The white lines show the access and the colourful lines show the propagated path of the satellite.

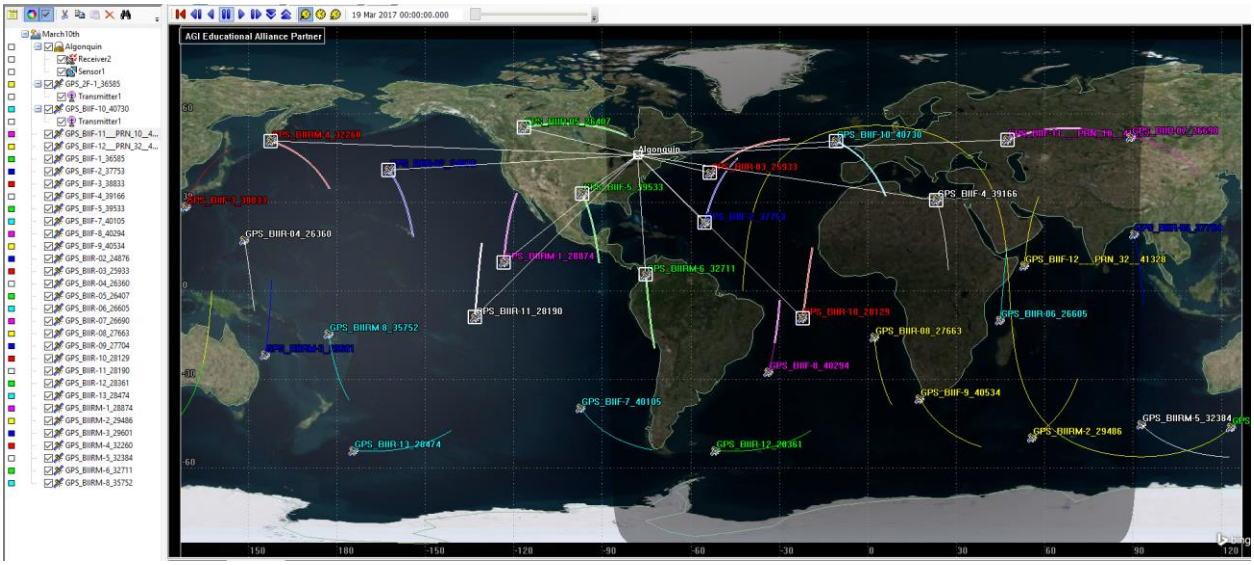

Figure 1: 2D view of the GPS satellites imported from the TLE projected onto the Earth

Here are the discrepancies\uncertainties:

- BIIF-4 can only be acquired until 19 Mar 2017 00:01:50.000 then until 1am
- BIIF-8 is acquired at 19 Mar 2017 00:26:00.000 then until 1am
- BIIF-9 maybe misidentified or misnamed by STK
    - BIIF-9 and BIIR-9 are out of view for STK
- BIIF-11 (PRN 10) can only be acquired until 19 Mar 2017 00:43:20.000
- BIIR-11 didn't get put into the table
- BIIRM-4 didn't get put into the table but may have been misidentified for BIIR-4
    - In this case, we are right
- BIIRM-5 is out of view for STK, but if misidentified for BIIR-5 or BIIF-5
    - In this case, we are right
- BIIRM-6 didn't get inserted into STK may have been misidentified for BIIR-6
    - In this case, we are right
- BIIF-7 is out of view for STK along with BIIR-7 and there was no BIIRM-7 imported into STK

Some of the discrepancies can be attributed to the naming convention that STK takes, it may not be accounting for the pointing limitations of the ARO (although they were set in STK), or that the TLE downloaded from CELESTRAK doesn't have all the satellites (which it didn't) in the GPS constellation.

Tracking Data

For the final roll, out of our software the Tracking Data table, as specified in 5.2 of Tracking Specifications lab manual. After the AOS/LOS table is calculated we:

1. Prompt the user to select a satellite to track from the AOS/LOS table
2. Use the satellite TLE parameters to calculate
    a. sat_ECI() to find the satellite ECI position and velocity
    b. sat_ECF() to find the satellite ECF position and velocity
    c. station_ECF() to find the station position
    d. range_ECF2topo() to find the position and velocity
    e. Range_topo2look_angles() to find the position and velocity angles and range from the topocentric coordinate system.

Here is what it looks like on the console. Simultaneously it is printed to a file named TrackingData.txt This information is what will be used to steer the ARO dish to the satellite and confirm its signal.

```
Enter the satellite number you would like to track from the table (values in the first column): 0

-------------------------------------------------------------------------------------------------------------------------------------------------------
UTC                    AZ          EL          AZ-vel      EL-vel      Range       Range-Rate  Doppler     Level
                       deg         deg         deg/sec     deg/sec     km          km/sec      kHz         dbm
-------------------------------------------------------------------------------------------------------------------------------------------------------
2017-03-19 00:00:00    64.726728   -22.897671  -0.000077   0.000122    25275.710905  3.159979                -195.507249
2017-03-19 00:04:59    63.049901   -20.904136  -0.000074   0.000123    25254.013698  3.168479                -195.490073
2017-03-19 00:09:59    61.425865   -18.877012  -0.000071   0.000123    25231.991827  3.175170                -195.472625
2017-03-19 00:14:59    59.845971   -16.822333  -0.000069   0.000123    25209.710261  3.179985                -195.454956
2017-03-19 00:19:59    58.301721   -14.745881  -0.000067   0.000123    25187.235078  3.182878                -195.437117
2017-03-19 00:24:59    56.784793   -12.653233  -0.000066   0.000123    25164.633298  3.183824                -195.419162
2017-03-19 00:29:59    55.287049   -10.549811  -0.000065   0.000122    25141.972669  3.182816                -195.401144
2017-03-19 00:34:59    53.800529   -8.440934   -0.000065   0.000120    25119.321487  3.179866                -195.383117
2017-03-19 00:39:59    52.317447   -6.331862   -0.000065   0.000118    25096.748393  3.175008                -195.365137
2017-03-19 00:44:59    50.830177   -4.227848   -0.000065   0.000115    25074.322160  3.168297                -195.347257
2017-03-19 00:49:59    49.331244   -2.134176   -0.000066   0.000111    25052.111478  3.159804                -195.329533
2017-03-19 00:54:59    47.813313   -0.056207   -0.000067   0.000107    25030.184737  3.149623                -195.312021
2017-03-19 00:59:59    46.269192   2.000580    -0.000069   0.000101    25008.609817  3.137867                -195.294774
```

Here is the how the code is printed to the screen and file. Further investigations of the code can be found on our [github page](#).

```
@@ -439,11 +439,17 @@ int main(void){
     fflush(stdout);
     scanf("%d", &satNum);
     Satellite sat = sats[satNum+1];
+    FILE *xp;
+    xp = fopen("TrackingData.txt", "w+");
     printf("\n-------------------------------------------------------------------------------------------------------------------------------------------------------
     \n");
     printf("UTC\t\t\tAZ\t\tEL\t\tAZ-vel\t\tEL-vel\t\tRange\t\tRange-Rate\t\tDoppler\t\tLevel\n");
     printf("\t\t\tdeg\t\tdeg\t\tdeg/sec\t\tdeg/sec\t\tkm\t\tkm/sec\t\t\tkHz\t\tdbm\n");
     printf("-------------------------------------------------------------------------------------------------------------------------------------------------------\n");

+    fprintf
+    (xp,"\n-------------------------------------------------------------------------------------------------------------------------------------------------------\n");
+    fprintf(xp,"UTC\t\t\tAZ\t\tEL\t\tAZ-vel\t\tEL-vel\t\tRange\t\tRange-Rate\t\tDoppler\t\tLevel\n");
+    fprintf(xp,"\t\t\tdeg\t\tdeg\t\tdeg/sec\t\tdeg/sec\t\tkm\t\tkm/sec\t\t\tkHz\t\tdbm\n");
+    fprintf(xp,"-------------------------------------------------------------------------------------------------------------------------------------------------------
     \n");

     double currentTime;
     currentTime = JulianDateStart;
@@ -491,9 +497,8 @@ int main(void){
         ss[num] = linkstrength(rtPos->mag);

         printf("%s\t%f\t%f\t%f\t%f\t%f\t\t\t\t\t%f\n", jd2dat(currentTime),LA->azimuth,LA->elevation,LA->azimuth_velocity,LA->elevation_velocity,sqrt(rtPos->x*rtPos->x+rtPos-
         >y*rtPos->y+rtPos->z*rtPos->z),sqrt(rtVel->x*rtVel->x+rtVel->y*rtVel->y+rtVel->z*rtVel->z),linkstrength(rtPos->mag));
-
-
+        fprintf(xp,"%s\t%f\t%f\t%f\t%f\t%f\t\t\t\t\t%f\n", jd2dat(currentTime),LA->azimuth,LA->elevation,LA->azimuth_velocity,LA->elevation_velocity,sqrt(rtPos->x*rtPos->x
         +rtPos->y*rtPos->y+rtPos->z*rtPos->z),sqrt(rtVel->x*rtVel->x+rtVel->y*rtVel->y+rtVel->z*rtVel->z),linkstrength(rtPos->mag));
     }
-
+    fclose(fp);
     return 0;
 }
```

Conclusion

Our software has demonstrated the integration of all the functions required to track the GNSS satellites. Next, we just need to add the Doppler equation, which was not a specified function for P1 or P2, but some information has been provided in the Tracking Specifications. Continual testing will be performed between now and the ARO trip, to improve the precision of the tracking, robustness of the code, and ease with the user input. This software project has increased our programming ability and proficiency with C. Our team has worked very hard and diligently every week to roll out the best software package.

Appendix – Full main (also on github)

```
#include <stdio.h>
#include <math.h>
#include "Propagate.h"
#include "Basic.h"
#include "FileIO.h"
#include "Vector.h"
#include "STKout.h"
#include "DateAndTimeCalculations.h"
#include "Matrix.h"
#include "Vector.h"
#include <time.h>
#include <stdlib.h>
#include <string.h>
#define CUBE_ROOT(X) (exp(log(X)/ 3.))
#define PI
3.14159265358979323846264338327950288419716939937510582097494459230781640628 6

int main(void){

    Banner();
    printf("\nImporting station data...\n\n");
    Station *stn = (Station*) malloc(sizeof(Station));
    ReadStationFile(stn, '0');
    printf("Complete\n\n");
    int sf;
    for (sf = 0; sf < 1;){
       printf("\nEnter the number next to the corresponding option:\n");
       printf("1   view station file data\n");
       printf("2   edit station file data\n");
       printf("3   continue\n\n");
       int input1;
       printf("Entry: ");
       fflush(stdout);
       scanf("%d", &input1);
       if (input1 == 1){
           printf("\nStation File Contents:\n");
           printf("1  Name: %s\n", stn->name);
           printf("2  Station Latitude: %f\n", stn->stnlat);
           printf("3  Station Longitude: %f\n", stn->stnlong);
           printf("4  Station Altitude: %f\n", stn->stnalt);
           printf("5  UTC Offset: %f\n", stn->utc_offset);
```

```
            printf("6  Azimuth Elevation nlim: %d\n", stn->az_el_nlim);
            printf("7  Azimuth Elevation Limit Azimuth: %f\n", stn->az_el_lim.az);
            printf("8  Azimuth Elevation Limit Elevation Min: %f\n", stn-
>az_el_lim.elmin);
            printf("9  Azimuth Elevation Limit Elevation Max: %f\n", stn-
>az_el_lim.elmax);
            printf("10 Station Azimuth Speed Max: %f\n", stn->st_az_speed_max);
            printf("11 Station Elevation Speed Max: %f\n", stn->st_el_speed_max);
        }
        if(input1 == 2){

            printf("\n1    name\n");
            printf("2    stnlat\n");
            printf("3    stnlong\n");
            printf("4    stnalt\n");
            printf("5    utc_offset\n");
            printf("6    az_el_nlim\n");
            printf("7    az_el_nlim.az\n");
            printf("8    az_el_nlim.elmin\n");
            printf("9    az_el_nlim.elmax\n");
            printf("10   st_az_speed_max\n");
            printf("11   st_el_speed_max\n");

            printf("\nWhat parameter would you like to edit: ");
            int num2;
            fflush(stdout);
            scanf("%d", &num2);
            printf("\n");
            if(num2 == 1){
               printf("name entry: ");
               char *n;
               fflush(stdout);
               scanf("%c", n);
               //stn->name = n;
            }
            if(num2 == 2){
               printf("stnlat entry: ");
               double n;
               fflush(stdout);
               scanf("%lf", &n);
               stn->stnlat = n;
            }
            if(num2 == 3){
               printf("stnlon entry: ");
               double n;
               fflush(stdout);
               scanf("%lf", &n);
               stn->stnlong = n;
            }
            if(num2 == 4){
               printf("stnalt: ");
               double n;
               fflush(stdout);
               scanf("%lf", &n);
               stn->stnalt = n;
```

```
}
if(num2 == 5){
   printf("utc_offset entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   stn->utc_offset = n;
}
if(num2 == 6){
   printf("az_el_nlim entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   stn->az_el_nlim = n;
}

if(num2 == 7){
   printf("az_el_lim.az entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   stn->az_el_lim.az = n;
}
if(num2 == 8){
   printf("az_el_lim.elmin entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   stn->az_el_lim.elmin = n;
}

if(num2 == 9){
   printf("az_el_lim.elmax entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   stn->az_el_lim.elmax= n;
}

if(num2 == 10){
   printf("st_az_speed_max entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   stn->st_az_speed_max = n;
}

if(num2 == 11){
   printf("st_el_speed_max entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   stn->st_el_speed_max = n;
}
```

```
    }
    if (input1 == 3){sf++;}

}

printf("\nImporting TLE file sats...\n\n");
char *file = "TLE.txt";
Satellite sats[32];
ReadNoradTLE(sats, file);
printf("Complete\n");
int x;
for(x = 0;x < 1;){
   printf("\n\nEnter the number next to the corresponding option:\n");
   printf("1   view TLE data\n");
   printf("2   edit TLE data\n");
   printf("3   continue\n\n");
   int input2;
   printf("Entry: ");
   fflush(stdout);
   scanf("%d", &input2);

   if(input2 == 1){
       printf("\nEnter the satellite number you would like to view: ");
       int num;
       fflush(stdout);
       scanf("%d", &num);

       printf("\nInformation for sat number %d\n", num);
       printf("\n    name is %s", sats[num].name);
       printf("    refepoch is %f\n", sats[num].refepoch);
       printf("    incl is %f\n", sats[num].incl);
       printf("    raan is %f\n", sats[num].raan);
       printf("    eccn is %f\n", sats[num].eccn);
       printf("    argper is %f\n", sats[num].argper);
       printf("    meanan is %f\n", sats[num].meanan);
       printf("    meanmo is %f\n", sats[num].meanmo);
       printf("    ndot is %f\n", sats[num].ndot);
       printf("    nddot6 is %f\n", sats[num].nddot6);
       printf("    bstar is %f\n", sats[num].bstar);
       printf("    orbitnum is %f\n", sats[num].orbitnum);
   }
   if(input2 == 2){
       printf("\nEnter the satellite number you would like to edit: ");
       int num;
       fflush(stdout);
       scanf("%d", &num);

       printf("\n1    name\n");
       printf("2    refepoch\n");
       printf("3    incl\n");
       printf("4    raan\n");
       printf("5    eccn\n");
       printf("6    argper\n");
       printf("7    meanan\n");
```

```
printf("8    meanmo\n");
printf("9    ndot\n");
printf("10   nddot6\n");
printf("11   bstar\n");
printf("12   orbitnum\n");

printf("\nWhat parameter would you like to edit: ");
int num2;
fflush(stdout);
scanf("%d", &num2);
printf("\n");
if(num2 == 1){
   printf("name entry: ");
   char *n;
   fflush(stdout);
   scanf("%c", n);
   //sats[num].name = n;
}
if(num2 == 2){
   printf("refepoch entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   sats[num].refepoch = n;
}
if(num2 == 3){
   printf("incl entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   sats[num].incl = n;
}
if(num2 == 4){
   printf("raan entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   sats[num].raan = n;
}
if(num2 == 5){
   printf("eccn entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   sats[num].eccn = n;
}
if(num2 == 6){
   printf("argper entry: ");
   double n;
   fflush(stdout);
   scanf("%lf", &n);
   sats[num].argper = n;
}

if(num2 == 7){
```

```
            printf("meanan entry: ");
            double n;
            fflush(stdout);
            scanf("%lf", &n);
            sats[num].meanan = n;
         }

         if(num2 == 8){
            printf("meanmo entry: ");
            double n;
            fflush(stdout);
            scanf("%lf", &n);
            sats[num].meanmo = n;
         }
         if(num2 == 9){
            printf("ndot entry: ");
            double n;
            fflush(stdout);
            scanf("%lf", &n);
            sats[num].ndot = n;
         }

         if(num2 == 10){
            printf("nddot6 entry: ");
            double n;
            fflush(stdout);
            scanf("%lf", &n);
            sats[num].nddot6 = n;
         }

         if(num2 == 11){
            printf("bstar entry: ");
            double n;
            fflush(stdout);
            scanf("%lf", &n);
            sats[num].bstar = n;
         }

         if(num2 == 12){
            printf("orbitnum entry: ");
            double n;
            fflush(stdout);
            scanf("%lf", &n);
            sats[num].orbitnum = n;
         }

      }
      if(input2 == 3){x++;}
   }

   //------------------------------AOS/LOS-----------------------------------------
---


   printf("\nOpening tracking file...\n\n");
```

```
    FILE *fp = fopen("tracking_sched.txt","r+");

    char line1[50];
    char line2[50];
    char line3[31];
    fgets(line1, 50, fp);
    fgets(line2, 50, fp);
    fgets(line3, 31, fp);

    fclose(fp);


    char date_start[20], date_stop[20], time_step[6];

    strncpy(date_start, line1+26, 19);
    strncpy(date_stop, line2+25, 19);
    strncpy(time_step, line3+24, 5);
    printf("Printing out the start date %s\n",date_start);
    printf("Printing out the stop date %s\n",date_stop);
    printf("Printing out the time step %s\n",time_step);

    printf("\nCalculating AOS and LOS...\n");

    double step;
    step = atof(time_step);

    double JulianDateStart, JulianDateStop;

    JulianDateStart = dat2jd(date_start);
    JulianDateStop = dat2jd(date_stop);


    char *NAME[31];
    double AOS[31], LOS[31];
    int NUM[31];
    int num = 0;

    double ss[31];
    for(int j=0; j<31; j++){

       double currentTime;
       currentTime = JulianDateStart;
       int acquired=0;
       int lost = 0;

       for( ;currentTime<JulianDateStop; currentTime = currentTime+frcofd(0,0,step)){
           double mA, mM;
           double satEpoch = sats[j].refepoch;
           double mA0 = sats[j].meanan;
           double nMM = sats[j].meanmo;
           double ndMM = sats[j].ndot;
           double n2dMM = sats[j].nddot6;
           mean_anomaly_motion(&mA, &mM, currentTime, satEpoch, mA0, nMM, ndMM,
n2dMM);
           double mMrev=mM/(2*PI);
```

```
            double eccAnom = KeplerEqn(mA, sats[j].eccn);

            Vector *eciPos, *eciVel;
            eciPos = (Vector*)malloc(sizeof(Vector));
            eciVel = (Vector*)malloc(sizeof(Vector));

            double sMA = CUBE_ROOT(398600.4418/(4*PI*PI*mMrev*mMrev));
            sat_ECI(eciPos, eciVel, sats[j].eccn, eccAnom, sMA,
sats[j].raan*PI/180,sats[j].argper*PI/180, sats[j].incl*PI/180, mM);
            Vector *ecfPos, *ecfVel;
            ecfPos = (Vector*)malloc(sizeof(Vector));
            ecfVel = (Vector*)malloc(sizeof(Vector));

            double thetat = THETAJ(currentTime,JulianDateStart);
            sat_ECF(ecfPos, ecfVel, thetat, eciPos, eciVel);
            Vector *stnPos, *rtPos, *rtVel;
            stnPos = (Vector*)malloc(sizeof(Vector));
            station_ECF(stnPos, stn->stnlong, stn->stnlat, stn->stnalt);
            rtPos = (Vector*)malloc(sizeof(Vector));
            rtVel = (Vector*)malloc(sizeof(Vector));
            range_ECF2topo(rtPos, rtVel, *stnPos, ecfPos, ecfVel, stn->stnlong, stn-
>stnlat);

            double az;
            double el;
            double azV;
            double elV;
            LookAngles *LA =(LookAngles*) malloc(sizeof(LookAngles));
            range_topo2look_angles(LA, az, el, azV, elV, rtPos, rtVel);

            ss[num] = linkstrength(rtPos->mag);
            if (LA->elevation <= stn->az_el_lim.elmax && LA->elevation >= stn-
>az_el_lim.elmin && acquired == 0){//Go in to this loop if the satellite is acquired.

               NUM[num] = j;
               NAME[num] = sats[j].name;
               NAME[num][strlen(NAME[num])-1] = '\0';
               NAME[num][strlen(NAME[num])-2] = '\0';
               AOS[num] = currentTime;
               acquired = 1;
            }
            if(LA->elevation >= stn->az_el_lim.elmax && LA->elevation <= stn-
>az_el_lim.elmin && acquired==1){// Go in to this loop if the satellite is lost.
               LOS[num] = currentTime;
               lost = 1;
               break;
            }
      }
      if(acquired==1)
            {
            if(lost==0)
               {
               LOS[num] = JulianDateStop;
               }num++;
```

```
            }
        }
        printf("\nComplete\n\n");

        FILE *filepoint;
        filepoint = fopen("AOSLOS.txt", "w+");
        printf("Sat No.       Name                AOS                 LOS          Min.
ExpectedLevel (dBm)\n");
        fprintf(filepoint, "Sat No.               Name                                AOS
                            LOS                                 Min. ExpectedLevel
(dBm)\n");
        for(int i=0;i<num;i++){
           printf("%d    %s %s       ", NUM[i], NAME[i], jd2dat(AOS[i]));
           printf("%s", jd2dat(LOS[i]));
           printf("\t%f\n",ss[i]);
           fprintf(filepoint, "%d          %s     %s     ", NUM[i], NAME[i],
jd2dat(AOS[i]));
           fprintf(filepoint, "%s", jd2dat(LOS[i]));
           fprintf(filepoint,"       %f\n",ss[i]);
        }
        fclose(filepoint);


        printf("\n\nEnter the satellite number you would like to track from the table
(values in the first column): ");
        int satNum;
        fflush(stdout);
        scanf("%d", &satNum);
        Satellite sat = sats[satNum+1];
        FILE *xp;
        xp = fopen("TrackingData.txt", "w+");
        printf("\n-------------------------------------------------------------------------
-------------------------------------------------------------------------------------
-----\n");
        printf("UTC\t\t\tAZ\t\tEL\t\tAZ-vel\t\tEL-vel\t\tRange\t\tRange-
Rate\t\tDoppler\t\tLevel\n");

printf("\t\t\tdeg\t\tdeg\t\tdeg/sec\t\tdeg/sec\t\tkm\t\tkm/sec\t\t\tkHz\t\tdbm\n");
        printf("-------------------------------------------------------------------------
-------------------------------------------------------------------------------------
---\n");

        fprintf(xp,"\n-------------------------------------------------------------------------
-------------------------------------------------------------------------------------
---------\n");
        fprintf(xp,"UTC\t\t\tAZ\t\tEL\t\tAZ-vel\t\tEL-vel\t\tRange\t\tRange-
Rate\t\tDoppler\t\tLevel\n");

fprintf(xp,"\t\t\tdeg\t\tdeg\t\tdeg/sec\t\tdeg/sec\t\tkm\t\tkm/sec\t\t\tkHz\t\tdbm\n"
);
        fprintf(xp,"-------------------------------------------------------------------------
-------------------------------------------------------------------------------------
-------\n");

        double currentTime;
```

```
        currentTime = JulianDateStart;
        int acquired=0;
        int lost = 0;

    for( ;currentTime<JulianDateStop; currentTime = currentTime+frcofd(0,0,step)){
            double mA, mM;
            double satEpoch = sat.refepoch;
            double mA0 = sat.meanan;
            double nMM = sat.meanmo;
            double ndMM = sat.ndot;
            double n2dMM = sat.nddot6;
            mean_anomaly_motion(&mA, &mM, currentTime, satEpoch, mA0, nMM, ndMM,
n2dMM);
            double mMrev=mM/(2*PI);

            double eccAnom = KeplerEqn(mA, sat.eccn);

            Vector *eciPos, *eciVel;
            eciPos = (Vector*)malloc(sizeof(Vector));
            eciVel = (Vector*)malloc(sizeof(Vector));

            double sMA = CUBE_ROOT(398600.4418/(4*PI*PI*mMrev*mMrev));
            sat_ECI(eciPos, eciVel, sat.eccn, eccAnom, sMA,
sat.raan*PI/180,sat.argper*PI/180, sat.incl*PI/180, mM);
            Vector *ecfPos, *ecfVel;
            ecfPos = (Vector*)malloc(sizeof(Vector));
            ecfVel = (Vector*)malloc(sizeof(Vector));

            double thetat = THETAJ(currentTime,JulianDateStart);
            sat_ECF(ecfPos, ecfVel, thetat, eciPos, eciVel);
            Vector *stnPos, *rtPos, *rtVel;
            stnPos = (Vector*)malloc(sizeof(Vector));
            station_ECF(stnPos, stn->stnlong, stn->stnlat, stn->stnalt);
            rtPos = (Vector*)malloc(sizeof(Vector));
            rtVel = (Vector*)malloc(sizeof(Vector));
            range_ECF2topo(rtPos, rtVel, *stnPos, ecfPos, ecfVel, stn->stnlong, stn-
>stnlat);

            double az;
            double el;
            double azV;
            double elV;
            LookAngles *LA =(LookAngles*) malloc(sizeof(LookAngles));
            range_topo2look_angles(LA, az, el, azV, elV, rtPos, rtVel);

            ss[num] = linkstrength(rtPos->mag);

            printf("%s\t%f\t%f\t%f\t%f\t%f\t%f\t\t\t\t%f\n", jd2dat(currentTime),LA-
>azimuth,LA->elevation,LA->azimuth_velocity,LA->elevation_velocity,sqrt(rtPos-
>x*rtPos->x+rtPos->y*rtPos->y+rtPos->z*rtPos->z),sqrt(rtVel->x*rtVel->x+rtVel-
>y*rtVel->y+rtVel->z*rtVel->z),linkstrength(rtPos->mag));
            fprintf(xp,"%s\t%f\t%f\t%f\t%f\t%f\t%f\t\t\t\t%f\n",
jd2dat(currentTime),LA->azimuth,LA->elevation,LA->azimuth_velocity,LA-
>elevation_velocity,sqrt(rtPos->x*rtPos->x+rtPos->y*rtPos->y+rtPos->z*rtPos-
```

```
>z),sqrt(rtVel->x*rtVel->x+rtVel->y*rtVel->y+rtVel->z*rtVel->z),linkstrength(rtPos-
>mag));
        }
    fclose(fp);
    return 0;
}
```