GEOMAGNETIC LOCALIZATION OF UNDERWATER VEHICLES BASED ON MATCHING ALGORITHM

COLLEGE OF
ELECTRICAL AND MECHANICAL ENGINEERING
NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY
RAWALPINDI
2009

College of Electrical & Mechanical Engineering

MS-54 (MTS)

THESIS REPORT

GEOMAGNETIC LOCALIZATION OF UNDERWATER VEHICLES BASED ON MATCHING ALGORITHM

Submitted to the Department of Mechatronics Engineering

in partial fulfillment of the requirements

for the degree of

Masters of Engineering
in
Mechatronics

2009

Submitted By:
Mr. Muhammad Ejaz
2006-NUST-MS PhD-MTS-05

ABSTRACT

The trend of ocean exploration puts forward higher requirements for automatic vehicles, while navigation system has been the development constraint for them. “Nature” reported that some sea turtles respond to geomagnetic map for their localization. The actual mechanism is not clear, but it offers a sign for underwater navigation research. Matching algorithms can be used for this purpose, which can match the path followed by the vehicle on a map. The traditional iterative closest point (ICP) algorithm for image registration combined with Inertial Navigation System (INS) can be used for location determination. This report emphasizes on how to modify and improve ICP algorithm according to underwater geomagnetic localization.

The algorithm’s applicability is concluded mathematically from the localization algorithms. Then the simulation experiments are carried out, which are based on real geomagnetic data, to show the validation of this algorithm and good results are obtained.

The simulation is done in MATLAB 7.4.0. The geomagnetic data used is from the standard WMM2005 (World Magnetic Map) of January 1, 2009, 100m below sea-level.

ACKNOWLEDGEMENTS

All kinds of appreciations are for “ALLAH” who created the universe and bestowed the mankind with knowledge and wisdom to search for his secrets, whose uniqueness and oneness is unchangeable and who is the most merciful, bountiful and gracious, whose grace enabled me to complete this thesis.

I deeply express my gratitude and appreciation to my research supervisor “Col. Dr. Javaid Iqbal” and co-supervisor “Mr. Nasir Ahsan” for their enthusiastic supervision, keen interest, stimulating discussion and unrivalled knowledge putting freely at my disposal.

I express my heart full gratitude to my parents, for their constant encouragement during whole life span and for providing me all possible facilities and cooperation during my whole career.

I am also highly indebted to my respective family members and friends for their prayers and moral supports throughout my career. Their happiness will know no boundaries when I will receive the masters degree from honorable National University of Science and Technology, Pakistan.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

Earth is covered by about 70 % of oceans. Oceans are a huge storage of resources. Therefore the need for ocean exploration arises. Autonomous robots which travel underwater named Autonomous Underwater Vehicles (AUV) or Unmanned Undersea Vehicles (UUV) are used for this exploration purpose. Furthermore, these vehicles can be used in military purposes and for the study of marine life. Submarines can also be used but due to dangerous conditions present underwater, the need of unmanned vehicles has increased. At present there are only a few submarines which can dive beyond 1000m and only one can dive down to 6500m. On the other hand, there are many unmanned vehicles that can operate to 6000m and one is capable of voyaging to the ocean's deepest location of 11000m [19].

But an accurate positioning system is a problem for its application. Using cables reduces its maneuverability, while autonomous localization schemes still have drawbacks. Inertial navigation system (INS) has the drawback of drift-error accumulation. While electromagnetic waves, from Global Positioning System (GPS) or any other source, encounter high attenuation in water. So there should be some research to make a novel and accurate positioning system for places where all other systems fail to work.

Recently, researchers have found that some animals use the magnetic field of the earth for navigation, that include ants [1] , pigeons [2], turtles [3], honeybees [4], lobsters [6], sharks [7] etc. NATURE reported a research about the green sea-turtles called Chelonia Mydas that they use geomagnetic field as a reference for their navigation [5]. Loggerhead turtles *Caretta caretta* also use geomagnetic map for navigation [3]. Scientists produced a magnetic field similar to the geomagnetic field in the oceans where the turtles belonged by using a coil. The turtles were showing a behavior, as if they were sensing the field. This shows that for navigation of turtles geomagnetic field can be a reference. Though the actual mechanism for navigation is still unclear, but it introduces one feasible way for underwater positioning. Therefore, this opens a door for research in the field of underwater localization and navigation.

1.1 The Magnetic Field

The lines of force surrounding a permanent magnet or a moving charged particle is called magnetic field [17]. The magnetic field at any given point is specified by both a direction and a magnitude; as such it is a vector field. The direction of this field is considered to diverge at the North Pole and converge at the South Pole of a magnet as shown in figure 1.

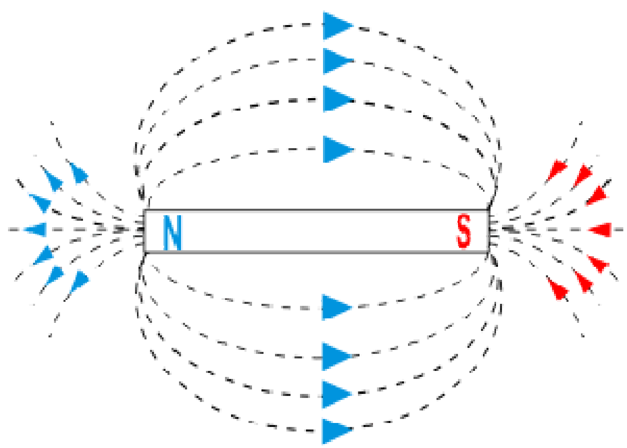

Figure 1: *A bar magnet and magnetic field lines*

1.1.1 Magnetism

The term magnetism is used to describe how materials respond on the microscopic level to an applied magnetic field; to categorize the magnetic phase of a material. For example, ferromagnetism is the most well known form of magnetism such that some ferromagnetic materials produce their own persistent magnetic field. Some well-known ferromagnetic materials that exhibit easily detectable magnetic properties (to form magnets) are nickel, iron, cobalt, gadolinium and their alloys. However, all materials are influenced to greater or lesser degree by the presence of a magnetic field. The property of some materials which are attracted to a magnetic field is called paramagnetism; others which are repulsed by a magnetic field, is called diamagnetism. Substances that are negligibly affected by magnetic fields are known as non-magnetic substances. They include copper, aluminum, water, and gases.

1.2 The Geomagnetic Field

There is no giant bar magnet near the earth's center but earth can still be considered as a magnet as shown in figure 2. The north pole of the earth is different from the north pole of a bar magnet in sense that for the north pole of the earth the magnetic field lines go into the earth, while opposite for a bar magnet. This is because the north pole of a magnet was actually named as north-seeking pole, now in short it is north pole; which points towards the north pole of the earth.

It is known that the Earth is stratified. In radius it is composed of layers having different chemical composition and different physical properties. The magnetic field produced by the earth is due to

1) The main part of the magnetic field, which is due to the liquid iron in the outer core.
2) The small part, which is due to the permanent magnetization in the crust of the Earth.

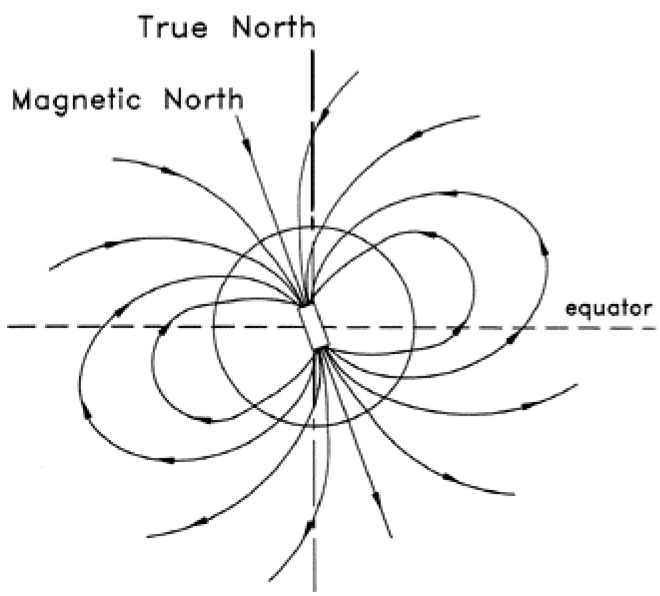

Figure 2: *Magnetic field lines of the Earth*

1.2.1 Dynamo theory of the geomagnetic field

Geophysical theory that explains the origin of the Earth's main magnetic field in terms of a self-sustaining dynamo is called dynamo theory. In this dynamo mechanism, as shown in figure 3, fluid motion in the Earth's outer core moves conducting material (liquid iron) across an already existing, weak magnetic field and generates an electric current. The electric current, in turn, produces a magnetic field that also interacts with the fluid motion to create a secondary magnetic field. Heat from radioactive decay in the core is thought to induce the convective motion in the outer core. Together, the two fields are stronger than the original and lie essentially along the axis of the Earth's rotation [].

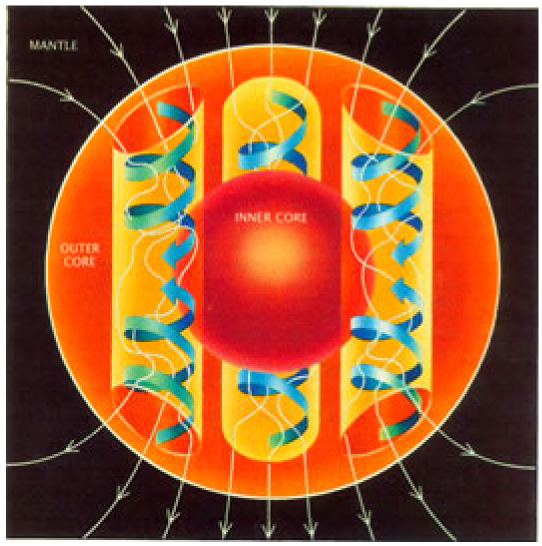

Figure 3: *Generation of the geomagnetic field lines*

1.2.2 Secular Variation

The Earth's magnetic field is slowly changing on time scales. Such changes are referred to as secular variation. The same fluid motion in the Earth's core that sustains the main part of the magnetic field also causes the field to slowly change with time. So, models and charts of the magnetic field at the Earth's surface need to be periodically updated. This variation can be seen in all vectorial parts of the magnetic field, but it was first noticed in declination several hundred years ago. On average the declination at the Earth's surface changes by about a fifth of a degree per year. [22]

1.2.3 Magnetosphere

Magnetosphere is a magnetized area around the earth as shown in figure 4. The magnetosphere shields the surface of the Earth from the charged particles of the solar wind and is generated by electric currents located in many different parts of the Earth. It is compressed on the day (Sun) side due to the force of the arriving particles, and extended on the night side.

Two factors determine the structure and behavior of the magnetosphere:

a) The Internal Field of the Earth

The internal field of the Earth (its "main field") appears to be generated in the Earth's core by a dynamo process, associated with the circulation of liquid metal in the core, driven by internal heat sources. Its major part resembles the field of a bar magnet ("dipole field") inclined by about 11° to the rotation axis of Earth.

b) The Solar Wind

The solar wind is a fast outflow of hot plasma from the sun in all directions and it is the cause of magnetic storms on the earth. Above the sun's equator it typically attains 400 km/s and above the sun's poles up to twice as much as on the equator. The flow is powered by the million-degree temperature of the sun's corona, for which no generally accepted explanation exists as yet. Its composition resembles that of the Sun i.e. about 95% of the ions are protons, about 4% helium nuclei, with 1% of heavier matter (C, N, O, Ne, Si, Mg, Fe, etc) and enough electrons to keep charge neutrality. At Earth's orbit its typical density is 6 ions/cm^3 (variable, as is the velocity), and it contains a variable interplanetary magnetic field (IMF) of (typically) 2–5 nT.

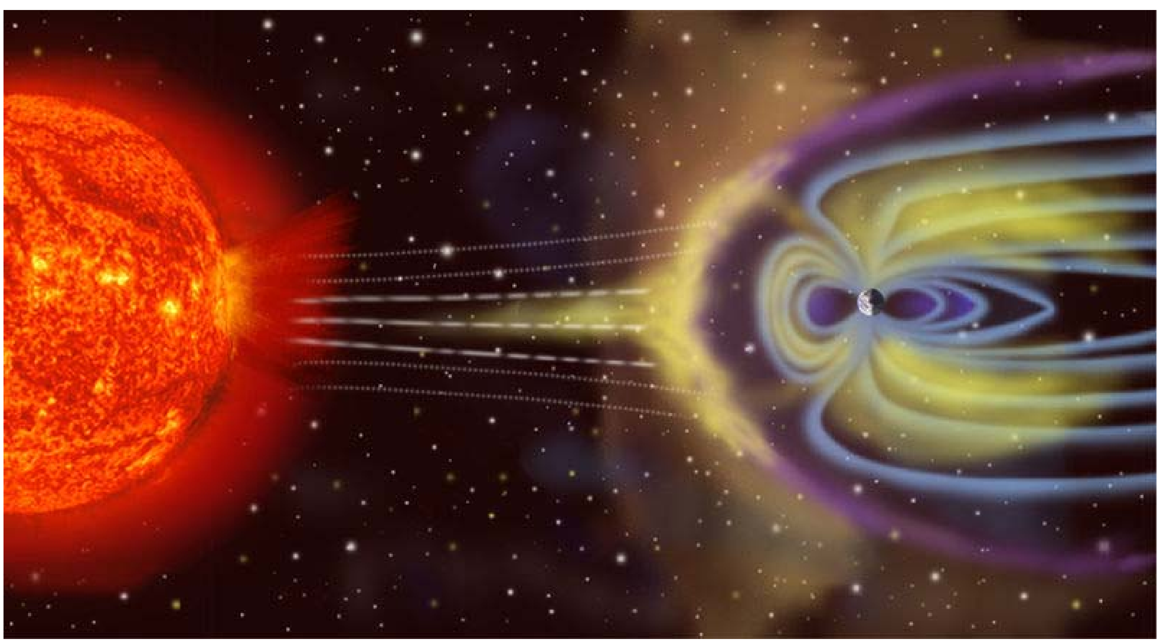

Figure 4: *Showing the magnetosphere of the earth*

1.3 World Geomagnetic Models

1.3.1 World Magnetic Model (WMM)

The World Magnetic Model (WMM) is the standard geomagnetic model of the United States Department of Defense (DoD), the Ministry of Defense (United Kingdom), the North Atlantic Treaty Organization (NATO), and the World Hydrographic Office (WHO) navigation and attitude/heading reference. It is also used widely in civilian navigation systems. The model, associated software, and documentation are distributed by the National Geophysical Data Center (NGDC) on behalf of National Geospatial-intelligence Agency (NGA). Updated model coefficients are released at 5-year intervals, with the current model expiring on December 31, 2009. [20]

1.3.2 International Geomagnetic Reference Field (IGRF)

The International Geomagnetic Reference Field (IGRF) is a standard mathematical description of the Earth's main magnetic field. It is the product of a collaborative effort between magnetic field modelers and the institutes involved in collecting and disseminating magnetic field data from satellites and from observatories and surveys around the world. An online calculator [23] is available which allows easy evaluation of the most recent (10th generation) IGRF model at any location and time between 1900 and 2010.

Mathematically, the IGRF model consists of the Gauss coefficients which define a spherical harmonic expansion of the geomagnetic potential:

$$V(r,\phi,\theta) = a\sum_{l=1}^{L}\sum_{m=0}^{l}\left(\frac{a}{r}\right)^{l+1}\left(g_l^m \cos m\phi + h_l^m \sin m\phi\right)P_l^m\left(\cos\theta\right) \tag{1}$$

where r is radial distance from the Earth's center, L is the maximum degree of the expansion, φ is East longitude, θ is co-latitude (the polar angle), a is the Earth's radius, g_l^m and h_l^m are Gauss coefficients, and $P_l^m\left(\cos\theta\right)$ are the Schmidt normalized associated Legendre’s functions of degree l and order m.

1.4 Geomagnetic Field Components

The geomagnetic field is usually resolved in either Cartesian components or Spherical components as shown in figure 5. Table 1 shows the magnetic field components generated by [25], for Islamabad, Pakistan at 500m above mean sea level.

Lat: 33°42'36" Lon: 73°3'36" Elev :500.00m	**Declination** +East -West	**Inclination** +Down -Up	**Horizontal Intensity**	**North Component** +North -South	**East Component** +East -West	**Vertical Component** +Down -Up	**Total Field**
7/26/2009	2° 8'	51° 55'	30,886.2 nT	30,864.7 nT	1152.3 nT	39,404.1 nT	50,066.4 nT
Change per year	1' per year	6' per year	-21.8 nT/year	-22.1 nT/year	7.9 nT/year	116.1nT/year	78.0 nT/year

Table 1: *Geomagnetic field components and their secular variation*

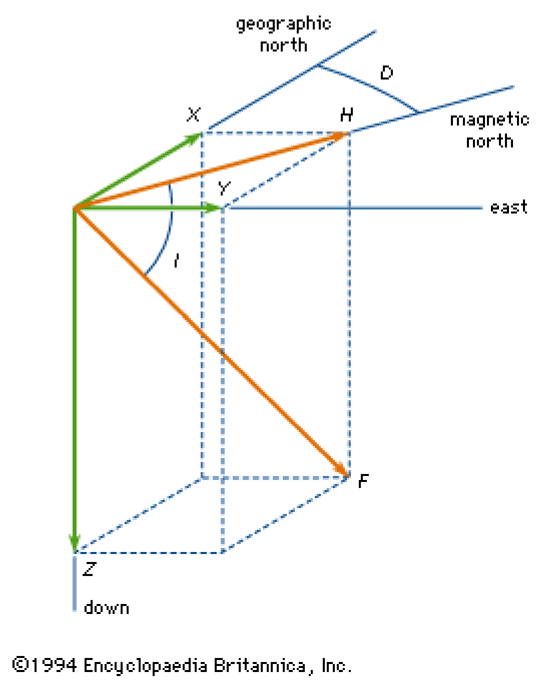

Figure 5: *Geomagnetic field components [18]*

1.4.1 Cartesian Coordinates

The X-component points towards the geographical north and Y-component points towards the east. Then the Z-component points towards the center of the earth as shown in figure 5.

The figures 6, 7 and 8 show the X, Y and Z components of the geomagnetic field of the world respectively. All these figures are drawn in MATLAB using real geomagnetic data from WMM2005. The X-axes (right side) show the longitude ranging from -180 to +180. The Y-axes show the latitude ranging from -90 to +90. The Z-axes show the geomagnetic field components. It can be seen that the Z-component can give a rough idea of latitude.

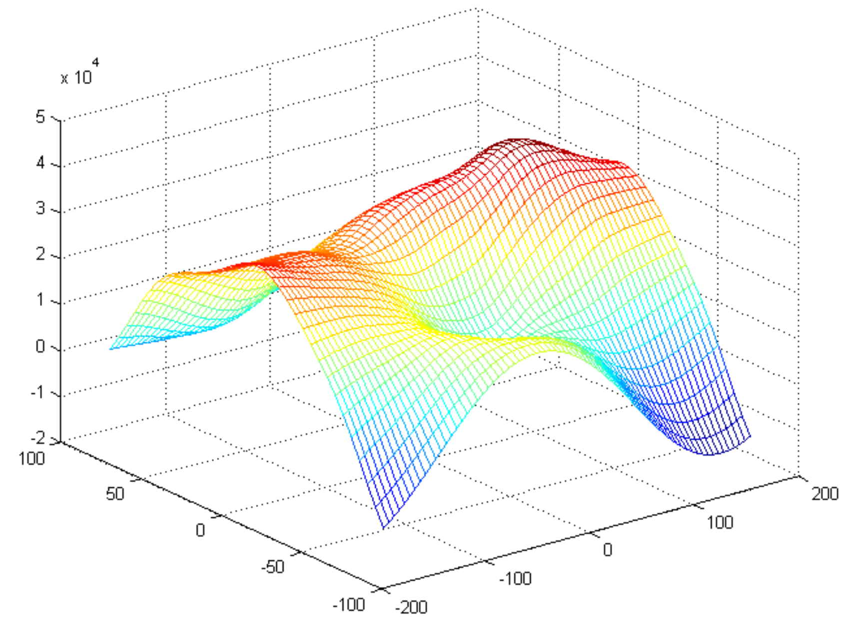

Figure 6: *3d plot of X-component of the total field of the world*

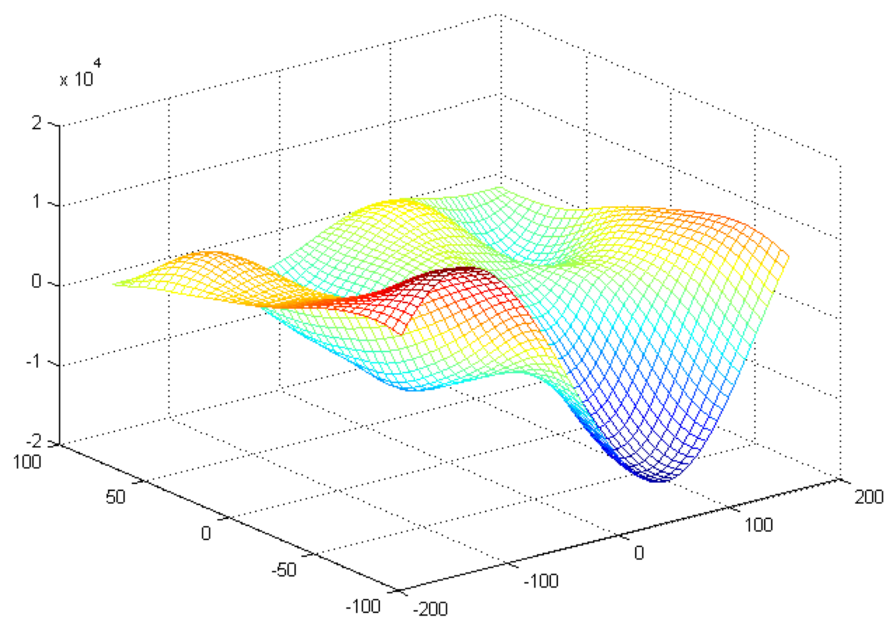

Figure 7: *3d plot of Y-component of the total field of the world*

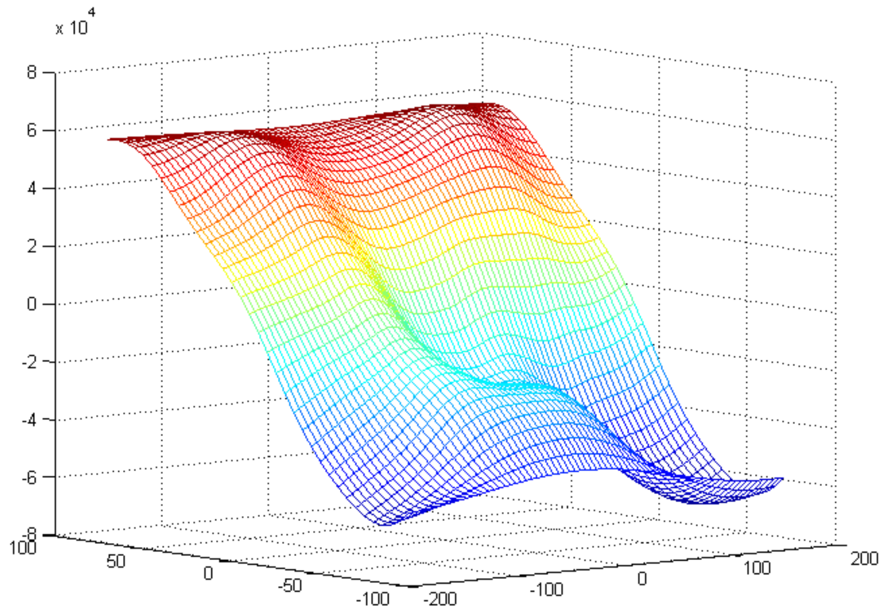

Figure 8: *3d plot of Z-component of the total field of the world*

1.4.2 Spherical Coordinates

The angle between the true North and the magnetic North is called Declination and the angle between the total intensity and the horizontal plane is called Inclination. Thus the third component is the total intensity denoted by F in figure 5.

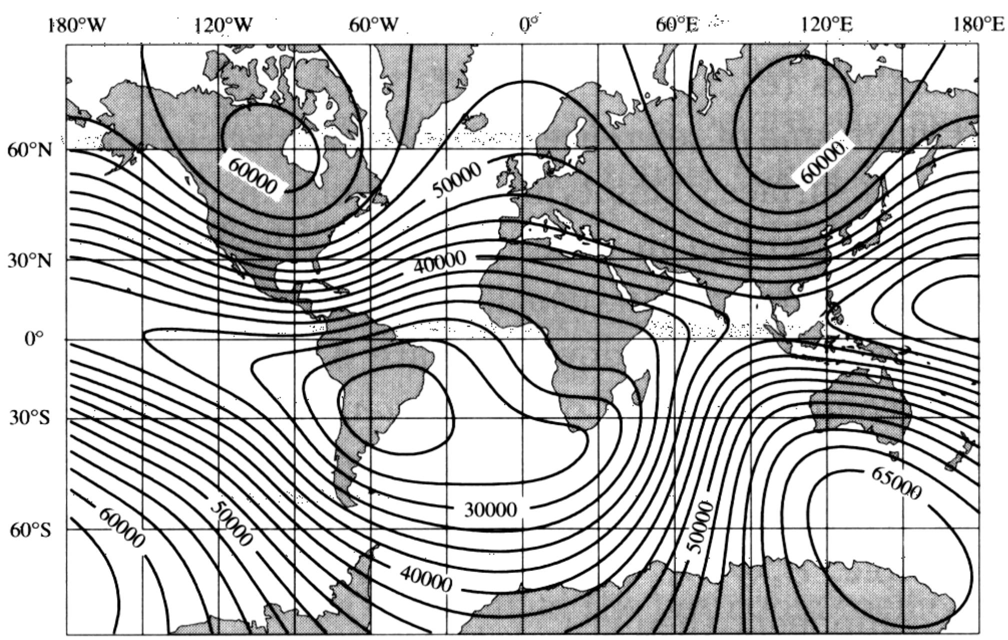

Figure 9: *Isogram of total field intensity [24]*

a) Total field intensity

The total magnetic field is the main magnetic field to which a compass points. A map joining same values is called isogram. The lines joining same geomagnetic field intensity points are called isodynamics. Figure 9 shows the world map and the isodynamics. A point having some intensity can be found at many places. It can be seen that there is no fixed pattern on which can be used for navigation.

b) Declination

The same declination lines are called isogonics and the line with zero declination is called agonic. The scale on figure 10 is the same as on figure 9 and the grey image drawn is the map of the world. It can be seen from figure 10 that an angle having some value can be found at many places on this map. So there is no fixed pattern on which can be used for localization.

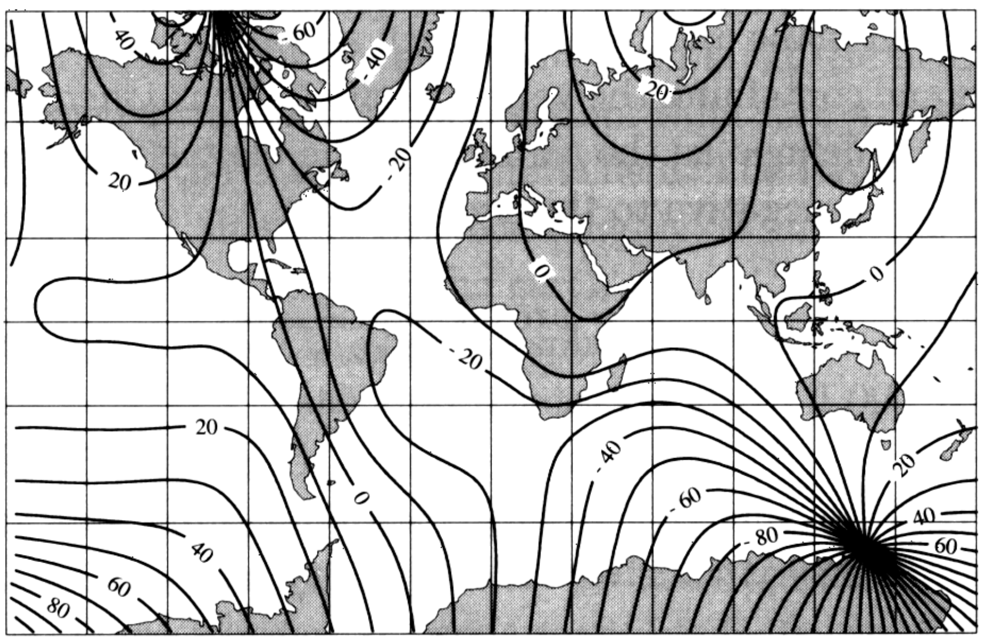

Figure 10**:** *Isogram of declination [25]*

c) Inclination

The same inclination lines are called isoclinics and the line with zero inclination is called aclinic. From the figure 11 it can be seen that the aclinic lies near the equator. As it can be seen from the figure the inclination can give a rough idea of latitude. At the North

Pole magnetic field vector goes into the earth and at the South Pole it goes out of the earth and the remaining angles are as shown in figure 11.

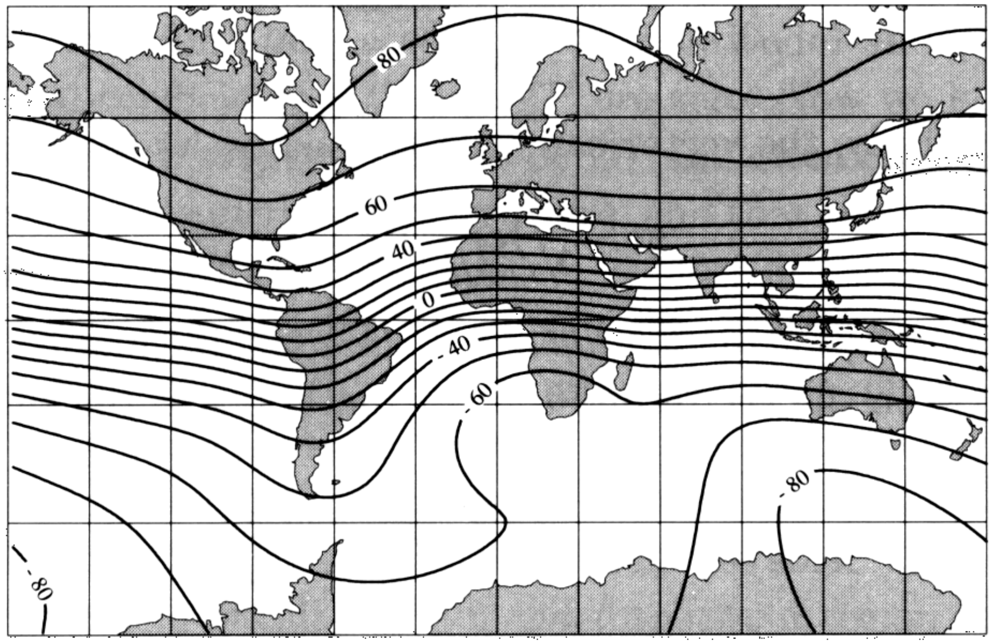

Figure 11: *Isogram of inclination [26]*

1.5 Sea Turtle Navigation

The moment hatchlings emerge from their underground nests; tiny loggerhead sea turtles face an enormous journey. They scramble to the sea and begin navigating 8,000 miles across the open ocean and then back home again. The turtles are headed to the North Atlantic gyre, a system of warm-water currents that spans the Atlantic between the eastern coast of North America and the western coast of Africa. The turtles may linger there for years, feed on seaweed, till they grow large enough to return to American feeding grounds without any fear of predators near the shore.

A turtle straying off course at the north end of the gyre could reach near Great Britain, could die due to cold water of North Atlantic Ocean. A turtle swimming south out of the gyre could be swept up in south Atlantic currents and carried away from its range.

How the turtles find their way back to the nesting ground has long been a mystery. New studies suggest that like many migrating birds and honeybees, the thumb-sized turtles are guided by very slight differences in the Earth's magnetic field. Taking subtle cues from the field, which is most intense at the poles, these turtles keep themselves within a circular swathe of warm ocean current known as the Atlantic gyre, which stretches from the eastern U.S. to the coasts of Spain and Africa. Getting out of it, the turtles would face cold waters and could die. After swimming in the gyre for years the turtles make their way home to Florida beaches. It's doubtful that the turtles find their way using visual cues since the open ocean offers few landmarks. Temperature is also an unlikely indicator since water temperatures vary due to factors like the Gulf Stream even within the Atlantic gyre.

To prove that the loggerhead turtles use magnetic fields for navigation, Lohmann and his team constructed a miniature ocean in a four-foot-wide fiberglass tank filled with seawater and surrounded it by a miniature magnetic field created by a carefully charged network of copper coils which is shown in figure 12. Then, one by one, they dressed 79 newborn turtles in tiny bathing suits and lowered them inside the tank. Attached to the Velcro/Lycra bathing suits were harnesses rigged to a fishing line. The fishing line connected to a mechanical arm that measured every small turn the turtle in the tank made. Each hatchling was tethered to a rotate-able lever-arm mounted on a digital encoder (located inside the central post of the orientation arena). The lever arm tracked the direction toward which the turtle swam; signals from the encoder were relayed to the data acquisition computer, which recorded the orientation of the turtle every 10 sec.

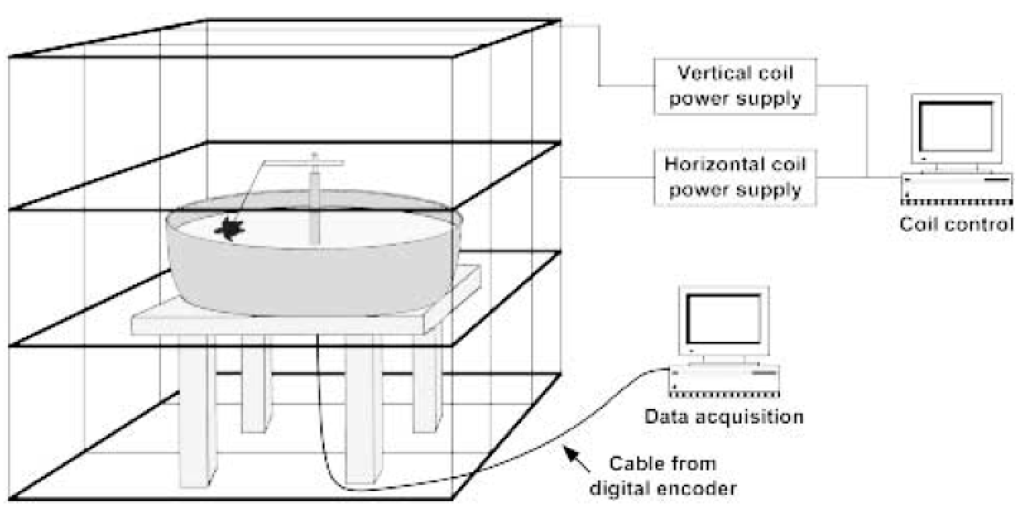

Figure 12: *Diagram of the orientation arena, magnetic coil system and data acquisition system used in studies of hatchling responses to magnetic field features*

The arena was enclosed by a magnetic coil system consisting of two different coils arranged orthogonally. One coil controlled the horizontal component of the field while the other controlled the vertical component. By manipulating the tank's magnetic field to mimic the magnetic angles and intensities that the turtles would experience at sea, Lohmann found the turtles immediately swam in a direction that would have kept them inside the warm current.

1.5.1 Detection of Magnetic Inclination Angle

The geomagnetic parameter most strongly correlated with latitude is field line inclination. To determine if loggerheads can distinguish between different inclination angles, hatchlings were tethered in a water-filled arena surrounded by a computerized coil system (Figure 12) that was used to generate earth-strength fields with different inclinations. Hatchlings exposed to a field with an inclination angle found along the northern boundary of the North Atlantic gyre swam south-southwest (Figure 13).

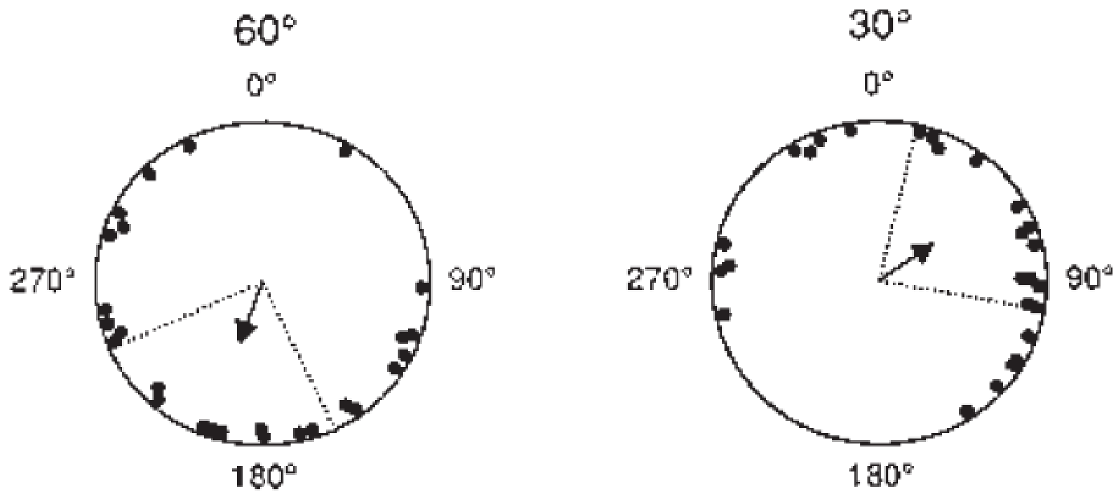

Figure 13: *Orientation of hatchlings tested in magnetic fields of same intensity but different inclinations*

Turtles exposed to a 60° inclination angle (an angle found near the northern edge of the North Atlantic gyre) were significantly oriented toward the south-southwest, whereas those exposed to an inclination angle of 30° (found near the southern border of the gyre) swam in a northeasterly direction. Dashed lines represent the 95% confidence interval for the mean bearing.

In contrast, hatchlings exposed to an inclination angle found near the southern boundary of the gyre swam in a northeasterly direction (Figure 13). Turtles exposed to inclination angles they do not normally encounter (i.e., from north or south of the North Atlantic gyre), or to a field inclination found well within the northern and southern extremes of the gyre, were not significantly oriented.

These results demonstrate that loggerheads can distinguish between different magnetic inclination angles. In addition, inclination angles found near the northern and southern gyre boundaries elicited orientation that would, in each case, direct turtles approximately toward the gyre center. The results are therefore consistent with the hypothesis that specific inclination angles in effect warn turtles that they have reached the latitudinal extremes of the gyre and must swim toward the gyre center to avoid straying out of the warm-water current system. For turtles that are safely within the gyre, drifting passively presumably poses no danger of displacement into undesirable areas. The absence of a directional preference among turtles exposed to an inclination angle found near the gyre's latitudinal center is consistent with this interpretation.

1.5.2 Detection of Magnetic Field Intensity

A second geomagnetic feature that varies across the surface of the earth is field intensity. To determine if hatchling loggerheads can perceive differences in intensity that they experience along their migratory route, hatchlings were exposed to one of two intensities (figure 14: top) that they normally encounter during their first months in the sea. The inclination angle of the field was held constant in these trials. Turtles tested in a field of 52,000 nT (a field 10.6% stronger than the natal beach field, and one that hatchlings first encounter near South and North Carolina, USA) swam eastward. Those exposed to a 43,000 nT field (a field 8.5% weaker than the natal beach field, and one first encountered on the eastern side of the Atlantic near Portugal) swam westward (Figure 14: bottom).

These results demonstrate that hatchlings can distinguish between field intensities that occur in different locations along their migratory route. Moreover, because eastward orientation near South Carolina and westward orientation near the coast of Portugal would both function to keep young turtles within the gyre, the results imply that turtles can derive positional information from geomagnetic field features.

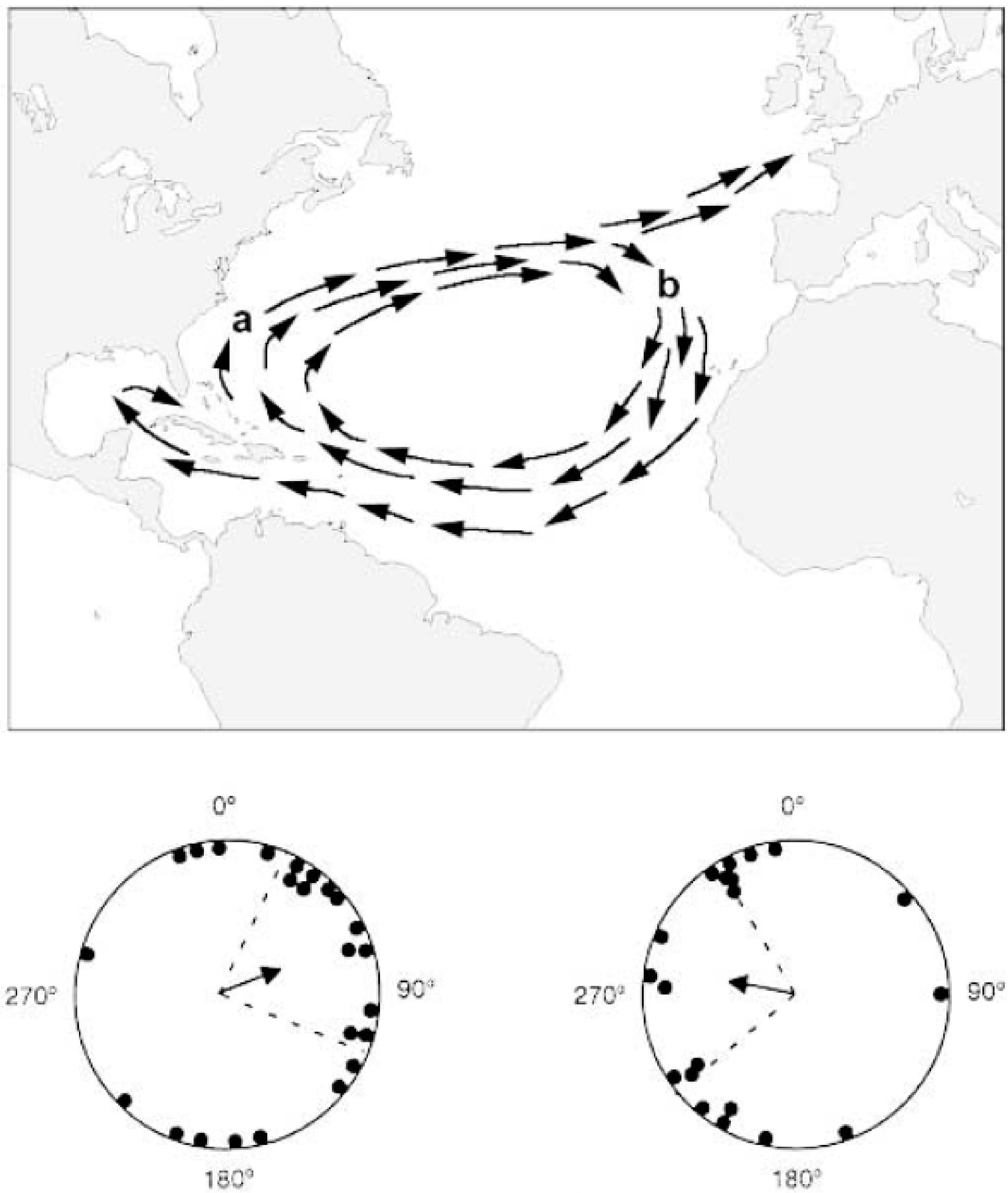

Figure 14: Top: *Generalized diagram of the North Atlantic gyre indicating the only location within the gyre where the field intensity is 52,000 nT (marked by "a") and the location where Florida loggerheads in the gyre presumably first encounter a field intensity of 43,000 nT (marked by "b")*

Bottom: *Orientation of hatchlings tested in a magnetic field of 52,000 nT (left) and 43,000 nT (right)*

1.5.3 Navigation by Adult Turtles

Adult turtles, however, can return to nesting sites by following forced displacements. Moreover, adults often follow essentially straight courses both day and night while migrating to specific destinations hundreds of kilometers away. Such precise targeting of specific destinations over immense distances is difficult to explain without hypothesizing an ability to determine geographic position relative to the goal.

Although the nature of the sea turtle position-finding system is still unknown, one hypothesis is that adult turtles use geomagnetic field features such as inclination and intensity to assess position during long-distance migrations. Geomagnetic field features could potentially be used by migrating adults in several different ways depending on the navigational task and the nature of the environment.

1.5.4 Inclination Angle Hypothesis for Adult Turtles

Many major sea turtle rookeries are located on continental coastlines that are aligned approximately north-south (e.g., Mexico, Costa Rica, the southeastern United States, and Africa). The scheme these turtles use for locating their nesting area is that they swim north or south along a coastline until the nesting location is reached and recognized. Thus, turtles might need only to detect a single feature that varies latitudinal to discriminate between different coastal regions as shown in figure 15.

In principle, the ability to detect either inclination angle or intensity could allow turtles to identify a particular area of a continental beach. Inclination angle, in particular, is strongly correlated with latitude; thus, for shorelines running approximately north-south, each beach segment is marked by a unique angle.

If turtles learn the inclination angle of their home beach as hatchlings, an adult attempting to return to the area might need only to swim along the African coast until the appropriate angle is encountered (Figure 15). Such turtles might also assess whether they are north or south of the goal by determining if the inclination angle is smaller or larger than that of the natal beach region. A similar process based on other magnetic features that vary latitudinally (e.g., total intensity, horizontal field intensity, or vertical field intensity) could also hypothetically be used [28].

Figure 15: *Shows the Isoclinics along the African coast. Each region of the eastern coastline is marked by a different inclination angle; a similar situation exists for the west coast. Adjacent isoclinics represent differences in inclination of 2°, while dark lines are spaced 10° apart. Isoclinics were derived from the IGRF 1995 model for the year 1995*

Chapter 2

BACKGROUND

Before starting the main topic of this thesis, this chapter shows an overview of some important topics. It will explain the overview of rotation in Euclidean space, normal matrices, orthogonal matrices, rotation matrices, dead reckoning and inertial navigation system.

2.1 Rotation in Euclidean Space

In mechanics and geometry, the rotation group is the group of all rotations about the origin of three-dimensional Euclidean space R^3 under the operation of composition. By definition, a rotation about the origin is a linear transformation that preserves length of vectors (it is an isometry) and preserves orientation (i.e. handedness) of space.

Composing two rotations results in another rotation; every rotation has a unique inverse rotation. Owing to the above properties, the set of all rotations is a group under composition. [10]

2.2.1 Length and Angle

Besides just preserving length, rotations also preserve the angles between vectors. This follows from the fact that the standard dot product between two vectors u and v can be written purely in terms of length:

$$u.v = \tfrac{1}{2}\left(\|u+v\|^2 - \|u\|^2 - \|v\|^2\right) \qquad (2)$$

It follows that any length-preserving transformation in R^3 preserves the dot product, and thus the angle between vectors. Rotations are often defined as linear transformations that preserve the inner product on R^3. This is equivalent to requiring them to preserve length.

2.2 Normal Matrices

A real square matrix A is a normal matrix if

$$A^T A = AA^T \qquad (3)$$

Where, A^T is the transpose of A. That is, a matrix is normal if it commutes with its transpose [14].

2.3 Orthogonal Matrices

In linear algebra, an orthogonal matrix is a square matrix with real entries whose columns (or rows) are orthogonal unit vectors (i.e. orthonormal).
Equivalently, a matrix Q is orthogonal if its transpose is equal to its inverse:

$$Q^T Q = QQ^T = I \tag{4}$$

Alternatively,

$$Q^T = Q^{-1} \tag{5}$$

The determinant of any orthogonal matrix is +1 or −1. This follows from basic facts about determinants, as follows:

$$1 = |I| = |D^T D| = |D^T||D| = |D|^2 \tag{6}$$

The converse is not true; having a determinant of +1 is no guarantee of orthogonality, even with orthogonal columns, as shown by the following counterexample. [11]

$$\begin{bmatrix} 2 & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$$

2.4 Rotation Matrices

In mathematics, the orthonormal basis for a Euclidean space is that they should be orthogonal to each other and their magnitude should be equal to unity. And the standard basis for a Euclidean space consists of one unit vector pointing in the direction of each axis of the Cartesian coordinate system [15]. For example, the standard bases for three-dimensional space are the vectors:

$$e_x = (1,0,0), e_y = (0,1,0), e_z = (0,0,1)$$

There is a subgroup E+(n) of the direct isometries, i.e., isometries preserving orientation, also called rigid motions; they are the rigid body moves. These include the translations, and the rotations, which together generate E+(n). [13]

Every rotation maps an orthonormal basis of R^3 to another orthonormal basis. Like any linear transformation, a rotation can always be represented by a matrix. Let R be a given rotation. With respect to the standard basis (e_1,e_2,e_3) of R^3 the columns of R are given by (Re_1,Re_2,Re_3). Since the standard basis is orthonormal, the columns of R form another orthonormal basis. This orthonormality condition can be expressed in the form:

$$R^T R = I \tag{7}$$

Where R^T denotes the transpose of R and I is the 3 × 3 identity matrix. Matrices for which this property holds are called orthogonal matrices. The group of all 3 × 3 orthogonal matrices is denoted O(3), and consists of all proper and improper rotations.

In addition to preserving length, proper rotations must also preserve orientation. A matrix will preserve or reverse orientation according to whether the determinant of the matrix is positive or negative. For an orthogonal matrix R, note that

$$\left|R^T\right| = \left|R\right| \rightarrow \left|R\right|^2 = 1 \tag{8}$$

so that

$$\left|R\right| = \pm 1 \tag{9}$$

The subgroup of orthogonal matrices with determinant +1 is called the special orthogonal group, denoted SO(3) as shown in figure 16.

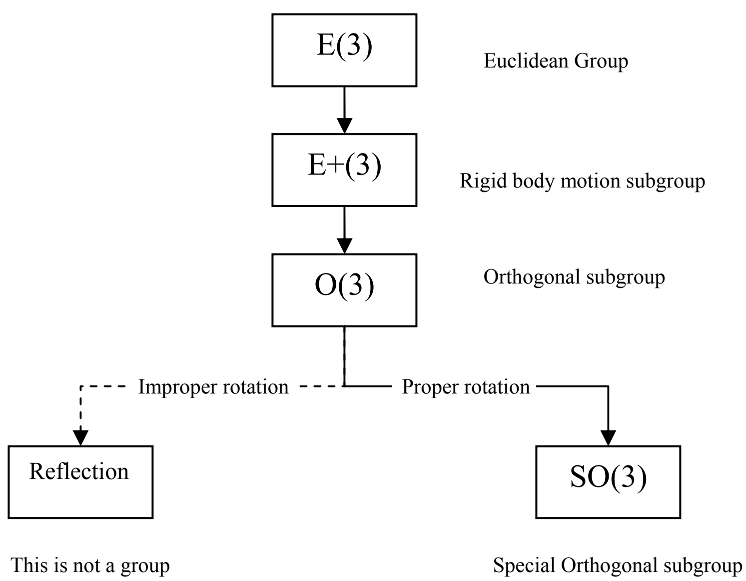

Figure 16: *Block diagram of the classification of Euclidean space*

The rotation group SO(3) can be described as a subgroup of E+(3), the Euclidean group of direct isometries of R^3. This larger group is the group of all motions of a rigid body. Each of these is a combination of a rotation about an arbitrary axis and a translation along the axis, or put differently, a combination of an element of SO(3) and an arbitrary translation.

Thus every rotation can be represented uniquely by an orthogonal matrix with unit determinant. Moreover, since composition of rotations corresponds to matrix multiplication, the rotation group is isomorphic to the special orthogonal group SO(3).

Improper rotations correspond to orthogonal matrices with determinant −1, and they do not form a group because the product of two improper rotations is a proper rotation.

The rotation group generalizes quite naturally to n-dimensional Euclidean space, R^n. The group of all proper and improper rotations in n dimensions is called the orthogonal group, O(n), and the subgroup of proper rotations is called the special orthogonal group, SO(n).

2.5 Dead Reckoning

It is a process of estimating the current position based upon previously determined position and speed. It is considered to be used in old navigation systems. Nowadays, Inertial Navigation System which is based on DR (Dead Reckoning), is used in modern navigation systems [27].

2.6 Inertial Navigation System

It is a navigation aid that uses computers and motion sensors (accelerometers, gyroscopes) to continuously calculate the position, orientation, and velocity of a moving object without the need for external references.

The INS is initially provided with its position and velocity from another source (a human operator, a GPS satellite receiver, etc.), and thereafter computes its own updated position and velocity by integrating information received from the motion sensors. The advantage of an INS is that it requires no external references in order to determine its position, orientation, or velocity once it has been initialized. Figure 17 shows an INS module connected to a laptop.

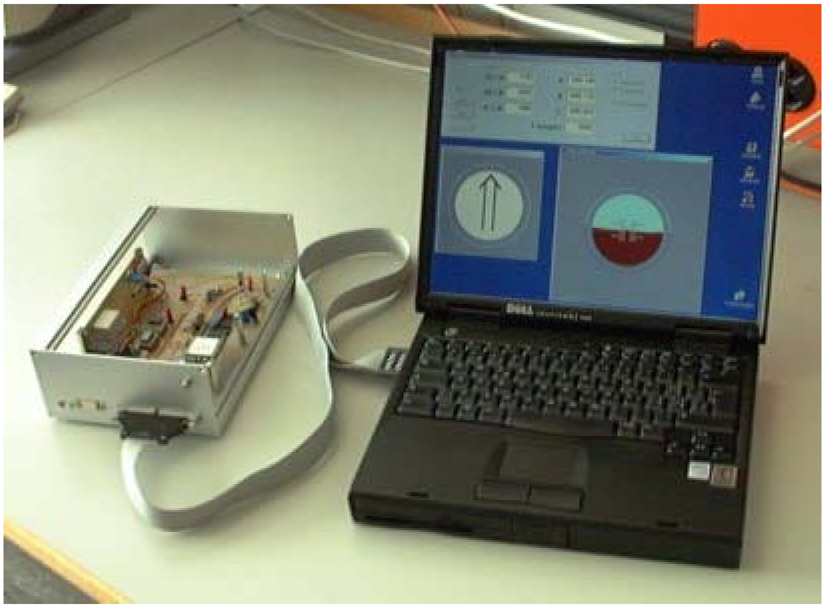

Figure 17: *An INS module*

An INS can detect a change in its geographic position (a movement east or north, for example), a change in its velocity (speed and direction of movement), and a change in its orientation (rotation about an axis). It does this by measuring the linear and angular accelerations applied to the system. Since it requires no external reference (after initialization), it is immune to jamming and deception.

2.6.1 Accelerometers

Accelerometers measure the linear acceleration of the system in the inertial reference frame, but these can not measure their own orientation. Its construction can be thought of a mass attached to a spring as shown in figure 18, where m is the mass, k is the spring, B is the damping and x is the direction in which acceleration is to be measured.

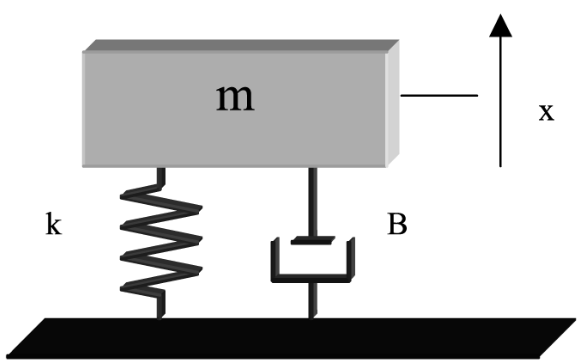

Figure 18: *Schematic of an accelerometer*

This can be thought of as the ability of a blindfolded passenger in a car to feel himself pressed back into his seat as the vehicle accelerates forward or pulled forward as it slows down; and feel himself pressed down into his seat as the vehicle accelerates up a hill or rise up out of his seat as the car passes over the crest of a hill and begins to descend. Based on this information alone, he knows how the vehicle is moving relative to itself, that is, whether it is going forward, backward, left, right, up (toward the car's ceiling), or down (toward the car's floor) measured relative to the car, but not the direction relative to the Earth, since he did not know what direction the car was facing relative to the Earth when he felt the accelerations.

2.6.2 Gyroscopes

Gyroscopes measure the angular velocity of the system in the inertial reference frame. By using the original orientation of the system in the inertial reference frame as the initial condition and integrating the angular velocity, the system's current orientation is known at all times. Figure 19 shows a simple gyroscope.

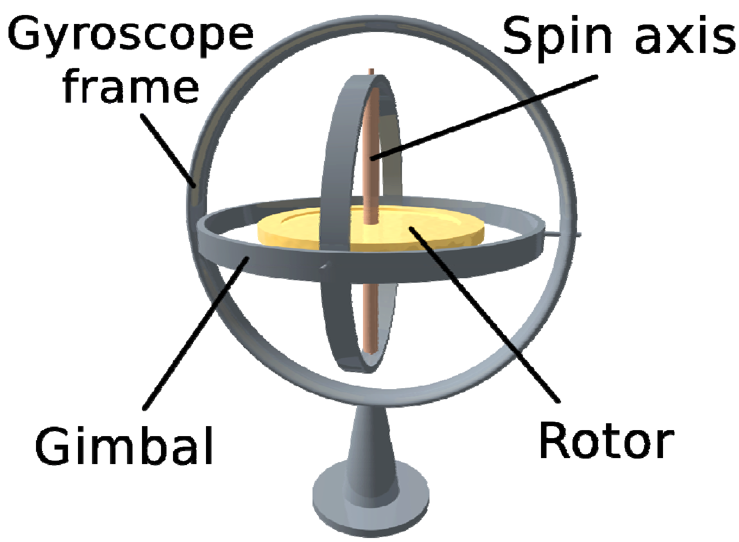

Figure 19: *Clipart of a gyroscope*

This can be thought of as the ability of a blindfolded passenger in a car to feel the car turn left and right or tilt up and down as the car ascends or descends hills. Based on this information alone, he knows what direction the car is facing but not how fast or slow it is moving, or whether it is sliding sideways.

2.6.3 Combined Effect

However, by tracking both the current angular velocity of the system and the current linear acceleration of the system measured relative to the moving system, it is possible to determine the linear acceleration of the system in the inertial reference frame. Performing integration on the inertial accelerations (using the original velocity as the initial conditions) using the correct kinematics equations yields the inertial velocities of the system, and integration again (using the original position as the initial condition) yields the inertial position.

In our example, if the blindfolded passenger knew how the car was pointed and what its velocity was before he was blindfolded, and he is able to keep track of both how the car has turned and how it has accelerated and decelerated since, he can accurately know the current orientation, position, and velocity of the car at any time.

2.6.4 Errors in INS

All inertial navigation systems suffer from "integration drift": small errors in the measurement of acceleration and angular velocity are integrated into progressively larger errors in velocity, which is compounded into still greater errors in position. This is a problem that is inherent in every open loop control system. Since the new position is calculated solely from the previous position, these errors are cumulative, increasing at a rate roughly proportional to the time since the initial position was input. Therefore the position fix must be periodically corrected by input from some other type of navigation system. So, an inertial navigation is usually used to supplement other navigation systems, providing a higher degree of accuracy than is possible with the use of any single system.

Control theory in general and Kalman filtering in particular provide a theoretical framework for combining information from various sensors. One of the most common alternative sensors is a satellite navigation radio, such as GPS. By properly combining the information from an INS and the GPS system, the errors in position and velocity are stable. Furthermore, INS can be used as a short-term fallback while other methods are unavailable, for example when a vehicle passes through a tunnel or diving underwater [21].

Chapter 3

GEOMAGNETIC LOCALIZATION

Localization means to determine the position of an object. For this purpose different techniques can be used. But it always needs some reference with which it can be synchronized. GPS is used as a reference for a vast number of projects. When dealing with AUV's, there is no such reference. Geomagnetic field can be taken as a reference for underwater localization.

Geomagnetic map is built using the values from the world magnetic models. Now, using the readings from the magnetic sensors localization has to be done. Following are the techniques which are devised for the solution of this problem.

3.1 Single Point Matching Localization

The easiest way of carrying out geomagnetic matching is to sense the value of the geomagnetic field and then look for that field in a given geomagnetic map. Cartesian components of this field from magnetic sensors known as 3-axis magnetometer can be obtained. On the other hand, the geomagnetic map of our desired resolution from the world known standards can be obtained.

In other words, using the readings of a specific location, search can be done on a given map and finally the specific location is obtained. Every geomagnetic vector can be resolved into three components in Cartesian coordinates. These components can be obtained from the magnetometer and then these can be searched in the map data.

But searching consumes a lot of memory and computational cost, so it can not be used directly for real time operations. If first component is searched first, then the second component is searched corresponding to that first one and the third accordingly, then the speed of the search increases. Another optimization is that all the map data is loaded into the RAM before starting any operation instead of hard disk; which is slow. In addition to these, different search optimization algorithms such as group search, binary search etc. can be used to increase the speed of searching.

Here first all data is loaded into the RAM and group search technique is implemented and results are shown in chapter 6. It is a localization technique which uses all components of a single location as an input and can be called single point matching localization technique.

3.2 Path Matching Localization

Two or more locations on the earth can have the same field values. At that location, the single point matching technique loses its efficiency. Therefore, when more than two samplings are used then it can be known as path matching algorithm.

Geomagnetic field can be used to rectify INS errors. Path matching geomagnetic localization can be regarded as a kind of localization using a map, which is finding the position of AUV (Autonomous Underwater Vehicle) on an existing geomagnetic map from onboard magnetic field sensor measurements. To avoid self induced magnetic influence, sensors must be placed far from the body.

Image processing's traditional iterative closest point (ICP) algorithm can be used for matching. ICP was proposed as a method for image registration [8]. This algorithm is an iterative form of the Variational algorithm [9]. The algorithm works by iterating the procedure of rigid transform – closest points' determination – rigid transform, which tries to decrease the error upon iteration. Usually geomagnetic maps are stored in grid form and this is similar to image pixels. Thus it is used in underwater geomagnetic localization, to make the trajectory indicated by INS gradually revert to the actual one.

Therefore this thesis emphasizes on the needed modification of ICP according to geomagnetic field. Its feasibility for underwater geomagnetic localization is analyzed and verified by simulation experiments.

3.2.1 Methodology

The scenario for an AUV using the geomagnetic field to be located is given as: The AUV starts from a position and travels along a path. This path is measured by INS or DR (dead reckoning) and it is given in the form of a set of 2D location points. Let's denote the measured path by [$D_j, j = 1,2,...,N$], where N is the total number of points in the path. Due to errors and drifts in instruments and random external effects the measured path will be different from the actual path. Let's denote the actual path by [A_j, $j = 1, 2, ..., N$], respectively.

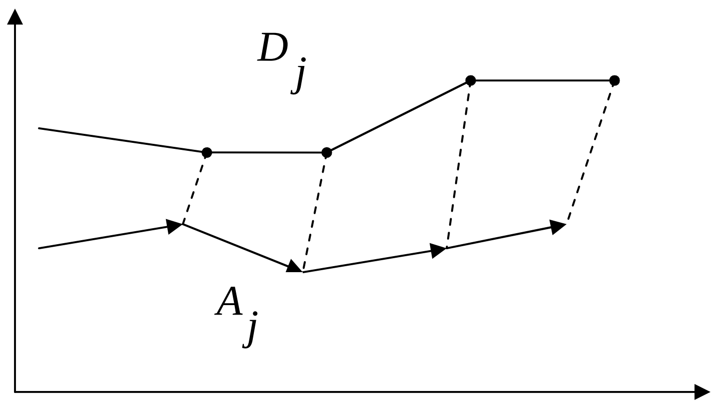

Figure 20: Illustration of Measured and Actual path

The actual path is unknown. Let's assume that the actual path is known. Now, given two point sets D_j and A_j as shown in Figure 20, there should be a transformation T , such that:

$$TD_j = A_j, j = 1, 2, ..., N \tag{10}$$

Generally speaking, the transformation T satisfying (1) does not exist, because this is an over-determined system. Usually, the problem is formulated as a least-squares problem, with objective function given by:

$$\varepsilon = \frac{1}{N}\sum_{j=1}^{N}\left\|A_j - TD_j\right\|^2 \tag{11}$$

The transformation between two point sets can be thought of as the result of a rigid-body motion, that is transformation T, SE(3), and can thus be decomposed into a rotation R set of SO(3) and a translation t set of R^3.

$$T = \left[\begin{array}{ccc|c} & R & & t \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \tag{12}$$

In the following section, the method to solve for R and t is discussed.

3.2.2 Positioning Algorithm based on ICP Algorithm

The proposed positioning algorithm based on ICP is executed after a vehicle collects a sequence of geomagnetic samplings in the matching area. The algorithm performs the iterative procedure of closest points' determination and rigid transform. The approximation is improved step by step. The theme of the algorithm is to match the path to a predefined map say M_j, to correct the errors in INS and finally regress to the actual path.

An assumption is made here that the path D_j is close to the path A_j because ICP has a drawback of matching distant cloud of points. The steps for this modified algorithm are as follows:

1) As the map and the path data is in 2D, so respective geomagnetic intensity is taken as the 3rd Dimension for the map and the path.
2) Find the closest points to the path D_j, in the map M_j and call it C_j.
3) Apply Variational algorithm to find the rotation matrix R between D_j and C_j.
4) Find the difference between the mean of D_j and C_j to find the translation t.
5) Apply the rotation and translation to find the transformed point set D'_j
6) Use this D'_j as D_j and repeat from step 2.
7) When the rate of change of transformation matrix T is less then a threshold τ, then terminate the iteration, otherwise continue.
8) At last, the measured path converges to the actual path.
9) To find the overall transformation matrix, the transformation matrices in all iterations are multiplied with each other to give the final transformation matrix.

3.2.3 Positioning Algorithm based on Menq's Algorithm

Unlike the ICP algorithm, this algorithm does not use any type of iteration and it doesn't perform rigid transformation. Detailed explanation is given in chapter 5, while the steps are written below:

1) As the map and the path data is in 2D, so respective geomagnetic intensity is taken as the 3rd Dimension for the map and the path.

2) Find the closest points to the path D_j, in the map M_j and call it C_j.

3) The following equation is used to find the transformation matrix T.

$$T = (AD^T)(DD^T)^{-1} \tag{13}$$

4) Now this transformation matrix T, when applied to D best fits the path to the geomagnetic map.

Chapter 4

ICP ALGORITHM

This chapter explains ICP algorithm in detail. The ICP algorithm was derived from the Varitional algorithm, afterwards two types of this algorithm is described. In the end of this chapter the methods for closest point determination and singular value decomposition are explained; which are used in this algorithm.

4.1 Variational Algorithm

According to the algorithm [9] statement:

Let

$$W = \sum_{j=1}^{n} c_j d_j^T \tag{14}$$

and write the singular value decomposition (SVD) of W as

$$W = U\Sigma V^T \tag{15}$$

$$W = U\begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} V^T \tag{16}$$

where $U,V \in \mathbb{R}^{3\times3}$ are unitary and $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq 0$ are the singular values of W. W can be said as a fake rotation matrix, because it is neither orthogonal nor belongs to the subgroup SO(3).

As it is known that U and V are orthogonal, so Σ can be dropped:

$$\begin{aligned} R &= UV^T \\ t &= \bar{c} - R\bar{d} \end{aligned} \tag{17}$$

4.2 Types of ICP Algorithm

Following are the two approaches for ICP algorithm:

4.2.1 Using Unit Quaternion

The formula stated in (13) can also be formulated in terms of unit quaternions, where the optimal solution $q_R = (q_0, q_1, q_{2,} q_3)^T$ is the unit eigenvector corresponding to the maximum eigenvalue of the matrix

$$Q(W) = \begin{bmatrix} tr(W) & b^T \\ b & W + W^T - tr(W)I_3 \end{bmatrix} \tag{18}$$

where

$$\hat{b} = W - W^T \tag{19}$$

is a skew-symmetric matrix and tr() stands for the trace operator. The unit eigenvector of the matrix Q(W) is found and the solution is set as:

$$R^k = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 + q_0 q_3) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & q_0^2 + q_2^2 - q_1^2 - q_3^2 & 2(q_2 q_3 + q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_3 - q_0 q_1) & q_0^2 + q_3^2 - q_1^2 - q_2^2 \end{bmatrix} \tag{20}$$

$$t^k = \overline{y} - R^k \overline{x}^k \tag{21}$$

4.2.2 Using SVD

The quaternion based algorithm is preferred over the SVD approach in 2 and 3 dimensions because reflections are not desired. The SVD approach, based on the cross covariance matrix of two point distributions, does, however, generalizes easily to n dimensions and would be our method of choice. [8]

Using (13) to calculate W, for the kth iteration:

$$W^k = C^k D^{Tk} \tag{22}$$

Now SVD for W is calculated

$$W^k = U^k \Sigma^k V^{Tk} \tag{23}$$

Dropping the non-orthogonal matrix and making it orthogonal:

$$R^k = U^k V^{Tk} \tag{24}$$

Computing the translation

$$t^k = \overline{c}^k - R^k \overline{d}^k \tag{25}$$

This iteration is done till

$$\left\| T^k - T^{k-1} \right\| < \tau \tag{26}$$

comes into existence and optimal path is obtained.

4.3 Closest Point Determination

To find the closest point, the Euclidean distance from every point of the data to every point on the map is taken and the minimum distance point is found, on the map to that point. Every point of the data has a corresponding closest point. The Euclidean distance is calculated by:

Euclidean distance between two points d and m is shown below:

$$C(d,m) = \|d - m\| = \sqrt{(d_x - m_x)^2 + (d_y - m_y)^2 + (d_z - m_z)^2} \tag{27}$$

Euclidean distance between a point and a set of points is given below:

$$C(d,M_j) = \min_{j\in\{1,2,\ldots,N\}} C(d,m_j) \tag{28}$$

In this case the Euclidean distance from every point in the set D_j to the set M_j is found and name it as C_j [8]. Following shows the code of closest point algorithm implemented in this thesis:

Closest Point Algorithm Code in MATLAB

```
function E=closest(Q,D)

for i=1:size(D,2)
    for j= 1:size(Q,2)

        N(j,i)=sqrt((D(1,i)-Q(1,j))^2+(D(2,i)-Q(2,j))^2+(D(3,i)-
Q(3,j))^2);

        if j==1
            temp=N(j,i);
            temp2=Q(:,j);
        end

        if temp > N(j,i)
            temp=N(j,i);
            temp2=Q(:,j);
        end
    end
    S(1,i)=temp;
    E(:,i)=temp2;
    clear temp temp2

end
```

4.4 Singular Value Decomposition

In linear algebra, the singular value decomposition (SVD) is an important factorization of a rectangular real or complex matrix, with many applications in signal processing and statistics. Applications which employ the SVD include computing the pseudo-inverse, least squares fitting of data, matrix approximation, and determining the rank, range and null space of a matrix.

Suppose A is an m-by-n matrix whose entries come from the field K, which is either the field of real numbers or the field of complex numbers. Then there exists a factorization of the form:

$$A = U\Sigma V^* \tag{29}$$

Where, U is an m-by-m unitary matrix over K, the matrix Σ is m-by-n diagonal matrix with nonnegative real numbers on the diagonal, and V* denotes the conjugate transpose of V, an n-by-n unitary matrix over K. Such a factorization is called a singular-value decomposition of A.

A common convention is to order the diagonal entries Σ(i,i) in non-increasing fashion. In this case, the diagonal matrix Σ is uniquely determined by A (though the matrices U and V are not). The diagonal entries of Σ are known as the singular values of A.

In $A = U\Sigma V^*$ the columns of V form a set of orthonormal "input" or "analysing" basis vector directions for A. The columns of U form a set of orthonormal "output" basis vector directions for A. The matrix Σ contains the singular values, which can be thought of as scalar "gain controls" by which each corresponding input is multiplied to give a corresponding output. [12]

Singular value decomposition takes a rectangular matrix of gene expression data (defined as A, where A is a $n \times p$ matrix) in which the n rows represents the genes, and the p columns represents the experimental conditions. Therefore:

$$U^T U = I_{n\times n} \tag{30}$$

$$V^T V = I_{p\times p} \tag{31}$$

Where the columns of U are the left singular vectors (gene coefficient vectors); Σ (the same dimensions as A) has singular values and is diagonal (mode amplitudes); and

V^T has rows that are the right singular vectors (expression level vectors). The SVD represents an expansion of the original data in a coordinate system where the covariance matrix is diagonal.

4.4.1 Calculating SVD

Calculating the SVD consists of finding the eigenvalues and eigenvectors of AA^T and A^TA. The eigenvectors of A^TA make up the columns of V; the eigenvectors of AA^T make up the columns of U. Also, the singular values in Σ are square roots of eigenvalues from AA^T or A^TA. The singular values are the diagonal entries of the Σ matrix and are arranged in descending order. The singular values are always real numbers. If the matrix A is a real matrix, then U and V are also real. [16]

To understand how to solve for SVD, let's take the example of the matrix:

$$A = \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

In this example the matrix is a 4x2 matrix.

4.4.2 Calculating Eigenvalues and Eigenvectors

It is known that for an n x n matrix W, then a nonzero vector x is the eigenvector of W if:

$$W\,x = \lambda\,x \qquad (32)$$

for some scalar λ. Then the scalar λ is called an eigenvalue of A, and x is said to be an eigenvector of A corresponding to λ.

So to find the eigenvalues of the above entity compute the matrices AA^T and A^TA are computed. As previously stated, the eigenvectors of AA^T make up the columns of U so the following analysis can be done to find U.

$$AA^T = \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 1 & 0 & 0 \\ 4 & 3 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 20 & 14 & 0 & 0 \\ 14 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = W$$

Now it is an n x n matrix and the eigenvalues of the matrix W can be determined. Since W x = λ x then (W- λI) x = 0

$$\begin{bmatrix} 20-\lambda & 14 & 0 & 0 \\ 14 & 10-\lambda & 0 & 0 \\ 0 & 0 & -\lambda & 0 \\ 0 & 0 & 0 & -\lambda \end{bmatrix} x = (W - \lambda I)x = 0$$

For a unique set of eigenvalues the determinant of the matrix (W-λI) must be equal to zero. Thus from the solution of the characteristic equation, |W-λI|=0 it is obtained:

$$\lambda=0,\ \lambda=0;\ \lambda = 0.1339;\ \lambda = 29.8661$$

(Four eigenvalues since it is a fourth degree polynomial). This value can be used to determine the eigenvector that can be placed in the columns of U. Thus the following equations can be obtained:

$$19.883\, x_1 + 14\, x_2 = 0$$
$$14\, x_1 + 9.883\, x_2 = 0$$
$$x_3 = 0$$
$$x_4 = 0$$

Upon simplifying the first two equations a ratio which relates the value of x_1 to x_2 is obtained. The values of x_1 and x_2 are chosen such that the elements of the Σ are the square roots of the eigenvalues. Thus a solution that satisfies the above equation x_1 = -0.58 and x_2 = 0.82 and $x_3 = x_4 = 0$ (this is the second column of the U matrix).

Substituting the other eigenvalues:

$$-9.883\, x_1 + 14\, x_2 = 0$$
$$14\, x_1 - 19.883\, x_2 = 0$$
$$x_3 = 0$$

$$x_4 = 0$$

Combining these the following is obtained:

$$U = \begin{bmatrix} 0.82 & -0.58 & 0 & 0 \\ 0.58 & 0.82 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Similarly A^TA makes up the columns of V so a similar analysis to find the value of V can be done.

$$A^T A = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 4 & 3 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

and similarly the following expression is obtained:

$$V = \begin{bmatrix} 0.40 & -0.91 \\ 0.91 & 0.4 \end{bmatrix}$$

Finally as mentioned previously the Σ is the square root of the eigenvalues from AA^T or A^TA and can be obtained directly giving us:

$$\Sigma = \begin{bmatrix} 5.47 & 0 \\ 0 & 0.37 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

So U, V and Σ are the solution of the given example

Chapter 5

MODIFIED MENQ'S ALGORITHM

This chapter explains the details of Modified Menq's algorithm. First of all over determined system is explained. Then, linear least squares problem and its solution is discussed. In the last, the original and modified Menq's algorithm is discussed.

5.1 Over Determined System

It is a system of linear equations in which the number of known in greater than the number of unknowns. In other words it can be said that the number of equations exceed the number of variables which is the case here. So this type of system does not have an exact solution. One way of solving this system is to formulate it as least squares problem; described next.

5.2 Linear Least Squares Problem

The problem of finding an approximate solution to an over determined system of linear equation; the function need not to be linear. The sum of squared residuals should be a minimum.

$$SSE = \sum_{i=1}^{N} \xi_i \tag{33}$$

To find the minimum value, it is differentiated and equated it to zero. Then the curve which best fits the set of points is gotten. It can be written as:

$$\sum_{i=1}^{M} \left| \sum_{j=1}^{N} X_{ij}\beta_j - Y_j \right|^2 \longrightarrow \min \tag{34}$$

5.3 Menq's Algorithm

The original Menq's algorithm [29] is described below.

First, the mean values are calculated for the path and closest points respectively:

$$\overline{D} = \frac{1}{N} \sum_{j=0}^{N} D_j \tag{35}$$

$$\overline{C} = \frac{1}{N}\sum_{i=0}^{N} C_i \tag{36}$$

And this is subtracted from every point of path and closest points

$$\tilde{D}_j = D_j - \overline{D} \tag{37}$$

$$\tilde{C}_j = C_j - \overline{C} \tag{38}$$

Now D and C are the sets to be used in the remaining algorithm

$$D = \left[\tilde{D}_1, \tilde{D}_2, \ldots, \tilde{D}_N\right] \tag{39}$$

$$C = \left[\tilde{C}_1, \tilde{C}_2, \ldots, \tilde{C}_N\right] \tag{40}$$

The rotated D (D_{rot}) can be found as:

$$RD = D_{rot} \tag{41}$$

Finally, the rotation matrix is obtained by using linear least squares

$$R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} = \left(CD^T\right)\left(DD^T\right)^{-1} \tag{42}$$

and the translation update is computed by:

$$t = \overline{C} - R\overline{D} \tag{43}$$

5.4 Modified Menq's Algorithm

The Menq's algorithm computes the rotation matrix and the translation matrix in two different steps. Both steps can be submerged and it can be written:

$$RD + t = C \tag{44}$$

The equation (44) is in matrix form and it can not be solved directly. So it is written in the following form to get it solved:

$$TD = C \tag{45}$$

where D and C are the normalized matrices and T is the transformation matrix.

Now the problem arises that the matrices D and C are rectangular and the inverse of a rectangular matrix does not exist. This is the problem of overdetermined system; where the number of equations is greater than the number of unknowns. Linear least squares is used to solve this problem. According to it, both sides of equation (45) are multiplied by D^T:

$$T_{4\times4}D_{4\times N}D_{N\times4}^{T}=A_{4\times N}D_{N\times4}^{T} \tag{46}$$

Finally,

$$T_{4\times4}=(CD^{T})_{4\times4}(DD^{T})_{4\times4}^{-1} \tag{47}$$

Now the inverse is possible and the approximate solution is obtained which best fits the closest points.

$$T=\begin{bmatrix} r_1 & r_2 & r_3 & t_1 \\ r_4 & r_5 & r_6 & t_2 \\ r_7 & r_8 & r_9 & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}=\left(CD^{T}\right)\left(DD^{T}\right)^{-1} \tag{48}$$

where C and D are the normalized matrices of closest points and path points respectively and T is the desired transformation matrix.

Chapter 6

SIMULATIONS AND RESULTS

This chapter shows the simulation and results for single point matching and path matching techniques to demonstrate the feasibility of these algorithms.

6.1 Single Point Matching

As explained previously in chapter 3 about this technique, a simulator in Visual Basic 2005 has been made for simulating it. All the functions explained previously in chapter 3 have been implemented here. The simulation is done on an Intel Pentium 4, 3.2GHz processor, 512MB RAM.

Group search is used to increase the speed of the search. The X-values on the map data are divided into two groups positive and negative and the result got very efficient, so there is no need to use other optimization techniques here.

6.1.1 Data

Geomagnetic crustal field data has been used. Its resolution is 0.25 degrees on latitude and longitude. All the three Cartesian components of this field have been used. All the data is loaded in the memory from a file in the form of a three dimensional array, which takes some time, but this operation is only once.

6.1.2 GUI Description

In the main window the world map is seen as shown in figure 21. The first window on the bottom having caption “Geographical Coordinates” shows the geographical coordinates to which the point on the map is placed. The second window on the bottom having caption “Magnetic Components” shows the magnetic crustal field components in Cartesian coordinate system. The third window on the bottom having caption “Geomagnetic Properties” shows the components in spherical coordinate system.

There are three indicators on the bottom. First from the left shows that if the simulator is ready or busy. While busy it shows the loading of single points to the memory. The second indicator shows the loading of single files and the third shows total progress of loading.

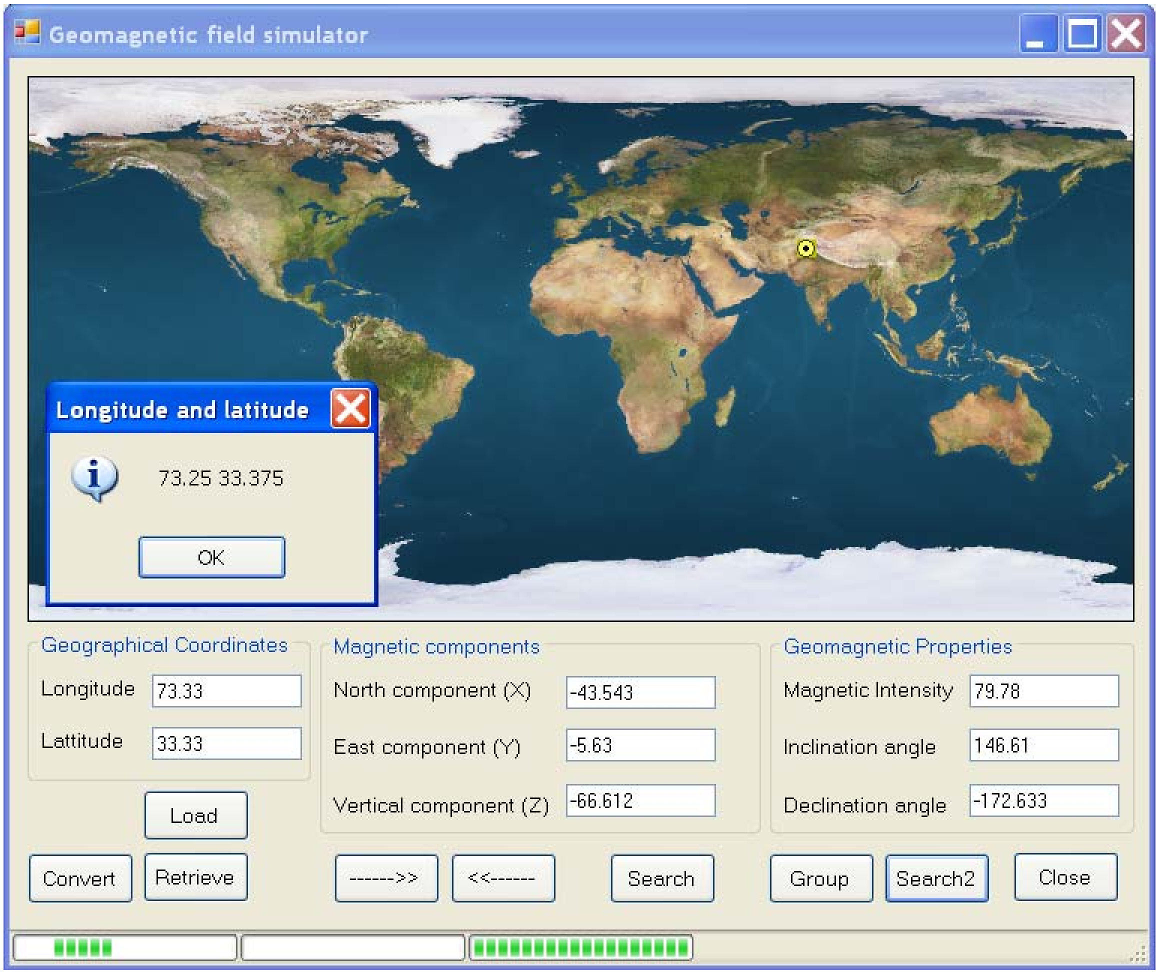

Figure 21: *GUI for single point matching technique*

6.1.3 GUI Functions

If double click is done on the map, the point is placed over there, and its geographical coordinates appears in the first window. If the coordinates in the first window is written then automatically the dot is placed on the map at that point. The first button named "Convert" retrieves its respective field components in the second window without loading anything into the memory.

The second button named "Load" loads all three files into the memory and the progress is shown in the progress bars. After loading, the "Retrieve" button shows all the coordinates in the first window. The button "------>>" converts all the Cartesian components to spherical components and shows them into the third window. The button "<<------" converts them back to Cartesian coordinates.

The "Search" button searches these Cartesian components in the memory and returns their location, which is shown in a popup window. The "Group" button groups all data in memory in two groups, less than zero and greater than zero. The grouping technique is used to reduce time for search. The button named "Search2" uses this technique and the results are quite fast as compared to the previous search. The "Close" button is used to quit.

6.1.4 Conclusion

All the values shown in figure 21 have been used. First some location having the coordinates shown in the first window has been entered. Then all the magnetic components, which are displayed in their respective windows, are retrieved. When search is done then the resulting coordinates are displayed in a popup window shown. These coordinates differ from those entered due to the resolution of the map data which is 0.25. It is found that there is only one location matching every time. Therefore, it is concluded that this technique can be implemented in AUV's for real time missions.

6.2 Path Matching

The AUV starts from initial position, travels along an arbitrary path and at the same time, onboard sensors measure the geomagnetic field periodically. The simulation is done in MATLAB 7.4.0.

6.2.1 Data

The geomagnetic map used is the X-component from the standard WMM2005 (World Magnetic Model) [20] of January 1, 2009, 100m below sea-level at the Arabian Sea from 60'0'0", 10'0'0" to 70'0'0", 20'0'0". The grid of the map is 0.5 degrees in latitude and longitude as shown in figure 22.

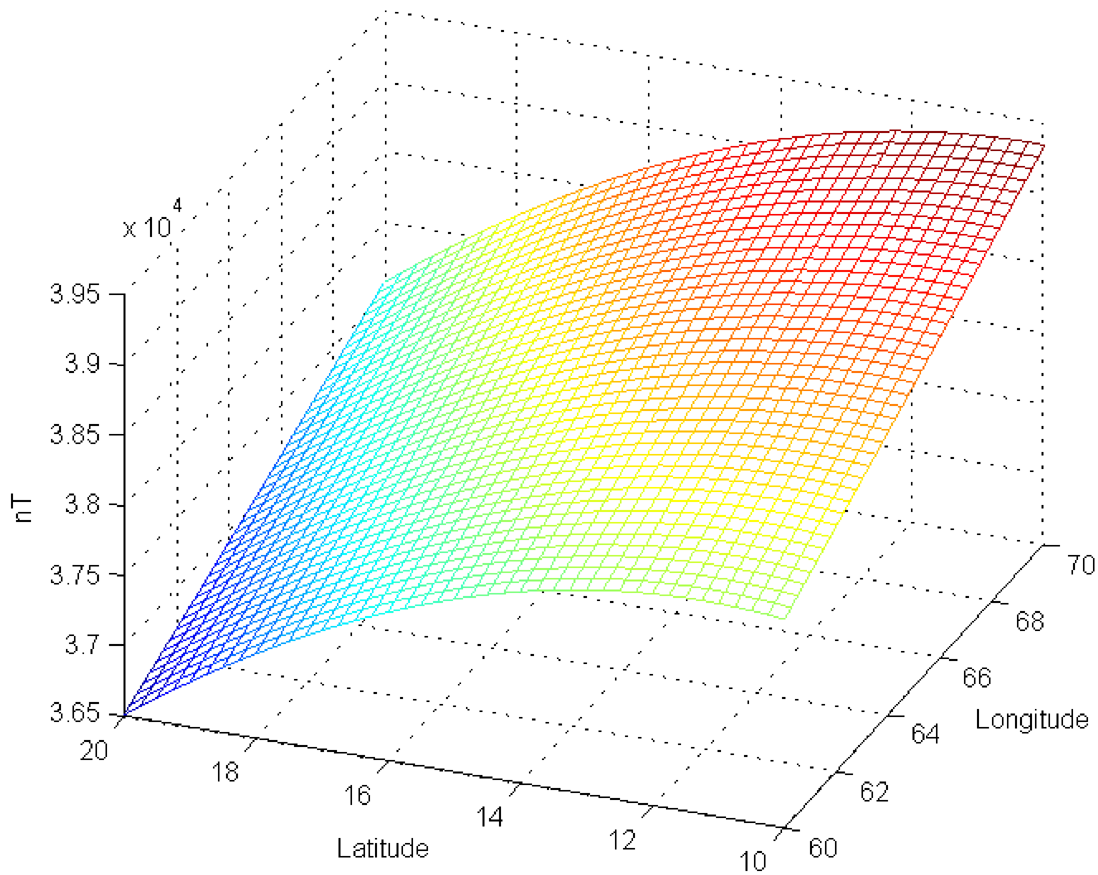

Figure 22: *Geomagnetic Reference map in 3D*

The geomagnetic samplings are taken such that the vehicle is considered to cover 1 degree in latitude and longitude. But due to drift and other environmental conditions the vehicle did not follow the specific route. For the sake of simulation, the error is added in the data as described below.

6.2.2 Error

All inertial navigation systems suffer from "integration drift" [21], which means a small error will accumulate with the passage of time and finally give us a large error. To simulate this error, a random error and the error due to the previous position is appended to it.

First this error is added to the longitude and then latitude, while the geomagnetic intensity is free of error for the first simulation. The starting point does not contain any error. The range for randomness is from -0.5 to 0.5 in both axis. The threshold τ is 1×10^{-5}.

6.2.3 Simulation Results of ICP Algorithm without Sensor Error

First it is assumed that there is no error in the sensors. So INS error is added to the path. The figure 23 is in 3D but to illustrate, it is shown in 2D.

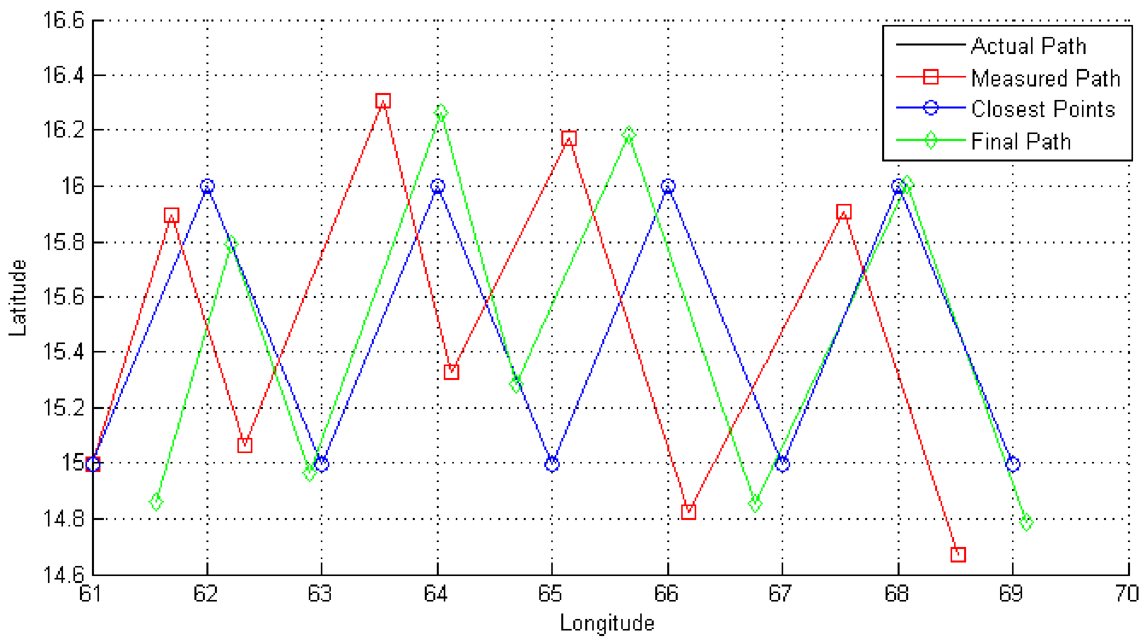

Figure 23: *Plot showing all the paths*

The first path shown in legend is the actual path. This path is usually not known in real time experiments and it is our required path. The second path is the measured path by INS. The third path is the path showing closest points to the measured path in the last iteration. The last one is the resultant path which is obtained after the application of the algorithm.

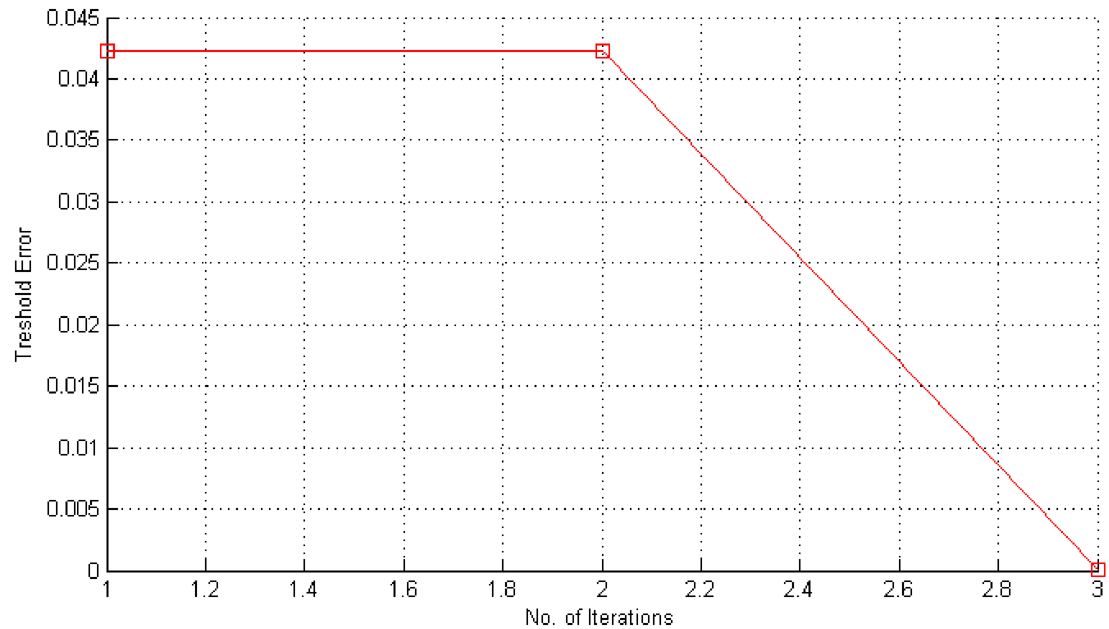

Figure 24: *Showing the convergence of the algorithm*

Here the threshold is achieved within three iterations as shown in figure 24. The threshold of 1×10^{-6} is set here.

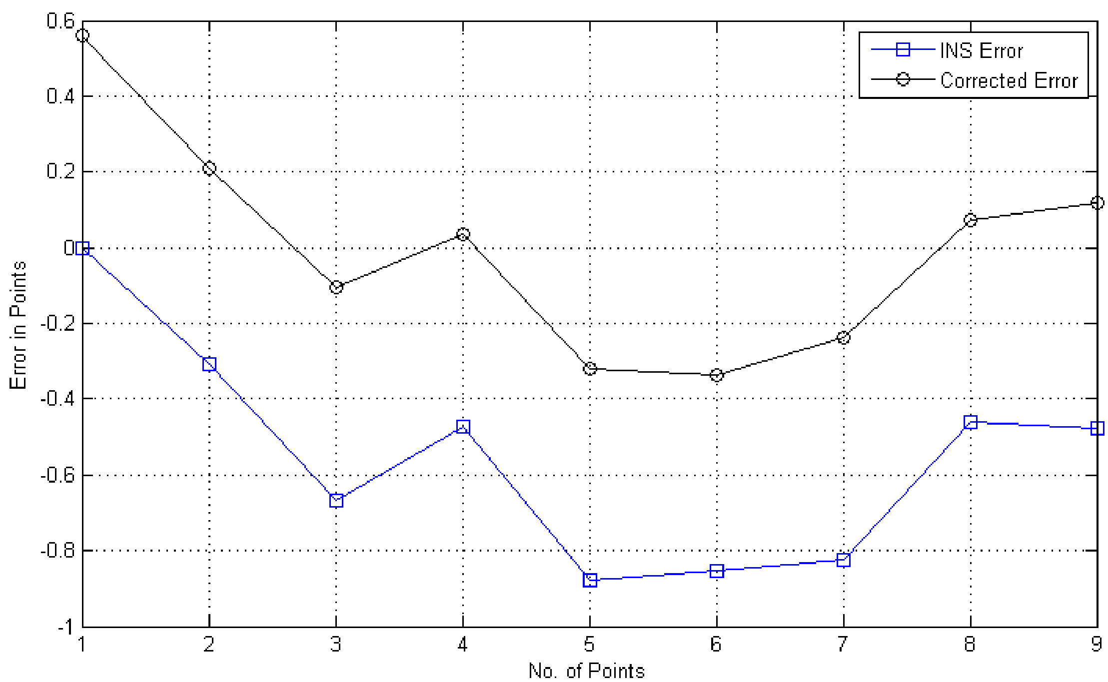

Figure 25: *Figure showing the longitudinal error*

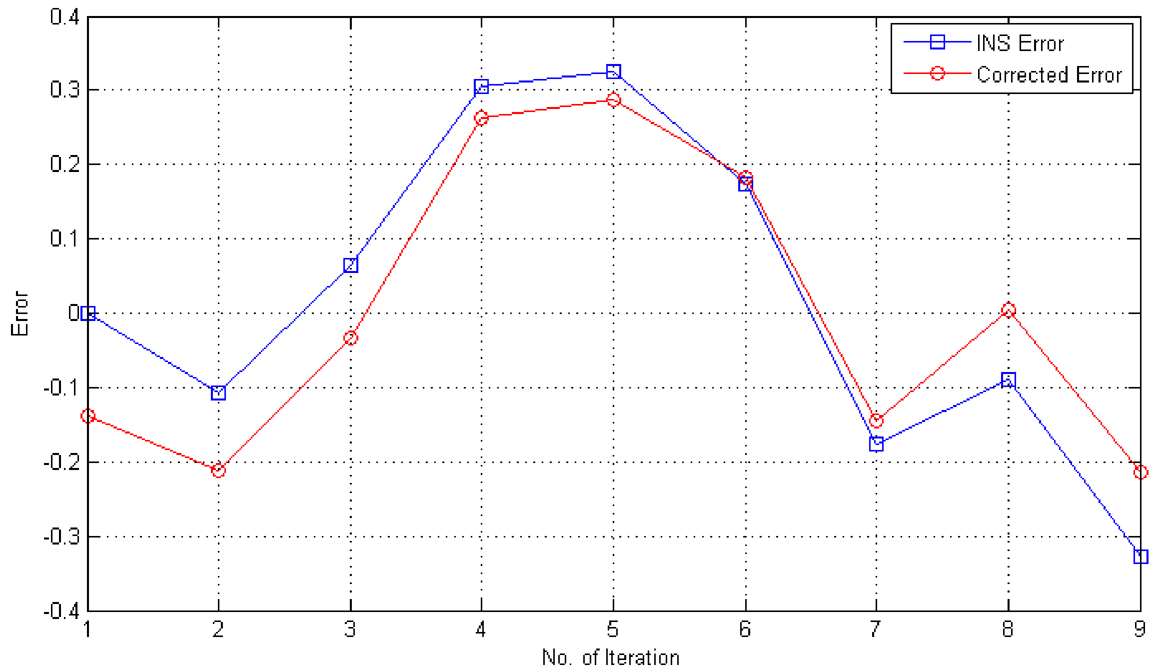

Figure 26: *Showing the latitudinal error*

The figures 25 and 26 show errors in longitude and latitude respectively. In both figures the graph starting from zero is the measured path error, while the other ones are after the application of the algorithm.

6.2.4 Simulation Results of ICP Algorithm with Sensor Error

In figure 27, the actual path was not visible because the actual path lied on the closest point's path. Here a random error of less than 5nT is used in the sensors for the simulation purpose. In the figure 27, it is seen that the first path in the legend is actual path. Another point can be noted that some closest points do not lie on the actual path but still this algorithm has reduced the error.

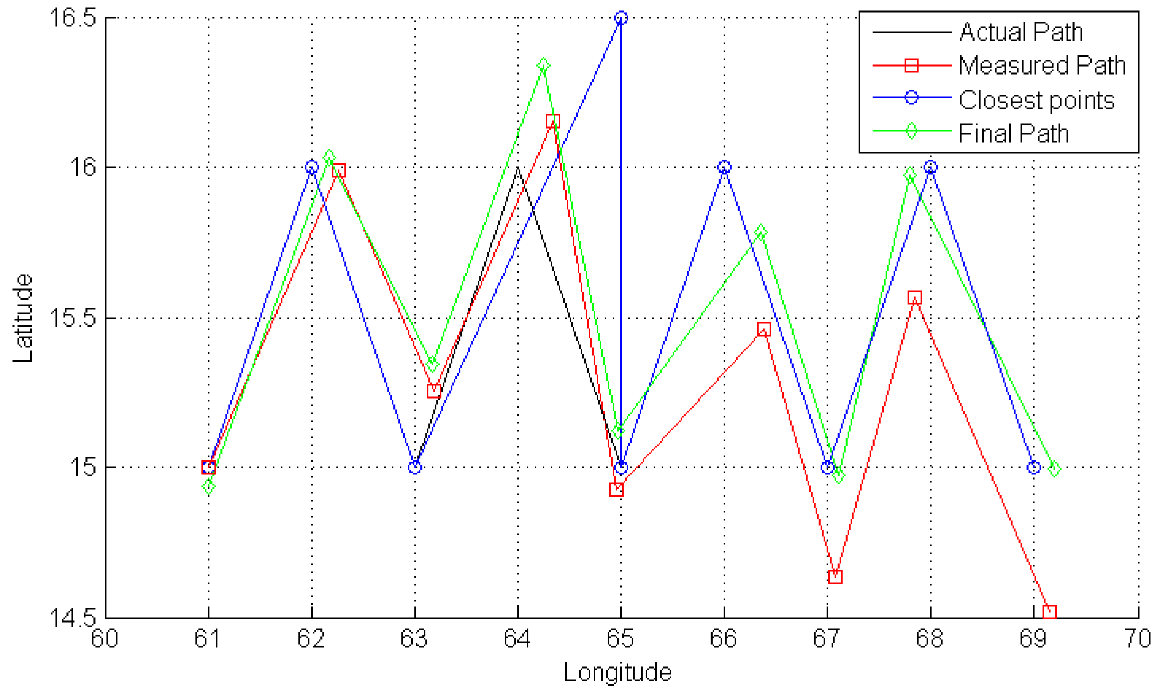

Figure 27: *Plot showing all the paths*

Usually the final position is emphasized, so in this case the error in the final position has been reduced despite the change of the closest points. It is seen that for errors greater than 5nT (and grid spacing 0.5 degrees) the optimum results are not achieved, because closest points are changed by the algorithm. But if there is no error in the sensors then it can converge to the actual path for large errors in the latitude and longitude.

The graph shown in figure 28 shows the convergence of the algorithm. On the x-axis it shows the number of iteration and y-axis shows the error.

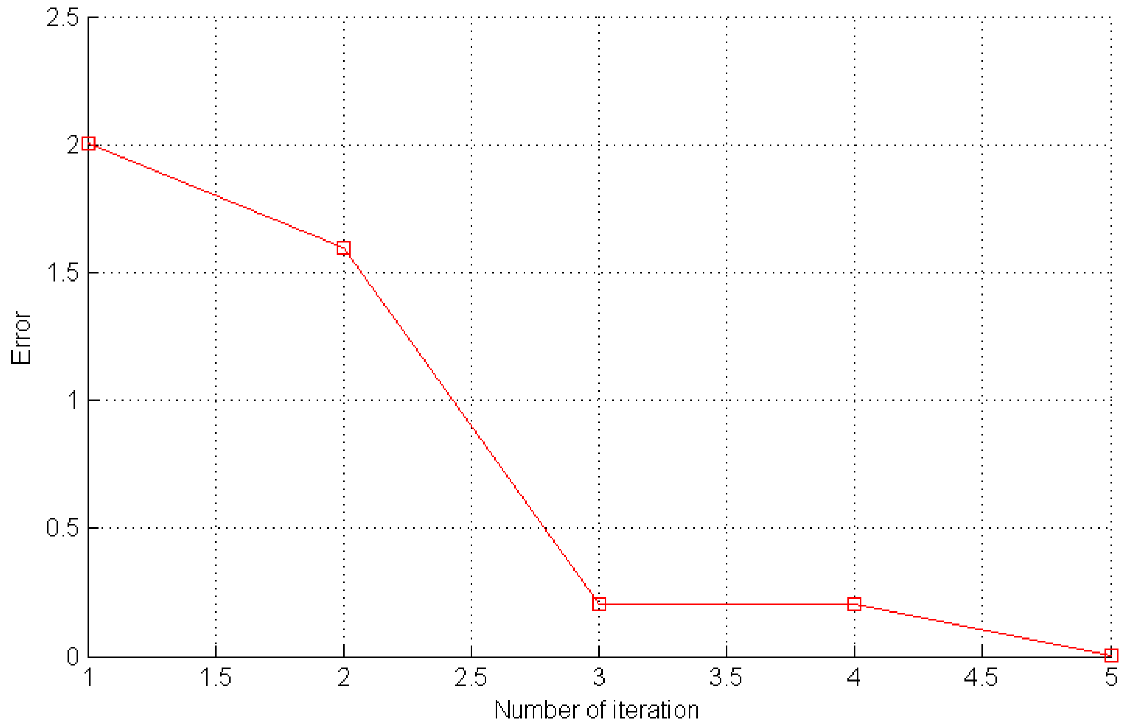

Figure 28: *Showing the convergence of the algorithm*

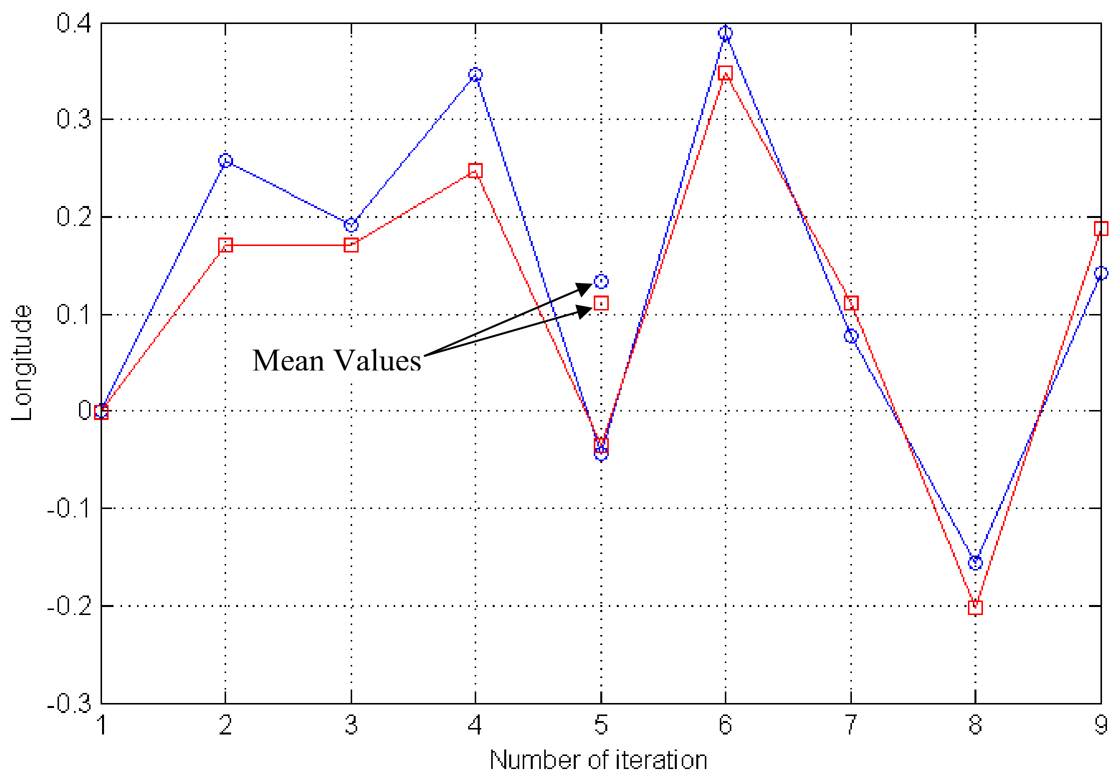

Figure 29: *Plot showing errors in longitude*

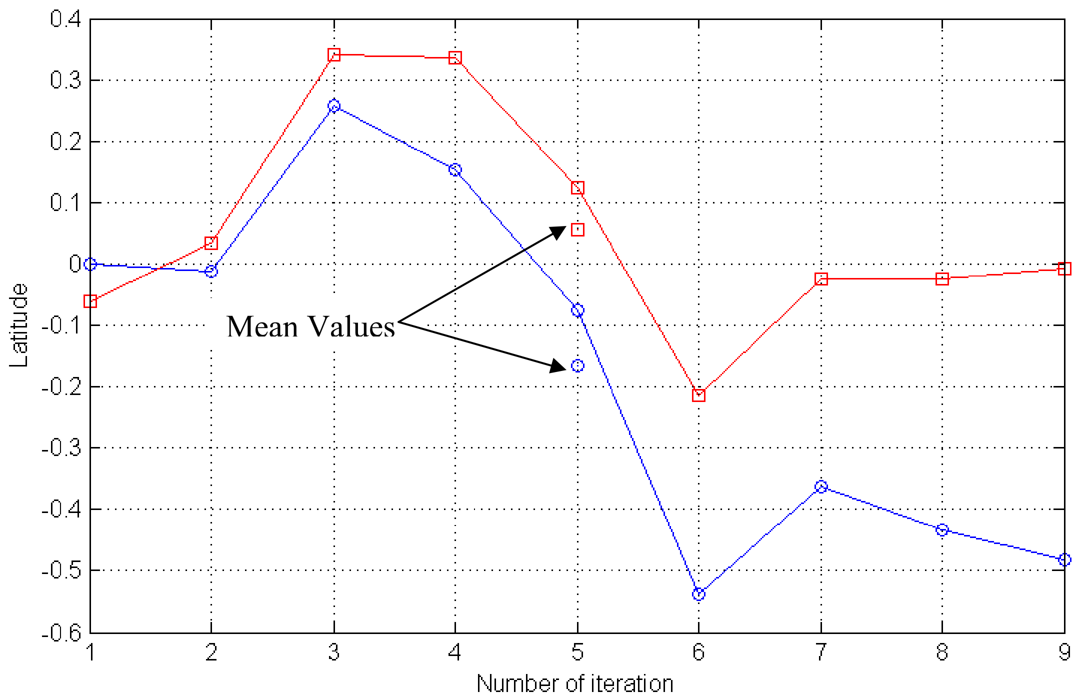

Figure 30: *Figure showing latitudinal error*

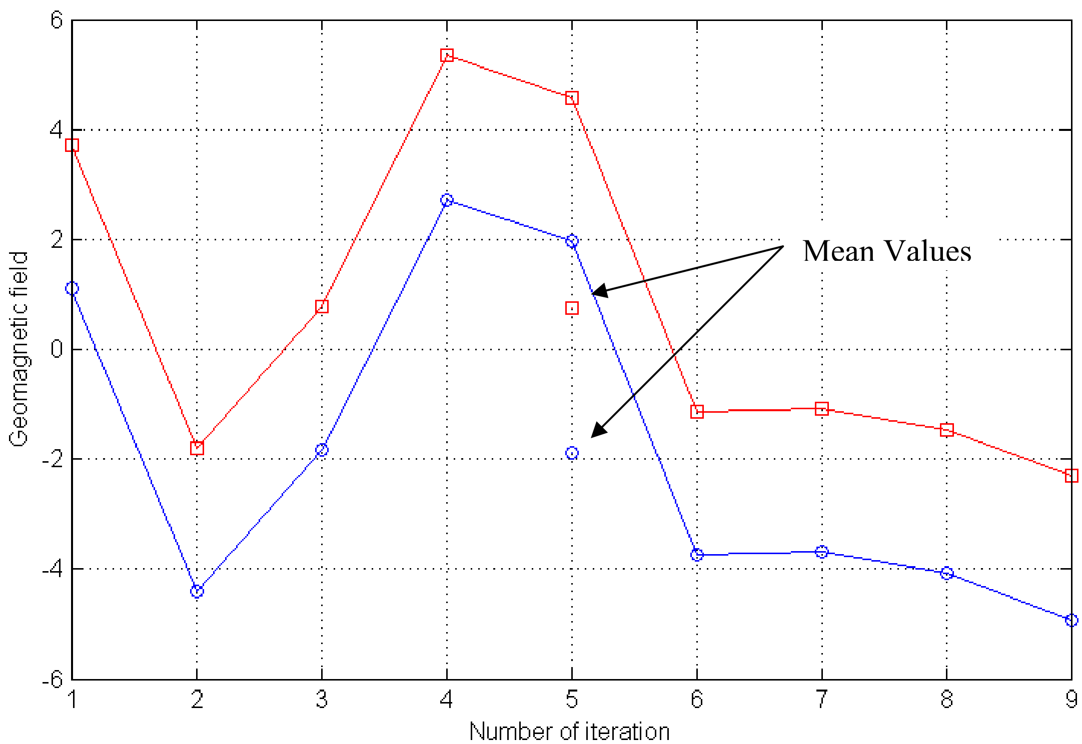

Figure 31: *Shows magnetic sensor error*

The figures 29 and 30 show errors in longitude and latitude respectively. In both figures the graph starting from zero is the measured path error, while the other one is after the application of the algorithm. The points in the middle show their respective means or average errors.

Figure 31 shows the error in sensors, and it is clearly seen that this algorithm has reduced the error in magnetometer readings which was intentionally induced.

6.2.5 Simulation Results of MMA without Sensor Error

First it is assumed that there is no error in the sensors. So INS error is added to the path. The figure 32 is in 3D but to illustrate, it is shown in 2D.

The first path shown in legend in figure 32 is the actual path which is known and it is not given in real time experiments. The second path is the measured path by INS. The third path is the path showing closest points to the measured path in the last iteration. The last one is the resultant path which is obtained after the application of the algorithm.

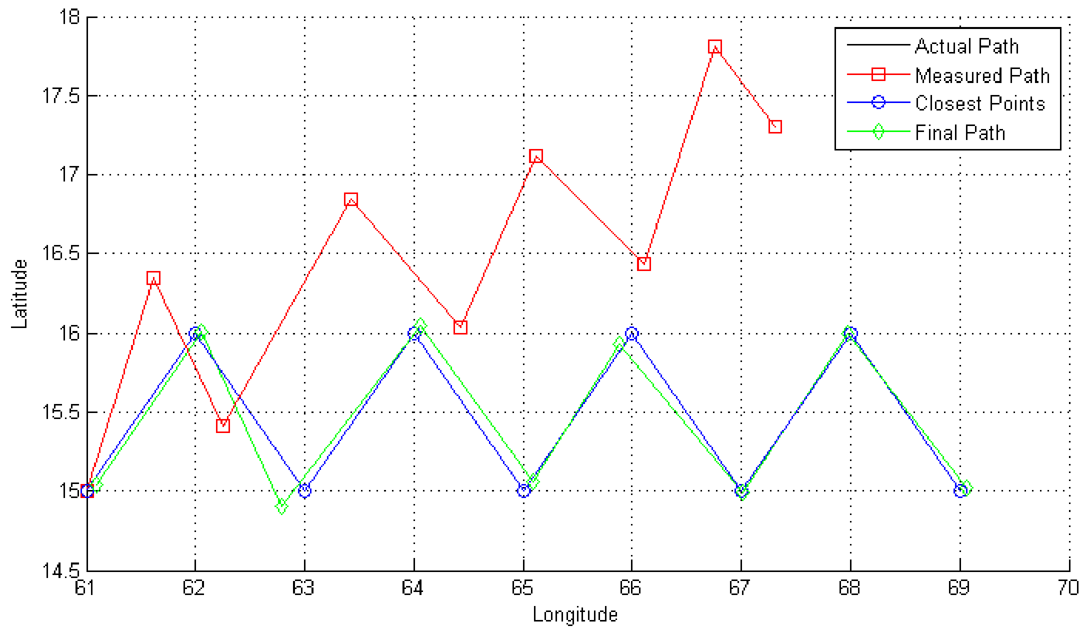

Figure 32: *Plot showing all the paths*

The figures 33 and 34 show errors in longitude and latitude respectively. In both figures the graph starting from zero is the measured path error, while the other one is after the application of the algorithm.

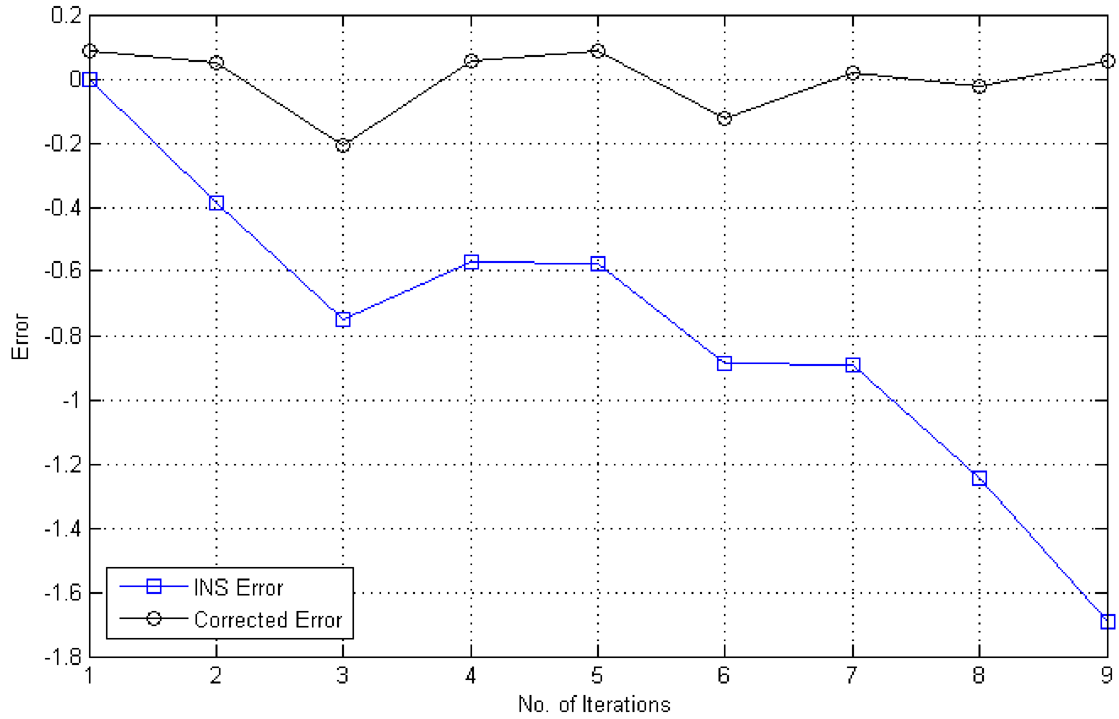

Figure 33: *Figure showing the longitudinal error*

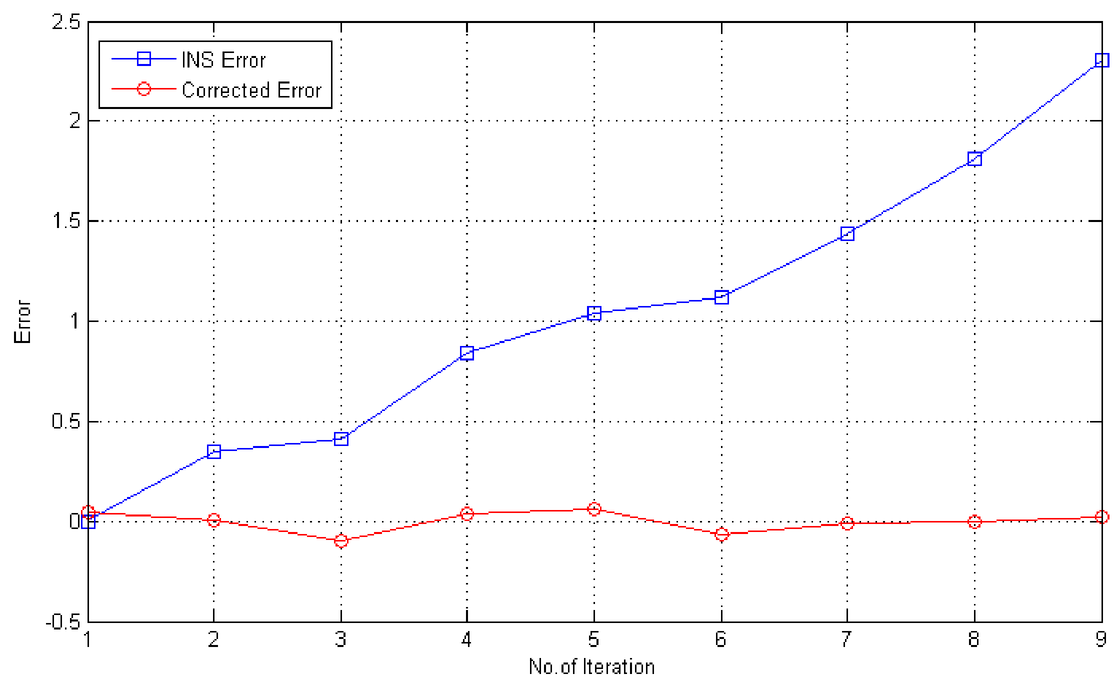

Figure 34: *Figure showing the latitudinal error*

6.2.6 Simulation Results of MMA with Sensor Error

In figure 35, the actual path was not visible because the actual path lied on the closest point's path. Here a random error of less than 3nT is used in the sensors for the simulation purpose. In the figure below it is seen that the first path in the legend is actual path. Another point can be noted that some closest points do not lie on the actual path but still this algorithm has reduced the error.

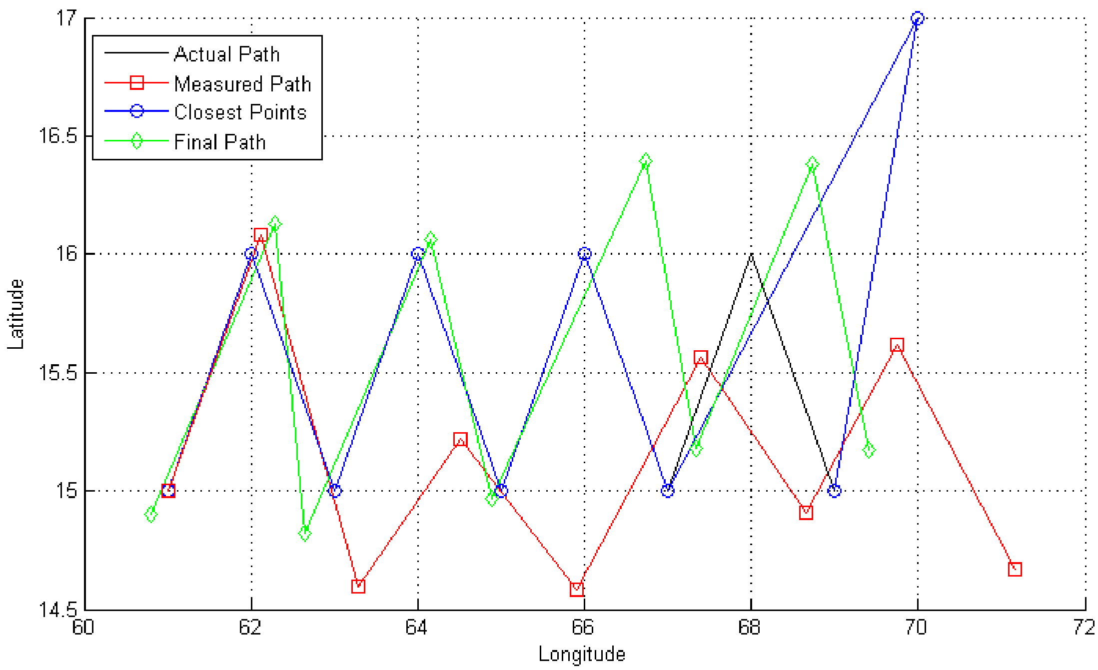

Figure 35: *Plot showing all the paths*

Usually the final position is emphasized, so in this case the error in the final position has been reduced despite the change of the closest points. It is seen that for errors greater than 3nT (and grid spacing 0.5 degrees) the optimum results are not achieved, because closest points are changed by the algorithm. But if there is no error in the sensors then it can converge to the actual path for large errors in the latitude and longitude.

The figures 36 and 37 show errors in longitude and latitude respectively. In both figures the graph starting from zero is the measured path error, while the other one is after the application of the algorithm. The points in the middle show their respective means or average errors.

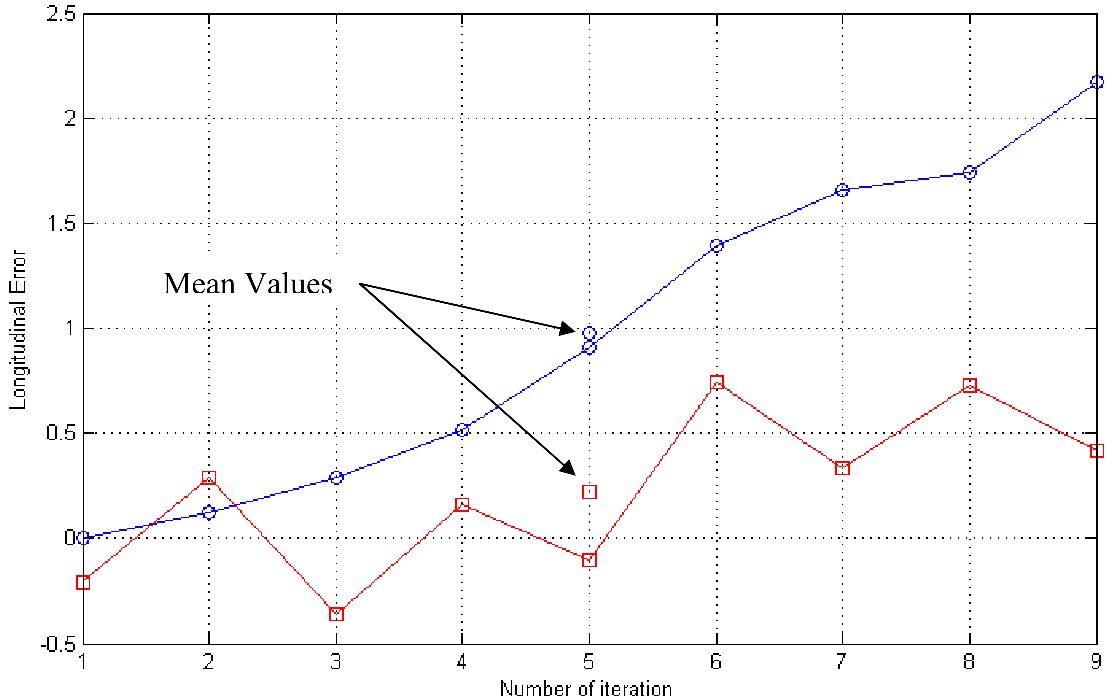

Figure 36: *Plot showing errors in longitude*

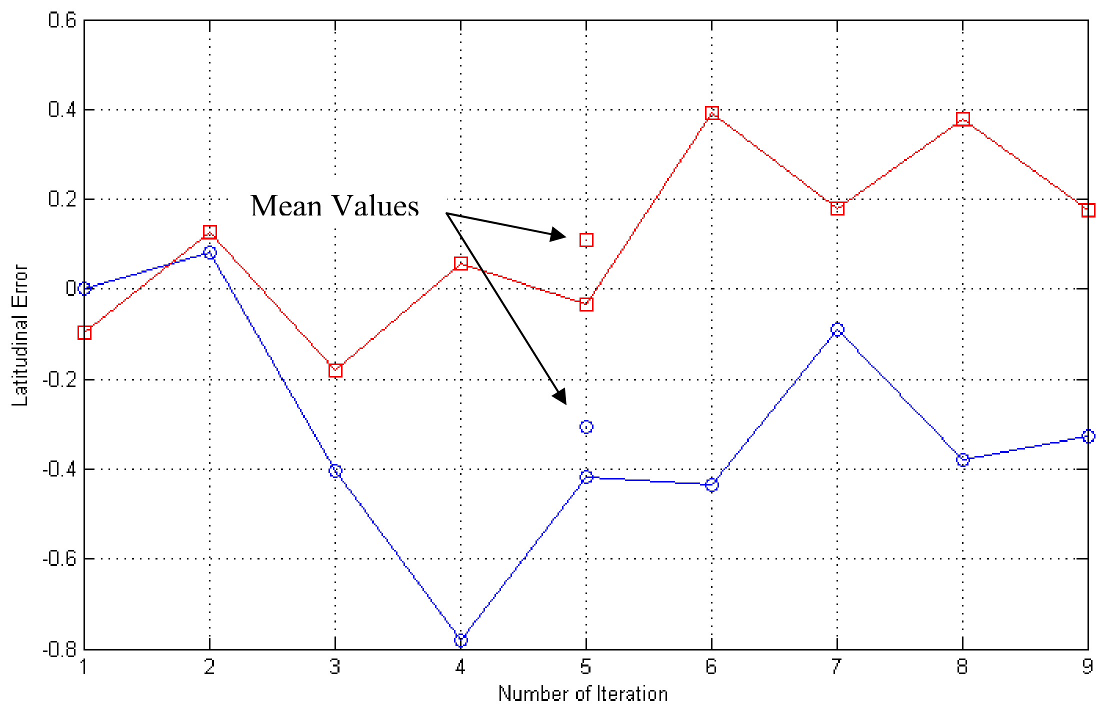

Figure 37: *Plot showing errors in latitude*

The figure 38 shows graph showing error in sensors, and it is clearly seen that this algorithm has reduced the error in magnetic field which has been intentionally induced.

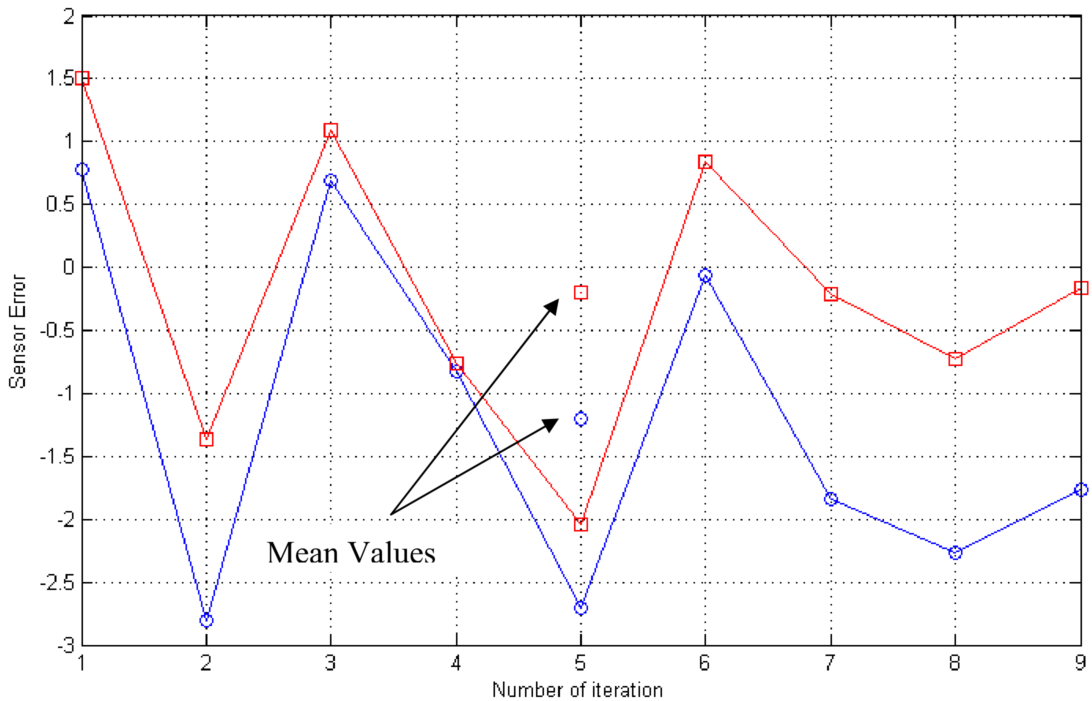

Figure 38: *Figure showing magnetic sensor error*

Chapter 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

We are still in dark about the sea turtles (and other animals) how they use the geomagnetic field for localization but this thesis has given us a clue that how can we use the this source as a reference for localization of underwater manned or unmanned vehicles. Turtles show a behavior by which we can infer that they can sense geomagnetic field intensity and inclination angle. We have pre-surveyed geomagnetic models by which we can find the geomagnetic field values at any point and at any time. In this thesis we have studied different algorithms which can be used for geomagnetic localization.

In Chapter 2, we have described different techniques which can be used for localization. First they are categorized into two methods namely, single point matching and path matching. Then path matching can be done by using two algorithms ICP and Menq's algorithms. In Chapter 3, ICP algorithm is explained in detail. Also different techniques used in it are explained thoroughly. Similarly, Chapter 4 explains the details of modified Menq's algorithm. Chapter 5 explains the other procedures which are used in this simulation and some properties of rotation. Chapter 6 shows the simulation and results for all the methods.

As we know that nature which is created by Allah is example-less in its beauty and perfection. Man can do anything except reaching even near nature. So the inherited ability of turtles of seeking their way out in such an immense ocean can not be fully understood or implemented. Man has learned to fly from birds and swim from fish. Similarly we can explore nature and implement it in our inventions. Specifically, we can learn from sea turtles how to localize in water.

This thesis work is a torch which gives researchers and scientists a vision to look for undiscovered techniques for localization underwater; where all other techniques fail. We have used geomagnetic field as a reference. The geomagnetic field has drawbacks of magnetic storm (which are caused due to sun) and its variation with time. Therefore, this field can not be fully trusted all the time. These techniques minimize error in INS as our simulations show. Finally, we conclude that these techniques can be implemented in real time operations and give good results.

7.2 Future Work

1) In this algorithm, to increase accuracy we have used only the X component of the geomagnetic field, we can use the Y and Z components as the 4^{th} and 5^{th} dimension.
2) In this thesis we have not taken depth into account; any changes in depth can not be corrected, to solve this we can use depth as the 3^{rd} dimension and magnetic field as higher dimensions.
3) We have used rigid transformation in ICP algorithm, which does not change the shape of the path. If we use any technique which can have non-rigid transformation then we can get better results.

References

[1] Y. Camlitepe and D. J. Stradling, "Wood Ants Orient to Magnetic Fields", 1995.

[2] T. E. Dennis, M. J. Rayner and M. M. Walker, "Evidence that pigeons orient to geomagnetic intensity during homing", proceedings of the royal society, 2007.

[3] K. J. Lohmann and C. M. F. Lohmann, "Orientation and Open-Sea Navigation In Sea Turtles", Journal of Experimental Biology, Vol 199, Issue 1 73-81, 1996.

[4] M. M. Walker, M. E. Bitterman, "Attached magnets disrupt magnetic field discrimination by honeybees", Journal of Experimental Biology, 141:447-451, 1988.

[5] NATURE, Vol. 428, www.nature.com/nature, page 909, 29 April 2004.

[6] L. C. Boles, and K. J. Lohmann, "True navigation and magnetic maps in spiny lobsters," Nature, no. 421, pp. 60-63, 2003.

[7] A. J. Kalmijn, "Electric and magnetic field detection in elasmobranch fishes," Science, no. 318, pp. 916–918, 1982.

[8] P. J. Besl and N. D. Mckay. "A method for registration of 3-d shapes", Pattern Analysis and Machine Intelligence, IEEE Transactions on, 14(2):239{256, 1992.

[9] Z. Li, J. Gou, and Y. Chu, "Geometric Algorithms for Workpiece Localization", IEEE Transactions on Robotics and Automation, Vol. 14, No. 6, DECEMBER 1998.

[10] http://en.wikipedia.org/wiki/Rotation_group

[11] http://en.wikipedia.org/wiki/Orthogonal_matrix

[12] http://en.wikipedia.org/wiki/Singular_value_decomposition

[13] http://en.wikipedia.org/wiki/Euclidean_group#Direct_and_indirect_isometries

[14] http://en.wikipedia.org/wiki/Normal_matrix

[15] http://en.wikipedia.org/wiki/Standard_basis

[16] http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm

[17] wordnetweb.princeton.edu/perl/webwn

[18] http://www.britannica.com/EBchecked/topic/229754/geomagnetic-field

[19] L. Whitcomb, D. Yoeger, H. Singh and J. Howland, "Advances in underwater robot vehicles for deep ocean exploration"

[20] http://www.ngdc.noaa.gov/geomag/WMM/

[21] http://en.wikipedia.org/wiki/Inertial_navigation_system

[22] http://geomag.usgs.gov/faqs.php

[23] http://www.ngdc.noaa.gov/geomagmodels/IGRFWMM.jsp

[24] http://geophysics.ou.edu/solid_earth/notes/mag_earth/magnetic_field_a.gif

[25] http://geophysics.ou.edu/solid_earth/notes/mag_earth/magnetic_field_b.gif

[26] http://geophysics.ou.edu/solid_earth/notes/mag_earth/magnetic_field_c.gif

[27] http://en.wikipedia.org/wiki/Dead_reckoning

[28] K.J. Lohmann, J.T. Hester and C.M.F. Lohmann, "Long-distance navigation in sea turtles", Ethology Ecology & Evolution 11: 1-23, 1999

[29] Zhixin Ren, Longwei Chen, Hui Zhang and Meiping Wu, "Research on Geomagnetic-matching Localization Algorithm for Unmanned Underwater Vehicles", Proceedings of IEEE International Conference on Information and Automation, China, 2008

[30] http://www.britannica.com/EBchecked/topic/175210/dynamo-theory

Appendix A

Single Point Matching Algorithm

```
Public Class Form1
    Dim M(1440, 720, 3)
    Dim G(4, 3110400) As Single

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        Dim Point1 As Point

        Point1.X = 2.042 * TextBox1.Text + 375.5
        Point1.Y = -2 * TextBox2.Text + 188

        RadioButton1.Location = Point1


    End Sub


    Private Sub PictureBox1_DoubleClick(ByVal sender As Object, ByVal e
As System.EventArgs) Handles PictureBox1.DoubleClick
        Dim point2 As New Point(10, 42)
        RadioButton1.Location = Windows.Forms.Cursor.Position -
Windows.Forms.Form.ActiveForm.Location - point2
        TextBox1.Text = Format$(((RadioButton1.Location.X.ToString -
375.5) / 2.042), "#.000")
        TextBox2.Text = Format$(((RadioButton1.Location.Y.ToString -
188) / -2), "#.000")

    End Sub


    Private Sub Button3_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button3.Click
        REM Dim fsys As New Scripting.FileSystemObject
        Dim oneline As String
        Dim strarr() As String
        Dim space = " "
        Dim Lon, Lat, GMF, lat1, lat2, lon1, lon2 As Double
        Dim Tblon, Tblat As Double
        Dim x, y, z As Single

        Tblon = Double.Parse(TextBox1.Text)
        Tblat = Double.Parse(TextBox2.Text)
        'MsgBox(Tblon & ":" & Tblat)
        If Tblon < 0 Then Tblon = 360 + Tblon
        'tstream = fsys.OpenTextFile("G:\Documents and Settings\EME\My
Documents\Visual Studio 2005\Projects\AUV 2\NGDCx.xyz",
Scripting.IOMode.ForReading)

        FileOpen(1, "C:\Documents and Settings\AUV\My Documents\Visual
Studio 2005\Projects\NGDCx.xyz", OpenMode.Input)
        Do While Not EOF(1)
```

```
            oneline = LineInput(1)
            'oneline = tstream.readline
            'MessageBox.Show(oneline)
            strarr = oneline.Split(space)
            y = 0
            z = strarr.Length
            For x = 0 To z - 1
                If strarr(x) = "" Then GoTo 100
                If y = 0 Then
                    y = 1
                    Lon = Double.Parse(strarr(x))
                ElseIf y = 1 Then
                    y = 2
                    Lat = Double.Parse(strarr(x))
                ElseIf y = 2 Then
                    GMF = Double.Parse(strarr(x))
                End If
                'MsgBox(Lon & Lat & GMF)
100:        Next


            'Lat = Math.Round(Lat, 2)
            lat1 = Lat - 0.125
            lat2 = Lat + 0.125
            lon1 = Lon - 0.125
            lon2 = Lon + 0.125
            If Tblat > lat1 And Tblat <= lat2 Then

                'Lon = Math.Round(Lon, 2)
                If Tblon > lon1 And Tblon <= lon2 Then
                    Beep()
                    TextBox6.Text = GMF
                    GoTo 200
                End If


            End If

        Loop
200:    FileClose(1)
    End Sub

    Private Sub Button4_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button4.Click
        Dim oneline As String
        Dim strarr() As String
        Dim space = " "
        Dim Lon, Lat, GMF As Double
        Dim Tblon, Tblat As Double
        Dim a, b, x, y, z As Single

        Tblon = Double.Parse(TextBox1.Text)
        Tblat = Double.Parse(TextBox2.Text)

        If Tblon < 0 Then Tblon = 360 + Tblon
```

```
        ToolStripStatusLabel1.Text = "Busy"
        FileOpen(1, "C:\Documents and Settings\AUV\My Documents\Visual
Studio 2005\Projects\NGDCx.xyz", OpenMode.Input)
        FileOpen(2, "C:\Documents and Settings\AUV\My Documents\Visual
Studio 2005\Projects\NGDCy.xyz", OpenMode.Input)
        FileOpen(3, "C:\Documents and Settings\AUV\My Documents\Visual
Studio 2005\Projects\NGDCz.xyz", OpenMode.Input)

        For a = 1 To 3
            Do While Not EOF(a)

                oneline = LineInput(a)
                strarr = oneline.Split(space)
                y = 0
                z = strarr.Length
                For x = 0 To z - 1
                    If strarr(x) = "" Then GoTo 100
                    If y = 0 Then
                        y = 1
                        Lon = Double.Parse(strarr(x))
                    ElseIf y = 1 Then
                        y = 2
                        Lat = Double.Parse(strarr(x))
                    ElseIf y = 2 Then
                        GMF = Double.Parse(strarr(x))
                    End If

100:            Next x

                M(4 * Lon, 4 * Lat + 359.5, a) = GMF
                b = b + 1

                ToolStripProgressBar2.Value = Lat + 90
                ToolStripStatusLabel1.Style = ProgressBarStyle.Blocks
                ToolStripStatusLabel1.Value = Lon
                ToolStripProgressBar1.Value = b
            Loop
            FileClose(a)

        Next a

        ToolStripProgressBar2.Value = 0
        ToolStripStatusLabel1.Style = ProgressBarStyle.Marquee

        MsgBox("All file Loadings completed", MsgBoxStyle.Information,
"Information")


    End Sub

    Private Sub Button5_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button5.Click

        Dim Tblon, Tblat As Double
        Dim Rlon, Rlat As Single
```

```
        Tblon = Double.Parse(TextBox1.Text)
        Tblat = Double.Parse(TextBox2.Text)

        If Tblon < 0 Then Tblon = Tblon + 360

        Beep()
        Rlon = Math.Round(4 * Tblon)
        Rlat = Math.Round(4 * Tblat + 359.5)

        TextBox6.Text = M(Rlon, Rlat, 1)
        TextBox7.Text = M(Rlon, Rlat, 2)
        TextBox8.Text = M(Rlon, Rlat, 3)


    End Sub

    Private Sub Button6_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button6.Click
        TextBox6.Text = Math.Round(TextBox3.Text *
Math.Sin(TextBox4.Text * Math.PI / 180) * Math.Cos(TextBox5.Text *
Math.PI / 180), 3)
        TextBox7.Text = Math.Round(TextBox3.Text *
Math.Sin(TextBox4.Text * Math.PI / 180) * Math.Sin(TextBox5.Text *
Math.PI / 180), 3)
        TextBox8.Text = Math.Round(TextBox3.Text *
Math.Cos(TextBox4.Text * Math.PI / 180), 3)
    End Sub

    Private Sub Button2_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button2.Click
        Me.Close()

    End Sub

    Private Sub Button7_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button7.Click
        TextBox3.Text = Math.Round(Math.Sqrt(TextBox6.Text ^ 2 +
TextBox7.Text ^ 2 + TextBox8.Text ^ 2), 3)
        TextBox5.Text = Math.Round((Math.Atan2(TextBox7.Text,
TextBox6.Text)) * 180 / Math.PI, 3)
        TextBox4.Text = Math.Round((Math.Atan2(Math.Sqrt(TextBox6.Text ^
2 + TextBox7.Text ^ 2), TextBox8.Text)) * 180 / Math.PI, 3)
    End Sub

    Private Sub TextBox1_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TextBox1.LostFocus
        Dim Point1 As New Point
        Point1.X = 2.042 * TextBox1.Text + 375.5
        Point1.Y = -2 * TextBox2.Text + 188
        RadioButton1.Location = Point1
    End Sub

    Private Sub TextBox2_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TextBox2.LostFocus
        Dim Point1 As New Point
        Point1.X = 2.042 * TextBox1.Text + 375.5
        Point1.Y = -2 * TextBox2.Text + 188
```

```
        RadioButton1.Location = Point1
    End Sub

    Private Sub Button1_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button1.Click
        Dim Mx, My, Mz, Rblon, Rblat As Double
        Dim x, y As Integer
        Mx = TextBox6.Text
        My = TextBox7.Text
        Mz = TextBox8.Text
        For x = 1 To 1440
            For y = 1 To 720
                If Math.Round(Mx, 2) = Math.Round(M(x, y, 1), 2) Then

                    If Math.Round(My, 2) = Math.Round(M(x, y, 2), 2)
Then

                        If Math.Round(Mz, 2) = Math.Round(M(x, y, 3), 2)
Then

                            Rblon = x / 4
                            Rblat = (y - 359.5) / 4
                            MsgBox(Rblon & "," & Rblat,
MsgBoxStyle.Information, "Point found")

                        End If
                    End If
                End If

            Next y
        Next x
        MsgBox("Search completed")
    End Sub


    Private Sub Button8_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button8.Click
        Dim x, y, z, i, j As Integer
        z = 1
        For y = 1 To 720
            For x = 1 To 1440
                If M(x, y, 1) < 0 Then
                    G(1, i) = M(x, y, z)
                    G(2, i) = x
                    G(3, i) = y
                    G(4, i) = z
                    i = i + 1
                Else
                    G(1, j + 518400) = M(x, y, z)
                    G(2, j + 518400) = x
                    G(3, j + 518400) = y
                    G(4, j + 518400) = z
                    j = j + 1
                End If
            Next
        Next
```

```
        MsgBox(i & " "c & j, MsgBoxStyle.Information, "Completed")

    End Sub

    Private Sub Button9_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button9.Click
        Dim i = 1, j, r As Integer
        Dim TbX, TbY, TbZ, Longi, Latti As Single
        TbX = TextBox6.Text
        TbY = TextBox7.Text
        TbZ = TextBox8.Text
        r = 2
        If TextBox6.Text >= 0 Then
            i = 518400
        End If

        For j = i To 1043146
            If Math.Round(TbX, r) = Math.Round(G(1, j), r) Then
                'MsgBox(M(G(2, j), G(3, j), 2) & " "c & G(2, j) & " "c &
G(3, j) & " "c & G(4, j))
                If Math.Round(M(G(2, j), G(3, j), 2), r) =
Math.Round(TbY, r) Then
                    If Math.Round(M(G(2, j), G(3, j), 3), r) =
Math.Round(TbZ, r) Then
                        'MsgBox("All components matched")
                        Longi = G(2, j) / 4
                        Latti = (G(3, j) - 359.5) / 4
                        MsgBox(Longi & " "c & Latti,
MsgBoxStyle.Information, "Longitude and latitude")
                    End If

                End If

            End If

        Next
        MsgBox("Search Complete", MsgBoxStyle.Information, "Search 2")

    End Sub

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    End Sub
End Class
```

Appendix B

ICP Algorithm Function

```
function T = myicp8(Q,D)

format short g

maxiter=50;                              % Max iteration and
Thres=1e-5;                              % error threshold

Dres=D;

Tpre=eye(4,4);
T=eye(4,4);
figure(2);
hold on;                                 % To draw a graph in a loop


for k= 1:maxiter

    E=closest(Q,Dres);                   % Find the closest point

    Rtemp=E*Dres';                       % Calculate Rotation
    [U S V]=svd(Rtemp);                  % Calculate singular-
                                         % value-decomposition
    U(:,end)=U(:,end)*det(U*V');         % Adjust negative determinant
    Rres = U*V';                         % Drop the diagonal

    Drot=Rres*Dres;                      % Rotate the data points to
                                         % the closest points
    Dm=mean(Drot,2);
    Em=mean(E,2);
    Tl=Dm-Em;                            % find the translation

    Dres=[Dres; ones(1,size(Dres,2))];   % normalize the data

    Tres=[Rres -Tl;[0 0 0 1]];           % minus sign to decrease the
error

    Dtf=Tres*Dres;                       % Transformed data


    N=norm(Tres-Tpre);                   % difference between the
                                         % previous and recent value

    Dres=Dtf(1:3,:);

    if k == 1                            % plot error in loop
        Nprev = N;
        Kprev = k;
    else
        plot([k Kprev],[N Nprev],'red');
        Nprev = N;
```

```
        Kprev = k;
    end

    if N < Thres                        % if error < threshold
        break                           % then break
    end
    T=Tres*T;
    Tpre=Tres;                          % for the next iteration this
                                        % value is prevoius value

end

k                                       % display all variables
T
Dres
N
MaxErr = max(Dres-E,[],2)               % Maximum error
Diff=norm(Dres-E)                       % difference

hold off;
figure(1);



hold on                                 % Plot all points
plot3(D(1,:),D(2,:),D(3,:),'r')
plot3(E(1,:),E(2,:),E(3,:),'blue')
plot3(Dres(1,:),Dres(2,:),Dres(3,:),'g')
hold off

% % % % % % % % % % % % % % End of program % % % % % % % % % % % % % % %
```

Appendix C

Modified Menq's Algorithm Function

```
function T=mymenq2(Q,D)

format short g

hold on

E=closest(Q,D);                    % Find closest points

O=ones(1,size(D,2));               % Normalize data
D=[D;O];
E=[E;O];

T=E*D'*(inv(D*D'))                 % Apply algorithm
Dnew=T*D;

plot3(D(1,:),D(2,:),D(3,:),'r')    % Plot all data
plot3(E(1,:),E(2,:),E(3,:),'b')
plot3(Dnew(1,:),Dnew(2,:),Dnew(3,:),'g')

hold off
```

Appendix D

Simulation M-file

```
clc
close all
clear all
download;                                  % Download Map data
Q=myload(X,Y,Z);                           % Load Map data
download1;                                 % Download Path data
D=myload(X,Y,Z);                           % Load Path data
D1=myerror(D2)                             % Add Error
hold on
plot3(D2(1,:),D2(2,:),D2(3,:),'black')     % Plot actual data
T=myicp8(Q,D1);                            % Apply ICP function
hold off
```

Index