Assignments for 30550 - Satellite Based Positioning Fall 2018

General Information

The assignments A-G corresponds basically to the processes that are carried out in a conventional code-based GPS receiver. In implementing the routines, simulated data is used, and finally real data is introduced with assignment G.
The assignments F and G focus on the data retrieval and analysis for precise point positioning.
Assignment H is not tied down to the processes in a GPS receiver, but deals with the integration of GPS and Galileo.

All assignments must be solved using Matlab, and procedures, results and source code must be documented and discussed in a short report.

The assignment reports will be given marks, and these marks contribute to the final mark in the following manner:

The assignments, A + B + C, are given one mark that contributes with 20 % to the final mark
The assignments, D + E, are given one mark that contributes with 20 % to the final mark
Assignment F is given one mark that contributes with 30 % to the final mark
The assignments, G + H, are given one mark that contribute with 30 % to the final mark

All assignments must be handed in by **Friday, December 7th , 2018**, at midnight.

Assignment A. Geographic and Cartesian Coordinates

Select a location in Denmark and determine Latitude, Longitude and Height. Use a topographic map or the internet to find realistic coordinates. If the height is determined as an MSL height (height above sea level), add 40 meter to obtain an approximate ellipsoidal height.

Implement routines in Matlab for conversion from Latitude, Longitude, and Height to X, Y, and Z (in WGS84), and also routines in the reverse direction (from X, Y, Z to Latitude, Longitude, Height)

Use the following parameters for the WGS84 ellipsoid:
$a = 6378137.0$ meter
$f = 1 / 298.257223563$

Note that the transformation routine should be able to convert positions with negative Latitude and Longitude, as well as positions on the Equator, and the North and South Poles. Consider other "dangerous" positions that might cause problems in the code.

The two transformation routines are acceptable, when the position, converted with your own code and with KMSTrans, differs less than a given tolerance defined by you.

Please keep in mind that trigonometric operations within Matlab are carried out using radians. Either convert the angular variables between degrees and radians and use the Matlab functions *sin* and *cos*, or use the functions *sind* and *cosd* that take decimal degrees as input.

Avoid using radians as input or output of your code. This will make it difficult to evaluate the results later in the course.

The KMSTrans program is available for download here:
ftp://ftp.sdfe.dk/download/transformationsprogram/

Or you can use the web service here: http://valdemar.kms.dk/trf/
Where you choose “GPS-transformation” and make transformations between the system labels crt_etrsw89 (cartesian coordinates) and geoEetrs89 (geodetic coordinates).

The report must contain:

Matlab code, results for conversion of the position both with your own code and with KMSTrans, list of coordinate differences, and an evaluation of how well your code performs. Describe also your considerations on the positions, where the equations are not applicable.

Assignment B. Satellite Coordinates from Kepler Elements

Based on the Kepler elements of the GPS satellites, implement a Matlab code to estimate satellite positions in the orbital coordinate system, convert the positions to the inertial system, and plot the result.

The result should be a 3D plot in the Cartesian inertial coordinate system showing the positions of the 24 satellites at the given instance in time (e.g. use the Matlab function *plot3* for this).

When the above part of the code is successfully implemented, a time counter is added, and the positions of the GPS satellites are simulated over 12 hours. Use the expression for the mean motion.

Finally generate another 3D plot showing the positions of the satellites every 15 or 30 minutes. The plot will basically show the satellite *orbits* in the inertial system.

The report must contain:

Matlab code, one plot showing the 24 satellite positions at one given instance in time, and one plot showing the positions over 12 hours. Also please briefly evaluate the plots – do they look realistic?

Assignment C. Global and Local Coordinate Systems

Implement routines for conversion of coordinates from X, Y, Z in WGS84 to N, E, U in a local Cartesian coordinate system. The origo of the local system must be the position used in Assignment A.

Define another position that can be used for testing of the code, e.g. a point located 100 meters above the point you used as origo.

Implement routines for computation of elevation and azimuth for the vector going from origo of the local coordinate system (your position from Assignment A) to the test point, and verify that the routines work well. Be aware that inverse tangent '*atan*' can provide other angles than you might expect or use the Matlab function '*atan2'*. Check that the program works according to your expectations.

Choose an epoch in time given by year, day, hour and minutes (by quarter i.e. 0, 15, 30 or 45 minutes).

Find the corresponding file with precise satellite positions at the web site of the IGS (International GPS Service): http://igscb.jpl.nasa.gov/components/prods_cb.html

Satellite positions are given for every 15 minutes in the sp3-files, and the coordinate unit is *km*. The sp3 format is described in the text file "Data File Format SP3" available on DTU Inside.

Since you only need to work with a smaller part of the 24-hour sp3 file, it might be an advantage to copy the relevant lines of the 24-hour sp3 file to a smaller file containing only the data you need. Also you are free to reformat the sp3-file to another format that might be easier for you to deal with.

Read satellite positions for the given time epoch and convert the satellite positions to N, E, U coordinates. Determine elevation and azimuth in the local coordinate system for the vectors to all GPS satellites at the given time epoch.

Now use the code to generate a list of coordinates of those satellites that are *visible* from the origo of the local system, and finally determine the distances from origo to the satellites.

If less than five satellites are visible you must pick another instant in time and repeat the exercise (we will be working with these results in the following assignments, and you must have at least five visible satellites for some of the following assignments).

The report must contain:

Matlab code, coordinate lists containing (X, Y, Z) in WGS84, (N, E, U) in the local system, and elevation, azimuth, and distance to the visible GPS satellites.

Evaluate whether the distances determined to the satellites are realistic and explain why (or why not) in the report.

Assignment D. Interpolation of Satellite Positions

Use the same sp3 file as in Assignment C.

The satellite positions in the sp3 file are provided every 15 minutes. To obtain satellite positions at any epoch in time, an interpolation routine is necessary.

Implement a Matlab routine for interpolation of satellite positions to obtain positions of the satellites at 12:05 hours UTC for the day you are working with.

Normally, Lagrange interpolation is used for interpolation in sp3 files, but here we will use cubic spline interpolation since this is supported by Matlab (e.g. with the function *interp1*).

When programming, it is always important to check the code. In this case it is difficult since we do not have given satellite positions as reference. One test approach can be to interpolate satellite positions for one satellite e.g. for every minute during an hour, and then plot the positions together with the positions given in the sp3 file. It is now possible to visually check whether the satellite orbit is continuous. Another approach for testing the interpolation is to remove one of the positions in the input file, estimate the position using the code, and compare with the known position. This, however, is not a good test either, since we will here
have 30 minutes between the known satellite positions. If you have another and better idea for testing the code, please feel free to try it out.

The report must contain:

Matlab code, plots for evaluation of the interpolation, a list of satellite positions at 12:05 for the given day, and an evaluation of how much the satellites move, in general, during 5 minutes (e.g. from 12:00 to 12:05).

Assignment E. Error Sources and Pseudoranges

1:

Implement the Saastamoinen troposphere model.

Use standard sea-level meteorological values for temperature, total pressure and relative humidity: T = 18° Celsius, P = 1013 hPa, RH = 50 %

In order to use Equation (5.39) in Misra and Enge (2011) p. 171, the relative humidity *RH* must be converted to the partial pressure of water vapor, e_0 (in hPa), using the following expression:

$$e_0 = 6.106 \cdot RH \cdot e^{\left(\frac{17.15 \cdot T - 4684}{T - 38.45}\right)}$$

where *T* is temperature, given in Kelvin.

For position information (station height and latitude) use the location you have worked with in the previous assignments (A and C). Retrieve elevation angles for the receiver-satellite vectors determined in Assignment C, and estimate the tropospheric delay for the visible satellites.

2:

The distances between receiver and satellites determined in Assignment C, now have to be modified to simulate real code pseudoranges from a GPS receiver. Therefore, you have to add atmospheric errors and clock errors.

Ionospheric delay
Find an appropriate vertical total electron content (VTEC) value using the "seSolstorm" service at http://sesolstorm.kartverket.no/moreplots.xhtml - click "Plasmainnhold i ionosfæren (VTEC)" in the right column of the site to get the VTEC map.

Use the date and time you selected in Assignment C, and estimate a value based on the map.
Convert the VTEC-value from TEC-units to electrons / m^2

Estimate the ionospheric delay for the L1 frequency (only the first order effect) using the standard expression given in the lecture slides. Remember to correct for the elevation angle. Use the expression from the lecture slides.

Tropospheric delay
Use the tropospheric estimate determined with the Saastamoinen model.

Satellite clock error
Take the value corresponding to the satellite positions as given in the sp3-file used in Assignment C. The clock error is listed in micro seconds.

Receiver clock error
Use a receiver clock error of 0.1 milliseconds for this exercise.

The four errors must be converted to metric units and added to the geometric distances to obtain pseudoranges – be careful with the signs (plus and minus) when doing this.

Save the pseudoranges for use in the following assignments.

The report must contain:

Matlab codes, a list of the errors added to each of the satellite signals, and a discussion of the size of the errors.

The discussion could include issues such as the size of the various errors, does the size of the errors seem reasonable, did anything surprise you etc. Use information from the literature provided and the lecture slides for this discussion.