Palestine Polytechnic University

College of Engineering and Technology

Civil Engineering Department

Surveying Engineering

Introduction/Graduation Project

DEVELOPING VISUAL BASIC SOFTWARE SOLUTION FOR GPS CALCULATIONS

Prepared By:

Hasan Awad

Supervisor :

Dr.Mutaz Qafisheh

Hebron-Palestine

2021-2022

ACKNOWLEDGMENT

I would like to request to Dr. Moataz Qafisha, who sincerely appreciates his many essential suggestions and advice. I am very grateful for his generosity.

Finally, my thanks to the members of the Civil Engineering Department, especially the Surveying Department, who helped me design my Project.

Hasan Awad

ABSTRACT

Developing Visual Basic Software Solution For GPS Calculations

Hasan Awad

Palestine Polytechnic University

Supervisor :

Dr. Mutaz Qafisheh

The problem in this project is the manual solution of complex GPS equations, where a program was created to facilitate finding the solution and giving correct and accurate results. And the results that were reached, the conversion between satellite time and local time and vice versa, Convert Earth Centered Earth Fixed (X,Y,Z) to Latitude , Longitude and Height , and vice versa , Familiarize yourself with GPS satellite orbits and sky plot , and Analysis of Position , Velocity and Time (PVT) solution methods for navigation message .

ملخص:

المشكلة في هذا المشروع هي الحل اليدوي لمعادلات GPS المعقدة ، حيث تم إنشاء برنامج لتسهيل إيجاد الحل وإعطاء نتائج صحيحة ودقيقة. والنتائج التي تم الحصول عليها ، التحويل بين توقيت القمر الصناعي والتوقيت المحلي والعكس ، تحويل (ECEF) إلى خطوط الطول والعرض والارتفاع (X، Y، Z)، والعكس ، التعرف على مدارات القمر الصناعي GPS ومخطط السماء ، وتحليل الموقع والسرعة وطرق حل الوقت (PVT) لرسالة الملاحة.

List of tables

List of figures

CHAPTER

1 *INTRODUCTION*

1.1 Background

1.2 Objective

1.3 Methodology

CHAPTER ONE

INTRODUCTION

1.1 Background

A Global Positioning System, also known as GPS, is a system of satellites designed to help navigate earth, air, and water. A GPS receiver shows where it is. It may also show how fast it is moving, which direction it is going, how high it is, and maybe how fast it is going up or down. Many GPS receivers have information about places. GPSs for automobiles have to travel data like road maps, hotels, restaurants, and service stations. GPSs for boats contain nautical charts of harbors, marinas, shallow water, rocks, and waterways. Other GPS receivers are made for air navigation, hiking, and backpacking, bicycling, or many other activities. The majority are on smart phones .Most GPS receivers can record where they have been, and help plan a journey. While traveling a planned journey.

A Global Positioning System (GPS) is a space-based satellite navigation system that provides position and time information in all weather conditions, anywhere on or near earth where there is an unobstructed line of sight for four or more GPS satellites. The system provides critical capabilities for military, civilian, and commercial users around the world. It is maintained by the U.S. government and is freely accessible to anyone with a GPS receiver.The GPS project was developed in 1973 to overcome the limitations of earlier navigation systems, incorporating ideas from several predecessors, including a number of classified engineering design studies from the 1960s. GPS was created and realized by the U.S. Department of Defense (DOD) and originally operated with 24 satellites. It began operating at full capacity in 1994. It is credited with inventing Bradford Parkinson, Roger L. Easton, and Evan A. Advances in technology and new demands on the existing system have led to efforts to modernize the GPS system and implement the Next Generation GPS III satellite and Next Generation Operational Control System (OCX). Advertisements From Vice President Al

Gore In 1998 these changes began. In 2000, the U.S. Congress authorized a modernization effort, GPS III.In addition to GPS, other systems are in use or under development. The Russian Global Navigation Satellite System (GLONASS) was developed in conjunction with the Global Positioning System (GPS), but suffered from incomplete coverage of the earth until the mid-2000s. There's also the European Union's Galileo GPS, China's Compass navigation system, and India's regional navigation satellite system.

This Project is a GPS Calculations tool programmed using Visual Basic 2012 programming language . This tool has the following functionalities ; the first one to convert (year, month, day and time[1]) To (J.D. , MJD , GPS JD , GPS week , GPS day) . The second functionality calculates (Latitude, Longitude, and Height) and converts them to (X , Y, and Z) .The third functionality calculates [north , east, and up] and convert them to azimuth and elevation angle, draws sky plot and calculates Velocity and Precise Time (PVT) solutions .

[1]The time must be entered in hours, minutes and seconds.

1.2 Objective

This Project deals with the Visual Basic 2012 programming language . This tool should be able the following GPS Calculations ,

1. **Convert (year, month, day, and time) To (J.D. , MJD , GPS JD , GPS week , GPS day).**
2. **Convert (Latitude, Longitude, and Height) to (X , Y, and Z) and vice versa.**
3. **Convert [north , east, and up] coordinates to azimuth and elevation angle.**
4. **Calculates Velocity and Precise Time (PVT) solutions .**
5. **Knowing the study area .**
6. **Software Solution For GPS Calculations Tutorials .**

1.3 Methodology

This Project has the following scope:

Chapter 1 : This chapter introduces the Project.

Chapter 2 : This chapter explains the time and referents frame .

Chapter 3 : This chapter describes coordinate conversion.

Chapter 4 : This chapter shows the results of azimuth and elevation angle .

Chapter 5 : This chapter Calculates PVT Solution Methods for Navigation Message of GPS Receiver.

Chapter 6 : This chapter shows study area .

Chapter 7 : This chapter shows Software Solution For GPS Calculations Tutorials .

CHAPTER

2 *Time and Reference Frame*

2.1 Introduction

2.2 Time in GPS

CHAPTER TWO

TIME AND REFERENCE FRAME

2.1 Introduction

Accurate and well-defined time references and coordinate frames are essential in GNSSs, where positions are computed from signal travel time measurements and provided as a set of coordinates.

2.2 Time in GPS

Everyday life follows the alternation of day[2] and night, and the seasons of the year, so the initial conception of time was based on the motion of the Sun. However, as science and technology evolved, more precise, uniform, and well-defined time scales were needed. Several time references are currently in operation, based on different periodic processes associated with Earth's rotation, celestial mechanics, or transitions between the energy levels in atomic oscillators.

Table 2.1- Different time systems

Periodic	Time
Earth rotation	Universal Time (UT0, UT1, UT2) Greenwich Sidereal Time (Θ)
Earth revolution	Terrestrial Dynamic Time (TDT) Barycentric Dynamic Time (TDB)
Atomic oscillators	International Atomic Time (TAI) Coordinated Universal Time (UTC) GNSS Reference Time

[2] **one mean sidereal day = 1 mean solar day – $3^{m}56^{s}.4$**

Table 2.2-The following relationships have been extracted

TAI=UTC+1^{s}X n
TAI=TDT−32^{s} .184
UTC=UT1+*dU*T1
|*dU*T1| < 0^{s}.9

But , where n is the number of leap seconds introduced for a given epoch

e.g / 1 Jan 1999 n = 32

GPS Time (GPST) is a continuous time scale (no leap seconds) defined by the GPS control segment on the basis of a set of atomic clocks at the MCS and onboard the satellites. It starts at 0^{h} UTC (midnight) of 5–6 January 1980 (6^{d}. 0). At that epoch, the difference TAI−UTC was 19s, hence GPS−UTC = n−19^{s}. GPST is synchronized with UTC(USNO) at the 1ms level.

In order to facilitate calculations for long time intervals, the Julian date is used (after Julio Scalier). It has as reference epoch the 1st of January of 4713 BC and, starting from there, days are counted in a correlative way. The Julian Day (J.D.) starts at 12^{h} of the corresponding civil day (e.g. 6^{d} · 0 January 1980 = JD 2444244.5). The current reference standard epoch for the scientific community is

$$J2000.0 = 1^{d}.5 \text{ January } 2000 = JD\ 2451545.0 \qquad (2.1)$$

The Modified Julian Day (MJD) is also used, and is obtained by subtracting 2400000.5 days from the J.D.

used equations

$$JD = int[365.25\ \times y] + int[30.6001\ \times (m+1)] + DD + \frac{UT(h)}{24} + 1720981.5 \qquad (2.2)$$

Where $y = YY - 1\ , m = MM + 12\ ,\ MM \leq 2$

$y = YY\ , m = MM\ , \qquad MM > 2$

When :Julian day (JD) ,year (Y) , month (M) , day (D) , time (U.T.)

From the J.D. and taking into account that GPS reference date ($6^{d} \cdot 0$ January 1980) corresponds to JD 2444244.5, one can immediately obtain the GPST. Then, using modulo 7, the GPS week can be found.

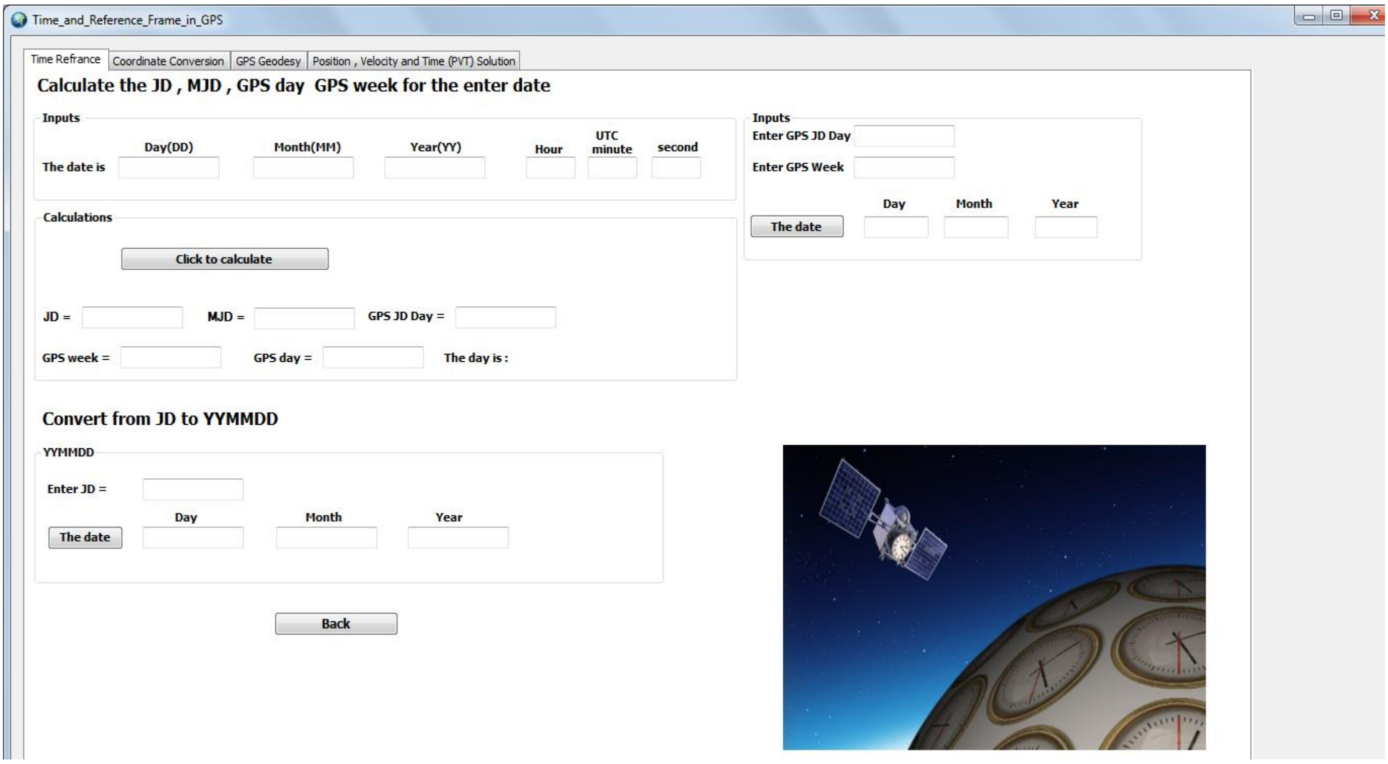


Figure 2.1 The program for calculating the reference time in GPS .

For a given civil date (year (Y) , month (M) , day (D), time(U.T.) , conversion to J.D. is given by :

$$\mathbf{JD = int[365.25 \times y] + int[30.6001 \times (m+1)] + DD + \frac{UT(h)}{24} + 1720981.5}$$

Where $\mathbf{y = YY - 1, m = MM + 12, MM \leq 2}$

$\mathbf{y = YY, m = MM, MM > 2}$

Then :

Modified Julian Day (**MJD**)

$\mathbf{MJD = JD - 2400000.5}$ **When :** **(2.3)**

<u>Inverse conversion , from J.D. to civilian date :</u>

$\mathbf{a = int\,(JD + 0.5)}$ **(2.4)**

$\mathbf{b = a + 1537}$ **(2.5)**

$\mathbf{c = int[(b - 122.1)/(365.25)]}$ **(2.6)**

$\mathbf{d = int(365.25 \times c)}$ **(2.7)**

$\mathbf{e = int[(b - d)/(30.6001)]}$ **(2.8)**

$\mathbf{D = b - d - int(30.600 \times e) + frac(JD + 0.5)}$ **(2.9)**

$\mathbf{M = e - 1 - 12 * int(e/14)}$ **(2.10)**

$\mathbf{Y = c - 4715 - int[(7 + M)/10]}$ **(2.11)**

$\mathbf{Day\ of\ the\ week\ (N) = modulo\{int(JD + 0.5), 7\}}$ **(2.12)**

$\mathbf{GPS\_WEEK = int[(JD - 2444244.5)]/7}$ **(2.13)**

<u>CHAPTER</u>

3 *Coordinate Conversion*

3.1 Introduction

3.2 From ECEF to Ellipsoidal

3.3 From ECEF to Topocentric

CHAPTER THREE

COORDINATE CONVERSION

3.1 Introduction

In geodesy, conversion among different geographic coordinate systems is made necessary by the different geographic coordinate systems in use worldwide and over time. Coordinate conversion is composed of several different types of conversion: format change of geographic coordinates, conversion of coordinate systems, or transformation to different geodetic datums. Geographic coordinate conversion has cartography, surveying, navigation and geographic information systems. Applications In geodesy, geographic coordinate conversion is translation among different coordinate formats or map projections all referenced to the same geodetic datum. A geographic coordinate transformation is a translation among different geodetic datums. Both geographic coordinate conversion and transformation.

3.2 From ECEF to Ellipsoidal

Computations when processing GPS data are typically done in a geocentric, EarthCentered, Earth-fixed system. The system has three right-handed orthogonal ses: X, Y, and Z. The Z axis coincides with Earth's rotation axis. The (O,X,Y) plane coincides with the equatorial plane. The (O,X,Z) plane contains the Earth's rotation axis and the prime meridian. Units are meters. In most applications, coordinates are, however, expressed as geodetic longitude, latitude, and height concerning a datum. The origin and the shape of the associated ellipsoid define the datum.

3.3 From ECEF to Topocentric

Positions are sometimes expressed in a local topocentric datum. The origin is any point one chooses on the surface of the earth. The datum has 3 left-handed orthogonal axes: u (for "up") is vertical and points upwards, n (for "north") is

in the local horizontal plane and points to the geographic north, e (for "east") is in the local horizontal plane and points to the geographic east. Units are meters.

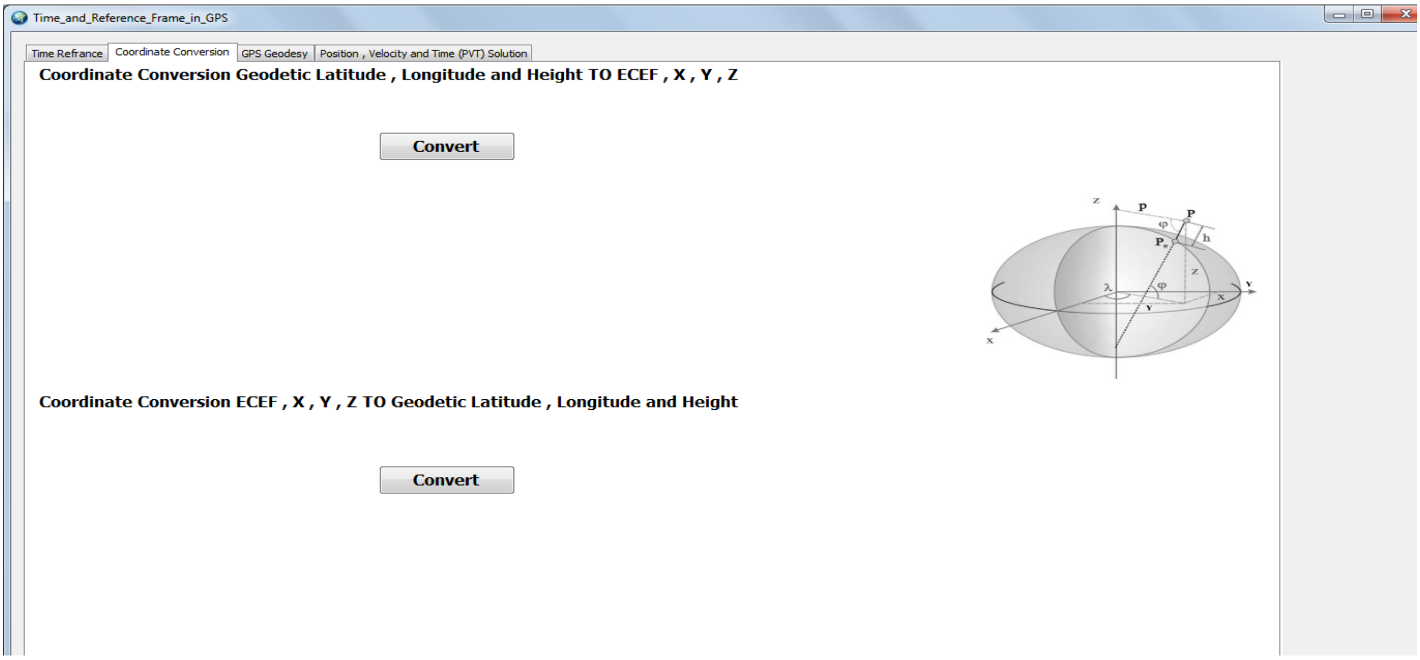


Figure 3.1 The program for calculating the Coordinate Conversion .

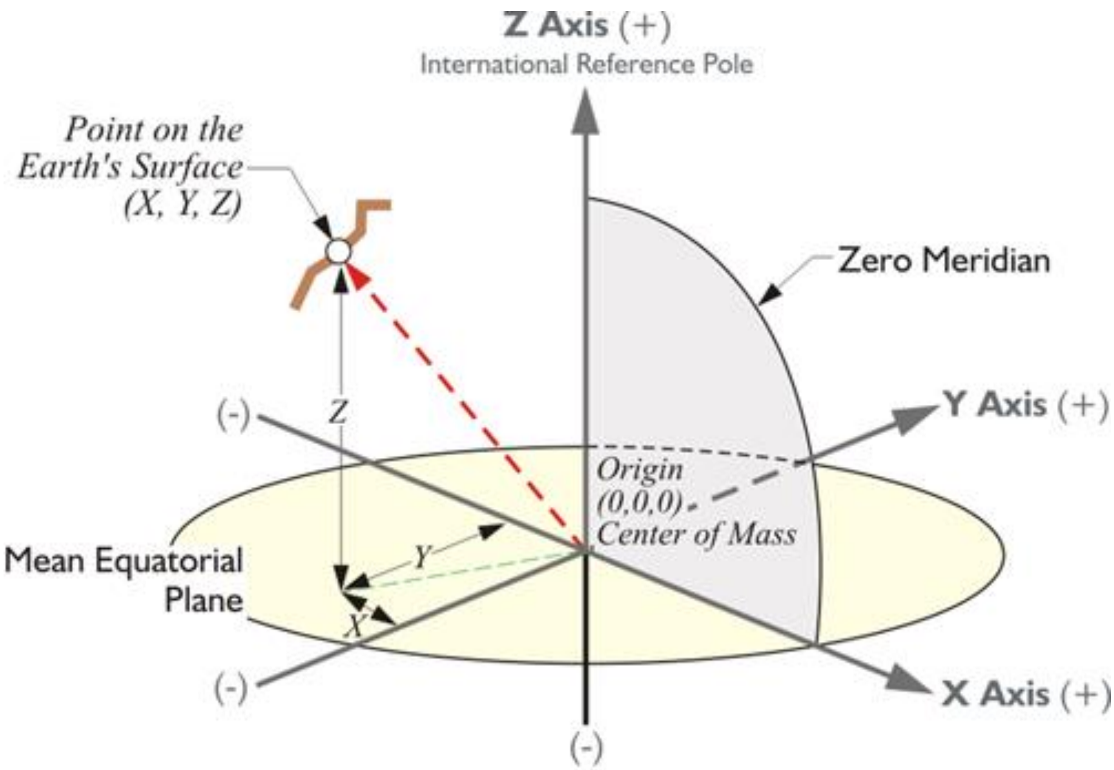


Figure 3.2 Cartesian Coordinate .

used equations

$$M(\emptyset) = \frac{a(1-e^2)}{(1-e^2 sin^2\ \emptyset)^{2/3}} \quad (3.1)$$

when :

M: Radius of curvature in the meridian .

a : Semi-major axis .

e^2: The first eccentricity squared

$\emptyset$: Latitude

$$N(\emptyset) = \frac{a}{\sqrt{(1-e^2 sin^2\ \emptyset)}} \quad (3.2)$$

when :

N : Radius of curvature in the prime vertical .

$$e^2 = \frac{a^2-b^2}{a^2} \quad (3.3)$$

when :

b: Semi-minor axis.

$$f = \frac{a-b}{a} \quad (3.4)$$

when :

f: The flattening of the ellipsoid .

$$X = (N + h) \cos \emptyset \cos \lambda \quad (3.5)$$

when :

X : Coordinates X .

h :Hight .

λ: Longitude .

$$Y = (N + h)\cos\emptyset \sin\lambda \quad (3.6)$$

when :

Y : Coordinates Y .

$$Z = ((1 - e^2)N + h)\sin\emptyset \quad (3.7)$$

when :

Z : Coordinates Z .

$$r = R + h \quad (3.8)$$

when :

R : Radius of earth=6371 km .

$$r = \sqrt[3]{X^2 + Y^2 + Z^2} \quad (3.9)$$

$$\lambda=\tan^{-1}\frac{Y}{X} \quad (3.10)$$

$$\emptyset=\tan^{-1}\frac{Z}{\sqrt{X^2+Y^2}}(1 - e^2)^{-1} \quad (3.11)$$

$$h = \frac{\sqrt{X^2+Y^2}}{\cos\emptyset} - N \quad (3.12)$$

CHAPTER 4 *GPS Geodesy*

4.1 Introduction

4.2 Limitations of the GPS measurement

4.3 Satellite SkyPlot

CHAPTER FOUR

GPS GEODESY

4.1 Introduction

GPS for Geodesy describes the use of Global Positioning System (GPS) measurements for geodetic applications. It covers the modelling and data processing strategies needed for the determination of precise coordinates for position and the monitoring of temporal changes of these coordinates in well-defined reference systems. complete observation equations are presented and discussed for the spectrum of geodetic applications, the GPS observables, the GPS orbits, the reference frames, and the GPS signal propagation characteristics. Models for single-receiver applications are followed by models for multi-receiver applications for which the network scale is used as the criterion. It includes the theory of GPS ambiguity resolution, the contribution of GPS to atmospheric modelling and GPS use for geodynamic applications. A file SP3 was used to obtain the coordinates of the satellite in orbit.

The main goals of satellite geodesy are:

1. **Determination of the figure of the Earth, positioning, and navigation (geometric satellite geodesy) .**
2. **Determination of geoid, Earth's gravity field and its temporal variations (dynamical satellite geodesy or satellite physical geodesy) .**
3. **Measurement of geodynamical phenomena, such as crustal dynamics and polar motion .**

Satellite geodetic data and methods can be applied to diverse fields such as navigation, hydrography, oceanography and geophysics. Satellite geodesy relies heavily on orbital mechanics.

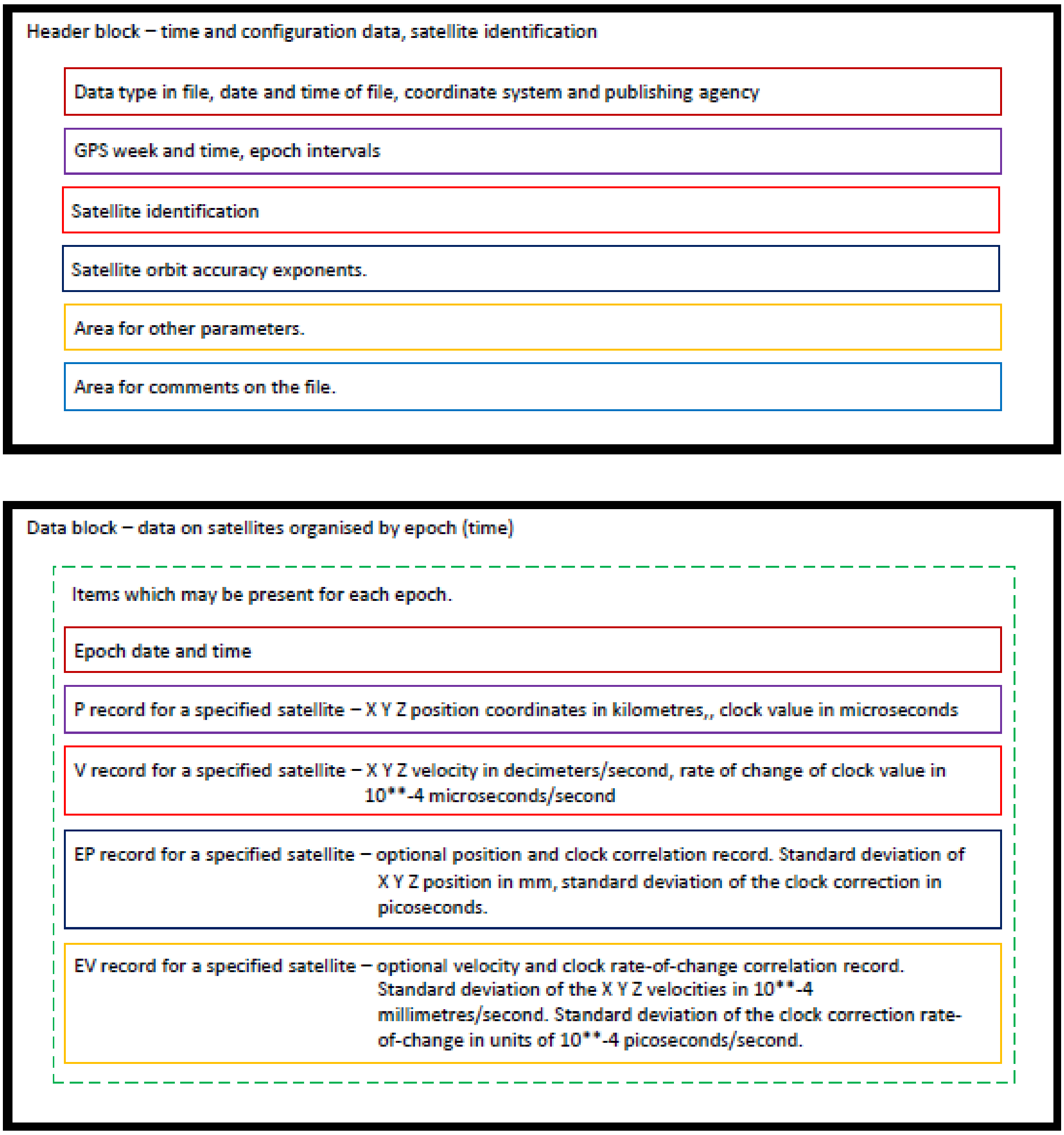


Figure 4.1 SP3-d file overview .

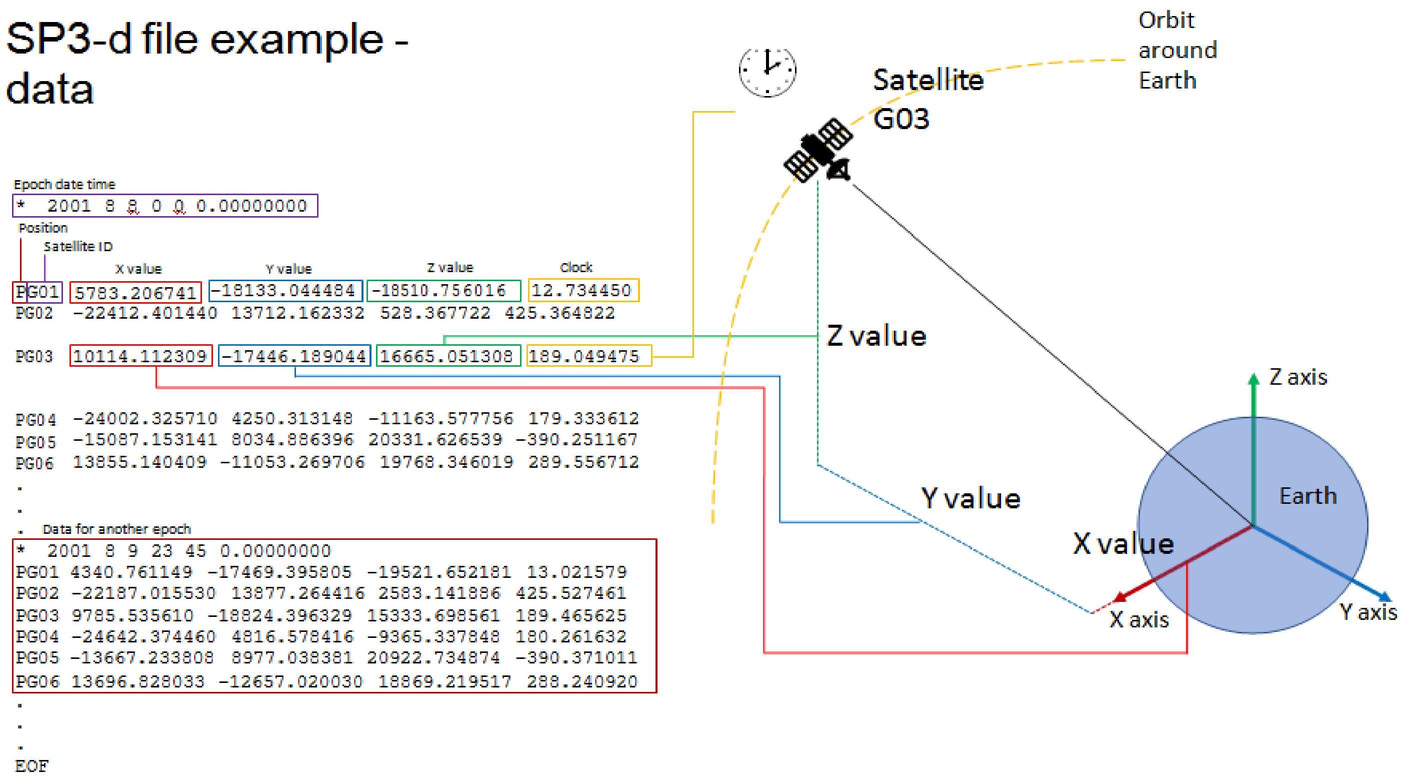


Figure 4.2 SP3-d file example - data .

SP3 file components :

```
#dp2001  8  8  0  0  0.00000000 192 ORBIT IGS97 HLM MGEX
## 1126 259200.00000000 900.00000000 52129 0.0000000000000
+  26 G01G02G03G04G05G06G07G08G09G10G11G13G14G17G18G20G21
+     G23G24G25G26G27G28G29G30G31  0  0  0  0  0  0  0  0
+       0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
+       0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
+       0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
++      7  8  7  8  6  7  7  7  7  7  7  7  7  8  8  7  9
++      9  8  6  8  7  7  6  7  7  0  0  0  0  0  0  0  0
++      0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
++      0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
++      0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
%c G  cc GPS ccc cccc cccc cccc cccc ccccc ccccc ccccc ccccc
%c cc cc ccc ccc cccc cccc cccc cccc ccccc ccccc ccccc ccccc
%f  1.2500000  1.025000000  0.00000000000  0.000000000000000
%f  0.0000000  0.000000000  0.00000000000  0.000000000000000
%i  0  0  0  0  0  0  0  0  0
%i  0  0  0  0  0  0  0  0  0
/* AN EXAMPLE ULTRA RAPID ORBIT, GPS ONLY.
/* NOTE THE "PREDICTED DATA" FLAGS FOR THE LAST EPOCH (IN COLUMNS 76 and 80).
```

- File version: SP3-d (#d)
- P: position data / V: velocity data (p)
- Time and date of data start: Year – month – day of month – hour – minute - second (2001 8 8 0 0 0.00000000)
- Number of epochs (time) contained in file (192)
- Type of data used – can be explained in the comment section. (ORBIT)
- Coordinate system used – typically realisations of ITRF2005 and ITRF2008. (IGS97)
- Orbit type: FIT (fitted), EXT (extrapolated), BCT (broadcast), HLM (Helmert) (HLM)
- Agency identifier (MGEX)
- Second record identifier (##)
- GPS week number (1126)
- Seconds within GPS week (259200.00000000)
- Epoch interval in seconds (900.00000000)
- Modified Julian day start (52129)
- Fractional part of Julian day start (0.0000000000000)
- Number of satellites in the file (26)
- Satellite identifiers: G – GPS, R – Glonass, E – Galileo, C – BeidouI – Indian IRNSS, J – Japanese QZSS, S – SBAS, L – LEO satellites
- Satellite line identifier. Must be a minimum of 5 lines – can identify up to 999 satellites. (+)
- 0s are the fillers to make up the 5 lines
- Satellite orbit accuracy exponent. Must be a minimum of 5 lines – matches up with the satellite identifiers above. (++)
- Satellite orbit accuracy exponents. For an exponent of 7, the accuracy is 2**7mm = ~1.3m
- 0s are the fillers to make up the 5 lines
- File type descriptor: G: GPS, M: mixed, R: Glonass, L: LEO, S: SBAS, I: IRNSS, E: Galileo,C: BeiDou, J: QZSS (G)
- Time system used. GPS: GPS, GLO: Glonass, GAL: Galileo, BDT: BeiDou, TAI: International Atomic time, UTC: UTC, IRN: IRNSS,QZS: QZSS (GPS)
- Floating-point base number used for computing the standard deviations for the components of the satellite position and velocity. (1.2500000)
- Floating-point base number for computing the standard deviations for the clock correction and the rate-of-change of the clock correction. (1.025000000)
- Comment lines starting with /* There should be a minimum of four up to as many are needed.

Website to get files SP3 :https://earthdata.nasa.gov/ .

4.2 Limitations of the GPS measurement

Obviously, the first limitation in GPS heightening is the quality of the GPS solutions used to obtain a height. Three broad categories of GPS observation types are possible:

- **Point Positioning is the stand alone navigation mode for which GPS was designed.**
- **Differential GPS (DGPS) uses a differential correction approach but is primarily based on pseudo range measurements**
- **GPS survey uses a different approach but is primarily based on the measurement of the phase of the GPS signals.**

While DGPS and even Point Positioning may be useful for producing heights in certain applications, the term GPS Heighting is typically taken to refer to the use of phase measurement techniques that can be grouped under the broad heading of GPS Surveying. This paper concentrates on heighting using these higher precision GPS Surveying techniques.

Within GPS Surveying, an overall consideration is whether the phase ambiguities have been resolved to integer values. Ambiguity resolution affects all three dimensions, not only height. For the measurement techniques known as Rapid Static and Real Time Kinematic (RTK), which are used for shorter baselines, ambiguity resolution is a prerequisite and should be achieved for most day to day GPS surveying applications. It is important to realize that RTK uses the smallest possible amount of data and even the best algorithms sometimes resolve the ambiguities incorrectly. To avoid such errors, which can reach the metre level, it is important to build redundancy into a survey by, for example, occupying stations more than once.

Two aspects that can affect the overall quality of the baseline solution are errors in the ephemeris or in the starting coordinates used in the processing. The effect of these can reach several parts per million and apply to all three dimensions.

Assuming that the broadcast ephemeris quality remains as high as in recent times, its effect will be minimal for most applications over short baselines. However, it should be noted that obtaining a WGS84 three-dimensional starting position of reasonable quality (say +/- 10m or better) could be more problematic in some areas of the world.

4.3 Satellite SkyPlot

When using a GPS, it is important to know how many satellites you are tracking and their location in the sky. Satellite close to the horizon, less than fifteen degrees, are less helpful in determining the probe's position due to the additional atmospheric interference. If there are too few satellites, the receiver will not be able to locate it. Typically five satellites are needed to locate and four to maintain a closed solution. Satellite Skyplot''s Skyplot's visual and graphic display helps determine when satellites are hidden by surrounding structures, trees, and mountains. The Skyplot satellite can be an invaluable tool to help you monitor the current configuration of a satellite.

The upper half of the skyplot screen adjacent to this window displays visible satellite information in the form of a chart. PRN is the satellite identification number. Azi is an acronym for SMT. The horizontal angle from north, in degrees measured clockwise, to the position of the satellite (0 to 360 degrees). Elv is an abbreviation for elevation; The vertical angle above the horizon where the satellite can be found (0 to 90 degrees). One entry is shown for each satellite tracked by the receiver.

The image in the lower half of the window displays the same information graphically. It shows a map of the sky with north at the top and east to the right. The central point, where the lines intersect, is a straight line. Each satellite appears as a symbol that looks like an "H". As you can see, most visible satellites were in the northeast when this image was taken. The inner circle represents a rise of sixty degrees. The outer circle is the horizon. Roughly speaking, any "H" that touches this circle is too low in the sky to be of much use. For GPS receivers that support GPS satellites.

For some types of GPS receivers, the receiver will only report the satellites that are used to calculate the location and which are being tracked. The satellite may only be tracked and not part of the solution if the satellite is very low on the horizon or when the signal is unclear. The sky chart will highlight the satellites that are part of the calculations.

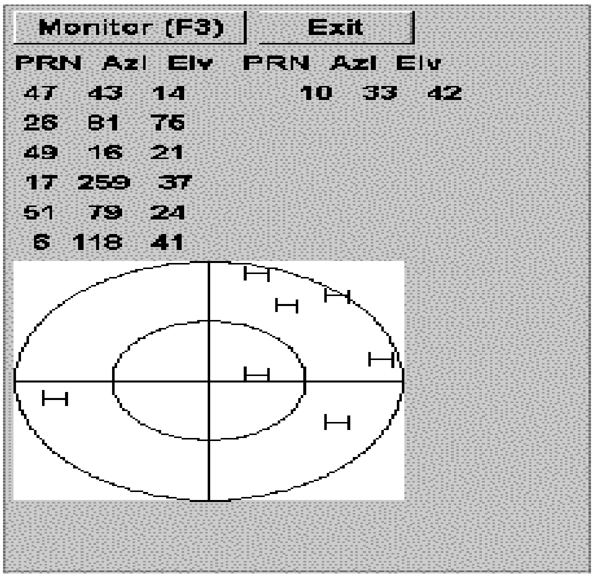


Figure 4.3 Satellite SkyPlot .

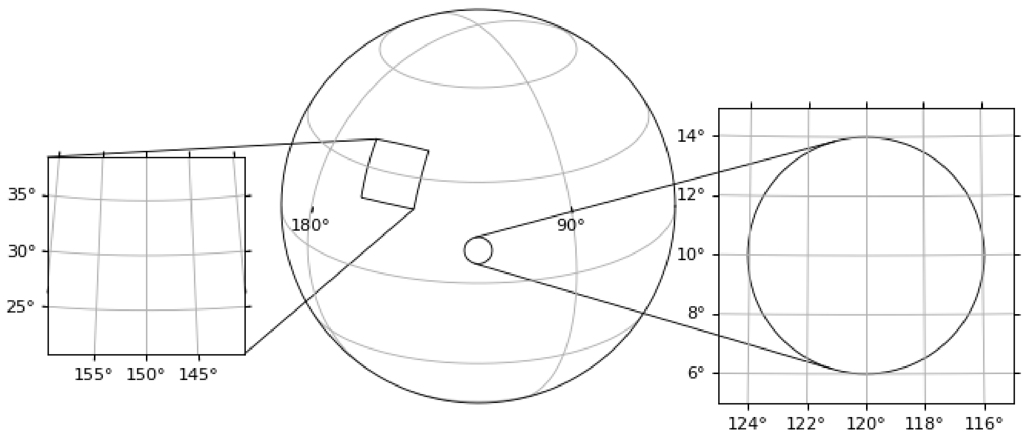


Figure 4.4 Sky Map Plotting .

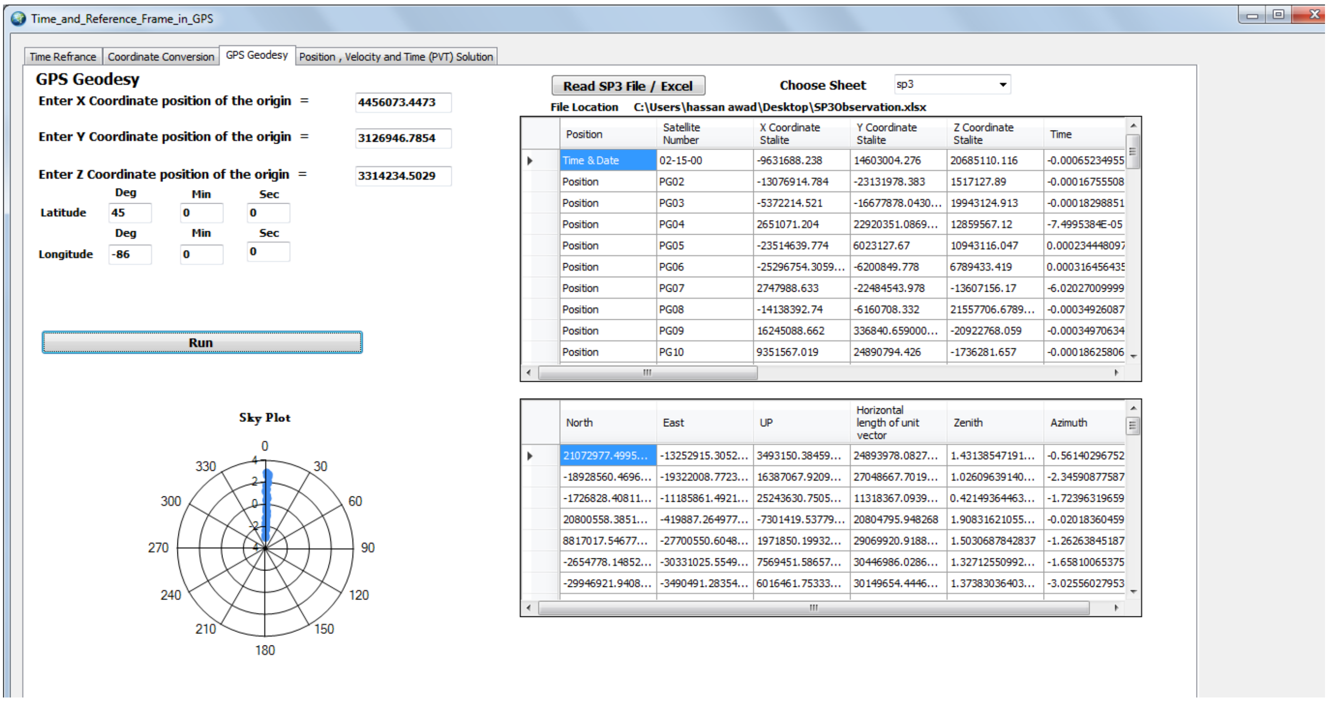


Figure 4.5 The program for calculating the GPS Geodesy .

The purpose of this is to begin learning about GPS satellite orbits. GPS orbits are distributed in various shapes, the simplest format, called "sp3", provides a position (X, Y, Z) of the satellite Center of mass under the ECEF every 15 minutes. This is the format used by the international GNSS service.

Calculate Sky plot:

1. **Extract satellite position information from sp3 file.**
2. **Compute ground station to satellite unit vector in geocentric XYZ coordinates.**
3. **Convert that vector to a unit vector (*i.e.* divide by range).**
4. **Rotate it into a local North, East, Up frame (= local [n, e, u] topocentric frame . using the Visual Basic function written for the appendix .**
5. **Convert [n, e, u] coordinates to azimuth and elevation angle.**
6. **Discard data when azimuth angle is below the horizon.**
7. **Plot data on a polar plot.**

Used Equations :

$$\begin{bmatrix} n \\ e \\ u \end{bmatrix} = \begin{bmatrix} -\sin\emptyset\cos\lambda & -\sin\emptyset\sin\lambda & \cos\emptyset \\ -\sin\lambda & \cos\lambda & 0 \\ \cos\emptyset\cos\lambda & \cos\emptyset\sin\lambda & \sin\emptyset \end{bmatrix} * \begin{bmatrix} X & - & X0 \\ Y & - & Y0 \\ Z & - & Z0 \end{bmatrix} \quad (4.1)$$

Where :

[n,e,u] is [north, east ,up] , [*X*,*Y*,*Z*] is the vector to be transformed (in meters) , [X_0,Y_0,Z_0] the position of origin of the NEP system , and λ and $\emptyset$ the longitude and latitude of that origin , respectively.

Horizontal length of unit vector $= \sqrt{n^2 + e^2}$ **(4.2)**

Zenith = $a\tan 2((\sqrt{n^2 + e^2} \ , u) * (\frac{180}{\pi}))$ **(4.3)**

Azimuth = $a\tan2(e, n)$ **(4.4)**

CHAPTER

5 *Velocity and Precise Time (PVT)*

5.1 Introduction

5.2 PVT Solution Mechanization of Navigation Message

5.3 Evaluation Algorithm of LSM

5.4 Kalman Filtering Evaluation Algorithm

CHAPTER FIVE

Velocity and Precise Time (PVT)

5.1 Introduction

GPS, the Global Positioning System, is a satellite systems consisted of 24 satellites with global coverage. This system ensures that four satellites can be observed simultaneously at any time and any point on Earth. It makes the satellite can collect the longitude and latitude and the height of the observation point, in order to achieve the navigation, positioning, time service and other functions. GPS system can provide all-weather, continuous and real-time PVT with high accuracy for global users. In the modern society with the increasing demand for information, GPS is widely applied to military and civil fields because of its ability of all-weather, continuous and realtime supply for three-dimension position, three-dimension velocity and time with high accuracy GPS, the global satellite positioning system, consists of three parts: the space part-GPS constellation; the ground control part - ground monitoring system; the user equipment part-GPS signal receiver. PVT includes the information of position, velocity and time. In the information age, the technology of PVT information processing in GPS receiving system has gotten great development in the military and civil field. Especially, it has become the important supporting system, greatly improving the capability of command and control, the coordination with many arms and rapid response, and accuracy and efficiency of arm equipment in the high technology war. Specifically, there are mainly some following aspects about the GPS receiving system's PVT information processing technology in the military applications: firstly, the full time domain autonomous navigation. The main function of GPS is autonomous navigation, using the GPS receiver's PVT information processing system to provide users with location and time information, and the system can be combined with the electronic map to display the mobile platform track, the route planning and the travel time estimates as well, thus greatly improve the military capability of mobile warfare and rapid reaction. Secondly, the command and control of various combat platforms. With the organic combination of PVT information about navigation and positioning and digital short message communication function, and the special positioning

system, transmit the moving target location information and other relevant information to the command post to complete a moving target dynamic visual display and command instruction issue, thereby achieve the command and control of a moving target in the war zone. Thirdly, the assessment of precision guided and battle damage. The PVT information guidance provided by the GPS receiver with high precision and flexible guided, has become an important guidance system of precision guided weapons. In recent several high-tech local wars, the ratio about the U.S. military using precision guided missiles and bombs has increased of nearly 100 times more Important ways to eliminate random errors caused by the satellite measurements are to optimize the GPS receiving system's PVT information processing technology. The PVT information processing technology refers to calculate GPS receiver system's position, velocity and time with the navigation messages. At present, the technology includes pseudo-range measurement, navigation messages decoding and navigation solvers mainly. Pseudo-range measurement is the basis of the GPS navigation and positioning, the margin of the pseudorange measurement error directly affects the positioning accuracy. The pseudo-range correction parameter and the correct calculation of position resolution parameters need accurate information provided by navigation messages with the correct solver algorithm for processing. At the moment, these three aspects involved in the PVT information processing technology still have problems, need to be further optimized in-depth study and discussion.PVT solution is the calculation of message information in GPS receiving system by some algorithm. LSM and Kalman filtering algorithm is used commonly. In order to satisfy the increasing demand of GPS positioning accuracy, the elimination of random errors in GPS navigation and positioning has become the focus of current research in this field through the use of various solution algorithm of navigation. At present, the general positioning solution methods of PVT information processing technology for GPS receiving system include LSM and Kalman filtering algorithm. It is very difficult for the traditional LSM to eliminate the random errors that affect the positioning accuracy existing in navigation data received by users, but Kalman filtering algorithm applies the optimal estimation theory to models of GPS positioning solution and makes full use of all kinds of statistical information, including the motion characteristics of carriers and statistical characteristics of GPS

measurement, so as to achieve the real-time and best evaluation of the real condition from the random noises for the purpose of eliminating errors. The advantage of Kalman filtering algorithm can be reflected more in dynamic positioning of high dynamic GPS receiver by Kamlan filtering algorithm. Especially, when a GPS receiver filtering prediction equation can not receive GPS signals, the position and velocity can be obtained accurately by Kalman. It is worthwhile to note that the key of Kalman filtering algorithm is the construction of precise dynamic models and noise models, but it's very difficult to get the accurate description of system state in a real system. Especially, it is pretty hard for a high dynamic GPS receiver to determine accurately the characteristics of dynamic noises and observed noises, and so the approximate modes should be used, which causes model errors that influence the characteristics of filtering and even bring about diffusion of filtering. At present, a great deal of study has been made on applying the best evaluation theory to GPS dynamic filtering, but many problems still exist on how to construct motion modes of carriers more reasonably and more accurately, the improvement of dynamic characteristics of filters in order to adjust to high mobility of carriers and simplify filtering models and the improvement of real-time.

5.2 PVT Solution Mechanization of Navigation Message

The solution of user position is to employ the observed distance between a satellite and a user receiver to determine the absolute position of the user receiver in the corresponding coordinate system relative to the origin of geodetic coordinate system. Signals of GPS satellites contain a variety of positioning information, from which different observed quantity can be obtained according to different requirements and methods. The location of PN code is to do correlative calculation by ranging code (C/A code or P code) transmitted by satellites and local PN code of receivers. The transmission time of satellites signals is calculated by measuring the maximum of correlative function, and thus the distance between the satellite and the receiver, which is called correlative measurement. The vector diagram of user position in the system of satellites navigation is given in Figure 1.

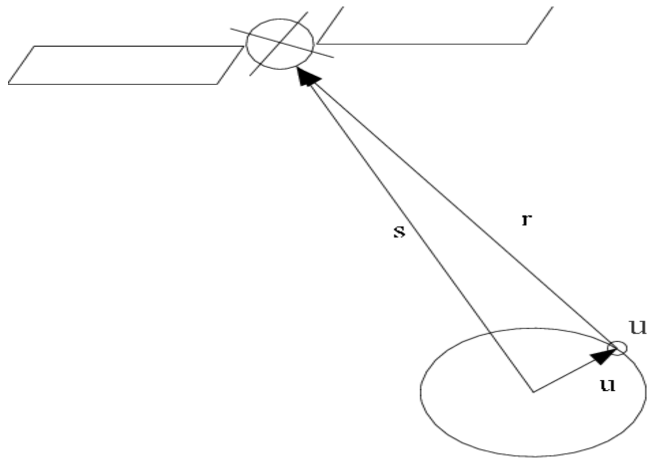


Figure 5.1 The Vector Diagram of User Position .

From Figure 1, it's easy to know that the vector from GPS satellite to a user is r = s – u , the form of its scalar is r = $\|s-u\|$. Let t_u be the divergence between the clock of a satellite and that of a receiver, and the measured pseudo range can be shown as $r_i' = \|s_i - u\| + c\,t_u$. It is easy to know that the location of a user can be calculated by a receiver which receives signals from four satellites at least. Let (x_i , y_i , z_i) how the position of a GPS satellite, let (x_u , y_u , z_u) show the position of a receiver, and the measured pseudo range can be shown as:

$$r_i' = \sqrt{(xi - xu)^2 + (yi - yu)^2 + (zi - zu)^2} + c\,t_u \qquad i = 1,2,3,4 \qquad (5.1)$$

Let (x_u' , y_u' , z_u') show the approximate position of a receiver, demonstrate formula (1) with Taylor series at (x_u' , y_u' , z_u') , formula (2) can be obtained with the first term and the first power.

In formula (2), $r_i' = \sqrt{(xi - xu')^2 + (yi - yu')^2 + (zi - zu')^2} \quad i = 1,2,3,4 \quad (5.2)$ **approximate pseudo range.**

$$\begin{cases} \Delta xu = xu - xu' \\ \Delta yu = yu - yu' \\ \Delta zu = zu - zu' \end{cases} \qquad (5.3)$$

Formula (3) shows the difference of coordinates. It is worthwhile to note that the

ignorance of some date leads to the poor accuracy of calculation in the process of linear. Meanwhile, in the above discussion, the causes of the poor accuracy also lie in the ignorance of noise in the process of measurement, the changes of the speed of light in the process of transmission and the effect of the theory of relativity.

Δr = H * Δu

In formula (4), $\Delta r = \begin{bmatrix} \Delta r1 \\ \Delta r2 \\ \Delta r3 \\ \Delta r4 \end{bmatrix}$, $H = \begin{bmatrix} ax1 & ay1 & az1 & 1 \\ ax2 & ay2 & az2 & 1 \\ ax3 & ay3 & az3 & 1 \\ ax4 & ay4 & az4 & 1 \end{bmatrix}$, $\Delta u = \begin{bmatrix} \Delta xu \\ \Delta yu \\ \Delta zu \\ -c\,\Delta tu \end{bmatrix}$ **(5.4)**

So, the solution of formula (4) is formula (5). It contains the coordinate of a user and the offset value of the clock of a receiver .

$\Delta u = H^{-1} * \Delta r$ **(5.5)**

5.3 Evaluation Algorithm of LSM

Evaluation algorithm of LSM is the best filtering for a series of giving data, which is utilized for the solution of navigation receivers with low speed. Formula (5) can be obtained by formula (4). The above-mentioned process is calculated again if the accuracy does not meet the requirement, and usually the realization of the accuracy can be gained with three-time or four – time calculation. Formula (6), the solution by LSM, is deduced by formula (4) with the help of evaluation algorithm of LSM when the number of observed satellites is bigger than 4.

$\Delta u_{LS} = (H^T H)^{-1} * H^T * \Delta r$ **(5.6)**

The solution procedure of evaluation algorithm of LSM is illustrated by

Figure 2.

Step 1: Initialization of iterative increment Δu_{LS} and the value of PVT of a user .
Step 2: Before the maximum iteration, calculating iterative increment Δu_{LS} , and substituting the result Δu_{LS} for the original *u* to the value of PVT of a user.

Step 3: Repeating step 2 until the iterative increment Δu_{LS} reaches the convergence threshold or the number of times of calculation is equal to the maximum iteration.

Step 4: The value of PVT of a user *u* is equal to the solved value of use's location only when the iterative increment Δu_{LS} is smaller than the convergence threshold. If the iterative Increment Δu_{LS} is not smaller than the convergence threshold with the maximum iteration, the conclusion is made that the solution can't achieve the expected accuracy and the initialization of iterative increment should be made again.

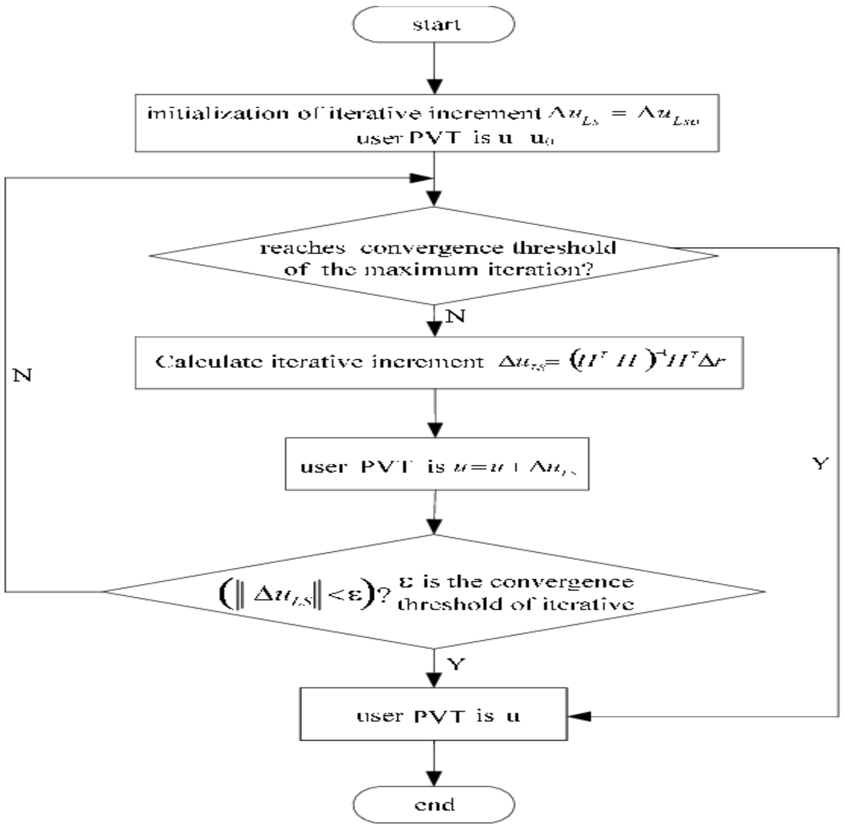


Figure 5.2 The Solution Procedure of Evaluation Algorithm of LSM .

5.4 Kalman Filtering Evaluation Algorithm

The theory of the Kalman Filtering is a modern filter theory proposed in 1960 by Kalman. The Kalman filtering is a time domain method which can get the recursive least-meanvariance estimation of the system's state for the linear systems with Gaussian noise distribution. The Kalman filtering introduces the state space thought of modern control theory into the optimal filtering theory for the first time, describe the system dynamic model with the equation of state and the system observation model with equation of observation, and can handle timevarying systems, non-stationary signals and multi-dimensional signal. Due to the Kalman filtering using recursive calculation, it can be achieved by computer suitably. The Kalman filtering method is calculated on the basic of the least-mean-square-error, a extension of the Wiener filtering method. The Kalman filtering method should establish the state vector model of the applicable system in the first, and then through the recursive process seek the state vector's best estimation in the least-mean-square-error. It is difficult for the traditional evaluation algorithm of LSM to reduce random errors of navigation data which influence location accuracy. The purpose of processing GPS navigation data lies in the reduction of effects which the errors make on the result of dynamic location as far as possible. One of the most important means is Kalman filtering. The theory of Kalman filtering, as the best real-time recursion algorithm, adopts state-space model of signals and noises, and take advantage of the previous estimate and the present observed value to renew the evaluation of state variable and get the estimate for the following moment which is suitable for real-time processing and computer operation The primary task of solution is the study of the following dynamic models in the system. State equation of the studied system:

$$x(n+1) = \varphi\,(n+1\,,n)\,x(n)+v_1\,(n) \tag{5.7}$$

Measuring equation of the studied system:

$$z(n) = H(n)x(n) + v_2\,(n) \tag{5.8}$$

$x(n)$ is M-dimension state vector of the system, to which the purpose of Kalman filtering is the best evaluation. $\varphi(n+1, n)$ is M*M-dimension state transition matrix. $v_1(n)$ is M-dimension noise vector of the system. $z(n)$ is N-dimension measurement vector. $H(n)$ is N*N-dimension measurement vector. $v_2(n)$ is N-dimension measurement noise vector. $v_1(n)$ and $v_2(n)$ is white noise vector, and has the following properties.

$$E\{v1(n)\} = 0 \tag{5.9}$$

$$E\{v2(n)\} = 0 \tag{5.10}$$

$$E\{v1(n)v_1^T(K)\} = \begin{cases} Q & n = K \\ 0 & n \neq K \end{cases} \tag{5.11}$$

It is known from formula (4) that the pseudo variable Δr is correspondent to the measurement vector $z(n)$ of the Kalman filtering model; H matrix is correspondent to the measurement matrix $H(n)$ of the Kalman filtering model; the value of position difference of user's receiver Δu is correspondent to the system state vector $x(n)$ is correspondent to the system noise vector $v_1(n)$ of the Kalman filtering model; the value of pseudo difference Δr is correspondent to the measurement noise vector $v_2(n)$ of Kalman filtering model. The Kalman filtering model of solution to the position of a user's receiver is then set up successfully. After the construction of the Kalman filtering model, the state vector of the system (the value of position difference of a user's receiver) is calculated according to the solution process of Kalman filtering algorithm on the basis of the input measurement vector (observed pseudo received by a user's receiver) and initial condition (the value of position difference estimated by a user's receiver initially), and the present location of a user's receiver can be obtained according to the value of position difference of a user's receiver and the previous position of a receiver. The solution procedure of evaluation algorithm of Kalman filtering is illustrated by Figure 3 .

Step 1: A user's receiver calculates the initial conditions $\hat{X}(1,0)$ and P(1,0) according to the estimated value of position difference at initial time.

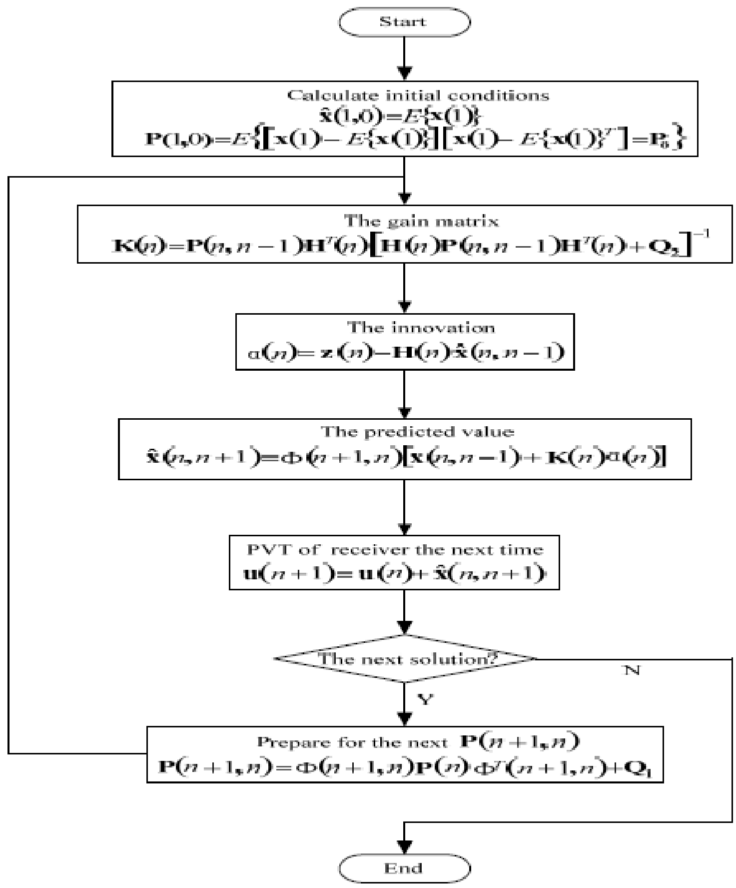
Step 2: Calculating the gain matrix $K(n)$, and calculating the innovation $\propto (n)$ according to the new observed pseudo received by a use's receiver.

Step 3: Renewing the predicted value $\hat{X}\ (n, n+1)$.

Step 4: Obtaining the PVT value $\hat{X}\ (n, n+1)$ of a user's receiver at the following moment according to the predicted value $\hat{X}\ (n, n+1)$.

Step 5: If the next solution is not needed, escape directly, if not, go back to calculate P $(n, n+1)$. So as to calculate the gain matrix K(n) of this time, the innovation $\propto (n)$ and the predicted value $\hat{X}\ (n, n+1)$ of this time, and complete the calculation of u$(n+1)$.

Figure 5.3 Procedure based on Kalman Filtering Evaluation Algorithm.



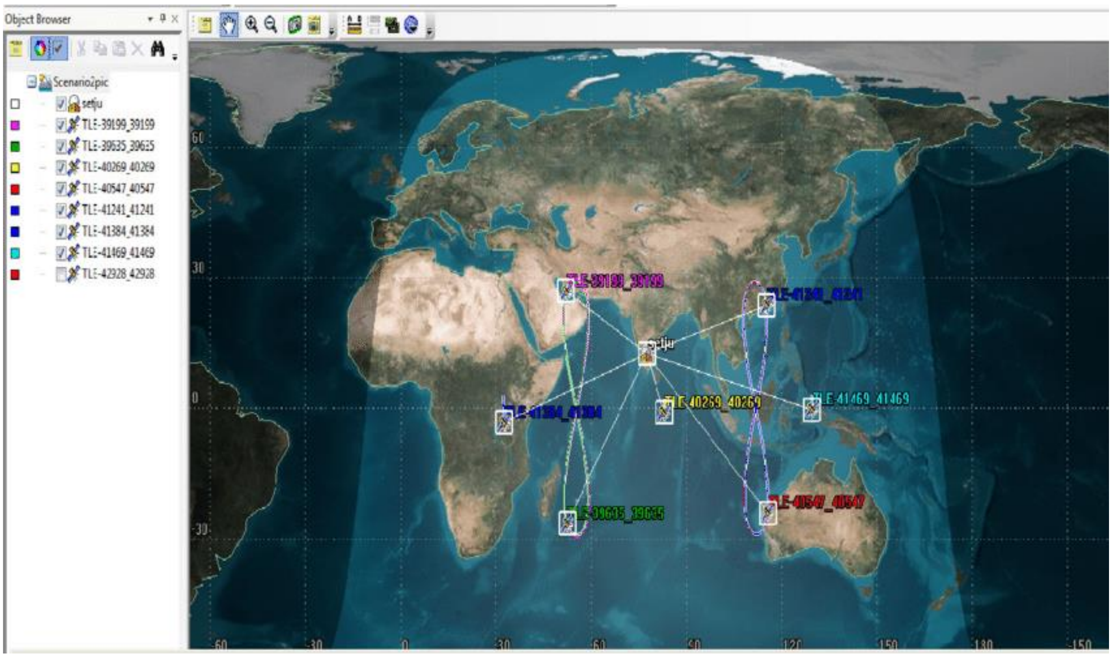


Figure 5.4 Path of Geo synchronous and Geo stationary satellites.

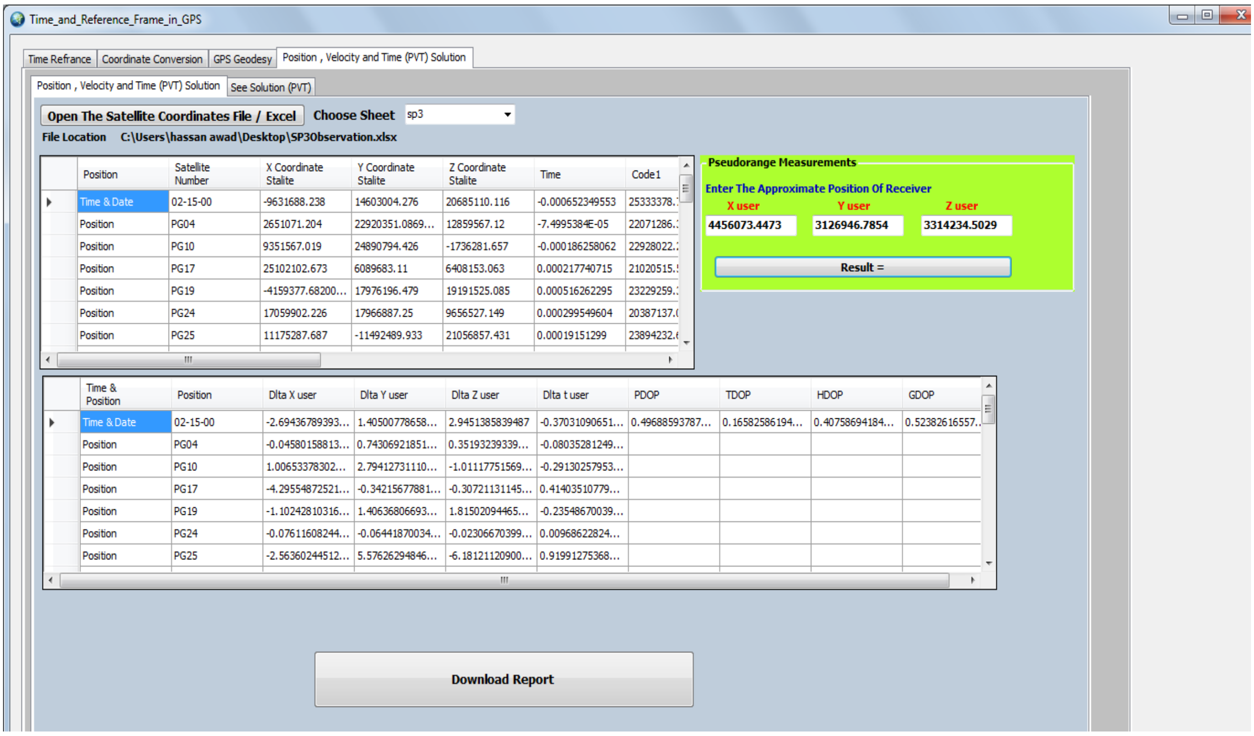


Figure 5.5 Example For PVT Solution .

<u>Note</u> : DOP For all monitoring period .

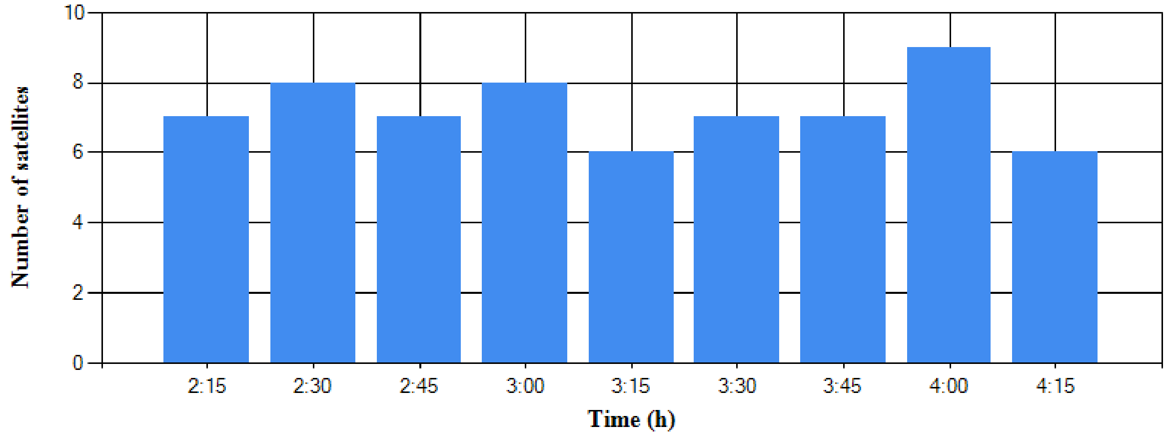


Figure 5.6 Satellites number.

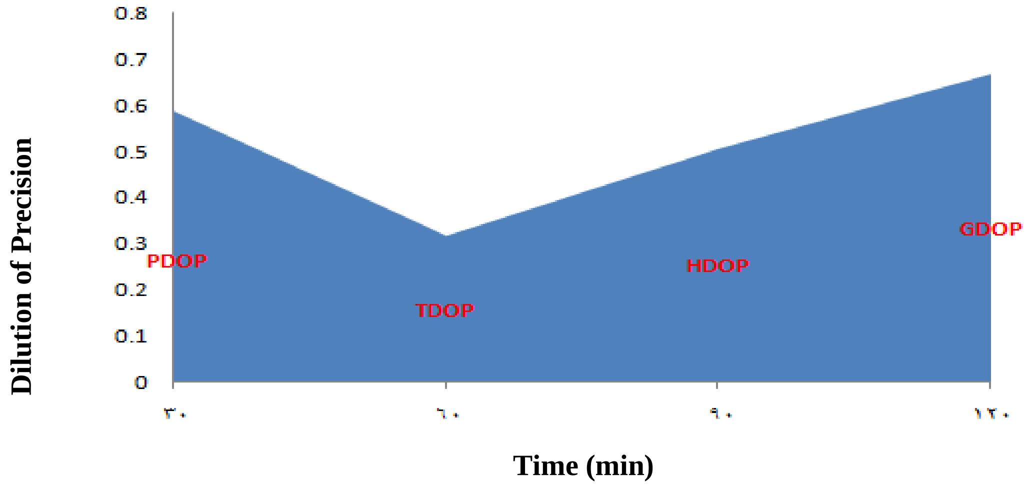


Figure 5.7 Dilution of Precision .

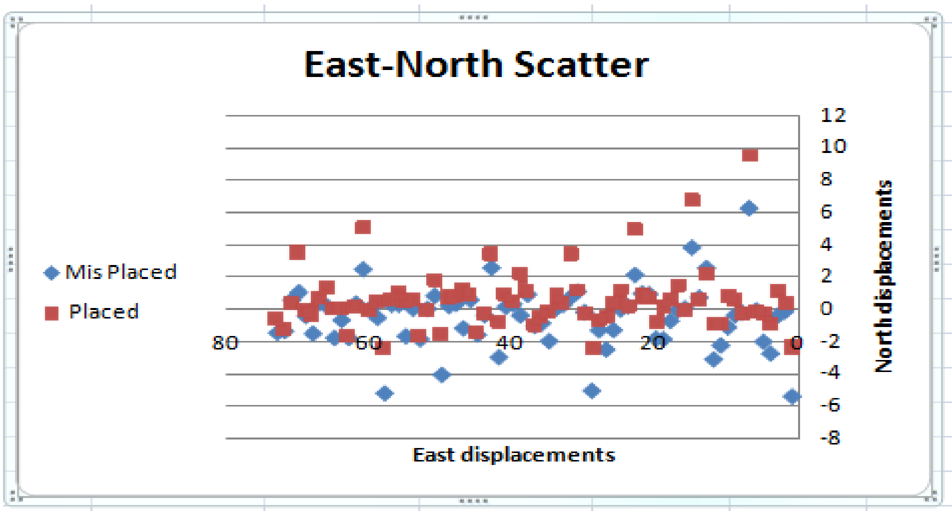


Figure 5.8 East - North Scatter .

CHAPTER

6 *Study Area*

6.1 Study area

CHAPTER SIX

Study Area

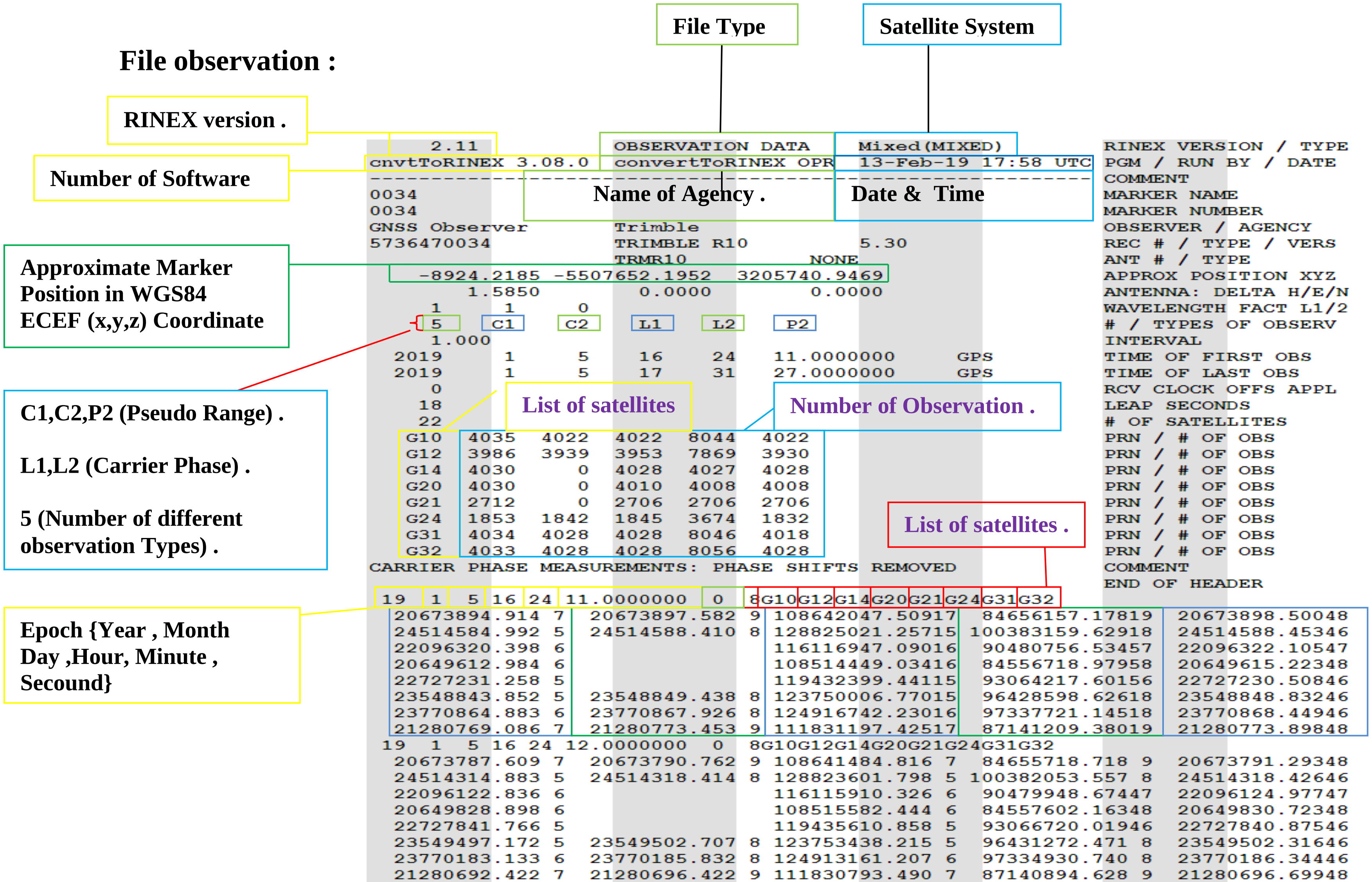
6.1 Study area

A reading file was taken of an observed point in Singer area in Dura city on 17/3/2022 and only two hours worked from 2:15:00 until 4:15:00 according to SP3 file , and using GPS satellites only .Convert user coordinate from Cartesian to Palestinian coordinate with used program in my project GPS Geodesy .

Table 6.1- Convert user coordinate from Cartesian to Palestinian .

The axis	Cartesian	Palestinian
X =	4456073.4473	35.05837222 E
Y =	3126946.7854	31.50488056 N

File observation :



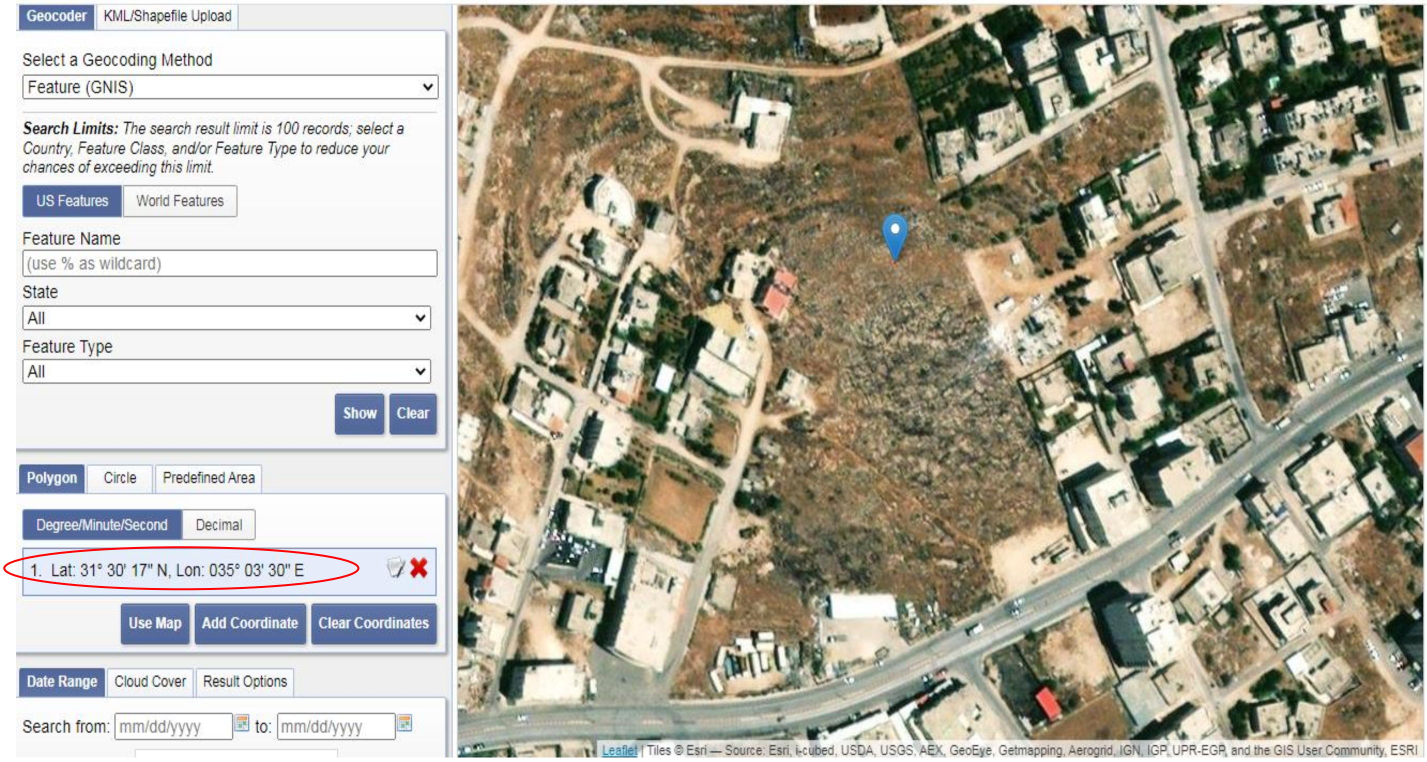


Figure 6.1 Convert user coordinate from Cartesian to Palestinian in my project .

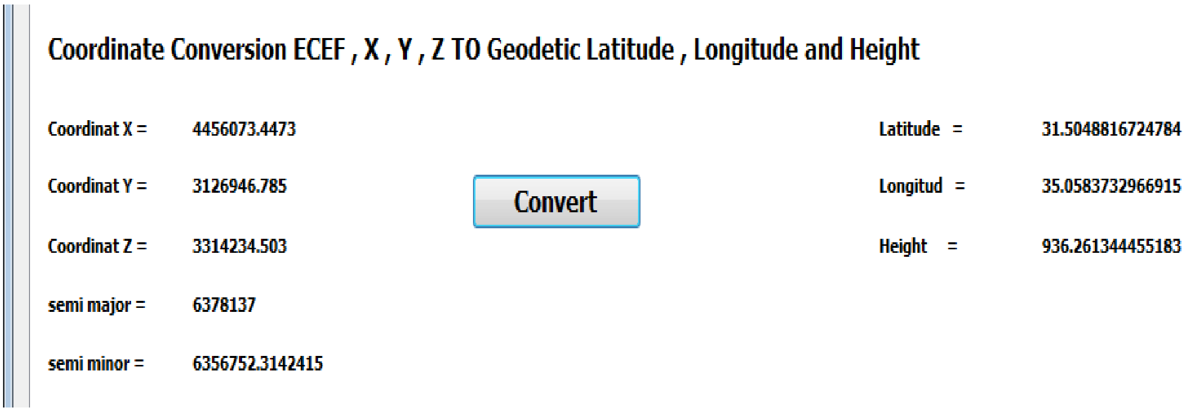


Figure 6.2 Use website USGS to determine use location .

CHAPTER 7

Software Solution For GPS Calculations Tutorials

7.1 Introduction

7.2 Time And Reference Frame

7.3 Coordinate Conversion

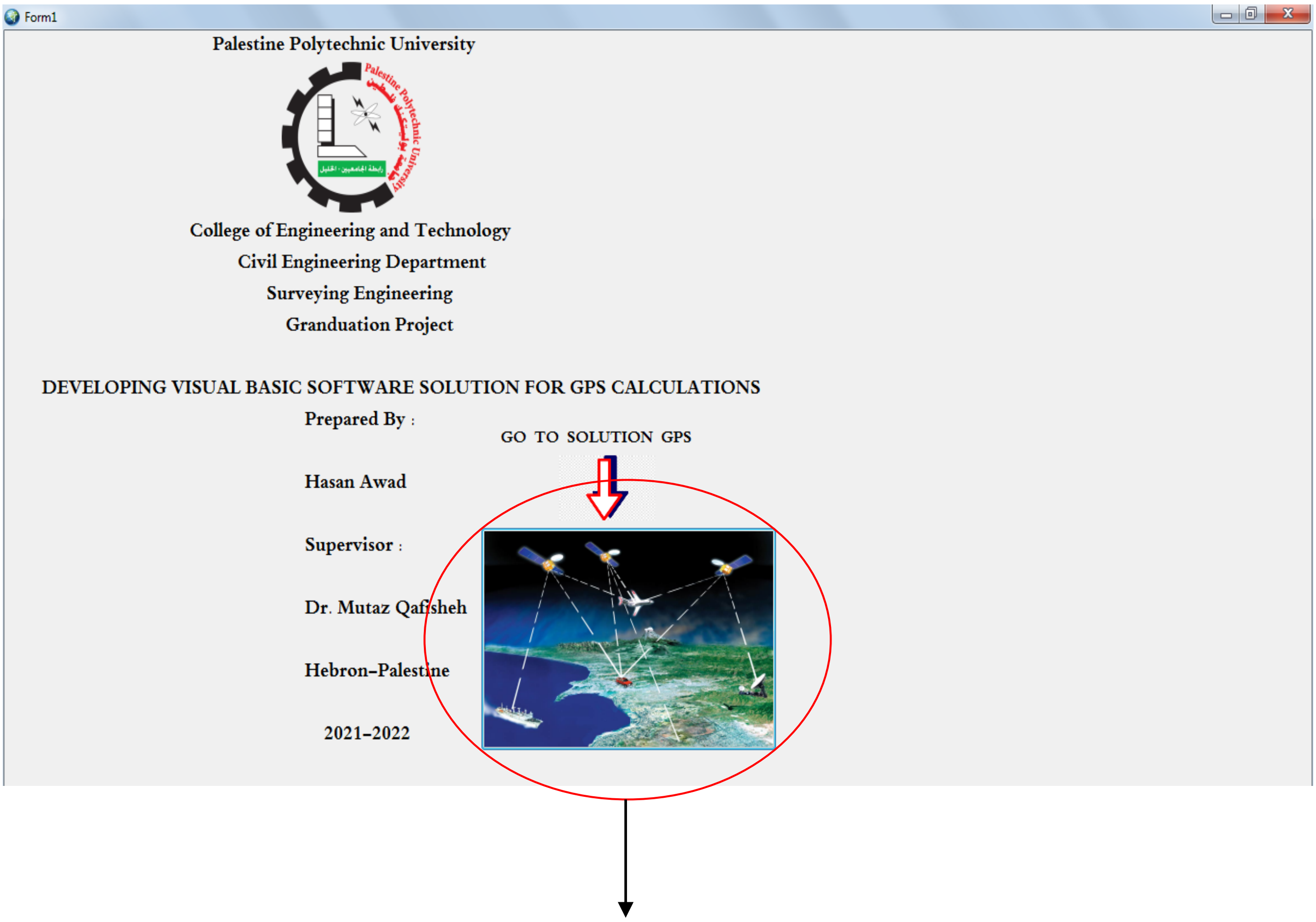
7.4 GPS Geodesy

7.5 Velocity And Precise Time (PVT)

CHAPTER SEVEN

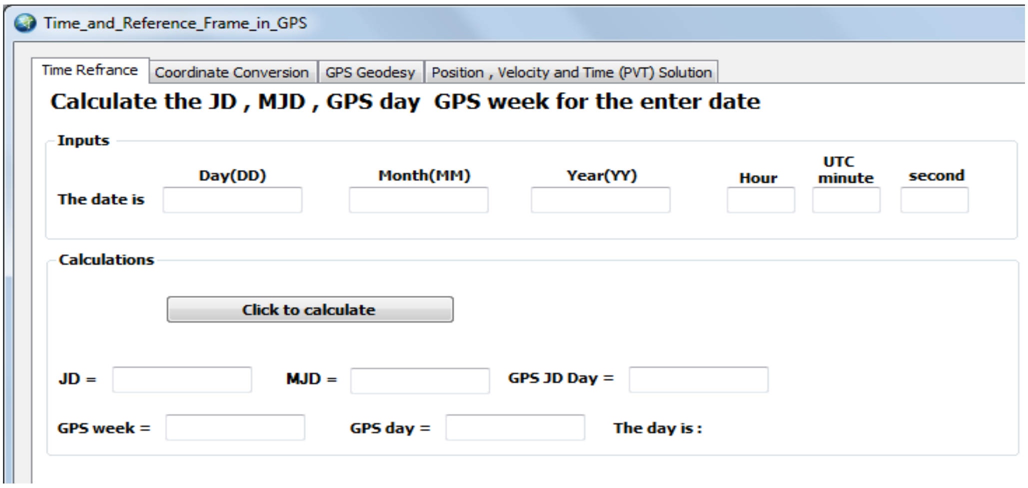
Software Solution For GPS Calculations Tutorials

7.1 Introduction



Click here to enter the program .

7.2 Time And Reference Frame



The user enters the date with the day, month, year and time in hours, minutes and seconds, then the program calculates (Julian day (JD) , Modified Julian day (MJD), GPS (JD) day , GPS week , GPS day and The day) .

Here it should be noted that the date of epoch GPS is (6/1/1980) and that any date before this the program does not calculate it. And the user must use logical values, meaning that there is no year more than 13 months, no month has more than 31 days, and February does not have more than 29 days .

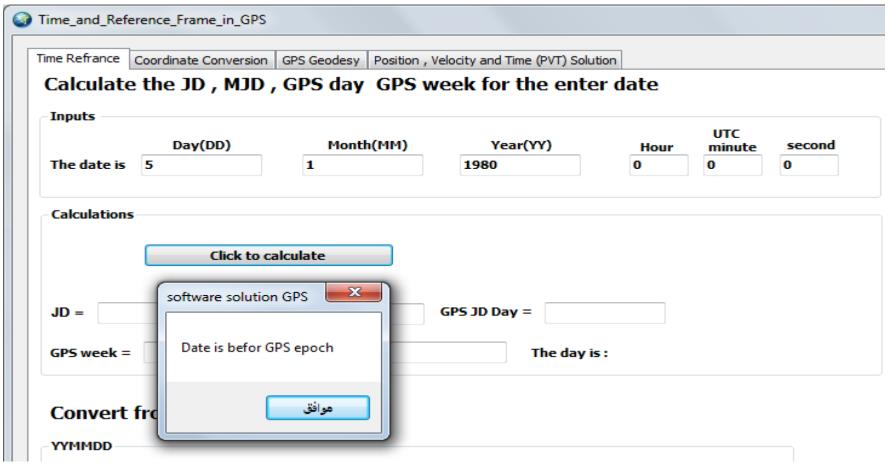


Figure 7.1 Message appear when enter data before GPS epoch date .

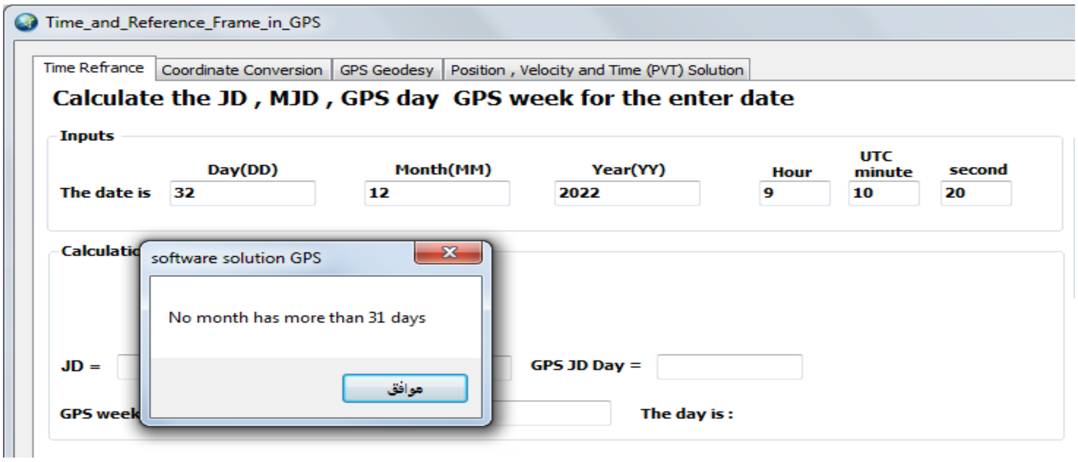


Figure 7.2 Message appear when 32 days in a month are entered .

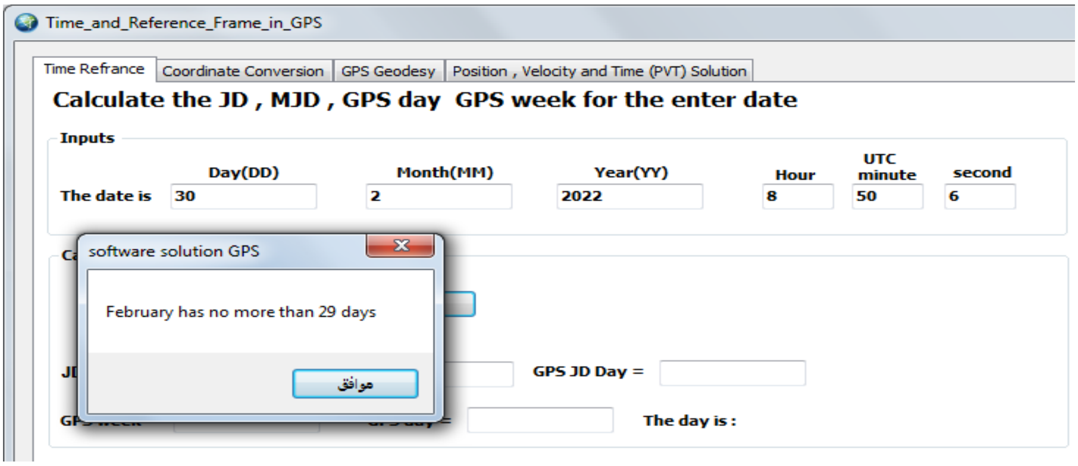
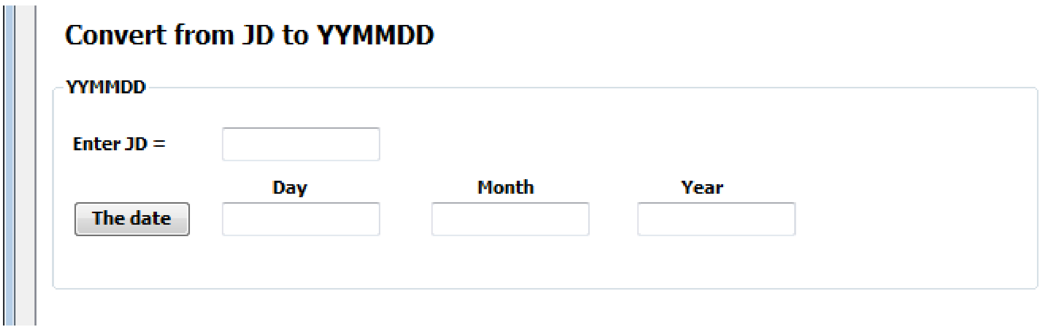
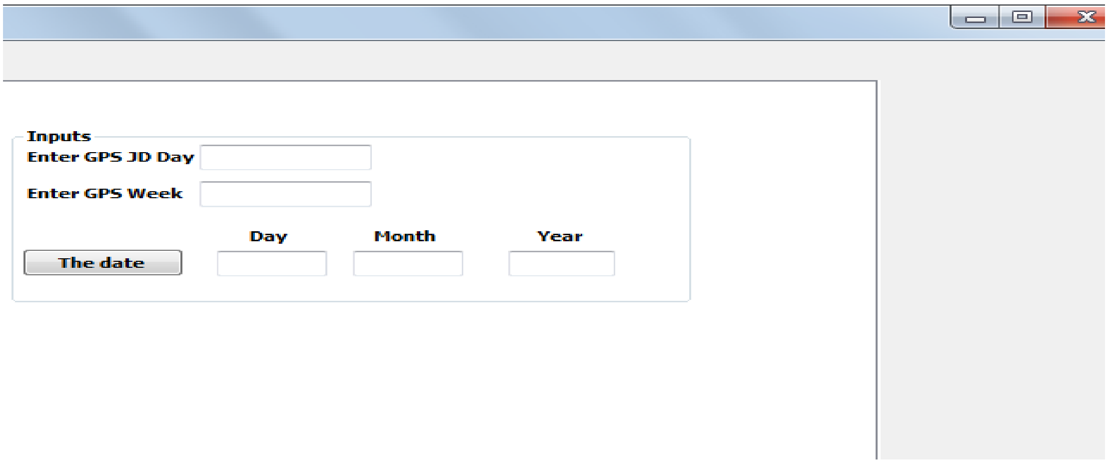


Figure 7.3 Message appear when 30 days in a February are entered .



In this part , the user enters Julian day and the program calculates the date with the day, month and year , reverse process .



In this last part the user enters (GPS JD Day and GPS week) then the program calculated the date with day , month , year .

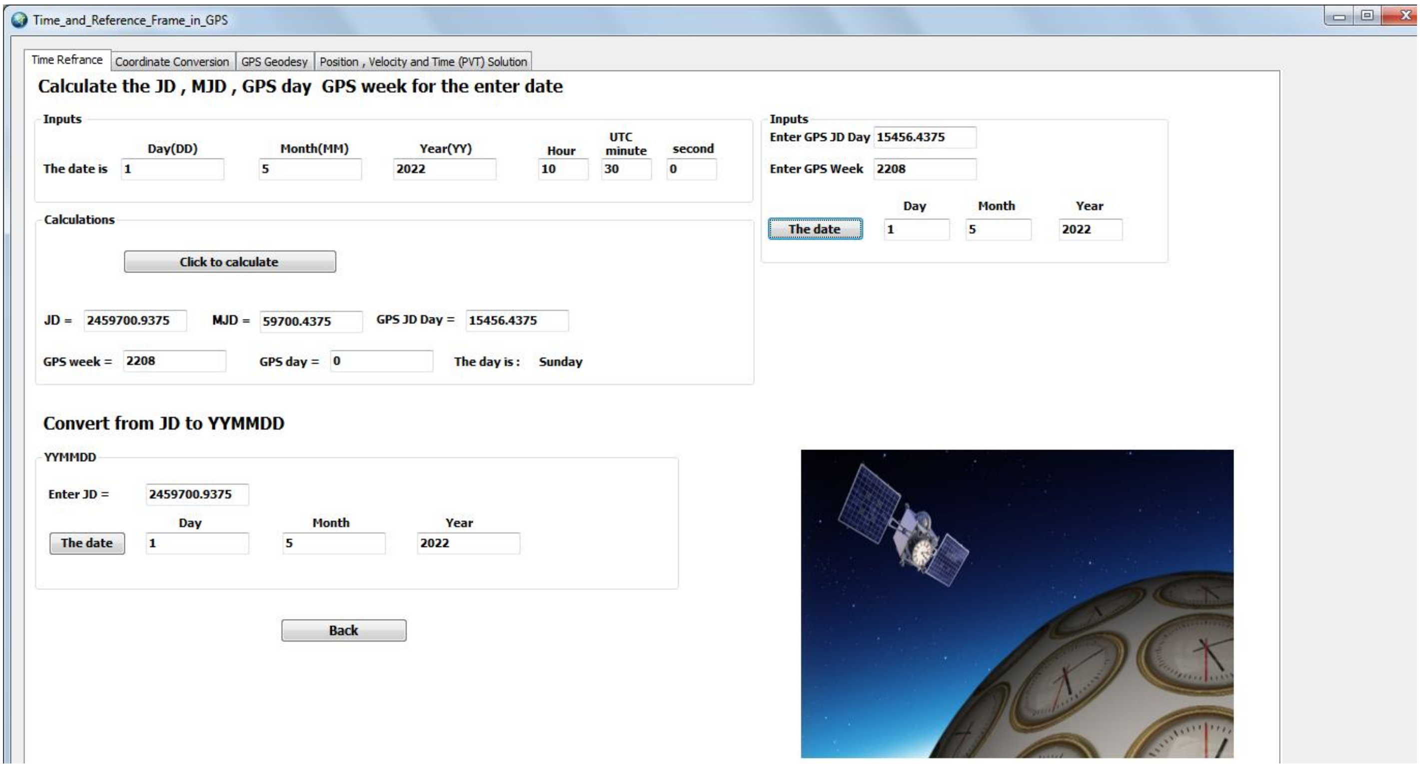
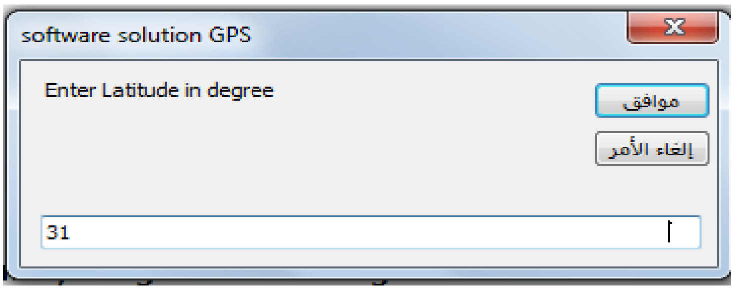


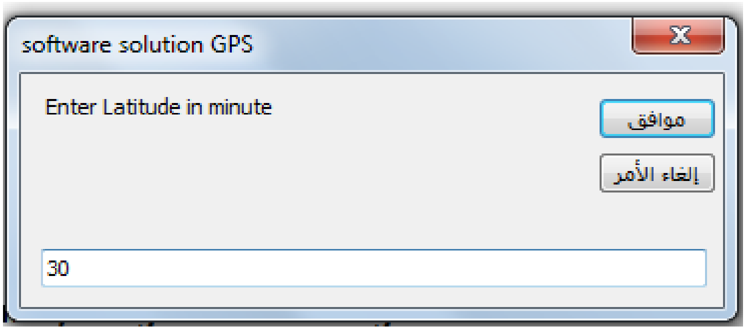
Figure 7.4 Example for Time And Reference Frame .

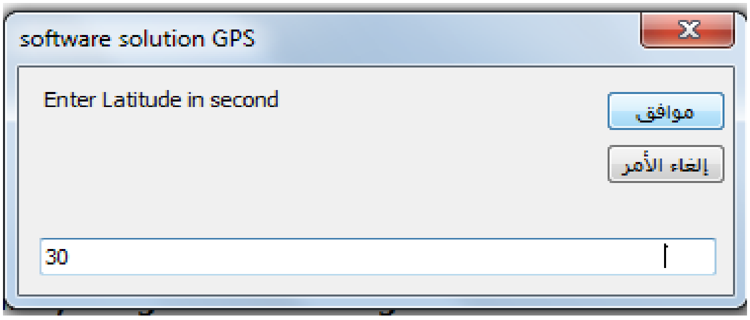
<u>Note:</u> There is a problem when use the 12 o'clock, it is better not to use it.

7.3 Coordinate Conversion

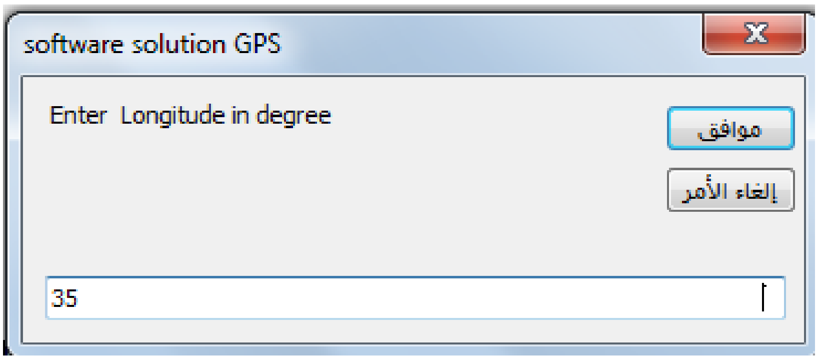
In the first section of the program, the user enters the angles of latitude and longitude in degrees, minutes and seconds, the semi major axis , the semi minor axis , and the height . then the program calculate X – Coordinate , Y – Coordinate , Z – Coordinate .

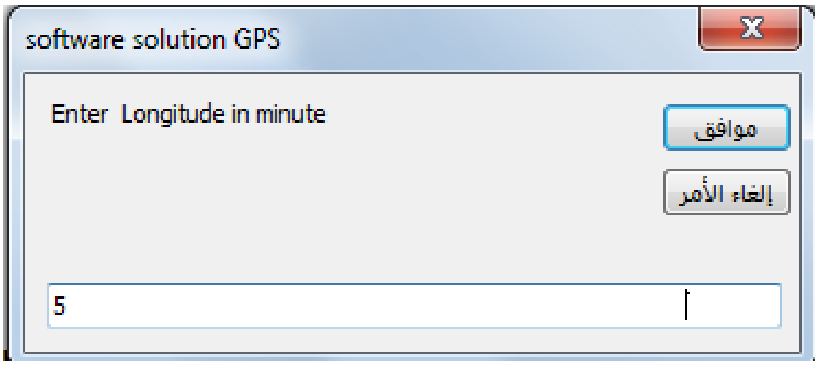


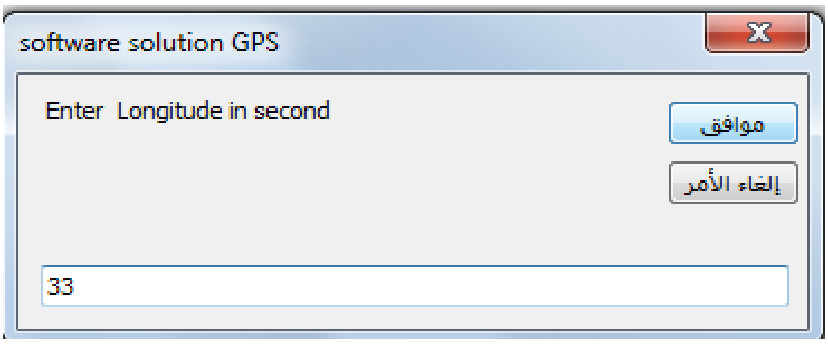




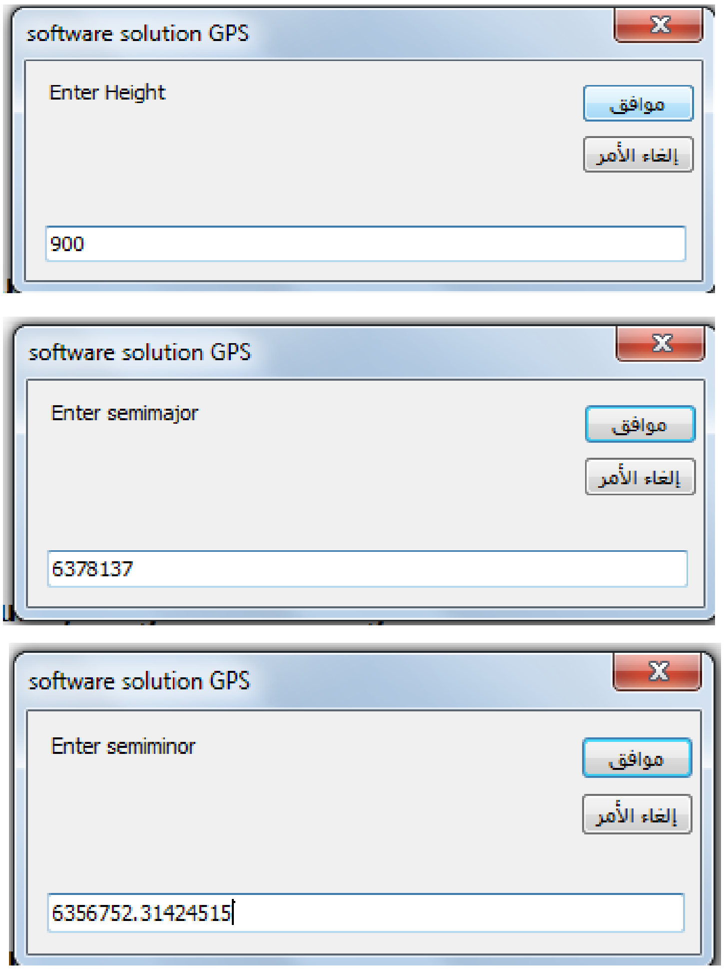
Enter the angle of latitude .







Enter the angle of latitude .



Enter the , the semi major axis , the semi minor axis , and the height.

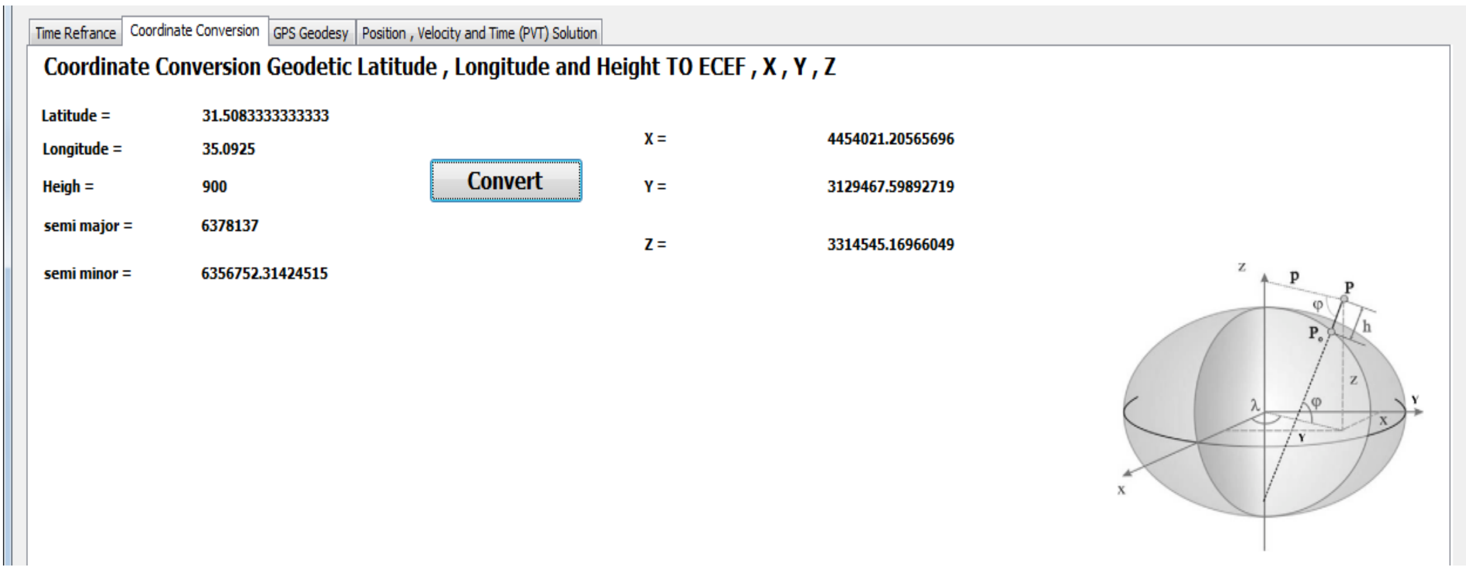
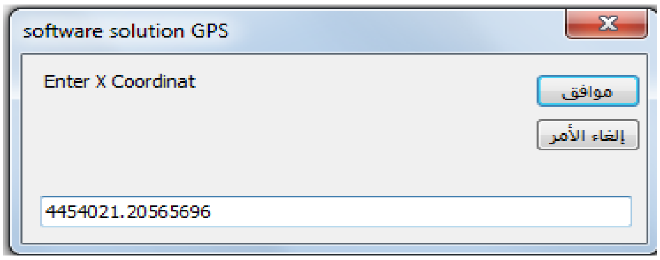
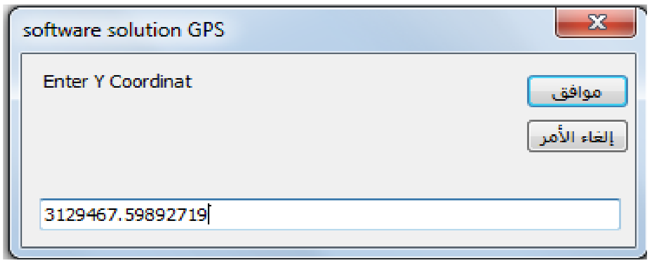
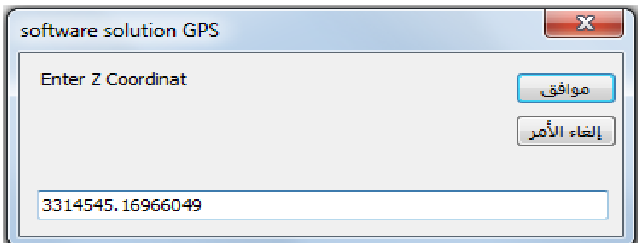


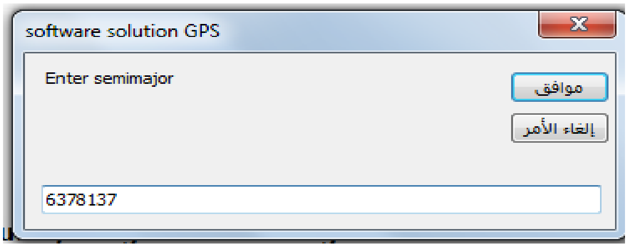
Figure 7.5 Example for Coordinate Conversion.

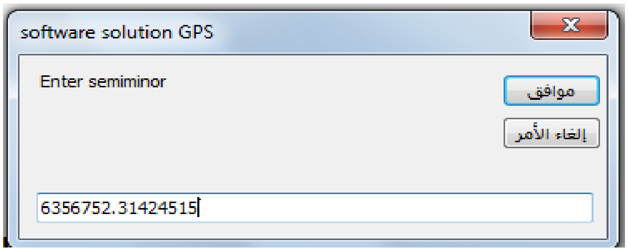
In the secound section of the program, the user enters X – Coordinate , Y – Coordinate , Z – Coordinate , semi major axis , semi minor axis then the program calculate the angles of latitude and longitude and the height .











Enter the , X – Coordinate , Y – Coordinate , Z – Coordinate semi major axis , semi minor axis.

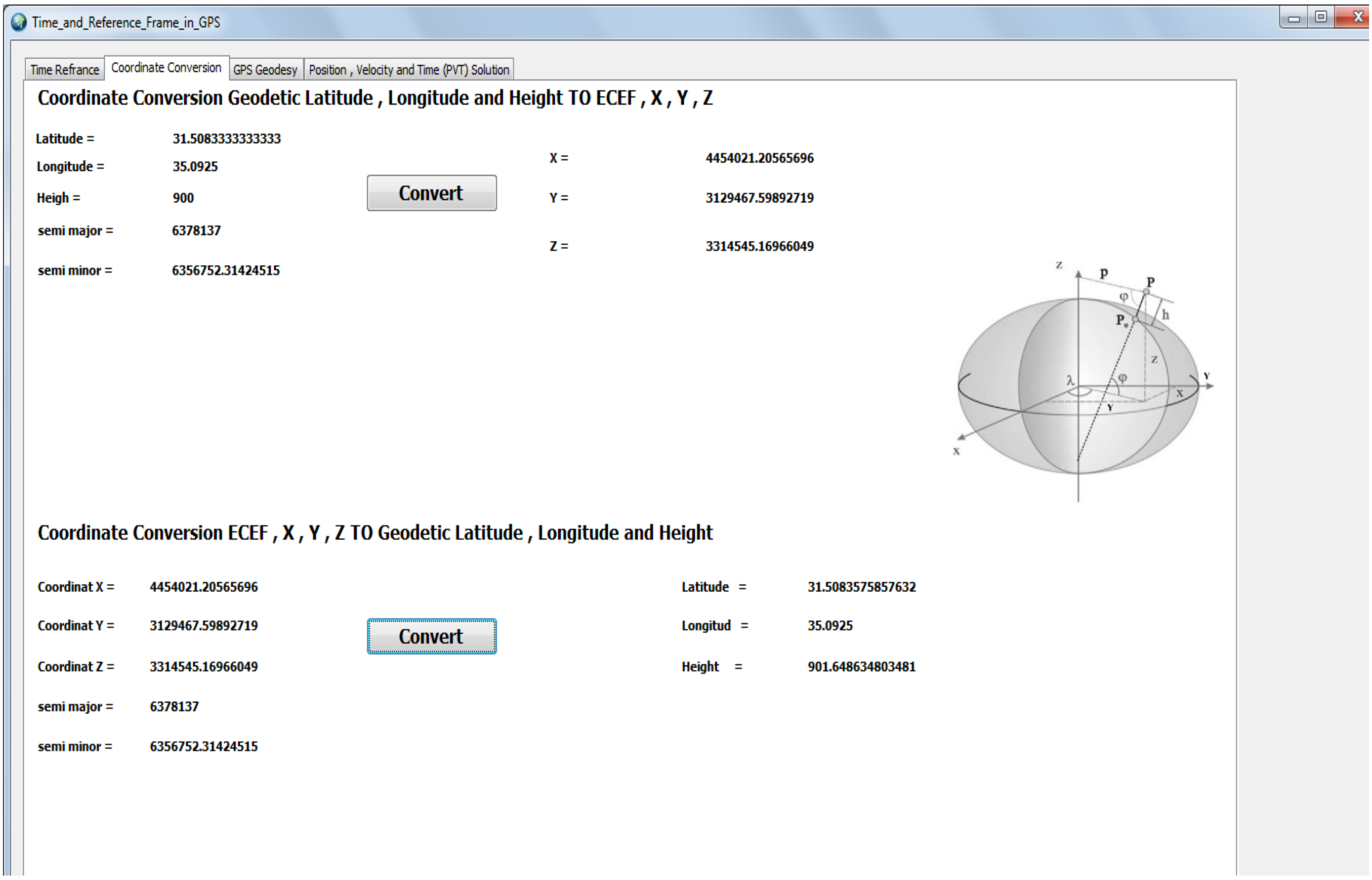
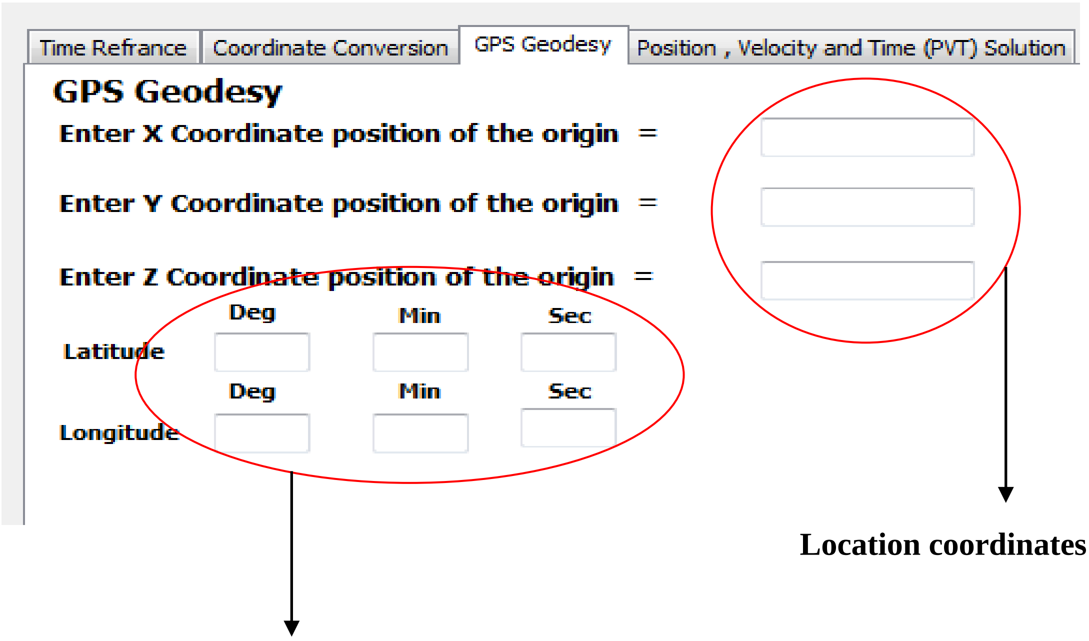


Figure 7.6 A complete example of coordinate conversion.

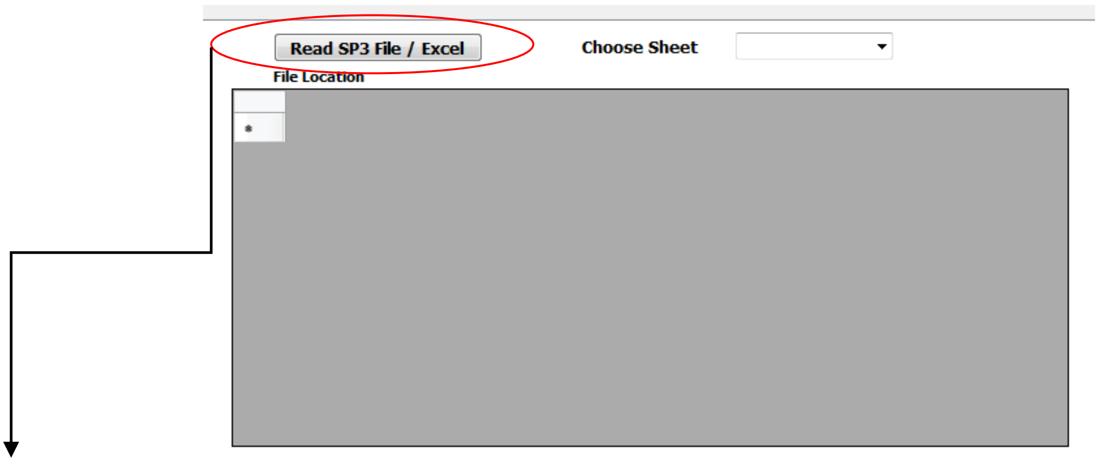
7.4 GPS Geodesy

The user enters the location coordinates, longitude and latitude angles, and enters the monitoring file in the form of an excel file, and the program calculates North , East , Up , Horizantal length of unit vector , Zenith and Azimuth .

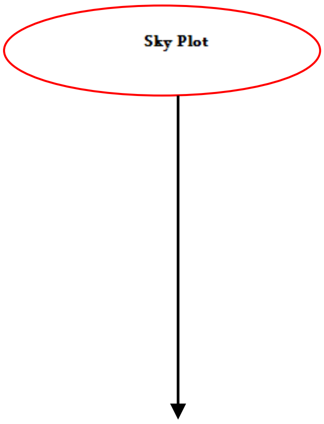


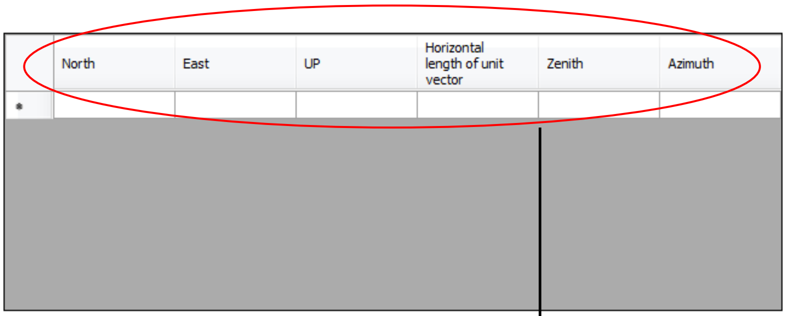
Location coordinates .

Longitude and latitude angles .



The monitoring file in the form of an excel .





Result of draw sky plot .

Result of North , East , Up , Horizantal length of unit vector , Zenith and Azimuth .

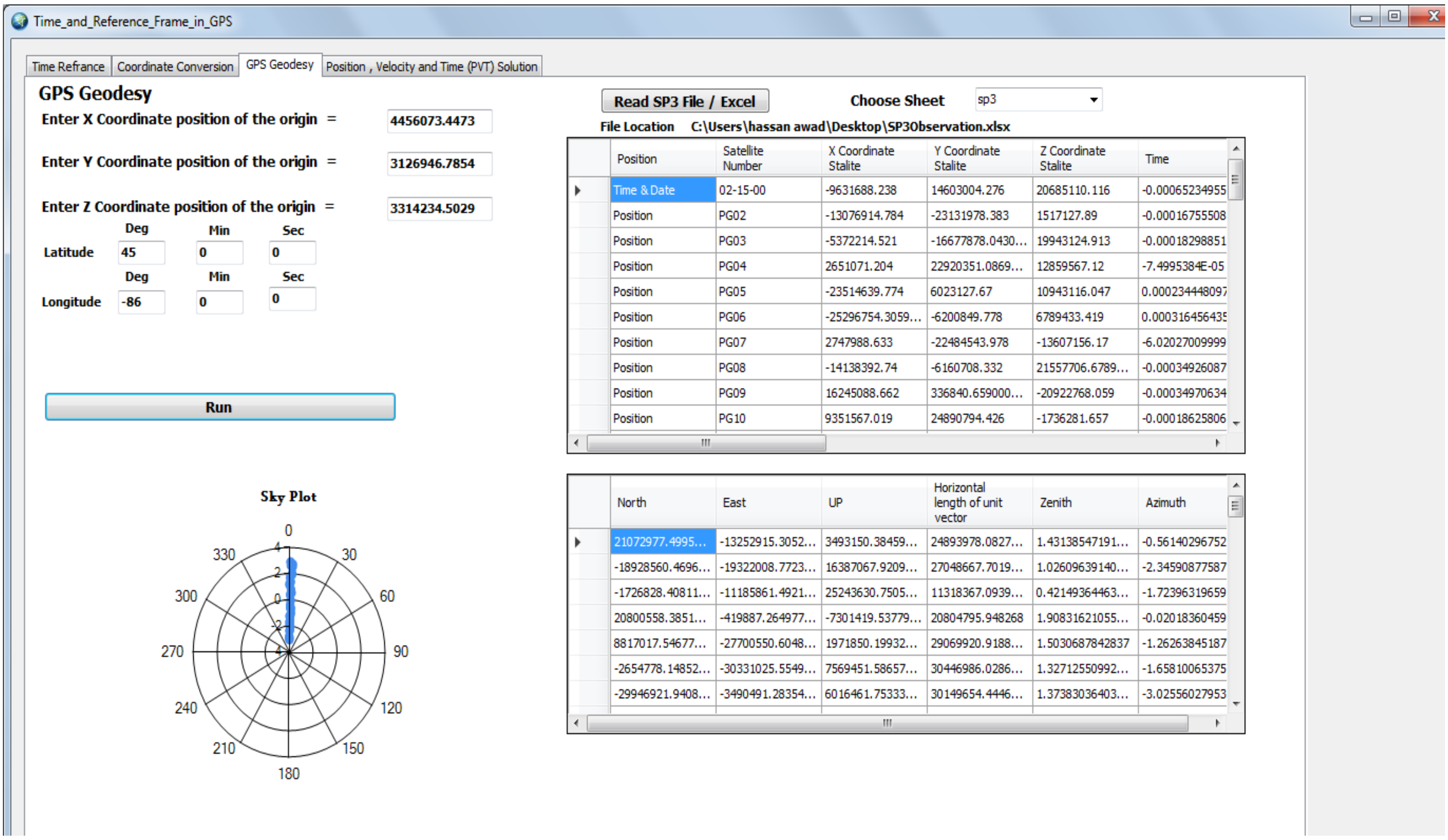
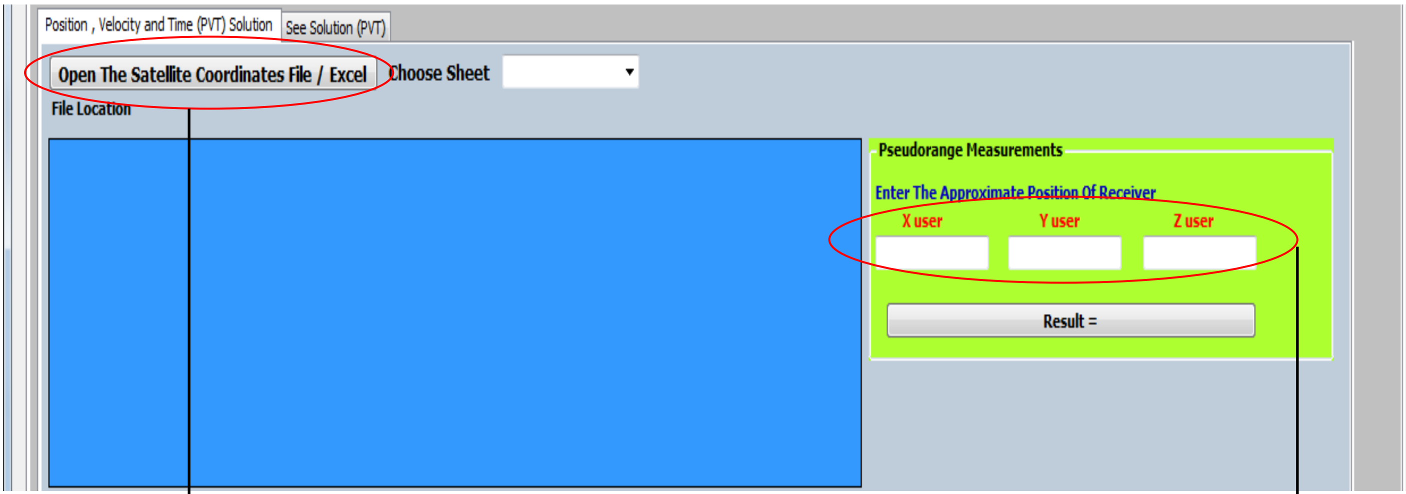


Figure 7.7 A complete example of GPS Geodesy.

7.5 Velocity And Precise Time (PVT)

In this section of the program the user enters the approximate location coordinates and the monitoring file in excel format, and the program calculates the accuracy of each satellite used in the monitoring. There is a report summarizing the accuracy of each satellite, and the program does not include the satellite whose accuracy exceeds 100 meters in the solution, but rather deletes it automatically.



The monitoring file in the form of an excel .

Approximate location coordinates .

A report summarizing the accuracy of the satellite and DOP.

Result of Time , Position , Dlta X user , Dlta Y user , Dlta Z user , Dlta t user and DOP .

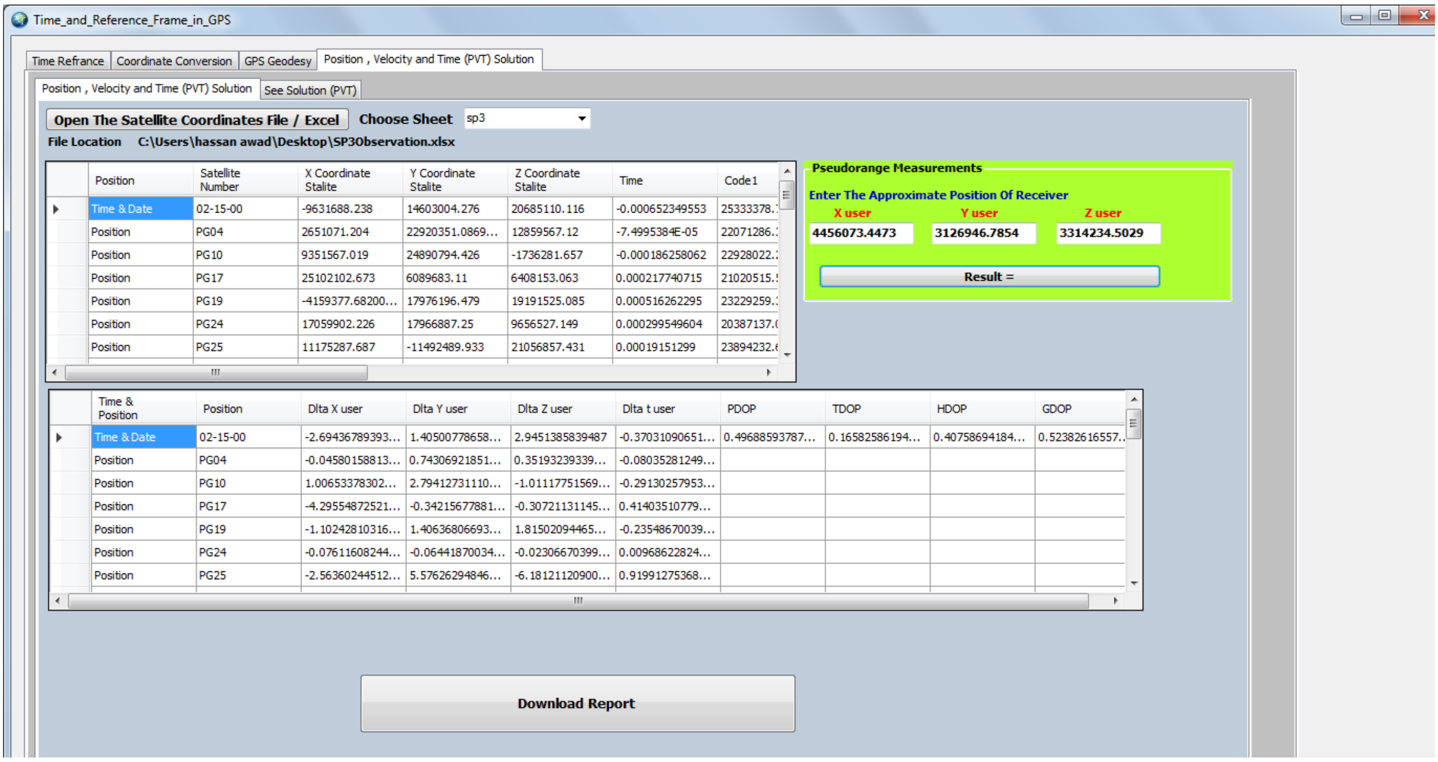


Figure 7.8 A complete example of Velocity And Precise Time (PVT).

Time	Satellite Number	Dlta X user	Dlta Y user	Dlta Z user	Dlta t user	PDOP	TDOP	HDOP	GDOP
Position	PG05	-5.4363846411293917	-2.2601466370401475	-1.1804683109617855	-1.1995756534075419	0.588188061528589	0.3181113		
Position	PG12	-0.14482584482178273	0.416437374571174	0.15255574057830512	-0.012498060017831787				
Position	PG18	-0.41840059962185538	1.1725069770593244	-0.90663467277134968	-0.16500917439532722				
Position	PG20	-2.7769051453950881	-0.842813774269124	-0.90584671326692945	-0.29856530090011035				
Position	PG25	-2.0392127749135232	-0.19884471251382951	0.017947977031116835	-0.40881291915982426				
Position	PG26	-0.084044880762109031	-0.0824604345141971	-0.051818418413751219	-0.017287661442158375				
Position	PG29	6.2475266204528133	9.501305390366678	1.7152340355488382	2.2700111423614326				
Time & Date	02-30-00	-0.23291911684254341	-0.16474957447882666	-0.36649311875510993	-0.0431798297524883				
Position	PG12	-0.39558325550709794	0.68835073209807307	0.35237523452129915	-0.046630957357351253				
Position	PG18	-1.1384087027741767	0.85238393810723423	-1.7264561175235036	-0.37920578814069505				
Position	PG20	-2.2666091430975559	-0.83272268080318712	-1.0317600633896933	-0.2942876380273054				
Position	PG23	-3.1303212176765167	-0.7886895612361875	-0.2059672184031669	-0.65575896922920784				
Position	PG25	2.5320166097374486	2.2407845876454431	5.5575713780999614	1.3027831617205021				
Position	PG26	0.69389951476527467	0.71220538002852229	0.35879913119935014	0.13700738803614107				
Position	PG29	3.8005743427985124	6.72935348330774	0.49155035356500232	1.535298499928379				
Position	PG31	0.039603606362654781	0.03757813041760788	0.067829511312842317	0.00825265352279436				

Figure 7.9 Contents of the report .

Resources and References

1. **GNSS Data Processing, *Vol. I: Fundamentals and Algorithms* (ESA TM-23/1, May 2013).**

2. **Springer International Publishing Switzerland 2016 Erik Grafarend .**

3. **Nevada Bureau of Mines and Geology/Seismological Laboratory, University of Nevada, Mail Stop 178, Reno, NV 89557, USA.**

4. **Dodson, A.H., P.J. Shardlow, L.C.M. Hubbard, G. Elgered and P.O.J. Jarlemark, 1996, Wet Tropospheric Effects on Relative GPS Height Determination, Journal of Geodesy, Vol. 70, pp 188-202.**

5. **Adam, J. (1991) edt., Proceedings of the International Seminar- GPS in Central Europe, Penc, Hungary.**

6. **Hoffman-Wellenhof, B, et. Al. (1997), GPS Theory and Practice, 4th Rev. Edition, Springer Wien, New York.**

7. **G.L. and J.R. MacKay, 1996, Calibration of GPS Antennas, IGS Workshop Proceedings, Silver Spring, Md. March 19-21, 1996. available from Jet Propulsion Laboratory, Pasadena, California. JPL Publication 96-23 10/96. pp.81-105.**

8. **Enrico Del Re, Marina Ruggieri. Satellite Communications and Navigation Systems. NewYork: Springer. 2008.**

9. **DJ Dailey, BM Bell. A Method for GPS Positioning. *IEEE Transactions on Aerospace and Electronic Systems*. 1996; 32(3): 1148-1154.**

10. **John B Moore. *Direct Kalman Filtering Approach for GPS/INS integration.* IEEE-IEE Vehicle Navigation & Information System Conference. 2002.**

APPENDIX

APPENDIX– (A)

APPENDIX– (B)

APPENDIX– (C)

APPENDIX– (D)

APPENDIX– (A)

Time And Reference Frame

Time And Reference Frame

```
Imports System.IO
Imports ExcelDataReader
Imports System
Imports System.Windows.Forms
Imports System.Text
Imports System.Linq
Imports System.Data.SqlClient
Public Class Time_and_Reference_Frame_in_GPS
Dim d, m, y, u, minute, second, h, JD, JDIN, MJD, GPSJD, GPSWeek, GPSDay,
GPSDayy, a, b, c, d1, e1, DD, MM, YY As Double
Dim INGPSDAY, INGPSWEEK As Double
d = TextBox1.Text
m = TextBox2.Text
y = TextBox3.Text
u = TextBox4.Text
minute = (TextBox13.Text) / 60
second = (TextBox14.Text) / 3600
h = u + minute + second
If (TextBox2.Text = 2) And (TextBox1.Text > 29) Then
MsgBox("February has no more than 29 days")
Hide()
End If
If TextBox1.Text > 31 Then
MsgBox("No month has more than 31 days")
Hide()
Form1.Show()
End If
If TextBox2.Text > 12 Then
MsgBox("The year just 12 month")
Hide()
Form1.Show()
End If
```

```
If m <= 2 Then
JD = Int(365.25 * (y - 1)) + Int(30.6001 * ((m + 12) + 1)) + d + ((h) / 24) +
1720981.5
TextBox5.Text = JD
Else
JD = Int(365.25 * y) + Int(30.6001 * (m + 1)) + d + ((h) / 24) + 1720981.5
TextBox5.Text = JD
End If
MJD = JD - 2400000.5
TextBox6.Text = MJD
GPSJD = JD - 2444244.5
TextBox7.Text = GPSJD
If (TextBox7.Text < 0) Then
TextBox5.Clear()
TextBox6.Clear()
TextBox7.Clear()
TextBox8.Clear()
TextBox9.Clear()
MsgBox("Date is befor GPS epoch")
End
End If
GPSWeek = Int(GPSJD / 7)
TextBox8.Text = GPSWeek
GPSDay = ((Int(JD) - Int(2444244.5)) / 7 - GPSWeek) * 7
TextBox9.Text = GPSDay
TextBox9.Text = Math.Round(Val(TextBox9.Text), 0,
MidpointRounding.AwayFromZero)
If TextBox9.Text = 0 Then
Label21.Text = "Sunday"
End If
If TextBox9.Text = 1 Then
Label21.Text = "Monday"
End If
If TextBox9.Text = 2 Then
```

```
Label21.Text = "Tuesday"
End If
If TextBox9.Text = 3 Then
Label21.Text = "Wednesday"
End If
If TextBox9.Text = 4 Then
Label21.Text = "Thrusday"
End If
If TextBox9.Text = 5 Then
Label21.Text = "Friday"
End If
If TextBox9.Text = 6 Then
Label21.Text = "Saturday"
End If
If TextBox9.Text >= 7 Then
Label21.Text = "GPSDays no more than 7 days"
End If

JDIN = TextBox15.Text
a = Int(JDIN + 0.5)
b = a + 1537
c = Int((b - 122.1) / 365.25)
d1 = Int(365.25 * c)
e1 = Int((b - d1) / 30.6001)
DD = b - d1 - Int(30.6001 * e1) + ((JDIN + 0.5) - Int(JDIN + 0.5))
TextBox10.Text = Int(DD)
MM = e1 - 1 - 12 * Int(e1 / 14)
TextBox11.Text = MM
YY = c - 4715 - Int((7 + MM) / 10)
TextBox12.Text = YY

INGPSDAY = TextBox16.Text
INGPSWEEK = TextBox16.Text + 2444244.5
a = Int(INGPSWEEK + 0.5)
b = a + 1537
```

```
c = Int((b - 122.1) / 365.25)
d1 = Int(365.25 * c)
e1 = Int((b - d1) / 30.6001)
DD = b - d1 - Int(30.6001 * e1) + ((INGPSWEEK + 0.5) - (INGPSWEEK + 0.5))
TextBox17.Text = (DD)
MM = e1 - 1 - 12 * Int(e1 / 14)
TextBox18.Text = MM
YY = c - 4715 - Int((7 + MM) / 10)
TextBox19.Text = YY
```

APPENDIX– (B)

Coordinate Conversion

Coordinate Conversion

```
Dim Latitude, Latitudedeg, Latitudemin, Latitudesec, Longitude, Longitudedeg,
Longitudemin, Longitudeesec, Heigh, N, f, e2, asemimajor, bsemiminor, ECEFx,
ECEFy, ECEFzAs Double
Dim xx, ycor, zz, aa, bb, lmda, phai, hh, nn, ee As Double
Private Function LatitudeLongitudeHeightToXYZ() As Double
Latitudedeg = InputBox("Enter Latitude in degree ")
Latitudemin = InputBox("Enter Latitude in minute ")
Latitudesec = InputBox("Enter Latitude in second ")
Longitudedeg = InputBox("Enter  Longitude in degree ")
Longitudemin = InputBox("Enter  Longitude in minute ")
Longitudeesec = InputBox("Enter  Longitude in second  ")
Heigh = InputBox("Enter Height ")
asemimajor = InputBox("Enter semimajor ")
bsemiminor = InputBox("Enter semiminor ")
Label29.Text = "Latitude = "
Label30.Text = "Longitude = "
Label31.Text = "Heigh= "
Label37.Text = "semi major = "
Label38.Text = "semi minor = "
Latitude = Latitudedeg + (Latitudemin / 60) + (Latitudesec / 3600)
Label32.Text = Latitude
Longitude = Longitudedeg + (Longitudemin / 60) + (Longitudeesec / 3600)
Label33.Text = Longitude
Label34.Text = Heigh
Label35.Text = asemimajor
Label36.Text = bsemiminor
f = (asemimajor - bsemiminor) / asemimajor
e2 = (2 * f) - (f) ^ 2
N = asemimajor / (1 - (e2) * (Math.Sin(Latitude * (pi / 180))) ^ 2) ^ 0.5
ECEFx = ((N) + Heigh) * (Math.Cos(Latitude * (pi / 180))) *
(Math.Cos(Longitude * (pi / 180)))
Label39.Text = ECEFx
```

```
ECEFy = ((N) + Heigh) * (Math.Cos(Latitude * (pi / 180))) *
(Math.Sin(Longitude * (pi / 180)))
Label41.Text = ECEFy
ECEFz = (((1 - (e2)) * N) + Heigh) * (Math.Sin(Latitude * (pi / 180)))
Label44.Text = ECEFz
Label40.Text = "X = "
Label42.Text = "Y = "
Label43.Text = "Z = "
Return MessageBox.Show("The Result")
End Function
Private Sub Button6_Click(sender As Object, e As EventArgs) Handles
Button6.Click
MessageBox.Show(XYZToLatitudeLongitudeHeight())
End Sub
Private Function XYZToLatitudeLongitudeHeight() As Double
xx = InputBox("Enter X Coordinat ")
ycor = InputBox("Enter Y Coordinat  ")
zz = InputBox("Enter Z Coordinat ")
aa = InputBox("Enter  semi major ")
bb = InputBox("Enter  semi minor ")
Label46.Text = "Coordinat X = "
Label47.Text = "Coordinat Y = "
Label48.Text = "Coordinat Z = "
Label49.Text = "semi major = "
Label50.Text = "semi minor = "
Label56.Text = "Longitud= "
Label57.Text = "Latitude   = "
Label58.Text = "Height     = "
Label51.Text = xx
Label52.Text = ycor
Label53.Text = zz
Label54.Text = aa
Label55.Text = bb
ee = ((aa) ^ 2 - (bb) ^ 2) / (aa) ^ 2
```

```
phai = (Math.Atan((zz / ((xx) ^ 2 + (ycor) ^ 2) ^ 0.5) * (1 - ee) ^ -1)) * (180 / pi)
Label59.Text = phai
nn = aa / (1 - (ee) * (Math.Sin(phai * (pi / 180))) ^ 2) ^ 0.5
lmda = Math.Atan((ycor / xx)) * (180 / pi)
Label60.Text = lmda
hh = (((xx) ^ 2 + (ycor) ^ 2) ^ 0.5 / (Math.Cos(phai * (pi / 180)))) - nn
Label61.Text = hh
Return MessageBox.Show("The Result")
End Function
```

APPENDIX– (C)

GPS GEODESY

```
Imports System.IO
Imports ExcelDataReader
Imports System
Imports System.Windows.Forms
Imports System.Text
Imports System.Linq
Dim xposition, yposition, zposition, phaiposition, phaipositiondeg,
phaipositionmin, phaipositionsec, lmdaposition, lmdapositiondeg,
lmdapositionmin, lmdapositionsec, xsta, ysta, zsta As Double
Const pi = 3.141592654
Const R = 6371
Dim tables As DataTableCollection
Dim table As New DataTable()
Private Sub Button7_Click(sender As Object, e As EventArgs) Handles
Button7.Click
Using ofd As OpenFileDialog = New OpenFileDialog() With {.Filter = "Excel
Workook|*.xlsx|Excel 97-2003 WorkBook|*.xls"}
If ofd.ShowDialog = Windows.Forms.DialogResult.OK Then
TextBox1.Text = ofd.FileName
Using Stream = File.Open(ofd.FileName, FileMode.Open, FileAccess.Read)
Using reader As IExcelDataReader =
ExcelReaderFactory.CreateReader(Stream)
Dim result As DataSet = reader.AsDataSet(New ExcelDataSetConfiguration()
With {
.ConfigureDataTable = Function(__) New ExcelDataTableConfiguration() With
{
.UseHeaderRow = True}})
tables = result.Tables
ComboBox1.Items.Clear()
For Each table As DataTable In tables
ComboBox1.Items.Add(table.TableName)
Next
End Using
End Using
End If
End Using
End Sub
Private Sub Button8_Click(sender As Object, e As EventArgs) Handles
Button8.Click
xposition = TextBox33.Text
yposition = TextBox40.Text
zposition = TextBox41.Text
phaipositiondeg = TextBox45.Text
phaipositionmin = TextBox46.Text
phaipositionsec = TextBox47.Text
lmdapositiondeg = TextBox48.Text
lmdapositionmin = TextBox49.Text
lmdapositionsec = TextBox50.Text
```

```
phaiposition = phaipositiondeg + (phaipositionmin / 60) + (phaipositionsec /
3600)
lmdaposition = lmdapositiondeg + (lmdapositionmin / 60) + (lmdapositionsec /
3600)
xsta = TextBox42.Text
ysta = TextBox43.Text
zsta = TextBox44.Text
TextBox21.Text = -1 * (Math.Sin((phaiposition * (pi / 180))) *
Math.Cos(lmdaposition * (pi / 180)))
TextBox22.Text = -1 * (Math.Sin((phaiposition * (pi / 180))) *
Math.Sin(lmdaposition * (pi / 180)))
TextBox23.Text = Math.Cos(phaiposition * (pi / 180))
TextBox24.Text = -1 * (Math.Sin((lmdaposition * (pi / 180))))
TextBox25.Text = (Math.Cos((lmdaposition * (pi / 180))))
TextBox26.Text = 0
TextBox27.Text = (Math.Cos((phaiposition * (pi / 180))) *
Math.Cos(lmdaposition * (pi / 180)))
TextBox28.Text = (Math.Cos((phaiposition * (pi / 180))) *
Math.Sin(lmdaposition * (pi / 180)))
TextBox29.Text = (Math.Sin((phaiposition * (pi / 180))))
TextBox30.Text = xsta - xposition
TextBox31.Text = ysta - yposition
TextBox32.Text = zsta - zposition
TextBox34.Text = (TextBox21.Text * TextBox30.Text) + (TextBox22.Text *
TextBox31.Text) + (TextBox23.Text * TextBox32.Text)
TextBox35.Text = (TextBox24.Text * TextBox30.Text) + (TextBox25.Text *
TextBox31.Text) + (TextBox26.Text * TextBox32.Text)
TextBox36.Text = (TextBox27.Text * TextBox30.Text) + (TextBox28.Text *
TextBox31.Text) + (TextBox29.Text * TextBox32.Text)
End Sub
Private Sub DataGridView1_CellContentClick(sender As Object, e As
DataGridViewCellEventArgs) Handles DataGridView1.CellContentClick
TextBox42.Text = DataGridView1.CurrentRow.Cells(2).Value
TextBox43.Text = DataGridView1.CurrentRow.Cells(3).Value
TextBox44.Text = DataGridView1.CurrentRow.Cells(4).Value
End Sub
Private Sub Button9_Click(sender As Object, e As EventArgs) Handles
Button9.Click
Dim horizantal, zenith, azimuth As Double
horizantal = Math.Sqrt((TextBox34.Text) ^ 2 + (TextBox35.Text) ^ 2)
TextBox37.Text = horizantal
zenith = Math.Atan2(horizantal, TextBox36.Text) * (180 / pi)
TextBox38.Text = zenith
azimuth = Math.Atan2(TextBox35.Text, TextBox34.Text) * (180 / pi)
TextBox39.Text = azimuth
Me.CreateGraphics().DrawPie(Pens.Pink, 50, 50, 200, 200, 0, 360)
End Sub
Private Sub Time_and_Reference_Frame_in_GPS_Load(sender As Object, e As
EventArgs) Handles MyBase.Load
table.Columns.Add("P", Type.GetType("System.String"))
```

```
table.Columns.Add("Satellite Number", Type.GetType("System.Double"))
table.Columns.Add("X Coordinates", Type.GetType("System.Double"))
table.Columns.Add("Y Coordinates", Type.GetType("System.Double"))
table.Columns.Add("Z Coordinates", Type.GetType("System.Double"))
table.Columns.Add("Time", Type.GetType("System.Double"))
DataGridView1.DataSource = table
End Sub
Private Sub ComboBox1_SelectedIndexChanged(sender As Object, e As
EventArgs) Handles ComboBox1.SelectedIndexChanged
Dim dt As DataTable = tables(ComboBox1.SelectedItem.ToString)
DataGridView1.DataSource = dt
End Sub
Private Sub Button10_Click(sender As Object, e As EventArgs) Handles
Button10.Click
Dim zenith, azimuth As Double
zenith = TextBox38.Text
azimuth = TextBox39.Text
Chart1.Series(0).Points.AddXY(zenith, azimuth)
End Sub
End Class
```

APPENDIX– (D)

Velocity and Precise Time (PVT)

Velocity and Precise Time (PVT)

```
Imports System.IO
Imports ExcelDataReader
Imports System
Imports System.Windows.Forms
Imports System.Text
Imports System.Linq
Dim pseudorange, ax, ay, az, dltap1, UserClockMisalignment, dltap2, dltap3,
dltap4 As Double
Dim a11, a12, a13, a14, a21, a22, a23, a24, a31, a32, a33, a34, a41, a42, a43, a44,
res, res1, res2, res3, res4, GDOP, PDOP, TDOP, HDOP As Double
Const SpeedLight = 300000000
Dim tables As DataTableCollection
Dim table As New DataTable()
Private Sub Button11_Click(sender As Object, e As EventArgs) Handles
Button11.Click
Using ofd As OpenFileDialog = New OpenFileDialog() With {.Filter = "Excel
Workook|*.xlsx|Excel 97-2003 WorkBook|*.xls"}
If ofd.ShowDialog = Windows.Forms.DialogResult.OK Then
Label91.Text = ofd.FileName
Using Stream = File.Open(ofd.FileName, FileMode.Open, FileAccess.Read)
Using reader As IExcelDataReader =
ExcelReaderFactory.CreateReader(Stream)
Dim result As DataSet = reader.AsDataSet(New ExcelDataSetConfiguration()
With {
.ConfigureDataTable = Function(__) New ExcelDataTableConfiguration() With
{
.UseHeaderRow = True}})
tables = result.Tables
ComboBox2.Items.Clear()
For Each table As DataTable In tables
ComboBox2.Items.Add(table.TableName)
Next
End Using
End Using
End If
End Using
End Sub
Private Sub ComboBox2_SelectedIndexChanged(sender As Object, e As
EventArgs) Handles ComboBox2.SelectedIndexChanged
Dim dt As DataTable = tables(ComboBox2.SelectedItem.ToString)
DataGridView2.DataSource = dt
End Sub
Private Sub DataGridView2_CellContentClick(sender As Object, e As
DataGridViewCellEventArgs) Handles DataGridView2.CellContentClick
TextBox54.Text = DataGridView2.CurrentRow.Cells(2).Value
TextBox55.Text = DataGridView2.CurrentRow.Cells(3).Value
TextBox56.Text = DataGridView2.CurrentRow.Cells(4).Value
End Sub
```

```
Private Sub Button12_Click(sender As Object, e As EventArgs) Handles
Button12.Click
pseudorange = ((TextBox54.Text - TextBox51.Text) ^ 2 + (TextBox55.Text -
TextBox52.Text) ^ 2 + (TextBox56.Text - TextBox53.Text) ^ 2) ^ 0.5
TextBox57.Text = pseudorange
End Sub
Private Sub Button13_Click(sender As Object, e As EventArgs) Handles
Button13.Click
ax = (TextBox54.Text - TextBox51.Text) / pseudorange
TextBox58.Text = ax
End Sub
Private Sub Button14_Click(sender As Object, e As EventArgs) Handles
Button14.Click
ay = (TextBox55.Text - TextBox52.Text) / pseudorange
TextBox59.Text = ay
End Sub
Private Sub Button15_Click(sender As Object, e As EventArgs) Handles
Button15.Click
az = (TextBox56.Text - TextBox53.Text) / pseudorange
TextBox60.Text = az
End Sub
Private Sub Button16_Click(sender As Object, e As EventArgs) Handles
Button16.Click
UserClockMisalignment = TextBox61.Text
dltap1 = ((TextBox54.Text - TextBox51.Text) ^ 2 + (TextBox55.Text -
TextBox52.Text) ^ 2 + (TextBox56.Text - TextBox53.Text) ^ 2) ^ 0.5 +
(SpeedLight * UserClockMisalignment)
TextBox62.Text = dltap1
End Sub
Private Sub Button17_Click(sender As Object, e As EventArgs) Handles
Button17.Click
UserClockMisalignment = TextBox61.Text
dltap2 = ((TextBox54.Text - TextBox51.Text) ^ 2 + (TextBox55.Text -
TextBox52.Text) ^ 2 + (TextBox56.Text - TextBox53.Text) ^ 2) ^ 0.5 +
(SpeedLight * UserClockMisalignment)
TextBox63.Text = dltap2
End Sub
Private Sub Button18_Click(sender As Object, e As EventArgs) Handles
Button18.Click
UserClockMisalignment = TextBox61.Text
dltap3 = ((TextBox54.Text - TextBox51.Text) ^ 2 + (TextBox55.Text -
TextBox52.Text) ^ 2 + (TextBox56.Text - TextBox53.Text) ^ 2) ^ 0.5 +
(SpeedLight * UserClockMisalignment)
TextBox64.Text = dltap3
End Sub
Private Sub Button19_Click(sender As Object, e As EventArgs) Handles
Button19.Click
UserClockMisalignment = TextBox61.Text
```

```
dltap4 = ((TextBox54.Text - TextBox51.Text) ^ 2 + (TextBox55.Text -
TextBox52.Text) ^ 2 + (TextBox56.Text - TextBox53.Text) ^ 2) ^ 0.5 +
(SpeedLight * UserClockMisalignment)
TextBox65.Text = dltap4
End Sub
Private Sub Button20_Click(sender As Object, e As EventArgs) Handles
Button20.Click
TextBox66.Text = TextBox58.Text
TextBox67.Text = TextBox59.Text
TextBox68.Text = TextBox60.Text
TextBox69.Text = 1
End Sub
Private Sub Button21_Click(sender As Object, e As EventArgs) Handles
Button21.Click
TextBox70.Text = TextBox58.Text
TextBox71.Text = TextBox59.Text
TextBox72.Text = TextBox60.Text
TextBox73.Text = 1
End Sub
Private Sub Button22_Click(sender As Object, e As EventArgs) Handles
Button22.Click
TextBox74.Text = TextBox58.Text
TextBox75.Text = TextBox59.Text
TextBox76.Text = TextBox60.Text
TextBox77.Text = 1
End Sub
Private Sub Button23_Click(sender As Object, e As EventArgs) Handles
Button23.Click
TextBox78.Text = TextBox58.Text
TextBox79.Text = TextBox59.Text
TextBox80.Text = TextBox60.Text
TextBox81.Text = 1
End Sub
Private Sub Button24_Click(sender As Object, e As EventArgs) Handles
Button24.Click
TextBox82.Text = TextBox66.Text
TextBox86.Text = TextBox67.Text
TextBox90.Text = TextBox68.Text
TextBox94.Text = TextBox69.Text
TextBox83.Text = TextBox70.Text
TextBox87.Text = TextBox71.Text
TextBox91.Text = TextBox72.Text
TextBox95.Text = TextBox73.Text
TextBox84.Text = TextBox74.Text
TextBox88.Text = TextBox75.Text
TextBox92.Text = TextBox76.Text
TextBox96.Text = TextBox77.Text
TextBox85.Text = TextBox78.Text
TextBox89.Text = TextBox79.Text
TextBox93.Text = TextBox80.Text
```

```
TextBox97.Text = TextBox81.Text
End Sub
Private Sub Button25_Click(sender As Object, e As EventArgs) Handles
Button25.Click
TextBox98.Text = ((TextBox82.Text * TextBox66.Text) + (TextBox83.Text *
TextBox70.Text) + (TextBox84.Text * TextBox74.Text) + (TextBox85.Text *
TextBox78.Text))
TextBox99.Text = ((TextBox82.Text * TextBox67.Text) + (TextBox83.Text *
TextBox71.Text) + (TextBox84.Text * TextBox75.Text) + (TextBox85.Text *
TextBox79.Text))
TextBox100.Text = ((TextBox82.Text * TextBox68.Text) + (TextBox83.Text *
TextBox72.Text) + (TextBox84.Text * TextBox76.Text) + (TextBox85.Text *
TextBox80.Text))
TextBox101.Text = ((TextBox82.Text * TextBox69.Text) + (TextBox83.Text *
TextBox73.Text) + (TextBox84.Text * TextBox77.Text) + (TextBox85.Text *
TextBox81.Text))
TextBox102.Text = ((TextBox86.Text * TextBox66.Text) + (TextBox87.Text *
TextBox70.Text) + (TextBox88.Text * TextBox74.Text) + (TextBox89.Text *
TextBox78.Text))
TextBox103.Text = ((TextBox86.Text * TextBox67.Text) + (TextBox87.Text *
TextBox71.Text) + (TextBox88.Text * TextBox75.Text) + (TextBox89.Text *
TextBox79.Text))
TextBox104.Text = ((TextBox86.Text * TextBox68.Text) + (TextBox87.Text *
TextBox72.Text) + (TextBox88.Text * TextBox76.Text) + (TextBox89.Text *
TextBox80.Text))
TextBox105.Text = ((TextBox86.Text * TextBox69.Text) + (TextBox87.Text *
TextBox73.Text) + (TextBox88.Text * TextBox77.Text) + (TextBox89.Text *
TextBox81.Text))
TextBox106.Text = ((TextBox90.Text * TextBox66.Text) + (TextBox91.Text *
TextBox70.Text) + (TextBox92.Text * TextBox74.Text) + (TextBox93.Text *
TextBox78.Text))
TextBox107.Text = ((TextBox90.Text * TextBox67.Text) + (TextBox91.Text *
TextBox71.Text) + (TextBox92.Text * TextBox75.Text) + (TextBox93.Text *
TextBox79.Text))
TextBox108.Text = ((TextBox90.Text * TextBox68.Text) + (TextBox91.Text *
TextBox72.Text) + (TextBox92.Text * TextBox76.Text) + (TextBox93.Text *
TextBox80.Text))
TextBox109.Text = ((TextBox90.Text * TextBox69.Text) + (TextBox91.Text *
TextBox73.Text) + (TextBox92.Text * TextBox77.Text) + (TextBox93.Text *
TextBox81.Text))
TextBox110.Text = ((TextBox94.Text * TextBox66.Text) + (TextBox95.Text *
TextBox70.Text) + (TextBox96.Text * TextBox74.Text) + (TextBox97.Text *
TextBox78.Text))
TextBox111.Text = ((TextBox94.Text * TextBox67.Text) + (TextBox95.Text *
TextBox71.Text) + (TextBox96.Text * TextBox75.Text) + (TextBox97.Text *
TextBox79.Text))
TextBox112.Text = ((TextBox94.Text * TextBox68.Text) + (TextBox95.Text *
TextBox72.Text) + (TextBox96.Text * TextBox76.Text) + (TextBox97.Text *
TextBox80.Text))
```

```
TextBox113.Text = ((TextBox94.Text * TextBox69.Text) + (TextBox95.Text *
TextBox73.Text) + (TextBox96.Text * TextBox77.Text) + (TextBox97.Text *
TextBox81.Text))
End Sub
Private Sub Button26_Click(sender As Object, e As EventArgs) Handles
Button26.Click
a11 = TextBox98.Text
a12 = TextBox99.Text
a13 = TextBox100.Text
a14 = TextBox101.Text
a21 = TextBox102.Text
a22 = TextBox103.Text
a23 = TextBox104.Text
a24 = TextBox105.Text
a31 = TextBox106.Text
a32 = TextBox107.Text
a33 = TextBox108.Text
a34 = TextBox109.Text
a41 = TextBox110.Text
a42 = TextBox111.Text
a43 = TextBox112.Text
a44 = TextBox113.Text
res1 = (a11 * a22 * a33 * a44) + (a11 * a23 * a34 * a42) + (a11 * a24 * a32 * a43) -
(a11 * a24 * a33 * a42) - (a11 * a23 * a32 * a44) - (a11 * a22 * a34 * a43)
res2 = -(a12 * a21 * a33 * a44) - (a13 * a21 * a34 * a42) - (a14 * a21 * a32 * a43)
+ (a14 * a21 * a33 * a42) + (a13 * a21 * a32 * a44) + (a12 * a21 * a34 * a43)
res3 = (a12 * a23 * a31 * a44) + (a13 * a24 * a31 * a42) + (a14 * a22 * a31 * a43) -
(a14 * a23 * a31 * a42) - (a13 * a22 * a31 * a44) - (a12 * a24 * a31 * a43)
res4 = -(a12 * a23 * a34 * a41) - (a13 * a24 * a32 * a41) - (a14 * a22 * a33 * a41)
+ (a14 * a23 * a32 * a41) + (a13 * a22 * a34 * a41) + (a12 * a24 * a33 * a41)
res = res1 + res2 + res3 + res4
TextBox114.Text = (1 / res) * ((a22 * a33 * a44) + (a23 * a34 * a42) + (a24 * a32 *
a43) - (a24 * a33 * a42) - (a23 * a32 * a44) - (a22 * a34 * a43))
GDOP = TextBox114.Text
TextBox115.Text = (1 / res) * (-(a12 * a33 * a44) - (a13 * a34 * a42) - (a14 * a32 *
a43) + (a14 * a33 * a42) + (a13 * a32 * a44) + (a12 * a34 * a43))
TextBox116.Text = (1 / res) * ((a12 * a23 * a44) + (a13 * a24 * a42) + (a14 * a22 *
a43) - (a14 * a23 * a42) - (a13 * a22 * a44) - (a12 * a24 * a43))
TextBox117.Text = (1 / res) * (-(a12 * a23 * a34) - (a13 * a24 * a32) - (a14 * a22 *
a33) + (a14 * a23 * a32) + (a13 * a22 * a34) + (a12 * a24 * a33))
TextBox118.Text = (1 / res) * (-(a21 * a33 * a44) - (a23 * a34 * a41) - (a24 * a31 *
a43) + (a24 * a33 * a41) + (a23 * a31 * a44) + (a21 * a34 * a43))
TextBox119.Text = (1 / res) * ((a11 * a33 * a44) + (a13 * a34 * a41) + (a14 * a31 *
a43) - (a14 * a33 * a41) - (a13 * a31 * a44) - (a11 * a34 * a43))
PDOP = TextBox119.Text
TextBox120.Text = (1 / res) * (-(a11 * a23 * a44) - (a13 * a24 * a41) - (a14 * a21 *
a43) + (a14 * a23 * a41) + (a13 * a21 * a44) + (a11 * a24 * a43))
TextBox121.Text = (1 / res) * ((a11 * a23 * a34) + (a13 * a24 * a31) + (a14 * a21 *
a33) - (a14 * a23 * a31) - (a13 * a21 * a34) - (a11 * a24 * a33))
```

```
TextBox122.Text = (1 / res) * ((a21 * a32 * a44) + (a22 * a34 * a41) + (a24 * a31 *
a42) - (a24 * a32 * a41) - (a22 * a31 * a44) - (a21 * a34 * a42))
TextBox123.Text = (1 / res) * (-(a11 * a32 * a44) - (a12 * a34 * a41) - (a14 * a31 *
a42) + (a14 * a32 * a41) + (a12 * a31 * a44) + (a11 * a34 * a42))
TextBox124.Text = (1 / res) * ((a11 * a22 * a44) + (a12 * a24 * a41) + (a14 * a21 *
a42) - (a14 * a22 * a41) - (a12 * a21 * a44) - (a11 * a24 * a42))
TDOP = TextBox124.Text
TextBox125.Text = (1 / res) * (-(a11 * a22 * a34) - (a12 * a24 * a31) - (a14 * a21 *
a32) + (a14 * a22 * a31) + (a12 * a21 * a34) + (a11 * a24 * a32))
TextBox126.Text = (1 / res) * (-(a21 * a32 * a43) - (a22 * a33 * a41) - (a23 * a31 *
a42) + (a23 * a32 * a41) + (a22 * a31 * a43) + (a21 * a33 * a42))
TextBox127.Text = (1 / res) * ((a11 * a32 * a43) + (a12 * a33 * a41) + (a13 * a31 *
a42) - (a13 * a32 * a41) - (a12 * a31 * a43) - (a11 * a33 * a42))
TextBox128.Text = (1 / res) * (-(a11 * a22 * a43) - (a12 * a23 * a41) - (a13 * a21 *
a42) + (a13 * a22 * a41) + (a12 * a21 * a43) + (a11 * a23 * a42))
TextBox129.Text = (1 / res) * ((a11 * a22 * a33) + (a12 * a23 * a31) + (a13 * a21 *
a32) - (a13 * a22 * a31) - (a12 * a21 * a33) - (a11 * a23 * a32))
HDOP = TextBox129.Text
End Sub
Private Sub Button27_Click(sender As Object, e As EventArgs) Handles
Button27.Click
TextBox130.Text = (GDOP + PDOP + TDOP + HDOP) ^ 0.5
TextBox131.Text = (GDOP + PDOP + TDOP) ^ 0.5
TextBox132.Text = (HDOP) ^ 0.5
TextBox133.Text = (GDOP + PDOP) ^ 0.5
End Sub
Private Sub Button28_Click(sender As Object, e As EventArgs) Handles
Button28.Click
For Count As Integer = 0 To DataGridView2.Rows.Count - 2
Chart2.Series(0).Points.AddXY(DataGridView2.Item(5, Count).Value,
DataGridView2.Item(1, Count).Value)
Next
Private Sub Button9_Click(sender As Object, e As EventArgs) Handles

Button9.Click

Dim sfg As New SaveFileDialog

sfg.FileName = " "

sfg.Filter = "Text Document|*.txt"

sfg.Title = "Save Datagridview as"

If sfg.ShowDialog = DialogResult.OK Then

Dim writer As TextWriter = New StreamWriter(sfg.FileName)

For i As Integer = 0 To DataGridView7.Rows.Count - 2

For j As Integer = 0 To DataGridView7.Columns.Count - 1

writer.Write(vbTab & DataGridView7.Rows(i).Cells(j).Value & vbTab & " | ")

Next
```

```
writer.WriteLine("   ")
writer.WriteLine("   ")
writer.WriteLine("   ")
writer.WriteLine("---------------------- ")
Next
writer.Close()
MessageBox.Show("Data Exported")
End If
End Sub
End Class
```