

Palestine Polytechnic University



College of Engineering & Technology

Civil & Architectural Engineering Department

Surveying Engineering

Graduation Project

GEO-TRANSFORM

Prepared By:

Manar Al Jabari

Sumya Zahdeh

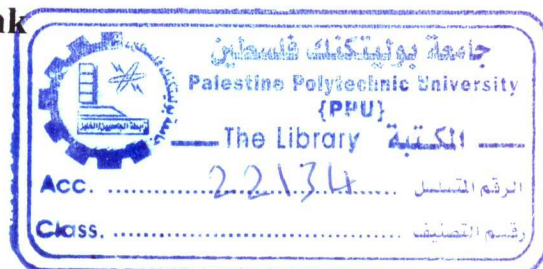
Supervisor:

MSc. Ghadi Zakarneh

Zakarneh

2008/5/2

Hebron-West Bank
Palestine
January-2008



إهداء

إلى الأم الحسنة والأب الغالي

إلى أحباب الله

إلى الذين يصافحون الشمس والقمر والسماء

إلى من حزن لحزني و فرح لفرحي . . . إلى أصدقائي الأعزاء.

إلى السائرين في سفينة شراعها دماء

إلى كل الشهداء

إلى كل من ساعد أناملنا على أن تحقق جزءاً من آمالنا

إلى كل هؤلاء فهدي هذا الجهد المتواضع

لو كان يهدى للأحباب قيمتهم
لأهديناهم الدنيا وما فيها

فريق البحث:

سمية زاهدة

منار الجعبري

ACKNOWLEDGMENT

We would like to extend our sincere appreciation to MSc. Zakarneh Ghadi who is sincerely appreciated for his many significant suggestions and advices. We are most grateful for his generosity. And we would like to express our grateful to teacher Ohwawi Maher for their devises.

Finally, we extended our thank to members in the Civil Engineering Dept. Especially the part of Surveying Engineering, which help us in designing our project.

Work Team

ABSTRACT

Geo - Transform

By:

Manar Al Jabari

Sumya Zahdeh

Palestine Polytechnic University

Supervisor

MSc. Ghadi Zakarneh

This project aims to build Geodetic Transformation software using visual Basic 6 programming language. This tool should be able to transform from Geographic coordinates (λ , ϕ , h) to geocentric coordinates (X , Y , Z), and from geographic coordinates (λ , ϕ) to projected coordinates (Easting, Northing) and in the opposite direction using casini, Transverse Mercator (TM) and Universal Transverse Mercator (UTM). So that for surveying point there would be no need to use expensive software, and the time will not be spent to draw the point in the software format like GIS and CAD software.

There are many software available to apply these calculation, such as Arcview, ArcGIS, Autodesk land desktop, ER Mapper, ERDAS, ...etc. but these software are expensive for single user.

The projected coordinates systems used are: Palestine_1923 Belt, Palestine_1923 old Grid, and the Israeli Transverse Mercator. In addition a separate datum transformation tool is to be built, to calculate the datum transformation parameters using different transformation method; similarity 7-parametres, similarity 6-parameters, similarity 3-parameters, Moldensky transformation and Helmert transformations.

The final product will be software can be setup at any computer to calculate different types of coordinates for single or multipoint calculations.

TABLE OF CONTENTS

Title	I
Dedication	II
Acknowledgment	III
Abstract	IV
Table of contents	V
List of figures	X
List of tables	XV
 CHAPTER 1	
INTRODUCTION	1
1.1 Background	2
1.2 Objectives	3
1.3 Previous studies	3
1.4 Project plan	3
1.5 Methodology	5
 CHAPTER 2	
COORDINATE SYSTEMS	6
2.1 Introduction	7
2.2 Spherical coordinate	8
2.3 Ellipsoidal coordinate	9
2.3.1 Ellipsoidal earth figure	9
2.3.2 Geographic coordinate	12
2.3.3 Geocentric coordinate	13

2.3.4 Topocentric coordinate	14
2.4 Map Projection	15
2.4.1 Map projections general classes	16
2.4.2 Transverse Mercator	21
2.4.3 Cassini_Soldner projection	25
2.4.4 (UTM) Universal Transverse Mercator	28
2.4.5 Projected coordinate	30
CHAPTER 3	DATUM TRANSFORMATION
	32
3.1 Introduction	33
3.2 Datums & Ellipsoids	34
3.3 Datum Transformations	37
3.3.1 Introduction	37
3.3.2 Least square solution	38
3.3.3 3D Similiarity 7-parameter	40
3.3.4 3D Similiarity 6-parameter	46
3.3.5 3D Similiarity 3-parameter	52
3.3.6 Helmert transformation	54
3.3.7 Moldensky method	57
3.3.7.1 Vector Viewpoint of Molodensky Transformation	57
3.3.7.2 The Usual Statement of the Molodensky Transformation	58

4.1 Introduction	63
4.2 using the Geo-transform	63
4.2.1 Transform from geographic to geocentric coordinates for a single point	64
4.2.2 Transform from Geocentric coordinates to Geographic coordinates for a single point	66
4.2.3 Transform from Geographic coordinates (λ, ϕ) to projected coordinates (Easting, Northing) for a single point	68
4.2.4 Transform from projected coordinates (Easting, Northing) to Geographic coordinates (λ, ϕ) for a single point:	73
4.2.5 Transform from Geographic coordinates to Geocentric coordinates for a point file	75
4.2.6 Transform from geocentric coordinates (X, Y, Z) to geographic coordinates (lon, lat, h) for a point file	79
4.2.7 Transform from geographic coordinates(λ, ϕ) to projected (Easting, Northing) for a point file	85

4.2.8 Transform from projected coordinates (Easting, Northing) to geographic coordinates (λ , ϕ)	91
4.3 Starting geo transform for using datum transformation	96
4.3.1 Transform using 7 parameter similarity for a single point	100
4.3.2 Transform using 6 parameter similarity for a single point	102
4.3.3 Transform using 3 parameter similarity for a single point	104
4.3.4 Transform using Helmert transformation	105
4.3.5 Transform using Meldensky transformation for a single point	107
4.3.6 Transform using 7- parameters similarity for points file	108
4.3.7 Transform using 6- parameters for points file	110
4.3.8 Transform using 3- parameters for points file	111
4.3.9 Transform using Helmert method for points file	113
4.3.10 Transform using moldensky method for points file	114

CHAPTER 5 CONCLUSION & RECOMMENDATION 117

5.1 Conclusion	118
5.2 Recommendation	118
Reference	119

APPENDIX 120

Appendix - A	121
Appendix - B	179
Appendix - C	201

LIST OF FIGURES

Figure No	Description	Page No
2.1	The Earth	7
2.2	Sphere as the earth's figure	8
2.3	Ellipsoid as the earth's figure	9
2.4	Geographic coordinates system	12
2.5	Geocentric system	13
2.6	Topocentric Coordinates system	14
2.7	Map projection (from 3D to Grid system)	16
2.8	Projection of a sphere onto a cylinder (tangent case)	17
2.9	Projection of a Sphere onto a Cylinder (Secant Case)	17
2.10	Transverse Projection of a Sphere onto a Cylinder (Tangent Case)	18
2.11	Oblique Projection of a Sphere onto a Cylinder (Tangent Case)	18
2.12	Projection of a Sphere onto a Cone (Tangent Case)	19
2.13	Projection of a Sphere onto a Cone (Secant Case)	19
2.14	Projection of a Sphere onto a Plane (Tangent Case)	20
2.15	Projection of a Sphere onto a Plane (Secant Case)	20
2.16	Transverse Mercator	21
2.17	Universal Transverse Mercator	29
3.1	(a) rotate ellipse, (b) making ellipsoid	34
3.2	Datum transformation	37
3.3	7-Parameter Transformation	40
3.4	6-Parameter Transformation	46

3.5	3-Parameter Transformation	52
4.1	coordinates menu	64
4.2	lon,lat,h $\rightarrow$ X,Y,Z dialog	65
4.3	lon,lat,h $\rightarrow$ X,Y,Z dialog	66
4.4	X,Y,Z $\rightarrow$ lon,lat,h dialog	67
4.5	X,Y,Z $\rightarrow$ lon,lat,h dialog	68
4.6	projection-direct dialog	69
4.7	projection-direct dialog	70
4.8	projection-direct dialog	71
4.9	projection-direct dialog	72
4.10	projection-inverse dialog	73
4.11	projection-inverse dialog	74
4.12	projection-inverse dialog	75
4.13	lon,lat,h $\rightarrow$ x,y,z dialog	76
4.14	open dialog	76
4.15	save as dialog	77
4.16	lon,lat,h $\rightarrow$ x,y,z dialog	77
4.17	open dialog	78
4.18	save as dialog	78
4.19	result dialog	79
4.20	result dialog	79
4.21	example 8	79
4.22	coordinate conversions dialog	80
4.23	open dialog	80

4.24	open dialog	81
4.25	save as dialog	81
4.26	geo transform dialog	82
4.27	open dialog	82
4.28	open dialog	83
4.29	geo transform dialog	83
4.30	save as dialog	84
4.31	results dialog	84
4.32	results dialog	84
4.33	results dialog	85
4.34	geo transform dialog	85
4.35	open dialog	86
4.36	open dialog	86
4.37	geo transform dialog	87
4.38	save as dialog	87
4.39	open dialog	88
4.40	geo transform dialog	89
4.41	save as dialog	89
4.42	results dialog	90
4.43	results dialog	90
4.44	geo transform dialog	91
4.45	open dialog	91
4.46	geo transform dialog	92
4.47	save as dialog	93

4.48	geo transform dialog	93
4.49	open dialog	94
4.50	save as dialog	95
4.51	results dialog	95
4.52	results dialog	95
4.53	results dialog	96
4.54	datum menu	96
4.55	datum menu	96
4.56	datum transformation dialog	97
4.57	datum transformation parameter dialog	98
4.58	open dialog	98
4.59	open dialog	99
4.60	select ellipsoid list	99
4.61	calculate key	99
4.62	save sa dialog	100
4.63	datum transformation dialog	100
4.64	open dialog	101
4.65	7-parameter simslarity dialog	102
4.66	open dialog	103
4.67	6 parameter similarity dialog	103
4.68	open dialog	104
4.69	3 parameter similarity dialog	105
4.70	open dialog	106
4.71	Helmert transformation dialog	106

4.72	open dialog	107
4.73	Moldensky transformation dialog	108
4.74	7 parameter similarity dialog	108
4.75	open dialog	109
4.76	6 parameter similarity dailog	110
4.77	open dialog	110
4.78	save as dialog	111
4.79	3 parameter similarity dialog	111
4.80	open dialog	112
4.81	save as dialog	112
4.82	helmert transformation dialog	113
4.83	open dialog	113
4.84	save as dialog	114
4.85	Moldensky transformation dialog	115
4.86	open dialog	115
4.87	save as dialog	116

LIST OF TABLES

Table No	Description	Page No
1.1	Time Table	4
2.1	Selected reference Ellipsoid	9
2.2	The relation between the different parameter	10
2.3	GRS80 ellipsoid parameters	11
2.4	GRS84 ellipsoid parameters	12
2.5	Projected Coordinate Systems Examples	30
3.1	The coordinate system with datum	35

CHAPTER

1

INTRODUCTION

1.1 Back ground

1.2 Objectives

1.3 Previous studies

1.4 Project plan

1.5 Methodology

CHAPTER ONE

INTRODUCTION

1-1 Background

This project is Geodetic Transformation tool programmed using visual Basic 6 programming language. This tool has the following functionalities; the First one to transform from Geographic coordinates (λ , ϕ , h) to geocentric coordinates (X , Y , Z), Where the same ellipsoid is used. The Second functionality to transform 3D geographic or geocentric coordinates (λ , ϕ) to 2D projected (Grid) coordinates (Easting, Northing), the used map projection in this transformation tool are: Cassini, Transverse Mercator (TM) and Universal Transverse Mercator (UTM). The Third is to apply the reverse transformation of the first and second functionalities.

In addition, a separate datum transformation tool is to be built to calculate the datum transformation parameters using different datum transformation methods similarity 7-parametres, similarity 6-parameters, similarity 3-parameters, Moldensky transformation and Helmert transformations.

In this tool three Basic map projections are used; Cassini, Transverse Mercator TM, and universal Transverse Mercator UTM. In Palestine there are four map projection and coordinate systems used, These are Palestine_1923_Palestine_Grid, Palestine_1923_Israel_CS_Grid, palestine_1923_Belt, and Israel_TM_Grid. Where each coordinates system has its own projection parameters.

1-2 Objectives

This project deals with visual Basic 6 programming language. This tool should be able to apply the following coordinate transformations,

1. Geographic coordinates (λ, ϕ, h) to Geocentric coordinates (X, Y, Z).
2. Geocentric coordinates (X, Y, Z) to Geographic coordinates (λ, ϕ, h).
3. Geographic coordinates (λ, ϕ) to projected coordinates (Easting, Northing), for Cassini, Transverse Mercator TM, and universal Transverse Mercator UTM.
4. Projected coordinates (Easting, Northing) to Geographic coordinates (λ, ϕ) using Cassini, Transverse Mercator (TM) and Universal Transverse Mercator (UTM).
5. Calculate the datum transformation parameters using different transformation method; similarity 7-parametres, similarity 6-parameters, similarity 3-parameters, Moldensky transformation and Helmert transformations

1-3 Previous studies

There are many software available to apply these calculations; such as Arcview, ArcGIS, Autodesk land desktop, ERMMapper, ERDAS ...etc. but these software are expensive for a single user. In addition, these software have large capabilities and steps of work because the major task is not the coordinates transformations, but coordinates transformations a secondary task, used to enhance the performance of the software.

1-4 Project plan

The project works will be achieved by the following steps:

1. literature review
2. Research plan, determining the problem solving scope.
3. Transform from Geographic coordinates (λ, ϕ, h) to Geocentric coordinates (X, Y, Z) and in the opposite direction.

4. Transform from Geographic coordinates (λ , ϕ) to projected coordinates (Easting, Northing) and in the opposite direction, using three basic map projections; Cassini, Transverse Mercator (TM) and Universal Transverse Mercator (UTM).
5. Four Basic coordinate systems are to be used: Palestine_1923_Palestine_Grid, Palestine_1923_Israel_CS_Grid, palestine_1923_Belt, Israel_TM_Grid. And insert false Easting, false Northing, latitude of origin, Central Meridian, and ellipsoid for each of coordinates system, are to be define in the software. And users define map projection using Cassini & TM is to allow.
6. Programming datum transformation tool as a separate software.

Table 1.1: Timetable

NO	Activities	Time(week)	Starting data (approximate)	Ending data (approximate)	Done yes/no
1	Problem definition	3	25/9/2007	6/10/2007	yes
2	Literature review	2	6/10/2007	20/10/2007	yes
3	Preparing research plan	6	20/10/2007	1/11/2008	yes
4	Coordinate system datum & datum transformation	4	1/11/2007	15/11/2008	yes
5	Introduction of Map projection	4	15/11/2008	1/12/2008	yes
6	Preparing a separate software	4	1/12/2008	1/2/2009	yes
7	Result of analysis	6	1/2/2009	15/3/2009	yes
8	Conclusion & recommendation	4	15/3/2009	15/4/2009	yes
9	Final Report	29	15/4/2009	Final date	yes

1-5 Methodology

This project has the following scope:

Chapter 1: This chapter introduces the project.

Chapter 2: This chapter explains the coordinates systems used in this project.

Chapter 3: This chapter describes the datum transformations methods and applications.

Chapter 4: The chapter shows the results and their analysis of the Geo-Transform.

Chapter 5: This chapter shows how the user can use the software The Geo-Transform.

Chapter 6: This chapter shows the conclusions and recommendations. For the project.

CHAPTER

2

COORDINATE SYSTEM

2.1 Introduction

2.2 Spherical Coordinate

2.3 Ellipsoidal Coordinate

2.4 Projected Coordinate

CHAPTER TWO

COORDINATE SYSTEMS

2.1 Introduction

Coordinate system is a system to determine location on the surface of the earth, different units and length to the angular distance have been used. The mathematical figure of the earth is applied to the classical definition of the geoid defined as equipotent surface of the earth gravitation field that nearly concides with the mean sea level(MSL).

A reference surface is chosen so that reductions are applied to the surface. At the beginning this surface was defenid as a sphere with a radius R (approximately $R=6378\text{km}$), later it was defiend as a rotational ellipsoid, the circle on the equator with Radius (a) and the distance from the center to the north or south pole is (b).(a) is called the semi major axis and (b) is called the semi minor axis, while always (a) is larger than (b).

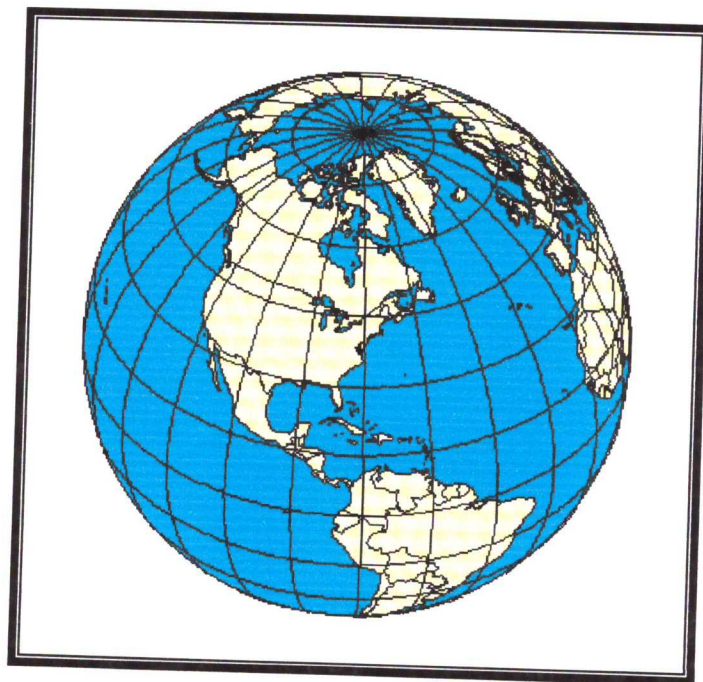


Figure 2.1: The Earth

2.2 Spherical coordinate

Considering the earth as a sphere with radius R , the position of the point is defined by polar coordinate (λ, ϕ, h) , or by geocentric coordinates (X, Y, Z) with origin at the earth center.

Where;

λ = longitude; defined as the angular distance on the equator from Greenwich meridian to the local meridian.

ϕ = latitude; defined as the angular distance long the meridian from the equator to the point.

h = ellipsoid height; defined as the distance along the normal from the ellipsoid surface to the point.

The conversion between the different coordinates can be calculated as follows:

$$X = (R + h) \cos\phi \cos\lambda \quad (2.1)$$

$$Y = (R + h) \cos\phi \sin\lambda \quad (2.2)$$

$$Z = (R + h) \sin\phi \quad (2.3)$$

The reverse formula are:

$$r = \sqrt{X^2 + Y^2 + Z^2} \quad (2.4)$$

$$h = r - R \quad (2.5)$$

$$\phi = \tan^{-1} \frac{Z}{\sqrt{X^2 + Y^2}} \quad (2.6)$$

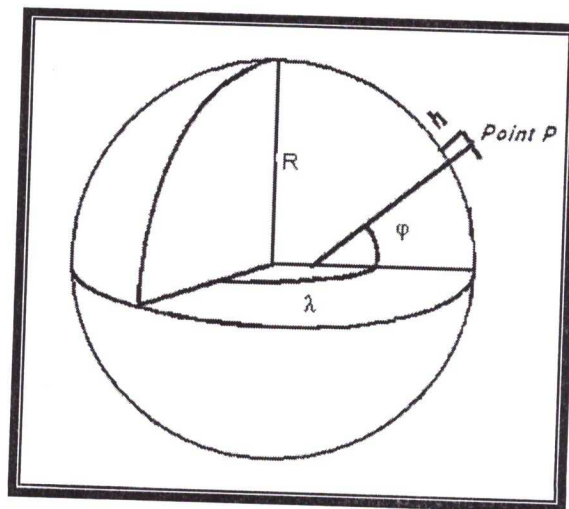


Figure (2.2): Sphere as the earth figure

2.3 Ellipsoidal Coordinates

2.3.1 Ellipsoidal earth figure

In classic definitions; earth is considered to be an ellipsoid with semi major axis (a) & semi minor axis (b), the other basic parameters can be calculated using the basic axis, table (2.1) show selected reference ellipsoids. Table (2-2), show the basic defining parameters of an ellipsoid.

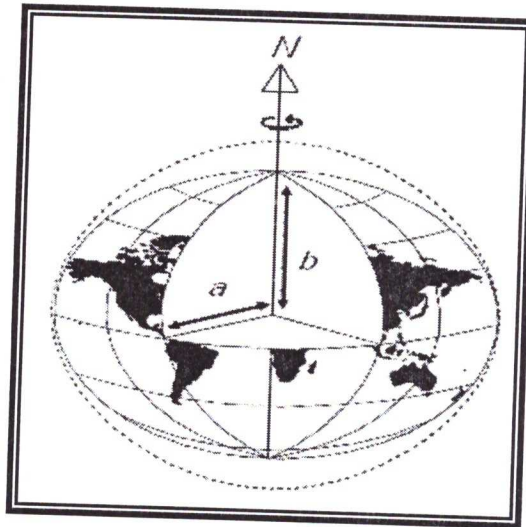


Figure (2.3): Ellipsoid as the figure of the earth

Table (2-1): Selected reference Ellipsoid [6]

Ellipse	Semi- Major axis (meter)	1/Flattening
WGS 60	6378165.0	298.3
WGS 66	6378145.0	298.25
WGS 72	6378135.0	298.26
WGS 84	6378137.0	298.257223563
South American 1969	6378160.0	298.25
Krassovsky	6378245.0	298.3
International	6378388.0	297.0
Clarck 1866	6378206.4	294.978698

Clark 1880	6378249.145	293.465
GRS 1975	6378140.0	298.257
GRS 1980	6378137.0	298.2572221

Table (2-2): The relation between the different parameters [6]

Notation	a/b	e^2	e'^2	f	n
$a =$					$\frac{c(1-n)}{1+n}$
$b =$		$a(1-e^2)^{1/2}$	$\frac{a}{(1+e'^2)^{1/2}}$	$a(1-f)$	
$\frac{a}{b} =$				$\frac{1}{1-f}$	
$\frac{b}{a} =$		$(1-e^2)^{1/2}$	$\frac{1}{(1+e'^2)^{1/2}}$	$(1-f)$	$\frac{1-n}{1+n}$
$c =$	$\frac{a^2}{b}$	$\frac{a}{(1-e^2)^{1/2}}$	$a(1+e'^2)^{1/2}$	$\frac{a}{1-f}$	
$e^2 =$	$\frac{a^2 - b^2}{a^2}$		$\frac{e'^2}{1+e'^2}$	$f(2-f)$	$\frac{4n}{(1+n)^2}$
$e'^2 =$	$\frac{a^2 - b^2}{b^2}$	$\frac{e^2}{(1-f)^2} = \frac{e^2}{1-e^2}$	$\frac{f(2-f)}{(1-f)^2}$		$\frac{4n}{(1-n)^2}$
$f =$	$\frac{a-b}{a}$	$1 - (1-e^2)^{1/2}$	$1 - (1+e'^2)^{-1/2}$		$\frac{2n}{1+n}$
$n =$	$\frac{a-b}{a+b}$	$\frac{1 - (1-e^2)^{1/2}}{1 + (1-e^2)^{1/2}}$	$\frac{(1+e'^2)^{1/2} - 1}{(1+e'^2)^{1/2} + 1}$	$\frac{f}{2-f}$	

In modern definition the ellipsoid is defined as an equipotential surface. the normal potential (U) on the reference ellipsoid is equal to the geopotential (W) on the geoid. The total mass of the reference ellipsoid is equal to that of the earth, and the

reference ellipsoid is rotating around its minor axis at the same angular velocity as the earth rotation.

Many ellipsoids were defined by physical definition depending on the principle of normal gravity. Examples of physically defined ellipsoids are GRS67, GRS80, and WGS84. The defining parameters for GRS80 ellipsoid are shown in table (2.3).

Table (2.3) GRS80 ellipsoid parameters [2]

Notation	Constant	Unit	Numerical value
a	Semi-major axis	m	6378137.000
GM	Product of G and total mass M	$\text{m}^3 \text{s}^{-2}$	$0.3986005 \cdot 10^{15}$
J2	Dynamic form factor $\frac{C-A}{Ma^2}$		0.00108263
ω	Angular velocity	s^{-1}	$0.72921151 \cdot 10^{-4}$
b	Semi-minor axis	meter	6356752.3141
f	Geometrical flattening		0.003352810681 $1/298.257222101$
e^2	First eccentricity squared		0.006694380023
$e^{2'}$	Second eccentricity squared	sec^{-1}	0.006739496775
U_0	Normal potential on the ellipsoid	$\text{m}^2 \text{sec}^2$	62636860.850
γ_p	Normal gravity on the poles	Gal	983.21863685
γ_e	Normal gravity on the equator	Gal	978.03267715
f^*	Gravity flattening		0.005302440112 $1/188.592417552$
k	$(b\gamma_p - a\gamma_e)/(a\gamma_e)$		0.001931851353
m	$\omega^2 a^2 b / (GM)$		0.003449786003
γ_{45}	Normal gravity at altitude 45°	Gal	980.6199203
$\bar{\gamma}$		Gal	979.7644656

The defining four parameters are shown in table (2.4) for WGS.

Table (2-4) WGS ellipsoid parameters [8]

Parameter	Notation	Value
Semi-major axis	a	6378137.0 meters
Reciprocal of Flattening	$1/f$	298.257223563
Angular Velocity of the Earth	ω	$7292115.0 \times 10 \text{ rad/s}$
Earth's Gravitational constant	GM	$3986004.418 \times 10 \text{ m}^3/\text{s}^2$

2.3.2 Geographic Coordinate

The most commonly used coordinate system today is the latitude, longitude & height. Equator & the prime meridian (Greenwich) are the reference planes used to define latitude and longitude.

The geodetic latitude (ϕ) of a point is the angle from the equatorial plane to the vertical direction of a line normal to the reference ellipsoid passing the point, the geodetic longitude (λ) of a point is the angle between a reference meridian (Greenwich) and the vertical plane passing through the point measure along the equator, both plans being perpendicular to the equatorial plane. The geodetic (ellipsoid or normal) height (h) at a point is the distance from the reference ellipsoid to the point in the direction normal to the ellipsoid. [5]

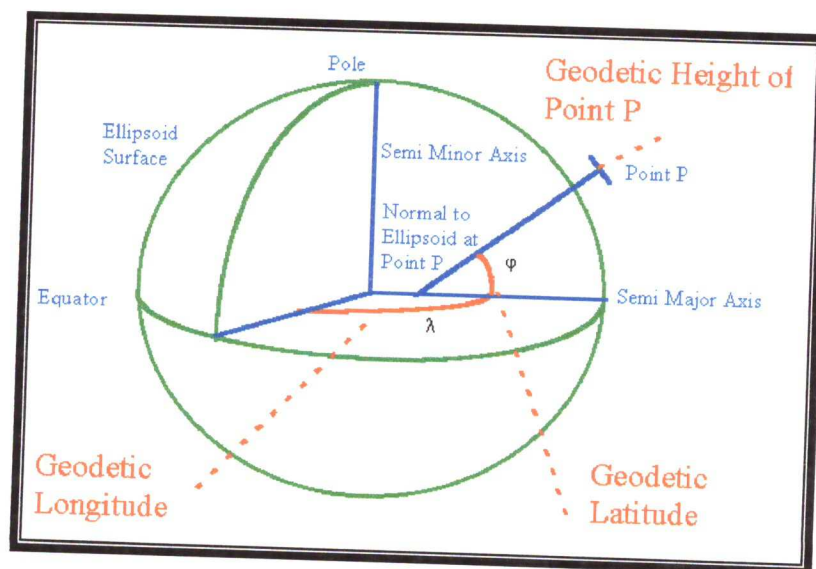


Figure (2.4): Geographic coordinates system

2. 3. 3 Geocentric coordinates

It is a system of three deminsional earth centered refrence system in which locations are identified by their X, Y, and Z values.

The x axis is in the equatorial plane of intersects the prime meridian (Greenwich).

The y axis is in the equatorial plane of intersects the $+90^\circ$ meridian.

The z axis coincides with the polar axis and is positive toward the North Pole.

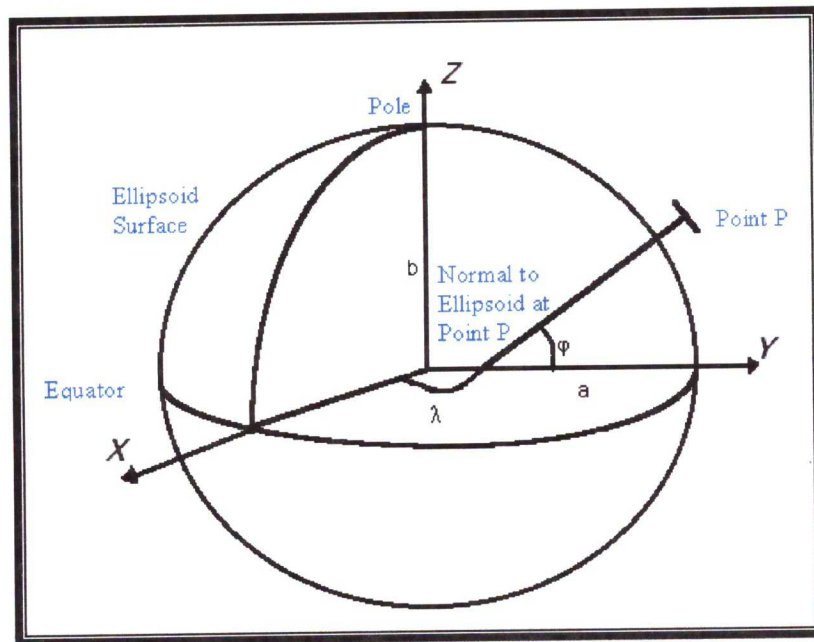


Figure (2.5) : Geocentric system

This system is not a Grid system. The earth is modeled as a sphere or spheroid in a right handed X, Y, Z system

$$X = (N + h) \cos \varphi \cos \lambda \quad (2.7)$$

$$Y = (N + h) \cos \varphi \sin \lambda \quad (2.8)$$

$$Z = ((1 - e^2)N + h) \sin \varphi \quad (2.9)$$

The inverse problem is solved in an iterative solution (Torge method): [9]

$$\lambda = \tan^{-1} \frac{Y}{X} \quad (\text{Does not need iteration}). \quad (2.10)$$

$$h = \frac{\sqrt{X^2 + Y^2}}{\cos \varphi} - N \quad (2.11)$$

$$\varphi = \tan^{-1} \left(\frac{Z}{\sqrt{X^2 + Y^2}} (1 - e^2 \frac{N}{N+h})^{-1} \right) \quad (2.12)$$

As initial value to start the iterative solution,

$$\varphi = \tan^{-1} \left(\frac{Z}{\sqrt{X^2 + Y^2}} (1 - e^2)^{-1} \right) \quad (2.13)$$

2. 3.4 Topocentric Coordinate

In the topocentric coordinates, we use the point of origin with know geographic coordinate $P_0 (\lambda, \varphi, h)$ or (X, Y, Z) , the(x) direction in to the north, the(y) direction is to the east and the (z) direction is perpendicular the x, y plane.

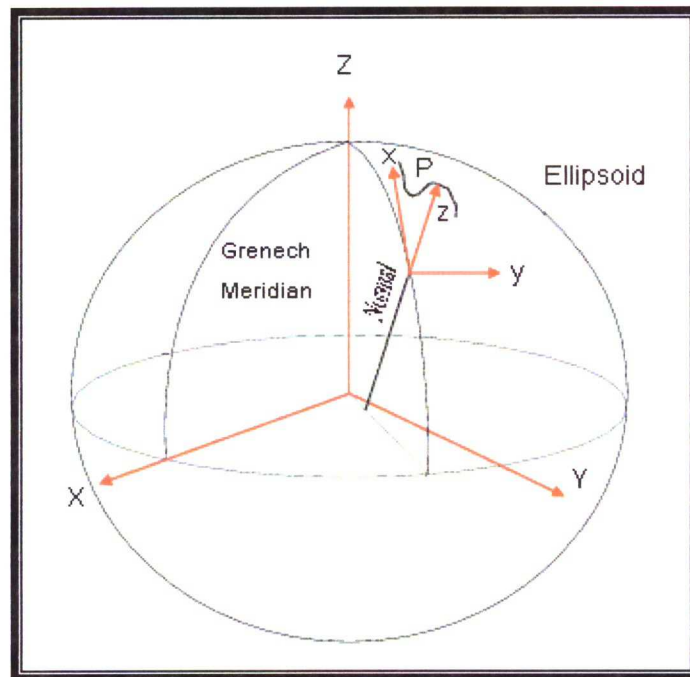


Figure (2.6): Topocentric Coordinates system

The position of the point is defined by the zenith (ze), distance (s) and Azimuth (Az) measured clockwise from the north,

Where;

$$x = S \cos Az \sin ze \quad (2.14)$$

$$y = S \sin Az \sin ze \quad (2.15)$$

$$z = S \cos ze \quad (2.16)$$

If geocentric coordinates are used:

$$X = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \quad x = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (2.17)$$

To convert from topocentric to geocentric coordinate the following can be applied in matrix form.

$$\Delta X = Ax \quad (2.18)$$

$$\begin{bmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{bmatrix} = \begin{bmatrix} -\sin \varphi_0 \cos \lambda_0 & -\sin \lambda_0 & \cos \varphi_0 \cos \lambda_0 \\ -\sin \varphi_0 \sin \lambda_0 & \cos \lambda_0 & \cos \varphi_0 \sin \lambda_0 \\ \cos \varphi_0 & 0 & \sin \varphi_0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (2.19)$$

$$X = X_{po} + \Delta X \quad (2.20)$$

$$x = A^{-1} \Delta X = A^T \Delta X$$

2.4 Map Projection (Grid coordinates systems)

The basic idea at map projection is to convert from geographic (φ, λ) to Local (E,N)Grid system and in the opposite direction, Map projections are attempts to portray the surface of the earth or a portion of the earth on a flat surface. Some distortions of conformality, distance, direction, scale, and area always result from this process. Some projections minimize distortions in some of these properties at the expense of maximizing errors in others. Some projections are attempts to only moderately distort all of these properties, the properties are:

1. **Conformality**: When the scale of a map at any point on the map is the same in any direction, the projection is conformal. Meridians (lines of longitude) and parallels (lines of latitude) intersect at right angles. Shape is preserved locally on conformal maps.
2. **Distance**: A map is equidistant when it portrays distances from the center of the projection to any other place on the map.
3. **Direction**: A map preserves direction when azimuths (angles from a point on a line to another point) are portrayed correctly in all directions.
4. **Scale**: **Scale** is the relationship between a distance portrayed on a map and the same distance on the Earth.
5. **Area**: When a map portrays areas over the entire map so that all mapped areas have the same proportional relationship to the areas on the Earth that they represent, the map is an equal-area map.

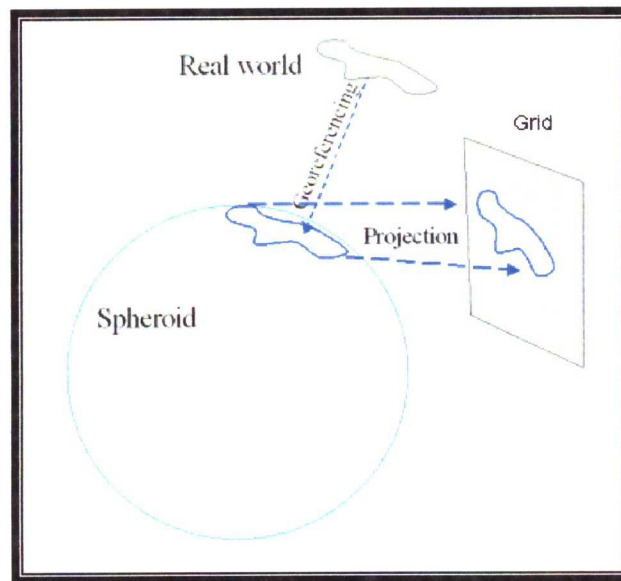


Figure (2.7): Map projection (from 3D to Grid system)

2.4.1 Map projections general classes.

Map projections fall into four general classes depend on the methods of projection,

1. **Cylindrical projections** result from projecting a spherical surface onto a cylinder.

- A. When the cylinder is **tangent** to the sphere contact is along a great circle (the circle formed on the surface of the Earth by a plane passing through the center of the Earth).

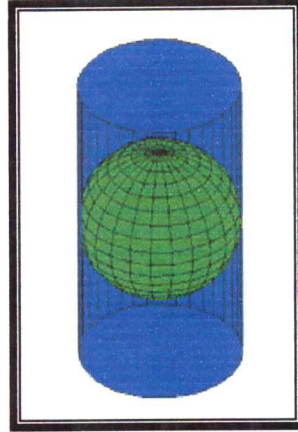


Figure (2.8) Projection of a sphere onto a cylinder (tangent case)

In the **secant** case, the cylinder touches the sphere along two lines.

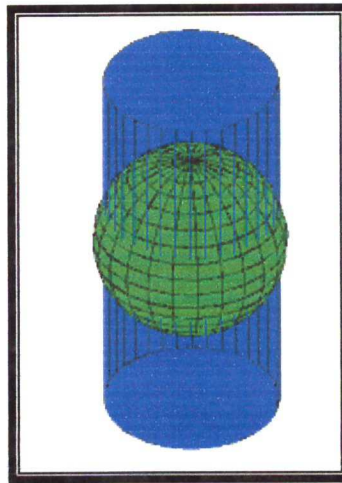


Figure (2.9) : Projection of a Sphere onto a Cylinder (Secant Case)

- B. When the cylinder upon which the sphere is projected is at right angles to the poles, the cylinder and resulting projection are **transverse**.

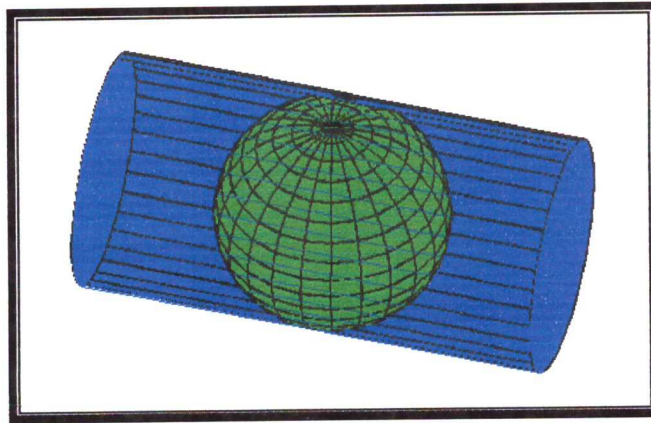


Figure (2.10) : Transverse Projection of a Sphere onto a Cylinder (Tangent Case)

- C. When the cylinder is at some other, **non-orthogonal**, angle with respect to the poles, the cylinder and resulting projection is **oblique**.

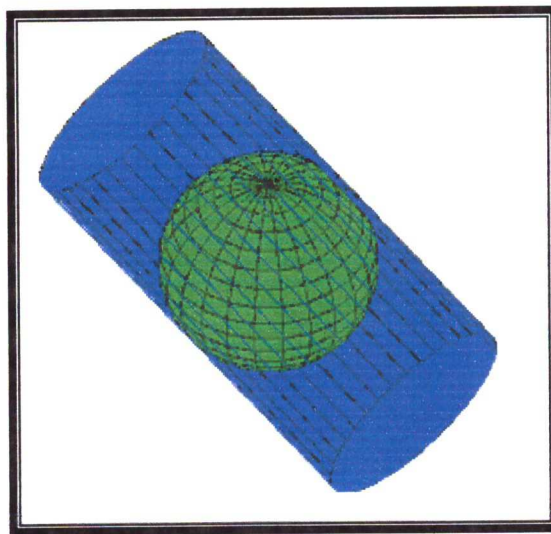


Figure (2.11) : Oblique Projection of a Sphere onto a Cylinder (Tangent Case)

2. **Conic projections** result from projecting a spherical surface onto a cone.

- A. When the cone is **tangent** to the sphere contact is along a small circle.

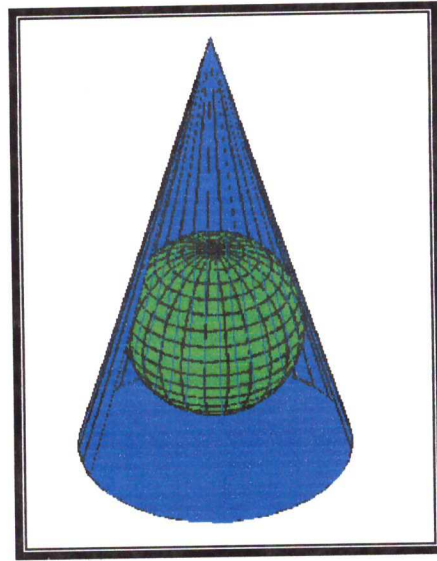


Figure (2.12): Projection of a Sphere onto a Cone (Tangent Case)

- B. In the **secant** case, the cone touches the sphere along two lines, one a great circle, the other a small circle.

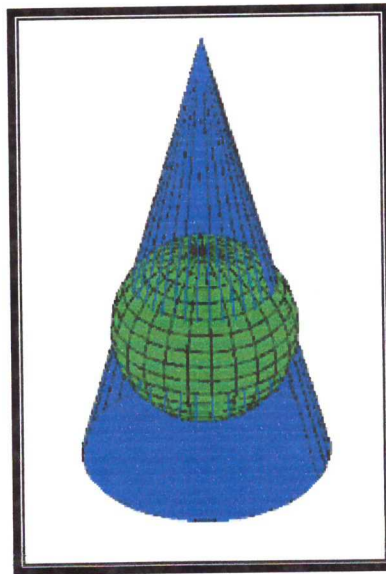


Figure (2.13) : Projection of a Sphere onto a Cone (Secant Case)

3. **Azimuthal projections** result from projecting a spherical surface onto a plane.

- A. When the plane is **tangent** to the sphere contact is at a single point on the surface of the Earth.

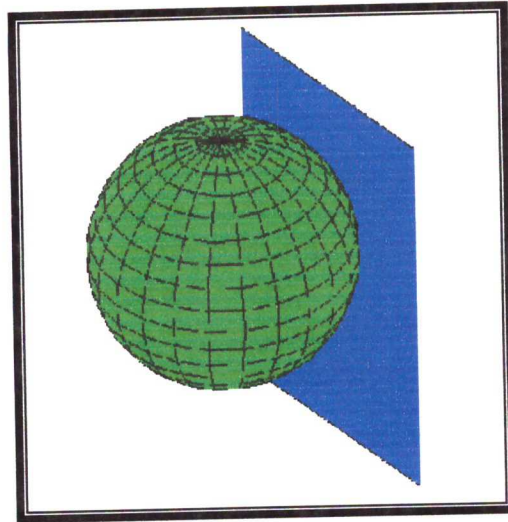


Figure (2.14): Projection of a Sphere onto a Plane (Tangent Case)

- B. In the **secant** case, the plane touches the sphere along a small circle if the plane does not pass through the center of the earth, when it will touch along a great circle.

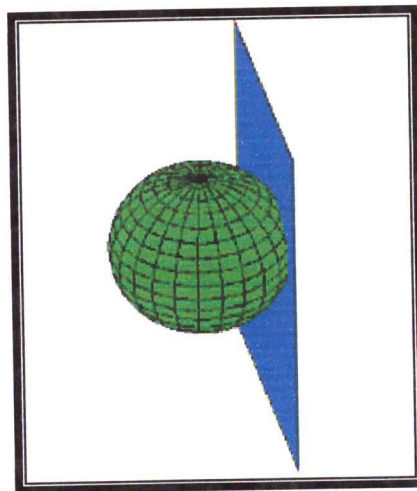


Figure (2-15): Projection of a Sphere onto a Plane (Secant Case)

4. **Miscellaneous projections** include unprojected ones such as rectangular latitude and longitude grids and other examples of that do not fall into the cylindrical, conic, or azimuthal categories

2.4.2 Transverse Mercator

It is known as Gauss-Kruger and similar to the Mercator, except that the cylinder is longitudinal along a meridian instead of the equator. The result is a conformal projection that does not maintain true directions exclude small areas. The central meridian is placed in the center of the region of interest. This centering minimizes distortion of all properties in that region. This projection is best suited for north-south shaped areas.

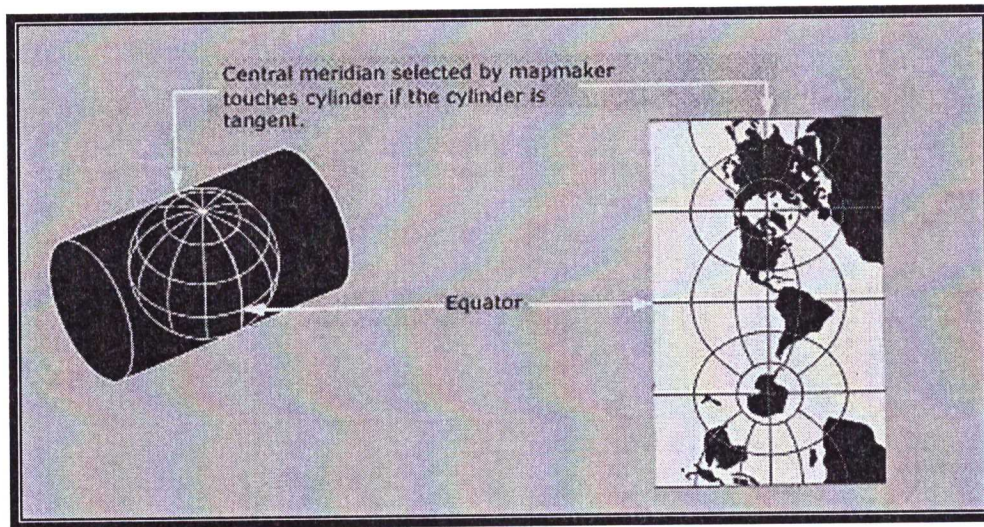


Figure (2.16): Transverse Mercator

The formulas to derive the projected Easting and Northing coordinates are in the form of a series as follows:

Easting:

$$E = FE + k_0 v \left[A + (1 - T + C) \frac{A^3}{6} + (5 - 18T + T^2 + 72C - 58e^2) \frac{A^5}{120} \right] \quad (2.21)$$

Northing:

$$N = FN + k_0 \left\{ M - M_0 + v \tan \varphi \left[\frac{A^2}{2} + (5 - T + 9C + 4C^2) \frac{A^4}{24} + (61 - 58T + T^2 + 600C - 330e^2) \frac{A^6}{720} \right] \right\} \quad (2.22)$$

Where;

E : Easting

N : Northing

FE : False easting

λ : longitude

φ : latitude

k : scale factor

$$T = \tan^2 \varphi \quad (2.23)$$

$$C = \frac{e^2}{1-e^2} \cos^2 \varphi = e'^2 \cos^2 \varphi \quad (2.24)$$

$$A = (\lambda - \lambda_0) \cos \varphi, \text{ with } \lambda \text{ and } \lambda_0 \text{ in radians}$$

M: arc meridian

$$M = a \cdot \begin{bmatrix} (1 - e^2/4 - 3e^4/64 - 5e^6/256 - \dots) \cdot \varphi \\ - (3e^2/8 + 3e^4/32 + 45e^6/1024 + \dots) \sin 2\varphi \\ + (15e^4/256 + 45e^6/1024 + \dots) \sin 4\varphi \\ - (35e^6/3072 + \dots) \sin 6\varphi + \dots \end{bmatrix} \quad (2.25)$$

v: radius of curvature in the prime vertical

p: radius of curvature in the meridian

With φ in radians and M for φ , the latitude of the origin, derived in the same way.

The reverse formulas to convert Easting and Northing projected coordinates to latitude and longitude are:

$$\varphi = \varphi_1 - \frac{v_1 \tan \varphi_1}{\rho_1} \left[\frac{D^2}{2} - (5 + 3T_1 + 10C_1 - 4C_1^2 - 9e^2) \frac{D^4}{24} + (61 + 90T_1 + 298C_1 + 45T_1^2 - 252e^2 - 3C_1^2) \frac{D^6}{720} \right] \quad (2.26)$$

$$\lambda = \lambda_0 + \left[\frac{D - (1 + 2T_1 + C_1) \frac{D^3}{6}}{(5 - 2C_1 + 28T_1 - 3C_1^2 + 8e^2 + 24T_1^2) \frac{D^5}{120}} \right] / \cos \varphi_1 \quad (2.27)$$

Where φ_1 may be found as for the Cassini projection from:

$$v_1 = \frac{a}{\sqrt{1 - e^2 \sin^2 \varphi_1}} \quad (2.28)$$

$$\rho_1 = \frac{a(1 - e^2)}{(1 - e^2 \sin^2 \varphi_1)^{3/2}} \quad (2.29)$$

$$\begin{aligned} \varphi_1 = & \mu_1 + (3e_1/2 - 27e_1^3/32 + \dots) \sin 2\mu_1 \\ & + (21e_1^2/16 - 55e_1^4/32 + \dots) \sin 4\mu_1 \\ & + (151e_1^3/96 + \dots) \sin 6\mu_1 \\ & + (1097e_1^4/512 - \dots) \sin 8\mu_1 + \dots \end{aligned} \quad (2.30)$$

And where:

$$e_1 = \frac{1 - (1 - e^2)^{1/2}}{1 + (1 - e^2)^{1/2}} \quad (2.31)$$

$$\mu_1 = \frac{M_1}{a(1 - e^2/4 - 3e^4/64 - 5e^6/256 - \dots)} \quad (2.32)$$

$$M_1 = M_0 + (N - FN)/k_0 \quad (2.33)$$

$$T_1 = \tan^2 \varphi_1 \quad (2.34)$$

$$C_1 = \frac{e^2}{1-e^2} \cos^2 \varphi = e'^2 \cos^2 \varphi \quad (2.35)$$

$$e'^2 = e^2 / [1 - e^2] \quad (2.36)$$

$$D = (E - FE) / (v_1 k_0) \quad (2.37)$$

To define coordinates system using transverse Mercator, the following parameters have to be defined reference ellipsoid

- False Easting
- False Northing
- Central Meridian
- Scale factor
- Latitude of origin
- Scale Factor at Central Meridian

In Palestine, there is a coordinates system named Palestinian transverse Mercator or Palestine-1923-Belt with the following parameters:

- False Easting : 170251.555000
- False Northing : 1126867.909000
- Central Meridian : 35.212081
- Scale factor : 1.000
- Latitude of origin : 31.734097
- Spheroid: Clarke_1880_Benoit
- Semi major Axis : 6378300.790000000
- Semi minor Axis: 6356566.430000036
- Inverse flattening: 293.466234570999970

The other common system is the Israeli traverse Mercator (Israeli TM Grid) with the following parameters:

- False Easting : 219529.584000
- False Northing : 626907.390000

- Central Meridian : 35.204517
- Scale factor : 1.000007
- Latitude of origin : 31.734394
- Spheroid: GRS_1980
- Semi major Axis : 6378300.79000000
- Semi minor Axis: 6356566.430000036
- Inverse flattening: 293.466234570999970

2.4.3 Cassini_Soldner projection

The name Cassini-Soldner refers to the more accurate ellipsoidal version, developed in the 19th century. This transverse cylindrical projection maintains scale along the central meridian and all lines parallel to it and is neither equal area nor conformal. It is most suited for large scale mapping of areas predominantly north-south in extent.

The formula to derive projected Easting and Northing coordinates are:

Easting:

$$E = FE + \nu \left[A - T.A^3 / 6 - (8 - T + 8C)T.A^3 / 120 \right] \quad (2.38)$$

Northing:

$$N = FN + M - M_0 + \nu . \tan \varphi \left[A^2 / 2 + (5 - T + 6C)A^4 / 24 \right] \quad (2.39)$$

Where;

FN: False Northing

FE : False Easting

φ :latitude

λ : longitude

λ_0 : longitude on the central maridian

$$A = (\lambda - \lambda_0) . \cos \varphi \quad (2.40)$$

$$T = \tan^2 \varphi \quad (2.41)$$

$$C = \frac{e^2}{1-e^2} \cos^2 \varphi \quad (2.42)$$

And M, the distance along the meridian from equator to latitude φ (Arc Meridian), is given by:

$$M = a \left[\begin{aligned} &(1 - e^2/4 - 3e^4/64 - 5e^6/256 - \dots) \cdot \varphi \\ &- (3e^2/8 + 3e^4/32 + 45e^6/1024 + \dots) \sin 2\varphi \\ &+ (15e^4/256 + 45e^6/1024 + \dots) \sin 4\varphi \\ &- (35e^6/3072 + \dots) \sin 6\varphi + \dots \end{aligned} \right] \quad (2.43)$$

With φ in radians, M_0 is the value of M calculated for the latitude of the chosen origin. This may not necessarily be chosen as the equator.

To compute latitude and longitude from Easting and Northing the reverse formula are:

$$\varphi = \varphi_1 - \frac{\nu_1 \tan \varphi_1}{\rho_1} \left[\frac{D^2}{2} - (1 + 3T_1) \frac{D^4}{24} \right] \quad (2.44)$$

$$\lambda = \lambda_0 + [D - T_1 \cdot D^3/3 + (1 + 3T_1)T_1 D^5/15] / \cos \varphi_1 \quad (2.45)$$

Where ρ_1 is ρ calculated at $\varphi = \varphi_1$, and φ_1 is the latitude of the point on the central meridian which has the same northing as the point whose coordinates are sought and is found from:

$$\nu_1 = \frac{a}{\sqrt{1 - e^2 \sin^2 \varphi_1}} \quad (2.46)$$

$$\rho_1 = \frac{a(1 - e^2)}{(1 - e^2 \sin^2 \varphi_1)^{3/2}} \quad (2.47)$$

$$\begin{aligned}
\varphi_1 = & \mu_1 + (3e_1/2 - 27e_1^3/32 + \dots)\sin 2\mu_1 \\
& + (21e_1^2/16 - 55e_1^4/32 + \dots)\sin 4\mu_1 \\
& + (151e_1^3/96 + \dots)\sin 6\mu_1 \\
& + (1097e_1^4/512 - \dots)\sin 8\mu_1 + \dots
\end{aligned} \tag{2.48}$$

Where;

$$e_1 = \frac{1 - (1 - e^2)^{1/2}}{1 + (1 - e^2)^{1/2}} \tag{2.49}$$

$$\mu_1 = \frac{M_1}{a(1 - e^2/4 - 3e^4/64 - 5e^6/256 - \dots)} \tag{2.50}$$

$$M_1 = M_0 + (N - FN) \tag{2.51}$$

$$T_1 = \tan^2 \varphi_1 \tag{2.52}$$

$$D = (E - FE)/v_1 \tag{2.53}$$

To define a coordinates system using Cassini projection the following parameters are to be considered Reference ellipsoid

- False Easting
- False Northing
- Central Meridian
- Scale factor =1
- Latitude of origin

The Palestinian grid named Palestine 1923 Grid is built using Cassini projection, which normally used in land surveying and engineering projects with the following parameters:

- False Easting : 170251.555000
- False Northing : 1126867.909000
- Central Meridian : 35.212081
- Scale factor : 1.000

- Latitude of origin : 31.734097
- Spheroid: Clarke_1880_Benoit
- Semi major Axis : 6378300.79000000
- Semi minor Axis: 6356566.430000036
- Inverse flattening: 293.466234570999970

Israeli old grid is the same of Palestine grid (Palestine-1923-Israili-Grid), but 1 million is added to the northing value:

- False Easting : 170251.555000
- False Northing : 1126867.909000
- Central Meridian : 35.212081
- Scale factor : 1.000
- Latitude of origin : 31.734097
- Spheroid: Clarke_1880_Benoit
- Semi major Axis : 6378300.79000000
- Semi minor Axis: 6356566.430000036
- Inverse flattening: 293.466234570999970

2.4.5 (UTM) Universal Transverse Mercator

UTM coordinate system is used in survey navigation and in GIS. And it's the most commonly used in transverse mercator. from the figure (2-17) below, we can see that UTM zone numbers designate 6 degree longitudinal strips extending from 80 degree south latitude to 84 degrees north longitude.

To find the central meridian of a UTM zone: [2]

$$\text{Central meridian} = (\text{zone\#} * 6 - 3) - 180 \quad (2.54)$$

To find which zone you belong to at a given longitude: [2]

$$\text{zone} = \text{int}\left(\frac{\lambda + 180}{6}\right) + 1 \quad (2.55)$$

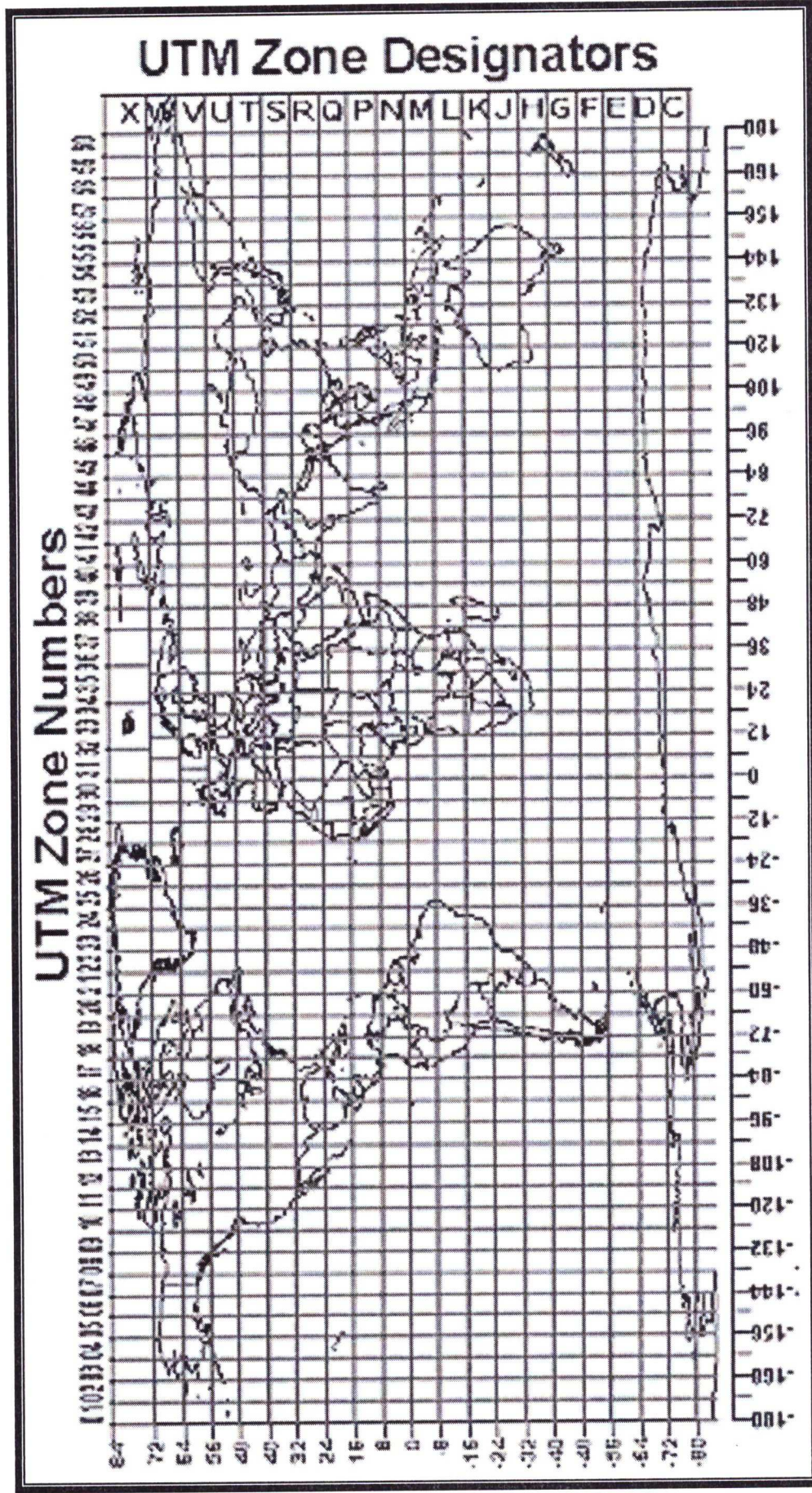


Figure (2-17) Universal Transverse Mercator

2.4.6 Projected coordinate

A projected coordinate system is defined on a flat, two dimensional surfaces, it has a constant lengths angle & areas across the two dimensions, and it's always based on a geographic coordinate system that is based on a sphere or spheroid

In a projected coordinate system, locations are identified by x, y coordinate on a grid, with the origin at the center of the grid. Each position has two values. One is horizontal position & the other is vertical position.

Table (2.5): Projected Coordinate Systems Examples [2]

Projected Coordinate Systems Examples							
<u>Name</u>	<u>Projection</u>	<u>Geographic Coordinate System (GCS)</u>	<u>Central Meridian</u>	<u>Central Parallel</u>	<u>Scale Factor</u>	<u>F. Easting</u>	<u>F. Northing</u>
Palestine Belt	Transverse Mercator	Palestine 1923	35.21206055556	31.73409694444	1	170,251.555	1,126,867.909
Palestine Grid	Cassini	Palestine 1923	35.21206055556	31.73409694444	1	170,251.555	126,867.909
Israel CS Grid	Cassini	Palestine 1923	35.21206055556	31.73409694444	1	170,251.555	1,126,867.909
Israel TM	Transverse Mercator	GCS 1980	35.20451694445	31.73439361	1.0000067	219,529.584	626,907.39
WGS 1984 UTM Zone 35N	Transverse Mercator	WGS 1984	27	0	0.9996	500,000	0 (the equator)
WGS 1984 UTM Zone 36N	Transverse Mercator	WGS 1984	33	0	0.9996	500,000	0 (the equator)
WGS 1984 UTM Zone 37N	Transverse Mercator	WGS 1984	39	0	0.9996	500,000	0 (the equator)
WGS 1984 UTM Zone 38N	Transverse Mercator	WGS 1984	45	0	0.9996	500,000	0 (the equator)
Egypt Blue Belt	Transverse Mercator	Egypt 1907	35	30	1	300,000	1,100,000
Egypt Red Belt	Transverse Mercator	Egypt 1907	31	30	1	615,000	810,000
Egypt Purple Belt	Transverse Mercator	Egypt 1907	27	30	1	700,000	200,000
Egypt Extended Purple Belt	Transverse Mercator	Egypt 1907	27	30	1	700,000	1,200,000
ED 1950 UTM	Transverse	European 1950	27	0	0.9996	500,000	0 (the

Zone 35N	Mercator						equator)
ED 1950 UTM Zone 36N	Transverse Mercator	European 1950	33	0	0.9996	500,000	0 (the equator)
ED 1950 UTM Zone 37N	Transverse Mercator	European 1950	39	0	0.9996	500,000	0 (the equator)
ED 1950 UTM Zone 38N	Transverse Mercator	European 1950	45	0	0.9996	500,000	0 (the equator)
ED 1950 UTM Zone 35N (Egypt)	Transverse Mercator	European 1950 (Egypt)	27	0	0.9996	500,000	0 (the equator)
ED 1950 UTM Zone 36N (Egypt and Jordan 1:50k English series topo maps)	Transverse Mercator	European 1950 (Egypt)	33	0	0.9996	500,000	0 (the equator)
ED 1950 UTM Zone 37N (Egypt and Jordan 1:50k English series topo maps)	Transverse Mercator	European 1950 (Egypt)	39	0	0.9996	500,000	0 (the equator)
ED 1950 UTM Zone 36N (Cyprus)	Transverse Mercator	European 1950 (Cyprus)	33	0	0.9996	500,000	0 (the equator)
ED 1950 UTM Zone 36N (Middle East)	Transverse Mercator	European 1950 (Middle East)	33	0	0.9996	500,000	0 (the equator)
ED 1950 UTM Zone 37N (Middle East)	Transverse Mercator	European 1950 (Middle East)	39	0	0.9996	500,000	0 (the equator)
ED 1950 UTM Zone 38N (Middle East)	Transverse Mercator	European 1950 (Middle East)	45	0	0.9996	500,000	0 (the equator)
FD 1958 Iraq	Lambert Conic Conformal	GCS FD 1958	45	32.5	0.99878640 8	1,500,000	0 (the equator)
Levant (Syria)	Double Stereographic	Deir-ez-Zor	37.35	34.65	0.9996256	0	0 (the equator)
Syria Lambert	Double Stereographic	Deir-ez-Zor	37.35	34.65	0.9996256	300,000	300,000
Jordan Transverse Mercator	Transverse Mercator	International (Hayford) 1927	37	0	0.9998	500,000	-3,000,000

CHAPTER

3

DATUM TRANSFORMATION

3.1 Introduction

3.2 Datum & Ellipsoids

3.3 Datum Transformations

CHAPTER THREE

DATUM TRANSFORMATION

3.1 Introduction

The coordinates of all locations on the earth are defined referring to a datum. While a spheroid nearly represents the shape of the earth, a datum defines the position of a spheroid relative to the center of the earth. A point on the surface of the earth is matched to a particular position on the surface of the ellipsoid. This point is known as the origin point of the coordinates system on the datum. The coordinates of the origin point of coordinates system are fixed, and all other points are calculated referring to it. The coordinate system origin of a local datum is not at the center of the earth. The center of the spheroid of a local datum is offset from the earth's center, depending on a global datum like WGS84.

A datum provides a frame or reference for measuring locations on the surface of the earth. It defines the origin and orientation of latitude and longitude lines. Whenever change the datum, or more correctly, the geographic coordinate system, the coordinate values of a point will change.

3.2 Datum and Ellipsoids

The shape of the earth is ellipsoid because the distance from the center of the earth to the equator is larger than the distance from the center to the poles by about 23 km. to make an ellipsoid model of the earth, Rotate the ellipse about the shorter polar axis (semi-minor axis b) to form a solid surface, see figure (3.1) A datum is defined by choosing an ellipsoid and then a primary reference point.

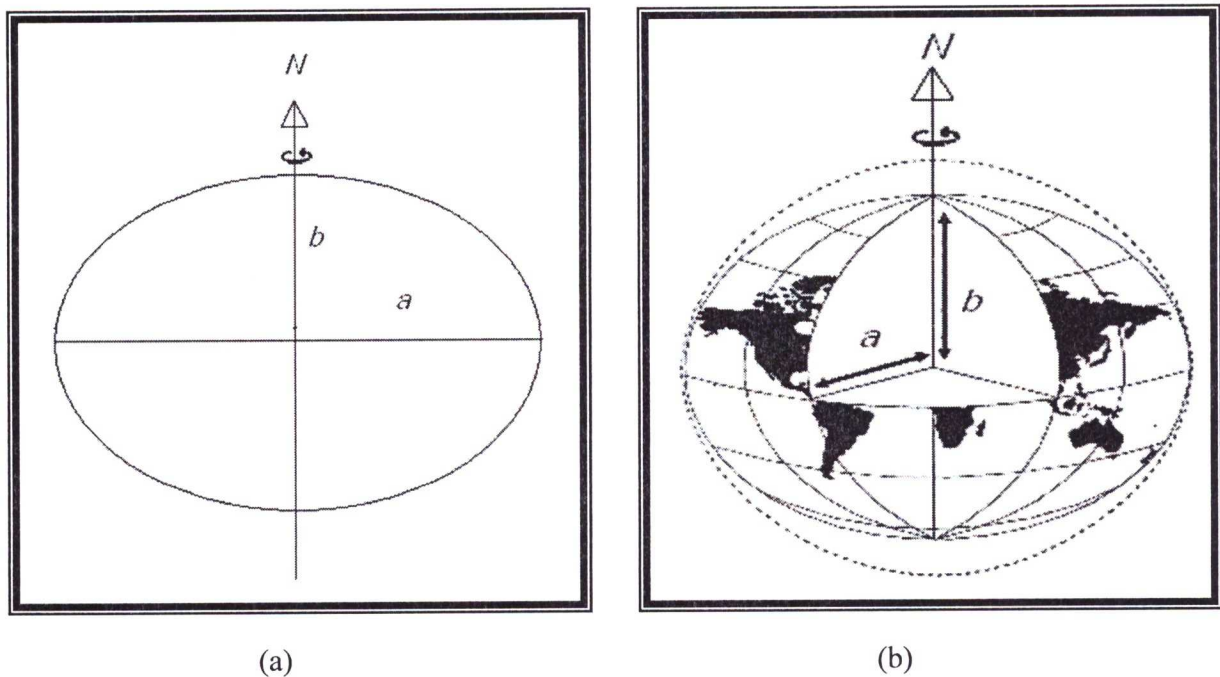


Figure (3.1) : (a) Rotate ellipse, (b) Making ellipsoid

The reference ellipsoid of the Palestine_1923_Grid, Palestine_1923_Israel_CS_Grid, and Palestine_1923_Palestine_Belt is the Clarke_1880_Benoit. The reference ellipsoid of the Israel_TM_Grid is the Geodetic Reference System of 1980 (GRS80). The following table (3.1) shows the coordinate systems with their datum definition:

Table (3.1) : The coordinate system with datum

Datum	Area	Name of Point	Latitude	Longitude	Ellipsoid
North American 1927	North America	Meades Ranch	39 13 26.686 N	98 32 30.506 W	Clarke 1866
Old Hawaiian	Hawaii	Oahu West Base Astro	21 18 13.89 N	157 50 55.79 W	Clarke 1866
Qornog	Greenland	Station 7008	64 31 06.27 N	51 12 24.86 W	International
Hjorsey 1955	Iceland	Hjorsey	64 31 29.260 N	22 22 05.840 W	International
Provisional South American 1956	Venezuela, Ecuador, Peru, Bolivia, Chile	La Canoa	08 34 17.17 N	63 51 34.88 W	International
Corrego Alegre	Brazil	Corrego Alegre	19 50 15.14 S	48 57 42.75 W	International
Chua Astro	Paraguay	Chua Astro	19 45 41.16 S	48 06 07.56 W	International
Campo Inchauspe	Argentina	Campo Inchauspe	35 58 16.56 S	62 10 12.03 W	International
Yacare	Uruguay	Yacare	30 35 53.68 S	57 25 01.30 W	International
European	Europe	Potsdam, Helmertturm	52 22 51.446 N	13 03 58.741 E	International
Odnance Survey of Great Britain 1936	Great Britain: Northern Ireland	Royal Greenwich Observatory, Herstmonceux	50 51 55.271 N	00 20 45.882 E	Airy
Ireland 1965	Ireland	Royal Greenwich, Herstmonceux	50 51 55.271 N	00 20 45.882 E	Modified Airy (8)
Merchich	Morocco	Merchich	33 26 59.672 N	07 33 27.295 W	Clarke 1880 (2)
Voirol	Algeria	Voirol Observatory	36 45 07.9 N	03 02 49.45 E	Clarke 1880 (2)
Adindan	Sudan	Adindan	22 10 07.110 N	31 29 21.608 E	Clarke 1880 (2)
Sierra Leone 1960	Sierra Leone	D.O.S. Astro SLX2	08 27 17.6 N	12 49 40.2 W	Clarke 1880 (2)
Liberia 1964	Liberia	Robertsfield Astro	06 13 53.02 N	10 21 35.44 W	Clarke 1880 (2)
Ghana	Ghana	GCS Pillar 547 Accra	05 32 43.30 N	00 11 52.30 W	War Office (3)
Nigeria	Nigeria	Minna	09 39 08.87 N	06 30 58.76 E	Clarke 1880 (2)

Arc 1950	Africa (South of Equator)	Buffelsfontein	33 59 32.00 S	25 30 44.622 E	Clarke 1880 (2)
Tananarive (Antananarivo) Obsy 1925	Malagasy Rep.	Tananarive (Antananarivo Obsy)	18 55 02.10 S	47 33 06.75 E	International
World Geodetic System 1972	Sino-Soviet Bloc				World Geodetic System 1972
Herat North	Afghanistan	Herat North Astro	34 23 09.08 N	64 10 58.94 E	International
Indian	India, Pakistan, Burma, Thailand, Southeast Asia	Kalianpur Hill	24 07 11.26 N	77 39 17.57 E	Everest (5)
Tokyo	Japan	Tokyo Obsy	35 39 17.515 N	139 44 40.502 E	Bessel
Hu-Tzu-Shan	Taiwan	Hu-Tzu-Shan	23 58 32.340 N	120 58 25.975 E	International
Luzon	Philippines	Balanacan	13 33 41.000 N	121 52 03.000 E	Clarke 1866
Kertau	West Malaysia	Kertau	03 27 50.71 N	102 37 24.55 E	Modified Everest (6)
Timbalai	East Malaysia	Timbalai	05 17 03.548 N	115 10 56.409 E	Everest
Bukit Rimpah	Indonesia (Bangka)	Bukit Rimpah	02 00 40.16 S	105 51 39.76 E	Bessel
G. Serindung	Kalimantan	Ep. A	01 06 10.60 N	105 00 59.82 E	Bessel
G. Segara	Indonesia (Kalimantan, East)	G. Segara (P5)	00 32 12.83 S	117 08 48.47 E	Bessel
Montiong Lowe	Indonesia (Sulawesi)	Montiong Lowe (Pl)	05 08 41.42 S	119 24 14.94 E	Bessel
Australian Geodetic	Australia	Johnston Memorial Cairn	25 56 54.5515S	133 12 30.0771E	Australian National (7)
Geodetic Datum 1949	New Zealand	Papatahi Trig Station	41 19 08.900 S	175 02 51.000 E	International
Guam 1963	Marianas Islands	Tagcha	13 22 38.490 N	144 45 51.560 E	Clarke 1866
Local Astrol					World Geodetic System 1972

3.3 Datum Transformations

3.3.1 Introduction

Equation-based transformation methods can be classified into the following basic four methods. Having data in one datum and needing the coordinates in other is a common task in geodesy, surveying and GPS. A transformation must be used to display coordinates from a GPS receiver in any other datum than WGS84. Over any small area (150x150 km), the transformation will be a constant shift in latitude, longitude and height, neglecting scaling and rotation. But for countries with large areas seven parameter are preferred; scale, 3 rotations around the 3 axes and 3 translations.

There are three common methods of making these transformations from one datum to on another. These methods are 3D similarity, Helmert and Moldensky method, Figure (3.2) shows the basic parameters for datum transformation.

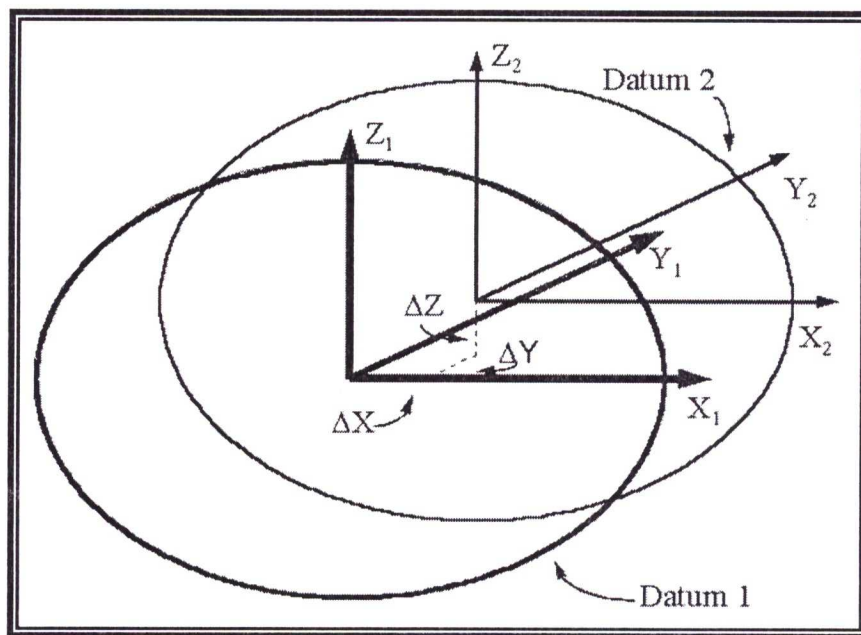


Figure 3.2: Datum transformation

3.3.2 Least square solution

To solve the datum transformation problem, control points with known positions in both datums have to be available. To solve the parameters, non linear least squares solution method is used with basic matrix equation. For (m) number of observation and (n) number of unknowns.

$$AX = L + V \quad (3.1)$$

Where A is the matrix of coefficients multiplied by the unknown parameters

$$A = \begin{bmatrix} a_{11} & a_{12} & \Lambda & a_{1n} \\ a_{21} & a_{22} & \Lambda & a_{2n} \\ M & M & M & M \\ a_{m1} & a_{m2} & \Lambda & a_{mn} \end{bmatrix} \quad (3.2)$$

And X is the matrix of unknowns

$$X = \begin{bmatrix} dx_1 \\ dx_2 \\ M \\ dx_n \end{bmatrix} \quad (3.3)$$

And L is observations matrix, which is the difference between the measured values and the computed ones using the initial values of the unknown parameters:

$$L = \begin{bmatrix} l_1 - l_{10} \\ l_2 - l_{20} \\ M \\ l_m - l_{m0} \end{bmatrix} \quad (3.4)$$

V is the residuals matrix;

$$V = \begin{bmatrix} v_1 \\ v_2 \\ M \\ v_m \end{bmatrix} \quad (3.5)$$

Multiply $AX = L + V$ on left by A^T to get:

$$A^T AX = A^T L \quad (3.6)$$

Where;

$$A^T A = \begin{bmatrix} a_{11} & a_{21} & \Lambda & a_{m1} \\ a_{12} & a_{22} & \Lambda & a_{m2} \\ M & M & M & M \\ a_{1n} & a_{2n} & \Lambda & a_{mn} \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & \Lambda & a_{1n} \\ a_{21} & a_{22} & \Lambda & a_{2n} \\ M & M & M & M \\ a_{m1} & a_{m2} & \Lambda & a_{mn} \end{bmatrix} = N \quad (3.7)$$

$$\begin{aligned} n_{11} &= \sum_{i=1}^m a_{i1}^2 & n_{12} &= \sum_{i=1}^m a_{i1} a_{i2} & \Lambda & n_{1n} &= \sum_{i=1}^m a_{i1} a_{in} \\ n_{21} &= \sum_{i=1}^m a_{i2} a_{i1} & n_{22} &= \sum_{i=1}^m a_{i2}^2 & \Lambda & n_{2n} &= \sum_{i=1}^m a_{i2} a_{in} \\ M & & M & & M & M & \\ n_{n1} &= \sum_{i=1}^m a_{in} a_{i1} & n_{n2} &= \sum_{i=1}^m a_{in} a_{i2} & \Lambda & n_{nn} &= \sum_{i=1}^m a_{in}^2 \end{aligned} \quad (3.8)$$

When observations are not weighted, then:

$$X = (A^T A)^{-1} A^T L \quad (3.9)$$

When weighted observation are used, then:

$$X = (A^T W A)^{-1} A^T W L \quad (3.10)$$

3.3.3 3D Similarity 7-Parameter

This method is more complex and accurate datum transformation. The 3D similarity 7-parameter has seven parameter transformations that include three translation parameters, three rotation parameters and a scale parameter.

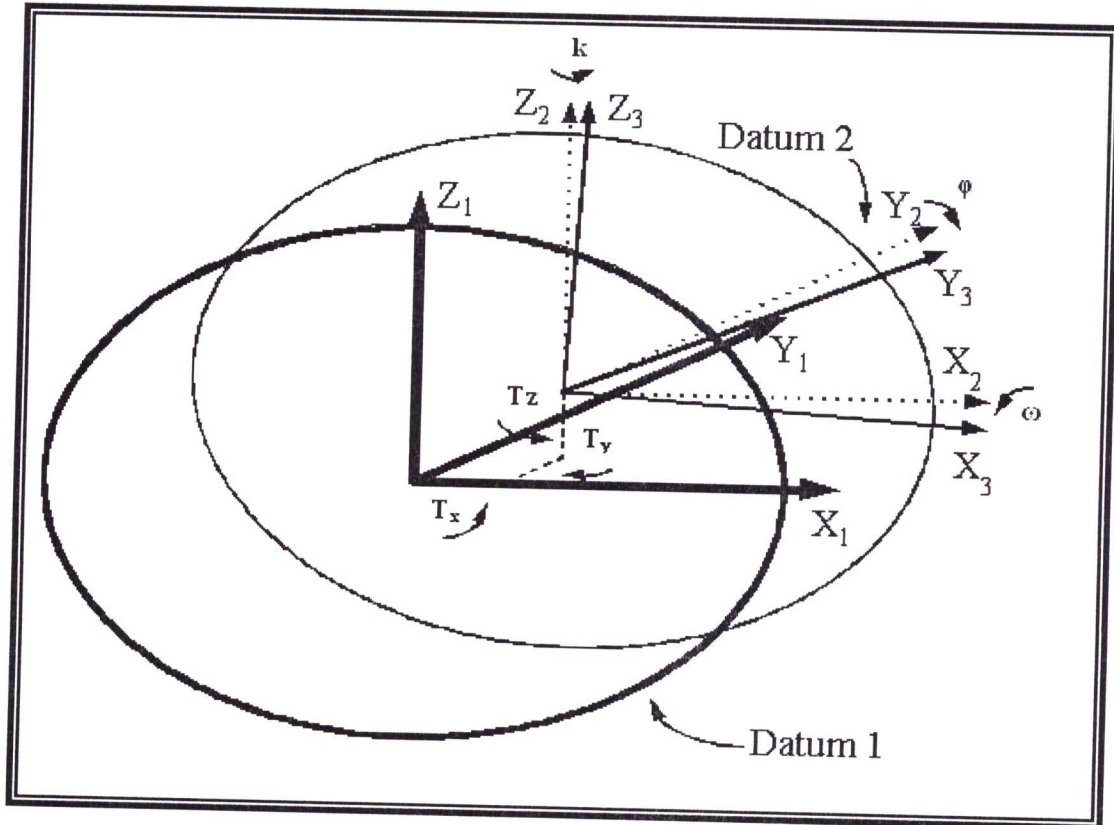


Figure (3.3) : 7-Parameter Transformation

Parameters are: S , ω , ϕ , k , T_x , T_y , and T_z the equations for the 3D similarity 7-parameter transformation are:

$$X = S(m_{11}x + m_{21}y + m_{31}z) + T_x \quad (3.11)$$

$$Y = S(m_{12}x + m_{22}y + m_{32}z) + T_y \quad (3.12)$$

$$Z = S(m_{13}x + m_{23}y + m_{33}z) + T_z \quad (3.13)$$

In matrix form:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = S \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (3.14)$$

$$X = S \cdot M \cdot x + T_x \quad (3.15)$$

$$Y = S \cdot M \cdot y + T_y \quad (3.16)$$

$$Z = S \cdot M \cdot z + T_z \quad (3.17)$$

Where;

X, Y, Z: point coordinations in the target system

x, y, z: point coordinations in the source

S: scale

M: rotation matrix

X, y, z: coordinate system in second datum

T: translation matrix

$$m_{11} = \cos \phi \cos \kappa \quad (3.18)$$

$$m_{12} = \sin \omega \sin \phi \cos \kappa + \cos \omega \sin \kappa \quad (3.19)$$

$$m_{13} = -\cos \omega \sin \phi \cos \kappa + \sin \omega \sin \kappa \quad (3.20)$$

$$m_{21} = -\cos \phi \sin \kappa \quad (3.21)$$

$$m_{22} = -\sin \omega \sin \phi \sin \kappa + \cos \omega \cos \kappa \quad (3.22)$$

$$m_{23} = \cos \omega \sin \phi \sin \kappa + \sin \omega \cos \kappa \quad (3.23)$$

$$m_{31} = \sin \phi \quad (3.24)$$

$$m_{32} = -\sin \omega \cos \phi \quad (3.25)$$

$$m_{33} = \cos \omega \cos \phi \quad (3.26)$$

In the system of the equations, seven-parameter require a minimum number of two horizontal control stations with known X-Y and x-y-z coordinates, in addition to three stations with known Z and x-y-z coordinates. If there are more than the minimum

number of observations, a least-squares solution can be applied. The equations are nonlinear with respect to the unknowns and must be linearized for a solution. The following linearized equations for a full control point (Horizontal and Vertical)

$$\begin{bmatrix} \left(\frac{\partial X}{\partial S}\right)_0 & 0 & \left(\frac{\partial X}{\partial \phi}\right)_0 & \left(\frac{\partial X}{\partial k}\right)_0 & 1 & 0 & 0 \\ \left(\frac{\partial Y}{\partial S}\right)_0 & \left(\frac{\partial Y}{\partial \omega}\right)_0 & \left(\frac{\partial Y}{\partial \phi}\right)_0 & \left(\frac{\partial Y}{\partial k}\right)_0 & 0 & 1 & 0 \\ \left(\frac{\partial Z}{\partial S}\right)_0 & \left(\frac{\partial Z}{\partial \omega}\right)_0 & \left(\frac{\partial Z}{\partial \phi}\right)_0 & \left(\frac{\partial Z}{\partial k}\right)_0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} dS \\ d\omega \\ d\phi \\ d\kappa \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} = \begin{bmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \quad (3.27)$$

Where;

$$\frac{\partial X}{\partial S} = m_{11}x + m_{21}y + m_{31}z \quad (3.28)$$

$$\frac{\partial Y}{\partial S} = m_{12}x + m_{22}y + m_{32}z \quad (3.29)$$

$$\frac{\partial Z}{\partial S} = m_{13}x + m_{23}y + m_{33}z \quad (3.30)$$

$$\frac{\partial Y}{\partial \omega} = -S(m_{13}x + m_{23}y + m_{33}z) \quad (3.31)$$

$$\frac{\partial Z}{\partial \omega} = S(m_{12}x + m_{22}y + m_{32}z) \quad (3.32)$$

$$\frac{\partial X}{\partial \phi} = S[-\sin(\phi)\cos(k)x + \sin(\phi)\sin(k)y + \cos(\phi)z] \quad (3.33)$$

$$\frac{\partial Y}{\partial \phi} = S[\sin(\omega)\cos(\phi)\cos(k)x - \sin(\omega)\cos(\phi)\sin(k)y + \sin(\omega)\sin(\phi)z] \quad (3.34)$$

$$\frac{\partial Z}{\partial \phi} = S[-\cos(\omega)\cos(\phi)\cos(k)x + \cos(\omega)\cos(\phi)\sin(k)y - \cos(\omega)\sin(\phi)z] \quad (3.35)$$

$$\frac{\partial X}{\partial k} = S(m_{21}x - m_{11}y) \quad (3.36)$$

$$\frac{\partial Y}{\partial k} = S(m_{22}x - m_{12}y) \quad (3.37)$$

$$\frac{\partial Z}{\partial k} = S(m_{23}x - m_{13}y) \quad (3.38)$$

The matrix solution is:

$$X = (A^T A)^{-1} A^T L$$

$$X = \begin{bmatrix} dS \\ d\omega \\ d\phi \\ d\kappa \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} \quad (3.39)$$

The seven-parameter are calculated after first iteration, an assigned as the initial value of second iteration; where:

$$S = S_o + dS \quad (3.40)$$

$$\omega = \omega_o + d\omega \quad (3.41)$$

$$\phi = \phi_o + d\phi \quad (3.42)$$

$$\kappa = \kappa_o + d\kappa \quad (3.43)$$

$$T_x = T_{x_o} + dT_x \quad (3.44)$$

$$T_y = T_{y_o} + dT_y \quad (3.45)$$

$$T_z = T_{z_o} + dT_z \quad (3.46)$$

The rotation matrix M for the transformation is:

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \quad (3.47)$$

To calculate the elements of last matrix we must give initial values for the parameters, using one point, normally, the following are useful.

$$S_o = 1$$

$$\omega_o = \phi_o = \kappa_o = 0$$

$$T_{x_o} = (X - x), \quad T_{y_o} = (Y - y), \quad T_{z_o} = (Z - z)$$

Then calculate the initial values of the observations.

$$X_o = (m_{11}x + m_{21}y + m_{31}z) + T_{x_o} \quad (3.48)$$

$$Y_o = (m_{12}x + m_{22}y + m_{32}z) + T_{y_o} \quad (3.49)$$

$$Z_o = (m_{13}x + m_{23}y + m_{33}z) + T_{z_o} \quad (3.50)$$

$$\begin{aligned} m_{11} &= \cos \phi \cos \kappa \\ &= \cos 0 \cos 0 = 1 \end{aligned} \quad (3.51)$$

$$\begin{aligned} m_{12} &= \sin \omega \sin \phi \cos \kappa + \cos \omega \sin \kappa \\ &= \sin 0 \sin 0 \cos 0 + \cos 0 \sin 0 = 0 \end{aligned} \quad (3.52)$$

$$\begin{aligned} m_{13} &= -\cos \omega \sin \phi \cos \kappa + \sin \omega \sin \kappa \\ &= -\cos 0 \sin 0 \cos 0 + \sin 0 \sin 0 = 0 \end{aligned} \quad (3.53)$$

$$\begin{aligned} m_{21} &= -\cos \phi \sin \kappa \\ &= -\cos 0 \sin 0 = 0 \end{aligned} \quad (3.54)$$

$$\begin{aligned} m_{22} &= -\sin \omega \sin \phi \sin \kappa + \cos \omega \cos \kappa \\ &= -\sin 0 \sin 0 \sin 0 + \cos 0 + \cos 0 = 1 \end{aligned} \quad (3.55)$$

$$\begin{aligned} m_{23} &= \cos \omega \sin \phi \sin \kappa + \sin \omega \cos \kappa \\ &= \cos 0 \sin 0 \sin 0 + \sin 0 \cos 0 = 0 \end{aligned} \quad (3.56)$$

$$\begin{aligned} m_{31} &= \sin \phi \\ &= \sin 0 = 0 \end{aligned} \quad (3.57)$$

$$\begin{aligned} m_{32} &= -\sin \omega \cos \phi \\ &= -\sin 0 \cos 0 = 0 \end{aligned} \quad (3.58)$$

$$\begin{aligned} m_{33} &= \cos \omega \cos \phi \\ &= \cos 0 \cos 0 = 1 \end{aligned} \quad (3.59)$$

And;

$$\frac{\partial X}{\partial S} = m_{11}x + m_{21}y + m_{31}z = x \quad (3.60)$$

$$\frac{\partial Y}{\partial S} = m_{12}x + m_{22}y + m_{32}z = y \quad (3.61)$$

$$\frac{\partial Z}{\partial S} = m_{13}x + m_{23}y + m_{33}z = z \quad (3.62)$$

$$\frac{\partial Y}{\partial \omega} = -S(m_{13}x + m_{23}y + m_{33}z) = -z \quad (3.63)$$

$$\frac{\partial Z}{\partial \omega} = S(m_{12}x + m_{22}y + m_{32}z) = y \quad (3.64)$$

$$\frac{\partial X}{\partial \phi} = S[-\sin(\phi)\cos(k)x + \sin(\phi)\sin(k)y + \cos(\phi)z] = z \quad (3.65)$$

$$\begin{aligned} \frac{\partial Y}{\partial \phi} &= S[\sin(\omega)\cos(\phi)\cos(k)x - \sin(\omega)\cos(\phi)\sin(k)y + \sin(\omega)\sin(\phi)z] \\ &= 0 \end{aligned} \quad (3.66)$$

$$\begin{aligned} \frac{\partial Z}{\partial \phi} &= S[-\cos(\omega)\cos(\phi)\cos(k)x + \cos(\omega)\cos(\phi)\sin(k)y - \cos(\omega)\sin(\phi)z] \\ &= -x \end{aligned}$$

$$\frac{\partial X}{\partial k} = S(m_{21}x - m_{11}y) = -y \quad (3.67)$$

$$\frac{\partial Y}{\partial k} = S(m_{22}x - m_{12}y) = x \quad (3.68)$$

$$\frac{\partial Z}{\partial k} = -S(m_{23}x - m_{13}y) = 0 \quad (3.69)$$

The linearized equations of each control point, at first iteration one:

$$\begin{aligned} &\begin{bmatrix} \left(\frac{\partial X}{\partial S}\right)_0 & 0 & \left(\frac{\partial X}{\partial \phi}\right)_0 & \left(\frac{\partial X}{\partial k}\right)_0 & 1 & 0 & 0 \\ \left(\frac{\partial Y}{\partial S}\right)_0 & \left(\frac{\partial Y}{\partial \omega}\right)_0 & \left(\frac{\partial Y}{\partial \phi}\right)_0 & \left(\frac{\partial Y}{\partial k}\right)_0 & 0 & 1 & 0 \\ \left(\frac{\partial Z}{\partial S}\right)_0 & \left(\frac{\partial Z}{\partial \omega}\right)_0 & \left(\frac{\partial Z}{\partial \phi}\right)_0 & \left(\frac{\partial Z}{\partial k}\right)_0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} dS \\ d\omega \\ d\phi \\ d\kappa \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} = \begin{bmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \\ &\begin{bmatrix} x & 0 & z & -y & 1 & 0 & 0 \\ y & -z & 0 & x & 0 & 1 & 0 \\ z & y & -x & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} dS \\ d\omega \\ d\phi \\ d\kappa \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} = \begin{bmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{bmatrix} \quad (3.70) \end{aligned}$$

Then calculate matrix X and calculate the seven parameter S, ω , ϕ , k, T_x , T_y , and T_z .

$$S = S_o + dS$$

$$\omega = \omega_o + d\omega$$

$$\phi = \phi_o + d\phi$$

$$\kappa = \kappa_o + d\kappa$$

$$T_x = T_{x_o} + dT_x$$

$$T_y = T_{y_o} + dT_y$$

$$T_z = T_{z_o} + dT_z$$

3.3.4 3D Similarity 6-parameter methods

The 3D similarity 6-parameter transformation has six transformation parameter that include three translation parameters, and three rotation parameters. The scale factor is fixed to equal one, and used as constant.

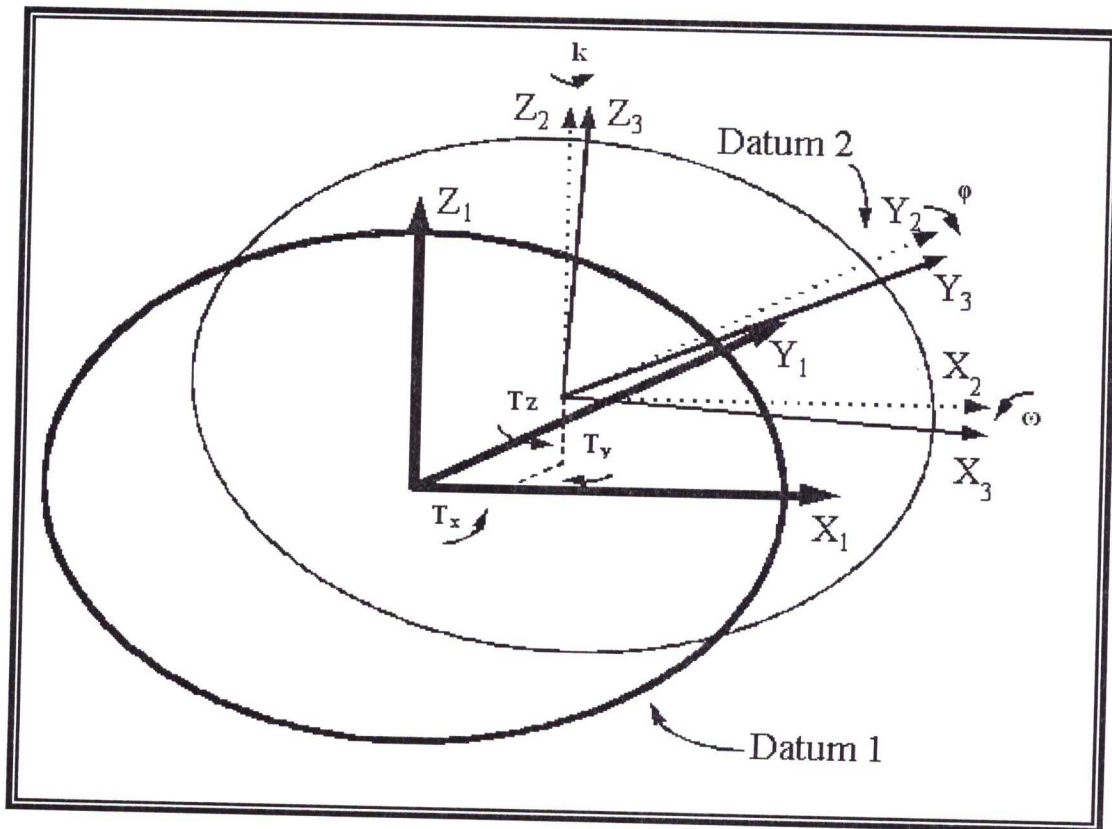


Figure (3.4) : 6-Parameter Transformation

The equations for the 3D similarity six-parameter transformation are:

$$X = (m_{11}x + m_{21}y + m_{31}z) + T_x \quad (3.71)$$

$$Y = (m_{12}x + m_{22}y + m_{32}z) + T_y \quad (3.72)$$

$$Z = (m_{13}x + m_{23}y + m_{33}z) + T_z \quad (3.73)$$

In matrix form:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (3.74)$$

$$X = M \cdot x + T_x \quad (3.75)$$

$$Y = M \cdot y + T_y \quad (3.76)$$

$$Z = M \cdot z + T_z \quad (3.77)$$

Where;

X, Y, Z: Coordinate system in first datum

S: scale, is not assigned while it equals to one.

M: rotation matrix

x, y, z: coordinate system in second datum

T: translation matrix

The equation for six-parameter can obtain by 3D similarity seven-parameter transformation

$$X = S(m_{11}x + m_{21}y + m_{31}z) + T_x \quad (3.78)$$

$$Y = S(m_{12}x + m_{22}y + m_{32}z) + T_y \quad (3.79)$$

$$Z = S(m_{13}x + m_{23}y + m_{33}z) + T_z \quad (3.80)$$

When;

$$S = 1$$

In the equations, six-parameter require a minimum number of two horizontal control stations with known X-Y and x-y-z coordinates, in addition to three stations with

known Z and x - y - z coordinates. If there are more than the minimum number of observations, a least-squares solution can be applied. The equations are nonlinear with respect to the unknowns and must be linearized for solution. The following linearized equations for a full control point are:

$$\begin{bmatrix} 0 & \left(\frac{\partial X}{\partial \phi}\right)_0 & \left(\frac{\partial X}{\partial k}\right)_0 & 1 & 0 & 0 \\ \left(\frac{\partial Y}{\partial \omega}\right)_0 & \left(\frac{\partial Y}{\partial \phi}\right)_0 & \left(\frac{\partial Y}{\partial k}\right)_0 & 0 & 1 & 0 \\ \left(\frac{\partial Z}{\partial \omega}\right)_0 & \left(\frac{\partial Z}{\partial \phi}\right)_0 & \left(\frac{\partial Z}{\partial k}\right)_0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d\omega \\ d\phi \\ d\kappa \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} = \begin{bmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \quad (3.81)$$

Where;

$$\frac{\partial Y}{\partial \omega} = -(m_{13}x + m_{23}y + m_{33}z) \quad (3.82)$$

$$\frac{\partial Z}{\partial \omega} = (m_{12}x + m_{22}y + m_{32}z) \quad (3.83)$$

$$\frac{\partial X}{\partial \phi} = [-\sin(\phi)\cos(k)x + \sin(\phi)\sin(k)y + \cos(\phi)z] \quad (3.84)$$

$$\frac{\partial Y}{\partial \phi} = [\sin(\omega)\cos(\phi)\cos(k)x - \sin(\omega)\cos(\phi)\sin(k)y + \sin(\omega)\sin(\phi)z] \quad (3.85)$$

$$\frac{\partial Z}{\partial \phi} = [-\cos(\omega)\cos(\phi)\cos(k)x + \cos(\omega)\cos(\phi)\sin(k)y - \cos(\omega)\sin(\phi)z] \quad (3.86)$$

$$\frac{\partial X}{\partial k} = (m_{21}x - m_{11}y) \quad (3.87)$$

$$\frac{\partial Y}{\partial k} = (m_{22}x - m_{12}y) \quad (3.88)$$

$$\frac{\partial Z}{\partial k} = (m_{23}x - m_{13}y) \quad (3.89)$$

This matrix from of the solution by least square is:

$$X = (A^T A)^{-1} A^T L \quad (3.90)$$

$$X = \begin{bmatrix} dS \\ d\omega \\ d\phi \\ d\kappa \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} \quad (3.91)$$

The six-parameters after the first iteration are calculation as in the equations (3.92-3.97), and initial value of second iteration equal:

$$\omega = \omega_o + d\omega \quad (3.92)$$

$$\phi = \phi_o + d\phi \quad (3.93)$$

$$\kappa = \kappa_o + d\kappa \quad (3.94)$$

$$T_x = T_{x_o} + dT_x \quad (3.95)$$

$$T_y = T_{y_o} + dT_y \quad (3.96)$$

$$T_z = T_{z_o} + dT_z \quad (3.97)$$

To calculate the elements of the matrixes, we must give initial values for the parameters, the following can give good results normally in GPS and Geodesy

$$\omega_o = \phi_o = \kappa_o = 0$$

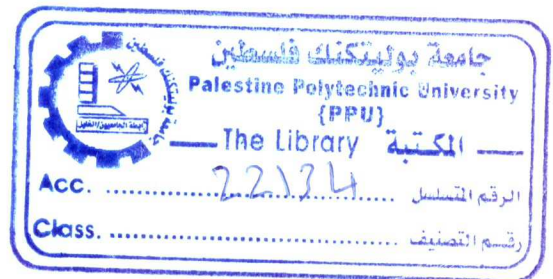
$$T_{x_o} = (X - x), \quad T_{y_o} = (Y - y), \quad T_{z_o} = (Z - z)$$

Then the initial values of the observations are calculated using the initial values of the unknown parameters:

$$X_o = (m_{11}x + m_{21}y + m_{31}z) + T_{x_o} \quad (3.98)$$

$$Y_o = (m_{12}x + m_{22}y + m_{32}z) + T_{y_o} \quad (3.99)$$

$$Z_o = (m_{13}x + m_{23}y + m_{33}z) + T_{z_o} \quad (3.100)$$



Where;

$$\begin{aligned} m_{11} &= \cos \phi \cos \kappa \\ &= \cos 0 \cos 0 = 1 \end{aligned}$$

$$\begin{aligned} m_{12} &= \sin \omega \sin \phi \cos \kappa + \cos \omega \sin \kappa \\ &= \sin 0 \sin 0 \cos 0 + \cos 0 \sin 0 = 0 \end{aligned}$$

$$\begin{aligned} m_{13} &= -\cos \omega \sin \phi \cos \kappa + \sin \omega \sin \kappa \\ &= -\cos 0 \sin 0 \cos 0 + \sin 0 \sin 0 = 0 \end{aligned}$$

$$\begin{aligned} m_{21} &= -\cos \phi \sin \kappa \\ &= -\cos 0 \sin 0 = 0 \end{aligned}$$

$$\begin{aligned} m_{22} &= -\sin \omega \sin \phi \sin \kappa + \cos \omega \cos \kappa \\ &= -\sin 0 \sin 0 \sin 0 + \cos 0 + \cos 0 = 1 \end{aligned}$$

$$\begin{aligned} m_{23} &= \cos \omega \sin \phi \sin \kappa + \sin \omega \cos \kappa \\ &= \cos 0 \sin 0 \sin 0 + \sin 0 \cos 0 = 0 \end{aligned}$$

$$\begin{aligned} m_{31} &= \sin \phi \\ &= \sin 0 = 0 \end{aligned}$$

$$\begin{aligned} m_{32} &= -\sin \omega \cos \phi \\ &= -\sin 0 \cos 0 = 0 \end{aligned}$$

$$\begin{aligned} m_{33} &= \cos \omega \cos \phi \\ &= \cos 0 \cos 0 = 1 \end{aligned}$$

And;

$$\frac{\partial Y}{\partial \omega} = -S(m_{13}x + m_{23}y + m_{33}z) = -z$$

$$\frac{\partial Z}{\partial \omega} = S(m_{12}x + m_{22}y + m_{32}z) = y$$

$$\frac{\partial X}{\partial \phi} = S[-\sin(\phi) \cos(k)x + \sin(\phi) \sin(k)y + \cos(\phi)z] = z$$

$$\frac{\partial Y}{\partial \phi} = S[\sin(\omega) \cos(\phi) \cos(k)x - \sin(\omega) \cos(\phi) \sin(k)y + \sin(\omega) \sin(\phi)z] = 0$$

$$\frac{\partial Z}{\partial \phi} = S[-\cos(\omega)\cos(\phi)\cos(k)x + \cos(\omega)\cos(\phi)\sin(k)y - \cos(\omega)\sin(\phi)z] = -x$$

$$\frac{\partial X}{\partial k} = S(m_{21}x - m_{11}y) = -y$$

$$\frac{\partial Y}{\partial k} = S(m_{22}x - m_{12}y) = x$$

$$\frac{\partial Z}{\partial k} = -S(m_{23}x - m_{13}y) = 0$$

The linearized equations of each control point in the first iteration is:

$$\begin{bmatrix} 0 & \left(\frac{\partial X}{\partial \phi}\right)_0 & \left(\frac{\partial X}{\partial k}\right)_0 & 1 & 0 & 0 \\ \left(\frac{\partial Y}{\partial \omega}\right)_0 & \left(\frac{\partial Y}{\partial \phi}\right)_0 & \left(\frac{\partial Y}{\partial k}\right)_0 & 0 & 1 & 0 \\ \left(\frac{\partial Z}{\partial \omega}\right)_0 & \left(\frac{\partial Z}{\partial \phi}\right)_0 & \left(\frac{\partial Z}{\partial k}\right)_0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d\omega \\ d\phi \\ d\kappa \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} = \begin{bmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \quad (3.101)$$

$$\begin{bmatrix} 0 & z & -y & 1 & 0 & 0 \\ -z & 0 & x & 0 & 1 & 0 \\ y & -x & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d\omega \\ d\phi \\ d\kappa \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} = \begin{bmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \quad (3.102)$$

Then calculate matrix X, and the six parameter ω , ϕ , k , T_x , T_y , and T_z are calculated as follows:

$$\omega = \omega_0 + d\omega$$

$$\phi = \phi_0 + d\phi$$

$$\kappa = \kappa_0 + d\kappa$$

$$T_x = T_{x0} + dT_x$$

$$T_y = T_{y0} + dT_y$$

$$T_z = T_{z0} + dT_z$$

3.3.5 3D Similarity 3-parameter

The simplest datum transformation method is three-parameter transformation or a geocentric translation. When the minor axes of the ellipsoids are assumed to be parallel, to transform assuming that rotations are zeros, then shifts (T_x , T_y , T_z) are defined from source geocentric coordinate system to target geocentric coordinate system or from a local datum to WGS 1984 or another geocentric datum as common in GPS. The three parameters are linear shifts and are always in distance units, and usually called datum shifts.

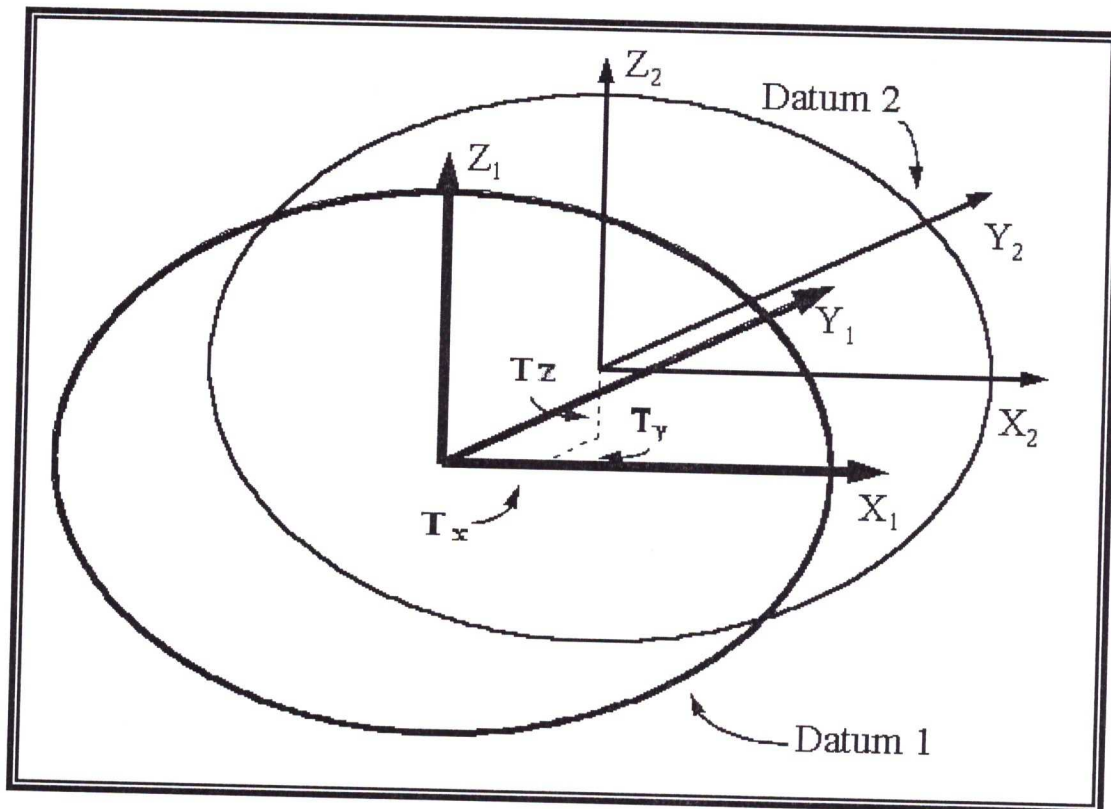


Figure (3.5) : 3-Parameter Transformation

The mathematical model for the 3D similarity three-parameter transformation is:

$$\begin{aligned} X &= x + T_x \\ Y &= y + T_y \\ Z &= z + T_z \end{aligned} \quad (3.103)$$

In matrix form;

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (3.104)$$

This equation can be obtained from 3D similarity seven-parameter transformation

$$\begin{aligned} X &= S(m_{11}x + m_{21}y + m_{31}z) + T_x \\ Y &= S(m_{12}x + m_{22}y + m_{32}z) + T_y \\ Z &= S(m_{13}x + m_{23}y + m_{33}z) + T_z \end{aligned}$$

When;

$$\begin{aligned} S &= 1 \\ \omega &= \phi = \kappa = 0 \end{aligned}$$

The least squares solution as linear solution:

$$\begin{aligned} AX &= L + V \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} &= \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \end{aligned} \quad (3.105)$$

The final solution of the unknowns is:

$$\begin{aligned} X &= (A^T A)^{-1} A^T L \\ X &= \begin{bmatrix} dT_x \\ dT_y \\ dT_z \end{bmatrix} \end{aligned} \quad (3.106)$$

The 3-parameter after first iteration, are calculated in equation (3.107), initial value of second iteration use the final results of the first iterat:

$$\begin{aligned} T_x &= T_{x_0} + dT_x \\ T_y &= T_{y_0} + dT_y \\ T_z &= T_{z_0} + dT_z \end{aligned} \quad (3.107)$$

In Palestine-1923-Grid, the transformation is defined by three translations. The values of the translation are taken from Trimble geomatics office software. According to the following equations:

$$\begin{aligned} X_{\text{Palestine-1923}} &= X_{\text{WGS84}} + \Delta X \\ Y_{\text{Palestine-1923}} &= Y_{\text{WGS84}} + \Delta Y \\ Z_{\text{Palestine-1923}} &= Z_{\text{WGS84}} + \Delta Z \end{aligned} \quad (3.108)$$

Where;

$$\Delta X = 230.00 \text{ m}, \Delta Y = 71.00 \text{ m}, \Delta Z = -273 \text{ m}$$

3.3.6 Helmert transformations

The Helmert transformations parameters are three linear shifts (TX, TY, TZ), three angular rotations around each axis (r_x , r_y , r_z), and scale factor(S). The rotation values are given in decimal seconds.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{\text{new}} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + S \cdot \begin{bmatrix} 1 & r_z & -r_y \\ -r_z & 1 & r_x \\ r_y & -r_x & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{\text{original}} \quad (3.109)$$

$$X_{\text{new}} = S(X + r_z \cdot Y - r_y \cdot Z) + T_x \quad (3.110)$$

$$Y_{\text{new}} = S(-r_z \cdot X + Y + r_x \cdot Z) + T_y \quad (3.111)$$

$$Z_{\text{new}} = S(r_y \cdot X - r_x \cdot Y + Z) + T_z \quad (3.112)$$

Where;

X, Y, Z: point coordinations in the target system

X, Y, Z: point coordinations in the source

S : scale factor

r_x, r_y, r_z : angular rotations

T_x, T_y, T_z : linear shifts

A least-squares solution can be applied. The equations are nonlinear with respect to the unknowns and must be linearized for a solution. The following linearized equations for a full control point (Horizontal and Vertical)

$$\begin{bmatrix} \left(\frac{\partial X}{\partial S}\right)_0 & \left(\frac{\partial X}{\partial r_x}\right)_0 & \left(\frac{\partial X}{\partial r_y}\right)_0 & \left(\frac{\partial X}{\partial r_z}\right)_0 & 1 & 0 & 0 \\ \left(\frac{\partial Y}{\partial S}\right)_0 & \left(\frac{\partial Y}{\partial r_x}\right)_0 & \left(\frac{\partial Y}{\partial r_y}\right)_0 & \left(\frac{\partial Y}{\partial r_z}\right)_0 & 0 & 1 & 0 \\ \left(\frac{\partial Z}{\partial S}\right)_0 & \left(\frac{\partial Z}{\partial r_x}\right)_0 & \left(\frac{\partial Z}{\partial r_y}\right)_0 & \left(\frac{\partial Z}{\partial r_z}\right)_0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} dS \\ dr_x \\ dr_y \\ dr_z \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} = \begin{bmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \quad (3.113)$$

Where;

$$\frac{\partial X}{\partial S} = X + r_z.Y - r_y.Z \quad (3.114)$$

$$\frac{\partial Y}{\partial S} = -r_z.X + Y + r_x.Z \quad (3.115)$$

$$\frac{\partial Z}{\partial S} = r_y.X - r_x.Y + Z \quad (3.116)$$

$$\frac{\partial X}{\partial r_x} = 0 \quad (3.117)$$

$$\frac{\partial X}{\partial r_y} = -S.Z \quad (3.118)$$

$$\frac{\partial X}{\partial rz} = SY \quad (3.119)$$

$$\frac{\partial Y}{\partial rx} = S.Z \quad (3.120)$$

$$\frac{\partial Y}{\partial ry} = 0 \quad (3.121)$$

$$\frac{\partial Y}{\partial rz} = -S.X \quad (3.122)$$

$$\frac{\partial Z}{\partial rx} = -S.Y \quad (3.123)$$

$$\frac{\partial Z}{\partial ry} = S.X \quad (3.124)$$

$$\frac{\partial Z}{\partial rz} = 0 \quad (3.125)$$

The matrix solution is:

$$X = (A^T A)^{-1} A^T L \quad (3.126)$$

$$X = \begin{bmatrix} dS \\ dr_x \\ dr_y \\ dr_z \\ dT_x \\ dT_y \\ dT_z \end{bmatrix} \quad (3.127)$$

The Helmert 7-parameter transformation are calculated after first iteration, an assigned as the initial value of second iteration; where:

$$S = S_o + dS \quad (3.128)$$

$$rx = rx_o + drx \quad (3.129)$$

$$ry = ry_o + dry \quad (3.130)$$

$$rz = rz_o + drz \quad (3.131)$$

$$T_x = T_{x_o} + dT_x \quad (3.132)$$

$$T_y = T_{y_o} + dT_y \quad (3.133)$$

$$T_z = T_{z_o} + dT_z \quad (3.134)$$

3.3.7 Molodensky method

The Molodensky method is a complex formula for the shift in latitude, longitude and height. The number of parameters in this method is small. It converts directly between two geographic coordinate systems without converting to an X, Y, Z system. This method have three shifts (ΔX , ΔY , ΔZ) and the differences between the semi-major axes (Δa) and the flattening (Δf) of the two spheroids. The system automatically calculates the spheroid differences according to the datums. Solve for $\Delta \lambda$ and $\Delta \Phi$. The amounts are added automatically by the system.

3.3.7.1 Vector Viewpoint of Molodensky Transformation

The Molodensky transformation is based of all differences due to:

1. A shift in origin by a vector with components (ΔX , ΔY , ΔZ)
2. A difference in ellipsoids of size, Δa and flattening Δf

The unit vectors in the east, north, and up directions as coming from topocentric system, and are written as follows:

$$\hat{e}_E = (-\sin \lambda, \cos \lambda, 0) \quad (3.135)$$

$$\hat{e}_N = (-\sin \phi \cos \lambda, -\sin \phi \sin \lambda, \cos \phi) \quad (3.136)$$

$$\hat{e}_U = (\cos \phi \cos \lambda, \cos \phi \sin \lambda, \sin \phi) \quad (3.137)$$

The dot product of the unit vectors is used to obtain the origin shift $\vec{\Delta}$. The distance units for the shift in East, North, and up is:

$$\Delta E = \vec{e}_E \bullet \vec{\Delta} \quad (3.138)$$

$$\Delta N = \vec{e}_N \bullet \vec{\Delta} + \frac{\Delta a}{a} e^2 R_N \cos \phi \sin \phi + \Delta f \left[\frac{a}{b} R_M + \frac{b}{a} R_N \right] \cos \phi \sin \phi \quad (3.139)$$

$$\Delta H = \vec{e}_U \bullet \vec{\Delta} - \Delta a \frac{a}{R_N} + \Delta f \frac{b}{a} R_N \sin^2 \phi \quad (3.140)$$

The procedure for the East-Change is to convert it to arc-seconds of longitude and to convert North-Change to arc-seconds of latitude. This is done with the correct radius of curvature and the sine of one arc-second.

The scaling lengths are:

$$L_E = (R_N + H) \cos \phi \sin 1'' \quad (3.141)$$

$$L_N = (R_M + H) \sin 1'' \quad (3.142)$$

The equations are found by dividing scaling lengths by these values:

$$\Delta \lambda'' = \frac{\Delta E}{L_E} \quad (3.143)$$

$$\Delta \phi'' = \frac{\Delta N}{L_N} \quad (3.144)$$

3.3.7.2 The Usual Statement of the Molodensky Transformation

The Moldensky transformation give directly the shifts in latitude, longitude, and height. The angular values are in arcseconds. These formulas will be given:

$$\Delta\varphi'' = \frac{\left\{ \begin{aligned} & -\Delta X \sin \varphi \cos \lambda - \Delta Y \sin \varphi \sin \lambda + \Delta Z \cos \varphi \\ & + \frac{\Delta a}{a} (R_N e^2 \sin \varphi \cos \varphi) + \Delta f \left(\frac{a}{b} R_M + \frac{b}{a} R_N \right) \sin \varphi \cos \varphi \end{aligned} \right\}}{[(R_M + h) \sin 1'']} \quad (3.145)$$

$$\Delta\lambda'' = \frac{\{-\Delta X \sin \lambda + \Delta Y \cos \lambda\}}{\{(R_N + h) \cos \varphi \sin 1''\}} \quad (3.146)$$

$$\begin{aligned} \Delta h &= \Delta X \cos \varphi \cos \lambda + \Delta Y \cos \varphi \sin \lambda + \Delta Z \sin \varphi \\ &\quad - \frac{\Delta a}{a} \cdot \frac{a}{R_N} + \Delta f b \frac{RN}{a} \sin^2 \varphi \end{aligned} \quad (3.147)$$

This is often shortened to the Abridged Molodensky Transform given by:

$$\Delta\varphi'' = \frac{\left[\begin{aligned} & -\Delta X \sin \varphi \cos \lambda - \Delta Y \sin \varphi \sin \lambda \\ & + \Delta Z \cos \varphi + (a\Delta f + f\Delta a) \end{aligned} \right]}{[R_M \sin 1'']} \quad (3.148)$$

$$\Delta\lambda'' = \frac{[-\Delta X \sin \lambda + \Delta Y \cos \lambda]}{[R_N \cos \varphi \sin 1'']} \quad (3.149)$$

$$\begin{aligned} \Delta h &= \Delta X \cos \varphi \cos \lambda + \Delta Y \cos \varphi \sin \lambda \\ &\quad + \Delta Z \sin \varphi - \Delta a + (a\Delta f + f\Delta a) \sin^2 \varphi \end{aligned} \quad (3.150)$$

Here;

h ellipsoid height (meters)

φ latitude , **λ** longitude in the source system

a semi-major axis of the spheroid (meters)

b semi-minor axis of the spheroid (meters)

f flattening of the spheroid, **e** eccentricity of the spheroid

RM and **RN** are the meridian and prime vertical radius of curvature, respectively,

at a given latitude. The equations for RM and RN are:

In matrix form;

$$L = \begin{bmatrix} \Delta\varphi'' \\ \Delta\lambda'' \\ \Delta h \end{bmatrix} \quad (3.151)$$

The least squares solution as linear solution:

$$AX = L + V \quad (3.152)$$

$$\begin{bmatrix} \frac{-\sin\varphi \cos\lambda}{R_M \sin 1''} & \frac{-\sin\varphi \sin\lambda}{R_M \sin 1''} & \frac{\cos\varphi}{R_M \sin 1''} \\ \frac{-\sin\lambda}{R_N \cos\varphi \sin 1''} & \frac{\cos\lambda}{R_N \cos\varphi \sin 1''} & 0 \\ \cos\varphi \cos\lambda & \cos\varphi \sin\lambda & \sin\varphi \end{bmatrix} \begin{bmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{bmatrix} = \begin{bmatrix} \Delta\varphi'' \\ \Delta\lambda'' \\ \Delta h \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \quad (3.153)$$

The final solution of the unknowns is:

$$X = (A^T A)^{-1} A^T L \quad (3.154)$$

$$X = \begin{bmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{bmatrix} \quad (3.155)$$

Where:

$$RM = \frac{a(1-e^2)}{(1-e^2 \sin^2 \varphi)^{3/2}} \quad (3.156)$$

$$RN = \frac{a}{(1-e^2 \sin^2 \varphi)^{1/2}} \quad (3.157)$$

$$\Delta a = a_{Target} - a_{source}$$

$$\Delta f = f_{Target} - f_{source}$$

The advantage of Moldensky over similarity transformation is that we can easily separate horizontal position (λ , ϕ) and the vertical position (n), while there is no need to refer to the Geocentric coordinates X , Y and Z , in addition on difference in the ellipsoid used are included in the transformation.

CHAPTER

4

USING THE GEO-TRANSFORM

4-1 Introduction

4.2 Using the Geo transform

4.3 Starting geo transform for using datum transformation

CHAPTER FOUR

USING THE GEO-TRANSFORM

4-1 Introduction

The geo transform is a tool programmed using visual Basic 6 programming language. This tool has the following functionalities; the first is to transform from Geographic coordinates (λ , ϕ , h) to geocentric coordinates (X , Y , Z), Where the same ellipsoid is used. The second is to transform 3D geographic or geocentric coordinates (λ , ϕ) to 2D projected (Grid) coordinates (Easting, Northing), the used map projection in this transformation tool are: Cassini, Transverse Mercator (TM) and Universal Transverse Mercator (UTM). The Third is to apply the reverse transformation of the first and second functionalities.

In addition, a separate datum transformation tool is to be built to calculate the datum transformation parameters using different datum transformation methods similarity 7-parametres, similarity 6-parameters, similarity 3-parameters, Moldensky transformation and Helmert transformations.

In this tool three Basic map projections are used; Cassini, Transverse Mercator TM, and universal Transverse Mercator UTM. In Palestine there are four map projection and coordinate systems used, These are Palestine_1923_Palestine_Grid, Palestine_1923_Israel_CS_Grid, palestine_1923_Belt, and Israel_TM_Grid. Where each coordinates system has its own projection parameters.

The Geo transform is a program capable to:-

1. Transform from Geographic coordinates (λ , ϕ , h) to Geocentric coordinates (X , Y , Z) on an ellipsoid.
2. Transform from Geocentric coordinates (X , Y , Z) to Geographic coordinates (λ , ϕ , h) on an ellipsoid.

3. Transform from Geographic coordinates (λ , ϕ) to projected coordinates (Easting, Northing), for Cassini, Transverse Mercator TM, and universal Transverse Mercator UTM using a coordinates system.
4. Transform from Projected coordinates (Easting, Northing) to Geographic coordinates (λ , ϕ) using Cassini, Transverse Mercator (TM) and Universal Transverse Mercator (UTM) using a coordinates system.
5. Calculate the datum transformation parameters using different transformation method; similarity 7-parametres, similarity 6-parameters, similarity 3-parameters, Moldensky transformation and Helmert transformations.
6. Transform the coordinates between different coordinates systems.

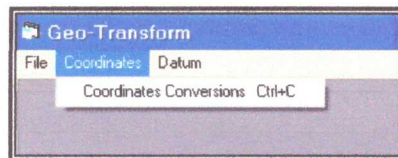
4.2 Using the Geo-transform

Using the Geo-transform program:-

The use of the Geo-transform program & its different steps are shown in the following sections.

4.2.1 Transform from geographic to geocentric coordinates for a single point:

1. Open the geo-transform program.
2. Click on coordinates menu & then select coordinates conversions.



fig(4.1) coordinates menu

3. Choose a single point key.
4. Choose the lon, lat, h key.
5. Enter the value of lon, lat & there direction, then enter the value of h. The direction of lon is East or West, but the direction of lat is North or South.

lon,Lat,h==>X Y Z XYZ==>Lon,Lat,h Projection - Direct Projection - Inverse

Input File

☒ Single point
☐ Points File

Select Input File Select Output File

Input Point

lon East
lat North
h

Ellipsoid

Select Ellipsoid

Clark 1880
Bessel
GRS80
GRS67
WGS84
user define

300.79
566.435

Calculate XYZ

Output

X Y Z

fig(4.2):lon,lat,h→X,Y,Z dialog

6. Select the ellipsoid you need, & then the value of a&b will appear according to the type of ellipsoid you select, but if you choose the user define you will enter the type of ellipsoid & the value of a&b.
7. Press the calculate key to have the result

Example (1):

- 1-In this example we choose lon, lat, h→X, Y, Z.
- 2-Choose a single point.
- 3-Enter the value lon=45 & the direction of it East, then enter the value of lat = 45 & its direction North. Enter the value of h=0.
- 4-Select Clarck 1880 ellipsoid, then the value of a&b will appear.
- 5-Press calculate key, then the value of X, Y, Z will appear as in fig (4.3).

lon,Lat,h=>X Y Z XYZ=>Lon,Lat,h Projection - Direct Projection - Inverse

Input File

☒ Single point
☐ Points File

Input Point

lon: 45 East
lat: 45 North
h: 0

Ellipsoid

Clarck 1880

a: 6,378,206.73
b: 6,356,583.43

Calculate XYZ

Output

X: 3194588.60521907 Y: 3194588.60587429 Z: 4487093.5511734

fig(4.3): lon,lat,h→X,Y,Z dialog

4.2.2 Transform from Geocentric coordinates to Geographic coordinates for a single point:

1. Choose the X Y Z → Lon, lat, h.
2. Choose a single point key.
3. Enter the value of the variables X Y Z .
4. Select the ellipsoid you need & the value of a&b will appear according to the type of the ellipsoid.
5. If you finish entering the information presses calculate key & the value of lon lat h will appear with their direction.

lon, Lat, h ==> X Y Z **X Y Z ==> Lon, Lat, h** Projection - Direct Projection - Inverse

Input File

☒ Single point
☐ Points File

Select Input File Select Output File

Input

X Y Z

Ellipsoid

GRS87
Clarck 1880
Bessel
GRS80
GRS87
WGS84
user define

a b

calculate lon, lat, h

Output

lon lat h

fig(4.4): X,Y,Z→lon,lat,h dialog

Example 2:

1. In this example we choose X Y Z → lon lat h
2. Choose a single point.
3. Enter the value of X=3194419.1444 , Y=3194419.145 , Z=4487348.409
4. Select Clarck 1880 ellipsoid & the value of a&b will appear directly.
5. Press calculate key ,then the result will appear in the screen.

lon,Lat,h==>X Y Z	X Y Z==>Lon,Lat,h	Projection - Direct	Projection - Inverse
Input File ☒ Single point ☐ Points File Select Input File Select Output File			
Input X 3194419.1444 Y 3194419.145 Z 4487348.409			
Ellipsoid Clarck 1880 a 6378300.79 b 6356583.894999999			
calculate lon, lat, h			
Output lon 44.9999999995 lat 45.0031464679 h 10.7604521270			

fig (4.5): X,Y,Z→lon,lat,h dialog

4.2.3 Transform from Geographic coordinates (λ , ϕ) to projected coordinates (Easting, Northing) for a single point:

1. Choose the projection-direct key.
2. Choose the single point to enter one point.
3. Enter the value of lat & its direction,& the value of lon & its direction.
4. Choose the coordinate system you need, if you choose:
 - Palestine 1923 Grid the projection will be Cassini & the ellipsoid will be Clarck 1880.
 - Palestine-1923-Israel-CS-Grid, the projection will be Cassini & the ellipsoid will be Clarck 1880.
 - UTM .the projection will be Transverse Mercator & the ellipsoid will be enabling & you will enter zone number.

- Palestine-1923-Belt, then the projection will be Transverse Mercator & the ellipsoid will be Clarck1880.
 - Israel-TM-Grid, then the projection will be Transverse Mercator, & the ellipsoid will be GRS80.
 - Users define, you will choose Cassini projection or Transverse Mercator, & the ellipsoid will be enabling.
5. Select the ellipsoid you need & the value of a&b will appear directly.
 6. Enter the value of the projection parameter FE, FN, lat0, lon0, k0.
 7. Press the calculate key to have the results.

lon,Lat,h==>X Y Z X Y Z==>Lon,Lat,h **Projection - Direct** Projection - Inverse

Input File

☒ Single point
☐ Points File

Select Input File Select Output File

Input lon, lat

lat 31.50526858 South lon 35.08906226 West

Coordinate System

Palestine_1923_Grid

Ellipsoide

Palestine_1923_Grid
Palestine_1923_Israel_CS_Grid
UTM
palestine_1923_Belt
Israel_TM_Grid
user define

a 126367.909 b 170221.555

Projection Parameter

FN 126367.909 lat0 31.7340363444 Norm
FE 170221.555 lon0 35.2120605556 East k0 1

Calculate easting,northing

Output

Easting Northing

fig (4.6): projection-direct dialog

Example (3): If you select Palestine-1923-Grid

- a. Select projection-Direct key.
- b. Choose a single point.

- c. Enter the value of lat=31.505 & its direction north, the value of lon=35.089 & its direction east.
- d. Select Palestine-1923-Grid.
- e. The ellipsoid will be clarck 1880 directly.
- f. Enter the value of projection parameters, FN=126867.909, FE=170251.555, lat0=31.734 & its direction north, lon0=35.212 & its direction east, k0=1.
- g. Press calculate key to show the result.

The screenshot shows the 'Projection - Direct' dialog box. It has four tabs at the top: 'lon,Lat,h==>XYZ', 'XYZ==>Lon,Lat,h', 'Projection - Direct' (which is selected), and 'Projection - Inverse'. The main area contains several input sections:

- Input File:** Two radio buttons, 'Single point' (selected) and 'Points File'. Below them are two buttons: 'Select input file' and 'Load input file'.
- Input lon , lat:** Two input fields. The 'lat' field contains '31.50526858' and has a dropdown menu set to 'North'. The 'lon' field contains '35.08906226' and has a dropdown menu set to 'East'.
- Coordinate System:** A dropdown menu set to 'Palestine_1923_Grid'.
- Ellipsoide:** A dropdown menu set to 'Clarck 1880'. To its right are input fields for 'a' (containing '6378137.0') and 'b' (containing '6356583.8').
- Projection Parameter:** A section with input fields for 'FN' (126867.909), 'FE' (170251.555), 'lat0' (31.734), 'lon0' (35.212), and 'k0' (1). Each of these fields has a dropdown menu for direction (North or East).
- Calculate easting,northing:** A button with a dashed border.
- Output:** Two output fields. 'Easting' contains '158566.61465794' and 'Northing' contains '101502.925748367'.

Fig (4.7): projection-direct dialog

Example (4): If you select UTM

- a. Choose Projection-Direct
- b. Choose single point.
- c. Enter the value of lat=31.50526858 & its direction North, the value of lon=35.08906226 & its direction East.
- d. Select the UTM coordinate system,& then enter the zone number=36.
- e. Select the ellipsoid you need, here we select GRS80.
- f. Enter the projection parameters, FN=0, FE=500000, lat0=0 & its direction North, lon0=33 & its direction East, k0=0.9996.
- g. Press the calculate key to show the result.

lon,Lat,h==>XYZ	XYZ==>Lon,Lat,h	Projection - Direct	Projection - Inverse
Input File ☒ Single point ☐ Points File Select Input File Select Output File			
Input lon , lat lat 31.50526858 North lon 35.08906226 East			
Coordinate System UTM zone 36 Transverse Meridian			
Ellipsoide GRS80 a 6378137 b 6356752.314			
Projection Parameter FN 0 lat0 0 North k0 0.9996 FE 500000 lon0 33 East			
Calculate easting,northing			
Output Easting 698395.35437982 Northing 3487491.18380941			

fig(4.8): projection-direct dialog

Example (5): If you select Palestine- 1923-Belt

- a. Choose Projection-Direct key.

- b. Choose a single point key.
- c. Enter lat=31.50526858 & its direction is North, Lon=35.08906226 & its direction is East.
- d. Select Palestine -1923-Belt, then the projection will be directly Transverse Mercator.
- e. The ellipsoid will enter directly Clarck 1880.
- f. Enter the projection parameter,
FN=1126867.909, FE=170251.555, lat0=31.734096944 & its direction North, lon0=35.2120605555 & its direction East, k0=1.
- g. Press the calculate key to show the result.

lon, Lat, h ==> X Y Z		X Y Z ==> Lon, Lat, h		Projection - Direct		Projection - Inv	
Input File ☒ Single point ☐ Points File Select Input File Select Output File							
Input lon , lat lat 31.50526858 North lon 35.08906226 East							
Coordinate System palestine_1923_Belt Transverse Mercator							
Ellipsoide Clarck 1880 a 6378300.78 b 6356583.435							
Projection Parameter FN 1126867.909 FE 170251.555 lat0 31.734096944 North lon0 35.2120605555 East k0 1							
Calculate easting,northing							
Output Easting 158566.6081015 Northing 1101502.92574839							

fig (4.9): projection-direct dialog

4.2.4 Transform from projected coordinates (Easting,Northing) to Geographic coordinates (λ,ϕ) for a single point:

1. Select Projection-Inverse key.
2. In input file check a single point.
3. Enter the value of Easting & Northing.
4. Select the coordinate system you need
5. Select the ellipsoid you need & then his parameter will enter directly.
6. Enter the value of the projection parameters FN, FE, lon0, lat0, k0.
7. Press the calculate key to show the result.

fig(4.10): projection-inverse dialog

Example (6): If you choose Palestine-1923-Israel-Cs-Grid coordinate system

1. Click Projection-Inverse type.
2. In input file click single point.

3. Enter the value of Easting= 158564.715 ,Northing= 101502.928
4. Choose Palestine-1923-Israel-Cs-Grid coordinate system then the projection will be Cassini directly.
5. The ellipsoid will be selected directly Clarck 1880 & his parameter a&b.
6. The value of projection parameter will enter a lone too.
7. Press calculate key to show the result.

lon,Lat,h==>XYZ XYZ==>Lon,Lat,h Projection - Direct **Projection - Inverse**

Input File

☒ Single point
☐ Points File

Select Input File Select Output File

Input

Easting 158564.715 Northing 101502.928

Coordinate System

Palestine_1923_Israel_CS_Grid Cassini

Ellipsoid

Clarck 1880 a 6376300.79 b 6356564.475

Projection Parameter

FE 170251.555 lat0 31.7340953444 North k0 1
FN 1126862.909 lon0 35.2120605556 East

Calculate easting,northing

Output

lat 22.4799524614851 North lon 35.0985018528565 East

fig(4.11): projection-inverse dialog

Example (7) : If you choose User define :

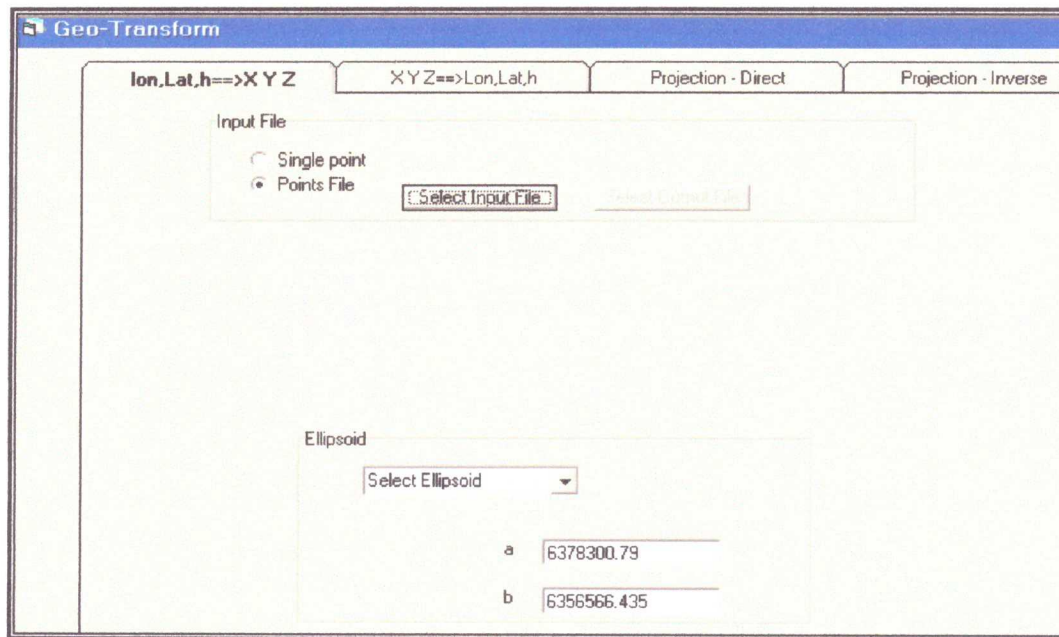
1. Click projection-inverse.
2. In input file click single point.
3. In the input enter the value of Easting & Northing.
4. In the coordinate tape choose user define 7 enter what do you need & your projection will be Transverse Mercator or Cassini.
5. Select Bassel ellipsoid for example & you can select any one you need.
6. Enter the value of the projection parameter you have .
7. Press calculate key to see the result.

lon,Lat,h==>XYZ	XYZ==>Lon,Lat,h	Projection - Direct	Projection - Inverse
Input File ☒ Single point ☐ Points File Select Input File Select Output File			
Input Easting 158564.715 Northing 101502.928			
Coordinate System user define Transverse Mercator Cassini Transverse Mercator			
Ellipsoid Bessel a 6377483.863 b 6356483.863			
Projection Parameter FE 170251.555 FN 126867.909 lat0 31.734097 lon0 35.212081 North East k0 1			
Calculate easting,northing			
Output lat 31.5052568579668 North lon 35.089044861591 East			

fig(4.12): projection-inverse dialog

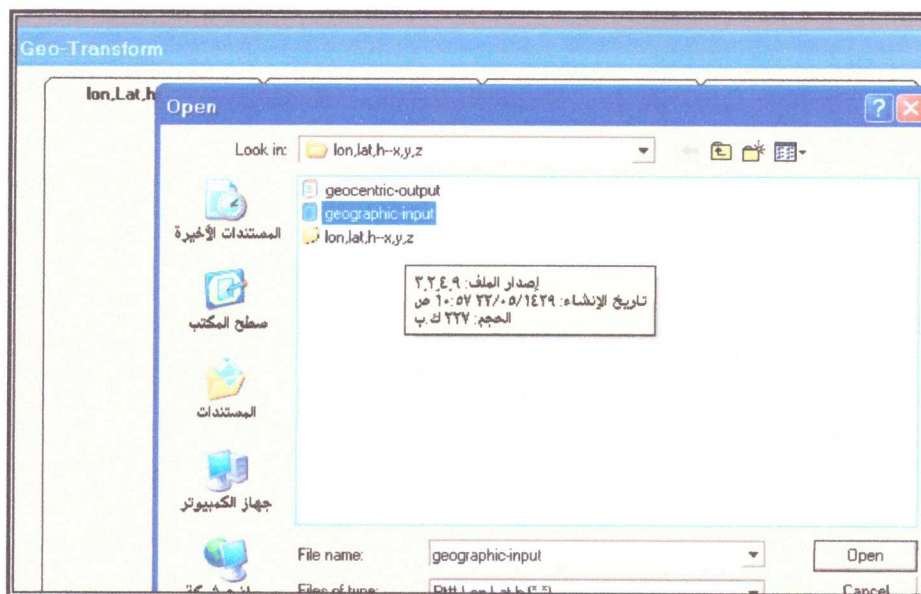
4.2.5 Transform from Geographic coordinates to Geocentric coordinates for a point file:

1. Select lon,lat,h→X,Y,Z.
2. Select point file.
3. Press select input file tape.



fig(4.13):lon,lat,h→x,y,z dialog

4. Select lon,lat,h→X,Y,Z folder.



fig(4.14):open dialog

5. Press open key.
6. Select the geographic input file,then press open key.
7. Select the ellipsoid you need from the main dialog.
8. press select output file tape,& then save as.
9. You will have the result you need in the file you save in

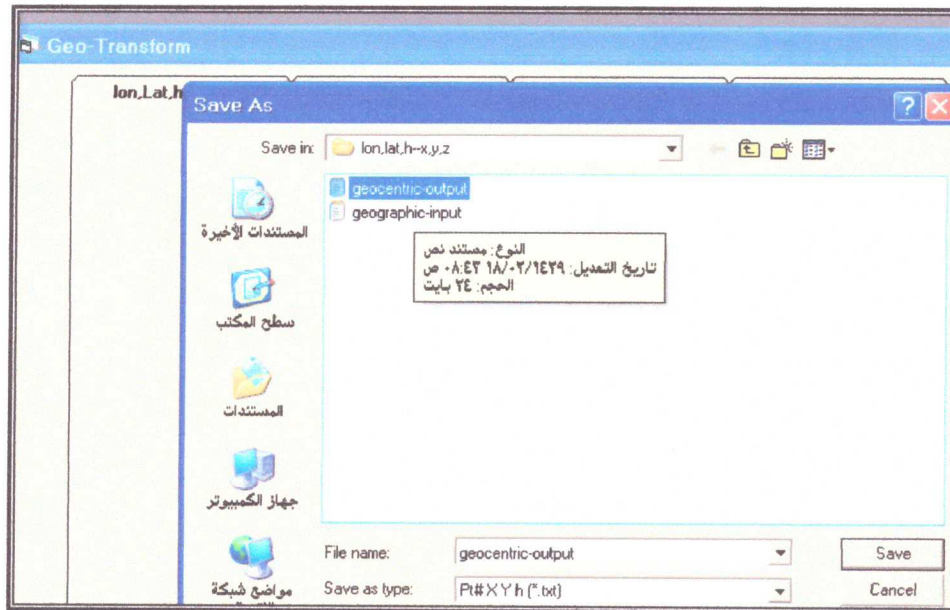
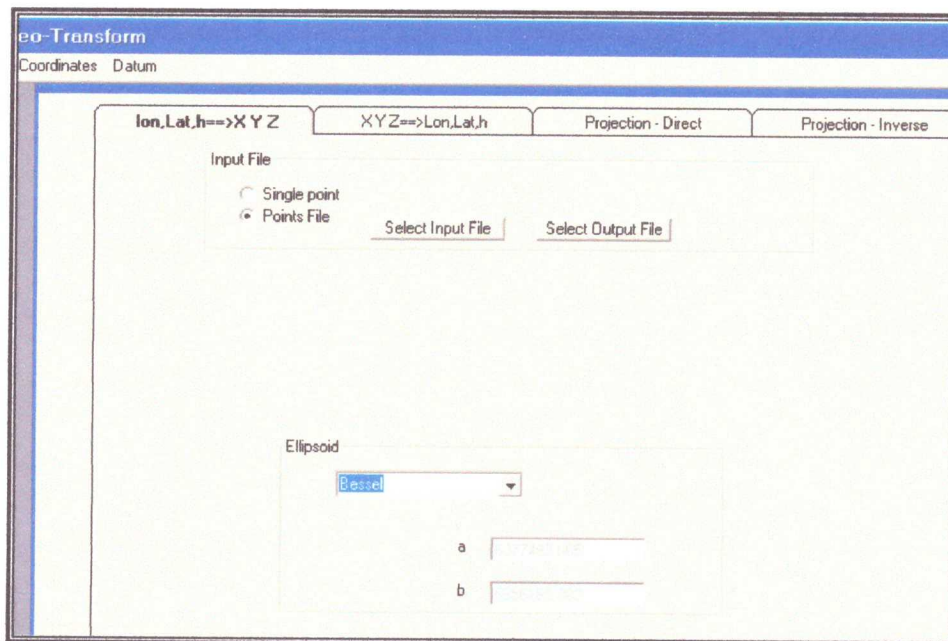


fig (4.15):save as dialog

Example (8):

1. Press lon, lat, h $\rightarrow$ X, Y, Z key.
2. Press point file key.

fig(4.16):lon,lat,h $\rightarrow$ x,y,z dialog

3. Press select input file tape.
4. Select geographic input files, and then click open.

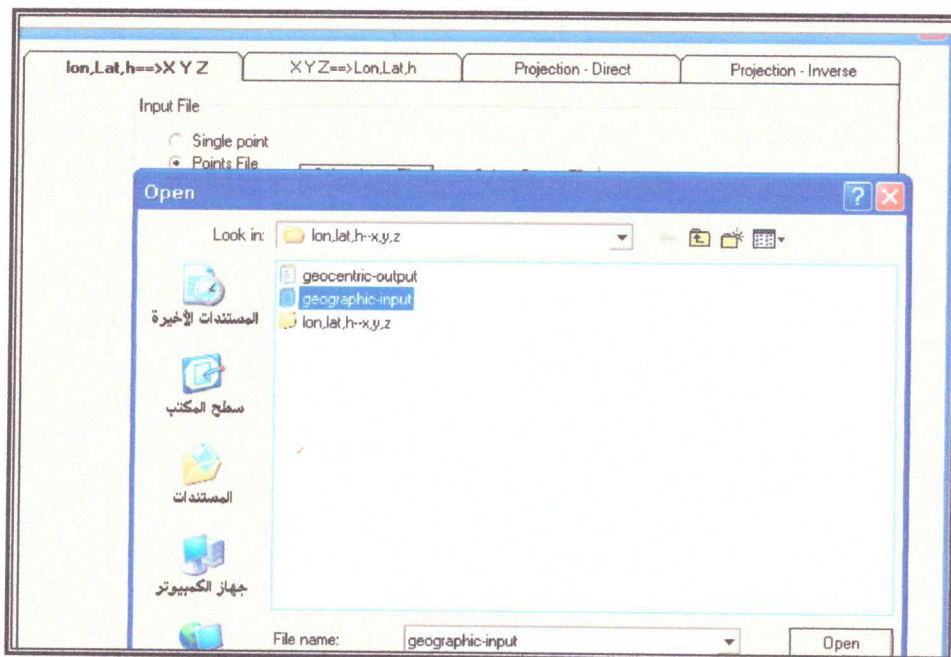


fig (4.17): open dialog

5. You will return to the main diagram, now select Bessel ellipsoid.
6. Press Select output file.
7. You will have this diagram, click geocentric-output, & then save.

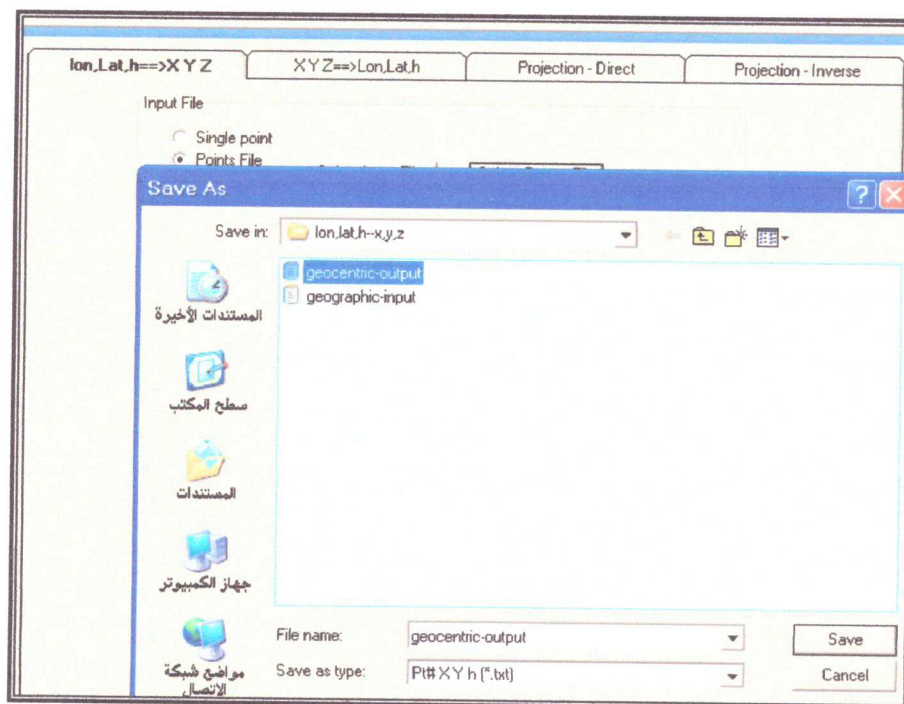


fig (4.18): save as dialog

8. We will have the result in the file we save in., open lon, lat, and h→X, Y, Z file.

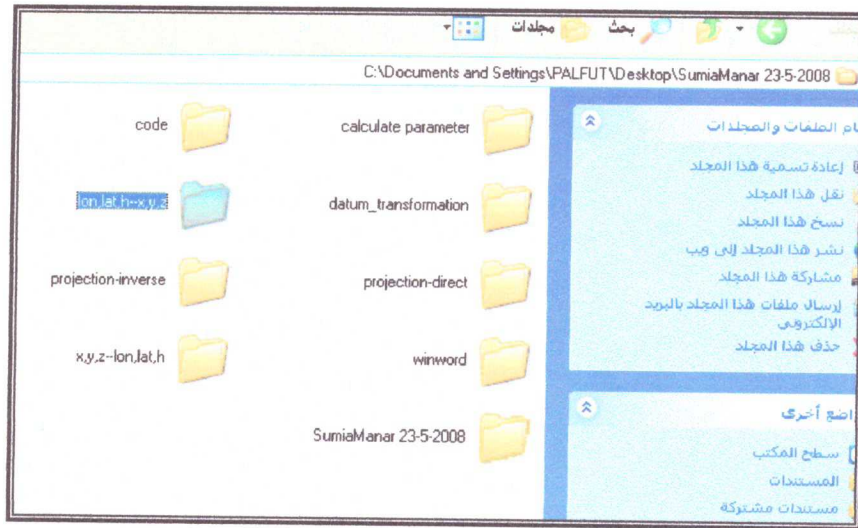
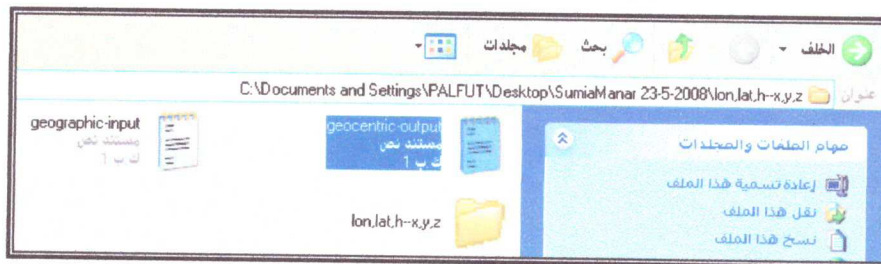


fig (4.19): result dialog

10. Click on the geocentric output, & the result will appear.



fig(4.20):result dialog

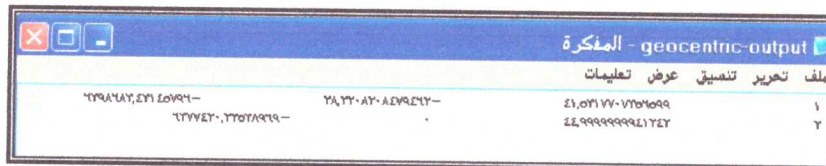


Fig (4.21): example 8

4.2.6 Transform from geocentric coordinates (X, Y, Z) to geographic coordinates (lon, lat, h) for a point file:

1. Press X, Y, Z→lon, lat, h key.
2. In input file select points file.
3. Press select input file.

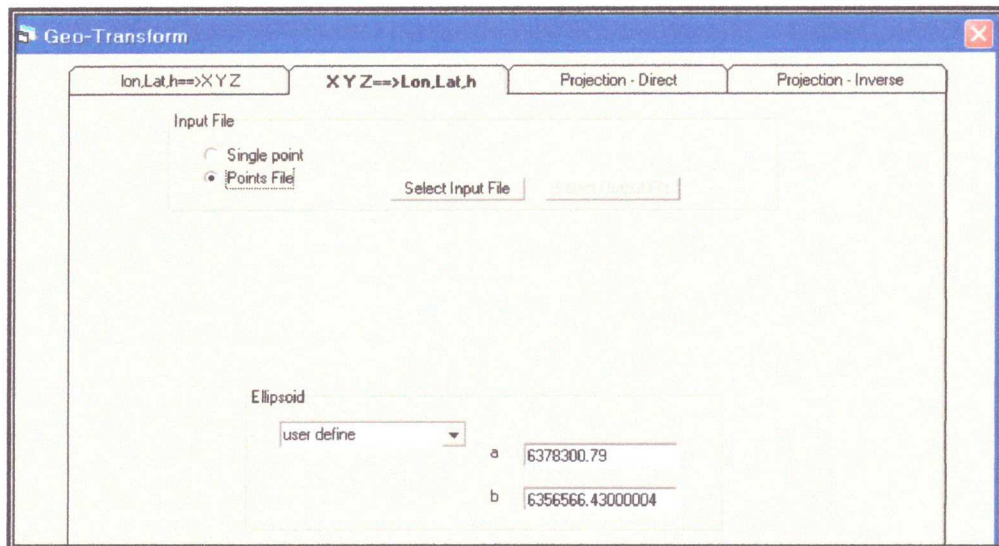


Fig (4.22): coordinate conversions dialog

4. Now you will have this diagram, you should choose $x, y, z \rightarrow \text{lon, lat, h}$, then open.

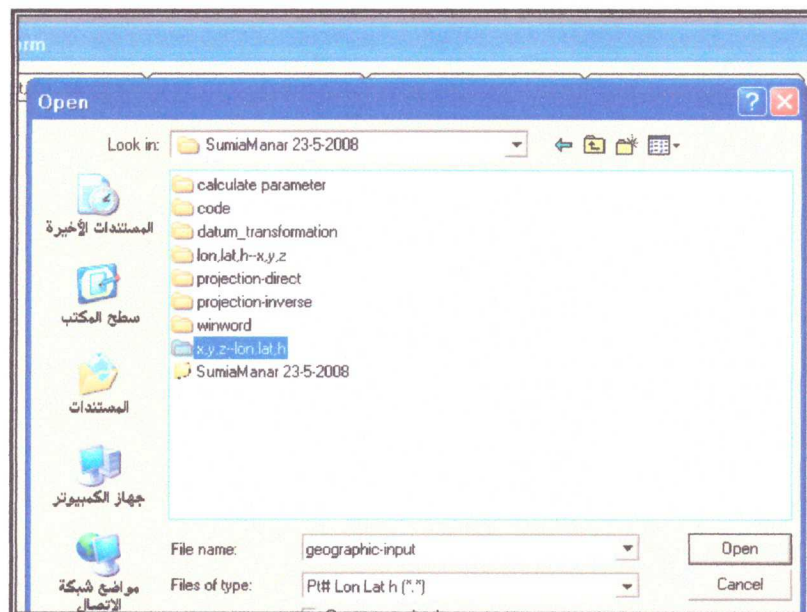
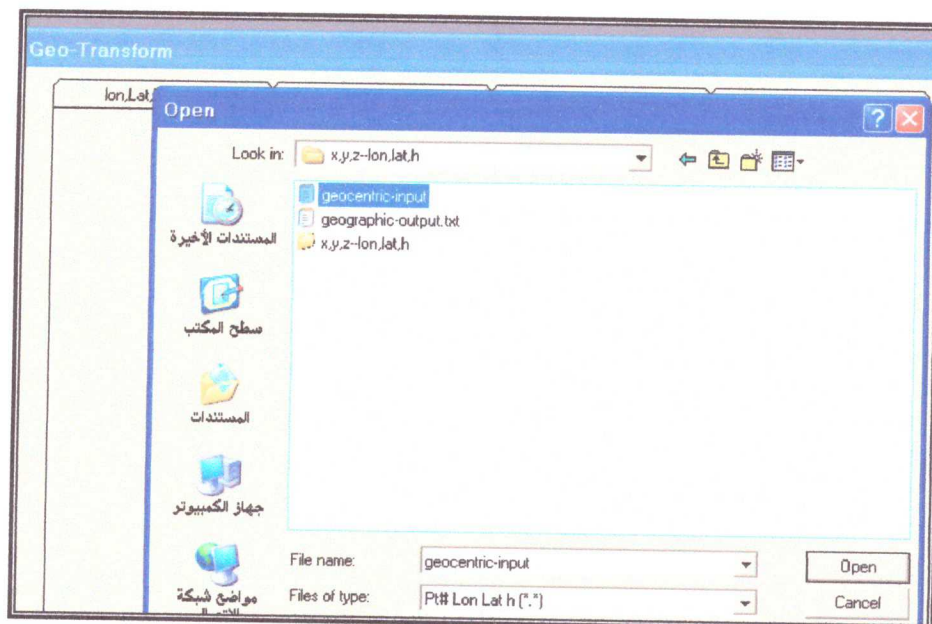


Fig (4.23): open dialog

5. You will have a new diagram, select geocentric input file, then open.



Fig(4.24):open dialog

6. After that you will return to the main dialog, select the ellipsoid you need.
7. Click select output file, & you will have this page. Click in the geographic output file & then save.

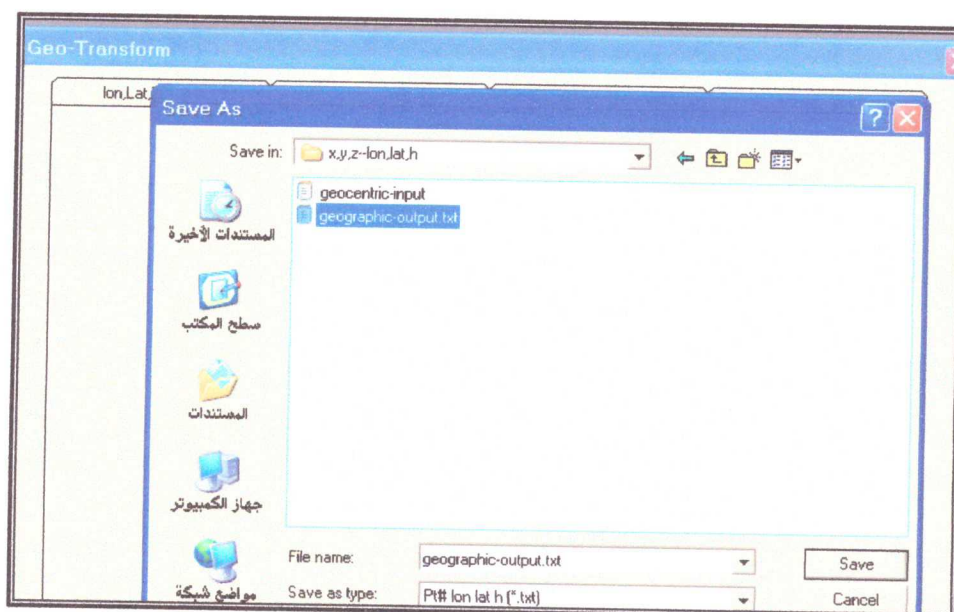
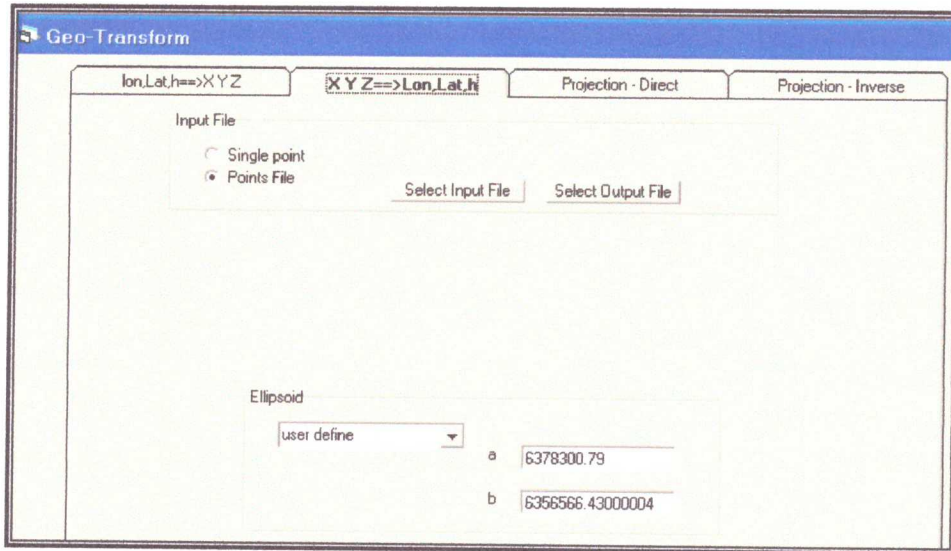


Fig (4.25): save as dialog

8. Now you can see the result in the file you save in.

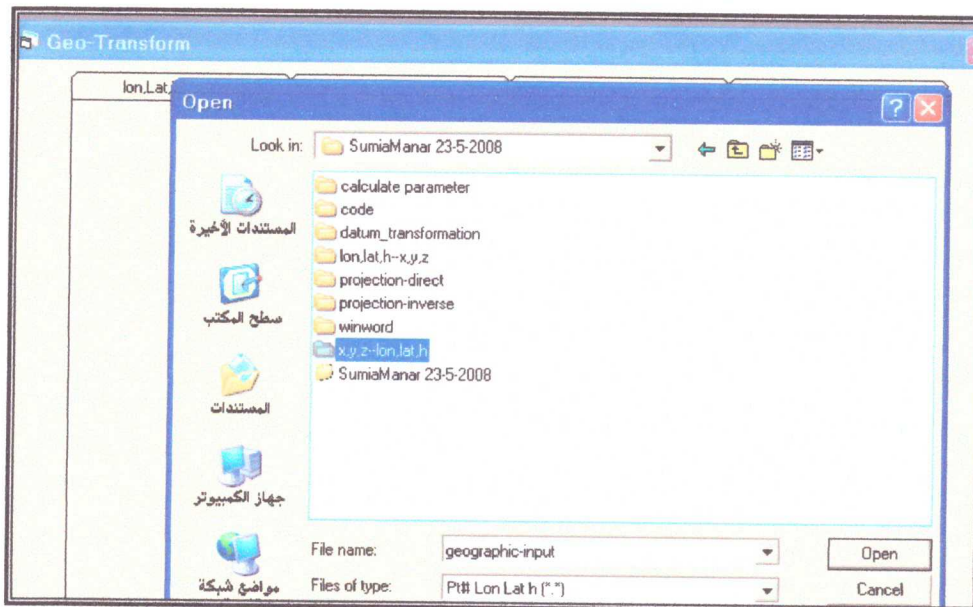
Example 9:

1. Select x, y, z $\rightarrow$ lon, lat, h.
2. Click on points file.



Fig(4.26):geo transform dialog

3. Click on select input file, & you will have this page.



Fig(4.27):open dialog

4. Click open & then you will select geocentric input file, then open.

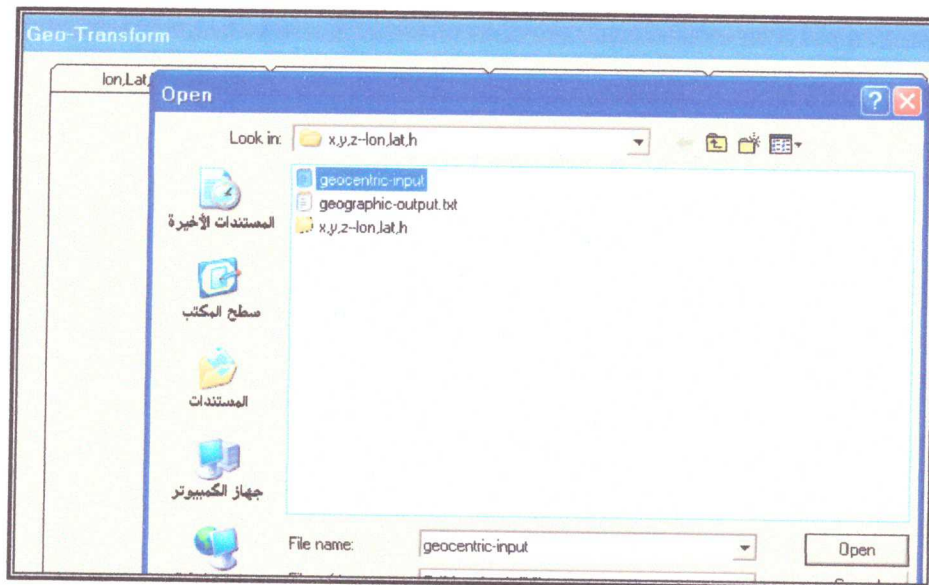


Fig (4.28): open dialog

5. Now you will return to the main diagram, select clarck 1880 ellipsoid.
6. Press select output file.

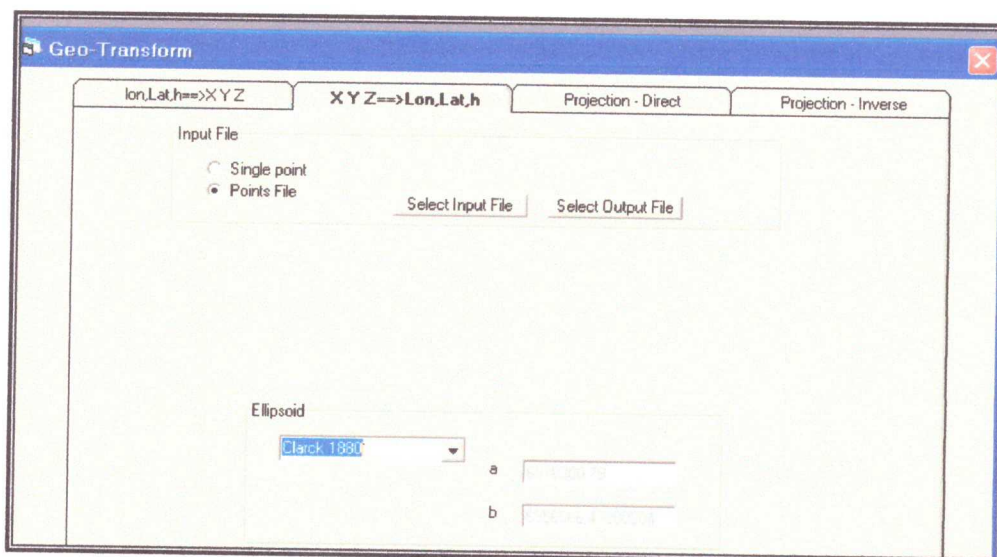


Fig (4.29): geo transform dialog

7. Anew page will appear, click on geographic output file & then save.

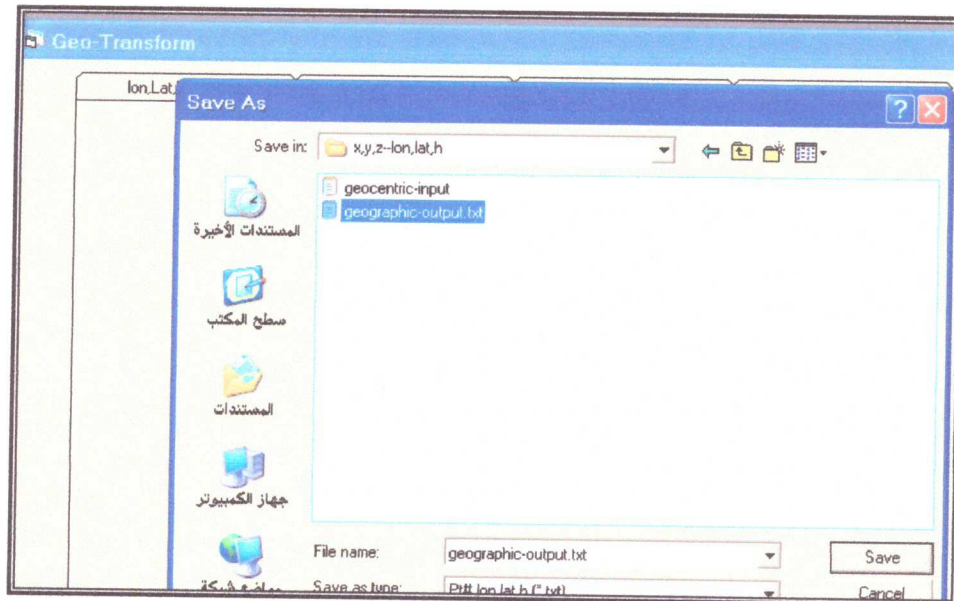


Fig (4.30): save as dialog

8. Now you can see the result from the file you save in like this.

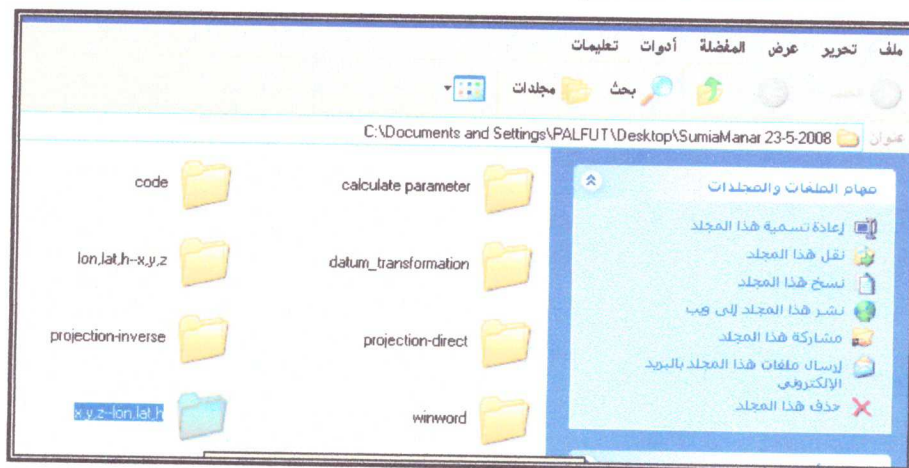


Fig (4.31): results dialog

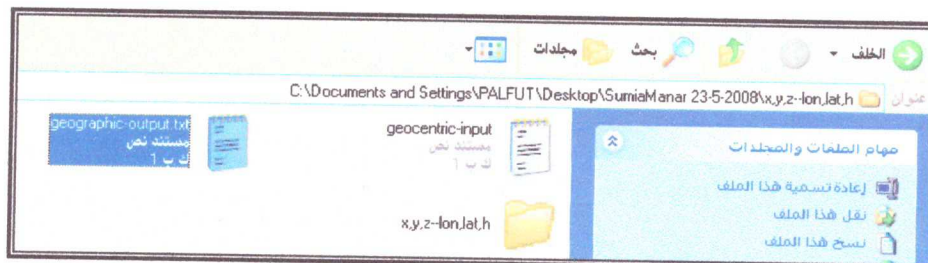


Fig (4.32): results dialog

geographic-output.txt - المفكرة

ملف	تعديل	تحميل
١٠,٧١٠,٤٥٦,٦٧٠,٤٤١	٤٤,٠٠٧,٤٤٦,٧٩,٤٤٦,٤٤٦	٤٤,٩٩٩,٩٩٩,٩٩٩,٠٠١
٦٩٥٩٦,٦٦٦,٦٦٦,٦٦٦,٦٦٦	٤٤,٧٤,٠٠٠,٦٤,٤٤,٦٦٦	٤٤,٩٩٩,٩٩٩,٩٩٩,٠٠٤

Fig (4.33): results dialog

4.2.7 Transform from geographic coordinates(λ, φ) to projected (Easting, Northing) for a point file:

1. Press projection-direct tape.
2. Select point file.

Geo-Transform

lon,Lat,h==>XYZ XYZ==>Lon,Lat,h **Projection - Direct** Projection - Inverse

Input File

☐ Single point
☒ Points File

Select Input File

Coordinate System

UTM zone 39 Datum WGS84

Ellipsoid

GRS80 a b

Projection Parameter

FN lat0 North k0
FE lon0 East

Fig (4.34): geo transform dialog

3. Press select input file, then this page will appear.

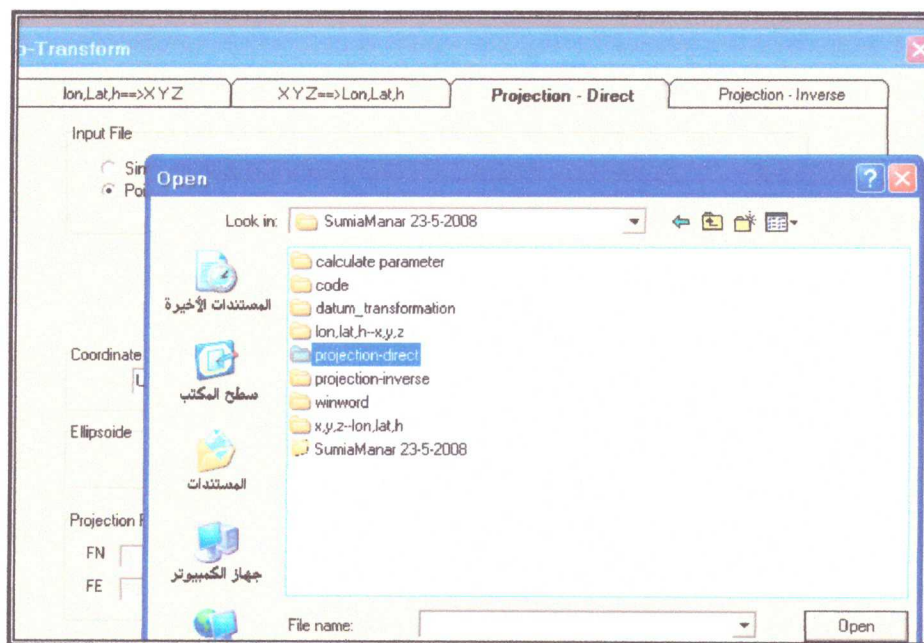


Fig (4.35): open dialog

4. Choose projection-direct folder, & then click open, a new dialog will open.

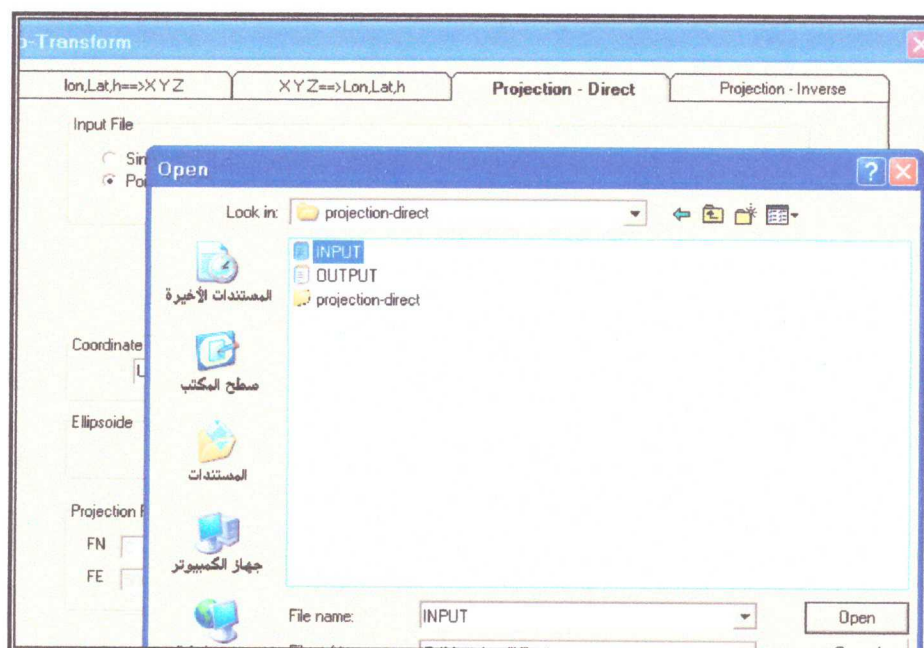


Fig (4.36): open dialog

5. We should choose input file & then open.
6. After that we will return to the main dialog & then select the coordinate system we need & his projection.

7. Then we should select the ellipsoid we need.
8. Now we should enter the projection parameters we have (FE, FN, lat0, lon0, k0).

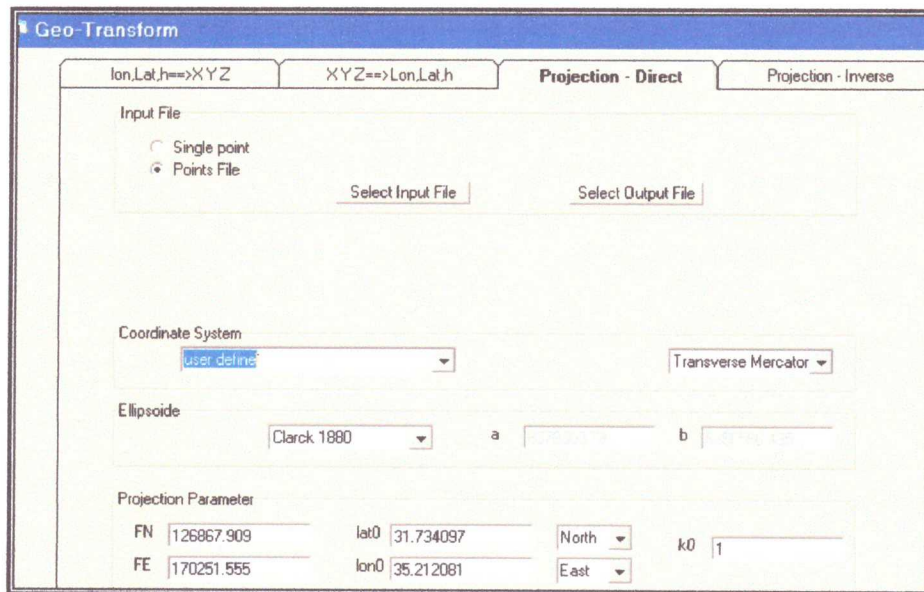


Fig (4.36): geo transform dialog

8. Now you should click select output file & a new dialog will appear, in this dialog click on output, then save key.

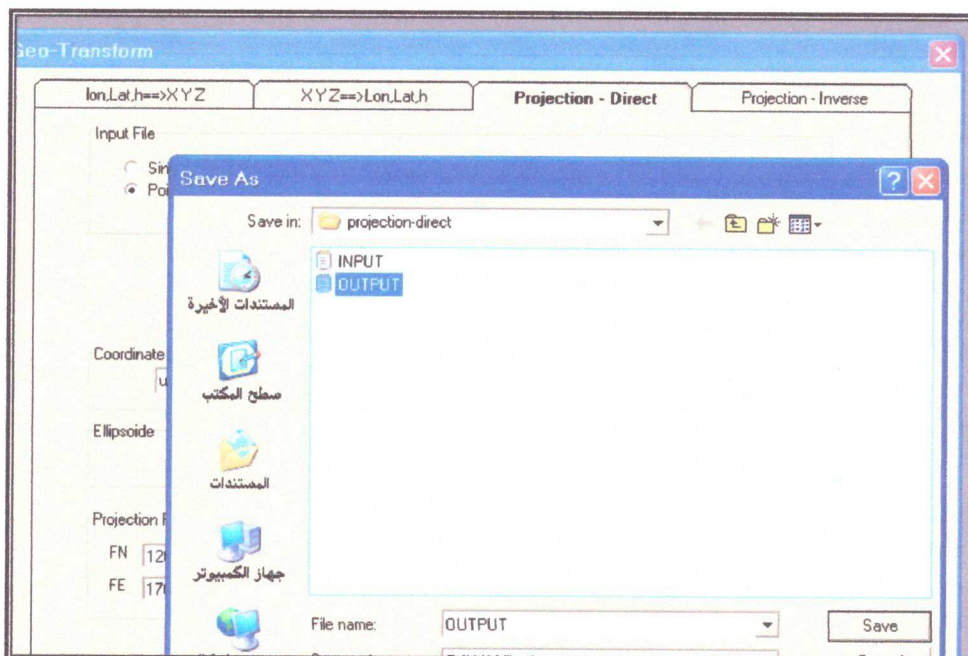


Fig (4.37): save as dialog

9. You can see the result from the file you save in.

Example (10):

1. Press projection-direct key.
2. In input file check point file.
3. Click select input file, & you will open the input file in the projection-direct folder.

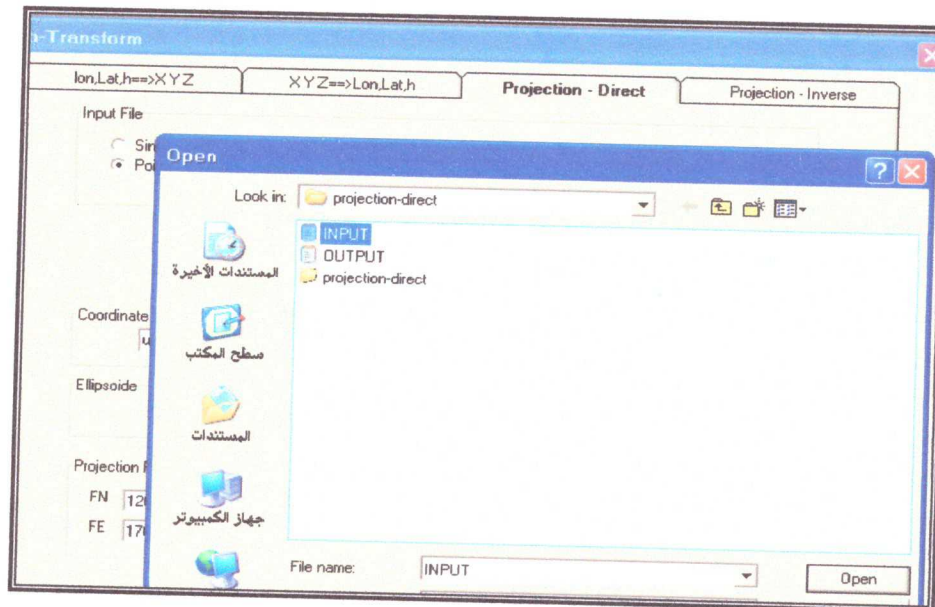


Fig (4.38): open dialog

4. In the main dialog, select UTM coordinate system & enter its zone # = 39, the projection will be transverse Mercator directory.
5. Choose WGS84 ellipsoid.
6. Then enter the value of the projection parameters
 $FE=500000$, $FN=0$, $lat0=0$, $lon0=30$, $k0=0.9996$
7. Now press select output file.

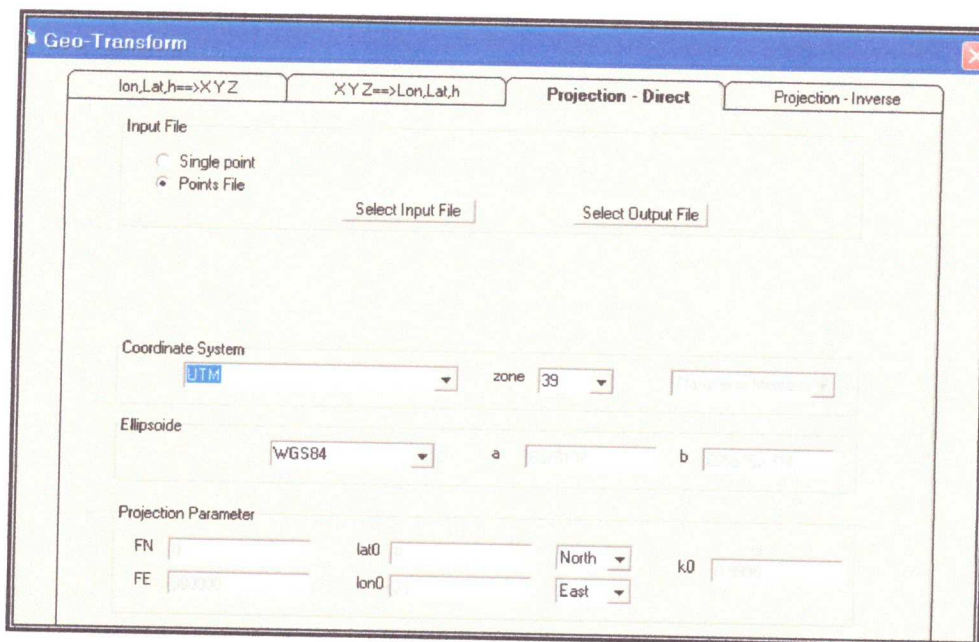


Fig (4.39): geo transform dialog

7. Select output file from projection-direct folder & then click save.

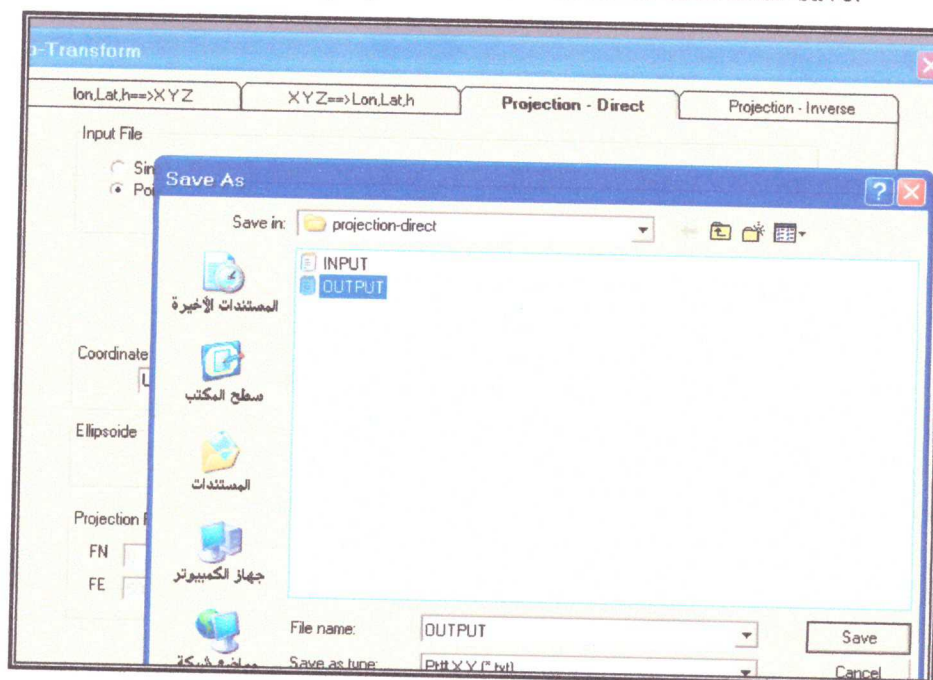


Fig (4.40): save as dialog

8. Now you can see the result from the output file you save, open projection-direct folder, then select output file & you will see the result.

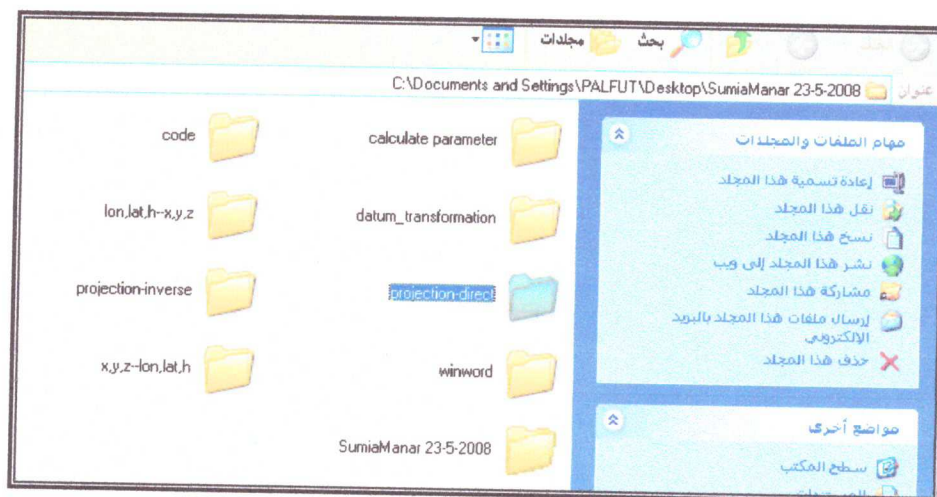
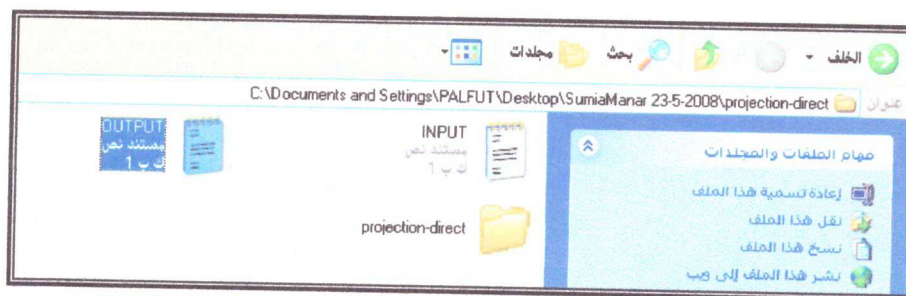


Fig (4.41): results dialog



fig(4.42): results dialog

OUTPUT - المفكرة			
ملف	تحرير	تنسيق	عرض تعليمات
YLA7891,1A780941	٦٩٨٢٩٥,٢٥٤٦٩٨١٢٢	١	
YLA7890,01007810	٦٩٨٢٩٤,٧٨٢٩٩٨٣٢٢	٢	

Fig (4.43): results dialog

4.2.8 Transform from projected coordinates (Easting, Northing) to geographic coordinates (λ , ϕ):

1. Press projection-inverse key.
2. In input file check point file.
3. Choose select input file.

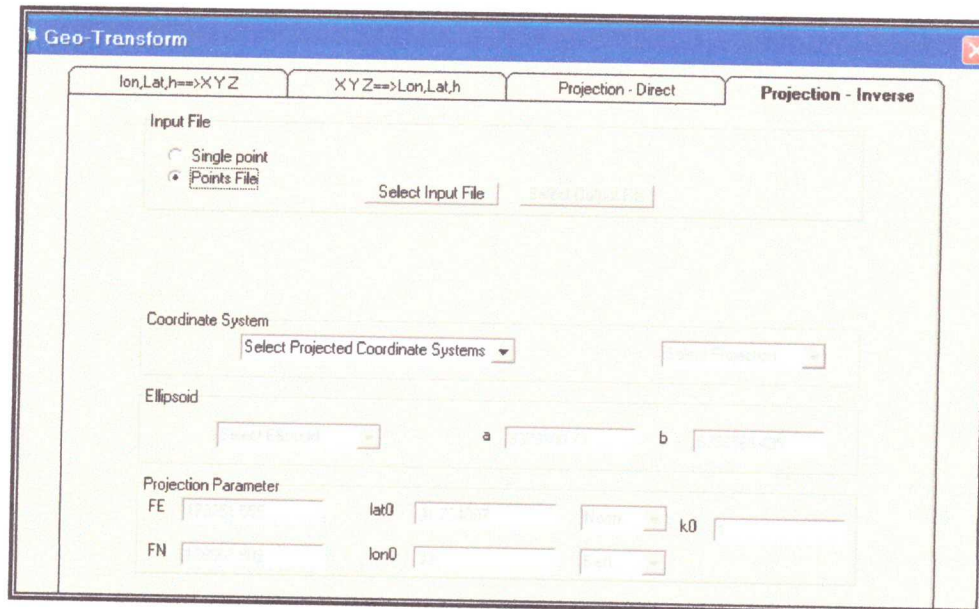


Fig (4.44): geo transform dialog

4. A new dialog will appear, select projection-inverse from it, & then open.
5. Select input file from it & then click open.

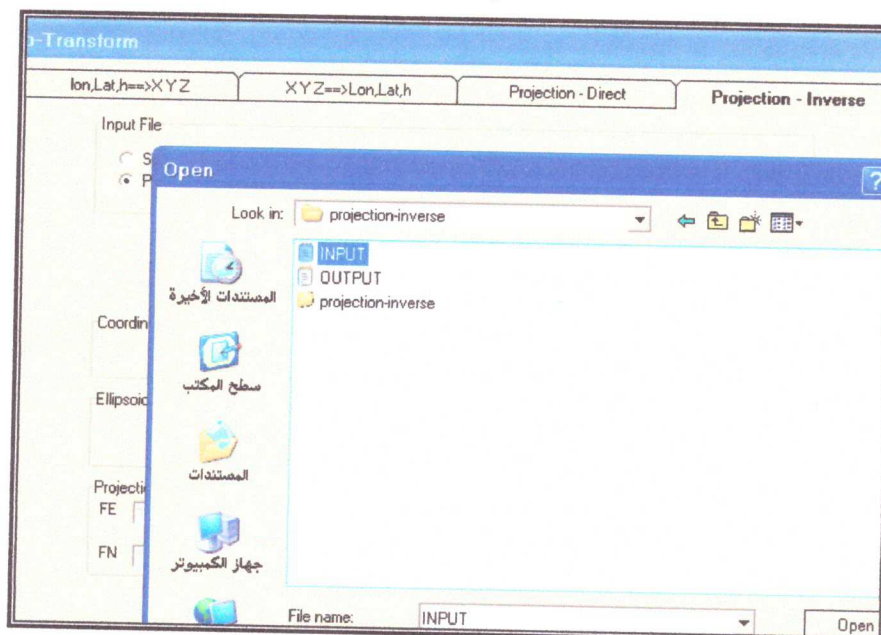


Fig (4.45): open dialog

6. Now you will return to the main dialog, select the coordinate system you need & its projection.
7. Then select the ellipsoid you need & its parameter will enter a lone.
8. Enter the projection parameters you have (FE, FN, lat0, lon0, k0).

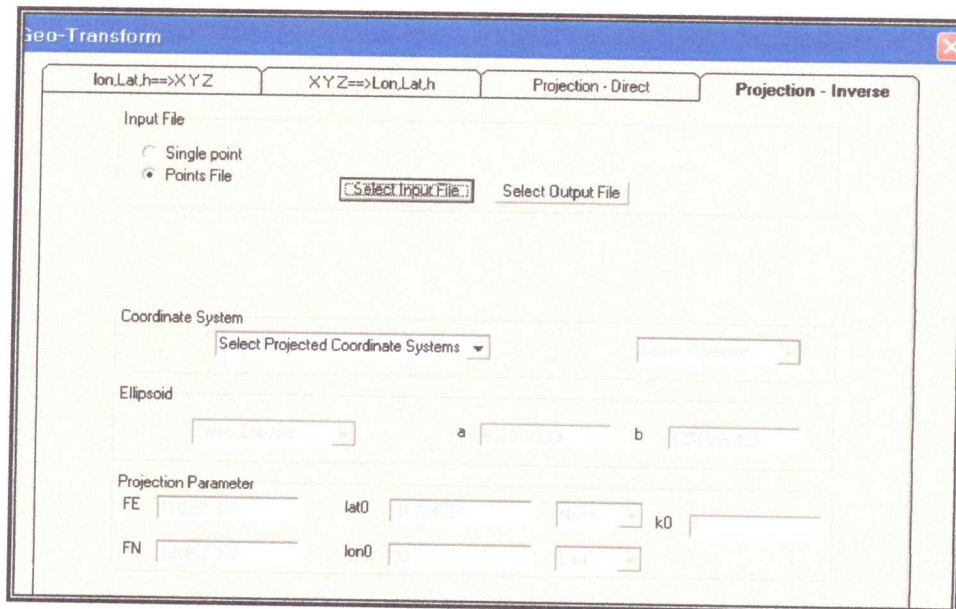


Fig (4.46): geo transform dialog

9. Press select output file.
10. Anew page will appeare, select output file & then click save.

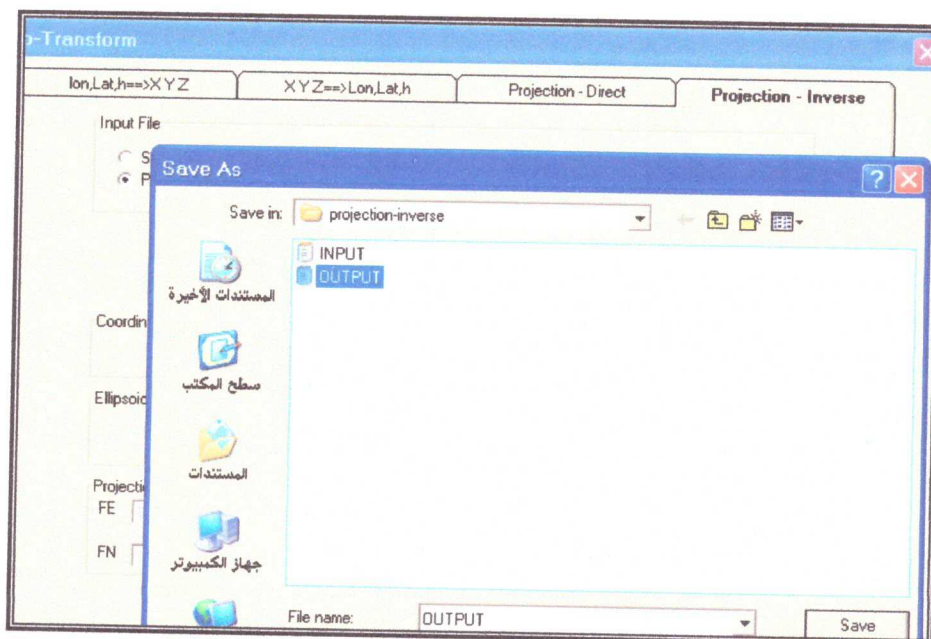


Fig (4.47): save as dialog

Example (11):

1. Select projection-inverse tape.
2. Check point file.
3. Press select input file.

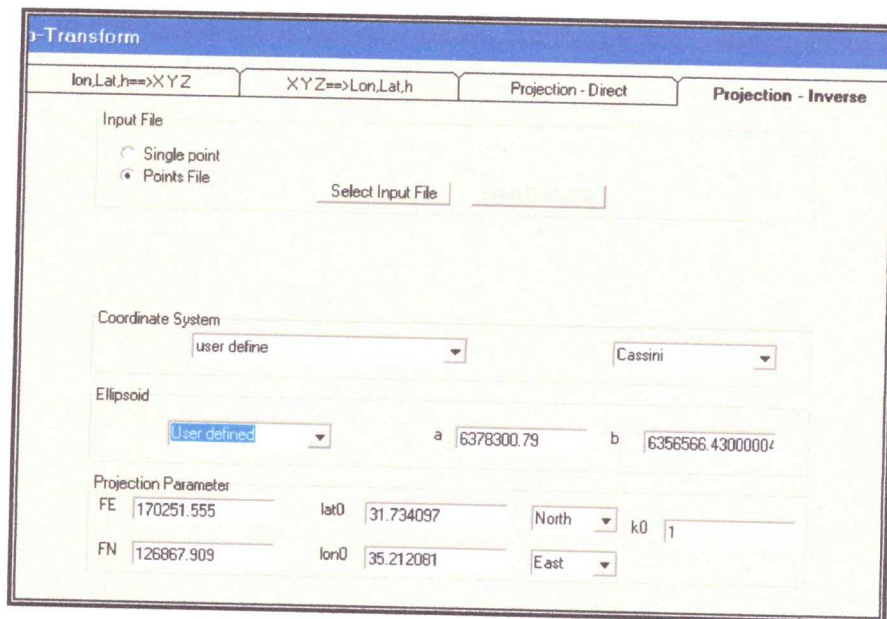


Fig (4.48): geo transform dialog

4. Press input from projection-inverse folder & then clicks open.

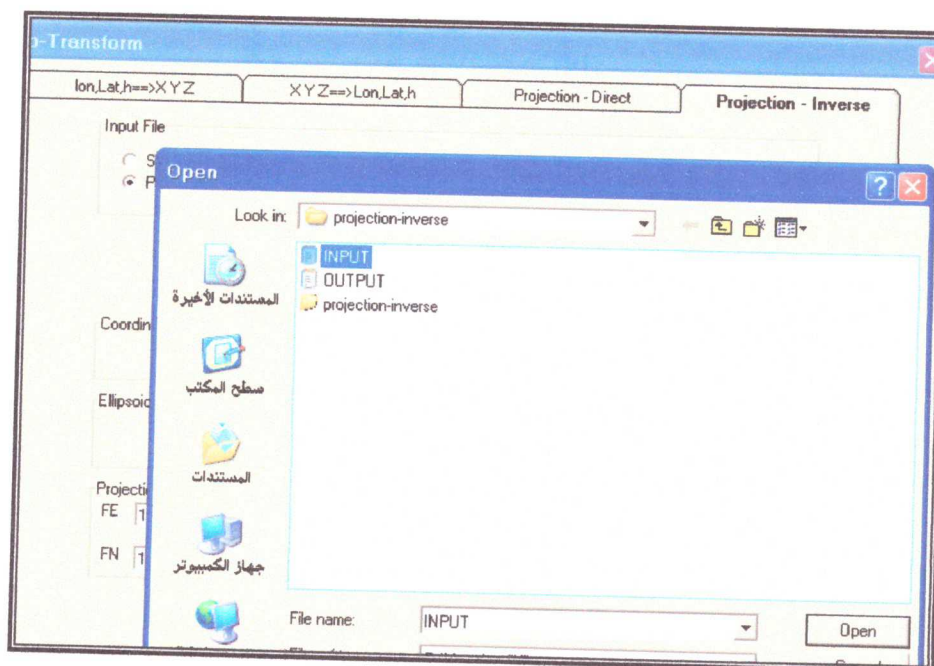


Fig (4.48): open dialog

5. In the main screen select User define from coordinate system & Cassini projection.
6. Select User define from ellipsoid & enters its parameters
 $a=6378300.79$, & $b=6356566.43000$
7. Enter the value of the projection parameters as:
 $FE=170251.555$, $FN=126867.909$,
 $Lat_0=31.734097$, $lon_0=35.212081$, $k_0=1$
8. Now press select output file, then a new page a rise to you.
9. Select output file from projection-inverse folder.

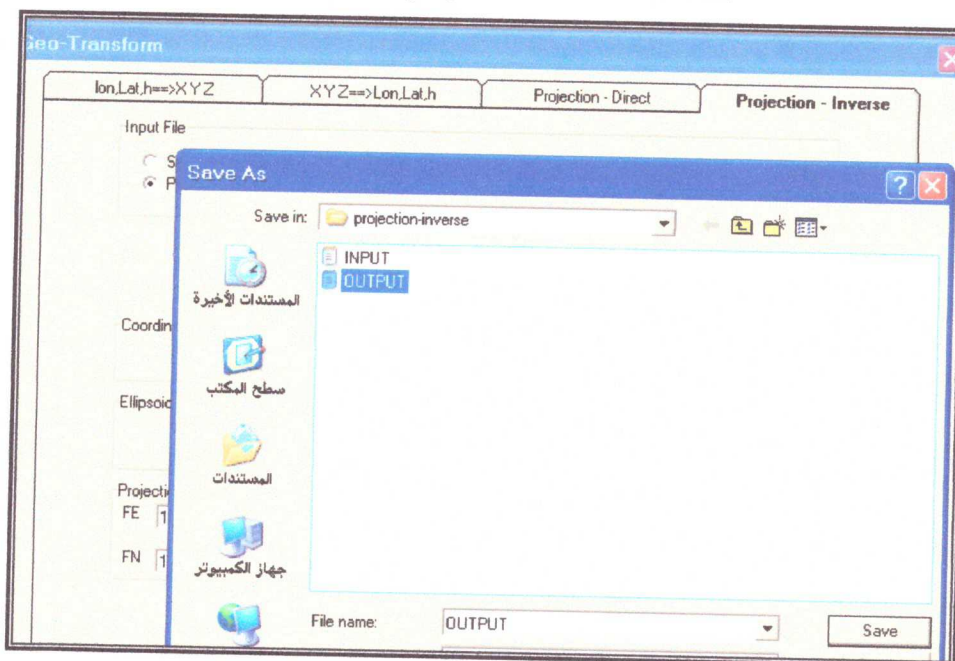
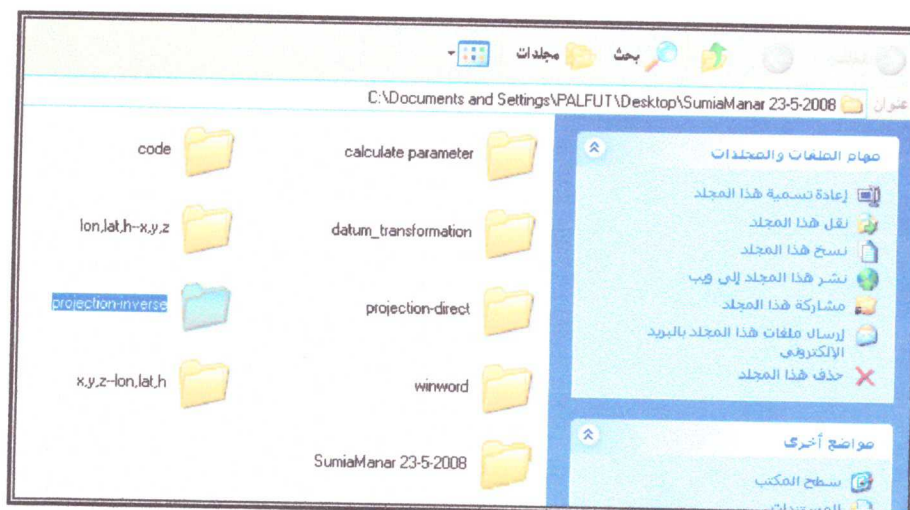


Fig (4.49): save as dialog

- 10- You can see the result from the file you save in.



fig(4.50): results dialog

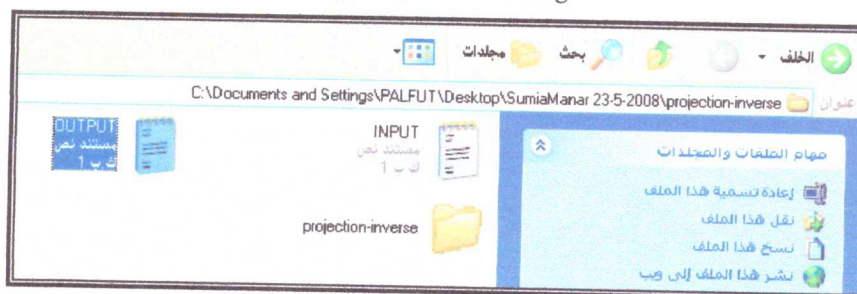


fig (4.51): results dialog

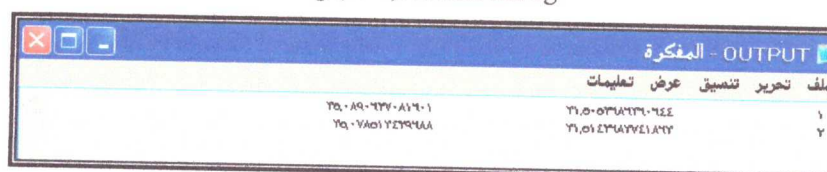
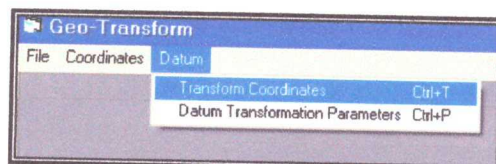


fig (4.52): results dialog

4.3 Starting geo transform for using datum transformation:

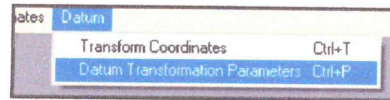
1. Open the geo transform program.



fig(4.53): datum menu

2. Click on datum tape & then select transform coordinates.

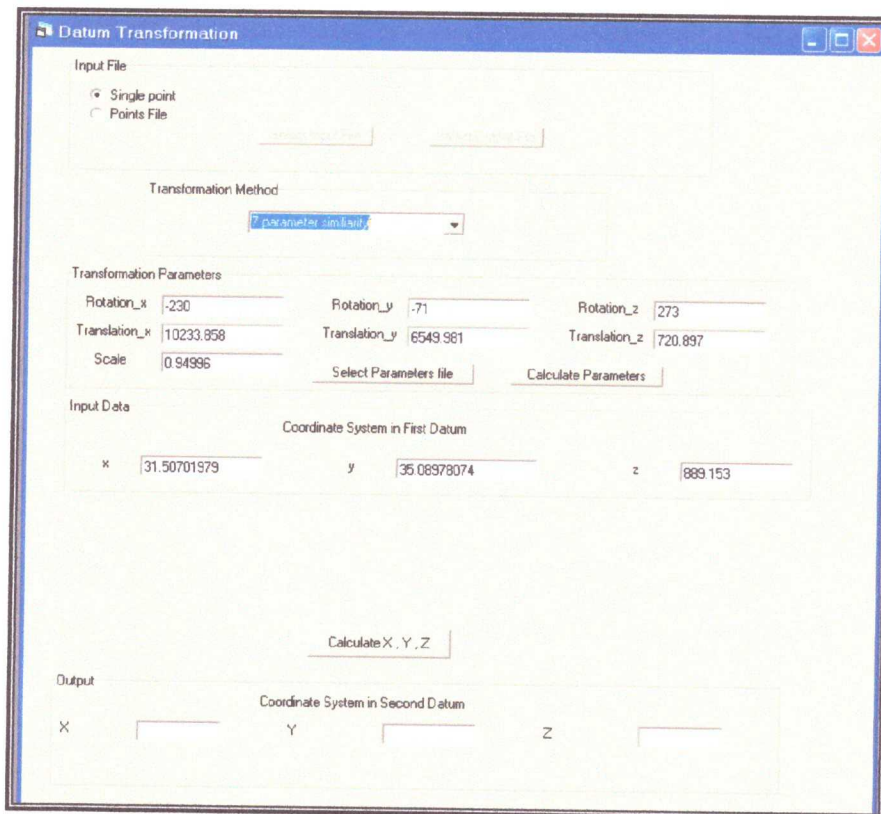
If you want to calculate the parameters:



fig(4.54):datum menu

To calculate the parameters there is two ways:

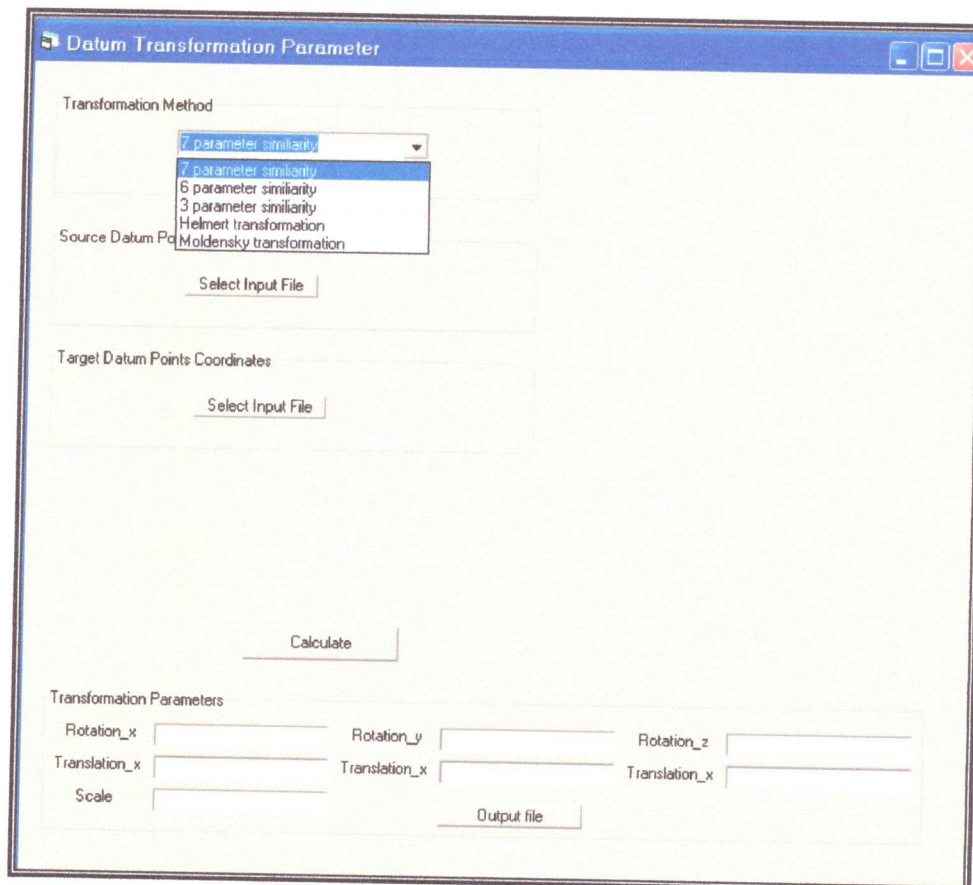
2. From datum menu press datum transformation parameters.
3. From transformation parameters click calculate parameters.



fig(4.55):datum transformation dialog

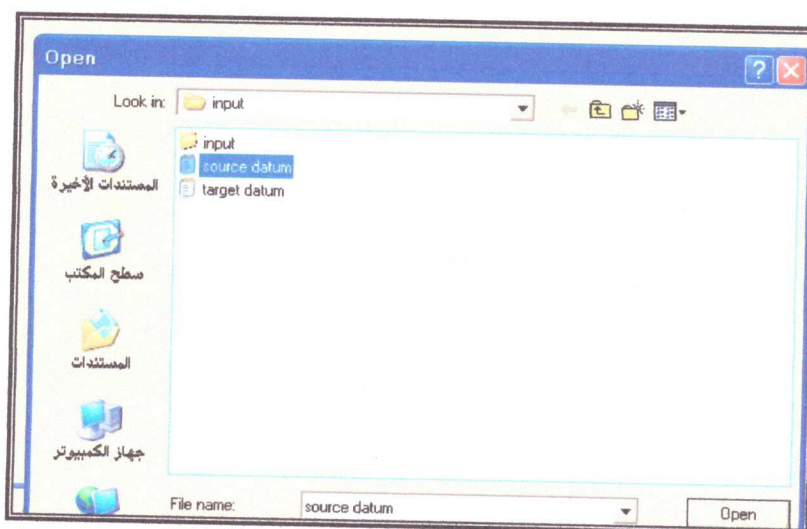
In both ways the result will be the same so you can use any one.

4. A new dialog will appear like in figure (4.56), now select the transformation method you need.



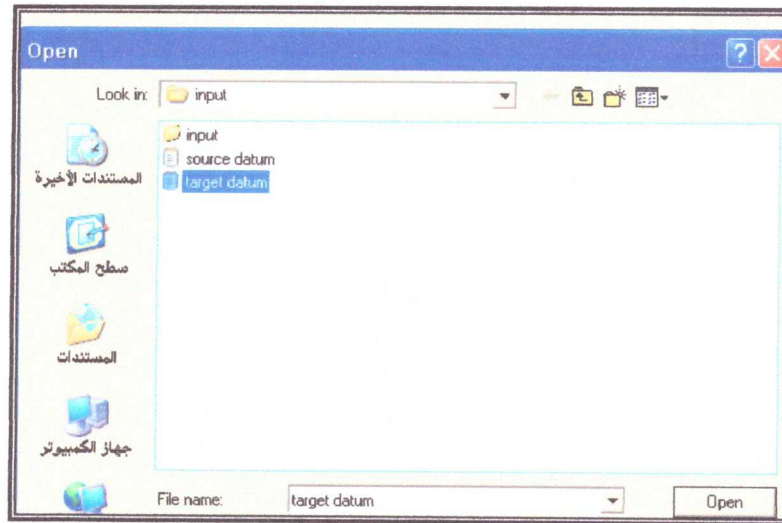
fig(4.56):datum transformation parameter dialog

5. Press on select input file (source datum points coordinates) then open dialog will appear.
6. In open dialog select source datum file & then press open.



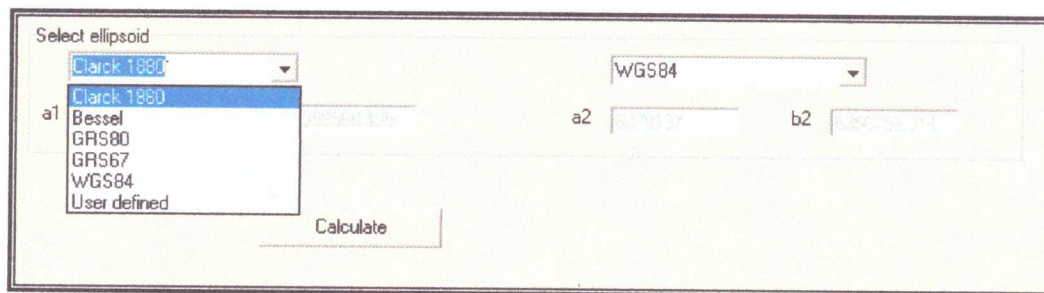
fig(4.57):open dialog

7. After that press on select input file(target datum points coordinates).
8. Select target datum file from open dialog &then press open.



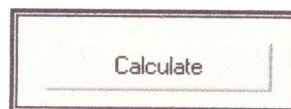
fig(4.58):open dialog

9. In moldensky transformation, you will select the first & the second ellipsoid from the main dialog.



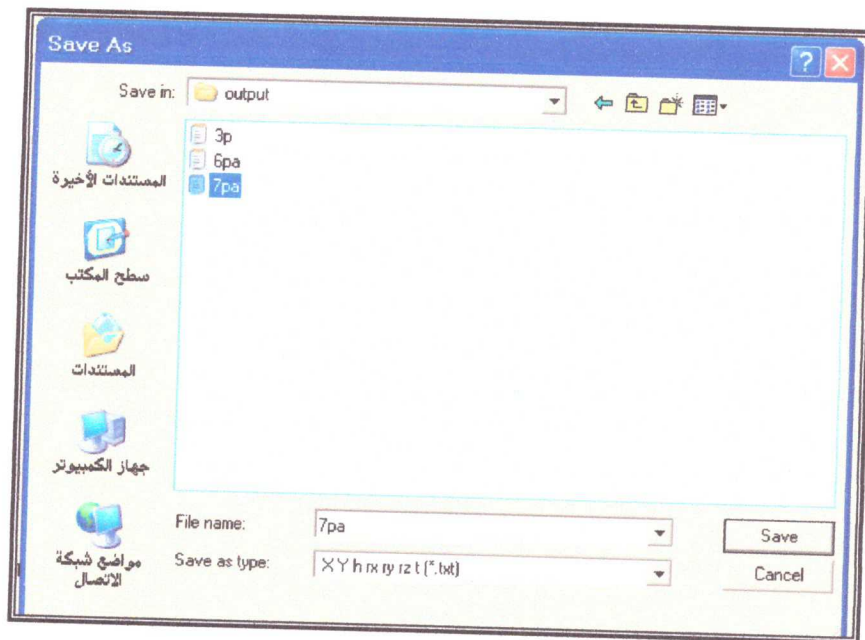
fig(4.59):select ellipsoid list

10. Now press calculate key from datum transformation parameter dialog.



fig(4.60):calculate key

11. Select output file to save the parameter in file.



fig(4.61):save sa dialog

4.3.1 Transform using 7 parameter similarity for a single point:

1. Choose a single point.

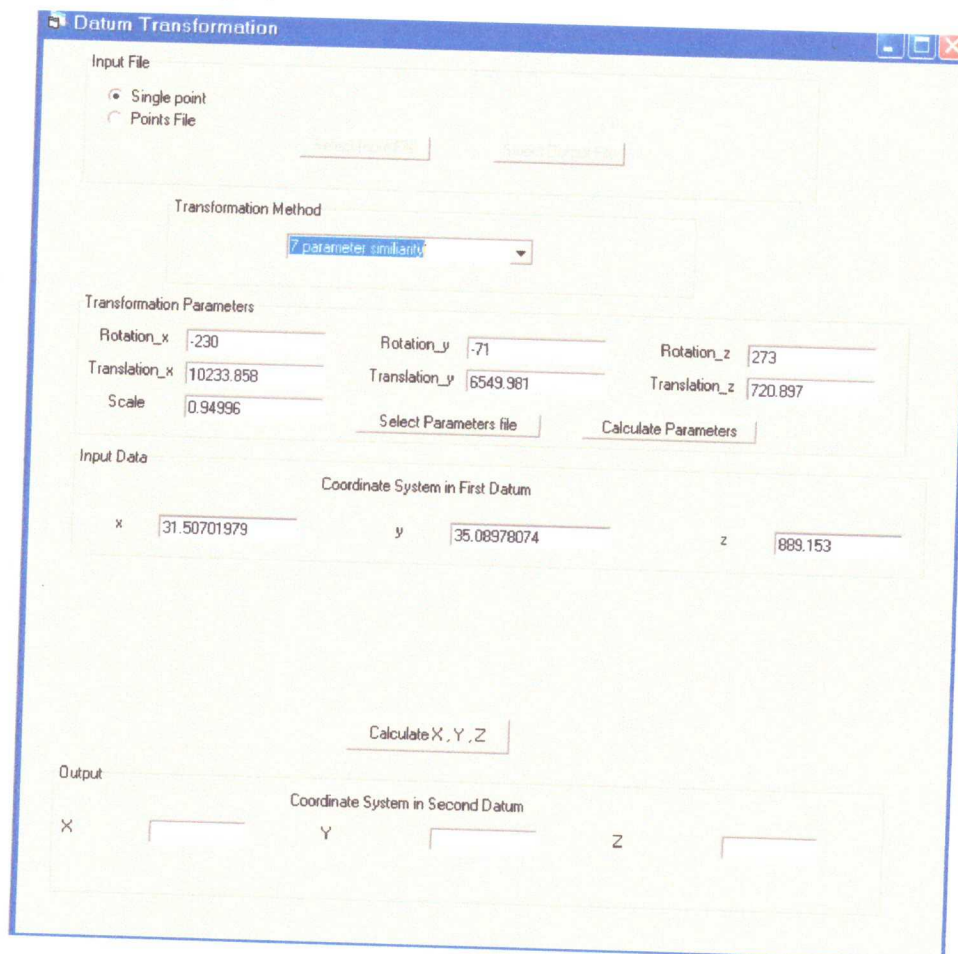
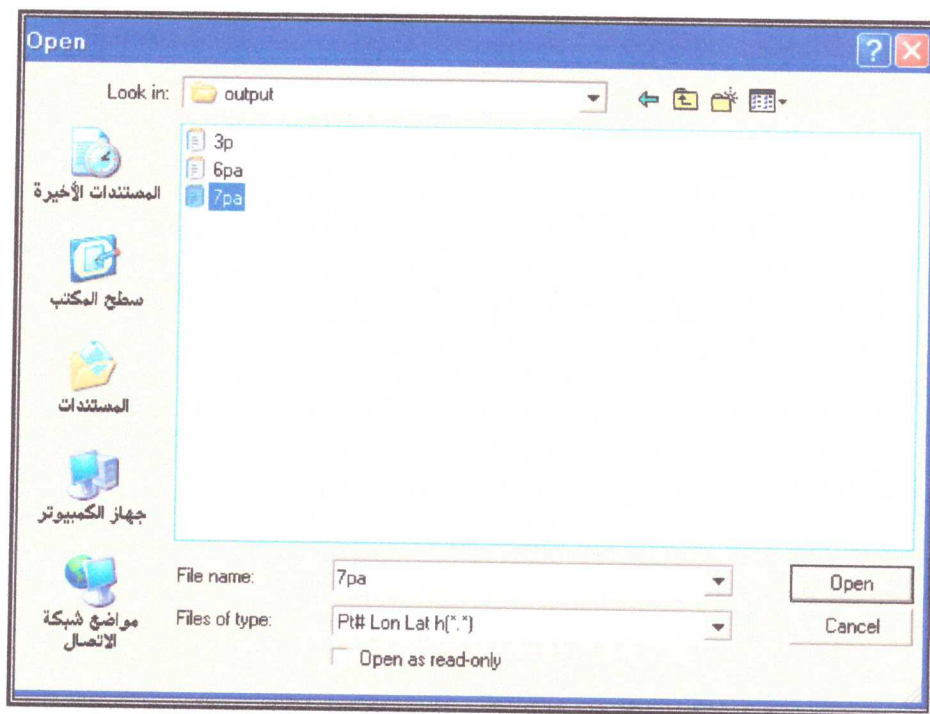


fig (4.62): datum transformation dialog

2. In transformation method select 7 parameter similarity.
3. Enter the value of the seven transformation parameters if you have it.
4. Enter the value of input data (x, y, and z) in the first datum.
5. Press calculate X, Y, Z key.

Example (12):

1. Choose a single point.
2. From transformation methods select 7-parameter similarity.
3. Enter the value of the seven transformation parameters or you can enter it from pressing select parameters file that you save in calculate parameters.



fig(4.63):open dialog

4. Enter the value of input data (x, y, and z) in the first datum as it appears in fig .5. Press calculates X, Y, Z key.

Datum Transformation

Input File
☒ Single point
☐ Points File

Transformation Method
 7 parameter similarity

Transformation Parameters
 Rotation_x: 2.28412825698399
 Rotation_y: -0.55080113488997
 Rotation_z: 224.53709869642
 Translation_x: 10233.8257617577
 Translation_y: 6549.9683394476
 Translation_z: 720.87965609007
 Scale: 0.949956940242124
 Select Parameters file
 Calculate Parameters

Input Data
 Coordinate System in First Datum
 x: 1094.883
 y: 820.065
 z: 109.821

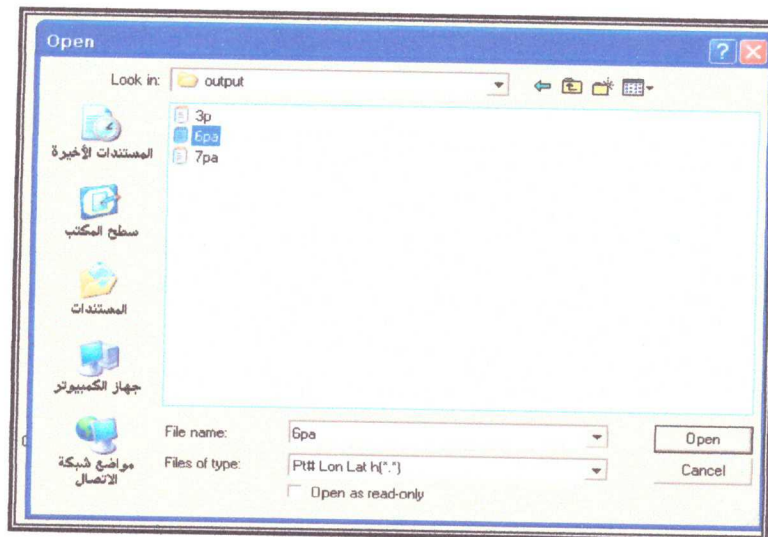
Calculate X, Y, Z

Output
 Coordinate System in Second Datum
 X: 10037.858021
 Y: 5262.11521711
 Z: 772.03900588

fig(4.64): 7-parameter similarity dialog

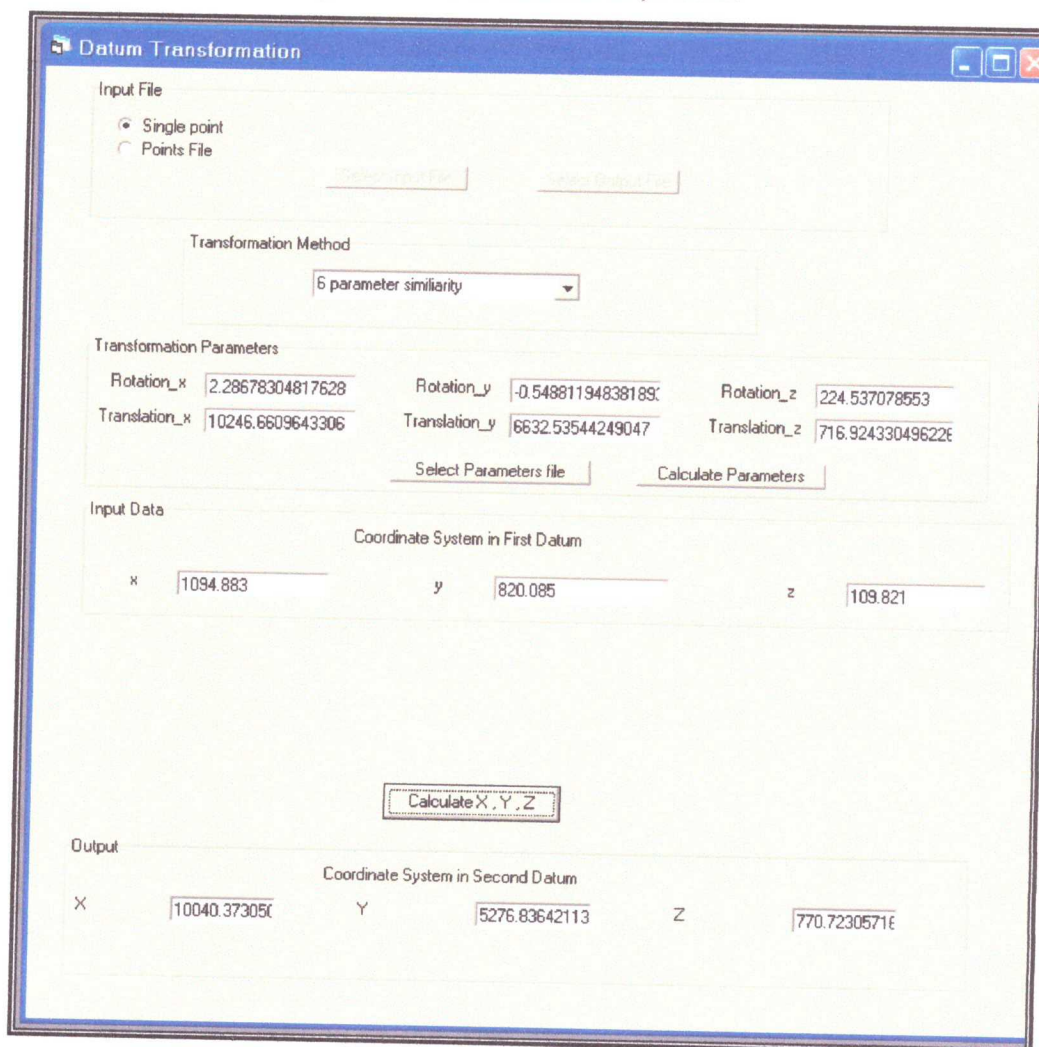
4.3.2 Transform using 6 parameter similarity for a single point:

1. Choose a single point.
2. In transformation method select 6 parameter similarity.
3. Enter the value of the six transformation parameters or you can enter it from pressing select parameters file that you save in calculate parameters it.



fig(4.65):open dialog

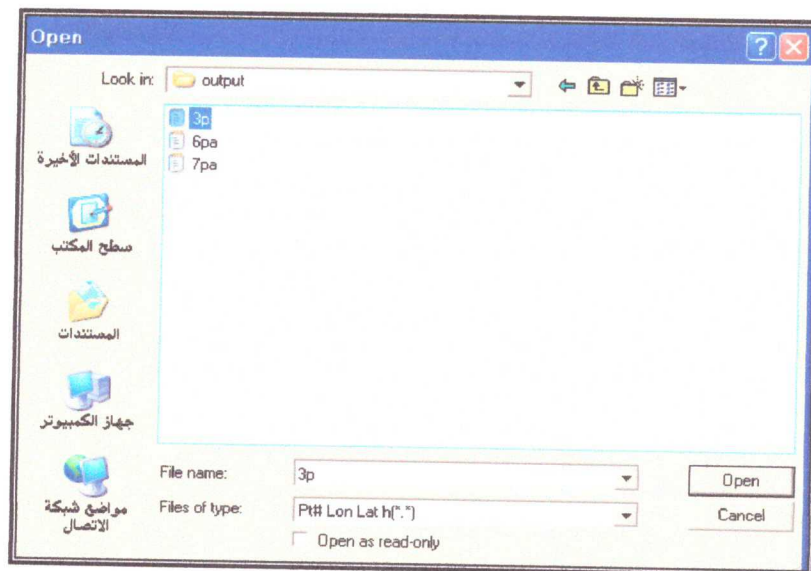
4. In input data enter the value of x, y& z for the first datum.
5. Press calculate key to show the value of X ,Y & Z.



fig(4.66):6 parameter similarity dialog

4.3.3 Transform using 3 parameter similarity for a single point:

1. Choose a single point.
2. In transformation method select 3 parameter similarity.
3. Enter the value of the three transformation parameters or you can enter it from pressing select parameters file that you save in calculate parameters.



fig(4.67):open dialog

4. In input data enter the value of x, y& z for the first datum.
5. Press calculate key to show the value of X, Y & Z.

The screenshot shows a software window titled "Datum Transformation". It contains several sections for configuring a coordinate transformation:

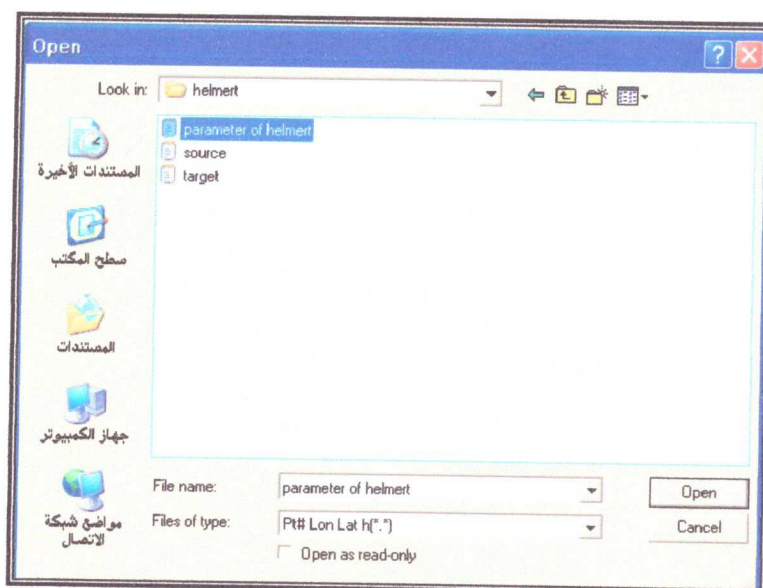
- Input File:** Two radio buttons are present: "Single point" (which is selected) and "Points File".
- Transformation Method:** A dropdown menu is set to "3 parameter similarity".
- Transformation Parameters:** Three text input fields are shown: "Translation_x" with the value "9990.0925", "Translation_y" with "4982.69", and "Translation_z" with "797.245". Below these are two buttons: "Select Parameters file" and "Calculate Parameters".
- Input Data:** A section labeled "Coordinate System in First Datum" contains three input fields: "x" with "1094.883", "y" with "820.085", and "z" with "109.821".
- Output:** A section labeled "Coordinate System in Second Datum" contains three input fields: "X" with "11084.9755", "Y" with "5802.775", and "Z" with "907.066".

A "Calculate X,Y,Z" button is located between the input and output sections.

fig(4.68): 3 parameter similarity dialog

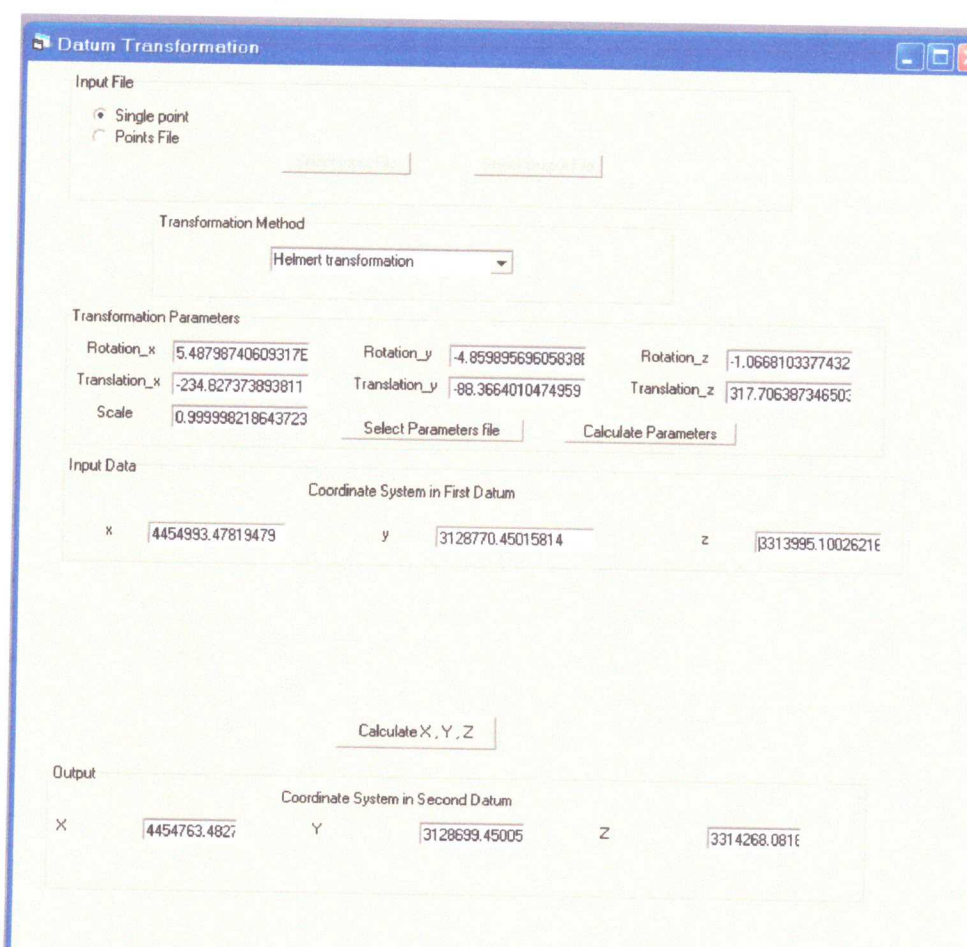
4.3.4 Transform using Helmert transformation:

1. Choose a single point.
2. In transformation method selects Helmert transformation.
3. Enter the value of the Helmert transformation parameters or you can enter it from pressing select parameters file that you save in calculate parameters.



fig(4.69): open dialog

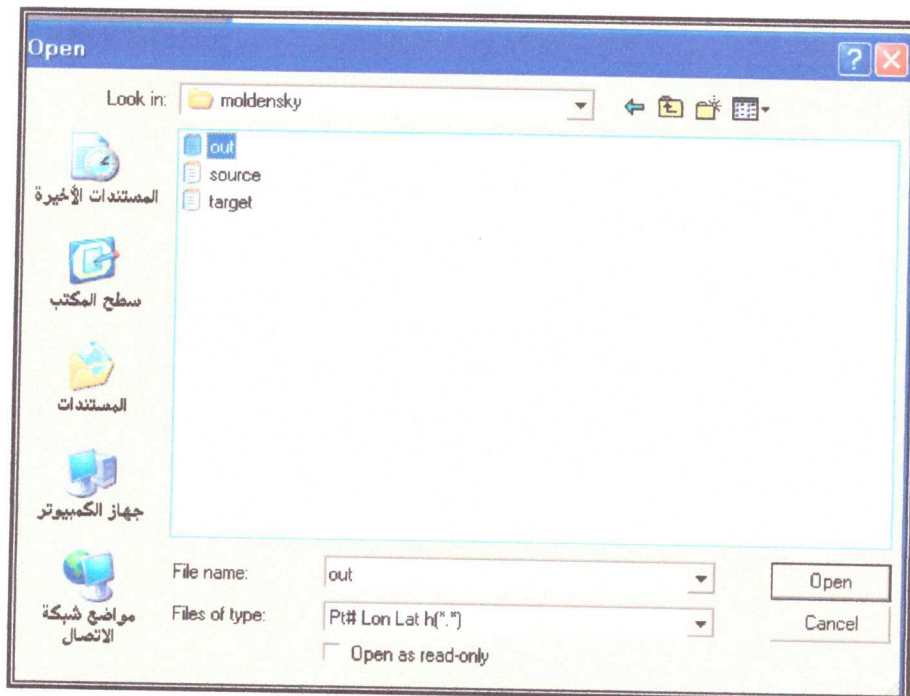
4. In input data, enter the value of x, y, z of the first datum.
5. Press calculate key to show the results.



fig(4.70):Helmert transformation dialog

4.3.5 Transform using Meldensky transformation for a single point:

1. Choose a single point.
2. In transformation method select meldensky transformation.
3. Enter the value of the meldensky transformation parameters (T_x , T_y , T_z) or you can enter it from pressing select parameters file that you save in calculate parameters.



fig(4.71): open dialog

4. In input data, enter the value of (lat,lon,h) of source datum.
5. Select the ellipsoid of source & target datum.
6. Press calculate key to show the results.

Datum Transformation

Input File
☒ Single point
☐ Points File

Transformation Method
 Moldensky transformation

Transformation Parameters
 Translation_x: -230 Translation_y: -71 Translation_z: 273
 Select Parameters file Calculate Parameters

Input Data
 Coordinate System in First Datum
 lat: 31.50504892 lon: 35.08061354 h: 880.663

Select ellipsoid
 Clark 1880 WGS84
 a1: b1: a2: b2:

Calculate X, Y, Z

Output
 Coordinate System in Second Datum
 Delta latitude: 76.87877487 Delta longitude: 160.830173078 Delta Height: 15.696144807
 latitude: 31.50542163 longitude: 35.0813932666 h: 896.359144807

fig(4.72): Moldensky transformation dialog

4.3.6 Transform using 7-parameters similarity for points file:

1. Click on points file.
2. Press on select input file

Datum Transformation

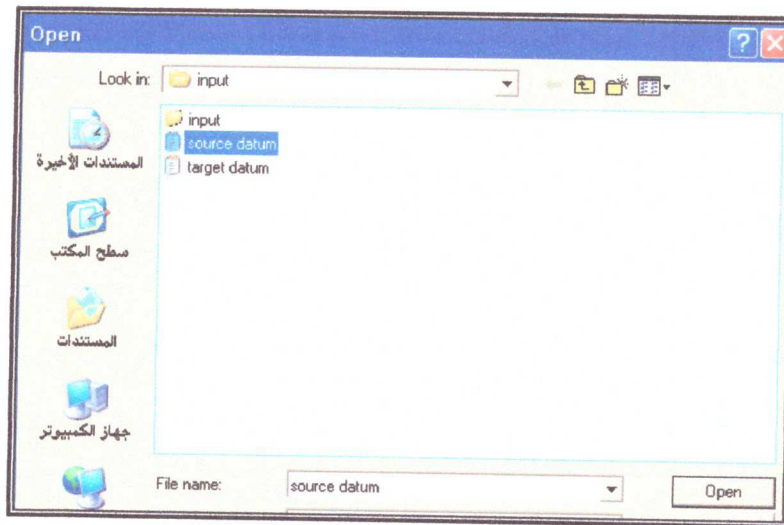
Input File
☐ Single point
☒ Points File

Transformation Method
 7 parameter similarity

Transformation Parameters
 Rotation_x: 2.284805556 Rotation_y: -0.55077777 Rotation_z: 224.5363611
 Translation_x: 10233.858 Translation_y: 6549.981 Translation_z: 720.897
 Scale: 0.94996
 Select Parameters file Calculate Parameters

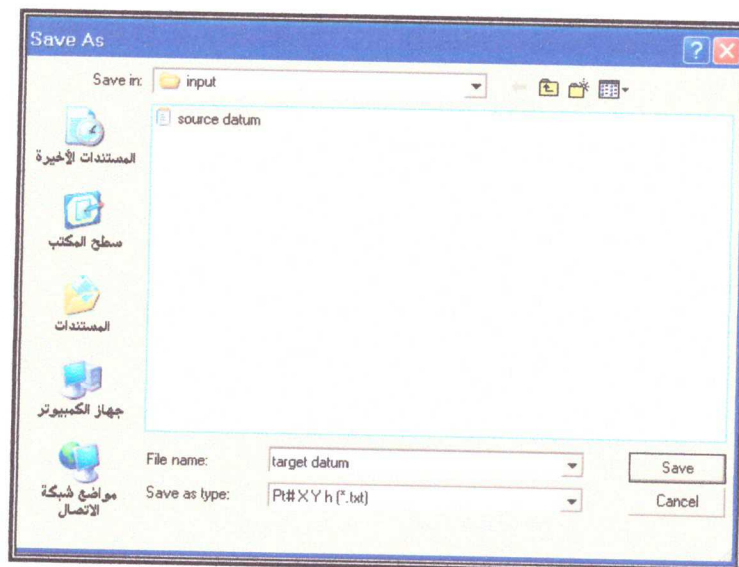
fig(4.73): 7 parameter similarity dialog

3. Open dialog will appear, from input folder choose source file & then press open.



fig(4.74): open dialog

4. From transformation methods select 7-parameters similarity.
5. Enter the value of the 7 parameters you have & if you don't have it, you can calculate it as we show in section 4.3.
6. In datum transformation dialog press on select output file tape.
7. A new dialog will appear, select target file from input folder.& then press save.

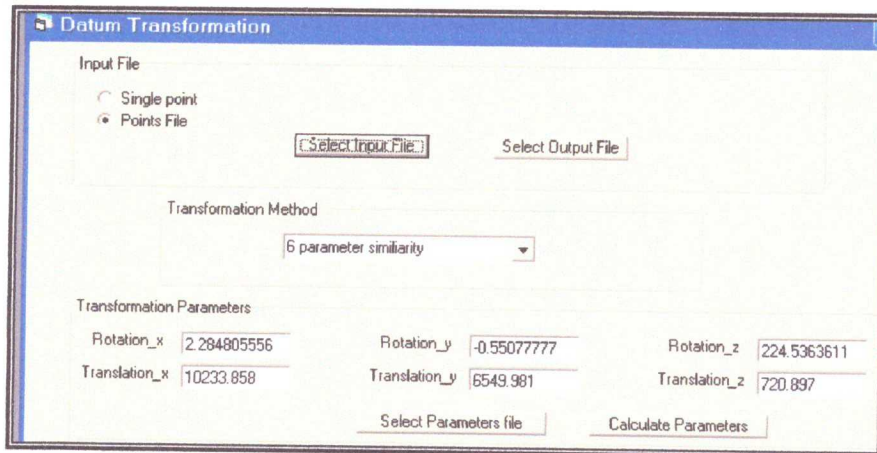


fig(4.75):save as dialog

8. You can see the result from the file you save in.

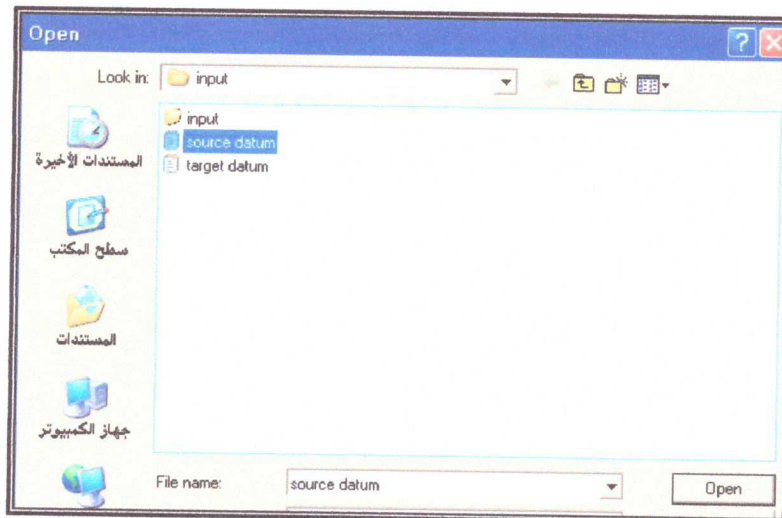
4.3.7 Transform using 6-parameters for points file:

1. Click on points file.
2. Press on select input file



fig(4.76): 6 parameter similarity dialog

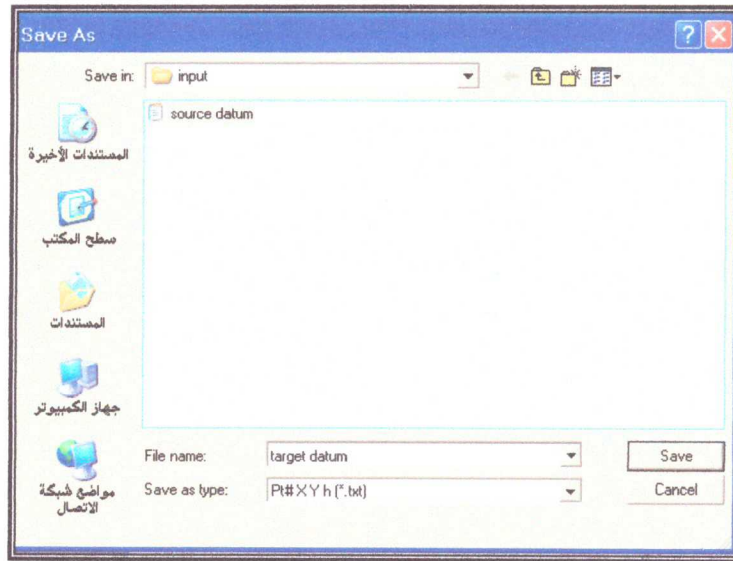
5. Open dialog will appear, from input folder choose source file & then press open.



fig(4.77):open dialog

6. From transformation methods select 6-parameters similarity.
7. Enter the value of the 6 parameters you have & if you don't have it, you can calculate it as we show in section 4.3.
8. In datum transformation dialog press on select output file tape.
9. A new dialog will appear, select target file from input folder. & then presses save.

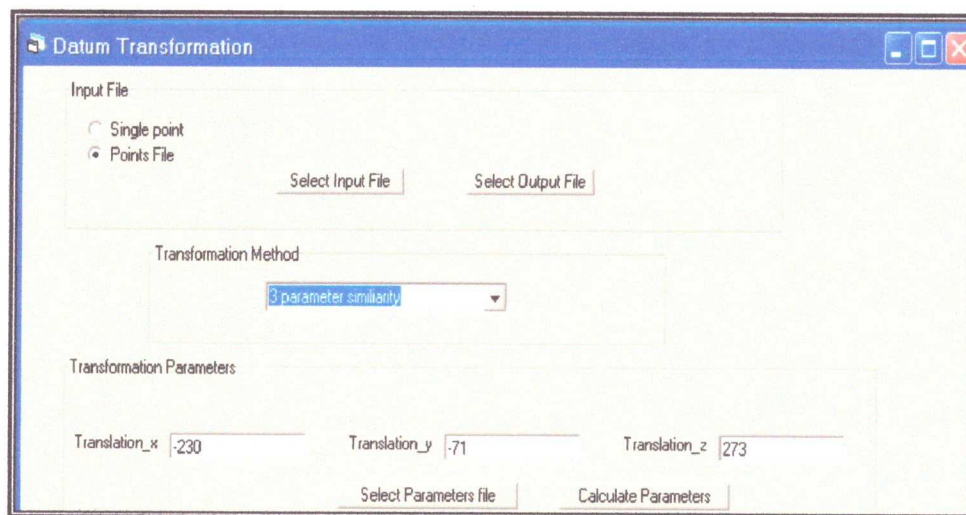
10. You can see the result from the file you save in.



fig(4.78): save as dialog

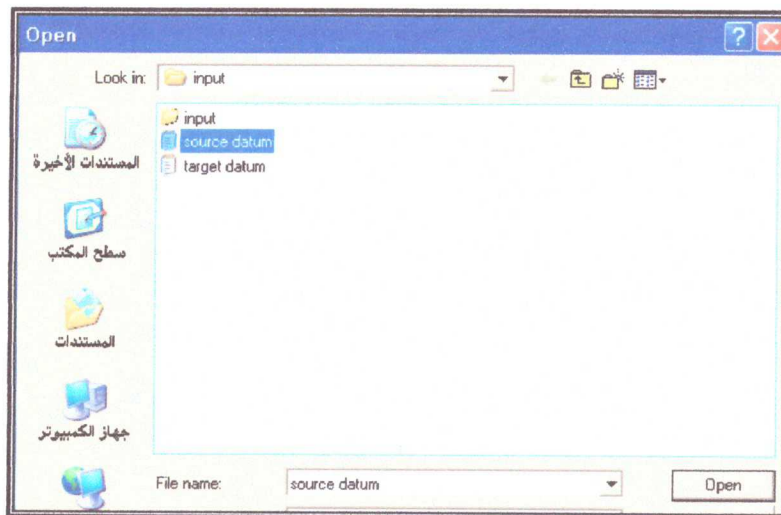
4.3.8 Transform using 3-parameters for points file:

1. Click on points file.
2. Press on select input file



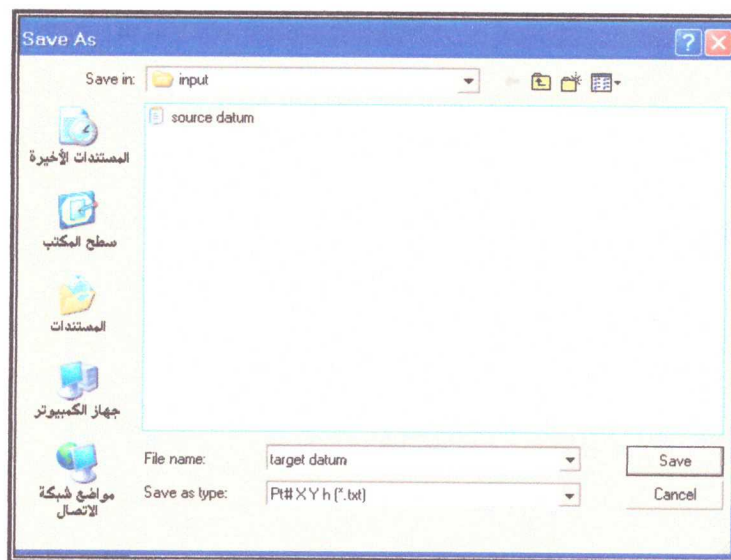
fig(4.79): 3 parameter similarity dialog

3. Open dialog will appear, from input folder choose source file & then press open.



fig(4.80): open dialog

4. From transformation methods select 3-parameters similarity.
5. Enter the value of the 3 parameters you have & if you don't have it, you can calculate it as we show in section 4.3.
6. In datum transformation dialog press on select output file tape.
7. A new dialog will appear, select target file from input folder. & then presses save.

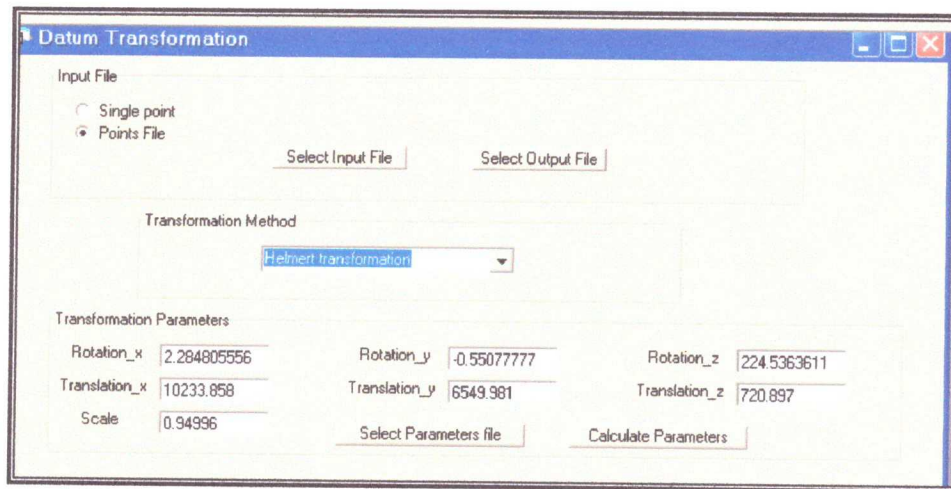


fig(4.81): save as dialog

8. You can see the result from the file you save in.

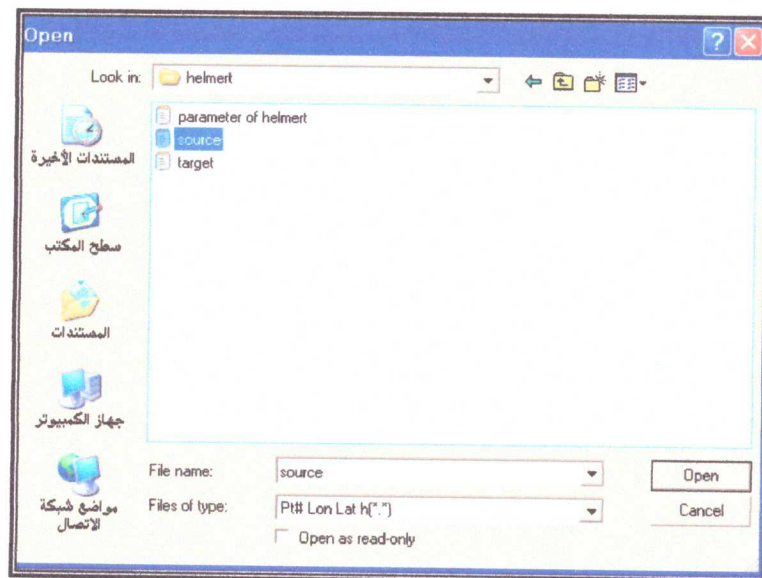
4.3.9 Transform using Helmert method for points file:

1. Choose points file.
2. Press on select input file.



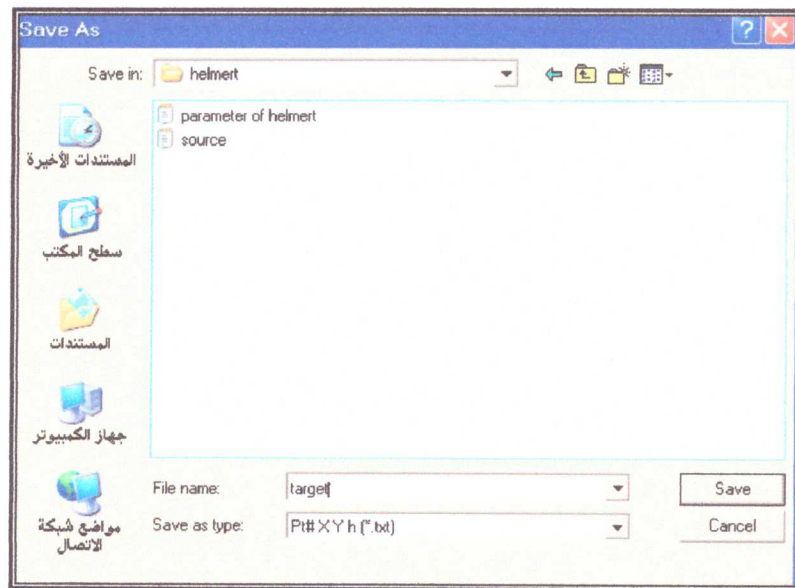
fig(4.82): helmert transformation dialog

3. Open dialog will appear to you, select helmert file from datum transformation folder, press open.
4. In the main dialog, select helmert method from transformation method list.
5. Enter the value of transformation parameters as shown in the main dialog.
6. Press select output file from datum transformation dialog.



fig(4.83): open dialog

7. From save as dialog, choose helmert file & then save.

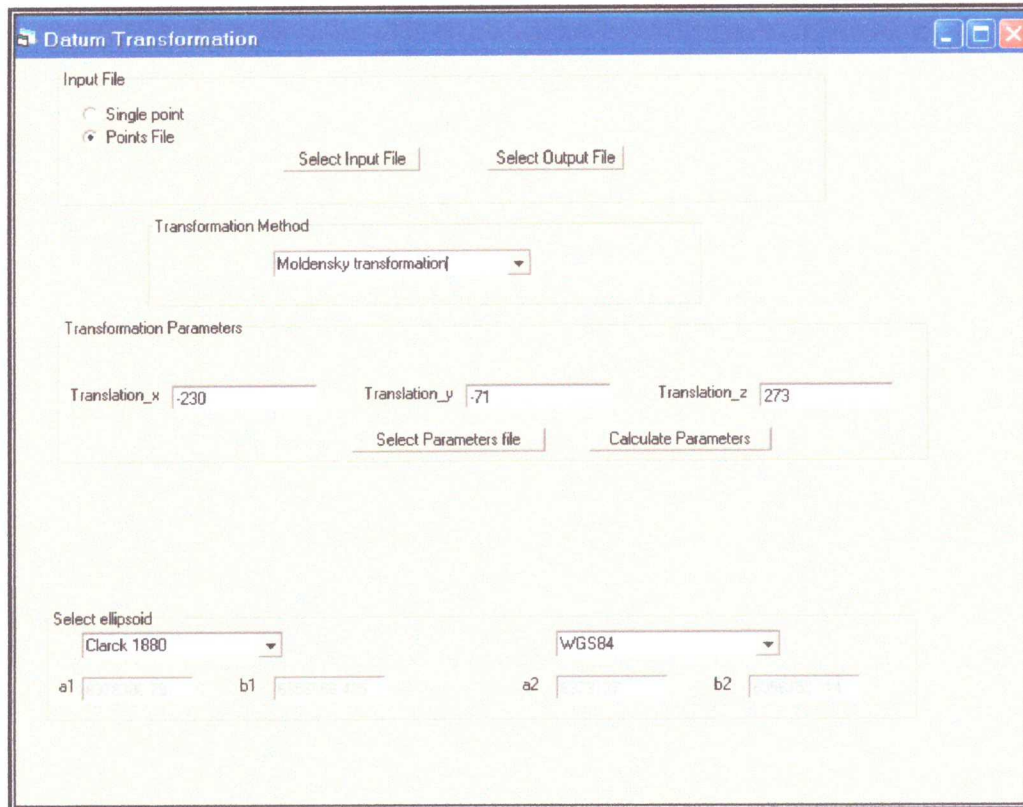


fig(4.84): save as dialog

8. You can see the result from the file you save in.

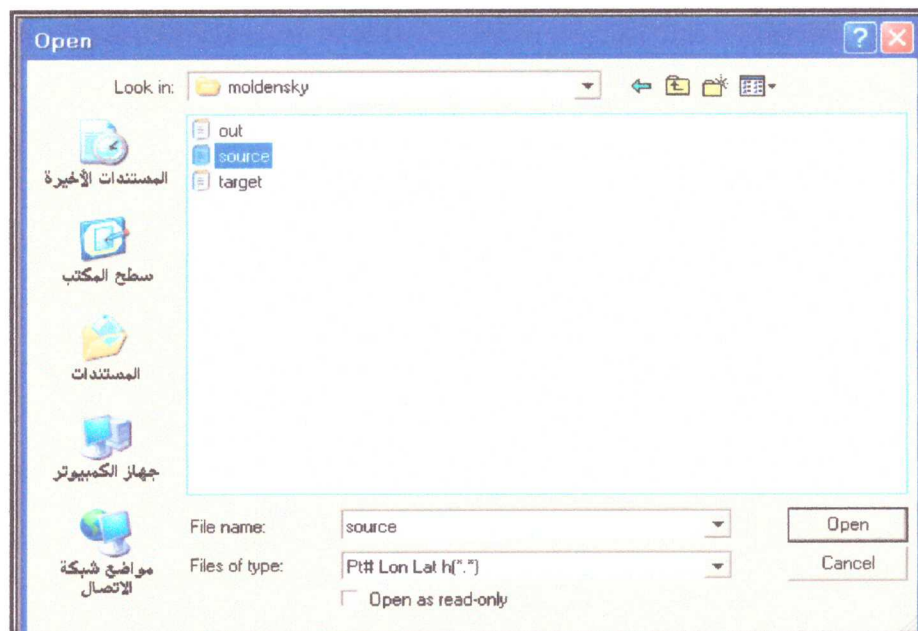
4.3.10 Transform using moldensky method for points file:

1. Choose points file.
2. Press in select input file.



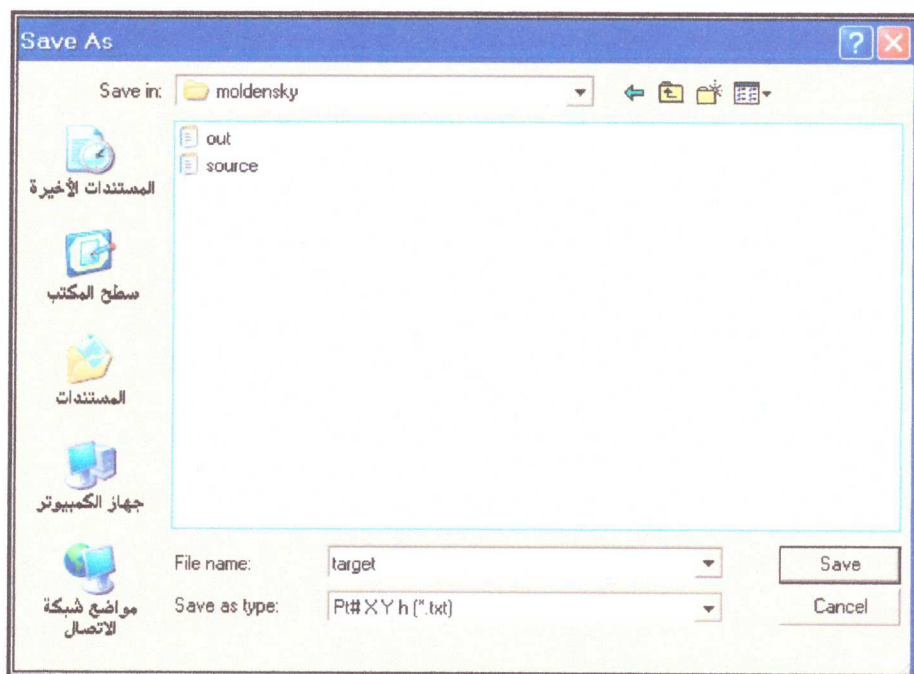
fig(4.85):Moldensky transformation dialog

3. In open dialog, press on moldensky file from datum transformation folder.



fig(4.86): open dialog

4. Now you will return to the main dialog, select moldensky method from transformation method list.
5. Then enter the value of transformation parameters (Tx, Ty, Tz,) as shown in the fig.
6. Now select the first & the second ellipsoid you need & there parameters (a&b) will enter directly.
7. Press on select output file from datum transformation dialog.
8. From save as dialog select moldensky file from datum transformation folder & then press save.
9. You can see the result from the file you save in.



fig(4.87):save as dialog

CHAPTER

5 ***CONCLUSION & RECOMMENDATION***

5.1 Conclusions

5.2 Recommendations

CHAPTER FIVE

CONCLUSION & RECOMMENDATION

5.1 Conclusions

During our work & programming we could conclude the following:

1. Geo-Transform has worked well.
2. Programming all coordinates systems used in Palestine.
3. Leaving the choice for a user to define his own coordinates system if not previously defined in the software.
4. Most common ellipsoids and coordinates are directly to be added in the software.
5. Different datum transformation solutions are to be used to make it more flexible use in other software like Trimble Geomatics office, Leica Geomatics, Arc GIS....etc.

5.2 Recommendations

1. In datum transformations, similarity 7-parameters and moldensky give the best results while the first one has more parameters, and moldensky takes in consideration the changes in the ellipsoids.
2. The Geo-Transform can be extended in further work to read files ^{from} different software like ArcGIS, TGO.....etc.
3. In small areas or countries similarity 3-parameters or 6-parameters can be used.
4. In large areas similarity 7-parameters and moldensky have to be applied.

References :

- 1) Bugayevskiy, Lev M. and John P. Snyder. 1995. Map Projections: A Reference Manual. London: Taylor and Francis.
- 2) Ghadi Zakarneh, Geodesy, Lecture Note, PPU, Hebron, Palestine, 2007/2008.
- 3) Gregory T. French , Understanding the GPS , First edition , Geo research, United states ,1996
- 4) Gunter Seeber, Satellite Geodesy , second edition , de Gruyter –Berlin .
- 5) Huann Fan , Theoretical Geodesy , Stockholm , August 2004.
- 6) Maarten Hooijberg , Practical Geodesy , springer- Verlag Berlin , Heidelberg, 1997.
- 7) Maher owiewi, cartography, Lecture Note ,PPU, Hebron, Palestine, 2006/2007
- 8) Wolfgang Torge , Geodesy , Second edition , de Gruyter , Berlin , New York ,1996.
- 9) www.earth_info.nga.mil/GandG/wgs84/gravitymod/wgs84_180/intptW.html
- 10) www.Geozille.de

APPENDIX

APPENDIX – (A)

APPENDIX – (B)

APPENDIX – (C)

APPENDIX – (A)
Coordinate Conversion

Coordinate Conversion

Option Explicit

```
Dim Xgc, Ygc, Zgc As Double      ' geocentric coordinate
Dim A, b, f, c, e2, ee2 As Double ' semimajor, semiminor, flattening, polar radius of curvature,
first eccentricity squared, second eccentricity squared
Dim M, N As Double ' radius of curvature in the meridian, radius of curvature in the prime vertical
Dim w, v As Double      ' auxiliary quantities
Dim lonrad, latrad, pie As Double      ' geographic coord in radian
Dim lon, lat, h, lon0 As Double      ' geographic coord
Dim flon(0 To 1000), flat(0 To 1000), fh(0 To 1000) As Double
Dim fXgc(0 To 1000), fYgc(0 To 1000), fZgc(0 To 1000) As Double
Dim feasting(0 To 1000), fnorthing(0 To 1000) As Double
Dim ptnr(0 To 1000) As Integer
Dim cnt As Integer
Dim ZONE As Integer
Private Declare Sub Sleep Lib "kernel32" Alias "sleep" (ByVal dwMilliseconds As Long)
```

```
Private Sub Combo1_Click()      ' Ellipsoid
If Combo1.ListIndex = 0 Then    ' Clarck 1880
    Text20.Enabled = False
    Text21.Enabled = False
    Text20 = 6378300.79          ' a:semimajor
    Text21 = 6356566.435        ' b:semiminor
End If
```

```
If Combo1.ListIndex = 1 Then    ' Bessel
    Text20.Enabled = False
    Text21.Enabled = False
    Text20 = 6377483.865         ' a:semimajor
    Text21 = 6356165.383        ' b:semiminor
End If
```

```
If Combo1.ListIndex = 2 Then    ' GRS80
    Text20.Enabled = False
    Text21.Enabled = False
    Text20 = 6378137             ' a:semimajor
    Text21 = 6356752.314        ' b:semiminor
End If
```

```
If Combo1.ListIndex = 3 Then ' GRS67
    Text20.Enabled = False
    Text21.Enabled = False
    Text20 = 6378160 ' a:semimajor
    Text21 = 6356774.516 ' b:semiminor
End If
```

```
If Combo1.ListIndex = 4 Then ' WGS84
    Text20.Enabled = False
    Text21.Enabled = False
    Text20 = 6378137 ' a:semimajor
    Text21 = 6356752.314 ' b:semiminor
End If
```

```
If Combo1.ListIndex = 5 Then ' user define
    Text20.Enabled = True
    Text21.Enabled = True
    Text20 = 6378300.79 ' a:semimajor
    Text21 = 6356566.43000004 ' b:semiminor
End If
End Sub
```

```
Private Sub Combo10_Click() ' zone (1- 60)
Dim ZONE, lon0 As Double
```

```
If Combo10.ListIndex = 0 Then
    ZONE = 1
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 1 Then
    ZONE = 2
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 2 Then
    ZONE = 3
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 3 Then
    ZONE = 4
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 4 Then
    ZONE = 5
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 5 Then
    ZONE = 6
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 6 Then
    ZONE = 7
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 7 Then
    ZONE = 8
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```


If Combo10.ListIndex = 8 Then

 ZONE = 9

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo20.ListIndex = 1

 Text27 = lon0

End If

If Combo10.ListIndex = 9 Then

 ZONE = 10

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo20.ListIndex = 1

 Text27 = lon0

End If

If Combo10.ListIndex = 10 Then

 ZONE = 11

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo20.ListIndex = 1

 Text27 = lon0

End If

If Combo10.ListIndex = 11 Then

 ZONE = 12

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo20.ListIndex = 1

 Text27 = lon0

End If

If Combo10.ListIndex = 12 Then

 ZONE = 13

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo20.ListIndex = 1

 Text27 = lon0

End If

If Combo10.ListIndex = 13 Then

 ZONE = 14

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo20.ListIndex = 1

 Text27 = lon0

End If

```
If Combo10.ListIndex = 14 Then
    ZONE = 15
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 15 Then
    ZONE = 16
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 16 Then
    ZONE = 17
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 17 Then
    ZONE = 18
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 18 Then
    ZONE = 19
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 19 Then
    ZONE = 20
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 20 Then
    ZONE = 21
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 21 Then
    ZONE = 22
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 22 Then
    ZONE = 23
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 23 Then
    ZONE = 24
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 24 Then
    ZONE = 25
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 25 Then
    ZONE = 26
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 26 Then
    ZONE = 27
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 27 Then
    ZONE = 28
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 28 Then
    ZONE = 29
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 29 Then
    ZONE = 30
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo20.ListIndex = 1
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 30 Then
    ZONE = 31
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 31 Then
    ZONE = 32
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```



```
If Combo10.ListIndex = 32 Then
    ZONE = 33
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 33 Then
    ZONE = 34
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 34 Then
    ZONE = 35
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 35 Then
    ZONE = 36
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 36 Then
    ZONE = 37
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 37 Then
    ZONE = 38
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 38 Then
    ZONE = 39
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 39 Then
    ZONE = 40
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 41 Then
    ZONE = 42
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 42 Then
    ZONE = 43
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 43 Then
    ZONE = 44
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 44 Then
    ZONE = 45
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 45 Then
    ZONE = 46
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 46 Then
    ZONE = 47
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 47 Then
    ZONE = 48
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 48 Then
    ZONE = 49
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 49 Then
    ZONE = 50
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 50 Then
    ZONE = 51
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 51 Then
    ZONE = 52
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 52 Then
    ZONE = 53
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 53 Then
    ZONE = 54
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 54 Then
    ZONE = 55
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 55 Then
    ZONE = 56
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 56 Then
    ZONE = 57
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 57 Then
    ZONE = 58
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 58 Then
    ZONE = 59
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
If Combo10.ListIndex = 59 Then
    ZONE = 60
    lon0 = (ZONE * 6 - 3) - 180
    Combo20.ListIndex = 0
    Text27 = lon0
End If
```

```
End Sub
```

```
Private Sub Combo2_Click()    ' projection
If Combo2.ListIndex = 0 Then 'palestine_1923_Palestine_Grid
    Text24.Enabled = False
    Text25.Enabled = False
    Text26.Enabled = False
    Text27.Enabled = False
    Text39.Enabled = False

    Combo1.Enabled = False
    Combo8.Enabled = False
    Combo19.Enabled = False
    Combo20.Enabled = False
    Text24 = 170251.555        ' false_Easting
    Text25 = 126867.909       ' false_Northing
    Text26 = 31.73409694444    ' latitude_of_origin
    Text27 = 35.21206055556    ' Central_Meridian
    Text39 = 1                 ' SCALE FACTOR
```


APPENDIX – (A)

```
Combo1.ListIndex = 0      ' Clarck1880
Combo8.ListIndex = 0      ' cassini
Combo19.ListIndex = 0     ' north
Combo20.ListIndex = 0     ' east
Label17.Visible = False   ' zone
Combo10.Visible = False   ' zone
End If                    ' palestine_1923_Palestine_Grid

If Combo2.ListIndex = 1 Then ' palestine_1923_Israel_CS_Grid
    Text24.Enabled = False
    Text25.Enabled = False
    Text26.Enabled = False
    Text27.Enabled = False
    Text39.Enabled = False

    Combo1.Enabled = False
    Combo8.Enabled = False
    Combo19.Enabled = False
    Combo20.Enabled = False

    Text24 = 170251.555      ' false_Easting
    Text25 = 1126867.909    ' false_Northing
    Text26 = 31.73409694444  ' latitude_of_origin
    Text27 = 35.21206055556  ' Central_Meridian
    Text39 = 1              ' SCALE FACTOR
    Combo1.ListIndex = 0     ' Clarck1880
    Combo8.ListIndex = 0     ' cassini
    Combo19.ListIndex = 0    ' north
    Combo20.ListIndex = 0    ' east
    Label17.Visible = False  ' zone
    Combo10.Visible = False  ' zone

End If                    ' palestine_1923_Israel_CS_Grid
If Combo2.ListIndex = 2 Then ' UTM
    Text24.Enabled = False
    Text25.Enabled = False
    Text26.Enabled = False
    Text27.Enabled = False
    Text39.Enabled = False
```

```
Text27.Enabled = False
Combo1.Enabled = True
Combo8.Enabled = False
Combo10.Enabled = True
Combo19.Enabled = True
Combo20.Enabled = True
```

```
Text24 = 500000      ' false_Easting
Text25 = 0           ' false_Northing
Text26 = 0           ' latitude_of_origin
Text27 = 33          ' Central_Meridian
Text39 = 0.9996      ' SCALE FACTOR
```

```
Combo1.ListIndex = 4  ' WGS84
Combo8.ListIndex = 1  ' TM
Combo19.ListIndex = 0 ' north
Combo20.ListIndex = 0 ' east
```

```
Label17.Visible = True  ' zone
Combo10.Visible = True  ' zone
```

```
End If      ' UTM
```

```
If Combo2.ListIndex = 3 Then ' palestine_1923_Belt
```

```
Text24.Enabled = False
Text25.Enabled = False
Text26.Enabled = False
Text27.Enabled = False
Text39.Enabled = False
```

```
Combo1.Enabled = False
Combo8.Enabled = False
Combo19.Enabled = False
Combo20.Enabled = False
```

```
Text24 = 170251.555      ' false_Easting
Text25 = 1126867.909     ' false_Northing
Text26 = 31.73409694444  ' latitude_of_origin
Text27 = 35.21206055556  ' Central_Meridian
Text39 = 1               ' SCALE FACTOR
```

APPENDIX – (A)

```
Combo1.ListIndex = 0      ' Clarck1880
Combo8.ListIndex = 1      ' TM
Combo19.ListIndex = 0     ' north
Combo20.ListIndex = 0     ' east
Label17.Visible = False  ' zone
Combo10.Visible = False  ' zone

End If                    ' palestine_1923_Belt

If Combo2.ListIndex = 4 Then  ' Israel_TM_Grid
    Text24.Enabled = False
    Text25.Enabled = False
    Text26.Enabled = False
    Text27.Enabled = False
    Text39.Enabled = False

    Combo1.Enabled = False
    Combo8.Enabled = False
    Combo19.Enabled = False
    Combo20.Enabled = False

    Text24 = 219529.584      ' false_Easting
    Text25 = 626907.39      ' false_Northing
    Text26 = 31.734394      ' latitude_of_origin
    Text27 = 35.204517      ' Central_Meridian
    Text39 = 1.000007       ' SCALE FACTOR
    Combo1.ListIndex = 2    ' GRS_1980
    Combo8.ListIndex = 1    ' TM
    Combo19.ListIndex = 0   ' north
    Combo20.ListIndex = 0   ' east
    Label17.Visible = False ' zone
    Combo10.Visible = False ' zone
End If                    ' Israel_TM_Grid

If Combo2.ListIndex = 5 Then  ' user define
    Text24.Enabled = True
    Text25.Enabled = True
    Text26.Enabled = True
    Text27.Enabled = True
    Text39.Enabled = True
```

```

Combo1.Enabled = True
Combo8.Enabled = True
Combo19.Enabled = True
Combo20.Enabled = True

Text24 = 170251.555      ' false_Easting
Text25 = 126867.909      ' false_Northing
Text26 = 31.734097       ' latitude_of_origin
Text27 = 35.212081       ' Central_Meridian
Text39 = 1               ' SCALE FACTOR
Combo1.ListIndex = 0      ' Clarck1880
Combo8.ListIndex = 1      ' TM

Label17.Visible = False  ' zone
Combo10.Visible = False  ' zone
If Combo15.ListIndex = 1 Then
    Combo19.ListIndex = 1  ' north
End If

If Combo16.ListIndex = 1 Then
    Combo20.ListIndex = 1  'east
End If

End If          ' user define
End Sub

Private Sub Combo3_Click()  ' Ellipsoid
If Combo3.ListIndex = 0 Then  ' Clarck 1880
    Text7.Enabled = False
    Text8.Enabled = False
    Text7 = 6378300.79        ' a:semimajor
    Text8 = 6356566.43000004  ' b:semiminor
End If

If Combo3.ListIndex = 1 Then  ' Bessel
    Text7.Enabled = False
    Text8.Enabled = False
    Text7 = 6377483.865       ' a:semimajor
    Text8 = 6356165.383       ' b:semiminor

```

End If

```
If Combo3.ListIndex = 2 Then ' GRS80
    Text7.Enabled = False
    Text8.Enabled = False
    Text7 = 6378137          ' a:semimajor
    Text8 = 6356752.314      ' b:semiminor
End If
```

```
If Combo3.ListIndex = 3 Then ' GRS67
    Text7.Enabled = False
    Text8.Enabled = False
    Text7 = 6378160          ' a:semimajor
    Text8 = 6356774.516      ' b:semiminor
End If
```

```
If Combo3.ListIndex = 4 Then ' WGS84
    Text7.Enabled = False
    Text8.Enabled = False
    Text7 = 6378137          ' a:semimajor
    Text8 = 6356752.314      ' b:semiminor
End If
```

```
If Combo3.ListIndex = 5 Then ' user define
    Text7.Enabled = True
    Text8.Enabled = True
    Text7 = 6378300.79        ' a:semimajor
    Text8 = 6356566.43000004  ' b:semiminor
End If
```

End Sub

```
Private Sub Combo4_Click()
If Combo4.ListIndex = 0 Then ' Clarck 1880
    Text15.Enabled = False
    Text16.Enabled = False
    Text15 = 6378300.79        ' a:semimajor
    Text16 = 6356566.43000004  ' b:semiminor
End If
```


APPENDIX – (A)

```
If Combo4.ListIndex = 1 Then ' Bessel
    Text15.Enabled = False
    Text16.Enabled = False
    Text15 = 6377483.865 ' a:semimajor
    Text16 = 6356165.383 ' b:semiminor
End If
```

```
If Combo4.ListIndex = 2 Then ' GRS80
    Text15.Enabled = False
    Text16.Enabled = False
    Text15 = 6378137 ' a:semimajor
    Text16 = 6356752.314 ' b:semiminor
End If
```

```
If Combo4.ListIndex = 3 Then ' GRS67
    Text15.Enabled = False
    Text16.Enabled = False
    Text15 = 6378160 ' a:semimajor
    Text16 = 6356774.516 ' b:semiminor
End If
```

```
If Combo4.ListIndex = 4 Then ' WGS84
    Text15.Enabled = False
    Text16.Enabled = False
    Text15 = 6378137 ' a:semimajor
    Text16 = 6356752.314 ' b:semiminor
End If
```

```
If Combo4.ListIndex = 5 Then ' user define
    Text15.Enabled = True
    Text16.Enabled = True
    Text15 = 6378300.79 ' a:semimajor
    Text16 = 6356566.43000004 ' b:semiminor
End If
```

```
End Sub
```

```
Private Sub Combo5_Click()
If Combo5.ListIndex = 0 Then ' Clarck 1880
    Text28.Enabled = False
```

```
Text29.Enabled = False
Text28 = 6378300.79      ' a:semimajor
Text29 = 6356566.435    ' b:semiminor
End If

If Combo5.ListIndex = 1 Then  ' Bessel
Text28.Enabled = False
Text29.Enabled = False
Text28 = 6377483.865      ' a:semimajor
Text29 = 6356165.383      ' b:semiminor
End If

If Combo5.ListIndex = 2 Then  ' GRS80
Text28.Enabled = False
Text29.Enabled = False
Text28 = 6378137          ' a:semimajor
Text29 = 6356752.314      ' b:semiminor
End If

If Combo5.ListIndex = 3 Then  ' GRS67
Text28.Enabled = False
Text29.Enabled = False
Text28 = 6378160          ' a:semimajor
Text29 = 6356774.516      ' b:semiminor
End If

If Combo5.ListIndex = 4 Then  ' WGS84
Text28.Enabled = False
Text29.Enabled = False
Text28 = 6378137          ' a:semimajor
Text29 = 6356752.314      ' b:semiminor
End If

If Combo5.ListIndex = 5 Then  ' user define
Text28.Enabled = True
Text29.Enabled = True
Text28 = 6378300.79        ' a:semimajor
Text29 = 6356566.43000004  ' b:semiminor
End If
End Sub
```

```
Private Sub Combo6_Click()      ' Inverse-projection
If Combo6.ListIndex = 0 Then  ' palestine_1923_Palestine_Grid
    Text30.Enabled = False
    Text31.Enabled = False
    Text32.Enabled = False
    Text33.Enabled = False
    Text38.Enabled = False

    Combo5.Enabled = False
    Combo7.Enabled = False
    Combo21.Enabled = False
    Combo22.Enabled = False

    Text30 = 170251.555        ' false_Easting
    Text31 = 126867.909        ' false_Northing
    Text32 = 31.73409694444    ' latitude_of_origin
    Text33 = 35.21206055556    ' Central_Meridian
    Text38 = 1                 ' scale factor
    Combo5.ListIndex = 0       ' Clarck1880
    Combo7.ListIndex = 0       ' Cassini
    Combo21.ListIndex = 0      ' North
    Combo22.ListIndex = 0      ' East
    Combo9.Visible = False     ' zone

End If

If Combo6.ListIndex = 1 Then  ' palestine_1923_Israel_CS_Grid
    Text30.Enabled = False
    Text31.Enabled = False
    Text32.Enabled = False
    Text33.Enabled = False
    Text38.Enabled = False

    Combo21.Enabled = False
    Combo22.Enabled = False
    Combo5.Enabled = False
    Combo7.Enabled = False

    Text30 = 170251.555        ' false_Easting
    Text31 = 1126867.909      ' false_Northing
```

```
Text32 = 31.73409694444    ' latitude_of_origin
Text33 = 35.21206055556    ' Central_Meridian
Text38 = 1                  ' scale factor
Combo5.ListIndex = 0        ' Clarck1880
Combo7.ListIndex = 0        ' Cassini
Combo21.ListIndex = 0       ' North
Combo22.ListIndex = 0       ' East
Combo9.Visible = False     ' zone
```

End If

```
If Combo6.ListIndex = 2 Then    ' UTM
```

```
Text30.Enabled = False
Text31.Enabled = False
Text32.Enabled = False
Text33.Enabled = False
```

```
Text38.Enabled = False
Combo5.Enabled = True
Combo7.Enabled = False
Combo21.Enabled = True
Combo22.Enabled = True
```

```
Text30 = 500000    ' false_Easting
Text31 = 0          ' false_Northing
Text32 = 0          ' latitude_of_origin
```

```
Text38 = 0.9996    ' scale factor
Combo5.ListIndex = 4    ' WGS84
Combo7.ListIndex = 1    ' TM
Combo21.ListIndex = 0   ' North
Combo22.ListIndex = 0   ' East
Combo9.Visible = True   ' zone
Combo21.ListIndex = 0   ' North
Combo22.ListIndex = 0   ' East
```

End If

If Combo6.ListIndex = 3 Then ' palestine_1923_Belt

Text30.Enabled = False

Text31.Enabled = False

Text32.Enabled = False

Text33.Enabled = False

Text38.Enabled = False

Combo5.Enabled = False

Combo7.Enabled = False

Combo21.Enabled = False

Combo22.Enabled = False

Text30 = 170251.555 ' false_Easting

Text31 = 1126867.909 ' false_Northing

Text32 = 31.73409694444 ' latitude_of_origin

Text33 = 35.21206055556 ' Central_Meridian

Text38 = 1 ' scale factor

Combo5.ListIndex = 0 ' Clarck1880

Combo7.ListIndex = 1 ' TM

Combo21.ListIndex = 0 ' North

Combo22.ListIndex = 0 ' East

Combo9.Visible = False ' zone

End If

If Combo6.ListIndex = 4 Then ' Israel_TM_Grid

Text30.Enabled = False

Text31.Enabled = False

Text32.Enabled = False

Text33.Enabled = False

Text38.Enabled = False

Combo5.Enabled = False

Combo7.Enabled = False

Combo21.Enabled = False

Combo22.Enabled = False

Text30 = 219529.584 ' false_Easting

Text31 = 626907.39 ' false_Northing

Text32 = 31.734394 ' latitude_of_origin

Text33 = 35.204517 ' Central_Meridian


```

Text38 = 1.000007      ' scale factor
Combo5.ListIndex = 2   ' GRS_1980
Combo7.ListIndex = 1   ' TM
Combo21.ListIndex = 0  ' North
Combo22.ListIndex = 0  ' East
Combo9.Visible = False ' zone
End If

If Combo6.ListIndex = 5 Then ' user define
    Text30.Enabled = True
    Text31.Enabled = True
    Text32.Enabled = True
    Text33.Enabled = True
    Text38.Enabled = True
    Combo5.Enabled = True
    Combo7.Enabled = True
    Combo21.Enabled = True
    Combo22.Enabled = True
    Text30 = 170251.555 ' false_Easting
    Text31 = 126867.909 ' false_Northing
    Text32 = 31.734097  ' latitude_of_origin
    Text33 = 35.212081  ' Central_Meridian
    Text38 = 1          ' scale factor
    Combo5.ListIndex = 0 ' Clarck1880
    Combo7.ListIndex = 1 ' TM
    Combo9.Visible = False ' zone
    Combo21.ListIndex = 0 ' North
    Combo22.ListIndex = 0 ' East
End If

End Sub

Private Sub Combo9_Click() ' zone (1-60)
Dim ZONE, lon0 As Double

If Combo9.ListIndex = 0 Then
    ZONE = 1
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo22.ListIndex = 1
    Text33 = lon0
End If

```

```
If Combo9.ListIndex = 1 Then
    ZONE = 2
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo22.ListIndex = 1
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 2 Then
    ZONE = 3
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo22.ListIndex = 1
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 3 Then
    ZONE = 4
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo22.ListIndex = 1
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 4 Then
    ZONE = 5
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo22.ListIndex = 1
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 5 Then
    ZONE = 6
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo22.ListIndex = 1
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 6 Then
    ZONE = 7
    lon0 = (-1) * ((ZONE * 6 - 3) - 180)
    Combo22.ListIndex = 1
    Text33 = lon0
End If
```

If Combo9.ListIndex = 7 Then

 ZONE = 8

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 8 Then

 ZONE = 9

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 9 Then

 ZONE = 10

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 10 Then

 ZONE = 11

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 11 Then

 ZONE = 12

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 12 Then

 ZONE = 13

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 13 Then

 ZONE = 14

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 14 Then

 ZONE = 15

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 15 Then

 ZONE = 16

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 16 Then

 ZONE = 17

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 17 Then

 ZONE = 18

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 18 Then

 ZONE = 19

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 19 Then

 ZONE = 20

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 20 Then

 ZONE = 21

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 21 Then

 ZONE = 22

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 22 Then

 ZONE = 23

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 23 Then

 ZONE = 24

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 24 Then

 ZONE = 25

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 25 Then

 ZONE = 26

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 26 Then

 ZONE = 27

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 27 Then

 ZONE = 28

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 28 Then

 ZONE = 29

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 29 Then

 ZONE = 30

 lon0 = (-1) * ((ZONE * 6 - 3) - 180)

 Combo22.ListIndex = 1

 Text33 = lon0

End If

If Combo9.ListIndex = 30 Then

 ZONE = 31

 lon0 = (ZONE * 6 - 3) - 180

 Combo22.ListIndex = 0

 Text33 = lon0

End If

```
If Combo9.ListIndex = 31 Then
    ZONE = 32
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 32 Then
    ZONE = 33
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 33 Then
    ZONE = 34
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 34 Then
    ZONE = 35
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 35 Then
    ZONE = 36
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 36 Then
    ZONE = 37
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 37 Then
    ZONE = 38
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 38 Then
    ZONE = 39
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 39 Then
    ZONE = 40
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 41 Then
    ZONE = 42
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 42 Then
    ZONE = 43
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 43 Then
    ZONE = 44
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 44 Then
    ZONE = 45
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 45 Then
    ZONE = 46
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 46 Then
    ZONE = 47
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 47 Then
    ZONE = 48
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 48 Then
    ZONE = 49
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 49 Then
    ZONE = 50
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 50 Then
    ZONE = 51
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 51 Then
    ZONE = 52
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 52 Then
    ZONE = 53
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 53 Then
    ZONE = 54
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 54 Then
    ZONE = 55
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```

```
If Combo9.ListIndex = 55 Then
    ZONE = 56
    lon0 = (ZONE * 6 - 3) - 180
    Combo22.ListIndex = 0
    Text33 = lon0
End If
```


If Combo9.ListIndex = 56 Then

 ZONE = 57

 lon0 = (ZONE * 6 - 3) - 180

 Combo22.ListIndex = 0

 Text33 = lon0

End If

If Combo9.ListIndex = 57 Then

 ZONE = 58

 lon0 = (ZONE * 6 - 3) - 180

 Combo22.ListIndex = 0

 Text33 = lon0

End If

If Combo9.ListIndex = 58 Then

 ZONE = 59

 lon0 = (ZONE * 6 - 3) - 180

 Combo22.ListIndex = 0

 Text33 = lon0

End If

If Combo9.ListIndex = 59 Then

 ZONE = 60

 lon0 = (ZONE * 6 - 3) - 180

 Combo22.ListIndex = 0

 Text33 = lon0

End If

End Sub

- **Conversion From longitude , latitude , height To X , Y , Z (single point)**

Private Sub Command1_Click()

lon = Text1

If Combo11.ListIndex = 1 Then ' lon west

lon = (-1) * lon

Else

Combo11.ListIndex = 0

End If

lat = Text2

If Combo12.ListIndex = 1 Then ' lat south

lat = (-1) * lat

Else

Combo12.ListIndex = 0

End If

h = Text3

A = Text7

b = Text8

f = (A - b) / A

c = A ^ 2 / b

e2 = (A ^ 2 - b ^ 2) / A ^ 2

ee2 = (A ^ 2 - b ^ 2) / b ^ 2

pie = 3.141592654

latrad = lat * pie / 180

lonrad = lon * pie / 180

v = (1 + ee2 * (Cos(latrad)) ^ 2) ^ 0.5

w = (1 - e2 * (Sin(latrad)) ^ 2) ^ 0.5

M = c / v ^ 3

N = c / v

Xgc = (N + h) * Cos(latrad) * Cos(lonrad)

Ygc = (N + h) * Cos(latrad) * Sin(lonrad)

Zgc = ((1 - e2) * N + h) * Sin(latrad)

```
Text4 = Xgc
Text5 = Ygc
Text6 = Zgc

End Sub

• Projection-Direct ( points file )
Private Sub Command10_Click()
Dim i As Integer
Dim FE, FN As Double           ' false_easting, false_northing
Dim j, G, t, v1, lat0, lat0rad, lon0rad As Double
Dim easting, northing As Double ' cassini projection coordinate
Dim L, L0 As Double           'distance along the meridian from equator to latitude
Dim k0, C1, a1 As Double

CommonDialog5.Filter = "Pt# X Y (*.txt)|*.txt|"
CommonDialog5.ShowSave

If CommonDialog5.FileName <> "" Then
Open CommonDialog5.FileName For Output As #5

For i = 1 To cnt

    A = Val(Text20)
    b = Val(Text21)
    lat = Val(flat(i))

    FE = Val(Text24)
    FN = Val(Text25)
    k0 = Val(Text39)
    lat0 = Val(Text26)
    lon = Val(flon(i))
    lon0 = Val(Text27)

    If Combo19.ListIndex = 1 Then           'lat0 south
        lat0 = (-1) * lat0
    Else
        Combo19.ListIndex = 0
    End If
```

```

If Combo20.ListIndex = 1 Then          'lon0 west
    lon0 = (-1) * lon0
Else
    Combo11.ListIndex = 0
End If

e2 = (A ^ 2 - b ^ 2) / A ^ 2
ee2 = e2 / (1 - e2)

pie = 3.141592654
latrad = lat * pie / 180

lonrad = lon * pie / 180
lat0rad = lat0 * pie / 180
lon0rad = lon0 * pie / 180

If Combo8.ListIndex = 0 Then          ' Cassini

    j = (lonrad - lon0rad) * Cos(latrad) 'A
    t = (Tan(latrad)) ^ 2

    G = e2 * (Cos(latrad)) ^ 2 / (1 - e2) 'C

    'L=M
    'L0=M0

    L = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * latrad - (3 * e2 / 8 + 3 * (e2)
^ 2 / 32 + 45 * (e2) ^ 3 / 1024) * Sin(2 * latrad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3 / 1024) *
Sin(4 * latrad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * latrad))
    L0 = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * (lat0rad) - (3 * e2 / 8 + 3 *
(e2) ^ 2 / 32 + 45 * (e2) ^ 3 / 1024) * Sin(2 * lat0rad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3 / 1024)
* Sin(4 * lat0rad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * lat0rad))

    v1 = A / (1 - e2 * (Sin(latrad)) ^ 2) ^ 0.5

    easting = FE + v1 * (j - t * j ^ 3 / 6 - (8 - t + 8 * G) * t * j ^ 5 / 120)
    northing = FN + L - L0 + v1 * Tan(latrad) * ((j ^ 2) / 2 + (5 - t + 6 * G) * (j ^ 4) / 24)
    Print #5, ptrn(i), easting, northing
End If ' Cassini

```

```

If Combo8.ListIndex = 1 Then      ' TM

    t = (Tan(latrad)) ^ 2
    C1 = ee2 * (Cos(latrad)) ^ 2 'C
    a1 = (lonrad - lon0rad) * Cos(latrad) 'A

    'L=M , L0=M0
    L = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * latrad - (3 * e2 / 8 + 3 * (e2)
        ^ 2 / 32 + 45 * (e2) ^ 3 / 1024) * Sin(2 * latrad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3
        / 1024) * Sin(4 * latrad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * latrad))
    L0 = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * (lat0rad) - (3 * e2 / 8 + 3 *
        (e2) ^ 2 / 32 + 45 * (e2) ^ 3 / 1024) * Sin(2 * lat0rad) + (15 * (e2) ^ 2 / 256 + 45 *
        (e2) ^ 3 / 1024) * Sin(4 * lat0rad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * lat0rad))

    v1 = A / (1 - e2 * (Sin(latrad)) ^ 2) ^ 0.5
    easting = FE + k0 * v1 * (a1 + (1 - t + C1) * a1 ^ 3 / 6 + (5 - 18 * t + t ^ 2 + 72 * C1 - 58
        * ee2) * a1 ^ 5 / 120)
    northing = FN + k0 * (L - L0 + v1 * Tan(latrad) * (a1 ^ 2 / 2 + (5 - t + 9 * C1 + 4 * C1 ^
        2) * a1 ^ 4 / 24 + (61 - 58 * t + t ^ 2 + 600 * C1 - 330 * ee2) * a1 ^ 6 / 720))
    Print #5, ptr(i), easting, northing

End If      ' TM

Next i
End If
Close #5
End Sub

```

- **Projection Inverse (points file)**

```

Private Sub Command11_Click()
    CommonDialog8.Filter = "Pt# Lon Lat (*.*)|*.*)"
    CommonDialog8.ShowOpen

    If CommonDialog8.FileName <> "" Then
        Open CommonDialog8.FileName For Input As #8

        cnt = 0
        Do Until EOF(8)
            cnt = cnt + 1

```



```
Input #8, ptnr(cnt), feasting(cnt), fnorthing(cnt)

Loop
Close #8
If CommonDialog8.CancelError = False Then
Command12.Enabled = True
End If

End If
End Sub

Private Sub Command12_Click()
Dim i As Integer
Dim p1, v1, D, M1, MU1, FE, FN, e1, f1, L1, L0, T1 As Double
Dim easting, northing, lat0, lat0rad, lon0rad As Double
Dim k0, C1 As Double
Dim ZONE As Double

CommonDialog7.Filter = "Pt# X Y h (*.txt)|*.txt"
CommonDialog7.ShowSave

If CommonDialog7.FileName <> "" Then
Open CommonDialog7.FileName For Output As #7

For i = 1 To cnt
A = Val(Text28)
b = Val(Text29) 'calculate lat , lon
k0 = Val(Text38)
FE = Val(Text30)
FN = Val(Text31)
lat0 = Val(Text32)

If Combo21.ListIndex = 1 Then 'lat0 south
lat0 = (-1) * lat0
Else
Combo21.ListIndex = 0
End If

lon0 = Val(Text33)
```

```

If Combo22.ListIndex = 1 Then      'lon0 west
    lon0 = (-1) * lon0
Else
    Combo22.ListIndex = 0
End If

easting = feasting(i)
northing = fnorthing(i)

pie = 3.141592654
lat0rad = lat0 * pie / 180
lon0rad = lon0 * pie / 180

If Combo7.ListIndex = 0 Then      ' Cassini
    e2 = (A ^ 2 - b ^ 2) / A ^ 2
    L0 = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * (lat0rad) - (3 * e2 / 8 + 3 * (e2) ^
        2 / 32 + 45 * (e2) ^ 3 / 1024) * Sin(2 * lat0rad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3 /
        1024) * Sin(4 * lat0rad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * lat0rad))      'M0      'M0
    L1 = L0 + (northing - FN) ' M
    MU1 = L1 / (A * (1 - e2 / 4 - 3 * e2 ^ 2 / 64 - 5 * e2 ^ 3 / 256))

    e1 = (1 - (1 - e2) ^ 0.5) / (1 + (1 - e2) ^ 0.5)
    f1 = MU1 + ((3 * e1 / 2) - (27 * (e1) ^ 3 / 32)) * Sin(2 * MU1) + ((21 * (e1) ^ 2 / 16) - (55 *
        (e1) ^ 4 / 32)) * Sin(4 * MU1) + ((151 * (e1) ^ 3 / 96)) * Sin(6 * MU1) + ((1097 * (e1) ^
        4 / 512)) * Sin(8 * MU1) 'lat

    T1 = (Tan(f1)) ^ 2
    v1 = A / (1 - e2 * (Sin(f1)) ^ 2) ^ 0.5
    p1 = A * (1 - e2) / (1 - e2 * (Sin(f1)) ^ 2) ^ 1.5
    D = (easting - FE) / v1

    latrad = f1 - (v1 * Tan(f1) / p1) * ((D ^ 2) / 2 - (1 + 3 * T1) * (D ^ 4) / 24)
    lat = latrad * 180 / pie

    lonrad = lon0rad + (D - T1 * D ^ 3 / 3 + (1 + 3 * T1) * T1 * D ^ 5 / 15) / Cos(f1)
    lon = lonrad * 180 / pie

Print #7, ptrn(i), lat, lon

End If ' Cassini

```

If Combo7.ListIndex = 1 Then ' TM

$$e2 = (A^2 - b^2) / A^2$$

$$ee2 = e2 / (1 - e2)$$

$$L0 = A * ((1 - e2 / 4 - 3 * (e2^2) / 64 - 5 * (e2^3) / 256) * lat0rad - (3 * e2 / 8 + 3 * (e2^2) / 32 + 45 * (e2^3) / 1024) * Sin(2 * lat0rad) + (15 * (e2^2) / 256 + 45 * (e2^3) / 1024) * Sin(4 * lat0rad) - (35 * (e2^3) / 3072) * Sin(6 * lat0rad))$$

$$L1 = L0 + (northing - FN) / k0$$

$$MU1 = L1 / (A * (1 - e2 / 4 - 3 * e2^2 / 64 - 5 * e2^3 / 256))$$

$$e1 = (1 - (1 - e2)^{0.5}) / (1 + (1 - e2)^{0.5})$$

$$f1 = MU1 + ((3 * e1 / 2) - (27 * (e1)^3 / 32)) * Sin(2 * MU1) + ((21 * (e1)^2 / 16) - (55 * (e1)^4 / 32)) * Sin(4 * MU1) + ((151 * (e1)^3 / 96)) * Sin(6 * MU1) + ((1097 * (e1)^4 / 512)) * Sin(8 * MU1) 'lat$$

$$T1 = (Tan(f1))^2$$

$$C1 = ee2 * (Cos(lat0rad))^2 'C$$

$$v1 = A / (1 - e2 * (Sin(f1))^2)^{0.5}$$

$$p1 = A * (1 - e2) / (1 - e2 * (Sin(f1))^2)^{1.5}$$

$$D = (easting - FE) / (v1 * k0) 'D$$

$$latrad = f1 - (v1 * Tan(f1) / p1) * (D^2 / 2 - (5 + 3 * T1 + 10 * C1 - 4 * C1^2 - 9 * ee2) * D^4 / 24 + (61 + 90 * T1 + 298 * C1 + 45 * T1^2 - 252 * ee2 - 3 * C1^2) * D^6 / 720)$$

$$lat = latrad * 180 / pie$$

$$lonrad = lon0rad + (D - (1 + 2 * T1 + C1) * D^3 / 6 + (5 - 2 * C1 + 28 * T1 - 3 * C1^2 + 8 * ee2 + 24 * T1^2) * D^5 / 120) / Cos(f1)$$

$$lon = lonrad * 180 / pie$$

Print #7, ptrn(i), lat, lon

End If ' TM

Next i

End If

Close #7

End Sub

```
Private Sub Command13_Click()  
Form3.Show  
End Sub
```

- **Conversion From X , Y , Z To longitude , latitude , height (single point)**

```
Private Sub Command2_Click()  
Dim i As Integer  
  
Xgc = Val(Text9)  
Ygc = Val(Text10)  
Zgc = Val(Text11)  
  
A = Val(Text15)  
b = Val(Text16)  
  
f = (A - b) / A  
c = A ^ 2 / b  
e2 = (A ^ 2 - b ^ 2) / A ^ 2  
ee2 = (A ^ 2 - b ^ 2) / b ^ 2  
  
pie = 3.141592654  
lonrad = Atn(Ygc / Xgc)      ' tan inverse  
lon = lonrad * 180 / pie  
  
If lon < 0 Then                ' lon west  
    Combo13.ListIndex = 1  
    lon = (-1) * lon  
Else  
    Combo13.ListIndex = 0  
End If  
  
Text12 = lon  
  
latrad = Atn((Zgc / (Xgc ^ 2 + Ygc ^ 2) ^ 0.5) * (1 - e2) ^ -1)  
lat = latrad * 180 / pie  
  
For i = 1 To 20
```

$v = (1 + ee2 * (\text{Cos}(\text{latrad}))^2)^{0.5}$

$N = c / v$

$\text{latrad} = \text{Atn}((Zgc / (Xgc^2 + Ygc^2)^{0.5}) * (1 - e2 * N / (N + h))^{-1})$

$\text{lat} = \text{latrad} * 180 / \text{pie}$

$h = (((Xgc^2 + Ygc^2)^{0.5} / \text{Cos}(\text{latrad})) - N$

$v = (1 + ee2 * (\text{Cos}(\text{latrad}))^2)^{0.5}$

$N = c / v$

Next i

If lat < 0 Then 'lat south

Combo14.ListIndex = 1

lat = (-1) * lat

Else

Combo14.ListIndex = 0

End If

Text13 = lat

Text14 = h

End Sub

• **Projection-diret (single point)**

Private Sub Command3_Click()

Dim FE, FN As Double ' false_easting, false_northing

Dim j, G, t, v1, lat0, lat0rad, lon0rad As Double

Dim easting, northing As Double ' cassini projection coordinate

Dim L, L0 As Double 'distance along the meridian from equator to latitude

Dim k0, C1, a1 As Double

A = Val(Text20)

b = Val(Text21)

lat = Val(Text22)

If Combo15.ListIndex = 1 Then ' lat south

lat = (-1) * lat

lat0 = (-1) * lat0

Combo19.ListIndex = 1

Else


```
    Combo15.ListIndex = 0
End If

FE = Val(Text24)
FN = Val(Text25)
lat0 = Val(Text26)

If Combo19.ListIndex = 1 Then      ' lat0 south
    lat0 = (-1) * lat0
Else
    Combo19.ListIndex = 0
End If

k0 = Val(Text39)


$$e2 = (A^2 - b^2) / A^2$$

ee2 = e2 / (1 - e2)

pie = 3.141592654
latrad = lat * pie / 180

If Combo8.ListIndex = 0 Then      ' Cassini
lon = Val(Text23)

If Combo16.ListIndex = 1 Then      ' lon west
    lon = (-1) * lon
    lon0 = (-1) * lon0
    Combo20.ListIndex = 1
Else
    Combo16.ListIndex = 0
End If

lon0 = Val(Text27)

If Combo20.ListIndex = 1 Then      ' lon0 west
    lon0 = (-1) * lon0
Else
    Combo11.ListIndex = 0
End If
```

APPENDIX – (A)

```
lonrad = lon * pie / 180
lat0rad = lat0 * pie / 180
lon0rad = lon0 * pie / 180

j = (lonrad - lon0rad) * Cos(latrad) 'A
t = (Tan(latrad)) ^ 2
G = e2 * (Cos(latrad)) ^ 2 / (1 - e2) 'C

'L=M
'L0=M0

L = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * latrad - (3 * e2 / 8 + 3 * (e2) ^ 2 / 32
+ 45 * (e2) ^ 3 / 1024) * Sin(2 * latrad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3 / 1024) * Sin(4
* latrad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * latrad))
L0 = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * (lat0rad) - (3 * e2 / 8 + 3 * (e2) ^ 2
/ 32 + 45 * (e2) ^ 3 / 1024) * Sin(2 * lat0rad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3 / 1024) *
Sin(4 * lat0rad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * lat0rad))

v1 = A / (1 - e2 * (Sin(latrad)) ^ 2) ^ 0.5

easting = FE + v1 * (j - t * j ^ 3 / 6 - (8 - t + 8 * G) * t * j ^ 5 / 120)
northing = FN + L - L0 + v1 * Tan(latrad) * ((j ^ 2) / 2 + (5 - t + 6 * G) * (j ^ 4) / 24)

Text18 = easting
Text19 = northing

End If ' Cassini

If Combo8.ListIndex = 1 Then ' TM
lon = Val(Text23)

If Combo16.ListIndex = 1 Then 'lon west
lon = (-1) * lon
lon0 = (-1) * lon0
Combo20.ListIndex = 1
Else
Combo16.ListIndex = 0
End If
lon0 = Val(Text27)
```

```

If Combo20.ListIndex = 1 Then          'lon0 west
    lon0 = (-1) * lon0
Else
    Combo20.ListIndex = 0
End If

lonrad = lon * pie / 180
lat0rad = lat0 * pie / 180
lon0rad = lon0 * pie / 180

t = (Tan(latrad)) ^ 2
C1 = ee2 * (Cos(latrad)) ^ 2 'C
a1 = (lonrad - lon0rad) * Cos(latrad) 'A

'L=M , L0=M0
L = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * latrad - (3 * e2 / 8 + 3 * (e2) ^ 2 / 32
    + 45 * (e2) ^ 3 / 1024) * Sin(2 * latrad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3 / 1024) * Sin(4
    * latrad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * latrad))
L0 = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * (lat0rad) - (3 * e2 / 8 + 3 * (e2) ^ 2
    / 32 + 45 * (e2) ^ 3 / 1024) * Sin(2 * lat0rad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3 / 1024) *
    Sin(4 * lat0rad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * lat0rad))

v1 = A / (1 - e2 * (Sin(latrad)) ^ 2) ^ 0.5

easting = FE + k0 * v1 * (a1 + (1 - t + C1) * a1 ^ 3 / 6 + (5 - 18 * t + t ^ 2 + 72 * C1 - 58 * ee2)
    * a1 ^ 5 / 120)
northing = FN + k0 * (L - L0 + v1 * Tan(latrad) * (a1 ^ 2 / 2 + (5 - t + 9 * C1 + 4 * C1 ^ 2) * a1
    ^ 4 / 24 + (61 - 58 * t + t ^ 2 + 600 * C1 - 330 * ee2) * a1 ^ 6 / 720))

Text18 = easting
Text19 = northing

End If 'TM

If Combo2.ListIndex = 2 Then ' UTM
    lon = Val(Text23)

    If Combo16.ListIndex = 1 Then          'lon west
        lon = (-1) * lon
        lon0 = (-1) * lon0
    
```

```

    Combo20.ListIndex = 1
    Else
        Combo16.ListIndex = 0
    End If

    lon0 = Val(Text27)

    If Combo20.ListIndex = 1 Then          'lon0 west
        lon0 = (-1) * lon0
    Else
        Combo20.ListIndex = 0
    End If

    lonrad = lon * pie / 180
    lat0rad = lat0 * pie / 180
    lon0rad = lon0 * pie / 180

    t = (Tan(latrad)) ^ 2
    C1 = ee2 * (Cos(latrad)) ^ 2 'C
    a1 = (lonrad - lon0rad) * Cos(latrad) 'A

    'L=M , L0=M0
    L = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * latrad - (3 * e2 / 8 + 3 * (e2) ^ 2 / 32
        + 45 * (e2) ^ 3 / 1024) * Sin(2 * latrad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3 / 1024) * Sin(4
        * latrad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * latrad))
    L0 = A * ((1 - e2 / 4 - 3 * (e2) ^ 2 / 64 - 5 * (e2) ^ 3 / 256) * (lat0rad) - (3 * e2 / 8 + 3 * (e2) ^ 2
        / 32 + 45 * (e2) ^ 3 / 1024) * Sin(2 * lat0rad) + (15 * (e2) ^ 2 / 256 + 45 * (e2) ^ 3 / 1024) *
        Sin(4 * lat0rad) - (35 * (e2) ^ 3 / 3072) * Sin(6 * lat0rad))

    v1 = A / (1 - e2 * (Sin(latrad)) ^ 2) ^ 0.5

    easting = FE + k0 * v1 * (a1 + (1 - t + C1) * a1 ^ 3 / 6 + (5 - 18 * t + t ^ 2 + 72 * C1 - 58 * ee2)
        * a1 ^ 5 / 120)
    northing = FN + k0 * (L - L0 + v1 * Tan(latrad) * (a1 ^ 2 / 2 + (5 - t + 9 * C1 + 4 * C1 ^ 2) * a1
        ^ 4 / 24 + (61 - 58 * t + t ^ 2 + 600 * C1 - 330 * ee2) * a1 ^ 6 / 720))

    Text18 = easting
    Text19 = northing
End If 'UTM
End Sub

```

- **Projection – Inverse (single point)**

Private Sub Command4_Click()

Dim p1, v1, D, M1, MU1, FE, FN, e1, f1, L1, L0, T1 As Double

Dim easting, northing, lat0, lat0rad, lon0rad As Double

Dim k0, C1 As Double

Dim ZONE As Double

A = Val(Text28)

b = Val(Text29) 'calculate lat , lon

k0 = Val(Text38)

FE = Val(Text30)

FN = Val(Text31)

lat0 = Val(Text32)

If Combo21.ListIndex = 1 Then 'lat0 south

lat0 = (-1) * lat0

Else

Combo21.ListIndex = 0

End If

lon0 = Val(Text33)

If Combo22.ListIndex = 1 Then 'lon0 west

lon0 = (-1) * lon0

Else

Combo22.ListIndex = 0

End If

easting = Val(Text34)

northing = Val(Text35)

pie = 3.141592654

lat0rad = lat0 * pie / 180

lon0rad = lon0 * pie / 180

If Combo7.ListIndex = 0 Then ' Cassini

$e2 = (A^2 - b^2) / A^2$

$$L0 = A * ((1 - e2 / 4 - 3 * (e2)^2 / 64 - 5 * (e2)^3 / 256) * (lat0rad) - (3 * e2 / 8 + 3 * (e2)^2 / 32 + 45 * (e2)^3 / 1024) * \sin(2 * lat0rad) + (15 * (e2)^2 / 256 + 45 * (e2)^3 / 1024) * \sin(4 * lat0rad) - (35 * (e2)^3 / 3072) * \sin(6 * lat0rad))$$
 'M0

$$L1 = L0 + (\text{northing} - FN) * M$$

$$MU1 = L1 / (A * (1 - e2 / 4 - 3 * e2^2 / 64 - 5 * e2^3 / 256))$$

$$e1 = (1 - (1 - e2)^{0.5}) / (1 + (1 - e2)^{0.5})$$

$$f1 = MU1 + ((3 * e1 / 2) - (27 * (e1)^3 / 32)) * \sin(2 * MU1) + ((21 * (e1)^2 / 16) - (55 * (e1)^4 / 32)) * \sin(4 * MU1) + ((151 * (e1)^3 / 96)) * \sin(6 * MU1) + ((1097 * (e1)^4 / 512)) * \sin(8 * MU1) * \text{'lat'}$$

$$T1 = (\tan(f1))^2$$

$$v1 = A / (1 - e2 * (\sin(f1))^2)^{0.5}$$

$$p1 = A * (1 - e2) / (1 - e2 * (\sin(f1))^2)^{1.5}$$

$$D = (\text{easting} - FE) / v1$$

$$\text{latrad} = f1 - (v1 * \tan(f1) / p1) * ((D^2) / 2 - (1 + 3 * T1) * (D^4) / 24)$$

$$\text{lat} = \text{latrad} * 180 / \pi$$

$$\text{Text36} = \text{lat}$$

If lat < 0 Then 'lat south

 Combo17.ListIndex = 1

 lat = (-1) * lat

Else

 Combo17.ListIndex = 0

End If

$$\text{lonrad} = \text{lon0rad} + (D - T1 * D^3 / 3 + (1 + 3 * T1) * T1 * D^5 / 15) / \cos(f1)$$

$$\text{lon} = \text{lonrad} * 180 / \pi$$

If lon < 0 Then 'lon west

 Combo18.ListIndex = 1

 lon = (-1) * lon

Else

 Combo18.ListIndex = 0

End If

 Text37 = lon

End If ' Cassini

If Combo7.ListIndex = 1 Then ' TM

$$e2 = (A^2 - b^2) / A^2$$

$$ee2 = e2 / (1 - e2)$$

$$L0 = A * ((1 - e2 / 4 - 3 * (e2 ^ 2) / 64 - 5 * (e2 ^ 3) / 256) * lat0rad - (3 * e2 / 8 + 3 * (e2 ^ 2) / 32 + 45 * (e2 ^ 3) / 1024) * Sin(2 * lat0rad) + (15 * (e2 ^ 2) / 256 + 45 * (e2 ^ 3) / 1024) * Sin(4 * lat0rad) - (35 * (e2 ^ 3) / 3072) * Sin(6 * lat0rad))$$

$$L1 = L0 + (\text{northing} - FN) / k0$$

$$MU1 = L1 / (A * (1 - e2 / 4 - 3 * e2 ^ 2 / 64 - 5 * e2 ^ 3 / 256))$$

$$e1 = (1 - (1 - e2) ^ 0.5) / (1 + (1 - e2) ^ 0.5)$$

$$fl = MU1 + ((3 * e1 / 2) - (27 * (e1) ^ 3 / 32)) * Sin(2 * MU1) + ((21 * (e1) ^ 2 / 16) - (55 * (e1) ^ 4 / 32)) * Sin(4 * MU1) + ((151 * (e1) ^ 3 / 96)) * Sin(6 * MU1) + ((1097 * (e1) ^ 4 / 512)) * Sin(8 * MU1) 'lat$$

$$T1 = (\text{Tan}(fl)) ^ 2$$

$$C1 = ee2 * (\text{Cos}(lat0rad)) ^ 2 'C$$

$$v1 = A / (1 - e2 * (\text{Sin}(fl)) ^ 2) ^ 0.5$$

$$p1 = A * (1 - e2) / (1 - e2 * (\text{Sin}(fl)) ^ 2) ^ 1.5$$

$$D = (\text{easting} - FE) / (v1 * k0) 'D$$

$$\text{latrad} = fl - (v1 * \text{Tan}(fl) / p1) * (D ^ 2 / 2 - (5 + 3 * T1 + 10 * C1 - 4 * C1 ^ 2 - 9 * ee2) * D ^ 4 / 24 + (61 + 90 * T1 + 298 * C1 + 45 * T1 ^ 2 - 252 * ee2 - 3 * C1 ^ 2) * D ^ 6 / 720)$$

$$\text{lat} = \text{latrad} * 180 / \text{pie}$$

If lat < 0 Then 'lat south

Combo17.ListIndex = 1

lat = (-1) * lat

Else

Combo17.ListIndex = 0

End If

Text36 = lat

$$\text{lonrad} = \text{lon0rad} + (D - (1 + 2 * T1 + C1) * D ^ 3 / 6 + (5 - 2 * C1 + 28 * T1 - 3 * C1 ^ 2 + 8 * ee2 + 24 * T1 ^ 2) * D ^ 5 / 120) / \text{Cos}(fl)$$

$$\text{lon} = \text{lonrad} * 180 / \text{pie}$$

If lon < 0 Then 'lon west

Combo18.ListIndex = 1

lon = (-1) * lon

Else

```

    Combo18.ListIndex = 0
End If

    Text37 = lon
End If      ' TM

If Combo6.ListIndex = 2 Then ' UTM

If Combo9.ListIndex < 30 Then
    lon0 = (-1) * lon0
    Combo22.ListIndex = 1
End If

lon0rad = lon0 * pie / 180
e2 = (A ^ 2 - b ^ 2) / A ^ 2
ee2 = e2 / (1 - e2)

L0 = A * ((1 - e2 / 4 - 3 * (e2 ^ 2) / 64 - 5 * (e2 ^ 3) / 256) * lat0rad - (3 * e2 / 8 + 3 * (e2 ^
2) / 32 + 45 * (e2 ^ 3) / 1024) * Sin(2 * lat0rad) + (15 * (e2 ^ 2) / 256 + 45 * (e2 ^ 3) /
1024) * Sin(4 * lat0rad) - (35 * (e2 ^ 3) / 3072) * Sin(6 * lat0rad))
L1 = L0 + (northing - FN) / k0

MU1 = L1 / (A * (1 - e2 / 4 - 3 * e2 ^ 2 / 64 - 5 * e2 ^ 3 / 256))

e1 = (1 - (1 - e2) ^ 0.5) / (1 + (1 - e2) ^ 0.5)
fl = MU1 + ((3 * e1 / 2) - (27 * (e1) ^ 3 / 32)) * Sin(2 * MU1) + ((21 * (e1) ^ 2 / 16) - (55 *
(e1) ^ 4 / 32)) * Sin(4 * MU1) + ((151 * (e1) ^ 3 / 96)) * Sin(6 * MU1) + ((1097 * (e1) ^
4 / 512)) * Sin(8 * MU1) 'lat

T1 = (Tan(fl)) ^ 2
C1 = ee2 * (Cos(lat0rad)) ^ 2 'C

v1 = A / (1 - e2 * (Sin(fl)) ^ 2) ^ 0.5
p1 = A * (1 - e2) / (1 - e2 * (Sin(fl)) ^ 2) ^ 1.5
D = (easting - FE) / (v1 * k0) 'D

latrad = fl - (v1 * Tan(fl) / p1) * (D ^ 2 / 2 - (5 + 3 * T1 + 10 * C1 - 4 * C1 ^ 2 - 9 * ee2) *
D ^ 4 / 24 + (61 + 90 * T1 + 298 * C1 + 45 * T1 ^ 2 - 252 * ee2 - 3 * C1 ^ 2) * D ^ 6
/ 720)
lat = latrad * 180 / pie

```

```
If lat < 0 Then          'lat south
    Combo17.ListIndex = 1
    lat = (-1) * lat
Else
    Combo17.ListIndex = 0
End If
Text36 = lat

lonrad = lon0rad + (D - (1 + 2 * T1 + C1) * D ^ 3 / 6 + (5 - 2 * C1 + 28 * T1 - 3 * C1 ^ 2 + 8
    * ee2 + 24 * T1 ^ 2) * D ^ 5 / 120) / Cos(fl)
lon = lonrad * 180 / pie

If lon < 0 Then          'lon west
    Combo18.ListIndex = 1
    lon = (-1) * lon
Else
    Combo18.ListIndex = 0
End If
Text37 = lon
End If      ' UTM
End Sub

Private Sub Command5_Click()
CommonDialog1.Filter = "Pt# Lon Lat h (*.*)|*.*)"
CommonDialog1.ShowOpen

If CommonDialog1.FileName <> "" Then
    Open CommonDialog1.FileName For Input As #1
    cnt = 0
    Do Until EOF(1)
        cnt = cnt + 1
        Input #1, ptrn(cnt), flon(cnt), flat(cnt), fh(cnt)

        Loop
    Close #1
    If CommonDialog1.CancelError = False Then
        Command6.Enabled = True
    End If
End If
End Sub
```

- **Conversion From longitude , latitude , height To X , Y , Z (points file)**

```
Private Sub Command6_Click()
```

```
Dim i As Integer
```

```
CommonDialog2.Filter = "Pt# X Y h (*.txt)|*.txt|"
```

```
CommonDialog2.ShowSave
```

```
If CommonDialog2.FileName <> "" Then
```

```
Open CommonDialog2.FileName For Output As #2
```

```
For i = 1 To cnt
```

```
lon = Val(flon(i))
```

```
lat = Val(flat(i))
```

```
h = Val(fh(i))
```

```
A = Val(Text7)
```

```
b = Val(Text8)
```

```
f = (A - b) / A
```

```
c = A ^ 2 / b
```

```
e2 = (A ^ 2 - b ^ 2) / A ^ 2
```

```
ee2 = (A ^ 2 - b ^ 2) / b ^ 2
```

```
pie = 3.141592654
```

```
latrad = lat * pie / 180
```

```
lonrad = lon * pie / 180
```

```
v = (1 + ee2 * (Cos(latrad)) ^ 2) ^ 0.5
```

```
w = (1 - e2 * (Sin(latrad)) ^ 2) ^ 0.5
```

```
M = c / v ^ 3
```

```
N = c / v
```

```
Xgc = (N + h) * Cos(latrad) * Cos(lonrad)
```

```
Ygc = (N + h) * Cos(latrad) * Sin(lonrad)
```

```
Zgc = ((1 - e2) * N + h) * Sin(latrad)
```

```
Print #2, ptrn(i), Xgc, Ygc, Zgc
```

```
Next i
```

```
End If
```

```
Close #2
```

```
End Sub
```


- **Conversion From X , Y , Z To longitude , latitude , height (points file)**

```
Private Sub Command7_Click() ' from flon flat fh to xgc ygc zgc
```

```
Dim j As Integer
```

```
Dim i As Integer
```

```
CommonDialog4.Filter = "Pt# lon lat h (*.txt)|*.txt|"
```

```
CommonDialog4.ShowSave
```

```
If CommonDialog4.FileName <> "" Then
```

```
Open CommonDialog4.FileName For Output As #4
```

```
For i = 1 To cnt
```

```
    Xgc = Val(fXgc(i))
```

```
    Ygc = Val(fYgc(i))
```

```
    Zgc = Val(fZgc(i))
```

```
    A = Val(Text15)
```

```
    b = Val(Text16)
```

```
    f = (A - b) / A
```

```
    c = A ^ 2 / b
```

```
    e2 = (A ^ 2 - b ^ 2) / A ^ 2
```

```
    ee2 = (A ^ 2 - b ^ 2) / b ^ 2
```

```
    pie = 3.141592654
```

```
    lonrad = Atn(Ygc / Xgc) 'tan inverse
```

```
    lon = lonrad * 180 / pie
```

```
    latrad = Atn((Zgc / (Xgc ^ 2 + Ygc ^ 2) ^ 0.5) * (1 - e2) ^ -1)
```

```
    lat = latrad * 180 / pie
```

```
For j = 1 To 20
```

```
    v = (1 + ee2 * (Cos(latrad)) ^ 2) ^ 0.5
```

```
    N = c / v
```

```
    latrad = Atn((Zgc / (Xgc ^ 2 + Ygc ^ 2) ^ 0.5) * (1 - e2 * N / (N + h)) ^ -1)
```

```
    lat = latrad * 180 / pie
```

```
    h = (((Xgc ^ 2 + Ygc ^ 2) ^ 0.5) / Cos(latrad)) - N
```

```
    v = (1 + ee2 * (Cos(latrad)) ^ 2) ^ 0.5
```

```
N = c / v
Next j

Print #4, ptrn(i), lon, lat, h
Next i

End If
Close #4
End Sub

Private Sub Command8_Click()
CommonDialog3.Filter = "Pt# Lon Lat h (*.*)|*.*)"
CommonDialog3.ShowOpen

If CommonDialog3.FileName <> "" Then
    Open CommonDialog3.FileName For Input As #3

    cnt = 0
    Do Until EOF(3)
        cnt = cnt + 1
        Input #3, ptrn(cnt), fXgc(cnt), fYgc(cnt), fZgc(cnt)

        Loop
    Close #3
    If CommonDialog3.CancelError = False Then
        Command7.Enabled = True
    End If
End If

End Sub

Private Sub Command9_Click()
CommonDialog6.Filter = "Pt# Lon Lat (*.*)|*.*)"
CommonDialog6.ShowOpen

If CommonDialog6.FileName <> "" Then
    Open CommonDialog6.FileName For Input As #6

    cnt = 0
    Do Until EOF(6)
```

```
        cnt = cnt + 1
        Input #6, ptrr(cnt), flat(cnt), flon(cnt)

        Loop
        Close #6
        If CommonDialog6.CancelError = False Then
            Command10.Enabled = True
        End If

    End If
End Sub

Private Sub Form_Load()
    Combo10.ListIndex = 35
    Combo9.ListIndex = 35
    Combo11.ListIndex = 0
    Combo12.ListIndex = 0
    Combo15.ListIndex = 0
    Combo16.ListIndex = 0

    optpt.Value = 1
    Command5.Enabled = False
    Command6.Enabled = False
    Frame14.Visible = True
    Command1.Visible = True
    Frame13.Visible = True

    Optpt2.Value = 1
    Command8.Enabled = False
    Command7.Enabled = False
    Frame17.Visible = True
    Command2.Visible = True
    Frame16.Visible = True

    Optpt3.Value = 1
    Command9.Enabled = False
    Command10.Enabled = False
    Frame1.Visible = True
    Command3.Visible = True
    Frame2.Visible = True
```

```
Optpt5.Value = 1
Command11.Enabled = False
Command12.Enabled = False
Frame7.Visible = True
Command4.Visible = True
Frame8.Visible = True
End Sub
```

```
Private Sub optfile_Click()
Command5.Enabled = 1
Command6.Enabled = 0
Frame14.Visible = 0
Command1.Visible = 0
Frame13.Visible = 0
End Sub
```

```
Private Sub Optfile1_Click()
Command8.Enabled = 1
Command7.Enabled = 0
Frame17.Visible = 0
Command2.Visible = 0
Frame16.Visible = 0
End Sub
```

```
Private Sub Optfile4_Click()
Command9.Enabled = 1
Command10.Enabled = 0
Frame1.Visible = 0
Command3.Visible = 0
Frame2.Visible = 0
End Sub
```

```
Private Sub Optfile6_Click()
Command11.Enabled = 1
Command12.Enabled = 0
Frame7.Visible = 0
Command4.Visible = 0
Frame8.Visible = 0
End Sub
```

```
Private Sub optpt_Click()
Command5.Enabled = False
Command6.Enabled = False
Frame14.Visible = True
Command1.Visible = True
Frame13.Visible = True
End Sub
```

```
Private Sub Optpt2_Click()
Command8.Enabled = False
Command7.Enabled = False
Frame17.Visible = True
Command2.Visible = True
Frame16.Visible = True
End Sub
```

```
Private Sub Optpt3_Click()
Command9.Enabled = False
Command10.Enabled = False
Frame1.Visible = True
Command3.Visible = True
Frame2.Visible = True
End Sub
```

```
Private Sub Optpt5_Click()
Command11.Enabled = False
Command12.Enabled = False
Frame7.Visible = True
Command4.Visible = True
Frame8.Visible = True
End Sub
```

```
Private Sub Text23_Change()
lon = Text23
ZONE = Int((lon + 180) / 6) + 1
lon0 = (ZONE * 6 - 3) - 180
Combo10.ListIndex = ZONE - 1
Text27 = lon0
End Sub
```


APPENDIX – (B)
Datum Transformation

Datum Transformation

Option Explicit

Dim m11, m12, m13, m21, m22, m23, m31, m32, m33 As Double

Dim x1, x2, y1, y2, z1, z2 As Double

Dim s, Tx, Ty, Tz As Double

Dim rx, ry, rz As Double ' omega,fay,kaba

Dim rx_rad, ry_rad, rz_rad, pie As Double

Dim fx1(0 To 1000), fy1(0 To 1000), fz1(0 To 1000) As Double

Dim frotation_x(0 To 1000), frotation_y(0 To 1000), frotation_z(0 To 1000) As Double

Dim fTranslation_x(0 To 1000), fTranslation_y(0 To 1000), fTranslation_z(0 To 1000) As Double

Dim fscal(0 To 1000) As Double

Dim ptrn(0 To 1000) As Integer

Dim cnt As Integer

Private Sub Combo1_Click() ' **Ellipsoid**

If Combo1.ListIndex = 0 Then ' Clarck 1880

 Text1.Enabled = False

 Text2.Enabled = False

 Text1 = 6378300.79 ' a:semimajor

 Text2 = 6356566.435 ' b:semiminor

End If

If Combo1.ListIndex = 1 Then ' Bessel

 Text1.Enabled = False

 Text2.Enabled = False

 Text1 = 6377483.865 ' a:semimajor

 Text2 = 6356165.383 ' b:semiminor

End If

If Combo1.ListIndex = 2 Then ' GRS80

 Text1.Enabled = False

 Text2.Enabled = False

 Text1 = 6378137 ' a:semimajor

 Text2 = 6356752.314 ' b:semiminor

End If

```
If Combo1.ListIndex = 3 Then    ' GRS67
    Text1.Enabled = False
    Text2.Enabled = False
    Text1 = 6378160            ' a:semimajor
    Text2 = 6356774.516       ' b:semiminor
End If
```

```
If Combo1.ListIndex = 4 Then    ' WGS84
    Text1.Enabled = False
    Text2.Enabled = False
    Text1 = 6378137            ' a:semimajor
    Text2 = 6356752.314       ' b:semiminor
End If
```

```
If Combo1.ListIndex = 5 Then    ' user define
    Text1.Enabled = True
    Text2.Enabled = True
    Text1 = 6378300.79         ' a:semimajor
    Text2 = 6356566.43000004   ' b:semiminor
End If
End Sub
```

```
Private Sub Combo2_Click()
If Combo2.ListIndex = 0 Then    ' Clarck 1880
    Text3.Enabled = False
    Text4.Enabled = False
    Text3 = 6378300.79         ' a:semimajor
    Text4 = 6356566.435       ' b:semiminor
End If
```

```
If Combo2.ListIndex = 1 Then    ' Bessel
    Text3.Enabled = False
    Text4.Enabled = False
    Text3 = 6377483.865        ' a:semimajor
    Text4 = 6356165.383        ' b:semiminor
End If
```

```
If Combo2.ListIndex = 2 Then    ' GRS80
    Text3.Enabled = False
```

```
Text4.Enabled = False
Text3 = 6378137          ' a:semimajor
Text4 = 6356752.314      ' b:semiminor
End If
```

```
If Combo2.ListIndex = 3 Then  ' GRS67
Text3.Enabled = False
Text4.Enabled = False
Text3 = 6378160            ' a:semimajor
Text4 = 6356774.516       ' b:semiminor
End If
```

```
If Combo2.ListIndex = 4 Then  ' WGS84
Text3.Enabled = False
Text4.Enabled = False
Text3 = 6378137            ' a:semimajor
Text4 = 6356752.314       ' b:semiminor
End If
```

```
If Combo2.ListIndex = 5 Then  'user define
Text3.Enabled = True
Text4.Enabled = True
Text3 = 6378300.79         ' a:semimajor
Text4 = 6356566.43000004   ' b:semiminor
End If
End Sub
```

- **7 parameter similiarity (single point)**

```
Private Sub Command1_Click()
```

```
If method.ListIndex = 0 Then  ' 7 parameter
```

```
    x1 = Val(x11)
    y1 = Val(y11)
    z1 = Val(z11)
    rx = Val(rotation_x)
    ry = Val(rotation_y)
    rz = Val(rotation_z)
    s = Val(scal)
```

```
Tx = Val(Translation_x)
Ty = Val(Translation_y)
Tz = Val(Translation_z)
```

```
pie = 3.141592654
rx_rad = rx * pie / 180 'omega
ry_rad = ry * pie / 180 'fay
rz_rad = rz * pie / 180 'kaba
```

```
m11 = Cos(ry_rad) * Cos(rz_rad)
m12 = Sin(rx_rad) * Sin(ry_rad) * Cos(rz_rad) + Cos(rx_rad) * Sin(rz_rad)
m13 = Sin(rx_rad) * Sin(rz_rad) - Cos(rx_rad) * Sin(ry_rad) * Cos(rz_rad)
```

```
m21 = -1 * Cos(ry_rad) * Sin(rz_rad)
m22 = Cos(rx_rad) * Cos(rz_rad) - Sin(rx_rad) * Sin(ry_rad) * Sin(rz_rad)
m23 = Sin(rx_rad) * Cos(rz_rad) + Cos(rx_rad) * Sin(ry_rad) * Sin(rz_rad)
```

```
m31 = Sin(ry_rad)
m32 = -1 * Sin(rx_rad) * Cos(ry_rad)
m33 = Cos(rx_rad) * Cos(ry_rad)
```

```
x2 = s * (m11 * x1 + m21 * y1 + m31 * z1) + Tx
y2 = s * (m12 * x1 + m22 * y1 + m32 * z1) + Ty
z2 = s * (m13 * x1 + m23 * y1 + m33 * z1) + Tz
```

```
x22 = x2
y22 = y2
z22 = z2
```

End If

- **6 parameter similiarity (single point)**

If method.ListIndex = 1 Then ' 6 parameter

```
x1 = Val(x11)
y1 = Val(y11)
z1 = Val(z11)
rx = Val(rotation_x)
ry = Val(rotation_y)
```



```

rz = Val(rotation_z)
Tx = Val(Translation_x)
Ty = Val(Translation_y)
Tz = Val(Translation_z)

pie = 3.141592654
rx_rad = rx * pie / 180 'omega
ry_rad = ry * pie / 180 'fay
rz_rad = rz * pie / 180 'kaba

m11 = Cos(ry_rad) * Cos(rz_rad)
m12 = Sin(rx_rad) * Sin(ry_rad) * Cos(rz_rad) + Cos(rx_rad) * Sin(rz_rad)
m13 = Sin(rx_rad) * Sin(rz_rad) - Cos(rx_rad) * Sin(ry_rad) * Cos(rz_rad)

m21 = -1 * Cos(ry_rad) * Sin(rz_rad)
m22 = Cos(rx_rad) * Cos(rz_rad) - Sin(rx_rad) * Sin(ry_rad) * Sin(rz_rad)
m23 = Sin(rx_rad) * Cos(rz_rad) + Cos(rx_rad) * Sin(ry_rad) * Sin(rz_rad)

m31 = Sin(ry_rad)
m32 = -1 * Sin(rx_rad) * Cos(ry_rad)
m33 = Cos(rx_rad) * Cos(ry_rad)

x2 = m11 * x1 + m21 * y1 + m31 * z1 + Tx
y2 = m12 * x1 + m22 * y1 + m32 * z1 + Ty
z2 = m13 * x1 + m23 * y1 + m33 * z1 + Tz

x22 = x2
y22 = y2
z22 = z2

```

End If

- **3 parameter similiarity (single point)**

If method.ListIndex = 2 Then ' 3 parameter

```

x1 = Val(x11)
y1 = Val(y11)
z1 = Val(z11)
Tx = Val(Translation_x)
Ty = Val(Translation_y)

```

Tz = Val(Translation_z)

x2 = x1 + Tx

y2 = y1 + Ty

z2 = z1 + Tz

x22 = x2

y22 = y2

z22 = z2

End If

- **Helmert Transformation (single point)**

If method.ListIndex = 3 Then ' Helmert

x1 = Val(x11)

y1 = Val(y11)

z1 = Val(z11)

rx = Val(rotation_x)

ry = Val(rotation_y)

rz = Val(rotation_z)

s = Val(scal)

Tx = Val(Translation_x)

Ty = Val(Translation_y)

Tz = Val(Translation_z)

x2 = s * (x1 + rz * y1 - ry * z1) + Tx

y2 = s * (-rz * x1 + y1 + rx * z1) + Ty

z2 = s * (ry * x1 - rx * y1 + z1) + Tz

x22 = x2

y22 = y2

z22 = z2

End If

- **Moldensky transformation (single point)**

If method.ListIndex = 4 Then ' Moldensky

Dim a1, a2, b1, b2, f1, f2 As Double

Dim Rm, Rn, s1 As Double

Dim da, df, dx, dy, dz As Double

Dim dlat, dlon, dh, e2, lon2, lat2, h2, h As Double

a1 = Val(Text1)

b1 = Val(Text2)

a2 = Val(Text3)

b2 = Val(Text4)

rx = Val(x11)

ry = Val(y11)

h = Val(z11)

dx = Val(Translation_x)

dy = Val(Translation_y)

dz = Val(Translation_z)

pie = 3.141592654

rx_rad = rx * pie / 180 'lat

ry_rad = ry * pie / 180 'lon

f1 = (a1 - b1) / a1

e2 = ((a1 ^ 2) - (b1 ^ 2)) / (a1 ^ 2)

f2 = (a2 - b2) / a2

da = a2 - a1

df = f2 - f1

Rm = (a1 * (1 - e2)) / (1 - e2 * (Sin(rx_rad)) ^ 2) ^ 1.5

Rn = (a1) / (1 - e2 * (Sin(rx_rad)) ^ 2) ^ 0.5

dlat = (-1 * dx * Sin(rx_rad) * Cos(ry_rad) - dy * Sin(rx_rad) * Sin(ry_rad) + dz *
Cos(rx_rad) + (da / a1) * (Rn * e2 * Sin(rx_rad) * Cos(rx_rad)) + df * ((a1 * Rm / b1) + b1 * Rn /
a1) * Sin(rx_rad) * Cos(rx_rad)) / ((Rm + h) * Sin(0.000004848136811))

dlon = (-1 * dx * Sin(ry_rad) + dy * Cos(ry_rad)) / ((Rn + h) * Cos(rx_rad) *
Sin(0.000004848136811))

```
dh = dx * Cos(rx_rad) * Cos(ry_rad) + dy * Cos(rx_rad) * Sin(ry_rad) + dz * Sin(rx_rad) - da  
+ (a1 * df + f1 * da) * (Sin(rx_rad)) ^ 2
```

```
x22 = dlat * 180 / pie
```

```
y22 = dlon * 180 / pie
```

```
z22 = dh
```

```
lat2 = ((dlat) / 3600) + rx
```

```
lon2 = ((dlon) / 3600) + ry
```

```
h2 = dh + h
```

```
Text5 = lat2
```

```
Text6 = lon2
```

```
Text7 = h2
```

```
End If
```

```
End Sub
```

```
Private Sub Command11_Click()
```

```
CommonDialog1.Filter = "Pt# Lon Lat h(*.*)|*.*|"
```

```
CommonDialog1.ShowOpen
```

```
If CommonDialog1.FileName <> "" Then
```

```
    Open CommonDialog1.FileName For Input As #1
```

```
    cnt = 0
```

```
    Do Until EOF(1)
```

```
        cnt = cnt + 1
```

```
        Input #1, ptrn(cnt), fx1(cnt), fyl(cnt), fz1(cnt)
```

```
    Loop
```

```
    Close #1
```

```
If CommonDialog1.CancelError = False Then
```

```
    Command12.Enabled = True
```

```
End If
```

```
End If
```

```
End Sub
```

```
Private Sub Command12_Click()
Dim i As Integer

CommonDialog2.Filter = "Pt# X Y h (*.txt)|*.txt"
CommonDialog2.ShowSave

If CommonDialog2.FileName <> "" Then
Open CommonDialog2.FileName For Output As #2

For i = 1 To cnt
```

- **7 Parameter simillarity (Points file)**

```
If method.ListIndex = 0 Then ' 7 parameter
    x1 = Val(fx1(i))
    y1 = Val(fy1(i))
    z1 = Val(fz1(i))

    rx = Val(rotation_x)
    ry = Val(rotation_y)
    rz = Val(rotation_z)

    s = Val(scal)

    Tx = Val(Translation_x)
    Ty = Val(Translation_y)
    Tz = Val(Translation_z)

    pie = 3.141592654
    rx_rad = rx * pie / 180 'omega
    ry_rad = ry * pie / 180 'fay
    rz_rad = rz * pie / 180 'kaba

    m11 = Cos(ry_rad) * Cos(rz_rad)
    m12 = Sin(rx_rad) * Sin(ry_rad) * Cos(rz_rad) + Cos(rx_rad) * Sin(rz_rad)
    m13 = Sin(rx_rad) * Sin(rz_rad) - Cos(rx_rad) * Sin(ry_rad) * Cos(rz_rad)

    m21 = -1 * Cos(ry_rad) * Sin(rz_rad)
    m22 = Cos(rx_rad) * Cos(rz_rad) - Sin(rx_rad) * Sin(ry_rad) * Sin(rz_rad)
    m23 = Sin(rx_rad) * Cos(rz_rad) + Cos(rx_rad) * Sin(ry_rad) * Sin(rz_rad)
```



```

m31 = Sin(ry_rad)
m32 = -1 * Sin(rx_rad) * Cos(ry_rad)
m33 = Cos(rx_rad) * Cos(ry_rad)

```

```

x2 = s * (m11 * x1 + m21 * y1 + m31 * z1) + Tx
y2 = s * (m12 * x1 + m22 * y1 + m32 * z1) + Ty
z2 = s * (m13 * x1 + m23 * y1 + m33 * z1) + Tz

```

```

Print #2, ptrn(i), x2, y2, z2
End If

```

- **6 Parameter simillarity (Points file)**

```

If method.ListIndex = 1 Then ' 6 parameter

```

```

    x1 = Val(fx1(i))
    y1 = Val(fy1(i))
    z1 = Val(fz1(i))

```

```

    rx = Val(rotation_x)
    ry = Val(rotation_y)
    rz = Val(rotation_z)

```

```

    Tx = Val(Translation_x)
    Ty = Val(Translation_y)
    Tz = Val(Translation_z)

```

```

    pie = 3.141592654
    rx_rad = rx * pie / 180 'omega
    ry_rad = ry * pie / 180 'fay
    rz_rad = rz * pie / 180 'kaba

```

```

    m11 = Cos(ry_rad) * Cos(rz_rad)
    m12 = Sin(rx_rad) * Sin(ry_rad) * Cos(rz_rad) + Cos(rx_rad) * Sin(rz_rad)
    m13 = Sin(rx_rad) * Sin(rz_rad) - Cos(rx_rad) * Sin(ry_rad) * Cos(rz_rad)

```

```

    m21 = -1 * Cos(ry_rad) * Sin(rz_rad)
    m22 = Cos(rx_rad) * Cos(rz_rad) - Sin(rx_rad) * Sin(ry_rad) * Sin(rz_rad)
    m23 = Sin(rx_rad) * Cos(rz_rad) + Cos(rx_rad) * Sin(ry_rad) * Sin(rz_rad)

```

```
m31 = Sin(ry_rad)
m32 = -1 * Sin(rx_rad) * Cos(ry_rad)
m33 = Cos(rx_rad) * Cos(ry_rad)
```

```
x2 = (m11 * x1 + m21 * y1 + m31 * z1) + Tx
y2 = (m12 * x1 + m22 * y1 + m32 * z1) + Ty
z2 = (m13 * x1 + m23 * y1 + m33 * z1) + Tz
```

```
Print #2, ptrn(i), x2, y2, z2
```

```
End If
```

- **3 Parameter similliariry (Points file)**

```
If method.ListIndex = 2 Then ' 3 parameter
```

```
    x1 = Val(fx1(i))
```

```
    y1 = Val(fy1(i))
```

```
    z1 = Val(fz1(i))
```

```
    Tx = Val(Translation_x)
```

```
    Ty = Val(Translation_y)
```

```
    Tz = Val(Translation_z)
```

```
    x2 = x1 + Tx
```

```
    y2 = y1 + Ty
```

```
    z2 = z1 + Tz
```

```
Print #2, ptrn(i), x2, y2, z2
```

```
End If
```

- **Helmert Transformation (Points file)**

```
If method.ListIndex = 3 Then ' Helmert
```

```
    x1 = Val(fx1(i))
```

```
    y1 = Val(fy1(i))
```

```
    z1 = Val(fz1(i))
```

```
    rx = Val(rotation_x)
```

```
    ry = Val(rotation_y)
```

```
    rz = Val(rotation_z)
```

```
    s = Val(scal)
```

Tx = Val(Translation_x)

Ty = Val(Translation_y)

Tz = Val(Translation_z)

$x2 = s * (x1 + rz * y1 - ry * z1) + Tx$

$y2 = s * (-rz * x1 + y1 + rx * z1) + Ty$

$z2 = s * (ry * x1 - rx * y1 + z1) + Tz$

Print #2, ptrn(i), x2, y2, z2

End If

- **Moldensky Transformation (Points file)**

If method.ListIndex = 4 Then ' Moldensky

Dim a1, a2, b1, b2, f1, f2 As Double

Dim Rm, Rn, s1 As Double

Dim da, df, dx, dy, dz As Double

Dim dlat, dlon, dh, e2, lon2, lat2, h2, h As Double

a1 = Val(Text1)

b1 = Val(Text2)

a2 = Val(Text3)

b2 = Val(Text4)

rx = Val(fx1(i))

ry = Val(fy1(i))

h = Val(fz1(i))

dx = Val(Translation_x)

dy = Val(Translation_y)

dz = Val(Translation_z)

pie = 3.141592654

rx_rad = rx * pie / 180 'lat

ry_rad = ry * pie / 180 'lon

f1 = (a1 - b1) / a1

$e2 = ((a1^2) - (b1^2)) / (a1^2)$

f2 = (a2 - b2) / a2

```

da = a2 - a1
df = f2 - f1

Rm = (a1 * (1 - e2)) / (1 - e2 * (Sin(rx_rad)) ^ 2) ^ 1.5
Rn = (a1) / (1 - e2 * (Sin(rx_rad)) ^ 2) ^ 0.5

dlat = (-1 * dx * Sin(rx_rad) * Cos(ry_rad) - dy * Sin(rx_rad) * Sin(ry_rad) + dz *
Cos(rx_rad) + (da / a1) * (Rn * e2 * Sin(rx_rad) * Cos(rx_rad)) + df * ((a1 * Rm / b1) + b1 * Rn /
a1) * Sin(rx_rad) * Cos(rx_rad)) / ((Rm + h) * Sin(0.000004848136811))
dlon = (-1 * dx * Sin(ry_rad) + dy * Cos(ry_rad)) / ((Rn + h) * Cos(rx_rad) *
Sin(0.000004848136811))
dh = dx * Cos(rx_rad) * Cos(ry_rad) + dy * Cos(rx_rad) * Sin(ry_rad) + dz * Sin(rx_rad) - da
+ (a1 * df + f1 * da) * (Sin(rx_rad)) ^ 2

x22 = dlat * 180 / pie
y22 = dlon * 180 / pie
z22 = dh

lat2 = ((dlat) / 3600) + rx
lon2 = ((dlon) / 3600) + ry
h2 = dh + h

Print #2, ptrn(i), lat2, lon2, h2

End If
Next i

End If
Close #2
End Sub

Private Sub Command2_Click()
CommonDialog3.Filter = "Pt# Lon Lat h(*.*)|*.*|"
CommonDialog3.ShowOpen

If CommonDialog3.FileName <> "" Then
    Open CommonDialog3.FileName For Input As #3

    cnt = 0
    Do Until EOF(3)

```

```
        cnt = cnt + 1
        Input #3, s, rx, ry, rz, Tx, Ty, Tz
        Loop
        Close #3
        scal = s
        rotation_x = rx
        rotation_y = ry
        rotation_z = rz
        Translation_x = Tx
        Translation_y = Ty
        Translation_z = Tz

    End If

End Sub

Private Sub Command3_Click()
Form4.Show
End Sub

Private Sub Form_Load()
Optpt5.Value = 1
Command11.Enabled = False
Command12.Enabled = False
Frame3.Visible = True
Command1.Visible = True
Frame4.Visible = True
method.ListIndex = 0
Combo1.ListIndex = 0
Combo2.ListIndex = 4

End Sub

Private Sub method_Click()

If method.ListIndex = 0 Then      ' 7-parameter simillarity
    Translation_x.Visible = True
    Translation_y.Visible = True
    Translation_z.Visible = True
```



```
rotation_x.Visible = True  
rotation_y.Visible = True  
rotation_z.Visible = True
```

```
Frame5.Visible = 0  
Combo1.Visible = 0  
Combo2.Visible = 0
```

```
Text1.Visible = 0  
Text2.Visible = 0  
Text3.Visible = 0  
Text4.Visible = 0
```

```
Label16.Visible = 0  
Label17.Visible = 0  
Label18.Visible = 0  
Label19.Visible = 0
```

```
Label1.Visible = True  
Label2.Visible = True  
Label3.Visible = True  
Label4.Visible = True  
Label5.Visible = True  
Label6.Visible = True  
Label7.Visible = True
```

```
Label1.Caption = "Rotation_x"  
Label2.Caption = "Rotation_y"  
Label3.Caption = "Rotation_z"
```

```
Label8.Caption = "x"  
Label9.Caption = "y"  
Label10.Caption = "z"  
scal.Visible = True
```

```
Label11.Caption = "X"  
Label12.Caption = "Y"  
Label13.Caption = "Z"
```

Label20.Visible = 0

Label21.Visible = 0

Label22.Visible = 0

Text5.Visible = 0

Text6.Visible = 0

Text7.Visible = 0

End If

If method.ListIndex = 1 Then ' 6-parameter simillarity

Translation_x.Visible = True

Translation_y.Visible = True

Translation_z.Visible = True

rotation_x.Visible = True

rotation_y.Visible = True

rotation_z.Visible = True

Frame5.Visible = 0

Combo1.Visible = 0

Combo2.Visible = 0

Text1.Visible = 0

Text2.Visible = 0

Text3.Visible = 0

Text4.Visible = 0

Label16.Visible = 0

Label17.Visible = 0

Label18.Visible = 0

Label19.Visible = 0

Label1.Visible = True

Label2.Visible = True

Label3.Visible = True

Label4.Visible = True

Label5.Visible = True

Label6.Visible = True

Label7.Visible = False

```
Label1.Caption = "Rotation_x"  
Label2.Caption = "Rotation_y"  
Label3.Caption = "Rotation_z"
```

```
Label8.Caption = "x"  
Label9.Caption = "y"  
Label10.Caption = "z"
```

```
scal.Visible = False
```

```
Label11.Caption = "X"  
Label12.Caption = "Y"  
Label13.Caption = "Z"
```

```
Label20.Visible = 0  
Label21.Visible = 0  
Label22.Visible = 0
```

```
Text5.Visible = 0  
Text6.Visible = 0  
Text7.Visible = 0
```

```
End If
```

```
If method.ListIndex = 2 Then      ' 3-parameter simillarity
```

```
    Translation_x.Visible = True  
    Translation_y.Visible = True  
    Translation_z.Visible = True
```

```
    rotation_x.Visible = 0  
    rotation_y.Visible = 0  
    rotation_z.Visible = 0
```

```
    Text1.Visible = 0  
    Text2.Visible = 0  
    Text3.Visible = 0  
    Text4.Visible = 0
```

```
    Label16.Visible = 0  
    Label17.Visible = 0
```

Label18.Visible = 0

Label19.Visible = 0

Frame5.Visible = 0

Combo1.Visible = 0

Combo2.Visible = 0

Label1.Visible = 0

Label2.Visible = 0

Label3.Visible = 0

Label4.Visible = True

Label5.Visible = True

Label6.Visible = 1

Label7.Visible = 0

Label1.Caption = "Rotation_x"

Label2.Caption = "Rotation_y"

Label3.Caption = "Rotation_z"

Label8.Caption = "x"

Label9.Caption = "y"

Label10.Caption = "z"

scal.Visible = 0

Label11.Caption = "X"

Label12.Caption = "Y"

Label13.Caption = "Z"

Label20.Visible = 0

Label21.Visible = 0

Label22.Visible = 0

Text5.Visible = 0

Text6.Visible = 0

Text7.Visible = 0

End If

If method.ListIndex = 3 Then ' Helmert

 Translation_x.Visible = True

Translation_y.Visible = True

Translation_z.Visible = True

rotation_x.Visible = 1

rotation_y.Visible = 1

rotation_z.Visible = 1

Text1.Visible = 0

Text2.Visible = 0

Text3.Visible = 0

Text4.Visible = 0

Label16.Visible = 0

Label17.Visible = 0

Label18.Visible = 0

Label19.Visible = 0

Frame5.Visible = 0

Combo1.Visible = 0

Combo2.Visible = 0

Label1.Visible = True

Label2.Visible = True

Label3.Visible = True

Label4.Visible = True

Label5.Visible = True

Label6.Visible = True

Label7.Visible = True

Label1.Caption = "Rotation_x"

Label2.Caption = "Rotation_y"

Label3.Caption = "Rotation_z"

Label8.Caption = "x"

Label9.Caption = "y"

Label10.Caption = "z"

scal.Visible = 1

Label11.Caption = "X"


```
Label12.Caption = "Y"
Label13.Caption = "Z"

Label20.Visible = 0
Label21.Visible = 0
Label22.Visible = 0

Text5.Visible = 0
Text6.Visible = 0
Text7.Visible = 0
End If

If method.ListIndex = 4 Then      ' Moldensky
    Translation_x.Visible = 1
    Translation_y.Visible = 1
    Translation_z.Visible = 1

    Frame5.Visible = 1
    Combo1.Visible = 1
    Combo2.Visible = 1

    Label3.Visible = 1
    Label1.Visible = 0
    Label2.Visible = 0
    Label3.Visible = 0
    Label7.Visible = 0

    Label20.Visible = 1
    Label21.Visible = 1
    Label22.Visible = 1

    Text5.Visible = 1
    Text6.Visible = 1
    Text7.Visible = 1

    Text1.Visible = True
    Text2.Visible = True
    Text3.Visible = True
    Text4.Visible = True
```

Label16.Visible = True

Label17.Visible = True

Label18.Visible = True

Label19.Visible = True

Label4.Caption = "Translation_x"

Label5.Caption = "Translation_y"

Label6.Caption = "Translation_z"

Label8.Caption = "lat"

Label9.Caption = "lon"

Label10.Caption = "h"

Label11.Caption = "Delta latitud"

Label12.Caption = "Delta longitud"

Label13.Caption = "Delta Height"

rotation_x.Visible = 0

rotation_y.Visible = 0

rotation_z.Visible = 0

scal.Visible = 0

End If

End Sub

Private Sub Optfile6_Click()

Command11.Enabled = 1

Command12.Enabled = 0

Frame3.Visible = 0

Command1.Visible = 0

Frame4.Visible = 0

End Sub

Private Sub Optpt5_Click()

Command11.Enabled = False

Command12.Enabled = False

Frame3.Visible = True

Command1.Visible = True

Frame4.Visible = True

End Sub

APPENDIX – (C)
Datum Transformation Parameter

Datum Transformation Parameter

Option Explicit

Dim x1(0 To 1000), y1(0 To 1000), z1(0 To 1000), x2(0 To 1000), y2(0 To 1000), z2(0 To 1000), s,
pie, Tx, Ty, Tz As Double

Dim x0(0 To 1000), y0(0 To 1000), z0(0 To 1000) As Double

Dim dTx, dTy, dTz, ds As Double

Dim dTx1, dTy1, dTz1, ds1 As Double

Dim x22, y22, z22 As Double

Dim drx_dx(0 To 1000), drx_dy(0 To 1000), drx_dz(0 To 1000), dry_dx(0 To 1000), dry_dy(0 To
1000), dry_dz(0 To 1000), dh_dx(0 To 1000), dh_dy(0 To 1000), dh_dz(0 To 1000) As Double

Dim rx, ry, rz As Double ' omega,fay,kaba

Dim rx0, ry0, rz0 As Double

Dim drx, dry, drz As Double

Dim dlat, dlon, dh As Double

Dim rx1, ry1, rz1 As Double

Dim s0, Tx0, Ty0, Tz0 As Double

Dim ptnr(0 To 1000) As Integer

Dim cnt As Integer

Dim rx_rad, ry_rad, rz_rad As Double

Dim rx_rad0, ry_rad0, rz_rad0 As Double

Dim dx2_dS0(0 To 1000), dx2_drx0(0 To 1000), dx2_dry0(0 To 1000), dx2_drz0(0 To 1000),
dx2_dTx0(0 To 1000), dx2_dTy0(0 To 1000), dx2_dTz0(0 To 1000) As Double

Dim dy2_dS0(0 To 1000), dy2_drx0(0 To 1000), dy2_dry0(0 To 1000), dy2_drz0(0 To 1000),
dy2_dTx0(0 To 1000), dy2_dTy0(0 To 1000), dy2_dTz0(0 To 1000) As Double

Dim dz2_dS0(0 To 1000), dz2_drx0(0 To 1000), dz2_dry0(0 To 1000), dz2_drz0(0 To 1000),
dz2_dTx0(0 To 1000), dz2_dTy0(0 To 1000), dz2_dTz0(0 To 1000) As Double

Dim m11, m12, m13, m21, m22, m23, m31, m32, m33 As Double

Dim m110, m120, m130, m210, m220, m230, m310, m320, m330 As Double

Dim fx1(0 To 1000), fy1(0 To 1000), fz1(0 To 1000) As Double

Dim fx2(0 To 1000), fy2(0 To 1000), fz2(0 To 1000) As Double

Dim A(), Xv() As Double

Dim order As Integer

Dim X(0 To 1000), Y(0 To 1000), he(0 To 1000), h(0 To 1000) As Double

Dim L(), t(), TA(), TAT() As Double

Dim M, N, j, k, i As Integer

APPENDIX – (C)

```
Private Sub Combo1_Click()
If Combo1.ListIndex = 0 Then    ' Clarck 1880
    Text1.Enabled = False
    Text2.Enabled = False
    Text1 = 6378300.79          ' a:semimajor
    Text2 = 6356566.435        ' b:semiminor
End If

If Combo1.ListIndex = 1 Then    ' Bessel
    Text1.Enabled = False
    Text2.Enabled = False
    Text1 = 6377483.865        ' a:semimajor
    Text2 = 6356165.383        ' b:semiminor
End If

If Combo1.ListIndex = 2 Then    ' GRS80
    Text1.Enabled = False
    Text2.Enabled = False
    Text1 = 6378137            ' a:semimajor
    Text2 = 6356752.314        ' b:semiminor
End If

If Combo1.ListIndex = 3 Then    ' GRS67
    Text1.Enabled = False
    Text2.Enabled = False
    Text1 = 6378160            ' a:semimajor
    Text2 = 6356774.516        ' b:semiminor
End If

If Combo1.ListIndex = 4 Then    ' WGS84
    Text1.Enabled = False
    Text2.Enabled = False
    Text1 = 6378137            ' a:semimajor
    Text2 = 6356752.314        ' b:semiminor
End If

If Combo1.ListIndex = 5 Then    ' user define
    Text1.Enabled = True
    Text2.Enabled = True
    Text1 = 6378300.79          ' a:semimajor
```


APPENDIX – (C)

```
Text2 = 6356566.43000004 ' b:semiminor
End If
End Sub
```

```
Private Sub Combo2_Click()
If Combo2.ListIndex = 0 Then ' Clarck 1880
Text3.Enabled = False
Text4.Enabled = False
Text3 = 6378300.79 ' a:semimajor
Text4 = 6356566.435 ' b:semiminor
End If
```

```
If Combo2.ListIndex = 1 Then ' Bessel
Text3.Enabled = False
Text4.Enabled = False
Text3 = 6377483.865 ' a:semimajor
Text4 = 6356165.383 ' b:semiminor
End If
```

```
If Combo2.ListIndex = 2 Then ' GRS80
Text3.Enabled = False
Text4.Enabled = False
Text3 = 6378137 ' a:semimajor
Text4 = 6356752.314 ' b:semiminor
End If
```

```
If Combo2.ListIndex = 3 Then ' GRS67
Text3.Enabled = False
Text4.Enabled = False
Text3 = 6378160 ' a:semimajor
Text4 = 6356774.516 ' b:semiminor
End If
```

```
If Combo2.ListIndex = 4 Then ' WGS84
Text3.Enabled = False
Text4.Enabled = False
Text3 = 6378137 ' a:semimajor
Text4 = 6356752.314 ' b:semiminor
End If
```

APPENDIX – (C)

```
If Combo2.ListIndex = 5 Then    'user define
    Text3.Enabled = True
    Text4.Enabled = True
    Text3 = 6378300.79          ' a:semimajor
    Text4 = 6356566.43000004    ' b:semiminor
End If
End Sub

Private Sub Command1_Click()
Dim iterat, i As Integer
Dim az1, az2 As Double
pie = 3.14159265358979

If method.ListIndex = 0 Then    ' 7-parameter
    For i = 1 To cnt
        x1(i) = Val(fx1(i))      ' source
        y1(i) = Val(fy1(i))
        z1(i) = Val(fz1(i))

        x2(i) = Val(fx2(i))      ' target
        y2(i) = Val(fy2(i))
        z2(i) = Val(fz2(i))
    Next i
    M = cnt * 3
    N = 7

    ReDim A(1 To M, 1 To 7)
    ReDim L(1 To M, 1 To 1)
    ReDim t(1 To 7, 1 To M)
    ReDim TA(1 To M, 1 To M)
    ReDim TAT(1 To M, 1 To M)
    ReDim Xv(1 To 7, 1 To 1)

    ' Iteration
    az2 = Atan2(x2(2) - x2(2), y2(2) - y2(1))
    az1 = Atan2(x1(2) - x1(2), y1(2) - y1(1))
    s0 = 1
    rx0 = 0
    ry0 = 0
    rz0 = az2 - az1
```

$$Tx0 = x2(1) - x1(1)$$

$$Ty0 = y2(1) - y1(1)$$

$$Tz0 = z2(1) - z1(1)$$

For iterat = 1 To 20

$$rx_rad0 = rx0 \quad 'omega$$

$$ry_rad0 = ry0 \quad 'fay$$

$$rz_rad0 = rz0 \quad 'kaba$$

$$m110 = \cos(ry_rad0) * \cos(rz_rad0)$$

$$m120 = \sin(rx_rad0) * \sin(ry_rad0) * \cos(rz_rad0) + \cos(rx_rad0) * \sin(rz_rad0)$$

$$m130 = \sin(rx_rad0) * \sin(rz_rad0) - \cos(rx_rad0) * \sin(ry_rad0) * \cos(rz_rad0)$$

$$m210 = -1 * \cos(ry_rad0) * \sin(rz_rad0)$$

$$m220 = \cos(rx_rad0) * \cos(rz_rad0) - \sin(rx_rad0) * \sin(ry_rad0) * \sin(rz_rad0)$$

$$m230 = \sin(rx_rad0) * \cos(rz_rad0) + \cos(rx_rad0) * \sin(ry_rad0) * \sin(rz_rad0)$$

$$m310 = \sin(ry_rad0)$$

$$m320 = -1 * \sin(rx_rad0) * \cos(ry_rad0)$$

$$m330 = \cos(rx_rad0) * \cos(ry_rad0)$$

For i = 1 To cnt

$$x0(i) = s0 * (m110 * x1(i) + m210 * y1(i) + m310 * z1(i)) + Tx0$$

$$y0(i) = s0 * (m120 * x1(i) + m220 * y1(i) + m320 * z1(i)) + Ty0$$

$$z0(i) = s0 * (m130 * x1(i) + m230 * y1(i) + m330 * z1(i)) + Tz0$$

$$dx2_dS0(i) = m110 * x1(i) + m210 * y1(i) + m310 * z1(i)$$

$$dx2_drx0(i) = 0$$

$$dx2_dry0(i) = s0 * ((-1) * \sin(ry_rad0) * \cos(rz_rad0) * x1(i) + \sin(ry_rad0) * \sin(rz_rad0) * y1(i) + \cos(ry_rad0) * z1(i))$$

$$dx2_drz0(i) = s0 * (m210 * x1(i) - m110 * y1(i))$$

$$dx2_dTx0(i) = 1$$

$$dx2_dTy0(i) = 0$$

$$dx2_dTz0(i) = 0$$

$$dy2_dS0(i) = m120 * x1(i) + m220 * y1(i) + m320 * z1(i)$$

$$dy2_drx0(i) = (-1) * s0 * (m130 * x1(i) + m230 * y1(i) + m330 * z1(i))$$

$$dy2_dry0(i) = s0 * (\sin(rx_rad0) * \cos(ry_rad0) * \cos(rz_rad0) * x1(i) - \sin(rx_rad0) * \cos(ry_rad0) * \sin(rz_rad) * y1(i) + \sin(rx_rad) * \sin(ry_rad0) * z1(i))$$

$$dy2_drz0(i) = s0 * (m220 * x1(i) - m120 * y1(i))$$

$$dy2_dTx0(i) = 0$$

$$dy2_dT_y0(i) = 1$$

$$dy2_dT_z0(i) = 0$$

$$dz2_dS0(i) = m130 * x1(i) + m230 * y1(i) + m330 * z1(i)$$

$$dz2_drx0(i) = s0 * (m120 * x1(i) + m220 * y1(i) + m320 * z1(i))$$

$$dz2_dry0(i) = s0 * ((-1) * \cos(rx_rad0) * \cos(ry_rad0) * \cos(rz_rad0) * x1(i) + \cos(rx_rad0) * \cos(ry_rad0) * \sin(rz_rad0) * y1(i) - \cos(rx_rad0) * \sin(ry_rad0) * z1(i))$$

$$dz2_drz0(i) = s0 * (m230 * x1(i) - m130 * y1(i))$$

$$dz2_dTx0(i) = 0$$

$$dz2_dT_y0(i) = 0$$

$$dz2_dT_z0(i) = 1$$

Next i

j = 0

For i = 1 To M ' make A

j = j + 1

$$A(i, 1) = dx2_dS0(j)$$

$$A(i + 1, 1) = dy2_dS0(j)$$

$$A(i + 2, 1) = dz2_dS0(j)$$

$$A(i, 2) = dx2_drx0(j)$$

$$A(i + 1, 2) = dy2_drx0(j)$$

$$A(i + 2, 2) = dz2_drx0(j)$$

$$A(i, 3) = dx2_dry0(j)$$

$$A(i + 1, 3) = dy2_dry0(j)$$

$$A(i + 2, 3) = dz2_dry0(j)$$

$$A(i, 4) = dx2_drz0(j)$$

$$A(i + 1, 4) = dy2_drz0(j)$$

$$A(i + 2, 4) = dz2_drz0(j)$$

$$A(i, 5) = dx2_dTx0(j)$$

$$A(i + 1, 5) = dy2_dTx0(j)$$

$$A(i + 2, 5) = dz2_dTx0(j)$$

$$A(i, 6) = dx2_dT_y0(j)$$

APPENDIX – (C)

$A(i + 1, 6) = dy2_dT_y0(j)$

$A(i + 2, 6) = dz2_dT_y0(j)$

$A(i, 7) = dx2_dT_z0(j)$

$A(i + 1, 7) = dy2_dT_z0(j)$

$A(i + 2, 7) = dz2_dT_z0(j)$

$i = i + 2$

Next i

j = 0

'L matrix

For i = 1 To M

j = j + 1

$L(i, 1) = x2(j) - x0(j)$

$L(i + 1, 1) = y2(j) - y0(j)$

$L(i + 2, 1) = z2(j) - z0(j)$

$i = i + 2$

Next i

Close #1

'A transpose

For i = 1 To 7

For j = 1 To M

$t(i, j) = A(j, i)$

Next j

Next i

'A transpose * A

For i = 1 To 7

For j = 1 To 7

$TA(i, j) = 0\#$

For k = 1 To M

$TA(i, j) = TA(i, j) + t(i, k) * A(k, j)$

Next k

Next j

Next i


```

' At*A inverse
For k = 1 To 7
For j = 1 To 7
    If j <> k Then TA(k, j) = TA(k, j) / TA(k, k)
Next j
TA(k, k) = 1 / TA(k, k)
For i = 1 To 7
If i <> k Then
    For j = 1 To 7
        If j <> k Then TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)
    Next j
    TA(i, k) = -TA(i, k) * TA(k, k)
End If
Next i
Next k

For i = 1 To 7
    For j = 1 To M
        TAT(i, j) = 0
        For k = 1 To 7
            TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)
        Next k
    Next j
Next i

' Inv(A transpose * A) *L
For i = 1 To 7
    Xv(i, 1) = 0
    For k = 1 To M
        Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)
    Next k
Next i

s0 = s0 + Xv(1, 1)
rx0 = rx0 + Xv(2, 1)
ry0 = ry0 + Xv(3, 1)
rz0 = rz0 + Xv(4, 1)
Tx0 = Tx0 + Xv(5, 1)
Ty0 = Ty0 + Xv(6, 1)
Tz0 = Tz0 + Xv(7, 1)

```

Next iterat

```
s = s0
rx = (rx0) * 180 / pie
ry = (ry0) * 180 / pie
rz = (rz0) * 180 / pie
Tx = Tx0
Ty = Ty0
Tz = Tz0
```

```
scal = s
rotation_x = rx
rotation_y = ry
rotation_z = rz
Translation_x = Tx
Translation_y = Ty
Translation_z = Tz
```

End If

If method.ListIndex = 1 Then ' 6 parameter

```
For i = 1 To cnt
    x1(i) = Val(fx1(i)) ' source
    y1(i) = Val(fy1(i))
    z1(i) = Val(fz1(i))

    x2(i) = Val(fx2(i)) 'target
    y2(i) = Val(fy2(i))
    z2(i) = Val(fz2(i))
```

Next i

M = cnt * 3

N = 6

ReDim A(1 To M, 1 To 6)

ReDim L(1 To M, 1 To 1)

ReDim t(1 To 6, 1 To M)

ReDim TA(1 To M, 1 To M)

APPENDIX – (C)

ReDim TAT(1 To M, 1 To M)

ReDim Xv(1 To 6, 1 To 1)

' Iteration

az2 = Atan2(x2(2) - x2(1), y2(2) - y2(1))

az1 = Atan2(x1(2) - x1(1), y1(2) - y1(1))

s0 = 1

rx0 = 0

ry0 = 0

rz0 = az2 - az1

Tx0 = x2(1) - x1(1)

Ty0 = y2(1) - y1(1)

Tz0 = z2(1) - z1(1)

For iterat = 1 To 20

rx_rad0 = rx0 'omega

ry_rad0 = ry0 'fay

rz_rad0 = rz0 'kaba

m110 = Cos(ry_rad0) * Cos(rz_rad0)

m120 = Sin(rx_rad0) * Sin(ry_rad0) * Cos(rz_rad0) + Cos(rx_rad0) * Sin(rz_rad0)

m130 = Sin(rx_rad0) * Sin(rz_rad0) - Cos(rx_rad0) * Sin(ry_rad0) * Cos(rz_rad0)

m210 = -1 * Cos(ry_rad0) * Sin(rz_rad0)

m220 = Cos(rx_rad0) * Cos(rz_rad0) - Sin(rx_rad0) * Sin(ry_rad0) * Sin(rz_rad0)

m230 = Sin(rx_rad0) * Cos(rz_rad0) + Cos(rx_rad0) * Sin(ry_rad0) * Sin(rz_rad0)

m310 = Sin(ry_rad0)

m320 = -1 * Sin(rx_rad0) * Cos(ry_rad0)

m330 = Cos(rx_rad0) * Cos(ry_rad0)

For i = 1 To cnt

x0(i) = s0 * (m110 * x1(i) + m210 * y1(i) + m310 * z1(i)) + Tx0

y0(i) = s0 * (m120 * x1(i) + m220 * y1(i) + m320 * z1(i)) + Ty0

z0(i) = s0 * (m130 * x1(i) + m230 * y1(i) + m330 * z1(i)) + Tz0

'scale is fixed to S=1 , not unknown

```

dx2_drx0(i) = 0
dx2_dry0(i) = s0 * ((-1) * Sin(ry_rad0) * Cos(rz_rad0) * x1(i) + Sin(ry_rad0) * Sin(rz_rad0) *
              y1(i) + Cos(ry_rad0) * z1(i))
dx2_drz0(i) = s0 * (m210 * x1(i) - m110 * y1(i))
dx2_dTx0(i) = 1
dx2_dTy0(i) = 0
dx2_dTz0(i) = 0

dy2_drx0(i) = (-1) * s0 * (m130 * x1(i) + m230 * y1(i) + m330 * z1(i))
dy2_dry0(i) = s0 * (Sin(rx_rad0) * Cos(ry_rad0) * Cos(rz_rad0) * x1(i) - Sin(rx_rad0) *
                  Cos(ry_rad0) * Sin(rz_rad0) * y1(i) + Sin(rx_rad0) * Sin(ry_rad0) * z1(i))
dy2_drz0(i) = s0 * (m220 * x1(i) - m120 * y1(i))
dy2_dTx0(i) = 0
dy2_dTy0(i) = 1
dy2_dTz0(i) = 0

'dz2_dS0(i) = m130 * x1(i) + m230 * y1(i) + m330 * z1(i)
dz2_drx0(i) = s0 * (m120 * x1(i) + m220 * y1(i) + m320 * z1(i))
dz2_dry0(i) = s0 * ((-1) * Cos(rx_rad0) * Cos(ry_rad0) * Cos(rz_rad0) * x1(i) + Cos(rx_rad0) *
                  Cos(ry_rad0) * Sin(rz_rad0) * y1(i) - Cos(rx_rad0) * Sin(ry_rad0) * z1(i))
dz2_drz0(i) = s0 * (m230 * x1(i) - m130 * y1(i))
dz2_dTx0(i) = 0
dz2_dTy0(i) = 0
dz2_dTz0(i) = 1

```

Next i

j = 0

For i = 1 To M ' make A

j = j + 1

A(i, 1) = dx2_drx0(j)

A(i + 1, 1) = dy2_drx0(j)

A(i + 2, 1) = dz2_drx0(j)

A(i, 2) = dx2_dry0(j)

A(i + 1, 2) = dy2_dry0(j)

A(i + 2, 2) = dz2_dry0(j)

A(i, 3) = dx2_drz0(j)

A(i + 1, 3) = dy2_drz0(j)

A(i + 2, 3) = dz2_drz0(j)

APPENDIX – (C)

$$A(i, 4) = dx2_dT_x0(j)$$

$$A(i + 1, 4) = dy2_dT_x0(j)$$

$$A(i + 2, 4) = dz2_dT_x0(j)$$

$$A(i, 5) = dx2_dT_y0(j)$$

$$A(i + 1, 5) = dy2_dT_y0(j)$$

$$A(i + 2, 5) = dz2_dT_y0(j)$$

$$A(i, 6) = dx2_dT_z0(j)$$

$$A(i + 1, 6) = dy2_dT_z0(j)$$

$$A(i + 2, 6) = dz2_dT_z0(j)$$

$$i = i + 2$$

Next i

$$j = 0$$

'L matrix

For i = 1 To M

$$j = j + 1$$

$$L(i, 1) = x2(j) - x0(j)$$

$$L(i + 1, 1) = y2(j) - y0(j)$$

$$L(i + 2, 1) = z2(j) - z0(j)$$

$$i = i + 2$$

Next i

'A transpose

For i = 1 To 6

For j = 1 To M

$$t(i, j) = A(j, i)$$

Next j

Next i

'A transpose * A

For i = 1 To 6

For j = 1 To 6

$$TA(i, j) = 0\#$$

For k = 1 To M


```

    TA(i, j) = TA(i, j) + t(i, k) * A(k, j)
  Next k
  Next j
Next i

' AT*A inverse
For k = 1 To 6
  For j = 1 To 6
    If j <> k Then TA(k, j) = TA(k, j) / TA(k, k)
  Next j
  TA(k, k) = 1 / TA(k, k)
  For i = 1 To 6
    If i <> k Then
      For j = 1 To 6
        If j <> k Then TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)
      Next j
      TA(i, k) = -TA(i, k) * TA(k, k)
    End If
  Next i
Next k

For i = 1 To 6
  For j = 1 To M
    TAT(i, j) = 0
    For k = 1 To 6
      TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)
    Next k
  Next j
Next i

' Inv (A transpose * A) *L
For i = 1 To 6
  Xv(i, 1) = 0
  For k = 1 To M
    Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)
  Next k
  Print #1, Xv(i, 1)
Next i

```

APPENDIX – (C)

$rx0 = rx0 + Xv(1, 1)$

$ry0 = ry0 + Xv(2, 1)$

$rz0 = rz0 + Xv(3, 1)$

$Tx0 = Tx0 + Xv(4, 1)$

$Ty0 = Ty0 + Xv(5, 1)$

$Tz0 = Tz0 + Xv(6, 1)$

Next iterat

$s = s0$

$rx = (rx0) * 180 / \text{pie}$

$ry = (ry0) * 180 / \text{pie}$

$rz = (rz0) * 180 / \text{pie}$

$Tx = Tx0$

$Ty = Ty0$

$Tz = Tz0$

$\text{scal} = s$

$\text{rotation_x} = rx$

$\text{rotation_y} = ry$

$\text{rotation_z} = rz$

$\text{Translation_x} = Tx$

$\text{Translation_y} = Ty$

$\text{Translation_z} = Tz$

End If

If method.ListIndex = 2 Then ' 3 parameter

For i = 1 To cnt

$x1(i) = \text{Val}(fx1(i))$ ' source

$y1(i) = \text{Val}(fy1(i))$

$z1(i) = \text{Val}(fz1(i))$

$x2(i) = \text{Val}(fx2(i))$ 'target

$y2(i) = \text{Val}(fy2(i))$

$z2(i) = \text{Val}(fz2(i))$

Next i

APPENDIX – (C)

M = cnt * 3

N = 3

ReDim A(1 To M, 1 To 3)

ReDim L(1 To M, 1 To 1)

ReDim t(1 To 3, 1 To M)

ReDim TA(1 To M, 1 To M)

ReDim TAT(1 To M, 1 To M)

ReDim Xv(1 To 3, 1 To 1)

j = 0

For i = 1 To M ' make A

j = j + 1

A(i, 1) = 1

A(i + 1, 1) = 0

A(i + 2, 1) = 0

A(i, 2) = 0

A(i + 1, 2) = 1

A(i + 2, 2) = 0

A(i, 3) = 0

A(i + 1, 3) = 0

A(i + 2, 3) = 1

i = i + 2

Next i

j = 0

'L matrix

For i = 1 To M

j = j + 1

L(i, 1) = x2(j)

L(i + 1, 1) = y2(j)

L(i + 2, 1) = z2(j)

i = i + 2

Next i

'A transpose

For i = 1 To 3

For j = 1 To M

t(i, j) = A(j, i)

Next j

Next i

'A transpose * A

For i = 1 To 3

For j = 1 To 3

TA(i, j) = 0#

For k = 1 To M

TA(i, j) = TA(i, j) + t(i, k) * A(k, j)

Next k

Next j

Next i

' AT*A inverse

For k = 1 To 3

For j = 1 To 3

If j <> k Then TA(k, j) = TA(k, j) / TA(k, k)

Next j

TA(k, k) = 1 / TA(k, k)

For i = 1 To 3

If i <> k Then

For j = 1 To 3

If j <> k Then TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)

Next j

TA(i, k) = -TA(i, k) * TA(k, k)

End If

Next i

Next k

For i = 1 To 3

For j = 1 To M

TAT(i, j) = 0

For k = 1 To 3

TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)

Next k

Next j

Next i

```
' inv(A transpose * A) *L
For i = 1 To 3
  Xv(i, 1) = 0
  For k = 1 To M
    Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)
  Next k
Next i
```

```
Tx0 = Xv(1, 1)
Ty0 = Xv(2, 1)
Tz0 = Xv(3, 1)
```

```
' Next iterat
```

```
Tx = Tx0
Ty = Ty0
Tz = Tz0
```

```
scal = 1
rotation_x = 0
rotation_y = 0
rotation_z = 0
Translation_x = Tx
Translation_y = Ty
Translation_z = Tz
```

```
End If
```

```
If method.ListIndex = 3 Then ' Helmert
```

```
For i = 1 To cnt
  x1(i) = Val(fx1(i)) ' source
  y1(i) = Val(fy1(i))
  z1(i) = Val(fz1(i))

  x2(i) = Val(fx2(i)) 'target
  y2(i) = Val(fy2(i))
  z2(i) = Val(fz2(i))
Next i
```


M = cnt * 3

N = 7

ReDim A(1 To M, 1 To 7)

ReDim L(1 To M, 1 To 1)

ReDim t(1 To 7, 1 To M)

ReDim TA(1 To M, 1 To M)

ReDim TAT(1 To M, 1 To M)

ReDim Xv(1 To 7, 1 To 1)

' Iteration

s0 = 1

rx0 = 0

ry0 = 0

rz0 = 0

Tx0 = x2(1) - x1(1)

Ty0 = y2(1) - y1(1)

Tz0 = z2(1) - z1(1)

For iterat = 1 To 1

For i = 1 To cnt

dx2_dS0(i) = x1(i) + rz0 * y1(i) - ry0 * z1(i)

dx2_drx0(i) = 0

dx2_dry0(i) = -1 * s0 * z1(i)

dx2_drz0(i) = s0 * y1(i)

dx2_dTx0(i) = 1

dx2_dTy0(i) = 0

dx2_dTz0(i) = 0

dy2_dS0(i) = -1 * rz0 * x1(i) + y1(i) + rx0 * z1(i)

dy2_drx0(i) = s0 * z1(i)

dy2_dry0(i) = 0

dy2_drz0(i) = -1 * s0 * x1(i)

dy2_dTx0(i) = 0

dy2_dTy0(i) = 1

dy2_dTz0(i) = 0

dz2_dS0(i) = ry0 * x1(i) - rx0 * y1(i) + z1(i)

dz2_drx0(i) = -1 * s0 * y1(i)

dz2_dry0(i) = s0 * x1(i)

$$dz2_drz0(i) = 0$$

$$dz2_dTx0(i) = 0$$

$$dz2_dT_y0(i) = 0$$

$$dz2_dTz0(i) = 1$$

$$x0(i) = s0 * (x1(i) + rz0 * y1(i) - ry0 * z1(i)) + Tx0$$

$$y0(i) = s0 * (-rz0 * x1(i) + y1(i) + rx0 * z1(i)) + Ty0$$

$$z0(i) = s0 * (ry0 * x1(i) - rx0 * y1(i) + z1(i)) + Tz0$$

Next i

j = 0

For i = 1 To M ' make A

j = j + 1

$$A(i, 1) = dx2_dS0(j)$$

$$A(i + 1, 1) = dy2_dS0(j)$$

$$A(i + 2, 1) = dz2_dS0(j)$$

$$A(i, 2) = dx2_drx0(j)$$

$$A(i + 1, 2) = dy2_drx0(j)$$

$$A(i + 2, 2) = dz2_drx0(j)$$

$$A(i, 3) = dx2_dry0(j)$$

$$A(i + 1, 3) = dy2_dry0(j)$$

$$A(i + 2, 3) = dz2_dry0(j)$$

$$A(i, 4) = dx2_drz0(j)$$

$$A(i + 1, 4) = dy2_drz0(j)$$

$$A(i + 2, 4) = dz2_drz0(j)$$

$$A(i, 5) = dx2_dTx0(j)$$

$$A(i + 1, 5) = dy2_dTx0(j)$$

$$A(i + 2, 5) = dz2_dTx0(j)$$

$$A(i, 6) = dx2_dT_y0(j)$$

$$A(i + 1, 6) = dy2_dT_y0(j)$$

$$A(i + 2, 6) = dz2_dT_y0(j)$$

```

A(i, 7) = dx2_dTz0(j)
A(i + 1, 7) = dy2_dTz0(j)
A(i + 2, 7) = dz2_dTz0(j)

```

```

i = i + 2

```

```

Next i

```

```

j = 0

```

```

'L matrix

```

```

For i = 1 To M

```

```

j = j + 1

```

```

    L(i, 1) = x2(j) - x0(j)

```

```

    L(i + 1, 1) = y2(j) - y0(j)

```

```

    L(i + 2, 1) = z2(j) - z0(j)

```

```

    i = i + 2

```

```

Next i

```

```

'A transpose

```

```

For i = 1 To 7

```

```

    For j = 1 To M

```

```

        t(i, j) = A(j, i)

```

```

    Next j

```

```

Next i

```

```

'A transpose * A

```

```

For i = 1 To 7

```

```

    For j = 1 To 7

```

```

        TA(i, j) = 0#

```

```

        For k = 1 To M

```

```

            TA(i, j) = TA(i, j) + t(i, k) * A(k, j)

```

```

        Next k

```

```

    Next j

```

```

Next i

```

```

' At*A inverse

```

```

For k = 1 To 7

```

```

For j = 1 To 7

```

```

    If j <> k Then TA(k, j) = TA(k, j) / TA(k, k)
Next j
TA(k, k) = 1 / TA(k, k)
For i = 1 To 7
If i <> k Then
    For j = 1 To 7
        If j <> k Then TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)
    Next j
    TA(i, k) = -TA(i, k) * TA(k, k)
End If

Next i
Next k

For i = 1 To 7
    For j = 1 To M
        TAT(i, j) = 0
        For k = 1 To 7
            TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)
        Next k
    Next j
Next i

' inv(A transpose * A) * L
For i = 1 To 7
    Xv(i, 1) = 0
    For k = 1 To M
        Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)
    Next k
Next i

s0 = s0 + Xv(1, 1)
rx0 = rx0 + Xv(2, 1)
ry0 = ry0 + Xv(3, 1)
rz0 = rz0 + Xv(4, 1)
Tx0 = Tx0 + Xv(5, 1)
Ty0 = Ty0 + Xv(6, 1)
Tz0 = Tz0 + Xv(7, 1)

```

Next iterat

```
s = s0
rx = rx0
ry = ry0
rz = rz0
Tx = Tx0
Ty = Ty0
Tz = Tz0

scal = s
rotation_x = rx
rotation_y = ry
rotation_z = rz
Translation_x = Tx
Translation_y = Ty
Translation_z = Tz
```

End If

If method.ListIndex = 4 Then ' Moldensky

```
Dim a1, a2, b1, b2, f1, f2 As Double
Dim Rm, Rn As Double
Dim da, df, dx, dy, dz As Double
Dim e2, lon2, lat2, h2 As Double
Dim rx10(0 To 1000), ry10(0 To 1000), h1(0 To 1000), rx20(0 To 1000), ry20(0 To 1000), h20(0 To 1000) As Double
Dim rx1(0 To 1000), ry1(0 To 1000), rz1(0 To 1000), rx2(0 To 1000), ry2(0 To 1000), rz2(0 To 1000) As Double
pie = 3.141592654
```

For i = 1 To cnt

```
rx10(i) = Val(fx1(i)) 'lat1 'source
ry10(i) = Val(fy1(i)) 'lon1
h1(i) = Val(fz1(i)) 'h1

rx20(i) = Val(fx2(i)) 'lat2 'target
ry20(i) = Val(fy2(i)) 'lon2
```

```

        h20(i) = Val(fz2(i))    'h2
    Next i

For i = 1 To cnt
    rx1(i) = rx10(i) * pie / 180    'lat_rad1    ' source
    ry1(i) = ry10(i) * pie / 180    'lon_rad1

    rx2(i) = rx20(i) * pie / 180    'lat_rad2    'target
    ry2(i) = ry20(i) * pie / 180    'lon_rad2
Next i

a1 = Val(Text1)
b1 = Val(Text2)
a2 = Val(Text3)
b2 = Val(Text4)

f1 = (a1 - b1) / a1
e2 = (a1 ^ 2 - b1 ^ 2) / (a1 ^ 2)
f2 = (a2 - b2) / a2

da = a2 - a1
df = f2 - f1

Rm = (a1 * (1 - e2)) / (1 - e2 * (Sin(rx1(1))) ^ 2) ^ 1.5
Rn = (a1) / (1 - e2 * (Sin(rx1(1))) ^ 2) ^ 0.5

For i = 1 To cnt

    drx_dx(i) = ((-1) * Sin(rx1(i)) * Cos(ry1(i))) / ((Rm + h1(i)) * Sin(0.000004848136811))
    drx_dy(i) = ((-1) * Sin(rx1(i)) * Sin(ry1(i))) / ((Rm + h1(i)) * Sin(0.000004848136811))
    drx_dz(i) = (Cos(rx1(i))) / ((Rm + h1(i)) * Sin(0.000004848136811))

    dry_dx(i) = (-1) * Sin(ry1(i)) / ((Rn + h1(i)) * (Cos(rx1(i))) * Sin(0.000004848136811))
    dry_dy(i) = (Cos(ry1(i))) / ((Rn + h1(i)) * (Cos(rx1(i))) * Sin(0.000004848136811))
    dry_dz(i) = 0

    dh_dx(i) = (Cos(rx1(i))) * (Cos(ry1(i)))
    dh_dy(i) = (Cos(rx1(i))) * (Sin(ry1(i)))
    dh_dz(i) = Sin(rx1(i))
Next i

```


M = cnt * 3

N = 3

ReDim A(1 To M, 1 To 3)

ReDim L(1 To M, 1 To 1)

ReDim t(1 To 3, 1 To M)

ReDim TA(1 To M, 1 To M)

ReDim TAT(1 To M, 1 To M)

ReDim Xv(1 To 3, 1 To 1)

j = 0

For i = 1 To M ' make A

 j = j + 1

 A(i, 1) = drx_dx(j)

 A(i + 1, 1) = dry_dx(j)

 A(i + 2, 1) = dh_dx(j)

 A(i, 2) = drx_dy(j)

 A(i + 1, 2) = dry_dy(j)

 A(i + 2, 2) = dh_dy(j)

 A(i, 3) = drx_dz(j)

 A(i + 1, 3) = dry_dz(j)

 A(i + 2, 3) = dh_dz(j)

 i = i + 2

Next i

' L matrix

j = 0

For i = 1 To M

 j = j + 1

 L(i, 1) = (rx20(j) - rx10(j)) * 3600

 L(i + 1, 1) = (ry20(j) - ry10(j)) * 3600

 L(i + 2, 1) = h20(j) - h1(j)

 i = i + 2

Next i

'A transpose

For i = 1 To 3

For j = 1 To M

t(i, j) = A(j, i)

Next j

Next i

' A transpose * A

For i = 1 To 3

For j = 1 To 3

TA(i, j) = 0#

For k = 1 To M

TA(i, j) = TA(i, j) + t(i, k) * A(k, j)

Next k

Next j

Next i

' At*A inverse

For k = 1 To 3

For j = 1 To 3

If j <> k Then TA(k, j) = TA(k, j) / TA(k, k)

Next j

TA(k, k) = 1 / TA(k, k)

For i = 1 To 3

If i <> k Then

For j = 1 To 3

If j <> k Then TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)

Next j

TA(i, k) = -TA(i, k) * TA(k, k)

End If

Next i

Next k

For i = 1 To 3

For j = 1 To M

TAT(i, j) = 0

For k = 1 To 3

TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)

Next k

```
Next j
Next i

' Inv(A transpose * A) *L
For i = 1 To 3
    Xv(i, 1) = 0
    For k = 1 To M
        Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)
    Next k
Next i
Next i

dx = Xv(1, 1)
dy = Xv(2, 1)
dz = Xv(3, 1)
Translation_x = dx
Translation_y = dy
Translation_z = dz

End If
End Sub

Private Sub Command11_Click()
    CommonDialog1.Filter = "Pt# Lon Lat h(*.*)|*.*|"
    CommonDialog1.ShowOpen

    If CommonDialog1.FileName <> "" Then
        Open CommonDialog1.FileName For Input As #1

        cnt = 0
        Do Until EOF(1)
            cnt = cnt + 1
            Input #1, ptnr(cnt), fx1(cnt), fy1(cnt), fz1(cnt)

            Loop
        Close #1
        If CommonDialog1.CancelError = False Then
            Command3.Enabled = 0
        End If
    End If
End Sub
```

```
Private Sub Command2_Click()
CommonDialog2.Filter = "Pt# Lon Lat h(*.*)|*.*|"
CommonDialog2.ShowOpen

If CommonDialog2.FileName <> "" Then
    Open CommonDialog2.FileName For Input As #2
    cnt = 0
    Do Until EOF(2)
        cnt = cnt + 1
        Input #2, ptrn(cnt), fx2(cnt), fy2(cnt), fz2(cnt)
    Loop
    Close #2
    If CommonDialog2.CancelError = False Then
        Command3.Enabled = True
    End If
End If
End Sub
```

```
Private Sub Command3_Click()
Dim iterat, i As Integer
Dim az1, az2 As Double

CommonDialog3.Filter = " X Y h rx ry rz t (*.txt)|*.txt|"
CommonDialog3.ShowSave
```

```
If CommonDialog3.FileName <> "" Then
Open CommonDialog3.FileName For Output As #3
```

```
pie = 3.14159265358979
```

```
If method.ListIndex = 0 Then    ' 7-parameter
    For i = 1 To cnt
        x1(i) = Val(fx1(i))    ' source
        y1(i) = Val(fy1(i))
        z1(i) = Val(fz1(i))

        x2(i) = Val(fx2(i))    'target
        y2(i) = Val(fy2(i))
        z2(i) = Val(fz2(i))
    Next i
```

M = cnt * 3

N = 7

ReDim A(1 To M, 1 To 7)

ReDim L(1 To M, 1 To 1)

ReDim t(1 To 7, 1 To M)

ReDim TA(1 To M, 1 To M)

ReDim TAT(1 To M, 1 To M)

ReDim Xv(1 To 7, 1 To 1)

' Iteration

az2 = Atan2(x2(2) - x2(2), y2(2) - y2(1))

az1 = Atan2(x1(2) - x1(2), y1(2) - y1(1))

s0 = 1

rx0 = 0

ry0 = 0

rz0 = az2 - az1

Tx0 = x2(1) - x1(1)

Ty0 = y2(1) - y1(1)

Tz0 = z2(1) - z1(1)

For iterat = 1 To 20

rx_rad0 = rx0 'omega

ry_rad0 = ry0 'fay

rz_rad0 = rz0 'kaba

m110 = Cos(ry_rad0) * Cos(rz_rad0)

m120 = Sin(rx_rad0) * Sin(ry_rad0) * Cos(rz_rad0) + Cos(rx_rad0) * Sin(rz_rad0)

m130 = Sin(rx_rad0) * Sin(rz_rad0) - Cos(rx_rad0) * Sin(ry_rad0) * Cos(rz_rad0)

m210 = -1 * Cos(ry_rad0) * Sin(rz_rad0)

m220 = Cos(rx_rad0) * Cos(rz_rad0) - Sin(rx_rad0) * Sin(ry_rad0) * Sin(rz_rad0)

m230 = Sin(rx_rad0) * Cos(rz_rad0) + Cos(rx_rad0) * Sin(ry_rad0) * Sin(rz_rad0)

m310 = Sin(ry_rad0)

m320 = -1 * Sin(rx_rad0) * Cos(ry_rad0)

m330 = Cos(rx_rad0) * Cos(ry_rad0)

For i = 1 To cnt

$$x0(i) = s0 * (m110 * x1(i) + m210 * y1(i) + m310 * z1(i)) + Tx0$$

$$y0(i) = s0 * (m120 * x1(i) + m220 * y1(i) + m320 * z1(i)) + Ty0$$

$$z0(i) = s0 * (m130 * x1(i) + m230 * y1(i) + m330 * z1(i)) + Tz0$$

$$dx2_dS0(i) = m110 * x1(i) + m210 * y1(i) + m310 * z1(i)$$

$$dx2_drx0(i) = 0$$

$$dx2_dry0(i) = s0 * ((-1) * \sin(ry_rad0) * \cos(rz_rad0) * x1(i) + \sin(ry_rad0) * \sin(rz_rad0) * y1(i) + \cos(ry_rad0) * z1(i))$$

$$dx2_drz0(i) = s0 * (m210 * x1(i) - m110 * y1(i))$$

$$dx2_dTx0(i) = 1$$

$$dx2_dTy0(i) = 0$$

$$dx2_dTz0(i) = 0$$

$$dy2_dS0(i) = m120 * x1(i) + m220 * y1(i) + m320 * z1(i)$$

$$dy2_drx0(i) = (-1) * s0 * (m130 * x1(i) + m230 * y1(i) + m330 * z1(i))$$

$$dy2_dry0(i) = s0 * (\sin(rx_rad0) * \cos(ry_rad0) * \cos(rz_rad0) * x1(i) - \sin(rx_rad0) * \cos(ry_rad0) * \sin(rz_rad0) * y1(i) + \sin(rx_rad0) * \sin(ry_rad0) * z1(i))$$

$$dy2_drz0(i) = s0 * (m220 * x1(i) - m120 * y1(i))$$

$$dy2_dTx0(i) = 0$$

$$dy2_dTy0(i) = 1$$

$$dy2_dTz0(i) = 0$$

$$dz2_dS0(i) = m130 * x1(i) + m230 * y1(i) + m330 * z1(i)$$

$$dz2_drx0(i) = s0 * (m120 * x1(i) + m220 * y1(i) + m320 * z1(i))$$

$$dz2_dry0(i) = s0 * ((-1) * \cos(rx_rad0) * \cos(ry_rad0) * \cos(rz_rad0) * x1(i) + \cos(rx_rad0) * \cos(ry_rad0) * \sin(rz_rad0) * y1(i) - \cos(rx_rad0) * \sin(ry_rad0) * z1(i))$$

$$dz2_drz0(i) = s0 * (m230 * x1(i) - m130 * y1(i))$$

$$dz2_dTx0(i) = 0$$

$$dz2_dTy0(i) = 0$$

$$dz2_dTz0(i) = 1$$

Next i

j = 0

For i = 1 To M ' make A

j = j + 1

$$A(i, 1) = dx2_dS0(j)$$

$$A(i + 1, 1) = dy2_dS0(j)$$

$$A(i + 2, 1) = dz2_dS0(j)$$

$$A(i, 2) = dx2_drx0(j)$$

$$A(i + 1, 2) = dy2_drx0(j)$$

$$A(i + 2, 2) = dz2_drx0(j)$$

$$A(i, 3) = dx2_dry0(j)$$

$$A(i + 1, 3) = dy2_dry0(j)$$

$$A(i + 2, 3) = dz2_dry0(j)$$

$$A(i, 4) = dx2_drz0(j)$$

$$A(i + 1, 4) = dy2_drz0(j)$$

$$A(i + 2, 4) = dz2_drz0(j)$$

$$A(i, 5) = dx2_dTx0(j)$$

$$A(i + 1, 5) = dy2_dTx0(j)$$

$$A(i + 2, 5) = dz2_dTx0(j)$$

$$A(i, 6) = dx2_dTy0(j)$$

$$A(i + 1, 6) = dy2_dTy0(j)$$

$$A(i + 2, 6) = dz2_dTy0(j)$$

$$A(i, 7) = dx2_dTz0(j)$$

$$A(i + 1, 7) = dy2_dTz0(j)$$

$$A(i + 2, 7) = dz2_dTz0(j)$$

$$i = i + 2$$

Next i

$$j = 0$$

' L matrix

For i = 1 To M

$$j = j + 1$$

$$L(i, 1) = x2(j) - x0(j)$$

$$L(i + 1, 1) = y2(j) - y0(j)$$

$$L(i + 2, 1) = z2(j) - z0(j)$$

$$i = i + 2$$

Next i

'A transpose

For i = 1 To 7

For j = 1 To M

t(i, j) = A(j, i)

Next j

Next i

'A transpose * A

For i = 1 To 7

For j = 1 To 7

TA(i, j) = 0#

For k = 1 To M

TA(i, j) = TA(i, j) + t(i, k) * A(k, j)

Next k

Next j

Next i

' At*A inverse

For k = 1 To 7

For j = 1 To 7

If j <> k Then TA(k, j) = TA(k, j) / TA(k, k)

Next j

TA(k, k) = 1 / TA(k, k)

For i = 1 To 7

If i <> k Then

For j = 1 To 7

If j <> k Then TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)

Next j

TA(i, k) = -TA(i, k) * TA(k, k)

End If

Next i

Next k

For i = 1 To 7

For j = 1 To M

TAT(i, j) = 0

For k = 1 To 7

TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)

Next k

```

    Next j
Next i

' Inv(A transpose * A) *L

For i = 1 To 7
    Xv(i, 1) = 0
    For k = 1 To M
        Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)
    Next k
Next i

s0 = s0 + Xv(1, 1)
rx0 = rx0 + Xv(2, 1)
ry0 = ry0 + Xv(3, 1)
rz0 = rz0 + Xv(4, 1)
Tx0 = Tx0 + Xv(5, 1)
Ty0 = Ty0 + Xv(6, 1)
Tz0 = Tz0 + Xv(7, 1)

Next iterat

s = s0
rx = (rx0) * 180 / pie
ry = (ry0) * 180 / pie
rz = (rz0) * 180 / pie
Tx = Tx0
Ty = Ty0
Tz = Tz0

End If          ' 7 parameter

If method.ListIndex = 1 Then ' 6 parameter
    For i = 1 To cnt
        x1(i) = Val(fx1(i)) ' source
        y1(i) = Val(fy1(i))
        z1(i) = Val(fz1(i))

        x2(i) = Val(fx2(i)) 'target
        y2(i) = Val(fy2(i))

```

z2(i) = Val(fz2(i))

Next i

M = cnt * 3

N = 6

ReDim A(1 To M, 1 To 6)

ReDim L(1 To M, 1 To 1)

ReDim t(1 To 6, 1 To M)

ReDim TA(1 To M, 1 To M)

ReDim TAT(1 To M, 1 To M)

ReDim Xv(1 To 6, 1 To 1)

' iteration

az2 = Atan2(x2(2) - x2(2), y2(2) - y2(1))

az1 = Atan2(x1(2) - x1(2), y1(2) - y1(1))

s0 = 1

rx0 = 0

ry0 = 0

rz0 = az2 - az1

Tx0 = x2(1) - x1(1)

Ty0 = y2(1) - y1(1)

Tz0 = z2(1) - z1(1)

For iterat = 1 To 20

rx_rad0 = rx0 'omega

ry_rad0 = ry0 'fay

rz_rad0 = rz0 'kaba

m110 = Cos(ry_rad0) * Cos(rz_rad0)

m120 = Sin(rx_rad0) * Sin(ry_rad0) * Cos(rz_rad0) + Cos(rx_rad0) * Sin(rz_rad0)

m130 = Sin(rx_rad0) * Sin(rz_rad0) - Cos(rx_rad0) * Sin(ry_rad0) * Cos(rz_rad0)

m210 = -1 * Cos(ry_rad0) * Sin(rz_rad0)

m220 = Cos(rx_rad0) * Cos(rz_rad0) - Sin(rx_rad0) * Sin(ry_rad0) * Sin(rz_rad0)

m230 = Sin(rx_rad0) * Cos(rz_rad0) + Cos(rx_rad0) * Sin(ry_rad0) * Sin(rz_rad0)

m310 = Sin(ry_rad0)

$$m320 = -1 * \sin(rx_rad0) * \cos(ry_rad0)$$

$$m330 = \cos(rx_rad0) * \cos(ry_rad0)$$

For i = 1 To cnt

$$x0(i) = s0 * (m110 * x1(i) + m210 * y1(i) + m310 * z1(i)) + Tx0$$

$$y0(i) = s0 * (m120 * x1(i) + m220 * y1(i) + m320 * z1(i)) + Ty0$$

$$z0(i) = s0 * (m130 * x1(i) + m230 * y1(i) + m330 * z1(i)) + Tz0$$

'scale is fixed to S=1 , not unknown

$$dx2_drx0(i) = 0$$

$$dx2_dry0(i) = s0 * ((-1) * \sin(ry_rad0) * \cos(rz_rad0) * x1(i) + \sin(ry_rad0) * \sin(rz_rad0) * y1(i) + \cos(ry_rad0) * z1(i))$$

$$dx2_drz0(i) = s0 * (m210 * x1(i) - m110 * y1(i))$$

$$dx2_dTx0(i) = 1$$

$$dx2_dTy0(i) = 0$$

$$dx2_dTz0(i) = 0$$

$$dy2_drx0(i) = (-1) * s0 * (m130 * x1(i) + m230 * y1(i) + m330 * z1(i))$$

$$dy2_dry0(i) = s0 * (\sin(rx_rad0) * \cos(ry_rad0) * \cos(rz_rad0) * x1(i) - \sin(rx_rad0) * \cos(ry_rad0) * \sin(rz_rad0) * y1(i) + \sin(rx_rad0) * \sin(ry_rad0) * z1(i))$$

$$dy2_drz0(i) = s0 * (m220 * x1(i) - m120 * y1(i))$$

$$dy2_dTx0(i) = 0$$

$$dy2_dTy0(i) = 1$$

$$dy2_dTz0(i) = 0$$

$$dz2_drx0(i) = s0 * (m120 * x1(i) + m220 * y1(i) + m320 * z1(i))$$

$$dz2_dry0(i) = s0 * ((-1) * \cos(rx_rad0) * \cos(ry_rad0) * \cos(rz_rad0) * x1(i) + \cos(rx_rad0) * \cos(ry_rad0) * \sin(rz_rad0) * y1(i) - \cos(rx_rad0) * \sin(ry_rad0) * z1(i))$$

$$dz2_drz0(i) = s0 * (m230 * x1(i) - m130 * y1(i))$$

$$dz2_dTx0(i) = 0$$

$$dz2_dTy0(i) = 0$$

$$dz2_dTz0(i) = 1$$

Next i

Close #1

j = 0

For i = 1 To M ' make A

j = j + 1

$$A(i, 1) = dx2_drx0(j)$$

$$A(i + 1, 1) = dy2_drx0(j)$$

$$A(i + 2, 1) = dz2_drx0(j)$$

$$A(i, 2) = dx2_dry0(j)$$

$$A(i + 1, 2) = dy2_dry0(j)$$

$$A(i + 2, 2) = dz2_dry0(j)$$

$$A(i, 3) = dx2_drz0(j)$$

$$A(i + 1, 3) = dy2_drz0(j)$$

$$A(i + 2, 3) = dz2_drz0(j)$$

$$A(i, 4) = dx2_dTx0(j)$$

$$A(i + 1, 4) = dy2_dTx0(j)$$

$$A(i + 2, 4) = dz2_dTx0(j)$$

$$A(i, 5) = dx2_dTy0(j)$$

$$A(i + 1, 5) = dy2_dTy0(j)$$

$$A(i + 2, 5) = dz2_dTy0(j)$$

$$A(i, 6) = dx2_dTz0(j)$$

$$A(i + 1, 6) = dy2_dTz0(j)$$

$$A(i + 2, 6) = dz2_dTz0(j)$$

$$i = i + 2$$

Next i

$$j = 0$$

'L matrix

For i = 1 To M

$$j = j + 1$$

$$L(i, 1) = x2(j) - x0(j)$$

$$L(i + 1, 1) = y2(j) - y0(j)$$

$$L(i + 2, 1) = z2(j) - z0(j)$$

$$i = i + 2$$

Next i

'A transpose

For i = 1 To 6

For j = 1 To M

t(i, j) = A(j, i)

Next j

Next i

'A transpose * A

For i = 1 To 6

For j = 1 To 6

TA(i, j) = 0#

For k = 1 To M

TA(i, j) = TA(i, j) + t(i, k) * A(k, j)

Next k

Next j

Next i

' At*A inverse

For k = 1 To 6

For j = 1 To 6

If j <> k Then TA(k, j) = TA(k, j) / TA(k, k)

Next j

TA(k, k) = 1 / TA(k, k)

For i = 1 To 6

If i <> k Then

For j = 1 To 6

If j <> k Then TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)

Next j

TA(i, k) = -TA(i, k) * TA(k, k)

End If

Next i

Next k

For i = 1 To 6

For j = 1 To M

TAT(i, j) = 0

For k = 1 To 6

TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)

```

Next k
Next j
Next i

' Inv(A transpose * A) *L
For i = 1 To 6
    Xv(i, 1) = 0
    For k = 1 To M
        Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)
    Next k
Next i

```

```

    rx0 = rx0 + Xv(1, 1)
    ry0 = ry0 + Xv(2, 1)
    rz0 = rz0 + Xv(3, 1)
    Tx0 = Tx0 + Xv(4, 1)
    Ty0 = Ty0 + Xv(5, 1)
    Tz0 = Tz0 + Xv(6, 1)
Next iterat

```

```

s = s0
rx = (rx0) * 180 / pie
ry = (ry0) * 180 / pie
rz = (rz0) * 180 / pie
Tx = Tx0
Ty = Ty0
Tz = Tz0

```

```

End If          '6 parameter

```

```

If method.ListIndex = 2 Then ' 3 parameter

```

```

    For i = 1 To cnt
        x1(i) = Val(fx1(i)) ' source
        y1(i) = Val(fy1(i))
        z1(i) = Val(fz1(i))

        x2(i) = Val(fx2(i)) 'target
        y2(i) = Val(fy2(i))
        z2(i) = Val(fz2(i))
    Next i

```

APPENDIX – (C)

$M = cnt * 3$

$N = 3$

ReDim A(1 To M, 1 To 3)

ReDim L(1 To M, 1 To 1)

ReDim t(1 To 3, 1 To M)

ReDim TA(1 To M, 1 To M)

ReDim TAT(1 To M, 1 To M)

ReDim Xv(1 To 3, 1 To 1)

j = 0

For i = 1 To M ' make A

j = j + 1

A(i, 1) = 1

A(i + 1, 1) = 0

A(i + 2, 1) = 0

A(i, 2) = 0

A(i + 1, 2) = 1

A(i + 2, 2) = 0

A(i, 3) = 0

A(i + 1, 3) = 0

A(i + 2, 3) = 1

i = i + 2

Next i

j = 0

'L matrix

For i = 1 To M

j = j + 1

L(i, 1) = x2(j)

L(i + 1, 1) = y2(j)

L(i + 2, 1) = z2(j)

i = i + 2

Next i

'A transpose

For i = 1 To 3

For j = 1 To M

t(i, j) = A(j, i)

Next j

Next i

'A transpose * A

For i = 1 To 3

For j = 1 To 3

TA(i, j) = 0#

For k = 1 To M

TA(i, j) = TA(i, j) + t(i, k) * A(k, j)

Next k

Next j

Next i

' At*A inverse

For k = 1 To 3

For j = 1 To 3

If j <> k Then TA(k, j) = TA(k, j) / TA(k, k)

Next j

TA(k, k) = 1 / TA(k, k)

For i = 1 To 3

If i <> k Then

For j = 1 To 3

If j <> k Then TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)

Next j

TA(i, k) = -TA(i, k) * TA(k, k)

End If

Next i

Next k

For i = 1 To 3

For j = 1 To M

TAT(i, j) = 0

For k = 1 To 3

TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)

Next k

```

Next j
Next i

' inv(A transpose * A) *L
For i = 1 To 3
  Xv(i, 1) = 0
  For k = 1 To M
    Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)
  Next k
Next i

```

```

Tx0 = Xv(1, 1)
Ty0 = Xv(2, 1)
Tz0 = Xv(3, 1)

```

```

' Next iterat

```

```

Tx = Tx0
Ty = Ty0
Tz = Tz0

```

```

End If      ' 3parameter

```

```

If method.ListIndex = 3 Then ' Helmert

```

```

For i = 1 To cnt
  x1(i) = Val(fx1(i)) ' source
  y1(i) = Val(fy1(i))
  z1(i) = Val(fz1(i))

  x2(i) = Val(fx2(i)) 'target
  y2(i) = Val(fy2(i))
  z2(i) = Val(fz2(i))

```

```

Next i

```

```

M = cnt * 3
N = 7

```

```

ReDim A(1 To M, 1 To 7)
ReDim L(1 To M, 1 To 1)
ReDim t(1 To 7, 1 To M)
ReDim TA(1 To M, 1 To M)
ReDim TAT(1 To M, 1 To M)
ReDim Xv(1 To 7, 1 To 1)

```

' iteration

```

s0 = 1
rx0 = 0
ry0 = 0
rz0 = 0
Tx0 = x2(1) - x1(1)
Ty0 = y2(1) - y1(1)
Tz0 = z2(1) - z1(1)

```

For iterat = 1 To 1

For i = 1 To cnt

```

dx2_dS0(i) = x1(i) + rz0 * y1(i) - ry0 * z1(i)
dx2_drx0(i) = 0
dx2_dry0(i) = -1 * s0 * z1(i)
dx2_drz0(i) = s0 * y1(i)
dx2_dTx0(i) = 1
dx2_dTy0(i) = 0
dx2_dTz0(i) = 0

dy2_dS0(i) = -1 * rz0 * x1(i) + y1(i) + rx0 * z1(i)
dy2_drx0(i) = s0 * z1(i)
dy2_dry0(i) = 0
dy2_drz0(i) = -1 * s0 * x1(i)
dy2_dTx0(i) = 0
dy2_dTy0(i) = 1
dy2_dTz0(i) = 0

dz2_dS0(i) = ry0 * x1(i) - rx0 * y1(i) + z1(i)
dz2_drx0(i) = -1 * s0 * y1(i)
dz2_dry0(i) = s0 * x1(i)
dz2_drz0(i) = 0
dz2_dTx0(i) = 0

```


$$dz2_dT_y0(i) = 0$$

$$dz2_dT_z0(i) = 1$$

$$x0(i) = s0 * (x1(i) + rz0 * y1(i) - ry0 * z1(i)) + Tx0$$

$$y0(i) = s0 * (-rz0 * x1(i) + y1(i) + rx0 * z1(i)) + Ty0$$

$$z0(i) = s0 * (ry0 * x1(i) - rx0 * y1(i) + z1(i)) + Tz0$$

Next i

j = 0

For i = 1 To M ' make A

j = j + 1

$$A(i, 1) = dx2_dS0(j)$$

$$A(i + 1, 1) = dy2_dS0(j)$$

$$A(i + 2, 1) = dz2_dS0(j)$$

$$A(i, 2) = dx2_drx0(j)$$

$$A(i + 1, 2) = dy2_drx0(j)$$

$$A(i + 2, 2) = dz2_drx0(j)$$

$$A(i, 3) = dx2_dry0(j)$$

$$A(i + 1, 3) = dy2_dry0(j)$$

$$A(i + 2, 3) = dz2_dry0(j)$$

$$A(i, 4) = dx2_drz0(j)$$

$$A(i + 1, 4) = dy2_drz0(j)$$

$$A(i + 2, 4) = dz2_drz0(j)$$

$$A(i, 5) = dx2_dTx0(j)$$

$$A(i + 1, 5) = dy2_dTx0(j)$$

$$A(i + 2, 5) = dz2_dTx0(j)$$

$$A(i, 6) = dx2_dT_y0(j)$$

$$A(i + 1, 6) = dy2_dT_y0(j)$$

$$A(i + 2, 6) = dz2_dT_y0(j)$$

$$A(i, 7) = dx2_dT_z0(j)$$

$$A(i + 1, 7) = dy2_dT_z0(j)$$

$$A(i + 2, 7) = dz2_dT_z0(j)$$

i = i + 2

Next i

```

j = 0
'L matrix
For i = 1 To M
  j = j + 1
  L(i, 1) = x2(j) - x0(j)
  L(i + 1, 1) = y2(j) - y0(j)
  L(i + 2, 1) = z2(j) - z0(j)
  i = i + 2
Next i

'A transpose
For i = 1 To 7
  For j = 1 To M
    t(i, j) = A(j, i)
  Next j
Next i

'A transpose * A
For i = 1 To 7
  For j = 1 To 7
    TA(i, j) = 0#
    For k = 1 To M
      TA(i, j) = TA(i, j) + t(i, k) * A(k, j)
    Next k
  Next j
Next i

' At*A inverse
For k = 1 To 7
  For j = 1 To 7
    If j <> k Then TA(k, j) = TA(k, j) / TA(k, k)
  Next j
  TA(k, k) = 1 / TA(k, k)
For i = 1 To 7
  If i <> k Then
    For j = 1 To 7
      If j <> k Then TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)
    Next j
    TA(i, k) = -TA(i, k) * TA(k, k)
  End If

```

Next i

Next k

For i = 1 To 7

For j = 1 To M

TAT(i, j) = 0

For k = 1 To 7

TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)

Next k

Next j

Next i

' inv(A transpose * A) *L

For i = 1 To 7

Xv(i, 1) = 0

For k = 1 To M

Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)

Next k

Next i

s0 = s0 + Xv(1, 1)

rx0 = rx0 + Xv(2, 1)

ry0 = ry0 + Xv(3, 1)

rz0 = rz0 + Xv(4, 1)

Tx0 = Tx0 + Xv(5, 1)

Ty0 = Ty0 + Xv(6, 1)

Tz0 = Tz0 + Xv(7, 1)

Next iterat

s = s0

rx = rx0

ry = ry0

rz = rz0

Tx = Tx0

Ty = Ty0

Tz = Tz0

End If 'helmert

If method.ListIndex = 4 Then ' Moldensky

Dim a1, a2, b1, b2, f1, f2 As Double

Dim Rm, Rn As Double

Dim da, df, dx, dy, dz As Double

Dim e2, lon2, lat2, h2 As Double

Dim rx10(0 To 1000), ry10(0 To 1000), h1(0 To 1000), rx20(0 To 1000), ry20(0 To 1000), h20(0 To 1000) As Double

Dim rx1(0 To 1000), ry1(0 To 1000), rz1(0 To 1000), rx2(0 To 1000), ry2(0 To 1000), rz2(0 To 1000) As Double

pie = 3.141592654

For i = 1 To cnt

rx10(i) = Val(fx1(i)) 'lat1 ' source

ry10(i) = Val(fy1(i)) 'lon1

h1(i) = Val(fz1(i)) 'h1

rx20(i) = Val(fx2(i)) 'lat2 'target

ry20(i) = Val(fy2(i)) 'lon2

h20(i) = Val(fz2(i)) 'h2

Next i

For i = 1 To cnt

rx1(i) = rx10(i) * pie / 180 'lat_rad1 ' source

ry1(i) = ry10(i) * pie / 180 'lon_rad1

rx2(i) = rx20(i) * pie / 180 'lat_rad2 'target

ry2(i) = ry20(i) * pie / 180 'lon_rad2

Next i

a1 = Val(Text1)

b1 = Val(Text2)

a2 = Val(Text3)

b2 = Val(Text4)

f1 = (a1 - b1) / a1

e2 = (a1 ^ 2 - b1 ^ 2) / (a1 ^ 2)

f2 = (a2 - b2) / a2

da = a2 - a1

df = f2 - f1

$$R_m = (a_1 * (1 - e_2)) / (1 - e_2 * (\sin(rx_1(1)))^2)^{1.5}$$

$$R_n = (a_1) / (1 - e_2 * (\sin(rx_1(1)))^2)^{0.5}$$

For i = 1 To cnt

$$drx_dx(i) = ((-1) * \sin(rx_1(i)) * \cos(ry_1(i))) / ((R_m + h_1(i)) * \sin(0.000004848136811))$$

$$drx_dy(i) = ((-1) * \sin(rx_1(i)) * \sin(ry_1(i))) / ((R_m + h_1(i)) * \sin(0.000004848136811))$$

$$drx_dz(i) = (\cos(rx_1(i))) / ((R_m + h_1(i)) * \sin(0.000004848136811))$$

$$dry_dx(i) = (-1) * \sin(ry_1(i)) / ((R_n + h_1(i)) * (\cos(rx_1(i)) * \sin(0.000004848136811)))$$

$$dry_dy(i) = (\cos(ry_1(i))) / ((R_n + h_1(i)) * (\cos(rx_1(i)) * \sin(0.000004848136811)))$$

$$dry_dz(i) = 0$$

$$dh_dx(i) = (\cos(rx_1(i))) * (\cos(ry_1(i)))$$

$$dh_dy(i) = (\cos(rx_1(i))) * (\sin(ry_1(i)))$$

$$dh_dz(i) = \sin(rx_1(i))$$

Next i

$$M = cnt * 3$$

$$N = 3$$

$$\text{ReDim } A(1 \text{ To } M, 1 \text{ To } 3)$$

$$\text{ReDim } L(1 \text{ To } M, 1 \text{ To } 1)$$

$$\text{ReDim } t(1 \text{ To } 3, 1 \text{ To } M)$$

$$\text{ReDim } TA(1 \text{ To } M, 1 \text{ To } M)$$

$$\text{ReDim } TAT(1 \text{ To } M, 1 \text{ To } M)$$

$$\text{ReDim } Xv(1 \text{ To } 3, 1 \text{ To } 1)$$

$$j = 0$$

For i = 1 To M ' make A

$$j = j + 1$$

$$A(i, 1) = drx_dx(j)$$

$$A(i + 1, 1) = dry_dx(j)$$

$$A(i + 2, 1) = dh_dx(j)$$

$$A(i, 2) = drx_dy(j)$$

$$A(i + 1, 2) = dry_dy(j)$$

$$A(i + 2, 2) = dh_dy(j)$$

APPENDIX – (C)

$A(i, 3) = drx_dz(j)$

$A(i + 1, 3) = dry_dz(j)$

$A(i + 2, 3) = dh_dz(j)$

$i = i + 2$

Next i

$j = 0$

'L matrix

For i = 1 To M

$j = j + 1$

$L(i, 1) = (rx20(j) - rx10(j)) * 3600$

$L(i + 1, 1) = (ry20(j) - ry10(j)) * 3600$

$L(i + 2, 1) = h20(j) - h1(j)$

$i = i + 2$

Next i

'A transpose

For i = 1 To 3

For j = 1 To M

$t(i, j) = A(j, i)$

Next j

Next i

'A transpose * A

For i = 1 To 3

For j = 1 To 3

$TA(i, j) = 0\#$

For k = 1 To M

$TA(i, j) = TA(i, j) + t(i, k) * A(k, j)$

Next k

Next j

Next i

'At*A inverse

For k = 1 To 3

For j = 1 To 3

If j <> k Then $TA(k, j) = TA(k, j) / TA(k, k)$

Next j

$TA(k, k) = 1 / TA(k, k)$

For i = 1 To 3

If i <> k Then

For j = 1 To 3

If j <> k Then $TA(i, j) = TA(i, j) - TA(i, k) * TA(k, j)$

Next j

$TA(i, k) = -TA(i, k) * TA(k, k)$

End If

Next i

Next k

For i = 1 To 3

For j = 1 To M

$TAT(i, j) = 0$

For k = 1 To 3

$TAT(i, j) = TAT(i, j) + TA(i, k) * t(k, j)$

Next k

Next j

Next i

' inv(A transpose * A) *L

For i = 1 To 3

$Xv(i, 1) = 0$

For k = 1 To M

$Xv(i, 1) = Xv(i, 1) + TAT(i, k) * L(k, 1)$

Next k

Next i

$dx = Xv(1, 1)$

$dy = Xv(2, 1)$

$dz = Xv(3, 1)$

Translation_x = dx

Translation_y = dy

Translation_z = dz

End If

End If

End Sub

Private Sub method_Click()

If method.ListIndex = 0 Then ' 7-parameter simillarity

Translation_x.Visible = True

Translation_y.Visible = True

Translation_z.Visible = True

rotation_x.Visible = True

rotation_y.Visible = True

rotation_z.Visible = True

Frame5.Visible = 0

Combo1.Visible = 0

Combo2.Visible = 0

Text1.Visible = 0

Text2.Visible = 0

Text3.Visible = 0

Text4.Visible = 0

Label16.Visible = 0

Label17.Visible = 0

Label18.Visible = 0

Label19.Visible = 0

Label1.Visible = True

Label2.Visible = True

Label3.Visible = True

Label4.Visible = True

Label5.Visible = True

Label6.Visible = True

Label7.Visible = True

```
Label1.Caption = "Rotation_x"  
Label2.Caption = "Rotation_y"  
Label3.Caption = "Rotation_z"
```

```
scal.Visible = True
```

```
End If
```

```
If method.ListIndex = 1 Then      ' 6-parameter simillarity
```

```
    Translation_x.Visible = True  
    Translation_y.Visible = True  
    Translation_z.Visible = True
```

```
    rotation_x.Visible = True  
    rotation_y.Visible = True  
    rotation_z.Visible = True
```

```
    Frame5.Visible = 0  
    Combo1.Visible = 0  
    Combo2.Visible = 0
```

```
    Text1.Visible = 0  
    Text2.Visible = 0  
    Text3.Visible = 0  
    Text4.Visible = 0
```

```
    Label16.Visible = 0  
    Label17.Visible = 0  
    Label18.Visible = 0  
    Label19.Visible = 0
```

```
    Label1.Visible = True  
    Label2.Visible = True  
    Label3.Visible = True  
    Label4.Visible = True  
    Label5.Visible = True  
    Label6.Visible = True  
    Label7.Visible = False
```

```
    Label1.Caption = "Rotation_x"
```

Label2.Caption = "Rotation_y"

Label3.Caption = "Rotation_z"

scal.Visible = False

End If

If method.ListIndex = 2 Then ' 3-parameter simillarity

Translation_x.Visible = True

Translation_y.Visible = True

Translation_z.Visible = True

rotation_x.Visible = 0

rotation_y.Visible = 0

rotation_z.Visible = 0

Text1.Visible = 0

Text2.Visible = 0

Text3.Visible = 0

Text4.Visible = 0

Label16.Visible = 0

Label17.Visible = 0

Label18.Visible = 0

Label19.Visible = 0

Frame5.Visible = 0

Combo1.Visible = 0

Combo2.Visible = 0

Label1.Visible = 0

Label2.Visible = 0

Label3.Visible = 0

Label4.Visible = True

Label5.Visible = True

Label6.Visible = 1

Label7.Visible = 0

Label1.Caption = "Rotation_x"

Label2.Caption = "Rotation_y"

```
Label3.Caption = "Rotation_z"

scal.Visible = 0

End If

If method.ListIndex = 3 Then      ' Helmert

    Translation_x.Visible = True
    Translation_y.Visible = True
    Translation_z.Visible = True

    rotation_x.Visible = 1
    rotation_y.Visible = 1
    rotation_z.Visible = 1

    Text1.Visible = 0
    Text2.Visible = 0
    Text3.Visible = 0
    Text4.Visible = 0

    Label16.Visible = 0
    Label17.Visible = 0
    Label18.Visible = 0
    Label19.Visible = 0

    Frame5.Visible = 0
    Combo1.Visible = 0
    Combo2.Visible = 0

    Label1.Visible = True
    Label2.Visible = True
    Label3.Visible = True
    Label4.Visible = True
    Label5.Visible = True
    Label6.Visible = True
    Label7.Visible = True

    Label1.Caption = "Rotation_x"
    Label2.Caption = "Rotation_y"
```

APPENDIX – (C)

```
Label3.Caption = "Rotation_z"

scal.Visible = 1

End If

If method.ListIndex = 4 Then      ' Moldensky
    Translation_x.Visible = 1
    Translation_y.Visible = 1
    Translation_z.Visible = 1

    Frame5.Visible = 1
    Combo1.Visible = 1
    Combo2.Visible = 1

    Label3.Visible = 1
    Label1.Visible = 0
    Label2.Visible = 0
    Label3.Visible = 0
    Label7.Visible = 0

    Text1.Visible = True
    Text2.Visible = True
    Text3.Visible = True
    Text4.Visible = True

    Label16.Visible = True
    Label17.Visible = True
    Label18.Visible = True
    Label19.Visible = True

    Label4.Caption = "Translation_x"
    Label5.Caption = "Translation_x"
    Label6.Caption = "Translation_x"

    rotation_x.Visible = 0
    rotation_y.Visible = 0
    rotation_z.Visible = 0

    scal.Visible = 0
```