GPS User position estimation using ephemeris data from satellites

Ishaan Sharma
Electrical and Computer Engineering Department
ID: 40053363
Concordia University
Avionics Navigation Systems ENGR 6461

***Abstract*—In this report ephemeris data received from satellites by global positioning system receivers is utilized to find out the user position. User's latitude and longitude is pinpointed on Google maps using GPS pseudo-range equation. Linearized GPS pseudo-range equation is solved by weighted least square. The radio signal propagation errors through ionosphere and troposphere are calculated and compensated in the calculation. Satellite clock bias and user clock bias are also included in the calculation model. Error analysis is carried out by constructing the error ellipses.**

1. Introduction

Global positioning system to find users position is an integral part of our most used devices like mobile phones, laptops, car navigation systems and numerous apps. These coordinates are generated by gps receivers which decode data received from satellites available at that particular instant. GPS positioning is based on trilateration, which is the method of determining position by measuring distances to points at known coordinates. At a minimum, trilateration requires 3 ranges to 3 known points. But in GPS 4 pseudo-ranges to satellites are utilized to find out four unknowns. GPS is owned by the US government and operated by the US air force. Other countries have also implemented their own GPS like systems. Examples are GLONASS by Russia, Galileo by the European Union and BeiDou by China.

The report comprises of the assumptions while calculating the gps position, flowchart of the algorithm for coding, explanation of individual components of the code followed by results and error analysis.

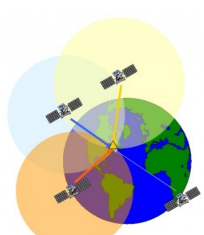

Figure 1. Trilateration using 4 satellites [1]

2. Assumptions

1) Earth's shape is modeled by an ellipsoid which means that geometrically, the equatorial radius is longer than the polar axis by about 23 km. The direction of gravity does not point to the center of the earth. This model is used to convert the geocentric coordinates of user[X,Y,Z] to the geodetic model and locate the latitude, longitude and height on the Google maps.

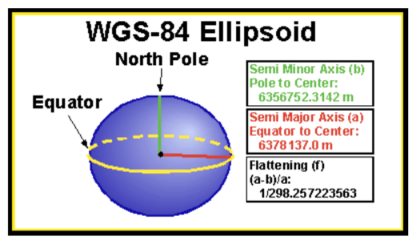

Figure 2. Earth ellipsoid model [2]

2) In order to get the user position from this algorithm at any instant 4 or more satellites must be visible to the user. The satellite constellation is designed in such a way.
3) Least square solution converges after only 4 or 5 iterations. Which means that the value of "dx" (Difference between the true and calculated position) and "db" (Difference in actual and calculated user clock bias) is reduced to e-3 in these iterations when the corrected pseudo range, user clock bias and errors are fed back in each iteration.
4) The values of constants for GPS equations like c (Speed of light), μ (Earth's gravitational constant), $\dot{\theta}$ (Earth's rotation rate), F (rel cor term constant). The initial value of τ = 0.075 sec. This value is updated in each iteration of algorithm.

3. Flowchart for the matlab code

Here the flowchart of the matlab main code is explained and in further sections each component of the code is explained specifying how each component is calculated.

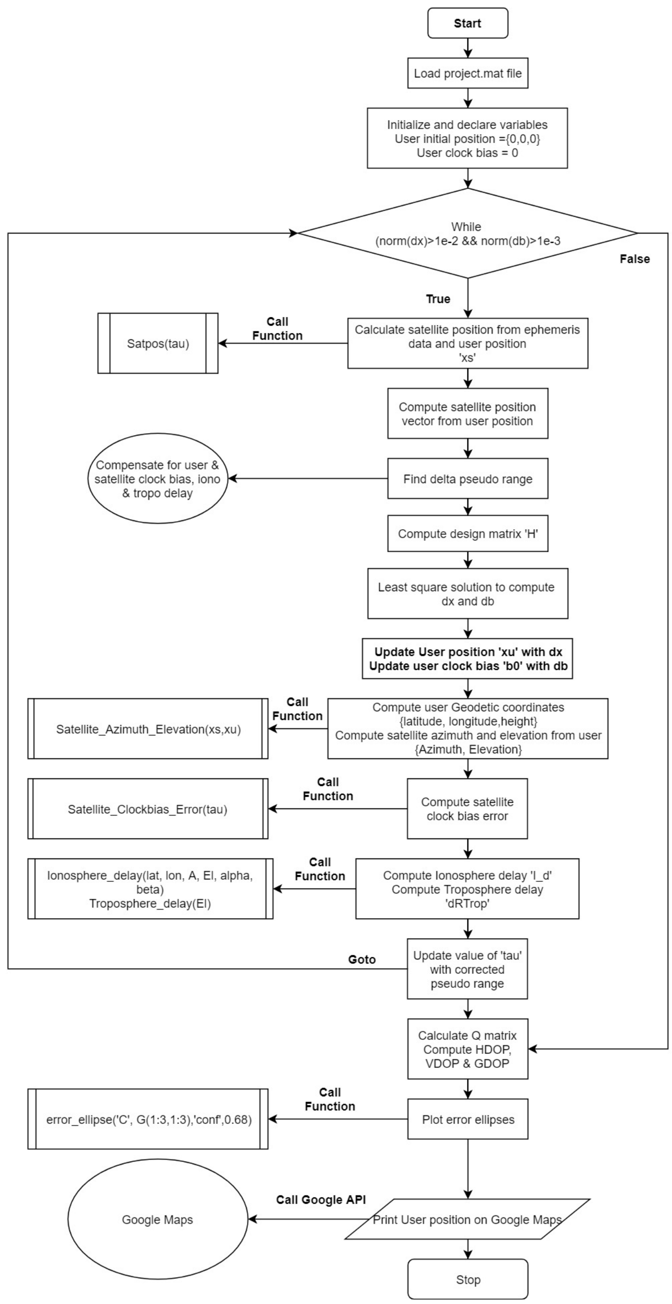

Figure 3. Flowchart for matlab code

4. Components of the code

4.1. Ephemeris data from satellites

Project_data· mat file consists of ephemeris data from 6 satellites. It also includes values of pseudorange measured by receiver as "pr". Iono error constants as "iono". Theses values are extracted into respective variables.

4.2. Initial position of user and user clock bias

For starting the algorithm initial position of user is geocentric origin of the ECEF coordinate system i.e [0,0,0]. User clock bias is also assumed to be [0].

4.3. Satellite position from Ephemeris data

Calculating Satellite position from ephemeris is given in [3] i.e IS-GPS-200D. Following images taken from [3] tell about the parameters involved in finding coordinates and also the equations for calculation. Output x^k, y^k, z^k are the ECEF coordinates. "Satpos.m" function does the calculation. The input for this function is τ which is updated in each iteration.

TABLE I
COMPONENTS OF EPHEMERIS DATA

Name	Description	Units
M_0	Mean anomaly at reference time	Semicircle
Δn	Mean motion difference from computed value	Semicircle/s
e	Eccentricity	Dimensionless
$\sqrt{a}$	Square root of semimajor axis	$m^{1/2}$
Ω_0	Longitude of ascending node of orbit plane at weekly epoch	Semicircle
i_0	Inclination angle at reference time	Semicircle
ω	Argument of perigee	Semicircle
$\dot{\Omega}$	Rate of right ascension	Semicircle/s
$IDOT$	Rate of inclination angle	Semicircle/s
C_{uc}	Amplitude of cosine harmonic correction term to the argument of latitude	Rad
C_{us}	Amplitude of sine harmonic correction term to the argument of latitude	Rad
C_{rc}	Amplitude of cosine harmonic correction term to the orbit radius	m
C_{rs}	Amplitude of sine harmonic correction term to the orbit radius	m
C_{ic}	Amplitude of cosine harmonic correction term to the angle of inclination	Rad
C_{is}	Amplitude of sine harmonic correction term to the angle of inclination	Rad
t_{0e}	Ephemeris reference time	s
$IODE$	Issue of data, ephemeris	Dimensionless

Figure 4.

TABLE 2
ALGORITHM FOR COMPUTING SATELLITE POSITION

Equation	Description
$\mu = 3.986008 \times 10^{14} m^3/s^2$	WGS 84 value of earth's universal gravitational parameter
$\dot{\Omega}_e = 7.292115167 \times 10^{-5} rad/s$	WGS 84 value of earth's rotation rate
$a = (\sqrt{a})^2$	Semimajor axis
$t_{n+1} = t - t_{0e}$	Time from ephemeris reference epoch
$f_n = \tan^{-1}\left\{\frac{\sqrt{(1-e^2}\sin E_n/(1-e\cos E_n)}{(\cos E_n - e)/(1-e\cos E_n)}\right\}$	True anomaly
$E_n = \cos^{-1}(\frac{e+\cos f_n}{1+e\cos f_n})$	Eccentric anomaly from cosine
$\phi_n = f_n + \omega$	Argument of latitude
$\delta\mu_n = C_{\mu C}\cos 2\phi_n + C_{\mu S}\sin 2\phi_n$	Second-harmonic correction to argument of latitude
$\delta r_n = C_{rC}\cos 2\phi_n + C_{rS}\sin 2\phi_n$	Second-harmonic correction to radius
$\delta i_n = C_{iC}\cos 2\phi_n + C_{iS}\sin 2\phi_n$	Second-harmonic correction to inclination
$\mu_n = \phi_n + \delta\mu_n$	Corrected argument of latitude
$r_n = a(1-e\cos E_n) + \delta r_n$	Corrected radius
$i_n = i_0 + \delta i_n + (IDOT)t_n$	Corrected inclination
$x_n' = r_n \cos\mu_n$	X coordinate in orbit plane
$y_n' = r_n \sin\mu_n$	Y coordinate in orbit plane
$\Omega_n = \Omega_0 + (\dot{\Omega} + \dot{\Omega}_e)t_n - \dot{\Omega}_e t_{0e}$	Corrected longitude of ascending node
$x_n = x_n'\cos\Omega_n - y_n'\cos i_n \sin\Omega_n$	ECEF X coordinate
$y_n = x_n'\sin\Omega_n + y_n'\cos i_n \sin\Omega_n$	ECEF Y coordinate
$z_n = x_n'\sin i_n$	ECEF Z coordinate

Figure 5.

4.4. Compute satellite position vector from user

Satellite position obtained in the previous step is utilized to find the position vector from user to satellite.

4.5. Calculate the $\delta\rho$

The equation used to calculate the $\delta\rho$ is mentioned:

$$\delta\rho = \rho_{measured} - \rho_{corrected} \tag{1}$$

$$\rho_{measured} = pr$$

$$\rho_{corrected} = \sqrt{(x^k - x_u)^2 + (y^k - y_u)^2 + (z^k - z_u)^2} + (b_0 - c * (\delta T^{st}_{clk})) + c * I_d + \delta RTrop$$

where

$$\begin{aligned} xu = x_u, y_u, z_u &= \text{User's coordinates in ECEF system} \\ xs = x^k, y^k, z^k &= \text{Satellite's coodinates in ECEF system} \\ b_0 &= \text{User's clock bias} \\ c &= \text{Speed of light} \\ \delta T^{st}_{clk} &= \text{Satellite's clock bias error} \\ I_d &= \text{Ionosphere delay} \\ \delta RTrop &= \text{Troposphere delay} \end{aligned}$$

4.6. Compute design matrix H

$$H = [-\text{Satellites position unit vectorones(1,6)}]_{\text{6x4}} \tag{2}$$

4.7. Least square solution

$$\delta r = (H^T \cdot H)^{-1} \cdot H^T \cdot \delta\rho \tag{3}$$

4.8. Update user position and user clock bias error

$$xu = xu + \delta x$$
$$b_0 = b0 + \delta b$$

where

$$\delta x = \delta r(1:3)$$
$$\delta b = \delta r(4)$$

4.9. Calculate satellite Azimuth and Elevation from user position

This is calculates by Satellite_Azimuth_Elevation function. The inputs to this function are user and satellite position in ECEF coordinates i.e xu & xs. Output of this function is satellite Azimuth “A”, Elevation “E” in radians and users latitude “lat”, longitude “lon” in degrees and height “h” in meters. This is done in three steps

1) Convert users ECEF coordinates to geodetic [lat, lon, h]
2) Convert the satellites position vector coordinate from user position into ENU coordinate system [E,N,U]
3) Calculate Azimuth and Elevation

$$A = \text{atan2(E,N)} \tag{4}$$

$$E = \text{asin(U/norm[E,N,U])} \tag{5}$$

4.10. Calculate clock bias error of satellites

The function used to calculate this is Satellite_Clockbias_Error. Input to this function is “τ” and output is δT^{st}_{clk} in seconds. The equation utilized to calculate is

$$\delta T^{st}_{clk} = af0 + af1 \cdot T + af2 \cdot (T)^2 + F \cdot e \cdot sqrtA \cdot sin(Ek) - Tgd \tag{6}$$

The values of constants is taken from project_data· mat file. The value of input “τ” is updated in each iteration

4.11. Calculate Ionosphere delay

This is done by the Ionosphere_delay function. There are 6 inputs to this function; user's latitude (lat), longitude (lon) in degrees and satellite azimuth (A) and elevation (E) in radians, alpha, beta values from iono matrix. The output of this function is I_d in seconds. Klobuchar's Algorithm [4] is utilized to calculate delay.

4.12. Calculate Troposphere delay

The Troposphere_delay function calculates it. The input to this function is the elevation (E) in radians of the satellite from user position. The output of this function is the δ RTrop in meters. the equation used to calculate is:

$$\delta RTrop = 2.312/sin(\sqrt{E \cdot E + 1.904e-3}) + 0.084/sin(\sqrt{E \cdot E + 0.6854e-3}) \tag{7}$$

4.13. Update value of τ for next iteration

$$\tau_{new} = \tau_{old} + \delta\rho/c \tag{8}$$

For each iteration a new value of τ is calculated which is passed to all the depending functions.

5. Results

1) The iteration runs for 5 times to reduce the value of δx and δb to the required threshold.
2) The final coordinates of the receiver in ECEF system are :

$$x = 3.8952 * 1.0e + 06$$
$$y = 0.3147 * 1.0e + 06$$
$$z = 5.0241 * 1.0e + 06$$

and in geodetic system i.e [lat,lon,h] are:

$$lat = 52.3093\,{}^{\circ}N$$
$$lon = 4.6197\,{}^{\circ}E$$
$$h = 197.7866\,m$$

3) These coordinates are plotted on Google maps as shown below by a red point:

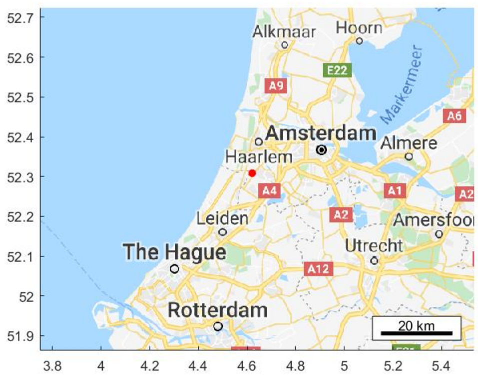

Figure 6. Final location on map

6. Error Analysis

1) Covariance matrix is calculated as follows:

$$G = (H^T \cdot H)^{-1} \tag{9}$$

$$G = \begin{Bmatrix} 10.3175 & 2.1986 & 4.7947 & 8.1054 \\ 2.1986 & 0.9658 & 1.1574 & 1.8529 \\ 4.7947 & 1.1574 & 3.7528 & 4.3210 \\ 8.1054 & 1.8529 & 4.3210 & 6.7478 \end{Bmatrix}$$

2) The values of DOP are also acceptable as calculated using formulas below:

$$HDOP = \sqrt{G(1,1) + G(2,2)} = 3.3591$$
$$VDOP = \sqrt{G(3,3)} = 1.9372$$
$$TDOP = \sqrt{G(4,4)} = 2.5977$$
$$PDOP = \sqrt{G(1,1) + G(2,2) + G(3,3)} = 3.8776$$
$$GDOP = \sqrt{G(1,1) + G(2,2) + G(3,3) + G(4,4)} = 4.6673$$

3) 1-sigma (68.3%)Error ellipses are plotted using the error_ellipse [5] function. Plots are given below:

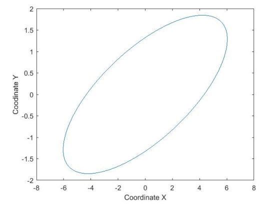

Figure 7. X-Y coordinate variance

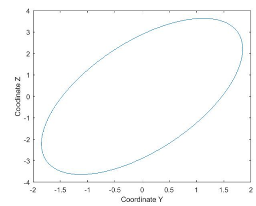

Figure 8. Y-Z coordinate variance

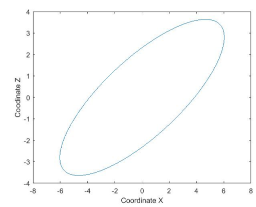

Figure 9. X-Z coordinate variance

7. Conclusion

The ephemeris data received from satellites was utilized to calculate the satellite position, satellite clock bias error, ionosphere delay, troposphere delay and then using pseudorange equation and least square solution user coordinates and user clock bias was calculated. GPS user location was ploted on Google maps using Google API. Error analysis was successfully carried out using dilution of precision and error ellipses. The values of DOP were also acceptable. The MATLAB code has been added in the appendix.

References

[1] https://www.sxbluegps.com/technology/gps-the-error-budget/

[2] https://www.unavco.org/education/resources/tutorials-and-handouts/tutorials/elevation-and-geoid.html

[3] https://www.navcen.uscg.gov/pdf/IS-GPS-200D.pdf

[4] https://gssc.esa.int/navipedia/index.php/Klobuchar_Ionospheric_Model

[5] https://www.mathworks.com/matlabcentral/fileexchange/4705-error_ellipse