# Project 2: Calculating GPS Satellite Position

## Objectives

1. Become familiar with MATLAB programming (if you’re not already).
2. Develop a routine to determine satellite position (which will be used throughout the rest of the course).
3. To develop a better understanding of coordinate frames and timing issues

## Collaboration

This is an individual lab. You are allowed to discuss any aspect of the lab with other students, and you may look at each other’s source code for debugging purposes. However, your programming must be your own (i.e., you may not copy or transcribe someone else’s program, in part or in whole).

## Overview

You will be calculating satellite positions using broadcast ephemeris data throughout this course, and the purpose of this lab is to develop a tool (i.e., computer program) for doing this. You will write a MATLAB function which can be called to calculate the coordinates of the satellite for use in positioning applications. It will also calculate the satellite clock error, which will be needed in future labs.

## Satellite Position Function Description

Your assignment is to write the following MATLAB function:

function [sv\_pos, sv\_clock\_err] = calc\_sv\_pos(prn, transmit\_time, rcvr\_pos)

% [sv\_pos, sv\_clock\_err] = calc\_sv\_pos(prn, transmit\_time, rcvr\_pos)

%

% This function returns the position of a satellite at a

% particular GPS time, expressed in the ECEF coordinate frame

% valid at the time the signal was received by a GPS receiver.

%

% Input parameters:

% prn - PRN of desired satellite

% transmit\_time – Desired time of transmission (GPS week

% seconds)

% rcvr\_pos - 1x3 ECEF position vector of the receiver

%

% Output parameters:

% sv\_pos - 1x3 ECEF position vector of satellite at time

% of transmission, expressed in ECEF

% coordinate frame at time of reception (m)

% sv\_clock\_err - SV clock error at the time of transmission,

% calculated from the clock error terms of the

% navigation message

Hint: If you copy the source as given directly above, then you will have a useful output when you type “help calc\_sv\_pos” in MATLAB. This document is located in Canvas, so you can cut-and-paste it if desired.

The algorithms for calculating the ECEF satellite position are provided in IS-GPS-200 (Table 20-IV), which can be obtained from the internet using the link given in the slides. In addition, you should reference the class slide handouts to get additional information that you’ll need to complete this lab. Note that the coordinates calculated by the equations in IS-GPS-200D (xk, yk, and zk) are the ECEF coordinates of the satellite at the time of transmission (shown as xt, yt, and zt in the slide titled “Coordinate Frame Rotation” in the fourth video). You will need to perform the rotation to express them as the ECEF coordinates at the time of reception (xr, yr, and zr).

The algorithms for calculating the SV clock correction (tsv) are found in the section entitled “User Algorithm for SV Clock Correction” of IS-GPS-200. Note that you need the eccentric anomaly (an intermediate calculation when calculating the position) in order to do this. This SV clock correction represents a significant error that needs to be removed if anything useful is done with GPS measurements.

## Obtaining the Ephemeris and Clock Correction Data

The raw ephemeris data and a MATLAB routine to access it can be found in Canvas. The raw ephemeris file is called 20jan99.eph. The MATLAB functions “load\_eph\_file” and “current\_ephemeris” are provided and are used to load up the raw ephemeris data and put it into a format easy to use in matlab.

The load\_eph\_file function reads all of the ephemeris data from the disk and saves it off in memory for later use. The current\_ephemeris function then retrieves the currently stored ephemeris for a particular satellite (PRN). The idea is that at the beginning of your MATLAB program, you call load\_eph\_file once. Then, each time you need an ephemeris (which will normally be only in the calc\_sv\_pos function), you retrieve the appropriate ephemeris record using the efficient current\_ephemeris function. Use the help function for more information about these functions.

For this lab, you should make the following MATLAB function call to load up the ephemeris:

load\_eph\_file(’20jan99.eph’, 252000);

This will load the ephemeris record for each PRN which has a t0e closest to 252000. (Note that the t0e will not be exactly at 252000, just somewhat close to it). Ephemeris variables can then be accessed using the current\_ephemeris function. For example, the matlab code

eph=current\_ephemeris(10);

will create a record of data called eph that has all of the ephemeris data corresponding with PRN 10.

For example, eph.i0 is the I0 ephemeris parameter for PRN 10. For a more detailed description type “help load\_eph\_file”.

Important Note: The call to load\_eph\_file should NOT be included within your calc\_sv\_pos function. The load\_eph\_file routine is designed to be called just once at the beginning of a program. The current\_ephemeris routine, on the other hand, can (and should) be used within the calc\_sv\_pos function. It is specifically designed to require very little processing time.

## Checking Your Work

Below is a sample run from a satellite positioning algorithm that you can use to check your work.

Eph File : 20jan99.eph

PRN : 4

Transmit time : 252000.000

Rcvr Position : -1485881.487 -5152018.353 3444641.847

**SV Coordinates : -7204040.857 -19188608.082 -16682791.631**

**SV Clock Error : 0.00043056387 (seconds)**

Hint: Here is sample source code that would generate the above printout. You should not have this in your function, but you could use something like it in a test program which would call your function.

Sample display code (to display the Rcvr Position line:

disp(sprintf(’Rcvr Position : %.3f %.3f %.3f’, rcvr\_pos))

## What to Turn In

For this project, please submit your function as a .m file in Canvas. Please name it calc\_sv\_pos\_<lastname>.m, where “<lastname>” is your last name in lower case.

If I were turning in the lab, the filename would be calc\_sv\_pos\_leishman.m.

Note that I only want your function so that I can call it myself, not a program that will generate an output similar to that shown in the “Checking Your Work” section above. In fact, your function should not print out anything at all—it should just return the output vectors.

## Grading

You will be graded primarily upon accuracy. If you get the correct positions for satellites and times that I pick when I run your function, then you will get full credit for the lab. I will try several different PRNs at different times. If you do not get the correct answers, then I will look at your source code and make a judgement on your grade based on what I see. Feel free to include any comments that you would like in

Canvas. Also, make sure that your function works with a 1x3 rcvr\_pos vector and returns a 1x3 sv\_pos vector. Three points will be deducted if either of these is wrong. Also make sure that your function can handle end of week rollover (as described in the ICD-GPS-200D) for both the satellite position and clock error calculations. Failure to do so will result in a three point deduction for each case in which the error is made (position and clock error).