DESIGN OF A NEUROSURGICAL MANIPULATOR FOR APPLICATIONS IN MRI ENVIRONMENT

BY

VIKRAM BANTHIA

A Thesis
Submitted to the Faculty of Graduate Studies
In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Department of Mechanical and Manufacturing Engineering
University of Manitoba
Winnipeg, Manitoba, Canada

ACKNOWLEDGEMENTS

Pursuing my Masters in Mechanical and Manufacturing engineering at the University of Manitoba has been one of the most rewarding experiences. Living and studying in Winnipeg has left me with valuable lessons and fond memories. I would like to thank god for his love and kindness which is unparalleled.

I will forever be indebted to Dr. S. Balakrishnan for giving me this opportunity to work under him. I have benefited immensely from Dr. Balakrishnan's knowledge and teaching methods. I thank him for the research assistantship provided. He has always been helpful, supportive and has encouraged me to the fullest extent. I thank him for guiding me throughout the research work many times when my knowledge regarding the subject was lacking. I would always consider it a privilege to work under Dr. Balakrishnan.

I wish to thank the University of Manitoba for providing me with this opportunity. I thank the University for awarding me the IGSES and the Manitoba Graduate Scholarship which have been of invaluable help in completing my masters. The scholarships have allowed me to focus completely on my studies and are a major reason for my success in the course work and the research as I was able to devote my entire time towards it.

I would also like to thank Ken Tarte of the Robotics and Automation lab for his support with designing and building the robotic manipulator.

I would like to thank Mrs. Janaki Balakrishnan for helping me out through the tough times when I was down with chickenpox. Her tips helped me recover fast and focus on my studies again.

No words are sufficient to describe my late mother's contribution to my life. I owe every bit of my existence to her. I wish to thank my father without whom I would never have achieved anything in life. Love, patience, support and understanding of my parents is the only reason I have come this far.

I would also like to thank my brother and sister for their support and kind words. I would like to thank my friends who have made my stay in Winnipeg so wonderful. Last but not the least I would like to thank my wife who has made my life so beautiful.

Thank you everyone who have made this possible...

Vikram Banthia

ABSTRACT

This thesis presents the design of a personal computer (PC) based needle insertion robotic manipulator for biopsy. The robot was designed and built using materials available in the research laboratory. The robot is intended primarily for use inside the confined area of a cylindrical magnetic resonance (MR) scanner. Selection of the robot geometry and novel locations for drive actuators allowed placement of actuators outside the MRI bore. The robot is modelled using Denavit-Hartenberg transformations. Custom developed software control is developed to test the functional aspects of the robot. The robot performs to the tolerance required for the stated clinical application. This thesis addresses only proof of concept chosen for the manipulator design and is not ready for any clinical trials. The work also addresses MRI compatible and safety issues and recommends appropriate materials for future development.

Traditionally, neurosurgical navigation has relied on preoperative images and the assumption that anatomical structures of interest remain in the same position with respect to each other and the fiducial markers used for registration. However, during surgery, tissue deformation and shift disrupt the spatial relation between the patient and the preoperative image volumes. This results in localization errors. Developing a manipulator that works inside an imaging machine guided by real time images is expected to minimize the problem of “tissue shift” during the surgery.

Dedicated to my late Mother

TABLE OF CONTENTS

1 INTRODUCTION 1

1.1 OBJECTIVE 1

1.2 MOTIVATION 2

1.3 CHAPTER OVERVIEW 5

2 BACKGROUND LITERATURE 7

2.1 INTRODUCTION 7

2.2 ROBOTS IN SURGERY 7

2.3 ROBOTIC TECHNIQUES FOR SURGERY 8

2.3.1 MINIMALLY INVASIVE PROCEDURES 8

2.3.2 IMAGE-BASED PROCEDURES 10

2.4 SURGICAL APPLICATIONS 10

2.5 MRI GUIDED INTERVENTION 12

2.6 MRI TECHNOLOGY 13

2.6.1 MR-SCANNERS, MR-SAFETY, AND MR-COMPATIBILITY 14

2.6.2 MR-COMPATIBLE MATERIALS 17

2.6.3 MR-COMPATIBLE ACTUATORS 19

2.6.4 MR-COMPATIBLE SENSORS 21

2.7 RESEARCH AND DEVELOPMENT IN MR-COMPATIBLE INTERVENTIONAL ROBOTIC SYSTEMS 22

2.7.1 MRI COMPATIBLE ROBOTS FOR BRAIN 23

2.7.2 GENERAL PURPOSE ROBOTS 27

2.8 SUMMARY 30

3 METHODOLOGY 32

3.1 INTRODUCTION 32

3.2 ISSUES TO BE OVERCOME 32

3.3 ROBOT SPECIFICATIONS 35

3.4 DESIGN OF A MRI COMPATIBLE NEEDLE MANIPULATOR 36

3.4.1 DESIGN CONSIDERATIONS 36

3.4.2 DESIGN SELECTED FOR THE MRI ROBOT 37

3.5 DESIGN CONCEPT 39

3.5.1 THE SUPPORTING STRUCTURE 40
3.5.2 THE ARM AND THE END-EFFECTOR 41
3.5.3 THE END-EFFECTOR 44
3.5.4 CONSTRUCTION 45
3.6 DENAVIT HARTENBERG PARAMETERS 45
3.6.1 FORWARD KINEMATICS 48
3.7 CONTROLLING THE ROBOT MANIPULATOR 53
3.7.1 CONTROL STRATEGY 53
3.7.2 OVERVIEW OF THE CONTROL HARDWARE AND SOFTWARE 56
3.7.3 HARDWARE COMPONENTS 58
3.7.4 CONTROL SOFTWARE 60
3.7.4.1 ALGORITHM 62
3.8 SUMMARY 66
4 RESULTS 68
4.1 INTRODUCTION 68
4.2 SYSTEM SETUP 68
4.2.1 DIMENSIONS AND RANGE 70
4.3 ANALYSIS OF MOTION 71
4.2.1 TEST SETUP FOR BACKLASH ERROR 72
4.3 ACCURACY OF THE SYSTEM 87
4.3.1 SETTING HOME POSITIONS 88
4.3.2 TEST STRUCTURES 97
4.4 SUMMARY 104
5 DISCUSSION 105
6 CONCLUSION & FUTURE WORK 108
6.1 OUTCOME OF THE WORK 108
6.2 EVALUATION 108
6.3 FUTURE WORK 110
BIBLIOGRAPHY 112
APPENDIX 116

LIST OF FIGURES

Figure 2.1: A master-slave system (da Vinci) for minimally invasive surgery…………..……9

Figure 2.2: MRI Scanner……………………………………………………………………….14

Figure 2.3: PneuStep motor…………………………………………………….……….....20

Figure 3.1: (a) Dimensions of MRI scanner;

(b) Available space around a human subject………………………………………….34

Figure 3.2: (a) Proposed 7 dof robotic system;

(b) Robot inside an MRI scanner……………………………………………….……..38

Figure 3.3: Seven degrees of freedom of the proposed manipulator……………………...40

Figure 3.4: Supporting structure with height adjustment and translations in axes Y & Z………………………………………………………………………………………41

Figure 3.5: Actuator mounted at the proximal end………………………………………42

Figure 3.6: Kinematic structure of the arm…………………………………………….....43

Figure 3.7: Side view of the end-effector with translational degree of freedom…………44

Figure 3.8: (a) Home position of the proposed robot; (b) Assignment of frames……......46

Figure 3.9: Control system employed for the robot manipulator…………………………54

Figure 3.10: Example of stereotactic path planning using 3D images to define the needle entry and target points………………………………………………………....55

Figure 3.11: System Layout………………………………………………………………….56

Figure 3.12: MultiFlex PCI 1000 Series Board Layout…………………………………...57

Figure 3.13: Assignment of Axis numbers…...58

Figure 3.14: (a) Electrocraft MAX-100 PWM Servo Drive; (b) TRD-S100-VD encoders; (c) Digiplan PK3 drive (d) Parker's Zeta 4 drive………………………………..59

Figure 3.15: Layout of the hardware components………………………………………...60

Figure 3.16: (Top) Dialogue box for entering skin entry point and target point;
(Bottom) Dialogue box for controlling the robot…………………………………...61
Figure 3.17: Algorithm to control the manipulator to bring its end-effector to
the prescribed insertion vector…………………………………………………...62
Figure 4.1: System Setup…………………………………………………………………69
Figure 4.2: Robot dimensions and range of motions……………………………………...70
Figure 4.3: Determining backlash error…………………………………………………...72
Figure 4.4: Limit switches for Axis 1……………………………………………………..73
Figure 4.5: Backlash Error for Axis 1……………………………………………………75
Figure 4.6: Backlash Error for Axis 2……………………………………………………76
Figure 4.7: Limit switches for Axis 5……………………………………………………..77
Figure 4.8: Backlash Error for Axis 5……………………………………………………79
Figure 4.9: Backlash Error for Axis 6……………………………………………………80
Figure 4.10: Limit switches for Axis 7……………………………………………………81
Figure 4.11: Backlash Error for Axis 7…………………………………………………...82
Figure 4.12: Limit switches for Axis 8……………………………………………………83
Figure 4.13: Backlash Error for Axis 8…………………………………………………...84
Figure 4.14: Home positions for axes 2 and 5…………………………………………....88
Figure 4.15: (a) Drawback of using a roller switch; (b) Home position for axis 6……….89
Figure 4.16: Hanging bob used to set home position for axis 1…………………………..90
Figure 4.17: (a) Home position for axis 1;
(b) Structure built to set home positions for axes 7 and 8…..91
Figure 4.18: Test Structures………………………………………………………………92
Figure 4.19: Determining maximum velocities for each axis…………………………......94
Figure 4.20: (a) Errors in needle positioning for test structure 1; (b) Final target point.....99

LIST OF TABLES

Table 2.1: Current Applications of Robotic Surgery…………………………………….11

Table 3.1: D-H Parameters……………………………………………………………48

Table 4.1: Backlash Errors for Axis 1…………………………………………………...74

Table 4.2: Backlash Errors for Axis 2…………………………………………………...76

Table 4.3: Backlash Errors for Axis 5…………………………………………………...78

Table 4.4: Backlash Errors for Axis 6…………………………………………………...80

Table 4.5: Backlash Errors for Axis 7…………………………………………………...82

Table 4.6: Backlash Errors for Axis 8…………………………………………………...84

Table 4.7: Resolution of the six axes…………………………………………………….86

Table 4.8: Backlash errors in terms of distances and degrees…………………………...87

Table 4.9: Errors for test structure 1……………………………………………………..99

Table 4.10: Errors for test structure 2…………………………………………………..101

Table 4.11: Errors for test structure 3…………………………………………………..101

Table 4.12: Errors for test structure 4…………………………………………………..102

Table 4.13: Errors for test structure 5…………………………………………………..103

CHAPTER 1

INTRODUCTION

1.1 OBJECTIVE

Commercial robots currently available are designed primarily to work in an industrial environment and are playing a key role in manufacturing industries. However their design is not suitable for applications in an MRI environment for the following reasons: (i) their geometry is not suitable for allowing the end effector to work within a very limited space. The work space within a MR bore is very limited. (ii) the material used in these robots is neither MRI compatible nor safe; and (iii) the commercial robot cannot directly interface with the data base produced by a MRI scanner. The focus of this thesis is to develop a manipulator that can work within the tight spaces of a MR bore and be controlled by an external personal computer (PC). PC's have the ability to read imaging data from a MRI scanner. The challenge of the thesis will be then to develop a manipulator with a geometry that can be easily modeled using mathematical tools and programmed using a PC. This thesis will develop all the electronic interfaces to control the arm and will be programmed to execute both forward and inverse mathematical manipulations. The software will be made user friendly. The work will also identify suitable materials for use that will eventually result in a MRI compatible robot. The robot manipulator will be built with standard material, some of which may not be MRI safe or compatible.

The system accuracy is targeted to be 2-3 millimeters which is sufficient for operations such as tumour therapy, tissue biopsy and particles placement. The needle insertion speed must be controlled at about 0.5 mm/s, meaning the needle must move slowly through the brain tissue without causing any damage.

1.2 MOTIVATION

According to the American Cancer Society, 22,020 men and women (11,980 men and 10,040 women) are estimated to be diagnosed with brain cancer in the United States in 2010 and an estimated 13,140 deaths will be attributed to it. About 4,030 new cases of childhood primary non-malignant and malignant brain tumours are expected to be diagnosed. Of these 4,030 new cases, an estimated 2,880 will be in children less than 15 years of age [1].

Brain tumours are the:

- second leading cause of cancer-related deaths in children under age 20 (leukemia is the first);
- second leading cause of cancer-related deaths in males up to age 39;
- second leading cause of cancer-related deaths in females under age 20; and
- fifth leading cause of cancer-related deaths in females of ages 20–39 [2].

The incidence rate of primary non-malignant and malignant brain and central nervous system tumours is 18.71 cases per 100,000 person-years. For all primary brain and other nervous system tumours, the incidence rate is 17.44 per 100,000 for males and 19.88 per 100,000 for females. The paediatric incidence rate is 4.71 per 100,000 person years, with a slightly higher rate in boys (4.75 per 100,000) than girls (4.66 per 100,000) [3].

In Canada, it is estimated that 2,600 new cases of brain cancer will be diagnosed in 2010 and an estimated 1,750 deaths will be attributed to it. The incidence rate of primary non-malignant and malignant brain and central nervous system tumours is 7 cases per 100,000 person-years [4].

The first and most crucial step in brain cancer treatment is diagnosis. Early detection and diagnosis of brain tumours is important to increase patients' chances for survival. Computed tomography (CT) and magnetic resonance imaging (MRI) scans are generally used in the discovery and location of brain tumours. The most accurate method for assessment of brain tumours is by biopsy. Biopsy is the removal of tissue to look for tumour cells. A pathologist looks at the cells under a microscope to check for abnormal cells. A biopsy can show cancer, tissue changes that may lead to cancer, and other conditions. A biopsy is the only sure way to diagnose a brain tumour. A biopsy can be performed as part of the surgery to remove a sample of the tumor, or as a separate diagnostic procedure [5].

For areas considered to be inoperable, the surgeon is often able to perform a needle biopsy through a small hole drilled into the skull called a burr hole. A narrow, hollow needle is inserted through the burr hole. Tumor tissue is removed from the core of the needle. Tumours buried deep in the brain sometimes cannot be approached safely. In those cases, a biopsy procedure involves using three-dimensional needle technique called stereo-taxic biopsy in which special imaging equipment guides the placement of a needle to allow cells to be drawn into the needle [6]. Stereo-taxic biopsy is a computer directed needle biopsy. The computer, using information from a Computer aided tomography (CT) or MRI scan, provides precise information about a tumour's location and its position relative to the many structures in the brain. Stereo-tactically-guided equipment is then moved into the burr hole to remove a sample of the tumour. Stereotactic biopsy procedures avoid patients from undergoing major surgical procedures; they are performed with local anaesthesia, and allow patients to return home on the same day or within 24 hours after the procedure [7].

As stated above, CT scan and MRI are the two imaging modalities used to demonstrate central nervous system lesions. Compared to MR imaging, the precise details of soft tissue (particularly the brain, including the disease processes) are less visible on CT scans. CT is not sensitive in detecting inflammation of the ménages - the membranes covering the brain. Thus, MRIs have now become the standard for finding tumours. The MRI doesn't use radiation and shows much more details than CT scans. The MRI uses radio waves to align the hydrogen atoms. Given the unique imaging capabilities of MRI, it is possible to diagnose and treat small, millimetre size tumours that cannot be detected by CT scans [8].

Where ever possible, brain tumours are removed through surgery. While many can be removed with little or no damage to the brain, others are located where surgical removal is difficult or impossible without destroying critical parts of the brain. Brain damage caused by surgery can lead to partial paralysis, changes in sensation (feeling), weakness and poor thinking. Thus it is imperative that a manipulator works inside the imaging machine, guided by real time images. This would insure that the surgical tool reaches the target tumour without damaging the neighbouring tissues [6].

There has been a substantial research in the development of in-bore MRI manipulation for diagnostic and surgical procedures in brain. Traditionally, neurosurgical navigation has relied on preoperative images and the assumption that anatomical structures of interest remain in the same position with respect to each other and the fiducial markers used for registration. However, during surgery, tissue deformation and shift disrupt the spatial relation between the patient and the preoperative image volumes. This results in localization errors. Intraoperative imaging techniques appear to be the best approach to counter these problems. Thus developing a manipulator that works inside a MRI machine guided by real time images would solve the problem of “brain shift” during the surgery [9]. The robots that have been developed for MRI compatible neurosurgery thus far have either been scanner specific, or have used custom made actuators that are not economically viable. A few designs have only been proposed in 3-D models and have not been built or tested for accuracy. Moreover, the accuracy of the prototypes that have already been built should be drastically improved if they are to find use for clinical applications.

1.3 CHAPTER OVERVIEW

This chapter has provided an overview of the background and issues related to the problem focussed in this thesis. It has also discussed as to how a MRI compatible robot would solve the problem of “brain shift” in neurosurgery. The objective of the research work is also outlined.

Chapter 2 presents an overview of the relevant literature pertinent to the focus of this thesis. It first gives information about MRI compatible technologies in the area of compatible materials, actuators and sensors, followed by a comprehensive literature review of MRI compatible robots for prostate, breast, heart and brain.

Chapter 3 presents the methodology followed for developing the neurosurgical manipulator. First, the design requirements of such a robot are discussed. Then the description of robotic system architecture, including the system operating procedure, hardware architecture and software architecture is provided.

Chapter 4 describes the experimental studies conducted to evaluate the performance of the robotic system. The results obtained from the accuracy tests are tabulated and analyzed.

Chapter 5 of the thesis provides discussion and conclusion resulting from this work. The performance of the robotic system is discussed first and then the application of this research work is presented.

Chapter 6 provides recommendations for future research topics and promising directions.

CHAPTER

2

BACKGROUND LITERATURE

2.1 INTRODUCTION

Research on robotics for medical applications started two decades ago and is very active today. The advantages are three-fold. 1) Robotic surgery can accomplish tasks that surgeons cannot, due to the high precision and repeatability of robotic systems. Also, robots are able to operate in a confined space inside the human body. Today, robots have found applications in heart, brain, spinal cord, throat, and knee surgeries at many hospitals around the world. 2) Robotics can be used effectively in medicine for diagnosis. Robotic diagnosis reduces invasiveness to the human body and improves the accuracy and scope of the diagnosis. 3) Robotics can also be used for providing artificial components to recover physical functions of human beings such as robotic prosthetic legs, arms and hands.

2.2 ROBOTS IN SURGERY

The most advanced aspect of medical robotics is surgical robotics, in which a robot actually performs surgery. It offers several advantages over conventional surgery, including greater spatial accuracy, increased dexterity and precision, reliability and reproducibility, measurable cost savings and ultimately better patient outcomes One reason surgical applications are progressing quickly is the large technology base that has been developed in robotics research in the past three decades. Results in mechanical design, kinematics, control algorithms, and programming that have been developed for industrial robots are directly applicable to many surgical applications [11].

2.3 ROBOTIC TECHNIQUES FOR SURGERY

Several trends in surgery are contributing to the growing acceptance of robots. Primary factors include the increasing emphasis on minimally invasive surgical techniques and the widespread availability of 3-D image data. Other robotic characteristics, particularly stability and the ability to work at small scales, provide the incentive for additional robotic applications. The following sections provide a quick review of literature in several application areas of robotics in medical field.

2.3.1 MINIMALLY INVASIVE PROCEDURES

Over the past decade, several surgical specialties have been rapidly transformed by adopting minimally invasive surgery. In this procedure, surgeons work through a set of three to five incisions approximately 1 cm in size. Long-handled instruments are used to grip and cut tissue within the body, and a video laparoscope provides a view of the internal operating field. This procedure avoids the long incision through the abdominal wall used in the conventional open procedure. As a result patients recover more quickly. Benefits include: less recovery time; less scarring; less pain [14]. The necessity for working through a few fixed incisions places severe limitations on dexterity in manipulation, and only a few procedures are possible with the current hand-held instruments. Lateral movement of the instrument shaft is not possible at the incision, which thus acts as a fulcrum, reversing the directions of the surgeon's hand motions at the instrument tip and varying the mechanical advantage as the instruments move in and out. Robotic manipulators promise to solve this problem. The challenge is to design devices with good dexterity and intuitive control that can be inserted through small incisions. One focus is the development of general purpose systems that can execute a range of procedures in thoracic and gynaecological surgery. These systems are often configured so that the surgeon sits at a console in the operating room and uses a master control manipulator that sends commands to the robots performing the surgical procedure as shown in Figure 2.1. Video images and force sensations are reproduced at the surgeon's console [13].

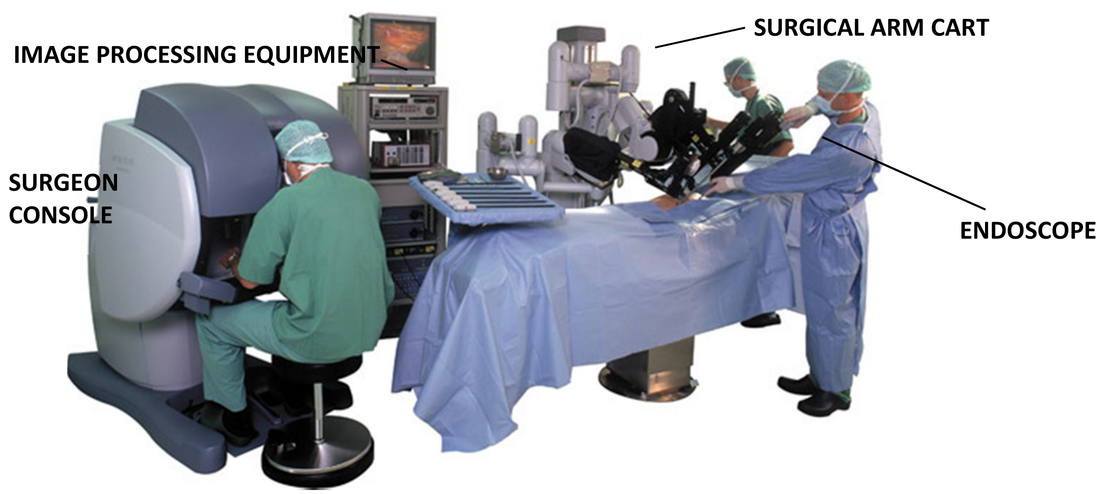


Figure 2.1 – A master-slave system (da Vinci) for minimally invasive surgery [15]

2.3.2 IMAGE-BASED PROCEDURES

Another stimulus for robotic surgery applications has been the development of non-invasive imaging techniques that include 3-D modalities such as magnetic resonance imaging and computed tomography (CT), and 2-D techniques such as ultrasonography, fluoroscopy, and conventional X-ray radiography. Because these images can reveal the precise location of pathologies, new computational and mechanical tools can guide treatments to the pathology while sparing the surrounding healthy tissue. A typical example is biopsy and removal of brain tumours. Preoperative magnetic resonance imaging can locate the tumour precisely within the skull. After opening the skull, a robot or human surgeon can guide instruments directly to the tumour, based on the image data. Collateral damage to brain tissue is minimized, and because brain structures can be distinguished in preoperative images, the instrument path can be planned to avoid critical regions [13].

2.4 SURGICAL APPLICATIONS

Surgical robotics was first introduced in the 1980s in orthopaedic surgery and neurosurgery, due to the ability of the respective organ systems involved to provide fixed landmarks and the precision required in their respective procedures. This involves manipulation of brain tissue in neurosurgery and precise cutting and coring of bone in orthopaedic surgery. Promising results in these fields subsequently led to the introduction of robotics in many other specialties including urology, otolaryngology, laparoscopic surgery and cardiac surgery as shown in Table 2.2 [11].

Current Applications of Robotic Surgery					
Orthopedic surgery	**Neurosurgery**	**Gynecologic surgery**	**Cardiothoracic surgery**	**Urology**	**General surgery**
Total hip arthroplasty: femur preparation	Complement image-guided-surgery	Tubal re-anastomosis	Mammary artery harvest	Radical prostatectomy	Cholecystectomy
Total hip arthroplasty: acetabular cup placement	Radiosurgery	Hysterectomies	CABG	Ureter repair	Nissen fundoplication
Knee surgery		Ovary resection	Mitral valve repair	Nephrectomy	Heller myotomy
Spine surgery					Gastric bypass
					Adrenalectomy
					Bowel resection
					Esophagectomy

Table 2.1 – Current Applications of Robotic Surgery [12]

The following section reviews the current state of research in surgical robotic applications, emphasizing neurosurgery.

NEUROSURGERY

Robotics was introduced in neurosurgical applications, due to the high precision that was required to localize and manipulate within the brain, and the relatively fixed landmarks of the cranial anatomy. The introduction of robot assistance into the surgical environment allowed the surgeon to work with greater accuracy at the microscopic level [16]. Neurosurgery was the first surgical specialty to use image-guided techniques, beginning with stereotactic frames that were attached to the patient's cranium before the imaging process and remained in place during surgery. The relationship between the frame and lesion observed in the image was used to guide the instruments within the brain. To enhance stability, accuracy, and ease of use, a number of robotic systems have been developed for image guided procedures over the past 15 years. One issue in image-guided neurosurgery is shifting of the brain during the procedure, which alters the spatial relationship between the preoperative image data and the anatomy of the patient. Solution to this problem is to perform the procedure inside an imaging system, which permits continuous monitoring of the anatomy and instrumentation. This requires robotic manipulators that are compatible with the imaging modality and space constraints such as MRI, CT and Ultrasound [13].

2.5 MRI GUIDED INTERVENTION

The standard modalities in interventional medicine are the X-ray-based fluoroscopy and computer tomography (CT) and ultrasound. Compared with those modalities, MRI provides several benefits that make it attractive for guiding interventions.

MRI is an intrinsically three-dimensional (3-D) modality that allows unrestricted selection of oblique 3-D or multi-slice imaging. Such capability would better suit the visualization of a procedure, without constraints or the need to manually reposition the patient or the imaging instrument as in the case of X-ray fluoroscopy. Recent technological advancements in modern MR scanners allow the dynamic “on-the-fly” adjustment of the imaging planes and volumes. This feature can be used to dynamically follow the movement of a MR-compatible interventional robotic device, thereby allowing a freehand dynamic guidance and control of the procedure, in a similar way with handheld ultrasound. Second, MRI does not use ionizing radiation and therefore is safer for the patient and medical staff. Another feature of MRI is that the quality of images is independent of the expertise of the operator, as is the case of ultrasound. In principle, MRI provides the means for single-modality and single-session-based procedures that integrate (a) diagnosis for identification of tissue pathophysiology and delineation of the targeted lesion; (b) guidance of the appropriate intervention, including positioning the tool and monitoring tissue-properties altering procedures, such as thermal ablations; and (c) assessment of the results of the interventional procedure. In view of the above benefits offered by MRI, major effort is devoted to the development and assessment of its clinical potential and merit in guiding interventional procedures [17].

2.6 MRI TECHNOLOGY

To understand the important aspects in designing a MRI compatible robot, it is useful to understand MRI. This section explains what MRI is and how it works. This will give insight into why designing a machine for the MRI environment is drastically different and has strict requirements.

2.6.1 MR-SCANNERS, MR-SAFETY, AND MR-COMPATIBILITY

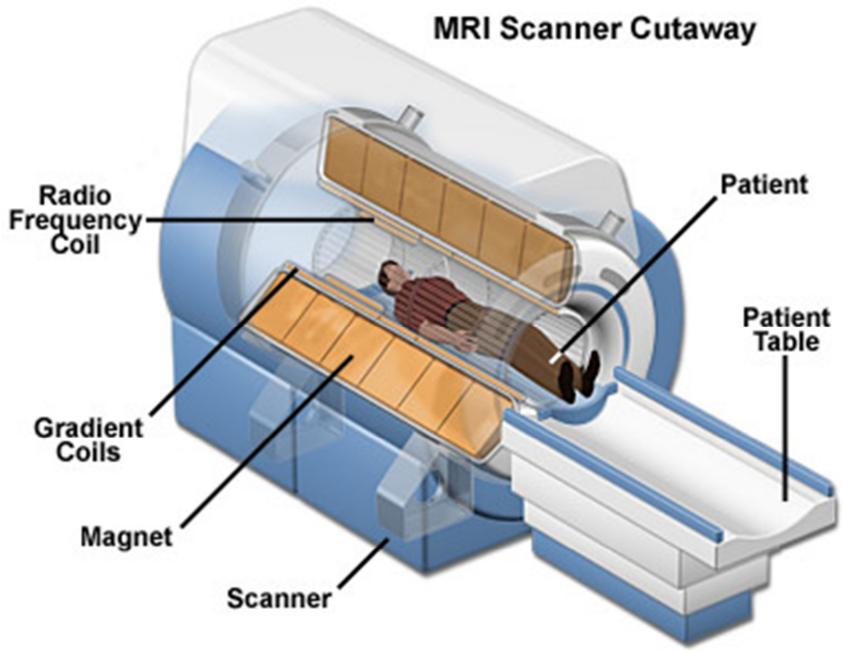


Figure 2.2 - MRI Scanner [18]

Magnetic Resonance Imaging (MRI) is a non-invasive procedure that allows the internal parts of the body to be imaged. This procedure is carried out by a MRI machine shown in Figure 2.2. MRI is used as a medical tool to assist in the diagnosis of illnesses and injuries and to allow the study of various parts of the body. The technology behind MRI is somewhat complex. MRI is based on altering the spin and orientation of an atom's nucleus and measuring the effects of this alteration. For the purpose of MRI, hydrogen is the atom of interest. This is because hydrogen is sensitive to magnetic fields and is present in both water and fat. The body is primarily composed of water and fat, making the measurement of these two materials particularly important. For simplicity, the process of MRI is typically described referring to the nucleus as a proton (since hydrogen atom has only one proton in its nucleus). The proton of an atom naturally has a spin and angular momentum which creates a magnetic field. In a normal environment the protons are randomly oriented and in turn their magnetic fields are also randomly oriented. This means that the magnetic field vectors of a group of protons sum to zero leaving no net magnetization. When a proton is placed in a magnetic field, it will begin to process about that magnetic field vector at a particular frequency. The next step is to expose the protons to a radio frequency (RF) pulse. The protons will absorb the energy of the RF pulse at the frequency known as Larmor Frequency unique to the nucleus of each element. This energy absorption will cause the proton to change state and thus orientation in the magnetic field. When the RF pulse is turned off, the protons will return to their original alignment emitting energy at the Larmor Frequency. The alteration in the alignment creates a signal which is detected by the scanner. The strength of the magnetic field determines the frequency at which the protons resonate. The damaged tissues can be detected as they take time in returning to the state of equilibrium. The parameters are set using the computer and contrast is created between different types of tissues. MRI with contrast is executed by injecting contrast agents in the body, in order to enhance the appearance of the blood vessels, and to detect tumours and inflammation in the body. The computer takes note of protons and the energy released by them to generate the image of the section of the body being examined [19, 20].

The development and use of MR-compatible devices are thus very challenging tasks owing to the nature of the modality, which uses high magnetic fields, fast-switching magnetic field gradients, and radiofrequency pulses, as well as being very sensitive to external noise. Standard clinical MR scanners use magnetic field strengths as high as 3.0

Tesla (1 Tesla = 10,000 Gauss). These strong magnetic fields result in extremely hazardous conditions: high forces can be exerted on ferrous apparatuses and interventional tools, making them harmful projectiles for the patient, medical personnel, and the instruments. Beyond being hazardous, ferromagnetic materials inside the MR scanner also affect the homogeneity of the main magnetic field, resulting in substantial loss of the signal. Susceptibility artifacts can also be induced by paramagnetic materials when their susceptibility is different from that of the tissue. In addition to the static main magnetic field, the MR scanner uses rapidly varying magnetic field gradients for spatial encoding during the imaging sequence. These gradient magnetic fields can induce electrical fields and currents (eddy-currents) inside conductive materials. These eddy-currents may alter the local homogeneity of the main magnetic field and severely affect the quality and linearity of MR images. Moreover, the MR scanners apply radiofrequency (RF) pulses to excite and manipulate the polarized spins for the collection of the MR signal. These RF pulses, as well as the magnetic field gradients, can heat interventional tools that contain conductive elements, such as needles or catheters, and become an additional risk to the patient [17].

MRI is very sensitive to electromagnetic noise and hence to prevent deterioration of the image signal to noise ratio (SNR), the entire scanner rooms employ highly shielded Faraday cages. Such electromagnetic noise may originate from electronic equipment needed for the operation of associated devices when they reside inside the MR scanner room. Even wires that pass from the outside to the inside of the scanner room can create problems because wires act as antennas that radiate electric noise. These environmental conditions require the careful selection of construction materials, actuation assemblies, sensors, and shielding of electronics [17].

A device is said to be "**MR safe**" if it has been demonstrated to present no additional risk to the patient when used in the MR environment (refers to the area within the 5 gauss line around the scanner). MR-safety does not ensure the quality of MR images, and an MR-safe device may or may not affect the images quality.

A device is said to be "**MR-compatible**" if it is MR safe; its use in the MR environment does not affect the image quality and it performs its function intended when used in the MR environment according to its specifications in a safe and effective manner [21].

The relevance of MR-compatibility and safety is not only limited to mechatronic systems but extends to any device exposed to the MRI environment. For example, electronic devices used in the MRI environment should be immune to the static magnetic field, the gradients, and the radiofrequency pulses of the scanner to avoid possible malfunctions, such as induced activations, deactivations, and damage [17].

2.6.2 MR-COMPATIBLE MATERIALS

Materials most often used in the construction of conventional robotic and mechatronic systems are ferromagnetic (e.g., carbon steel) because of their desirable mechanical properties, such as strength, rigidity, and machinability. However, these materials are, in general, not suitable for the construction of MR-compatible devices. Ferromagnetic materials are subject to strong magnetic forces and can become potentially dangerous projectiles if they are placed close to the MR scanner without being securely attached to a fixed structure. Another source of MR-incompatibility is the generation of eddy-currents inside conductive materials, such as aluminum, which may cause image artifacts [17, 22].

Materials suitable for MR-compatible devices are nonmagnetic and nonconductive. Combinations of plastic, ceramic, fibreglass, carbon fibre, and other composites have been extensively used for the development of MR-compatible systems. A main drawback associated with many of these materials is their limited structural stiffness, which can have a negative effect on the manipulability and accuracy of robotic and mechatronic devices. In many robotic/mechatronic systems developed for MRI applications, a limited number of metallic parts (such as aluminum, copper, and stainless steel) have often been incorporated into the otherwise MR-compatible structures. MR compatibility studies have demonstrated that small parts, such as screws, bearings, and gears, made of MR-incompatible materials do not present substantial problems or image artifacts as long as they are of small size and appropriately positioned relative to the imaged area [17, 22].

Materials suitable for MRI compatible devices are:

Non-ferrous metals: aluminum, beryllium, copper, lead, magnesium, nickel, gold, silver, platinum, palladium, tin, titanium, zinc, zirconium, etc;

Ferrous metals: 300 series stainless steel;

Special materials and plastics: carbon, advanced ceramics, glass, acetal, acrylic, alkyd, allyl, epoxy, flouroplastics, etc. [23].

2.6.3 MR-COMPATIBLE ACTUATORS

The MR-compatible interventional systems require appropriate forms of actuation. The commonly used electromagnetic actuators are, in general, not compatible with the MRI environment owing to their principle of operation. Therefore, alternative types of actuation have been considered and novel ones have been proposed for MR compatible applications [17]. Possible choice is hydraulic actuation given that hydraulic actuators are magnetically inert and can be used inside the MR scanner. The main advantages for hydraulic actuators are higher stiffness and durability than other forms of actuation. The down side is, hydraulic systems are highly nonlinear and have large parameter uncertainties due to volume change, friction, stick-slip, orifice area openings, load and supply pressure, etc. Hence, accurate position or force control is hard to achieve. Leakage of oil is almost unavoidable for hydraulic actuation systems, which may not be acceptable for the clinical personnel or patients in the MR scanner room because of the contamination with surrounding environment and possible chemical reactions involved. Pneumatics is another form of actuation used with MR devices that eliminates problems associated with hydraulic systems. Pneumatic systems are cleaner and operate at higher speeds compared with hydraulic systems. However, they are suitable only for relatively low-force applications and have limited stiffness owing to the compressibility of the air. Moreover, intermediate positions cannot be controlled. Another disadvantage is that age and use deteriorate the performance, which increases risk of bursting of the diaphragms [21]. Recently, a highly efficient and controllable pneumatic motor called ‘PneuStep’ (Figure 2.3) was introduced that is suitable for MR applications. ‘PneuStep’ uses the principles of stepper motor to achieve precise motion of the order of 0.050 mm, is simple in design and construction, and its operation is safe and fully MR compatible. This new motor uses pneumatics for actuation and optics for encoding, which are both decoupled from electromagnetism. The motor has been successfully tested in MR scanners of up to 7 Tesla without imager interference, artifacts, or loss of motion accuracy [24].

Figure 2.3 – ‘PneuStep’ motor [24]

With most of the systems developed so far, the preferred actuators have been the ultrasonic, piezoelectric motors (USM). Their operation is based on the piezoelectric phenomenon and the fact that motion is produced by the ultrasonic vibration of a piezoelectric ceramic when high-frequency voltage is applied. They are suitable for MRI applications because they are magnetically immune and they do not produce any magnetic fields either. Ultrasonic motors are bidirectional with a high torque-to-weight ratio, small size, and compact shape. A special feature of the USM is their high breaking torque, which allows a robotic system to maintain its current position and support its own weight when not actuated. The active elements of the ultrasonic motors are not affected by the magnetic field, but their encasings contain conductive material that result in image artifacts [17].

Many types of actuators are only MR compatible when they are away from the imaging area, such as the commercially available USM. To address the compatibility issue in most systems using this form of actuation, the motors remain outside the scanner and a motion transmission system is used to transfer the motion to the distant actuated points. Remote actuation can be implemented using drive shafts, belt or chain drive systems, cable-driven systems, linkages, etc. Performances limitations are typically associated with robotic/mechatronic systems that employ remotely actuated joints and are known to suffer from joint flexibility, backlash, and friction.

Other nonconventional types of MR-compatible actuation include actuators based on electroactive polymers (EAP). EAP are polymers that exhibit a change in size or shape when stimulated by an electric field. EAP can have several configurations, but are generally divided in two principal classes namely, dielectric and ionic. Electro-strictive polymer actuators, which belong to the dielectric category, have been investigated by Vogan et al. [25] who examined their application in reconfigurable imaging coils. Electro-rheological fluids (ERF), which belong to the category of ionic polymers, have provided an alternative way for generating resistive forces inside a MR scanner. An ERF-powered rehabilitation device that can apply controllable resistive forces on a person's hand has been developed and tested inside a MRI [26].

2.6.4 MR-COMPATIBLE SENSORS

Robotic devices for MRI applications also require special sensors, which are made of suitable materials and operate on principles compatible with the scanning environment, such as pneumatics, fibre optics or external camera systems. Charge coupled device (CCD) vision system for position measurements have been implemented and used for testing the positioning repeatability of a MR compatible manipulator inside a scanner. Custom designed incremental encoders for translational as well as rotational motion measurements have been proposed, which use glass grating patterns for counting motion and fibre optic cables for the transfer of signals to the remotely-placed optical components and circuitry. Commercial MR-compatible rotary and linear encoders are available and are based on fibre optic technology. Various robotic applications often require force sensory information. A six-axis force sensor for MRI applications has been developed using fibre optic components and a similar sensor was also used on a MR-compatible haptic device. In both cases, light is transmitted through a fibre cable to the remotely-located MR-compatible part of the sensor and continues through a returning cable. A physical displacement of the sensing element due to the applied force affects the passage of the light between the two cables. By measuring the intensity of the returning light, the size of the applied force can be derived [22].

2.7 RESEARCH AND DEVELOPMENT IN MR-COMPATIBLE INTERVENTIONAL ROBOTIC SYSTEMS

Over the past decade, several groups worldwide have developed MR-compatible interventional robotic systems. These works demonstrate the feasibility as well as the challenges associated with the development of such devices. All these systems use different kinematic structures and actuation mechanisms depending on the targeted applications and the available space in the scanner.

This section presents a comprehensive literature review of MRI compatible robots for brain and general purpose applications. This extensive review would help in determining the kinematics, materials and actuators best suited for the proposed MRI compatible robot.

2.7.1 MRI COMPATIBLE ROBOTS FOR BRAIN

One of the pioneering works in the field of MR-compatible interventional robotics is that of Masamune et al. [27], who developed a MR-compatible manipulator dedicated to neurosurgical applications. The device was designed to be mounted on the patient couch and above the head of the subject, providing six dof (degrees of freedom) for MR guided stereotactic needle biopsies. The stereotactic frame was based on an isocentric mechanism. This arc-style mechanism is useful in medical robots because it is easy to ensure the structural safety of the design. An XYZ stage localizes the position of the target, the rotation angle, and the latitude angle of the arc to decide on the orientation of the needle and the depth-control axis for needle insertion.The frame of the system was primarily made of the MR-inert polyethylene terephthalate (PET). Polyether ether ketone (PEEK) is used for the bonding screws. Other parts (e.g. linear feed screw, gear) that should be strong or that require precise fabrication are made of non-magnetic steel, brass (e.g., gear), aluminum and delrin (e.g., feed screw), or ceramics (e.g., bearing). This work provided the first insights into the challenges associated with MR-compatible mechatronics. Systematic assessment of the mechanical errors associated with each dof, as well as investigation of MR-guided targeting on phantoms, demonstrated an overall positioning accuracy of less than 3 mm. MR-compatibility studies in a 0.5 Tesla scanner revealed that when actuated, the ultrasonic motors substantially deteriorated the image quality and it was required to power off the motor drivers during MR scanning.

Another MRI compatible robot for neurosurgery was developed by Goldenberg et al. [28]. This robotic manipulator is based on master-slave system. The surgical arm is attached to a surgical table through a set of screws. It consists of a navigation module and a biopsy module. The navigation module is a six degree of freedom parallel mechanism consisting of a base plate, a moving plate, and six legs (struts). Six ultrasonic motors and six lead screws are used to provide required linear displacement for each leg. Each leg consisted of a universal joint, a spherical joint, and a prismatic joint which connects two joints together. The biopsy module is attached to a navigation module. The biopsy module provides proper mechanisms for gripping, advancing, and rotating the biopsy needle. It is fixed to the moving plate of the parallel mechanism. The advantages of using the parallel mechanism for the navigation module are as follows: (i) compact design (ii) light weight due to simple mechanical structure (iii) high rigidity with light structure; (iv) capability of selecting an arbitrary pivot point; and (v) high position and orientation accuracy.

Recently, the same authors have developed a hydraulically/pneumatically actuated MR compatible robot for MRI-guided neurosurgery [29]. The robot is similar in structure to the one discussed above. The surgical needle is held and advanced by the biopsy module. Again, the biopsy module is attached to the navigation module. The navigation module is a six degrees of freedom parallel mechanism consisting of a base and a platform interconnected through 6 legs (or struts). Six linear hydraulic actuators are used to provide required linear displacement for each leg. A locking mechanism is used to guide the needle as well as lock the robot at desired orientation. It is fixed to the base of the parallel mechanism through a connecting arm by screws. Again, all three units (the navigation module, biopsy module, and the locking mechanism) are held by a surgical arm. It consists of a base plate and a moving plate interconnected through 6 links.

A robot for deep brain stimulation has been developed by Cole et al. [30]. Deep brain stimulation (DBS) is a technique for influencing brain function though the use of implanted electrodes. The robot consists of three prismatic motions, two angular motions and a manual cannula guide. Parallelogram linkage was used for this robot. The benefit of parallelogram linkage is that it does not suffer from large wear surfaces, or high velocities that reduce precision associated with other sliding beam linkages. Piezoelectric motor was selected for use in this robot. In this robot, all electrical and metallic components are isolated from the patient's body. All linkages are made of high strength, biocompatible plastics including Ultem and PEEK.

A robotic system was developed by Koseki et al [31] dedicated to MR-guided manipulation of endoscopes for use in transnasal neurosurgical procedures. This four-dof device was specifically designed for use with open MR scanners and it took advantage of the available horizontal space between the vertical poles of such scanners. The kinematic structure of this robot incorporates a five-bar linkage mechanism for the transfer of motion to the remotely actuated joints. Like most of the MR-compatible devices, the system is driven by ultrasonic motors. To improve structural stiffness as well as reduce the construction cost, the system also includes MR incompatible materials. However, their size and location in relation to the imaging volume of the scanner were carefully considered to achieve MR compatibility. In this work, the accuracy, repeatability, and stiffness of the prototype system were examined experimentally. MR-compatibility studies inside a 0.3 Tesla open scanner confirmed that the scanner had no effect on the operation of the robotic system, and SNR phantom studies demonstrated that the presence of the manipulator had little effect on the image quality. However, the operation of the ultrasonic motors was found to cause considerable noise in the images and the manipulator was not actuated during MR imaging.

Another robot based on ultrasonic motors was developed by Hong et al. [32]. This robotic system automatically and interactively aligns and orients the surgical needle throughout the intra-operative MRI guided neurosurgical procedures, such as biopsy and brachytherpy. The robot consists of a DELTA parallel mechanism with 3 dof and a serial manipulator with 2 dof, which functions to align and orient the needles respectively. DELTA mechanism is characterized by three identical kinematical chains, symmetrically placed at 120º between each other, which drive a moving platform with respect to a fixed base. The all motion stage was actuated by non-magnetic (piezoelectric) ultrasonic motors (USM, with rotary encoder units), harmonic-drive reducers and ceramic bearing. The harmonic-drive reducers and ceramic bearing were custom made for this application. The harmonic drive reducers reduce the velocity of motors to safe level. The electrical signals of rotary encoder are optically transmitted to the outside of operation room. All parts of the robot were made from paramagnetic materials. The insertion stage and orientation stage were made from a titanium alloy. For frames of Delta parallel manipulator polycarbonate resin was used. Only titanium alloy or brass screws and bolts were used.

The most extensive work on MR compatible robot devices for brain was carried out by Sutherland et al. [9, 33]. Sutherland developed a neurosurgical robot called NeuroArm. It performs image-guided biopsies or haptically controlled microsurgery. NeuroArm has a unique MR-compatible platform that can be completed with certain tools for a particular neurosurgical (micro neurosurgery) and stereotaxic operation. NeuroArm consists of two manipulator arms, one digitizing arm, and a field camera mounted on an adjustable mobile base. The two manipulators mimic the two arms of the primary neurosurgeon. Each manipulator has seven dof, including one dof for tool actuation. Operating under a master-slave control system, the manipulators replicate the surgeon's movements of the hand controllers at the workstation. During surgery, after the perioperative images have been obtained, the robot is positioned next to the operating room table, where the primary neurosurgeon would stand, and locked in place using a wheel brake. For MR compatibility, the manipulators are made from titanium and two plastic polymers, polyetheretherketone and polyoxymethylene. The end effector of each manipulator contains custom-made titanium multi-axis force and torque sensors, located directly between the tool and the end effector. These force sensors provide haptic feedback to the hand controllers at the workstation in three translational dof.

2.7.2 GENERAL PURPOSE ROBOTS

The first versatile surgical robot with MRI compatibility was proposed by Okamoto et al. [34]. This surgical robot system has been designed to work in vertical magnetic field type Open-MRI units. This manipulator can be used for various surgeries for example heart surgery, liver surgery, gallbladder surgery and so on. The type of manipulator is SCARA (Selective Compliant Assembly Robot Arm), which is suitable for operation in the vertically confined MRI gantry space because the moving range of the SCARA-arm is limited only to a horizontal direction. Although this manipulator is composed of mainly aluminum (low magnetic materials) as a prototype, the authors plan to fabricate a completely MRI compatible manipulator composed of non-magnetic material such as titanium and PEEK (Polyether ether ketone) instead of aluminum. The manipulator is actuated by ultrasonic motors located outside the MRI gantry and transmitted by timing belts to avoid the influence of the magnetic field. The timing belts transmit high torque with very low backlash. They are made of rubbers to be MRI compatible.

Another general purpose MR-compatible interventional system was developed by Tsekos et al. [22]. The device has seven dof and consists of a Cartesian positioner with three orthogonal dof located in front of the MR scanner and a robotic arm that is deployed inside the scanner. The arm has three rotational dof and a linear one for the insertion of interventional tools. For MR compatibility, actuators are placed at the proximal end of the arm, i.e., outside the scanner, and motion is transferred to the distant joints using a system of drive shafts and universal joints for the through-joint transmission. MR-compatibility studies included the assessment of the signal to noise ratio (SNR) and contrast to noise ratio (CNR) of MR images on phantoms and in vivo inside a 1.5 Tesla cylindrical scanner. The phantom studies demonstrated that the noise induced by the operation of the ultrasonic motors can be substantially reduced by enclosing the electronics inside a Faraday cage and shielding the wires.

As mentioned above, the only currently commercially available MR-compatible interventional robotic system is InnoMotion (Innomedic GmbH, Herxheim, Germany). This system has been developed by Melzer et al. [35]. The system is a robotic arm that can assist in percutaneous interventions inside MR scanners. It has six degrees of freedom and was developed for a variety of applications, including spinal procedures for pain therapy, tumour therapy, and biopsies. The kinematics consists of an arm that is pneumatically driven in 5 dof The robot arm is attached to a 180 degree orbiting ring that is mounted to the patient table of the scanner, and can be manually prepositioned into the orbit region, at the angles 0, 35 and degree 67, on either side of the orbit ring, depending on the region of interest (e.g., spine, liver, kidney, and breast). The arm is fixed with a spring-loaded bolt and secured with a screw. Its actuation was based on specially developed MRI compatible pneumatic linear cylinders.

Another general purpose MR robotic system was introduced by Tajima et al. [36].This unique prototype master-slave system is designed for use with the all-around open scanners. The manipulator has six degrees of freedom so that its arm tip can assume any configuration. Three translational dof are located on its base and three dof on its arm component, which approaches the surgical site inside the scanner. The base of the manipulator is primarily made of aluminum and the arm of plastic materials. The presented studies included assessment of its manipulability as well as its MR compatibility by performing SNR studies on a phantom placed inside the head coil of a 0.3 open scanner. With the manipulator touching the phantom, a slight artifact and minor effect on the image were observed, as well as approximately 10% reduction in SNR when the manipulator was in motion.

2.8 SUMMARY

Robotic technology is enhancing surgery through improved precision, stability, and dexterity. In image-guided procedures, robots use magnetic resonance image data to guide instruments to the treatment site. Minimally invasive procedures use remotely controlled robots that allow the surgeon to work inside the patient's body without making large incisions. Specialized mechanical designs and sensing technologies are needed to maximize dexterity under these access constraints. Robots have applications in many surgical specialties. In neurosurgery, image-guided robots can perform biopsy brain lesions with minimal damage to adjacent tissue. Over the years, many authors have developed robots that work inside the MRI machine, guided by real time images. This is expected to solve the problem of "brain shift" during surgery. Developing such robots is a challenging task owing to the strong magnetic field and the limited space inside MRI machine. To counter these problems researchers have developed MRI compatible materials, actuators and sensors and proposed several kinematic structures to fit inside the MRI bore. Although a lot of research has been done on this subject, there are some issues that still remain unaccounted for. The proposed work is intended to address the following issues:

1) **Scanner Specific:** Most of the MRI compatible robots developed so far have been scanner specific which means the robot can be used either inside a closed cylindrical scanner or in an open scanner. Although the proposed robotic system will be primarily designed for a closed cylindrical scanner (since closed scanners provide better resolution images), it can be easily adapted to be used inside an

open scanner by varying the height of the robot to fit inside the scanner. This feature would be beneficial for sites that already have an open scanner installed in place.

2) **Custom made actuators:** Several researchers have developed and incorporated custom made actuators in their designs. Modifications have been made primarily in pneumatic and hydraulic actuators to find their use in MRI compatible robots. They have either modified the design of these actuators or suggested complicated control system to control the intermediate positions in these actuators. Developing such robots would not be economically viable. Commercially available ultrasonic motors provide an option. The robot proposed in this research work will be designed for using any type of electric actuators.

With most of the systems developed so far, the preferred actuators have been the ultrasonic piezoelectric motors (USM). However, researchers have found that, while the active elements of the ultrasonic motors are not affected by the magnetic field, their encasings contain conductive material that result in image artifacts. To address the compatibility issue, this research work proposes a design which will allow the placement of the actuators outside the magnetic bore. With this arrangement, any type of electic motors as actuator can be employed.

CHAPTER

3

METHODOLOGY

3.1 INTRODUCTION

Stereotactic surgical intervention uses three dimensional coordinates to locate small targets inside the body. Stereotactic neurosurgery requires precise positioning of surgical apparatus using predetermined anatomical information. These include biopsy, injection of drugs, laser surgery, radiation treatment, and positioning of electrodes [27]. This chapter presents the details of the design and development of a robot for potential application in a MRI guided environment to accurately guide a needle to a precise location within the brain.

3.2 ISSUES TO BE OVERCOME

Two main difficulties in using MRI-guided robot for interventional purposes are the limited accessibility to the patient and the associated magnetic fields involved in the operation of a MR scanner. The standard, high field superconductive scanners (1.0–3.0 T) provide the highest MR signal and best field homogeneity, but their shape is far from ideal for interventional purposes because of the limited accessibility to the patient. The low-field C-arm or open MRI scanners (0.2–0.5 T and more recently 1 T), as well as the unique 'double-donut' design (0.5 T), offer the best accessibility to the patient. However these scanners typically have a space height of 350 mm and the available space around a human subject inside a MRI scanner for surgeon and robot is hence significantly decreased. The magnetic fields within the MR environment pose a major challenge for robots. Conventional materials, actuators, and sensors all interfere with the static magnetic field, and radio frequency (RF) signals emitted and detected by the scanner. The strong static magnetic field of 0.3-3T in current MRI scanner will attract any ferromagnetic materials (missile effect) working in MRI environment, and affect the safety of patient, surgeon and MRI scanner [37].

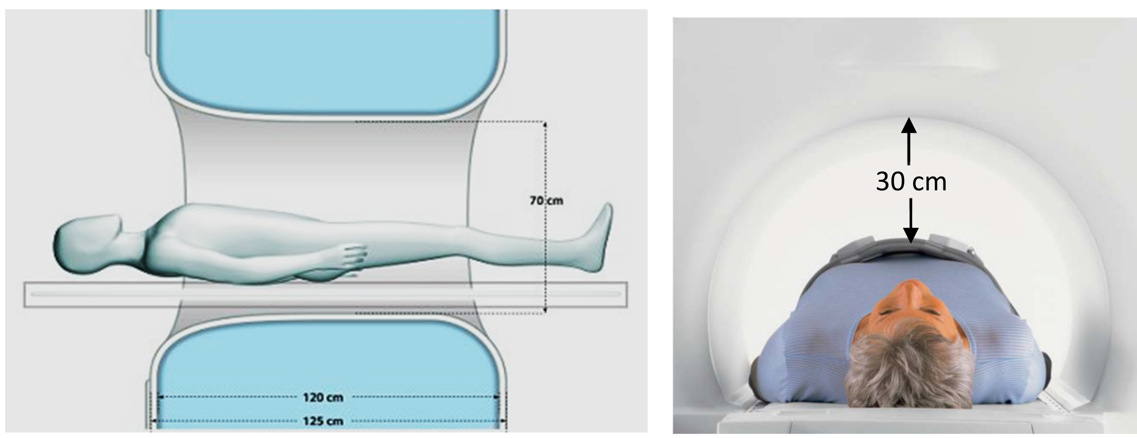
Thus, developing MRI-guided robot-assisted surgery support involves the following issues:

1) A strong magnetic imaging field.

To prevent imaging from adversely affecting robot operation and vice versa, the robot must be MR-compatible, i.e., use MR-compatible materials, actuators, and sensors. Combinations of plastic, ceramic, fibreglass, carbon fibre, and other composites have been extensively used for the development of MR-compatible systems. Even though material selection will not be the focus of this research work, suitable materials will be identified for MRI compatible robots. The focus will be on evolving a novel geometry and placement of actuators at safe distance to avoid interference with MRI images. This is critical since commercial motors available to work inside MRI environment (for e.g. ultrasonic motors) are MRI safe at best but not MRI compatible.

2) Robot size and precision.

The bore of a typical MRI is 70cm in diameter and 125cm in length [38]. There is a cradle that the patient lies on, which reduces the diameter to about 55cm. The limited accessibility to the patient is demonstrated in Figure 3.1 and shows the available space around a human subject (height toe to head: 1.80 m, weight: 82 kg) inside a MRI scanner with a diameter of 70 cm. These results illustrate that the available space inside the MRI scanner is highly limited. The limited space inside the MR scanner dictates a manipulator design which utilizes the free space above the patient.



(a) (b)

Figure 3.1- (a) Dimensions of MRI scanner [39]; (b) Available space around a human subject [38]

Needle positioning precision must be 1-2mm of the target point and needle speed must be controlled at about 0.5mm/s, meaning the needle must move slowly so that brain tissue is not damaged [40].

3) Involuntary patient movement during surgery.

Even anesthetised patients may move unconsciously during an operation, so relative positioning of the patient’s head and the robot must be prevented, i.e., the head must be fixed in place [40].

3.3 ROBOT SPECIFICATIONS

Based on the aforementioned considerations and basic material research, the following robot specifications were decided:

- The mechanical body of the robot should be mainly fabricated from aluminum, Teflon, and 300 series stainless steel. Parts that should be strong or that require precise fabrication can be made of non-magnetic steel and aluminum. Other parts can be constructed of polymers like Teflon, PEEK and Delrin. Since this work is a proof-of-concept of design and control of the proposed robot, materials which are readily available have been used. Not all of them will be MRI compatible.
- Similarly, the actuator best suited for MRI compatible robots are nonmagnetic ultrasonic motor (USM) using an optical rotary encoder at the axis to detect positioning information. This again has been substituted by available servo and stepper motors.
- The robot uses the free space above the patient. The MAGNETOM Espree provides 30 cm of space above the patient's face and hence the vertical range for the robot would be less than 30 cm.
- Needle insertion is assumed to be done by a surgeon, after the robotic end-effector has been aligned to the proper orientation. Technically, the robot can perform the insertion; however, the current plan reserves this task for surgeon owing to ethical and legal considerations. It is assumed that incisions have already been made in the brain for the needle to go through and remove a sample of tissue to test for tumor cells. So, the needle will work with soft tissues only.

3.4 DESIGN OF A MRI COMPATIBLE NEEDLE MANIPULATOR

Once the robot specifications are decided, the next step is to study the potential designs for the robot model. In this section, the criterion upon which the system is based is discussed first, and then an overview of its components is presented.

3.4.1 DESIGN CONSIDERATIONS

The three important design considerations for a MRI compatible robot are:

1) The robot should have the required minimum number of degrees of freedom to align the needle to the required position and orientation before insertion into the brain. Lesser the number of degrees of freedom, easier would be the control. From the knowledge of robotics, at-least 6 degrees of freedom would be required to position a tool at a given point in space at a specified orientation.

   Many authors in literature have designed 5 degree-of-freedom (dof) systems, and assumed the end-effector to have the remaining dofs. By using this strategy various surgical tools can be utilized as end-effectors. Similarly, in this work the last degree of freedom (needle insertion) is independent of the other robot movements. This gives the surgeon flexibility to use as an end-effector; a biopsy needle, electrode or syringe for injection of drugs.

2) It is preferable that majority if not all degrees of freedom of the robot do not work against gravitational forces during the movement from one position to another. In case of power failure/malfunction of the motors the links that work against gravity

would collapse and seriously injure the patient. Thus, it is imperative that safety features are built into the robot design.

3) Robot design should be such that all the actuators are placed outside the MRI bore. It has been found in many robot designs that even placing MRI compatible actuators (e.g ultrasonic motors) inside the MRI bore causes image artifacts. Subsequently, it has been concluded that the active elements of these motors are not affected by the magnetic field but the encasings contain conductive material that result in image artifacts. Therefore, it is essential that the actuators are placed outside the MRI bore, with some arrangements made to transmit the motion from them.

3.4.2 DESIGN SELECTED FOR THE MRI ROBOT

In this section, the design of a PC based neurosurgical manipulator for performing procedures with the patient located inside standard cylindrical scanners is presented. The manipulator has seven degrees of freedom as shown in Figure 3.2 (a) and is composed of a base unit, which resides outside the scanner and hosts all the mechatronic subassemblies, and an articulated arm which extends inside the gantry and manoeuvres above the patient to facilitate the positioning of an interventional tool along a user defined trajectory as shown in Figure 3.2 (b).

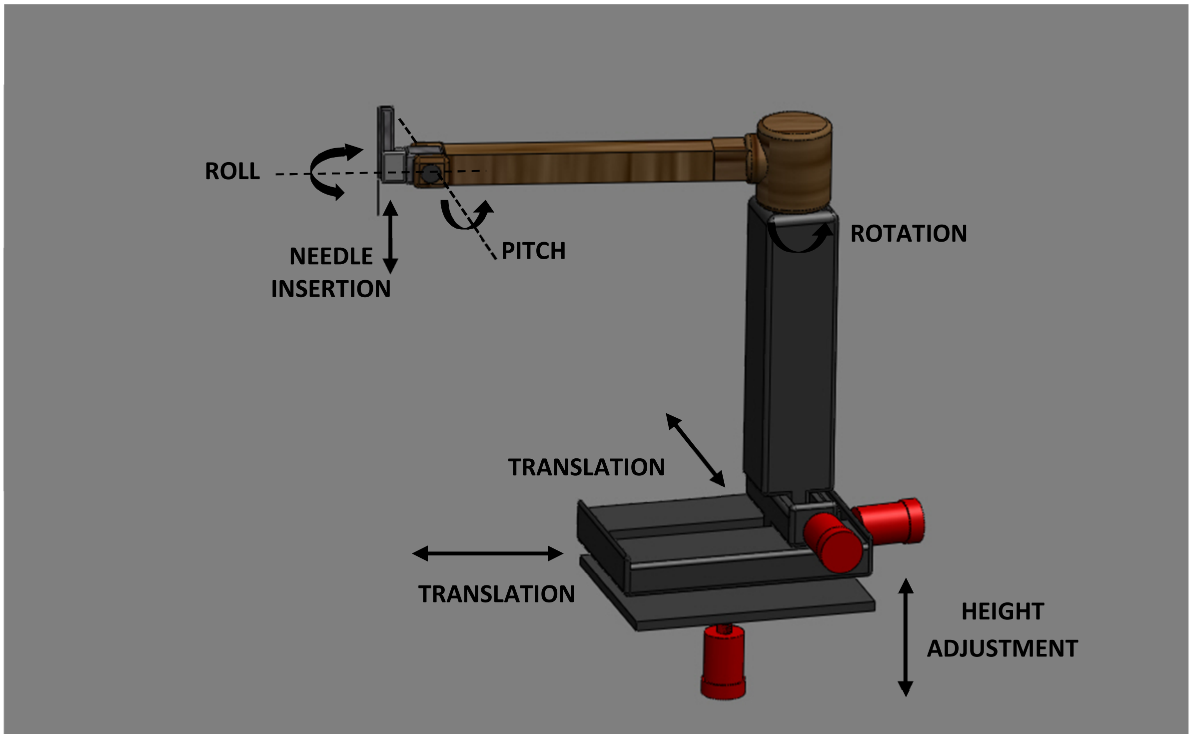


Figure 3.2 (a) Proposed 7 dof robotic system

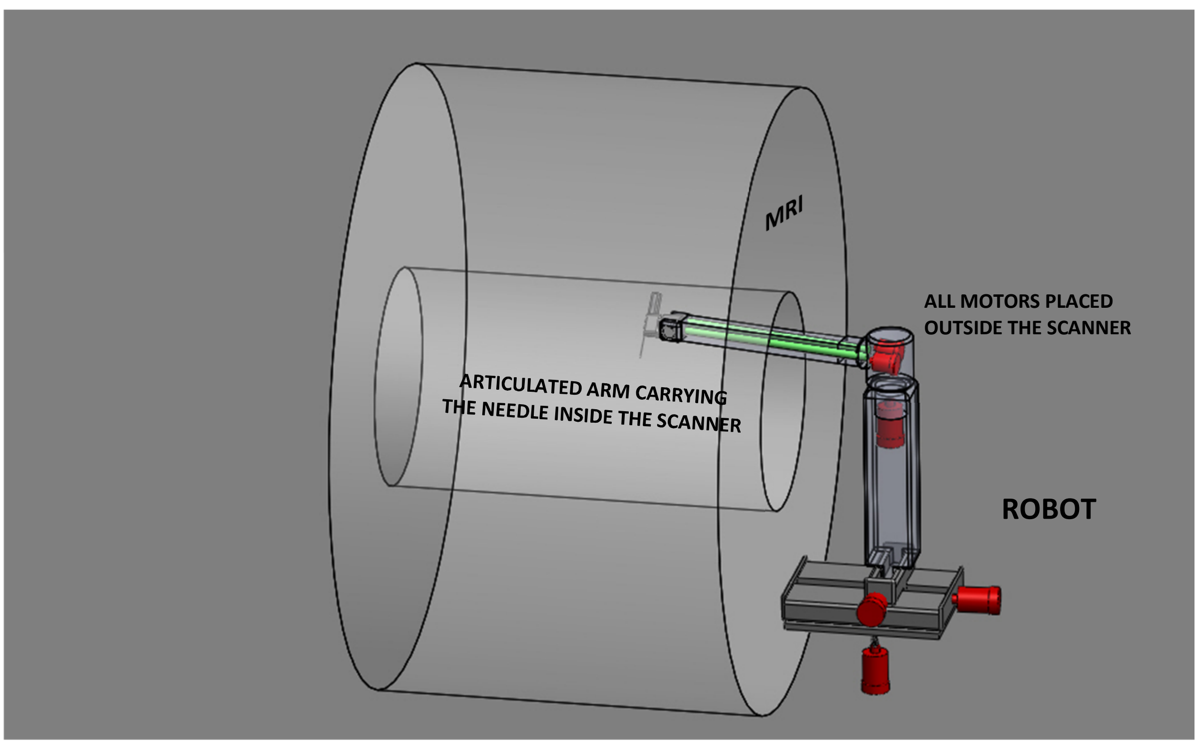


Figure 3.2 (b) Robot inside the MRI scanner

The development of the system adhered to three primary design criteria as discussed before. The robot operates safely with regard to the subject inside the confined spaces of a standard 70-cm horizontal bore magnet and the vertically limited space between the poles of an open MR scanner. The robotic device has sufficient dof to manoeuvre a probe to set an insertion path for a defined procedure. The motors can be safely placed outside the MRI bore. Finally, the fabrication and operation of the device is simple and intuitive.

3.5 DESIGN CONCEPT

The robot consists of four main parts: (1) the supporting structure, (2) a Cartesian positioning stage which is directly attached to the supporting structure, (3) a robotic arm which is attached to the Cartesian positioner and extends inside the bore of the scanner to reach the patient's head, and (4) a end-effector attached at the tip of the arm. The system is provided with a total of six dof: three translational dof are provided by the Cartesian positioner and three rotational ones by the arm as shown in Figure 3.3. The end-effector has an additional dof used for the advancement of a needle towards a target.

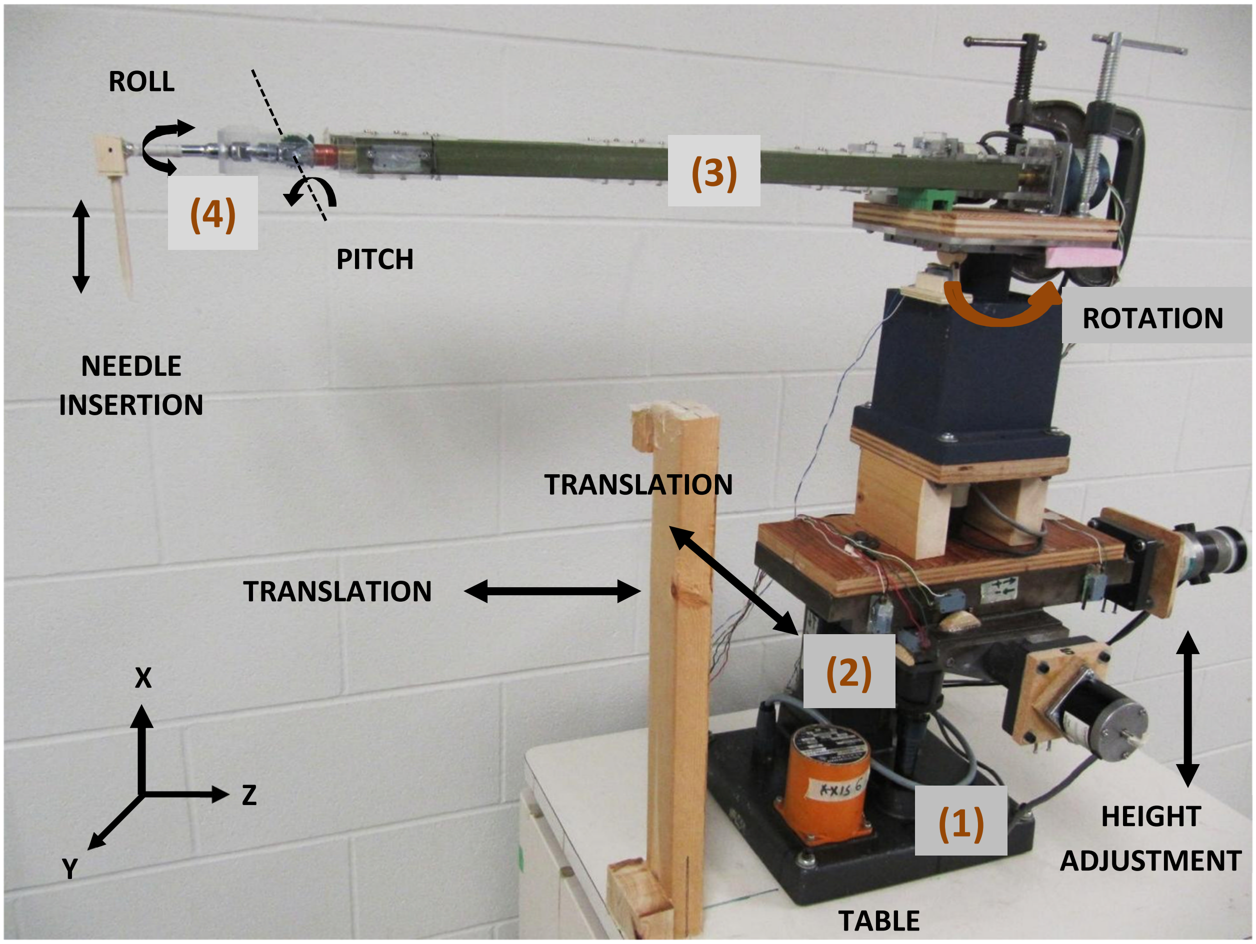


Figure 3.3 – Seven degrees of freedom of the proposed manipulator

3.5.1 THE SUPPORTING STRUCTURE AND THE CARTESIAN POSITIONING STAGE

The robot manipulator is mounted on a supporting structure which sits on a table as shown in Figure 3.3. It allows the placement of the system at the back of the MR scanner embracing the patient's couch. The Cartesian positioning stage has three orthogonal dof denoted as x, z and y in Figure 3.3.

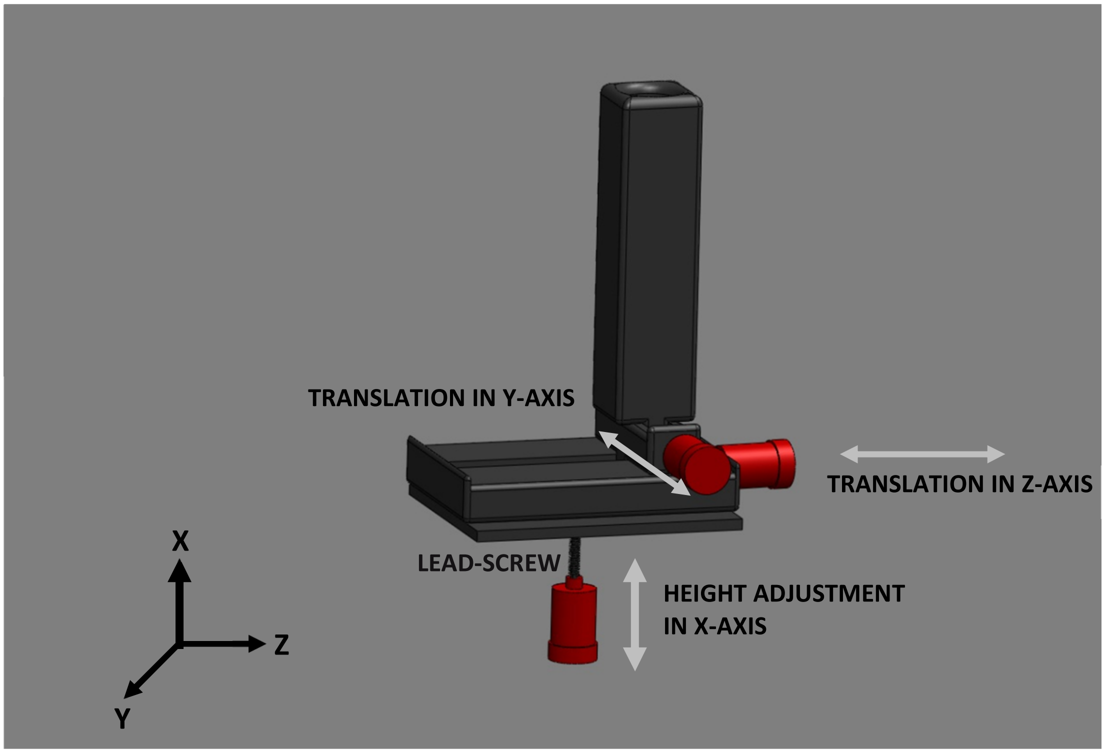


Figure 3.4 – Supporting structure with height adjustment and translations in axes Y & Z

The three linear translation motions (x, y & z) are carried out by low-friction sliding bearings and driven by lead-screw mechanism. The Cartesian positioner (shown in Figure. 3.4) provides for the global positioning of the arm and carries the actuators and transmission lines. Counterbalancing weights are also added which effectively neutralize the weight supported by the vertical motion system. Consequently, a low-power motor suffices for the motion along the x-axis. It also reduces the loading and associated frictional forces of the lead-screw system along the horizontal motion in the z-axis.

3.5.2 THE ARM AND THE END-EFFECTOR

The arm shown in Figure 3.3 is mounted on the Cartesian positioning stage with a rotational joint. This provides an additional rotational degree of freedom at the arm joint itself. The kinematic structure of the upper arm is shown in Figure 3.5 and 3.6. It consists of two interconnected links with two rotational joints. The corresponding rotations are represented by θ_5 and θ_6 respectively. The degree of freedom at the arm joint is θ_4.

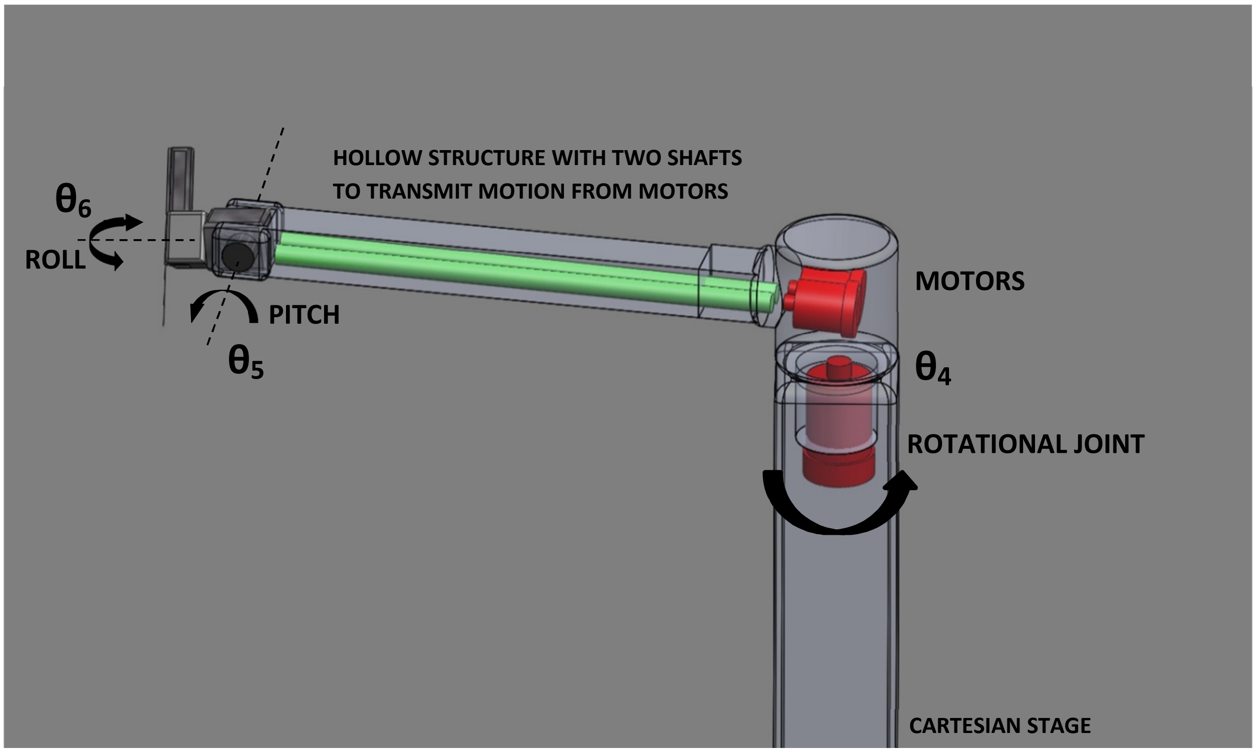


Figure 3.5 – Actuator mounted at the proximal end

The overall length of the arm was determined by the need to perform procedures in the vicinity of the iso-center of the scanner. In that area the main magnetic field is homogeneous and the magnetic field gradients are linear. This should result in better signal-to-noise ratio and spatially undistorted images. For MR magnetic compatibility reasons, the arm is designed with the actuators mounted at the proximal end which resides outside the scanner as shown in Figure 3.5.

The motion is transferred from the motors to the joints through shafts which run across the length of the arm, inside the hollow structure of the links. The drive shafts power a universal joint, installed at the wrist joint of the arm to allow the through-joint transmission of torques as shown in Figure 3.6. The motion of the drive shafts is finally transferred to the actuated joints via bevel gears.

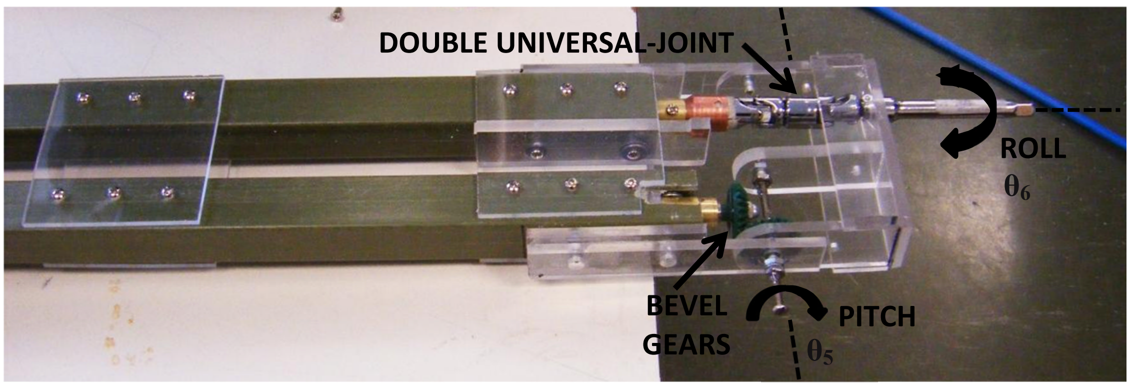


Figure 3.6 – Kinematic structure of the arm

A double universal joint is used for the wrist joint as shown in the figure above. This joint would overcome the limitations of a single universal joint commonly used for through-joint transmission of torques. The drawbacks associated with the use of single u-joint are the restricted misalignment angle that can be accommodated and the fact that the motion transmission is not uniform. Even though one revolution of the driver shaft corresponds to a complete revolution of the follower, the output velocity fluctuates depending on the misalignment angle between the two shafts.

3.5.3 THE END-EFFECTOR

The end-effector is attached to the tip of the arm as shown in Figure 3.7. It is designed to hold and insert a needle, with the needle holder moving along a rail.

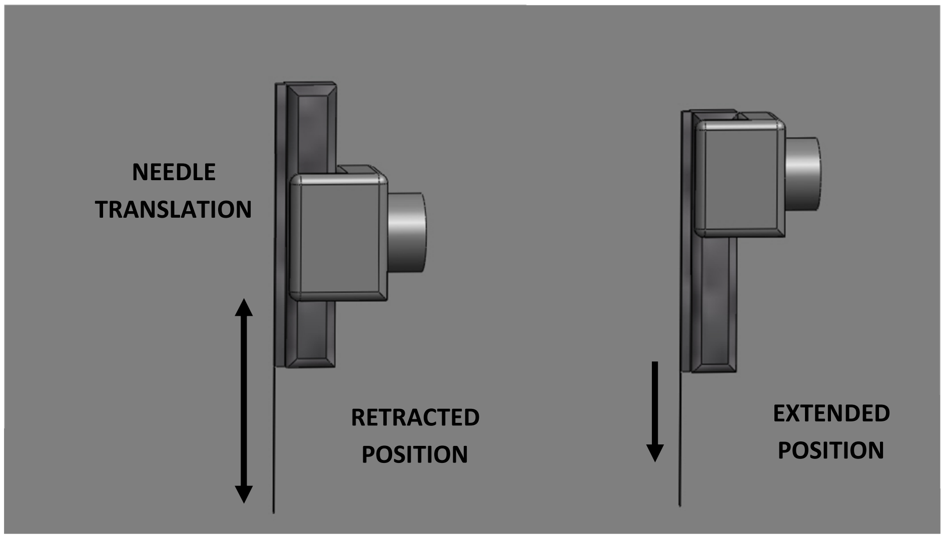


Figure 3.7 – Side view of the end-effector with translational degree of freedom

The insertion and removal of the needle is controlled using a hand-held manually operated device. Another issue to be considered here is that the commercially available MRI compatible needles are not visible in the MR images. A special marker has to be attached at the front of the end-effector where the tip of the needle is. This is particularly essential when the robot is used inside the MRI scanner. The marker will have pockets filled with MR contrast agent (3% Gadolinium solution-Gd) which appear as bright lines in the MR images to specify the current location and orientation of the needle and allow image-guided targeting.

3.5.4 CONSTRUCTION

In this work, a prototype of the proposed neurosurgical system with its potential application in MRI environment has been fabricated. Ideally MRI compatible materials and actuators should have been used, but since this work is a proof-of-concept of the design and control of the proposed system, readily available materials and actuators have been used to build the robot manipulator.

The support structure and the frame of the Cartesian positioner (lead screws) are made of cast steel. The arm of the robotic system is built with fibreglass and nylon. The bevel gears used at the end of the arm are made of polyacetal commonly known as Delrin. The double u-joint used is made of steel.

The motors used in the system should preferably have been ultrasonic motors. A working model with commercial servo and stepper motors has been constructed.

3.6 DENAVIT HARTENBERG PARAMETERS

In order to find the position of wrist point with respect to the base of the robot, a transformation model from tool tip to the base of the manipulator needs to be developed. This was accomplished using Denavit Hartenberg transformation [41]. The links are numbered starting from the immobile base of the arm, which is called link 0. The first moving body is link 1, and so on, all the way to the free end of the arm, which is link *n*. For assigning the frames as per this convention, first a home position for the model has to be decided. Usually the home position is set at a point where all the joint variables are zero. The home position for the proposed MRI compatible robot is set as shown in Figure 3.8 (a) below:

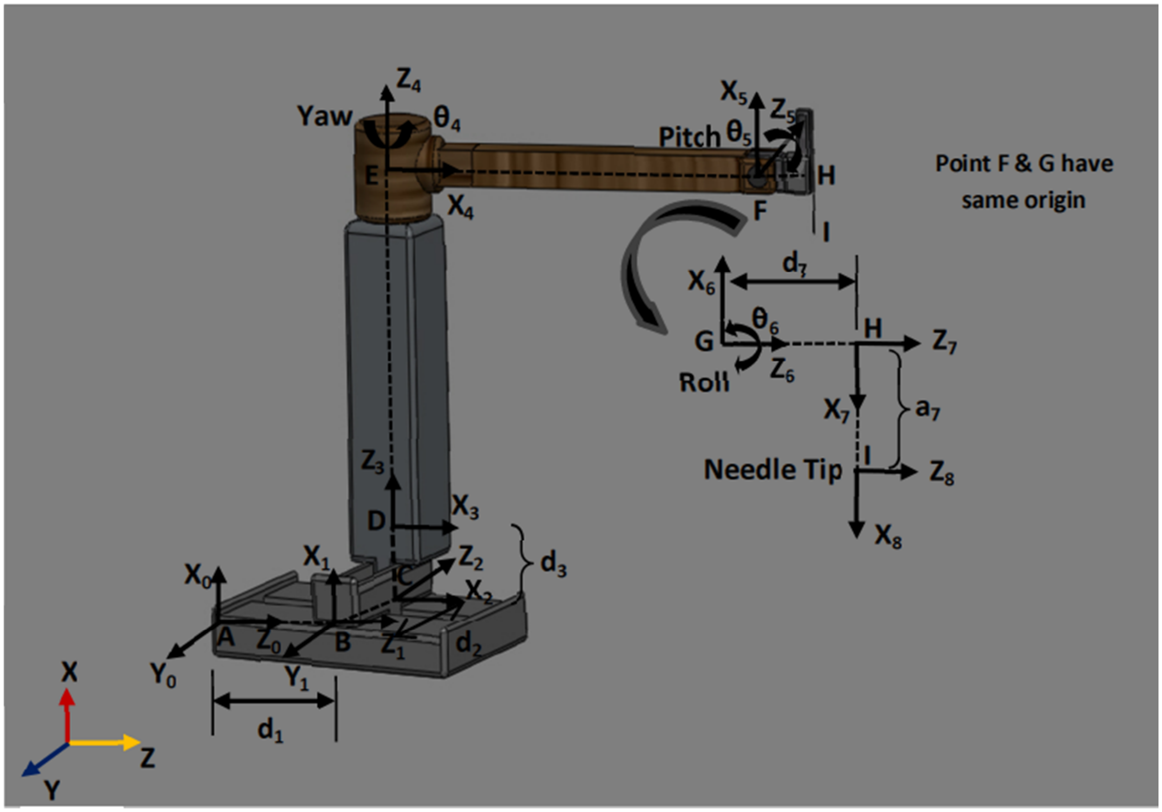


Figure 3.8 - (a) Home position of the proposed neurosurgical robot

Based on the Figure 3.8 (a) above, frames are assigned as shown in Figure 3.8 (b).

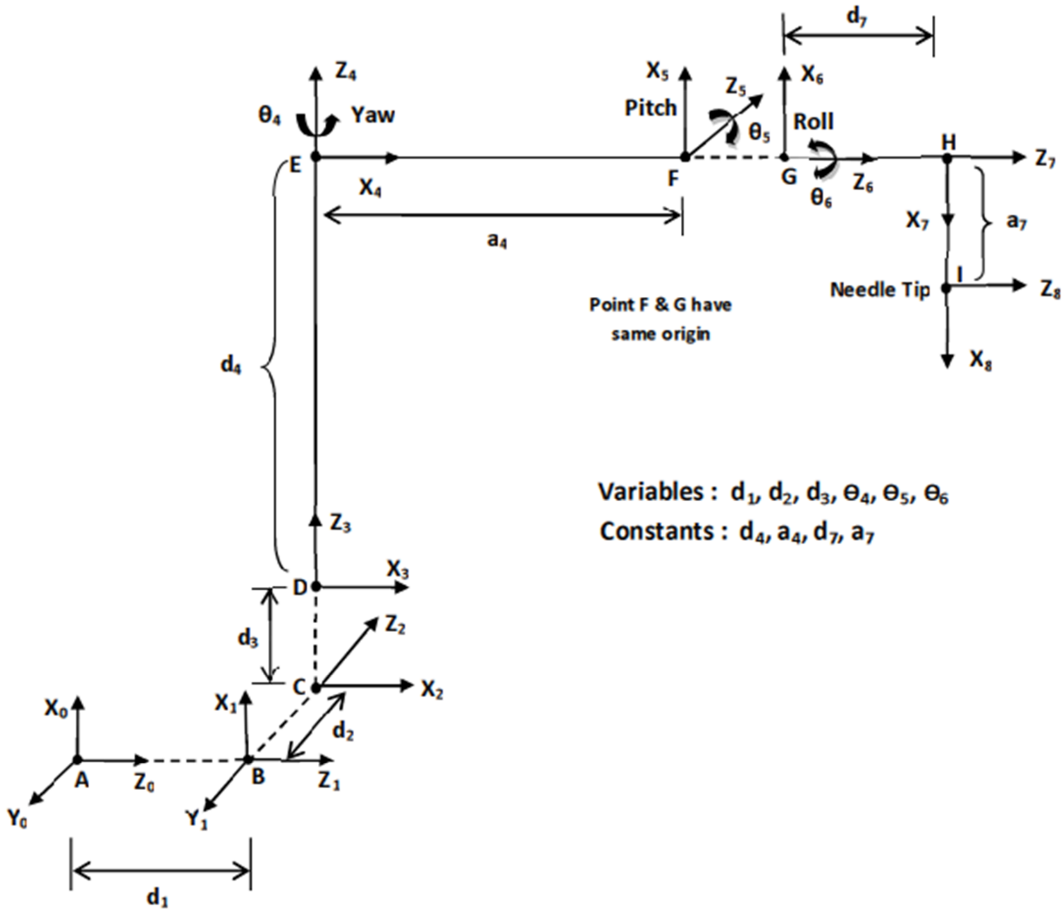


Figure 3.8 - (b) Assignment of frames

Note, for this robot, the joint axes of joints 5 and 6 (point F & G) intersect at a common point, and this point of intersection coincides with the origin of frames {5} and {6}. The gearing arrangement described previously at the wrist of the manipulator couples together the motions of joints 5 and 6.

After the reference frames are assigned, a D-H parameter table (Table 3.1) which relates frame $\{i\}$ to $\{i+1\}$ is made.

Table 3.1: **D-H PARAMETERS**

JOINT	α_{i-1} (X_{i-1})	a_{i-1} (X_{i-1})	θ_i (Z_i)	d_i (Z_i)
1	0	0	0	d_1
2	90	0	90	d_2
3	90	0	0	d_3
4	0	0	θ_4	d_4
5	-90	a_4	θ_5 - 90	0
6	-90	0	θ_6 - 90	0
7	0	0	-90	d_7
8	0	a_7	0	0

3.6.1 FORWARD KINEMATICS

Using the Denavit-Hartenberg notation (60), the transformation between the adjacent {i} and {i-1} frames is noted as ${}^{i-1}T_i$, which is a 4x4 homogeneous matrix.

$$
{}^{i-1}T_i = \begin{pmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -d_i s\alpha_{i-1} \\ s\theta_i s\alpha_{i-1} & s\alpha_{i-1} c\theta_i & c\alpha_{i-1} & -d_i c\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

There are four parameters in the transformation matrix, including link length (a), link twist (α), link offset (d), and joint angle (θ).

a_{i-1} = the distance from Z_i to Z_{i-1} measured along X_i;

α_{i-1} = the angle between Z_i and Z_{i-1} measured about X_i ;

d_i = the distance from X_{i-1} to X_i along Z_i ; and

θ_i = the angle between X_{i-1} and X_i measured about Z_i .

$$
{}^{0}T_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{1}T_2 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -d_2 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{2}T_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -d_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{3}T_4 = \begin{pmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & 0 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 1 & d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{4}T_{5} = \begin{pmatrix} \sin\theta_5 & \cos\theta_5 & 0 & a_4 \\ 0 & 0 & 1 & 0 \\ \cos\theta_5 & -\sin\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{5}T_{6} = \begin{pmatrix} \sin\theta_6 & \cos\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \cos\theta_6 & -\sin\theta_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{6}T_{7} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & d_7 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{7}T_{8} = \begin{pmatrix} 1 & 0 & 0 & a_7 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

${}^0T_1$ represent transformation linking frame 0 to 1, ${}^1T_2$ represent transformation linking frame 1 to 2 and so on. Multiplication of all above transformation matrices will give a single transformation linking base frame 0 to final frame 8.

Using the matrix multiplication of the individual link matrixes, the transformation matrix from the tool center point (TCP) to the robot base is then calculated as shown below.

${}^0T_1\,{}^1T_2\,{}^2T_3\,{}^3T_4\,{}^4T_5\,{}^5T_6\,{}^6T_7\,{}^7T_8 = {}^0T_8 =$

(Using abbreviations $\cos\theta_i = C_i$, $\sin\theta_i = S_i$, hence, $\cos\theta_4 = C_4$, $\sin\theta_4 = S_4$ and so on)

$$\begin{pmatrix} -C_5C_6 & C_5S_6 & -S_5 & -a_7C_5C_6 + d_4 - d_7S_5 + d_3 \\ S_4S_5C_6 - C_4S_6 & -S_4S_5S_6 - C_4C_6 & -S_4C_5 & a_7(S_4S_5C_6 - C_4S_6) - d_7S_4C_5 - a_4S_4 - d_2 \\ -C_4S_5C_6 - S_4S_6 & C_4S_5S_6 - S_4C_6 & C_4C_5 & -a_7(C_4S_5C_6 + S_4S_6) + d_7C_4C_5 - a_4C_4 + d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

…………………... (i)

Also,

$${}^0T_8 = \begin{pmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{…………(ii)}$$

In the above equation, $[p_x , p_y , p_z]$ is the position vector and $[n_x, n_y, n_z]$, $[s_x, s_y, s_z]$ and $[a_x, a_y, a_z]$ are orthonormal vectors that describes the orientation. $[p_x, p_y, p_z]$ represent the coordinates of tip of the end effector with respect to the base.

Equating the terms of matrices (i) and (ii):

$$
\left.
\begin{aligned}
n_x &= -\cos\theta_5*\cos\theta_6 \\
n_y &= \sin\theta_4*\sin\theta_5*\cos\theta_6 - \cos\theta_4 \sin\theta_6 \\
n_z &= -\cos\theta_4*\sin\theta_5*\cos\theta_6 - \sin\theta_4*\sin\theta_6 \\
s_x &= \cos\theta_5*\sin\theta_6 \\
s_y &= -\sin\theta_4*\sin\theta_5*\sin\theta_6 - \cos\theta_4*\cos\theta_6 \\
s_z &= \cos\theta_4*\sin\theta_5*\sin\theta_6 - \sin\theta_4*\cos\theta_6 \\
a_x &= -\sin\theta_5 \\
a_y &= -\sin\theta_4*\cos\theta_5 \\
a_z &= \cos\theta_4*\cos\theta_5 \\
p_x &= -a_7*\cos\theta_5*\cos\theta_6 + d_4 - d_7*\sin\theta_5 + d_3 \\
p_y &= a_7*(\sin\theta_4 \sin\theta_5 \cos\theta_6 - \cos\theta_4 \sin\theta_6) - d_7*\sin\theta_4*\cos\theta_5 - a_4*\sin\theta_4 - d_2 \\
p_z &= -a_7*(\cos\theta_4*\sin\theta_5*\cos\theta_6 + \sin\theta_4*\sin\theta_6) + d_7*\cos\theta_4*\cos\theta_5 - a_4*\cos\theta_4 + d_1
\end{aligned}
\right\} \quad \text{(iii)}
$$

The equations shown above constitute the kinematics of the proposed MRI compatible robot. They specify how to compute the position and orientation of frame {7} relative to frame {0} of the robot. These are the basic equations for all kinematic analysis of this manipulator.

The forward kinematics presented above would solve the following problem: given the joint positions, what is the corresponding end effector's position. The next step would be to solve the inverse kinematics to identify what would be the corresponding joint positions given the actual end effector position. The needle of the proposed robot manipulator has to reach different points in the brain inside the MRI scanner. The database of MRI scanner will provide this information and is a well-established technique. To control the robot, a control system has to be developed to command the robot manipulator to move certain distances to reach to the target point. Due to the nature of geometry, the inverse kinematics of the proposed robot is difficult to solve using traditional DH transformations techniques alone. An algorithm has been developed to solve the inverse kinematics problem as presented in section 3.7.4.1.

3.7 CONTROLLING THE ROBOT MANIPULATOR

3.7.1 CONTROL STRATEGY

The control system employed for the robot manipulator is a hybrid between supervisory controlled robotic system and shared control system as shown in Figure 3.9. The surgeon plans the operation off line and specifies the motions the robot must follow to perform the operation. The robot then performs the specified motions autonomously under the supervision of the surgeon. After the robot has completed its motion, the surgeon takes over completely to conclude the task.

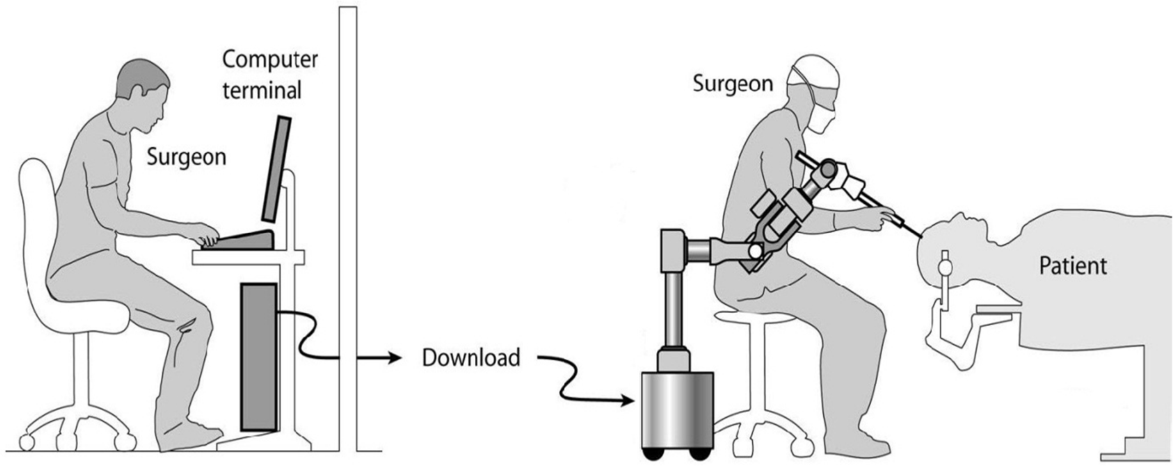


Figure 3.9 - Control system employed for the robot manipulator [42]

Control of the robot manipulator is done in two steps:

a) Preoperative planning for the placement of the end-effector prior to insertion, and

b) alignment and insertion of the needle.

The surgeon will examine a set of MR scans and define a skin entry point and the target point. Figure 3.10 shows target planning on a three-dimensional representation of the anatomy of brain. This preoperative planning procedure yields the desired task-space positioning vector of the end-effector, at which the robot can automatically position itself after solving the inverse kinematics problem.

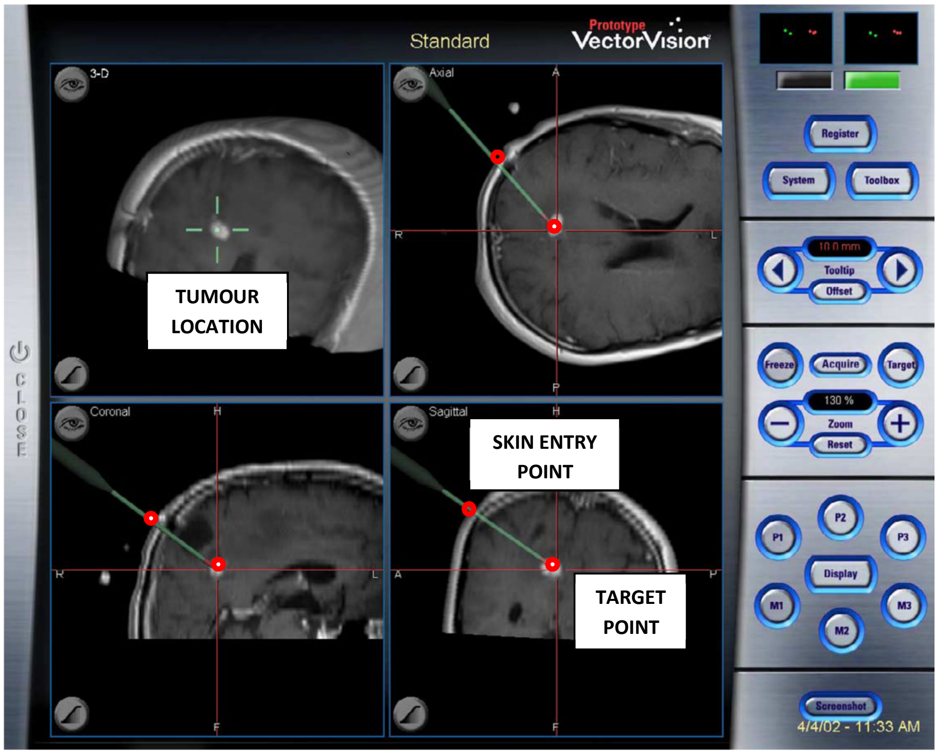


Figure 3.10 - Example of stereotactic path planning using 3D images to define the needle entry and target points [9]

The information regarding the entry and target point is sent to the real-time target PC. The PC calculates the inverse kinematics problem to find how much each link should move to position the end-effector to the proper orientation. This controls the manipulator to bring its end-effector to the prescribed insertion vector. The surgeon then fine adjusts the position and inserts the needle/probe to the target using manual activation of the insertion dof.

3.7.2 OVERVIEW OF THE CONTROL HARDWARE AND SOFTWARE

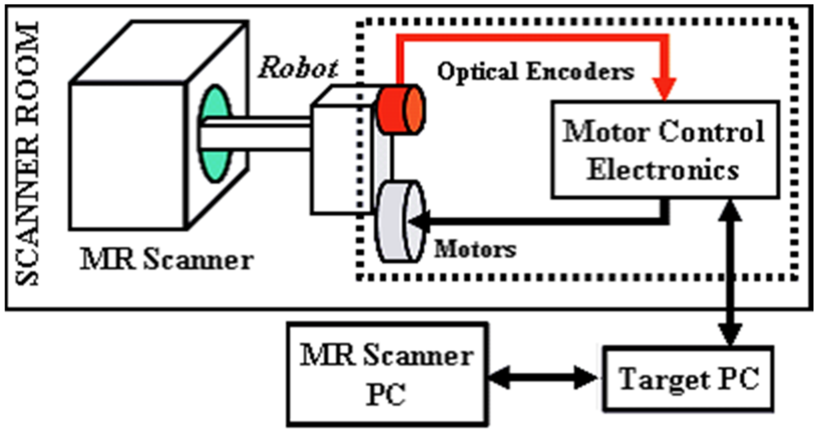


Figure 3.11 – System Layout

Figure 3.11 shows the overall layout of the system. The control of the device is performed with an in-house developed software and commercial hardware system, which would be interfaced to the manipulator.

The Real-time target PC (2.8 GHz Pentium 4, 512 MB RAM), has a motion control card (PMC MultiFlex PCI 1440) installed to control the motors. The target PC is dedicated to the control of the manipulator. A MultiFlex PCI 1440 motion controller (Figure 3.12) was chosen for high-performance PC-based multi-axis motion control applications. It can control up to 4 axes of analog servo and up to 4 axes of stepper motors. Optional encoder inputs for closed-loop stepper motor control and general-purpose analog inputs are also available for this model. The card provides 16 bi-directional opto-isolated inputs with individually configurable supply & return: 5 to 24 Volts (for home, ± limits, amp fault, etc.) and 12 high-current outputs (sinking up to 100 mA): 5 to 24 Volts (for drive enable/disable, full/half current, step/microstep, etc.). The controller card is fully Programmable in Visual Basic, Delphi, LabVIEW and Microsoft Visual Studio C++.

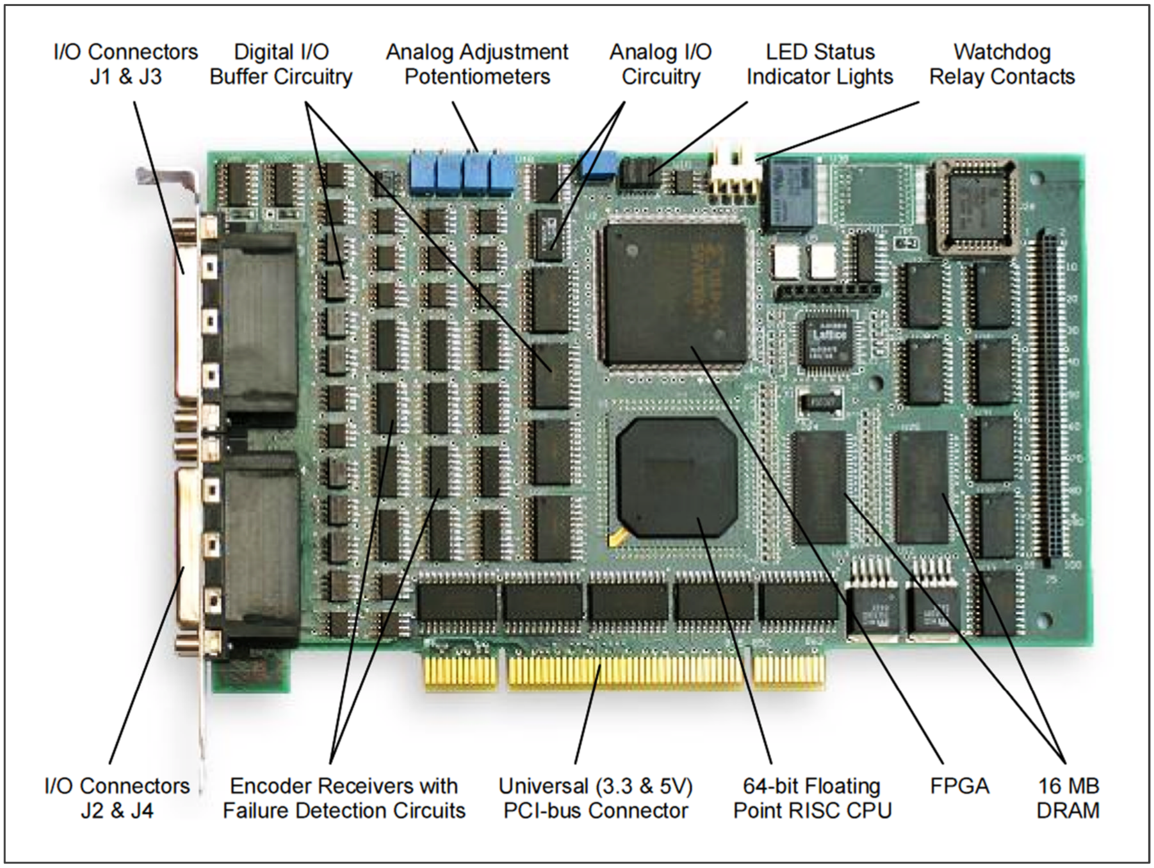


Figure 3.12 - MultiFlex PCI 1000 Series Board Layout

The motors of the manipulator are numbered as per the connections with the PCI control card. The controller card controls 4 axes of analog servo with connectors J1 and J2 and 4 axes of stepper motors with connectors J3 and J4. Since the robot has 2 servos and 4 stepper motors, connection J2 is not used. Thus, in effect Axis 3 and Axis 4 are not used in this application. The assignment of axis numbers is shown in Figure 3.13 below.

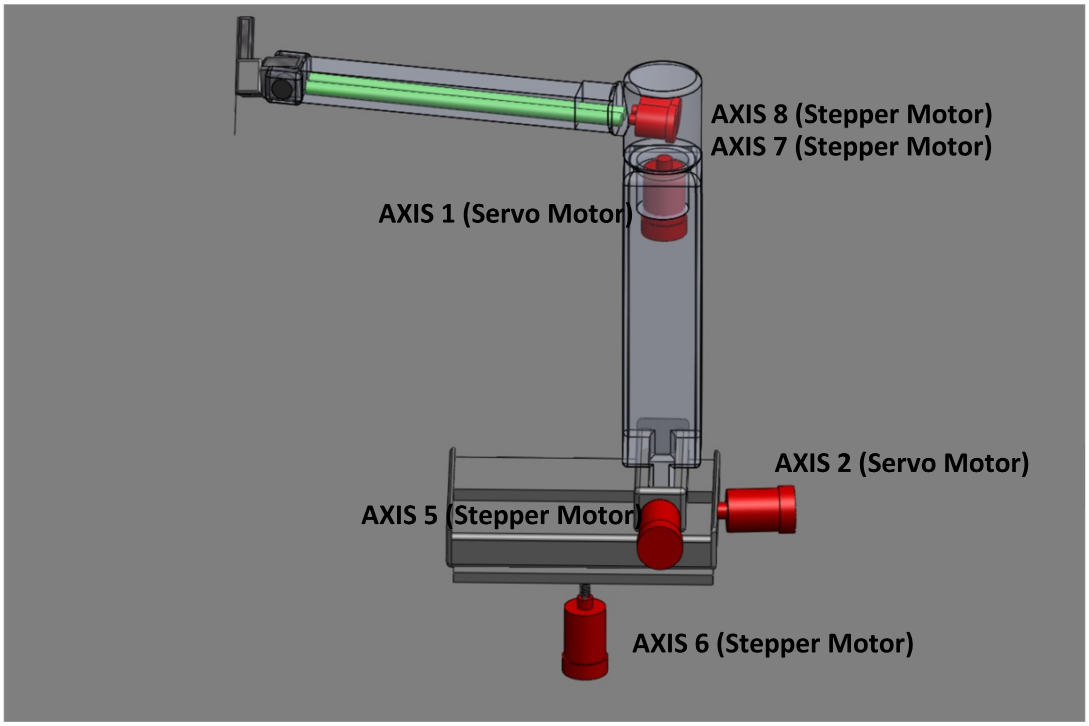
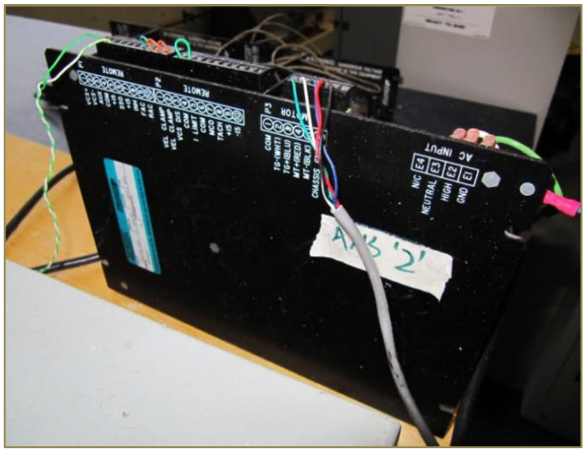


Figure 3.13 – Assignment of Axis numbers

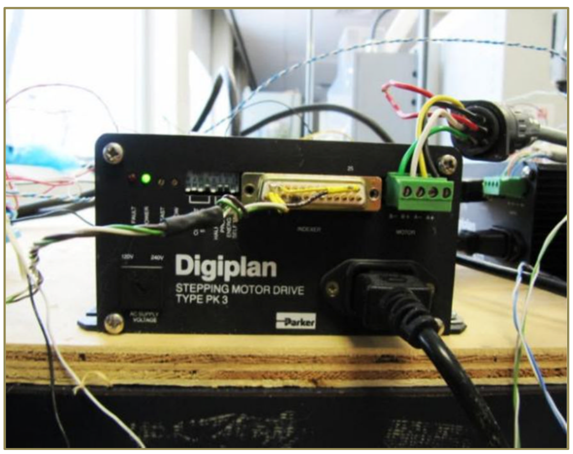
3.7.3 HARDWARE COMPONENTS

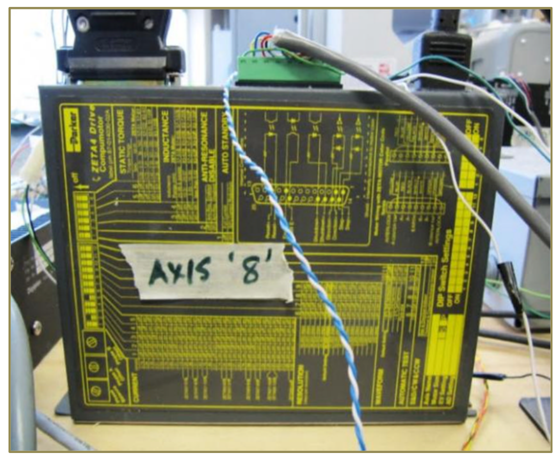
The proposed manipulator has 6 motors in total, with four stepper motors and the other two being servo motors with encoder feedback. The two servo motors used are Electrocraft's ES0586-30-500 model with constant stall torque of 20 oz-in and peak torque of 40 oz-in. at 5000 rpm. The encoders used are 5 VDC light duty standard shaft incremental encoders TRD-S100-VD. The servo drive used to control these two motors is Electrocraft MAX-100 PWM Servo Drive.

Two of the four stepper motors are Compumotor's, PK3-5-51. The drive system used for these stepper motors is Digiplan PK3 drive. The other two stepper motors are Superior Electric Slo-Syn Stepping motors M092-FD09 (constant torque 200 oz-in). These motors are driven by Parker's Zeta 4 drive (Figure 3.14).



(a) (b)





(c) (d)

Figure 3.14 – (a) Electrocraft MAX-100 PWM Servo Drive; (b) TRD-S100-VD encoders; (c) Digiplan PK3 drive (d) Parker's Zeta 4 drive

Layout of the hardware components is shown in Figure 3.15 below.

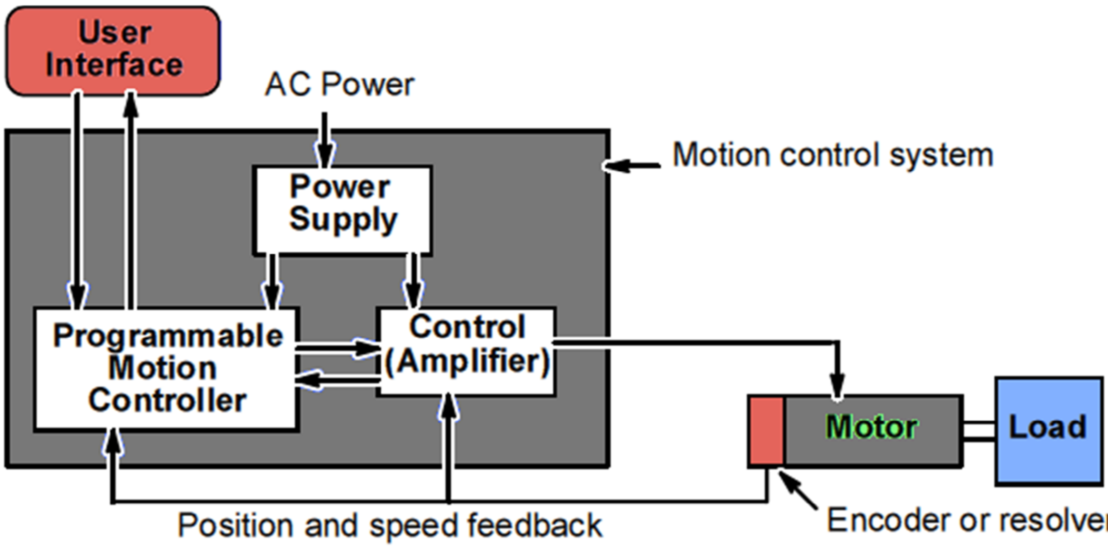


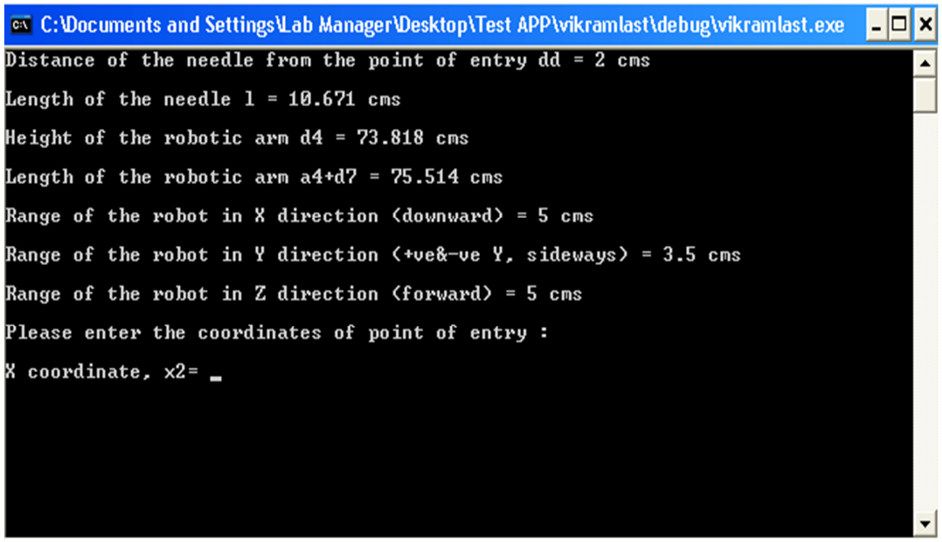
Figure 3.15 – Layout of the hardware components

For more information about the motors, drives and wirings please refer to Appendix A.1 - A.2.

3.7.4 CONTROL SOFTWARE

Programming of the PCI controller card is done in Microsoft Visual studio C++ 2005. An intuitive user interface has been developed as shown in Figure 3.16 to input the values of skin entry point and target point and to control the robot. The first dialogue box provides the details of the manipulator dimensions and prompts the user to input the skin entry point and target point. It then calculates the forward/inverse kinematics and specifies how much each motor must move to align the end-effector to the required orientation. The second dialogue box gives the user options such as switch on/off the motors, move the robot, send the robot back to its home position, etc. In addition, it gives the actual and target position of the motors and the following errors. This dialogue box is linked to the first dialogue box in a way that the output of the first dialogue box (i.e.

required motor movements) serves as input for the second dialogue box (to physically move the motors). The complete C++ code is presented in the Appendix B.1 – B.2.



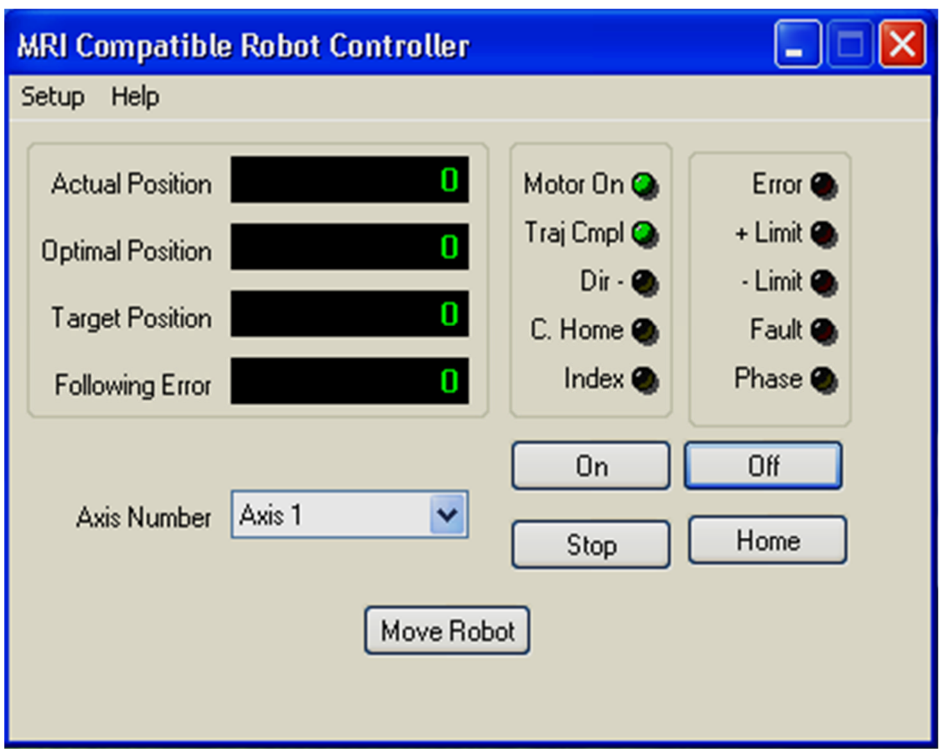


Figure 3.16 – (Top figure) Dialogue box for entering skin entry point and target point; (Bottom figure) Dialogue box for controlling the robot.

3.7.4.1 ALGORITHM

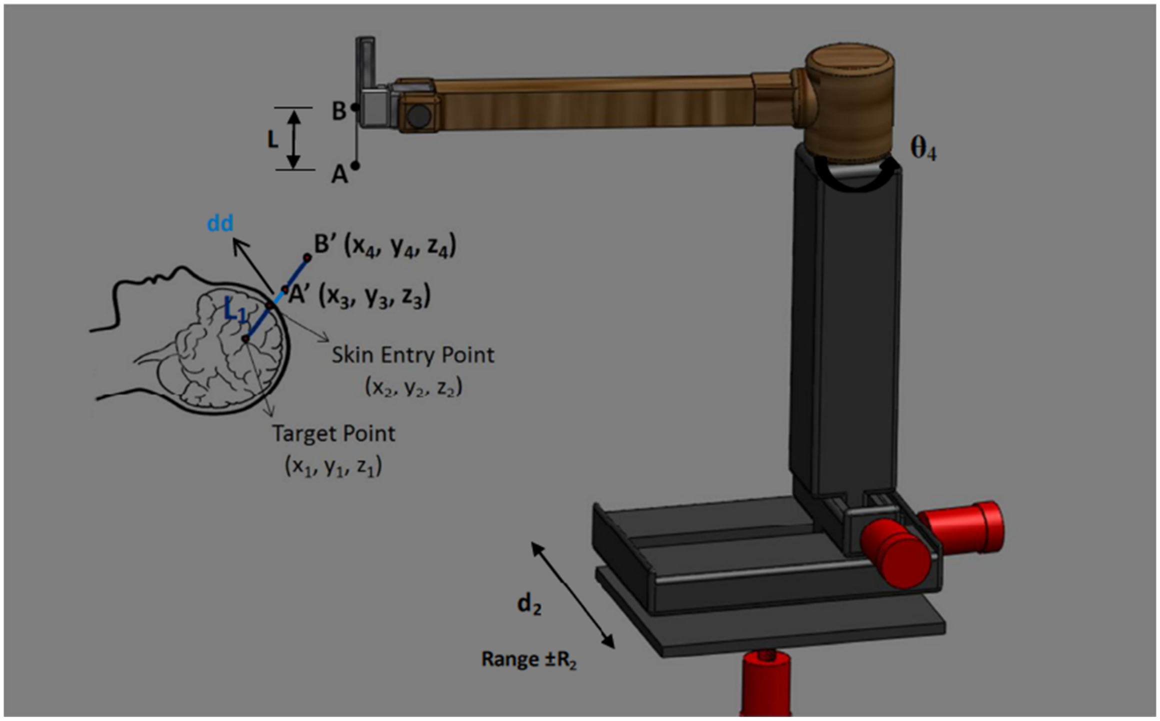


Figure 3.17 – Algorithm to control the manipulator to bring its end-effector to the prescribed insertion vector

The algorithm upon which the complete programming is based is discussed below:

Length of the needle "L" (line AB) is assumed known and will be user input. To incorporate inaccuracies if any in the system and for safety concerns, the needle tip has to be at some distance "dd" away from the skin entry point. The needle tip has to reach the point (x_3, y_3, z_3) at an angle described by points (x_1, y_1, z_1) & (x_2, y_2, z_2) i.e. line L_1. After the needle has been aligned parallel to line L_1, linear translation of the needle would make the needle reach point (x_1, y_1, z_1).

Linear degrees of freedom d_1, d_2, d_3 have a limited range of motion. Hence, θ_4 (Yaw) is employed to increase the range of motion. So, if the point "B'" is within the range of "d_2", then θ_4 would not be used. The algorithm is based on the limit of linear dof "d_2". Range of "d_2" is $\pm R_2$ cm.

As per the DH notations, mathematical equations to calculate the coordinates of point A (needle tip) are:

$$\left.\begin{aligned}
p_{xA} &= -a_7*\cos\theta_5*\cos\theta_6 + d_4 - d_7*\sin\theta_5 + d_3 \\
p_{yA} &= a_7*(\sin\theta_4 \sin\theta_5 \cos\theta_6 - \cos\theta_4 \sin\theta_6) - d_7 \sin\theta_4 \cos\theta_5 - a_4 \sin\theta_4 - d_2 \\
p_{zA} &= -a_7*(\cos\theta_4*\sin\theta_5*\cos\theta_6 + \sin\theta_4*\sin\theta_6) + d_7*\cos\theta_4*\cos\theta_5 - a_4*\cos\theta_4 + d_1
\end{aligned}\right\} \quad \text{(iv)}$$

The above equations result from the DH formulation shown in Section 3.6.1 equations (iii).

By assuming a value of $a_7 = 0$, the coordinates of point B are:

$$\left.\begin{aligned}
p_{xB} &= d_4 - d_7*\sin\theta_5 + d_3 \\
p_{yB} &= - d_7 \sin\theta_4 \cos\theta_5 - a_4 \sin\theta_4 - d_2 \\
p_{zB} &= d_7*\cos\theta_4*\cos\theta_5 - a_4*\cos\theta_4 + d_1
\end{aligned}\right\} \quad \text{(v)}$$

The algorithm is carried out using the following steps:

1) The user inputs the values of skin entry point (x_2, y_2, z_2) and target point (x_1, y_1, z_1) to the system.
2) The needle should be located at point (x_3, y_3, z_3) at distance "dd" away from the skin entry point and parallel to line L_1.

Calculating the value of coordinates (x_3, y_3, z_3) and (x_4, y_4, z_4):

$$\left.\begin{aligned}
x_3 &= x_1 + D*(x_2-x_1)/d;\\
y_3 &= y_1 + D*(y_2-y_1)/d;\\
z_3 &= z_1 + D*(z_2-z_1)/d;\\
x_4 &= x_1 + D'*(x_2-x_1)/d;\\
y_4 &= y_1 + D'*(y_2-y_1)/d;\\
z_4 &= z_1 + D'*(z_2-z_1)/d;
\end{aligned}\right\} \quad \text{(vi)}$$

where,

d = distance between point (x_1, y_1, z_1) & (x_2, y_2, z_2)

$= \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2 + (z_2-z_1)^2}$

D = distance of point (x_3, y_3, z_3) from point (x_1, y_1, z_1) = d + dd

D' = distance of point (x_4, y_4, z_4) from point (x_1, y_1, z_1) = d + dd + l

3) If $\pm y_4 <= R_2$, i.e. if point B' is within the range of d_2, then θ_4 will not be used. So in this case $\theta_4 = 0$.

$$\left.\begin{aligned}
d_1' &= z_4 - a_4 - d_7;\\
d_2' &= -y_4;\\
d_3' &= x_4 - d_4;
\end{aligned}\right\} \quad \text{(vii)}$$

where, d_1', d_2' and d_3' are the linear translations needed to align the point B reach point B'.

$$\left.\begin{aligned}
\theta_6 &= \sin^{-1}((-d_2' - y_3)/a_7)\\
\theta_5 &= \sin^{-1}((a_4 + d_7 + d_1' - z_3)/(a_7*\cos\theta_6))
\end{aligned}\right\} \quad \text{(viii)}$$

θ_5, θ_6 are the pitch and roll angles to make the needle parallel to line L_1. But since the two rotations are carried out at a distance d_7 from the end-effector, there has to be a change in the values of d's to make the needle reach point (x_1, y_1, z_1) and be parallel to line L_1. The revised values are:

$$\left.\begin{aligned}
&d_1 = z_3 + a_7*\sin(\theta_5)*\cos(\theta_6) - d_7*\cos(\theta_5) - a_4;\\
&d_2 = -a_7*\sin(\theta_6) - y_3;\\
&d_3 = x_3 + a_7*\cos(\theta_5)*\cos(\theta_6) + d_7*\sin(\theta_5) - d_4;
\end{aligned}\right\}\quad \text{(ix)}$$

So the motors have to move by distances d_1, d_2, d_3, θ_4, θ_5, and θ_6 to align the needle parallel to line L_1 and reach to point (x_3, y_3, z_3).

4) If $y_4 > R_2$, θ_4 will also be incorporated since point B' is out of range for d_2. In that case,

$$\left.\begin{aligned}
&d_3' = x_4 - d_4,\\
&d_2' = -R_2 \qquad (\text{if } y_4 > R_2 \text{ then } d_2' = -R_2;\ \text{if } y_4 < -R_2 \text{ then } d_2' = R_2)\\
&\theta_4 = \sin^{-1}((-y_4 - d_2)/(a_4 + d_7))\\
&d_1' = z_4 - (a_4+d_7)*\cos(\theta_4);
\end{aligned}\right\}\quad \text{(x)}$$

again d_1', d_2' and d_3' are the three translations required to make point B reach point B'.

In addition, the arm has to rotate by θ_4.

$$\left.\begin{aligned}
&\theta_6 = \sin^{-1}((a + b)/a_7));\ \text{and}\\
&\theta_5 = \sin^{-1}((((a_4+d_7)*\cos(\theta_4) - z_3 + d_1 - a_7*\sin(\theta_4)*\sin(\theta_6))/(a_7*(\cos(\theta_4))*(\cos(\theta_6)));
\end{aligned}\right\}\quad \text{(xi)}$$

Here,

$a = -(y_3 + d_2 + (a_4+d_7)*\sin(\theta_4))*\cos(\theta_4);$

$b = -(z_3 - d_1 - (a_4+d_7)*\cos(\theta_4))*\sin(\theta_4);$

Now,

$$\left.\begin{aligned} d_1 &= z_3 + a_7{*}\cos(\theta_4){*}\sin(\theta_5){*}\cos(\theta_6) - d_7{*}\cos(\theta_4){*}\cos(\theta_5) - a_4{*}\cos(\theta_4); \\ d_2 &= a_7{*}\sin(\theta_4){*}\sin(\theta_6){*}\cos(\theta_6) - a_7{*}\cos(\theta_4){*}\sin(\theta_6) - d_7{*}\sin(\theta_4){*}\cos(\theta_5) - a_4{*}\sin \\ &\quad (\theta_4) - y3; \\ d_3 &= x_3 + a_7{*}\cos(\theta_5){*}\cos(\theta_6) + d_7{*}\sin(\theta_5) - d_4; \end{aligned}\right\} \text{(xii)}$$

So the motors have to move by distances d_1, d_2, d_3, θ_4, θ_5, and θ_6 to align the needle parallel to line L_1 and reach to point (x_3, y_3, z_3).

3.8 SUMMARY

This chapter presented the methodology for developing a personal computer (PC) based needle insertion robotic manipulator for minimally invasive interventions in the brain. The challenges in designing such a robot were discussed first and then robot specifications were decided based on these considerations. After the robot specifications are decided, potential designs for the robot is studied. The critical design considerations include, placing all the actuators outside the MRI bore to ensure that they do not cause any image artifacts; limiting the number of degrees of freedom of the robot to make the controls simpler; and ensuring that the links of manipulator are not subjected to gravitational forces which otherwise would be a safety concern. The design selected for robot consists of six degrees of freedom with an additional degree of freedom at the end-effector. The forward kinematics of this design is solved by using the algorithm developed by Denavit and Hartenberg models. A prototype of this robot is then fabricated. This work is a proof-of-concept of the design and control of the proposed system. If proof of concept for the design of manipulator is fully tested, MRI compatibility can be the next step. The MRI compatible robot will not be developed as part of this thesis. After the manipulator is constructed, an algorithm was developed to control the six degrees of freedom of the manipulator. Commercially available controller card was used to control the motors of the manipulator. The controller card was programmed in Visual Studio C++. The program commands the robot to manipulate its end effector (needle) to a prescribed insertion vector defined by the surgeon. The user can then fine adjusts it and inserts the needle/probe to the target using manual activation of the insertion dof.

CHAPTER

4

RESULTS

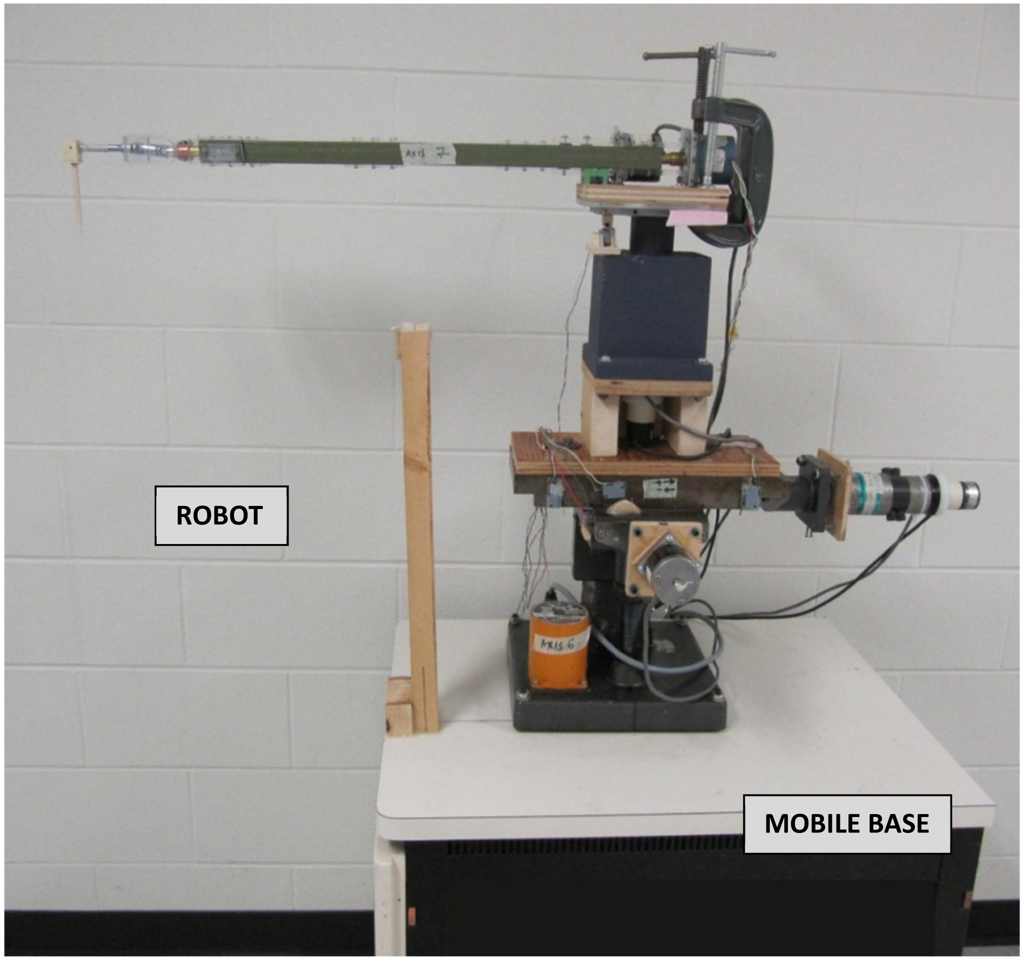
4.1 INTRODUCTION

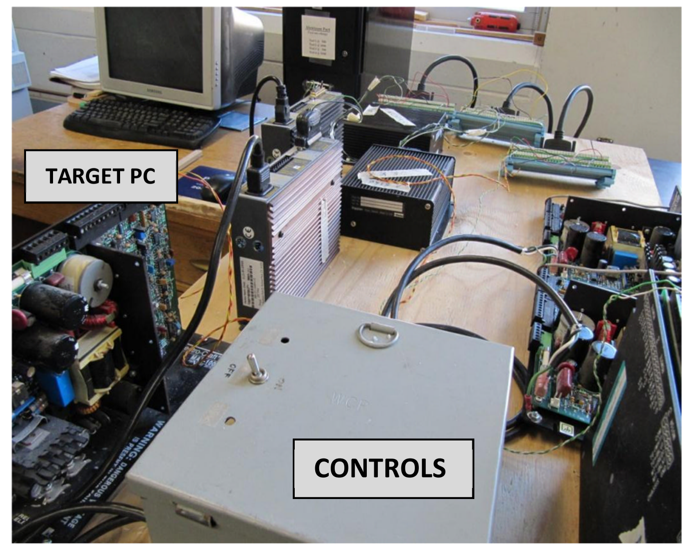
After the robotic system was designed and constructed, the next important step is to carry out its performance evaluation. Experiments are conducted to evaluate the needle positioning accuracy of the robotic system by building test structures. These test structures contain points in space that the needle is required to reach. The results obtained from the accuracy tests are tabulated and analyzed. The accuracy of the system in positioning the needle to the required points in space would also test the control algorithm developed for this system.

4.2 SYSTEM SETUP

The complete system with manipulator and its control hardware is shown in Figure 4.1. The manipulator is mounted on a mobile base which gives the flexibility to move the manipulator in and out of the operation envelope. This feature would be useful in surgical applications. After the preoperative images have been obtained, the robot can be positioned next to the operating room table and locked in place using a wheel brake. A counter-balance mechanism, based on elevator design, allows easy adjustment of the base height.

Figure 4.1 - System Setup





The PC shown in figure controls the actuators of the manipulator. Also shown in the figure are the motor drives and amplifiers. The details of the wirings are presented in the Appendix A.1- A.2. The complete system is now set in place and the next important step of testing its control and positional accuracy can be carried out.

4.2.1 DIMENSIONS AND RANGE

The dimensions and range of motion of the proposed model are shown in Figure 4.2 below.

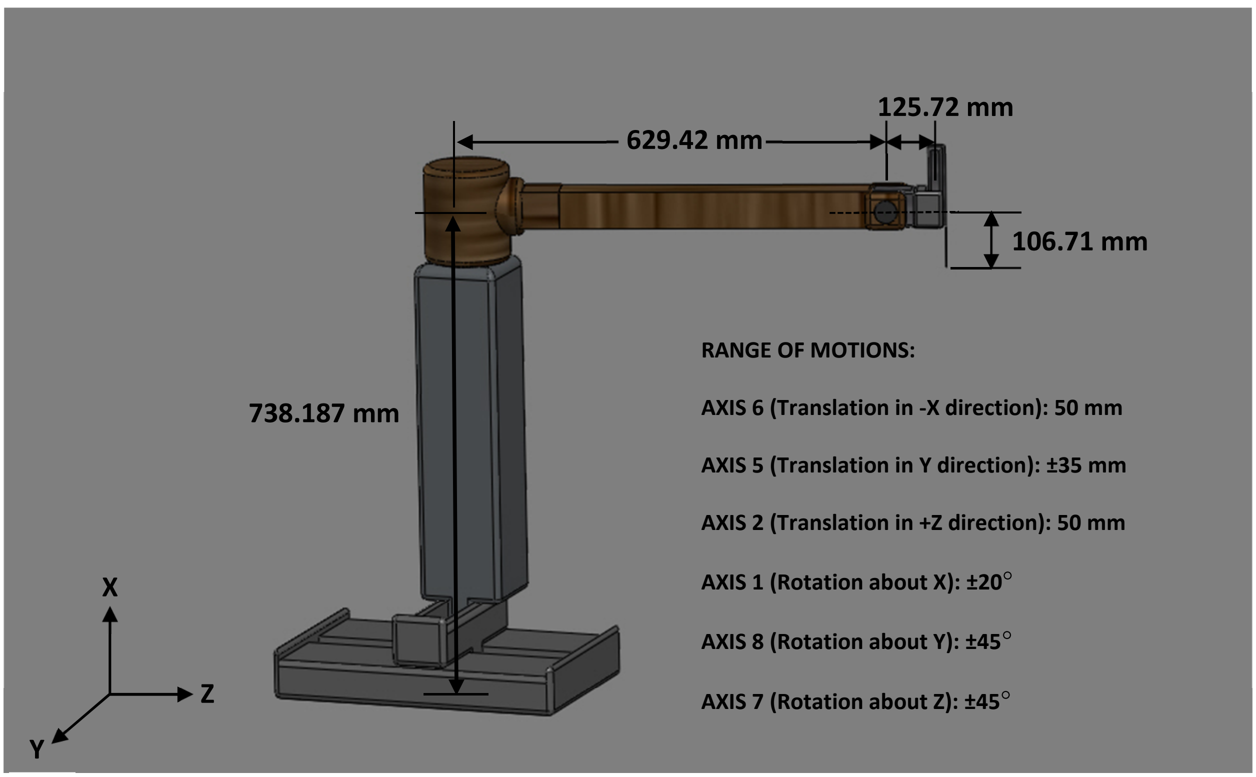


Figure 4.2 – Robot dimensions and range of motions

The robot is tested for the range of motions mentioned above. In the actual model the range of motions have to be increased to cover the complete anatomy of the brain.

4.3 ANALYSIS OF MOTION

Experiments are performed to verify that the control algorithm would command the robot to move to the correct position. Before this was conducted, the accuracy and precision of the robotic system was analysed.

Limit switches installed at strategic locations on all the axes will enable quantification of mechanical backlash errors. For this measurement, the motors are commanded to move using the inbuilt software provided with the PCI controller card called “Motor mover”. PMC’s Motor Mover program allows absolute, relative, and cyclic move sequences to be executed, monitor position and status of the axis. By selecting the “Setup” button, velocity parameters (maximum velocity, acceleration, and deceleration), velocity profile (Trapezoidal, S curve, or Parabolic) can be set. This card also enables over travel limits which is very useful for measuring the backlash errors. When the limit of motion is reached the limit switch will be triggered, which stops the motor.

Backlash error was determined by the following procedure. Figure 4.3 shows a single axis drive unit. The carriage moves over a saddle using a lead-screw. The two extreme positions are monitored by limit switches A and B.

1) Motor was moved in one particular direction to trigger the first limit switch. The software monitors the status of this switch and stops the monitor when the switch is triggered.

2) From this position it is moved in the opposite direction by a suitable distance to trigger the second limit switch B.

3) The readings are then noted, as to how many encoder counts (in case of servo motors) or how many steps (in case of stepper motors) it took to trigger the second limit switch B.

4) Again the motor is moved by an appropriate distance to trigger the first limit switch A.

5) The number of encoder counts or number of steps needed to reach the first switch are noted.

6) The procedure is repeated for 50 times. The difference between any two readings is measured; which by definition is the repetition error, i.e., backlash.

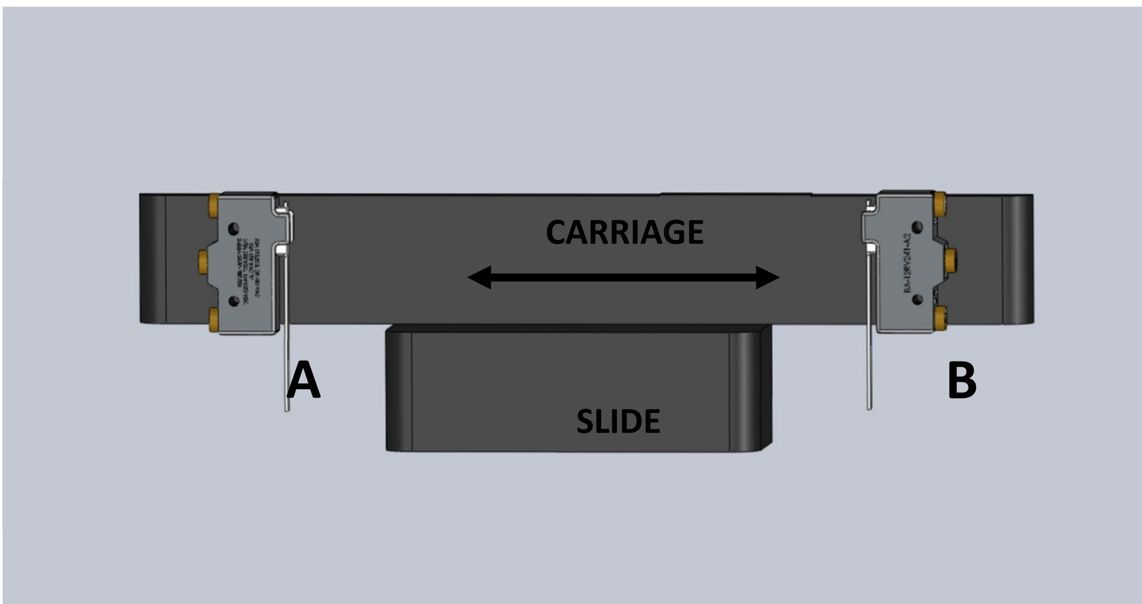


Figure 4.3 – Determining backlash error

4.2.1 TEST SETUP FOR BACKLASH ERROR

This section gives the details about the setup for measuring the backlash errors in all the 6 axes. Limit switches were attached/installed directly on to the manipulator wherever possible (for axes 1, 2, 5, 6). For remaining axes (7 and 8), separate structures were built to mount the limit switches.

Axis 1 (Rotational dof):

Axis 1 consists of the rotating arm of the manipulator driven by a servo motor. The motor's angular position is monitored by an encoder attached to it for positional feedback. This axis provides the yaw motion of the end-effector. The limit switches for axis 1 are mounted on to the manipulator itself.

The motor is rotated in both clockwise and anticlockwise direction, so that the arm triggers the switches A & B back and forth as shown in Figure 4.4. This is carried out for 50 times and the readings are noted according to the procedure discussed above.

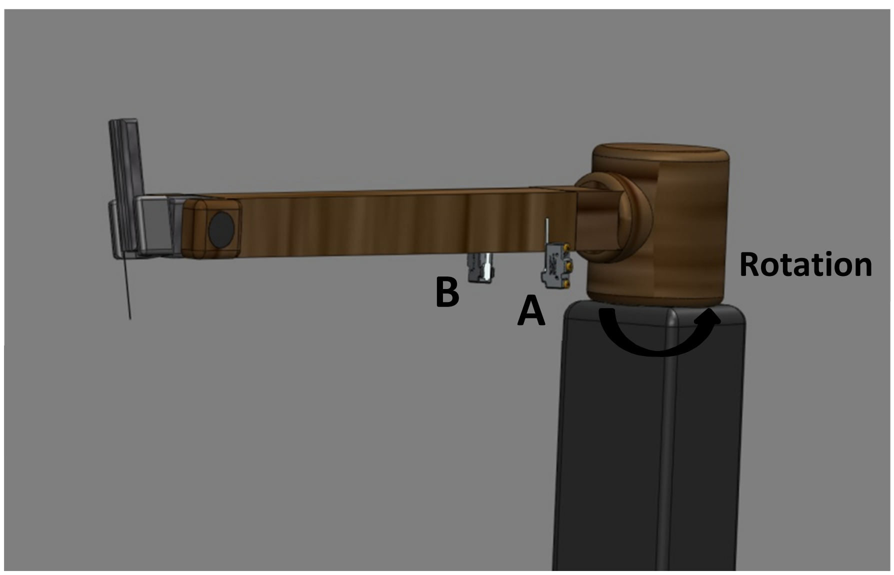


Figure 4.4 – Limit switches for Axis 1

Again, the difference between any two consecutive readings would give the backlash error for the axis. The readings have been tabulated as shown in Table 4.1 below.

Table 4.1 – Backlash Errors for Axis 1: Difference between any two values

NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS
1	-2518	11	-2517	21	-2517	31	-2517	41	-2516
2	2517	12	2516	22	2518	32	2517	42	2516
3	-2517	13	-2517	23	-2518	33	-2517	43	-2516
4	2518	14	2517	24	2517	34	2517	44	2516
5	-2518	15	-2516	25	-2517	35	-2516	45	-2516
6	2519	16	2516	26	2517	36	2515	46	2517
7	**-2519**	17	-2516	27	-2518	37	-2515	47	-2516
8	**2516**	18	2516	28	2518	38	2515	48	2515
9	-2516	19	-2517	29	-2517	39	-2516	49	-2516
10	2518	20	2518	30	2517	40	2516	50	2516

As seen from the table, the backlash error is very small. The maximum difference between any two consecutive readings is 3 encoder counts. Average value of backlash error for 50 readings is 0.5306 encoder counts.

A graphical representation of the range of backlash error is shown in Figure 4.5 below.

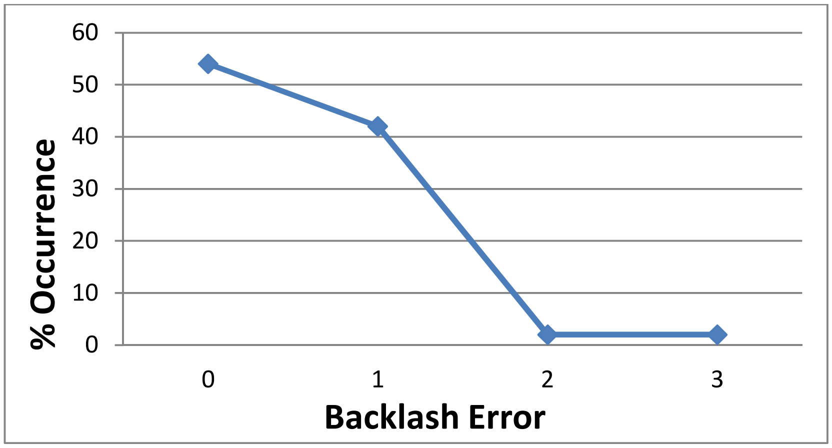


Figure 4.5 - Backlash Error for Axis 1

From the above graph it is seen that over the 50 readings, 54 % of the time the backlash error is 0. The highest backlash error is 2 to 3 encoder counts and happens only 2 % of the time.

Axis 2 (Translational dof):

This axis consists of a linear joint with servo motor and has an encoder attached to it for positional feedback. The limit switches are mounted at strategic locations.

The axis motor is moved back and forth as in previous step 50 times. This allows measurement of backlash in the 2nd axis. The results from this study are shown in a tabulated as well as in graphical form in Table 4.2 and Figure 4.6.

Table 4.2 – Backlash Errors for Axis 2: Difference between any two values

NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS
1	-18274	11	-18273	21	-18273	31	-18274	41	-18276
2	18272	12	18272	22	18274	32	18275	42	18276
3	-18272	13	-18272	23	-18273	33	-18274	43	-18276
4	18272	14	18272	24	18270	34	18273	44	18276
5	-18269	15	-18272	25	-18271	35	-18274	45	-18278
6	18270	16	18272	26	18273	36	18274	46	18276
7	-18272	17	-18272	27	**-18275**	37	-18274	47	-18275
8	**18270**	18	18273	28	**18272**	38	18276	48	18276
9	**-18273**	19	-18272	29	-18273	39	-18278	49	-18278
10	18274	20	18273	30	18274	40	18277	50	18278

From the table, it is seen that the maximum difference between any two consecutive readings is 3 encoder counts. Average value of backlash error for 50 readings is 1.081 encoder counts. A graphical representation of the range of backlash error is shown below.

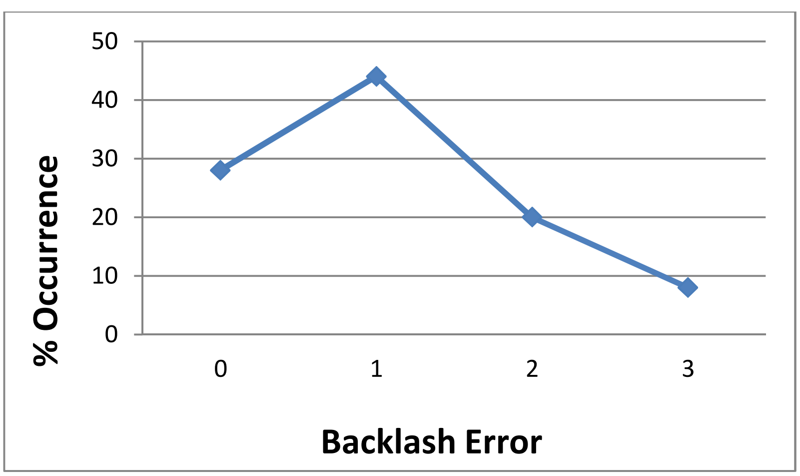


Figure 4.6 - Backlash Error for Axis 2

Axis 5 (Translational dof):

This axis consists of a linear joint with stepper motor attached to it. The location of limit switches is shown in Figure 4.7.

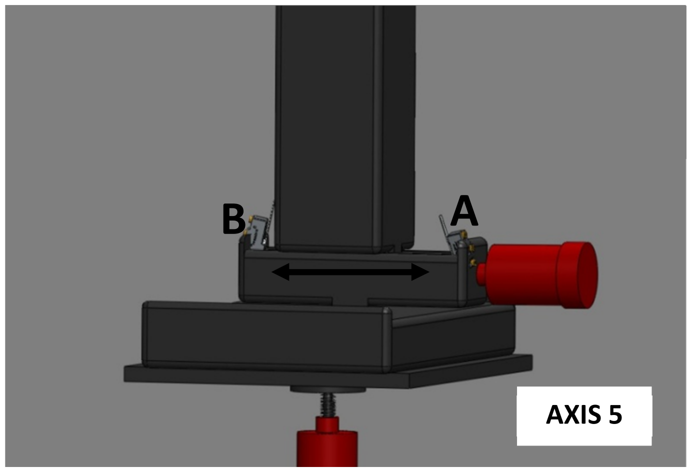


Figure 4.7 – Limit switches for Axis 5

The stepper motor is allowed to move until the slide hits the limit switch A. And then it is moved in the opposite direction to trigger the limit switch B. This is repeated for 50 times. The results are shown in Table 4.3 and Figure 4.8.

Table 4.3 – Backlash errors for Axis 5

NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS
1	919423	11	919674	21	919649	31	919756	41	919756
2	-919587	12	-919687	22	-919655	32	-919759	42	-919778
3	919590	13	919679	23	919674	33	919780	43	919801
4	-919722	14	-919690	24	-919693	34	-919778	44	-919824
5	919671	15	919634	25	919703	35	919792	45	919826
6	-919724	16	-919657	26	-919698	36	-919796	46	-919812
7	919649	17	919623	27	919714	37	919782	47	919814
8	-919685	18	-919643	28	-919722	38	-919782	48	-919833
9	919614	19	919668	29	919749	39	919820	49	919818
10	-919654	20	-919671	30	-919759	40	-919831	50	-919820

As seen from the table above, the maximum difference between any two consecutive readings is 164 steps. Average value of backlash error of 50 readings is 26.79 steps.

A graphical representation of the range of backlash error is shown below.

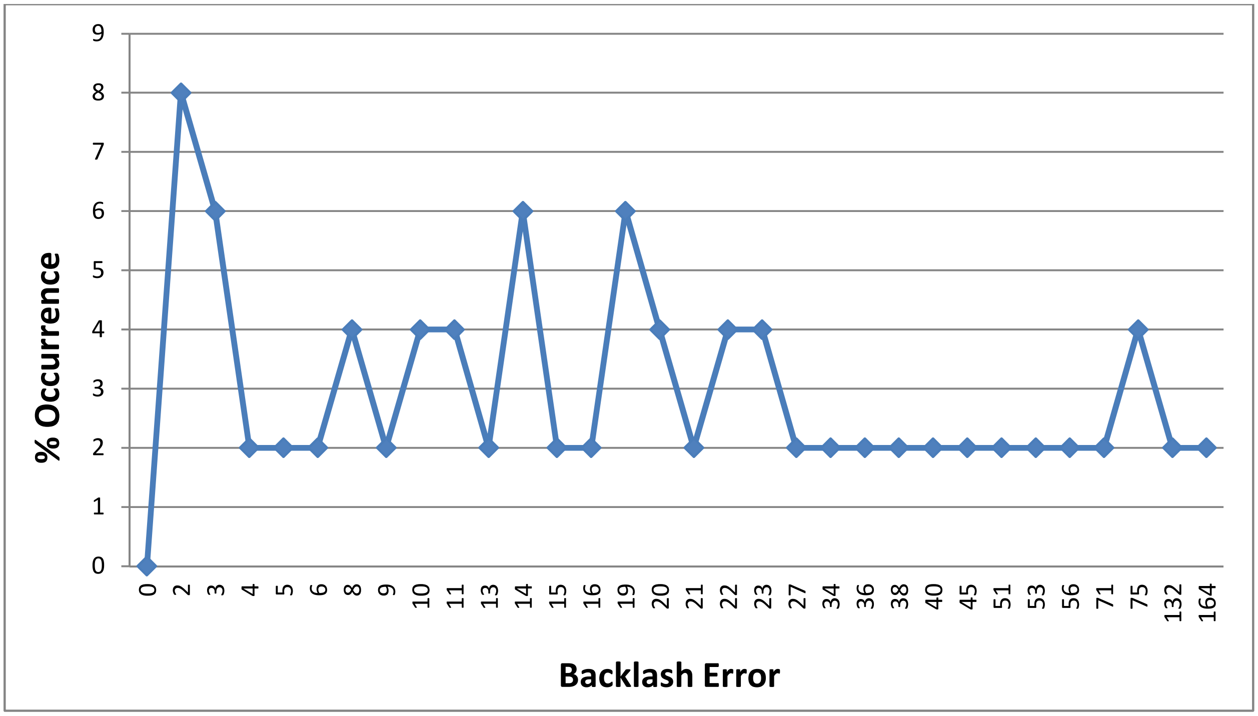


Figure 4.8: Backlash Error for Axis 5

Axis 6 (Translational dof):

This axis also consists of a linear joint with stepper motor attached to it. This axis is used to adjust the height of the manipulator. The limit switches are installed at strategic locations and the study is repeated. Results are shown in Table 4.4 and Figure 4.9.

Table 4.4 – Backlash errors for Axis 6

NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS
1	23158	11	23131	21	23120	31	23106	41	23106
2	-23158	12	-23131	22	-23118	32	-23108	42	-23104
3	23153	13	23124	23	23114	33	23109	43	23100
4	**-23149**	14	-23122	24	-23116	34	-23112	44	-23099
5	**23141**	15	23125	25	23113	35	23108	45	23100
6	-23142	16	-23125	26	-23112	36	-23107	46	-23102
7	23135	17	23124	27	23113	37	23108	47	23100
8	-23135	18	-23122	28	-23115	38	-23107	48	-23101
9	23130	19	23117	29	23112	39	23107	49	23102
10	-23130	20	-23120	30	-23110	40	-23104	50	-23103

As seen from the table above, the maximum difference between any two consecutive readings is 8 steps. Average value of backlash error for 50 readings is 2.26 steps.

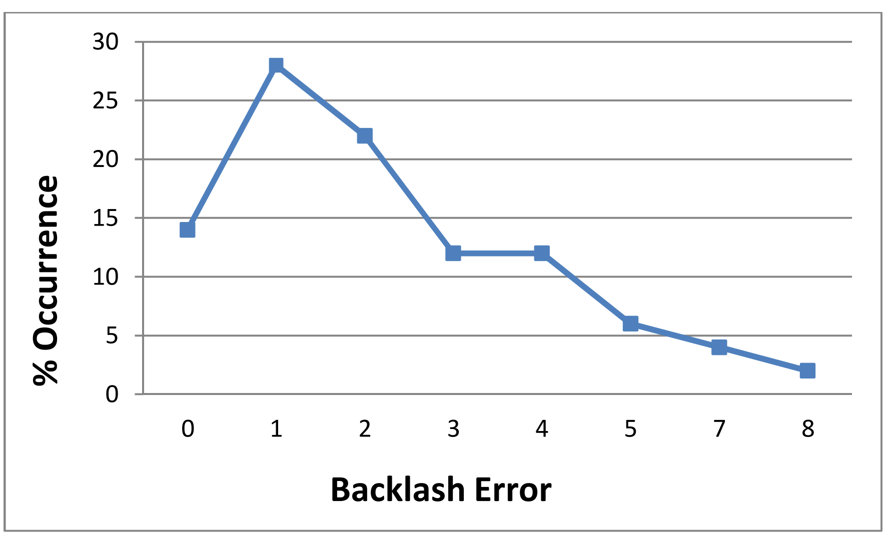


Figure 4.9: Backlash Error for Axis 6

Axis 7 (Translational dof):

This axis consists of a rotational joint powered by a stepper motor. This provides the roll motion for the end-effector.

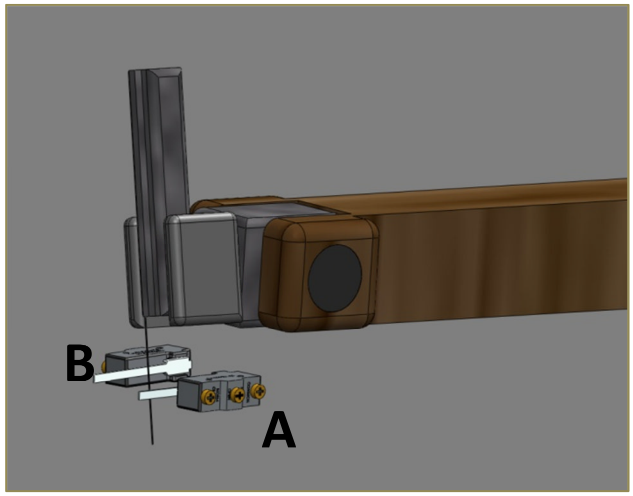


Figure 4.10 – Limit switches for Axis 7

As in the previous cases, the stepper motor is allowed to move until the end-effector triggers each limit switch as shown in Figure 4.10. The procedure is exactly identical as in previous measurements. Results are shown in Table 4.5 and Figure 4.11.

Table 4.5 – Backlash errors for Axis 7

NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS
1	-155	11	-159	21	-160	31	-160	41	-160
2	156	12	160	22	160	32	160	42	161
3	-157	13	-160	23	-160	33	-160	43	-161
4	157	14	160	24	160	34	160	44	161
5	-157	15	-160	25	-162	35	-160	45	-162
6	158	16	160	26	160	36	160	46	160
7	-157	17	-161	27	-160	37	-160	47	-161
8	158	18	159	28	160	38	160	48	161
9	-159	19	-160	29	-160	39	-160	49	-161
10	159	20	160	30	160	40	160	50	161

As seen from the table above, the maximum difference between any two consecutive readings is 3 steps. Average value of backlash error for 50 readings is 0.408 steps.

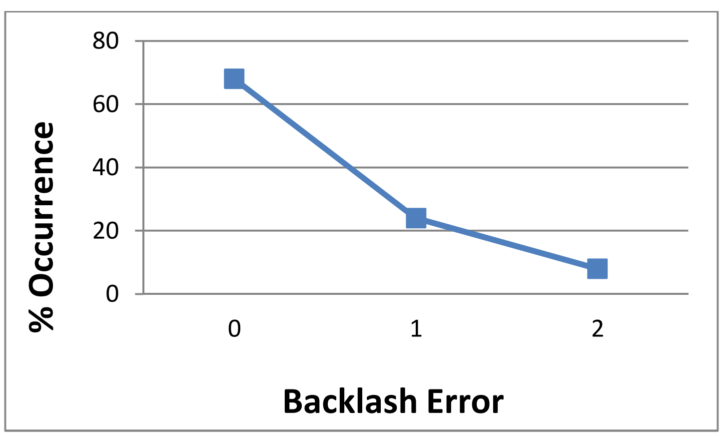


Figure 4.11 - Backlash Error for Axis 7

Axis 8 (Translational dof):

This axis also consists of a rotational joint powered by a stepper motor. This axis provides the pitch motion for the end-effector.

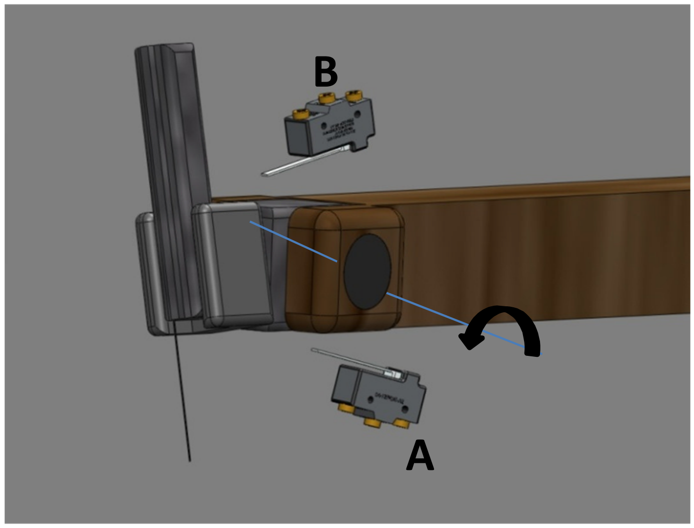


Figure 4.12 - Limit switches for Axis 8

As in the previous cases, the stepper motor is allowed to move until the end-effector hits the limit switch A as shown in Figure 4.12. It is then moved in the opposite direction to trigger the limit switch B. This is repeated for 50 times. Results are shown in Table 4.6 and Figure 4.13.

Table 4.6 – Backlash errors for Axis 8

NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS	NO.	ENCODER COUNTS
1	5080	11	5022	21	4942	31	4968	41	4938
2	-5087	12	-5024	22	-4942	32	-4970	42	-4943
3	4949	13	4967	23	4932	33	4972	43	4943
4	-4953	14	**-4968**	24	-4931	34	-4974	44	-4944
5	4946	15	**5122**	25	4939	35	4982	45	4944
6	-4952	16	-5113	26	-4940	36	-4982	46	-4945
7	4993	17	5019	27	4940	37	4948	47	4945
8	-5000	18	-5018	28	-4942	38	-4949	48	-4950
9	4938	19	4956	29	4943	39	4936	49	4948
10	-4943	20	-4957	30	-4944	40	-4939	50	-4948

The table shows that the maximum difference between any two consecutive readings is 144 steps. Average value of backlash error for 50 readings is 14.89 steps. A graphical representation of the range of backlash error is shown below.

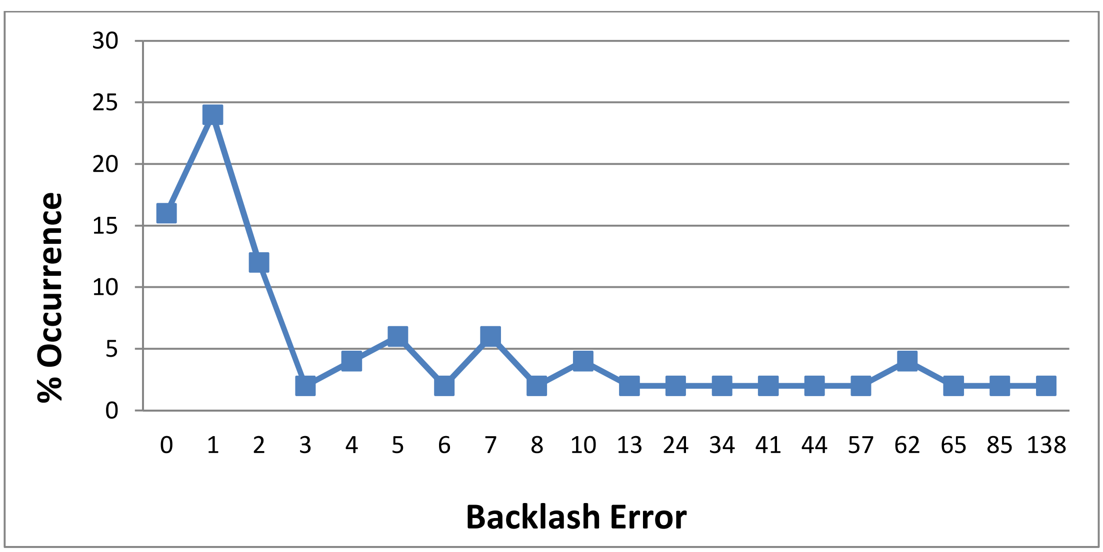


Figure 4.13 - Backlash Error for Axis 8

The backlash errors have been measured in terms of encoder counts and number of steps. The next step is to calculate it in terms of degrees in case of rotational joints and in terms of distance for translational joints.

All the rotational joints are directly connected to the motors. Also, in case of axis 8 (pitch motion), the bevel gears have transmission ratio of 1:1, which results in a direct drive. Therefore, for axes 7 and 8 that have stepper motors attached to it, the resolution would be 400 steps/rev. and 25,000 steps/rev. respectively. For axis 1, which consists of a servo motor, the resolution would be 7200 encoder counts per 90º.

A dial gauge has been used to find the resolution of the axes 2, 5 and 6. These axes have linear motions and the dial gauge can be securely placed on the table to measure the movements.

The resolution for all the six axes have been determined and tabulated below.

Table 4.7 – Resolution of the six axes

AXIS NUMBER	TYPE OF MOTION	MOTOR	RESOLUTION
1	Rotational/yaw	Servo	7200 encoder counts = 90° 1 encoder count = 0.0125°
2	Linear/prismatic	Servo	400 encoder counts (1 rev.) = 1.8034 mm 1 encoder count = 0.0045085 mm
5	Linear/prismatic	Stepper	25000 steps (1 rev.) = 1.905 mm 1 step = 7.62×10^{-5} mm
6	Linear/prismatic	Stepper	1000 steps = 2.4638 mm 1 step = 0.0024638 mm
7	Rotational/roll	Stepper	400 steps = 360° 1 step = 0.9
8	Rotational/pitch	Stepper	25000 steps = 360° 1 step = 0.0144

Now that the motor movements have been established in terms of degrees and distances, backlash errors can also be expressed in the same terms.

Table 4.8 – Backlash errors in terms of distances and degrees

Axis number	Backlash error (encoder counts/steps)	Backlash error (degrees/distance)
1	0.5306	0.0066°
2	1.081	0.00487 mm
5	26.79	0.00204 mm
6	2.26	0.00557 mm
7	0.408	0.3672°
8	14.89	0.2144°

As seen from the table above, positioning repeatability for the three prismatic degrees of freedom is less than 0.006 mm and the rotation angle repeatability is less than 0.4°.

4.3 ACCURACY OF THE SYSTEM

After establishing the accuracy and repeatability tests, the next step is to evaluate the kinematics developed in this thesis. The accuracy is defined as the difference between the target that the user selects and the position that the tip of the instrument really reaches. For some operations such as the therapy of Parkinson's disease, the application's accuracy in the range of tenth of mm is essential. But for majority of other clinical operations such as tumour therapy, tissue biopsy and particles placement, the application accuracy of 2-3 millimeters is acceptable [10]. The needle placement accuracy of the proposed robotic system itself is targeted to be less than 1 mm in free space. The actual accuracy of the complete system is expected to be somewhat less when registration errors and mechanical deflection are introduced.

4.3.1 SETTING HOME POSITIONS

Before the accuracy of the complete system can be tested, the home positions for all the axes have to be set. To set the home positions simple limit switches and micro roller switches have been located at pre-determined home positions of various dof of the robot. For the linear axes 2 and 5, installing micro roller switches were fairly straight forward as shown in the Figure 4.14.

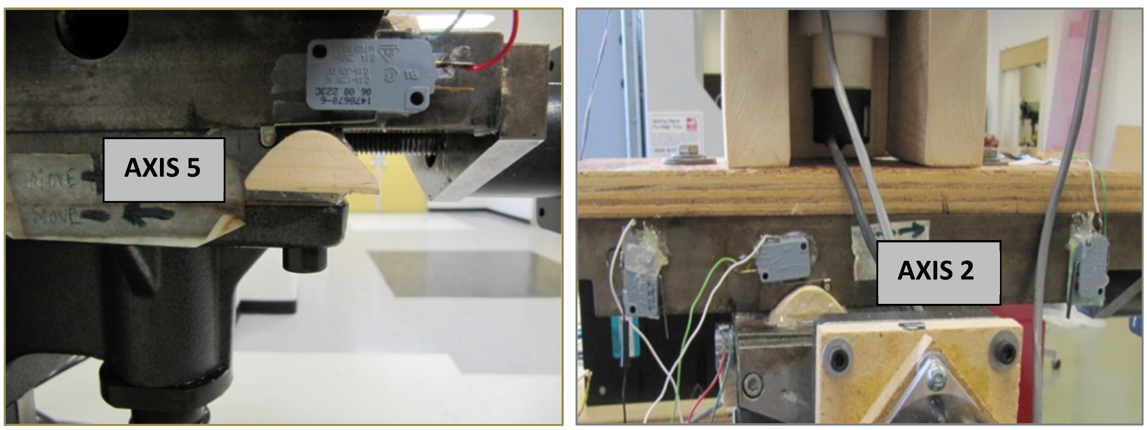
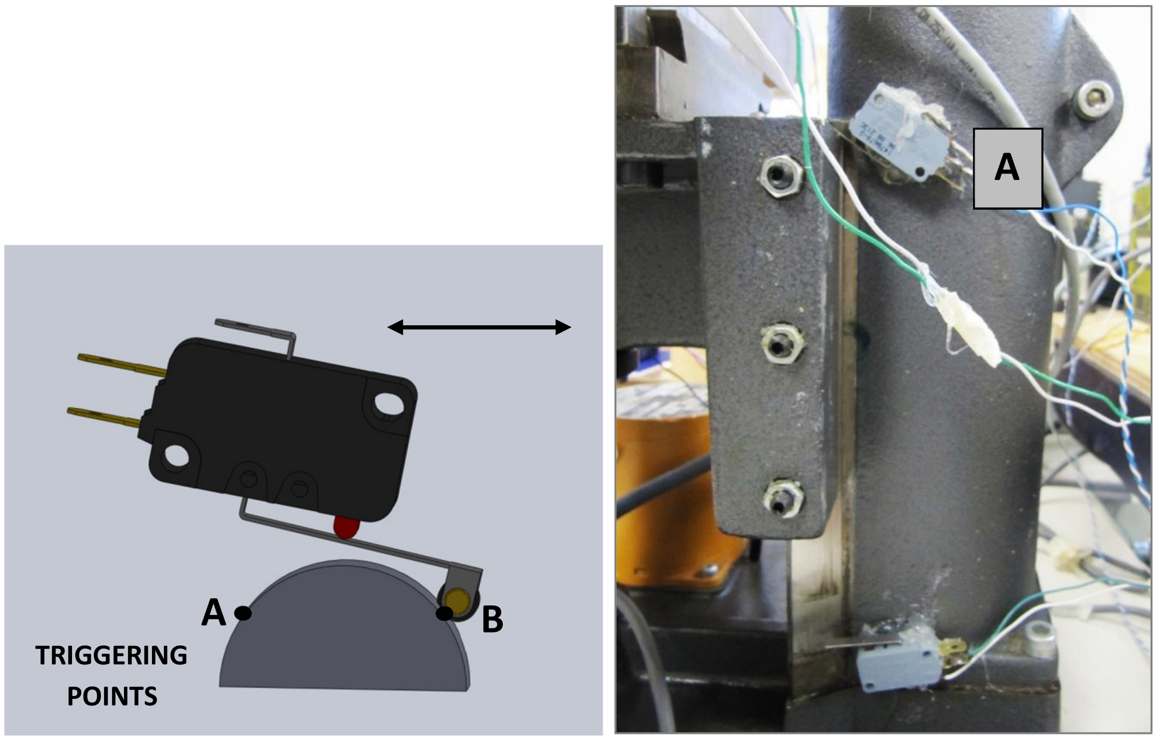


Figure 4.14 – Home positions for axes 2 and 5

When a roller switch is triggered, the corresponding motor stops and this sets the home position for that particular axis. The drawback of the roller switch is that it can be triggered at two points as shown in the Figure 4.15 (a). When the motor moves towards the left hand side the roller switch is triggered at point B and similarly when it moves towards the right hand side the roller switch is triggered at point A. To overcome this problem, the motor is set to stop only when it is moving in a particular direction and the roller switch is triggered. For example, for axis 5, the slide is manually moved to the right side of the curved wedge and then the slide is made to move towards the negative direction. As the slide moves the roller switch gets triggered at a point and the motor stops. This sets the home position for axis 5.



(a) (b)

Figure 4.15 – (a) Drawback of using a roller switch; (b) Home position for axis 6

Similar procedure has been adopted for setting the home position of axis 2. For axis 6, a simple limit switch has been used as shown in Figure 4.15 (b). When the switch A is triggered the motor stops.

For axis 1 (rotating arm), setting the home position is more involved. This arm has to be perfectly parallel with the slide of axis 2. For this task a hanging bob has been used. The bob is attached to the centre of the arm and reaches the small wooden plate as shown in the Figure 4.16. The rotating arm needs to be parallel to marked line on the plate which extends from the slide of axis 2. The rotating arm is manually moved to align the tip of the bob reach the marked line on the wooden plate. This position of rotating arm at which they meet would be the home position of axis 1. To set this position a roller switch has been installed similar to that of axis 5 (Figure 4.17(a)).

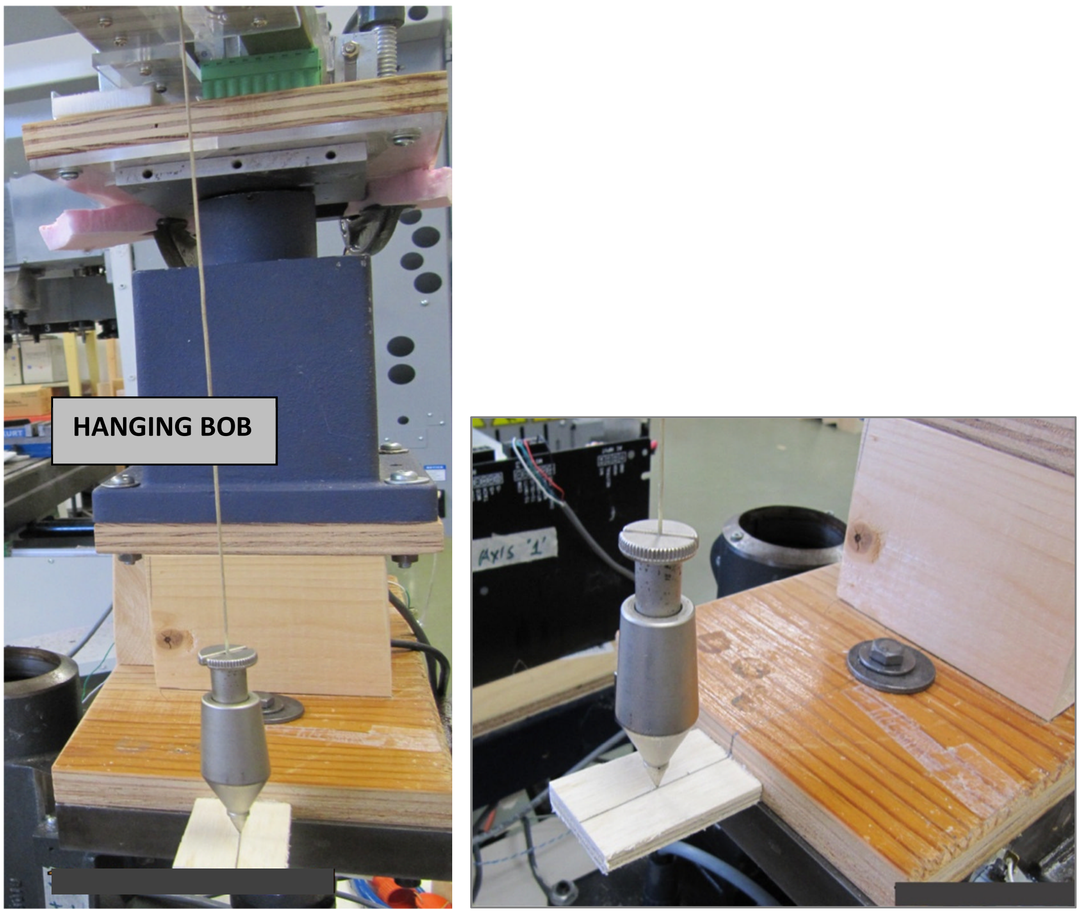
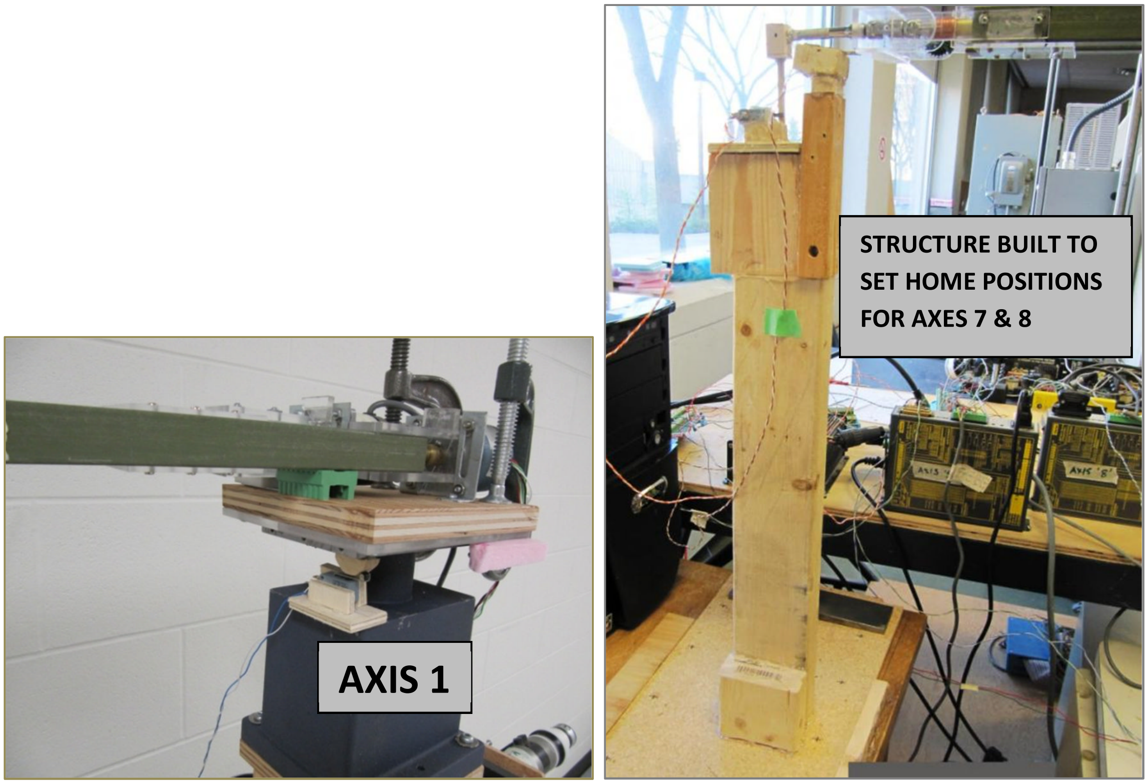


Figure 4.16 – Hanging bob used to set home position for axis 1

For axes 7 and 8, a separate structure has been built to set the home positions for both the axes simultaneously as shown in Figure 4.17(b). Limit switches has been used for both the axes. When the switches are triggered the motors stops which then sets the home positions for the axes. The built structure is detached from the system after each time the home position is set for axes 7 and 8.



(a) (b)

Figure 4.17 – (a) Home position for axis 1; (b) Structure built for axes 7 & 8.

After the home position for all the axes is set, testing the accuracy of the complete system can be carried out. Experiments were conducted on the precision of the robot’s motion by fabricating test sample structures as shown in the Figure 4.18. The samples structures contained various points in space that the robot end-effector is programmed to reach.

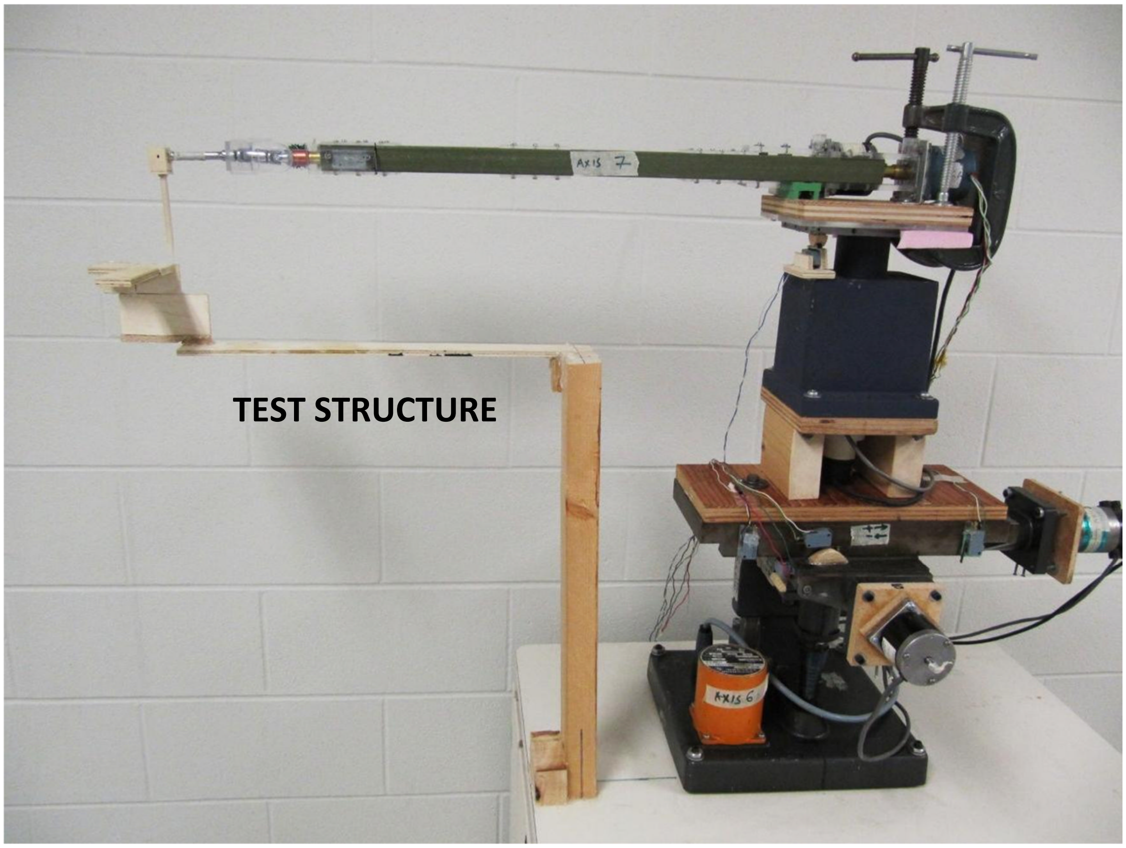


Figure 4.18 – Test Structures

The procedure employed to test the accuracy of the robot is as follows:

1) User is prompted to input the skin entry point (x_2, y_2, z_2) and the final target point (x_1, y_1, z_1) the needle is supposed to reach. These values are entered as data in a custom developed C++ program. As mentioned in Section 3.7.4.1, the needle tip should be at a certain safe distance away from the brain. The distance in this case is chosen as 2 cm. This distance would act as a buffer for any misalignment. So, in reality the needle tip has to reach to a point (x_3, y_3, z_3) which is 2 cm away from point (x_2, y_2, z_2) and lies in the same line as formed by points (x_1, y_1, z_1) and (x_2, y_2, z_2). The C++ program then computes the coordinates of points A (x_3, y_3, z_3) and B (x_4, y_4, z_4).

2) After the coordinates of points A and B are known, a pre-fabricated test sample structure that contains the coordinates of point A is employed to verify the position robot will move to. This is the point where the needle tip is supposed to reach.

3) The next step is to command the robot to move to the point A. This is carried out by another C++ program as discussed in Section 3.7.4.

4) After the robot has completed its motion, the next step is to analyse the inaccuracies associated with its motion. The robot has three translational degrees of freedom in x, y and z direction. If the needle falls short of the target point, the robot is manually moved in the appropriate direction to reach the final point. The distance of travel needed in x, y and z direction to reach this point would be the inaccuracy of the system in the respective directions.

The testing of the system is carried out at the highest speeds the motor can run without slipping. This will give the maximum inaccuracies of the system since the subsequent vibrations at high speeds would deteriorate the performance of the system. This will also ensure that the actual accuracy of the system would be much better than what is obtained from the following tests.

The speeds are set to the maximum by trial and error and any speeds above this would make the motor slip. Speeds are entered in a custom made dialogue box for each axis as shown below in Figure 4.19.

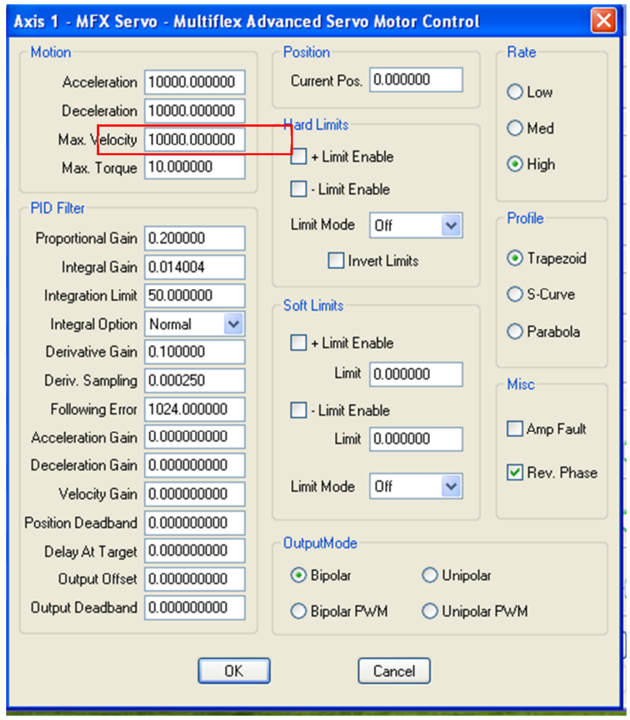


Figure 4.19 - Determining maximum velocities for each axis

The maximum velocity shown above is expressed in terms of encoder counts/sec in case of servo motors and steps/sec in case of stepper motors. This study will help identify the speeds at which the robot can work. The robot has 3 translational and 3 rotational degrees of freedom. All the degrees of freedom have different speeds they can work at without slipping.

Rotational degrees of freedom (Orientation)

Axis 1: Yaw (Servo motor)

Acceleration: 1000 encoder counts per second per second

Deceleration: 1000 encoder counts per second per second

Maximum Velocity: 1000 encoder counts per second

From Table 4.7, resolution of Axis 1,

1 encoder count = 0.0125°

Thus, speed of Axis 1 in terms of degrees would be = 1000 x 0.0125 = **12.5° per sec.**

Axis 7: Roll (Stepper motor)

Acceleration: 500 steps per second per second

Deceleration: 500 steps per second per second

Maximum Velocity: 50 steps per second

From Table 4.7, resolution of Axis 7,

1 step = 0.9°

Thus, speed of Axis 7 in terms of degrees would be = 50 x 0.9 = **45° per sec.**

Axis 8: Pitch (Stepper motor)

Acceleration: 1000 steps per second per second

Deceleration: 1000 steps per second per second

Maximum Velocity: 1000 encoder counts per second

From Table 4.7, resolution of Axis 1,

1 step = 0.0144°

Thus, speed of Axis 8 in terms of degrees would be = 1000 x 0.0144 = **14.4° per sec.**

Translational degree of freedom (Positioning)

Axis 2: Translation in Z direction (Servo motor)

Acceleration: 10000 encoder counts per second per second

Deceleration: 10000 encoder counts per second per second

Maximum Velocity: 500 encoder counts per second

From Table 4.7, resolution of Axis 2,

1 encoder count = 0.0045085 mm

Thus, speed of Axis 2 in terms of mm would be = 500 x 0.0045085 = **2.25 mm per sec.**

Axis 5: Translation in Y direction (Stepper motor)

Acceleration: 20000 steps per second per second

Deceleration: 20000 steps counts per second per second

Maximum Velocity: 20000 steps per second

From Table 4.7, resolution of Axis 5,

1 step = 7.62×10^{-5} mm

Thus, speed of Axis 5 in terms of mm would be = 20000 x 7.62×10^{-5} = **1.524 mm per sec**.

Axis 6: Translation in X direction (Stepper motor)

Acceleration: 10000 encoder counts per second per second

Deceleration: 10000 encoder counts per second per second

Maximum Velocity: 1000 encoder counts per second

From Table 4.7, resolution of Axis 6,

1 step = 0.0024638 mm

Thus, speed of Axis 1 in terms of degrees would be = 1000 x 0.0024638 = **2.4638 mm per sec.**

4.3.2 TEST STRUCTURES

Five different test structures have been built to test the accuracy of the system at different points in space. The points are chosen so as to cover the complete range of the robot. Moreover, as the distance of travel for the robot is increased, any error associated with a movement would become more observable.

Test structure 1:

User input values (in cm):

Skin Entry Point	Target point
$X_2 = 60$	$X_1 = 50$
$Y_2 = 3$	$Y_1 = 4$
$Z_2 = 78$	$Z_1 = 80$

These values are fed to the first C++ program. The actual values that the needle is supposed to reach considering a safe distance of 2 cm away from the brain are:

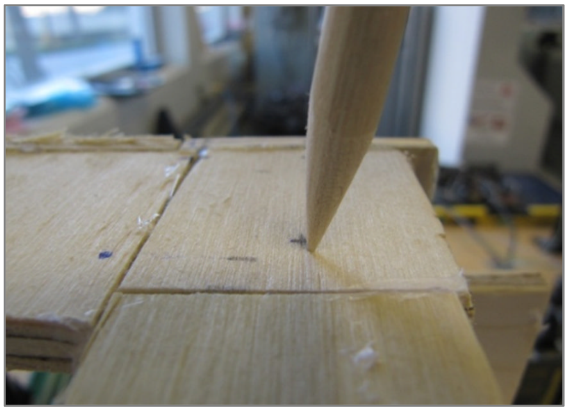
$X_4 = 72.36$	$X_3 = 61.95$
$Y_4 = 1.763$	$Y_3 = 2.80$
$Z_4 = 75.52$	$Z_3 = 77.60$

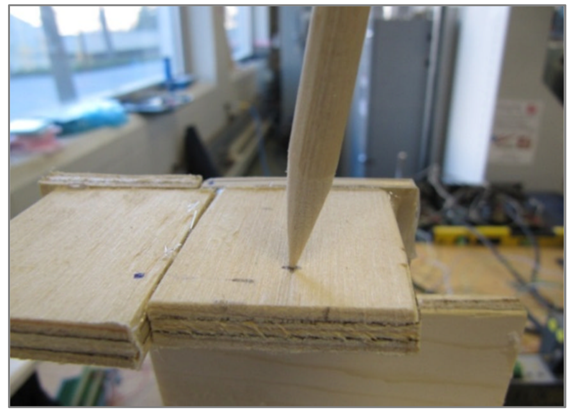
The distance of travel for each degree of freedom to reach to the final target point is also established from this C++ program. The values of these variable parameters are as follows:

Variable	Distance of travel
D_1	0.257 cm
D_2	-1.8 cm
D_3	-3.919 cm
θ_4	0º
θ_5	-11.31º
θ_6	-5.6º

The values of the above parameters are automatically fed to the second C++ program. This program physically moves the motor to the above distances. As seen from the Figure 4.20 (a), the needle tip is slightly off from the target point. At this point the robot is manually moved to appropriate distances to reach the final target point as shown in Figure 4.20 (b).

As mentioned above the distance of travel needed in x, y and z direction to reach to this point would be the inaccuracy of the system in the respective directions.





(a) (b)

Figure 4.20 – (a) Errors in needle positioninig for test structure 1; (b) Final target point

Table 4.9 – Errors for test structure 1

Direction of motion	**Axis/Motor number**	**Distance moved**	**Conversion factor**	**Error in millimeter**
X	6 - Stepper	Nil	405 steps = 1 mm	Nil
Y	5 - Stepper	19250 steps	13123 steps = 1 mm	1.466
Z	2 - Servo	270 encoder counts	222 encoder counts = 1 mm	1.217

Test Structure 2:

User input values (in cm):

Skin Entry Point	**Target point**
$X_2 = 60$	$X_1 = 50$
$Y_2 = 6$	$Y_1 = 8$
$Z_2 = 80$	$Z_1 = 82$

These values are again fed to the first C++ program. The actual values that the needle is supposed to reach considering a safe distance of 2 cm away from the brain are:

$X_4 = 72.19$	$X_3 = 61.92$
$Y_4 = 3.56$	$Y_3 = 5.62$
$Z_4 = 77.56$	$Z_3 = 79.62$

The values of the variable parameters are as follows:

Variable	**Distance of travel**
D_1	2.288 cm
D_2	-3.401 cm
D_3	-4.097 cm
θ_4	-0.122º
θ_5	-11.33º
θ_6	-11.07º

Table 4.10 – Errors for test structure 2

Direction of motion	**Axis/Motor number**	**Distance moved**	**Conversion factor**	**Error in millimeter**
X	6 - Stepper	Nil	405 steps = 1 mm	Nil
Y	5 - Stepper	22500 steps	13123 steps = 1 mm	1.714
Z	2 - Servo	250 encoder counts	222 encoder counts = 1 mm	1.127

Test Structure 3:

User input values (in cm):

Skin Entry Point	**Target point**
$\mathbf{X_2 = 60}$	$\mathbf{X_1 = 50}$
$\mathbf{Y_2 = -5}$	$\mathbf{Y_1 = -6}$
$\mathbf{Z_2 = 79}$	$\mathbf{Z_1 = 80}$

Table 4.11 – Errors for test structure 3

Direction of motion	**Axis/Motor number**	**Distance moved**	**Conversion factor**	**Error in millimeter**
X	6 - Stepper	200	405 steps = 1 mm	0.615
Y	5 - Stepper	11000 steps	13123 steps = 1 mm	0.838
Z	2 - Servo	300 encoder counts	222 encoder counts = 1 mm	1.35

Test Structure 4:

User input values (in cm):

Skin Entry Point	Target point
$X_2 = 62$	$X_1 = 52$
$Y_2 = -8$	$Y_1 = -10$
$Z_2 = 80$	$Z_1 = 82$

Table 4.12 – Errors for test structure 4

Direction of motion	Axis/Motor number	Distance moved	Conversion factor	Error in millimeter
X	6 - Stepper	Nil	405 steps = 1 mm	Nil
Y	5 - Stepper	5250 steps	13123 steps = 1 mm	0.400
Z	2 - Servo	190 encoder counts	222 encoder counts = 1 mm	0.86

Test Structure 5:

User input values (in cm):

Skin Entry Point	Target point
$X_2 = 62$	$X_1 = 52$
$Y_2 = -4$	$Y_1 = -2$
$Z_2 = 78$	$Z_1 = 80$

Table 4.13 – Errors for test structure 5

Direction of motion	Axis/Motor number	Distance moved	Conversion factor	Error in millimeter
X	6 - Stepper	Nil	405 steps = 1 mm	Nil
Y	5 - Stepper	2600 steps	13123 steps = 1 mm	0.198
Z	2 - Servo	180 encoder counts	222 encoder counts = 1 mm	0.811

From the results of the accuracy tests, it is seen that the positioning errors are between 1-3 mm. The accuracy of the complete system was initially targeted to be less than 1 mm. Limitations of the system such as manufacturing imprecision, joint position errors from the controller, robot actuation, and human errors account for the error seen in positioning. Also, the testing was done at the highest speeds possible for this robot. Resulting vibrations also account for the inaccuracies in the system. Though the accuracy obtained from this robotic system is acceptable for a large amount of operations such as tumour therapy, tissue biopsy and particles placement, accuracies below 1 mm is also possible with this system if all the limitations are rectified.

4.4 SUMMARY

In this chapter the overall performance of the robotic system has been evaluated. First, the mechanical aspects of the robot design are tested by moving each axis back and forth 50 times to determine the backlash error. This was done so that it would be known if errors in the surgical systems motion could be attributed to the robot arm. After it was identified that the robot has minimal error associated with its positioning repeatability, analysis was conducted on the kinematics developed in this thesis. Experiments were also performed to verify that the control algorithm would command the robot to move to the correct position. Five different test structures were built to test the accuracy of the system to reach different points in space. The points were chosen as to cover the complete range of the robot. The average accuracy of robotic system was found to be less than 3 mm. The inaccuracies in the system were mainly due to manufacturing imprecision primarily of the bevel gears and universal joint, joint position errors from the controller, and position registration errors.

CHAPTER 5

DISCUSSION

The objective of this research was to design a PC based neurosurgical robot for biopsy with potential applications in MRI environment. To this date a clear understanding has been established regarding the requirements for designing a neurosurgical robot to work inside a MRI scanner. This background was essential to understand how to design a robot for use in a MRI scanner. Most of the MRI compatible robots developed in literature so far had been scanner specific. This work proposed a design that can be adapted for both open and closed scanners. This feature is particularly important in sites that already have a scanner installed in place. The cost of a MRI machine is in the range of 2-3 million dollars. Thus, it is imperative that ability to adapt to different types of MRI scanners is built into the design of the robot. The proposed robot consists of a base unit, which resides outside the scanner (open or closed) and hosts all the mechatronic subassemblies, and an articulated arm which extends inside the MRI bore and manoeuvres above the patient to facilitate the positioning of an interventional tool along a user defined trajectory.

Developing a novel design that can allow the placements of actuators outside the MRI bore has been the next contribution of this work. The proposed robot has been designed for using ultrasonic piezoelectric motors as actuators. Researchers have found that the encasings of ultrasonic motors contained conductive material that results in image artifacts. To address the compatibility issue, this research work proposed a design which allows the placement of actuators outside the magnetic bore. A gearing arrangement has been used to transmit the motion from the motors to the end-effector.

Another important design feature of this robot is that the end-effector has an additional degree of freedom: needle insertion, which is independent of the other degrees of freedom. The needle insertion is done by the surgeon. Although, technically the robot can perform the insertion, the current plan reserves this task to the surgeon owing to safety, ethical and legal considerations. It is assumed that incisions have already been made in the brain for the needle to go through and remove a sample of tissue to test for tumour cells. This assumes that the needle will primarily insert through soft tissues only.

By employing the strategy of having an additional degree of freedom at the end-effector, various surgical tools can be utilized as end-effectors. This gives the surgeon flexibility to use as an end-effector; a biopsy needle, electrode or syringe for injection of drugs.

After the design features were ascertained, a study on the kinematics of the robot was done. The robot was modelled using Denavit Hartenberg conventions for direct kinematics. Due to the unconventional nature of the robot geometry an algorithm has been developed upon which the inverse kinematics has been based. The inverse kinematics in this case was different since the end-effector has to reach the target point, with its orientation parallel to the path of insertion of needle.

The prototype neurosurgical robot was then built. Since this work was a proof-of-concept of the design and control of the proposed MRI compatible robot, the robot was fabricated with materials and actuators available in the research lab which were not MRI compatible.

A PC based control system was then developed. Commercially available controller card programmed in Visual Studio C++, was used to control the motors of the manipulator. The program commands the robot to manipulate its end effector (needle) to a prescribed insertion vector defined by the surgeon. The surgeon then fine adjusts it and inserts the needle/probe to the target using manual activation of the insertion dof.

After the complete system was set in place, qualitative testing of the robot was carried out. Test structures were built to assess the positional accuracy of the system. It was seen that the positional accuracy of the system was 1-3 mm. Though this accuracy is acceptable for a large amount of operations such as tumour therapy, tissue biopsy and particles placement, accuracies below 1 mm is also possible with this system if all the limitations are rectified. The inaccuracies in the system were mainly due to manufacturing imprecision primarily of the bevel gears and universal joint. This is evident from the backlash error Table 4.8 for the six axes. The backlash errors for axes 7 and 8 (which constitute bevel gears and universal joint) are more than 0.3º each for the complete range of its motion. If these are manufactured to greater tolerances the overall accuracy of the system would substantially increase. Other reason for the accuracies include joint position errors from the controller, robot actuation, and human registration errors, all of which can be easily rectified with proper resources.

CHAPTER

6

CONCLUSION & FUTURE WORK

6.1 OUTCOME OF THE WORK

This thesis has presented the design of a personal computer (PC) based needle insertion robotic manipulator for minimally invasive interventions in the brain. The device developed in this work was intended primarily for use inside the confined area of a cylindrical MR scanner. The required background for design in MR environment was discussed and the design process was outlined. The focus was on the development of appropriate geometry, determining strategic locations of actuators and development of appropriate PC interface. Since this work was a proof-of-concept of the design and control of the proposed system, readily available materials and actuators had been used to build the robot manipulator. The prototype has been thoroughly tested and has demonstrated the feasibility of such a device. The success of this project and suggestions for future work will be discussed in this chapter.

6.2 EVALUATION

The goal of this research was to design a PC based neurosurgical robot with potential application in MRI environment. The following tasks towards the goal have been accomplished. First, a clear understanding was established regarding the requirements for designing a neurosurgical robot to work inside a MRI scanner. For example, topics such as size limitations and MRI compatibility were explored in detail. This background was essential to understand how to design a robot for use in a MRI scanner. Most of the MRI compatible robots developed in literature so far had been scanner specific. This work proposed a design that can be adapted for both open and closed scanners.

The next contribution made in this work was developing a novel design that can allow the placements of actuators outside the MRI bore. With most of the systems developed so far, the preferred actuators had been the ultrasonic piezoelectric motors (USM). However, researchers had found that the encasings of ultrasonic motors contained conductive material that resulted in image artifacts. To address the compatibility issue, this research work proposed a design which allows the placement of actuators outside the magnetic bore.

An additional contribution of this work was developing a PC based control system for the robotic manipulator. An algorithm was developed upon which the complete control strategy was based. Also, the robot design was described in detail providing direct and inverse kinematics governing its motions.

Finally, the prototype neurosurgical robot was built and qualitatively tested. A variety of problems were fixed based on observations made during assembly and debugging. Additionally, suggestions were made for improvements that could be made to the robot design. Overall, the neurosurgical robot was functional and successful.

6.3 FUTURE WORK

There are several suggestions that can be made for future work in this project. Replicating the complete design with MRI compatible materials and actuators is the first important task that is required to be done. The control system would have to be modified according to the actuators and control card used, but the algorithm would remain the same. The next step would be a more rigorous testing in the laboratory to determine if the robot has met the functional requirements. After testing in the laboratory is complete a second set of testing inside the MRI scanner is required. This is to ensure that the robot is in fact MRI compatible and would function properly in the MRI environment. This needs cooperation with MRI manufacturers. Again design changes can be made as a result of testing in the MRI machine to improve the performance or compatibility of the robot. Once the complete system is set in place, the next step would be integrating the complete system with a MRI scanner.

Testing with phantoms is a next important step towards clinical use that should take place both outside and within the bore of a MRI machine. Control of the device and coordination with the MR images will need to be addressed before clinical trials can occur.

This device, once fully developed has the potential to improve or make possible other in-bore MRI procedures. Many treatments and procedures would benefit from the increased visual resolution of MRI, such as breast cancer biopsy and treatment, elastrography of the breast, liver and other organs to detect abnormalities.

The increased accuracy of the system will help find smaller tumours and allow for earlier, non-invasive treatments. Cancer patients will benefit from earlier diagnostics, more effective treatment, reduced recovery time and reduced risk of side effects.

BIBLIOGRAPHY

1. American Cancer Society, "Cancer Facts & Figures 2010", Website: http://www.cancer.org/acs/groups/content/@nho/documents/document/acspc-024113.pdf

2. American Brain Tumor Association, "Facts & Statistics, 2010", Website: http://www.abta.org/sitefiles/pdflibrary/ABTA-FactsandStatistics2010v3.pdf

3. Central Brain Tumor Registry of the United States 2010, Website: http://www.cbtrus.org/factsheet/factsheet.html

4. Canadian Cancer Society, "Canadian Cancer Statistics, 2010", Website: http://www.cancer.ca/~/media/CCS/Canada%20wide/Files%20List/English%20files%20heading/pdf%20not%20in%20publications%20section/Canadian20Cancer20Statistics2020102020English.ashx.

5. National Cancer Institute, "What you need to know about; Brain Tumours", Website: http://www.cancer.gov/cancertopics/wyntk/brain/page6/print.

6. Cedars-Sinai, "Brain Tumor and Brain Cancer", Website:

http://www.cedars-sinai.edu/Patients/Health-Conditions/Brain-Tumors-and-Brain-Cancer.aspx.

7. Brady S, Thornhill A and Colapinto E., "Stereotactic Biopsy Procedures for Brain Tumor Diagnosis", AORN, Vol. 65, pp. 890-900, 1997.

8. ct-scan-info.com, "Is MRI Better Than CT?", Website: http://www.ct-scan-info.com/mri.html.

9. Paul B. McBeth, M.A.Sc., Deon F. Louw, M.D., Peter R. Rizun, B.A.Sc.,Garnette R. Sutherland, M.D., "Robotics in neurosurgery", The American Journal of Surgery, Issue: 68S–75S, Vol. 188, 2004.

10. Junchuan Liu, Yuru Zhang, Zhen Li, "The Application Accuracy of NeuroMaster: a Robot System for Stereotactic Neurosurgery", Mechatronic and Embedded Systems and Applications, Proceedings of the 2nd IEEE/ASME International Conference, pp. 1 - 5, 2006.

11. Govindarajan, Anand, "Robot-Assisted Surgery : A Review", University of Toronto Medical Journal, Vol. 78, pp. 108-112, 2001.

12. Anthony R. Lanfranco, BAS, Andres E. Castellanos, MD, Jaydev P. Desai, PhD, and William C. Meyers, MD, "Robotic Surgery, A current perspective", Annals of Surgery, Vol. 239, pp. 14–21, ISBN: 0003-4932/04/23901-0014, 2004.

13. Matsuoka, Robert D. Howe and Yoky, "Robotics for Surgery", Harvard University, Cambridge, Massachusetts : Annual Review of Biomedical Engineering, Vol. 01, pp. 211–240, 1999.

14. The University of Chicago Medical Center, "Benefits of Minimally Invasive Procedures", Website: http://www.uchospitals.edu/specialties/minisurgery/benefits/index.html.

15. Intuitive Surgical, "Product Image Gallery: da Vinci Standard Surgical System", Website: http://www.intuitivesurgical.com/corporate/newsroom/mediakit/gallery_davincistandard.aspx.

16. Robotic Surgery, "Neurosurgery", Website: http://biomed.brown.edu/Courses/BI108/BI108_2005_Groups/04/neurology.html.

17. Nikolaos V. Tsekos, Azadeh Khanicheh, Eftychios Christoforou and Constantinos Mavroidis, "Magnetic Resonance– Compatible Robotic and Mechatronics Systems for Image-guided Interventions and Rehabilitation: A Review Study", The Annual Review of Biomedical Engineering, Vol. 9, pp. 351–387, 2007.

18. Coyne, Kristen "MRI: A Guided Tour. Magnet Lab, National High Magnetic Field Laboratory", Website: http://www.magnet.fsu.edu/education/tutorials/magnetacademy/mri/fullarticle.html.

19. Joseph P. Hornak, Ph.D.,"The Basics of MRI", Website: http://www.cis.rit.edu/htbooks/mri/inside.htm.

20. Wikipedia, "Magnetic Resonance Imaging", Website: http://en.wikipedia.org/wiki/Magnetic_resonance_imaging.

21. Ningbo Yu, Robert Riener, "Review on MR-Compatible Robotic Systems", Biomedical Robotics and Biomechatronics, BioRob 2006, First IEEE/RAS-EMBS International Conference, pp. 661 - 665, February 20-22 , 2006.

22. Eftychios G. Christoforou, Nikolaos V. Tsekos, Alpay Özcan, "Design and Testing of a Robotic System for MR Image-guided Interventions", Journal of Intelligent & Robotic Systems, Springer Science + Business Media B.V. 2006, Vol. 47, pp. 175-196, 2006.

23. Stoianovici, D., "Multi-imager compatible actuation principles in surgical robotics", The International Journal of Medical Robotics and Computer Assisted Surgery, Vol. 1, pp. 86-100, 2005.

24. B. Vigaru, D. Petrisor, A. Patriciu, D. Mazilu, D. Stoianovici, "MR compatible actuation for medical instrumentation", Cluj-Napoca: Automation, Quality and Testing, Robotics, 2008. AQTR 2008, IEEE International Conference, pp. 49 - 52, 2008.

25. John Vogan, Andreas Wingert, Jean-Sebastian Plante, Steven Dubowsky, Moustapha Hafez, Daniel Kacher, Ferenc Jolesz, "Manipulation in MRI devices using electrostrictive polymer actuators: with an application to reconfigurable imaging coils", Robotics and Automation, 2004, Proceedings ICRA '04 2004 IEEE International Conference , Vol. 3, pp. 2498 - 2504, 2004.

26. Khanicheh A., Muto A., Triantafyllou C., Weinberg B., Astrakas L., Tzika A. and Mavroidis C., "MR Compatible ERF Driven Hand Rehabilitation Device", Proceedings of the 2005 IEEE, 9th International Conference on Rehabilitation Robotics, pp. 7 - 12, 2005.

27. Ken Masamune, Etsuko Kobayashi, M.S., Yoshitaka Masutani, Makoto Suzuki, PH.D., Takeyoshi Dohi, PH.D., Hiroshi Iseki, M.D., and Kintomo Takakura, M.D., "Development of an MRI-Compatible Needle Insertion Manipulator for Stereotactic Neurosurgery", Journal of Image Guided Surgery, Wiley-Liss, Inc., Vol. 1, pp. 242-248, 1995.

28. Kucharczyk, Cyrus Raoufi and Andrew A. Goldenberg and Walter, "Control Paradigm and Design for a Novel MR-Compatible Tele-Robotic System for MRI-Guided Neurosurgery", Proceedings of the 2007 IEEE International Conference on Mechatronics and Automation, pp. 1445 - 1451, 2007.

29. Cyrus Raoufi, Andrew A. Goldenberg and Walter Kucharczyk, "A New Hydraulically/Pneumatically Actuated MR-Compatible Robot for MRI-Guided Neurosurgery", Bioinformatics and Biomedical Engineering, 2008 ICBBE, pp. 2232 - 2235, 2008.

30. Gregory Cole, Julie Pilitsis and Gregory S. Fischer, "Design of a Robotic System for MRI-Guided Deep Brain Stimulation Electrode Placement", 2009 IEEE International Conference on Robotics and Automation Kobe International Conference Center, pp. 4450 - 4456, 2009.

31. Yoshihiko Koseki, Toshikatsu Washio, Kiyoyuki Chinzei, and Hiroshi Iseki, "Endoscope Manipulator for Trans-nasal Neurosurgery, Optimized for and Compatible to Vertical Field Open MRI", T. Dohi and R. Kikinis (Eds.): MICCAI 2002, Springer-Verlag Berlin Heidelberg, pp. 114-121, 2002.

32. Hong Zaidi, Yun Chao, Zhao Lei and Wang Yang, "Design and Optimization Analysis of Open-MRI Compatibile Robot for Neurosurgery", Bioinformatics and Biomedical Engineering, ICBBE 2008, pp. 1773 - 1776, 2008.

33. Shelly Lwu, MD, MSc, Garnette R. Sutherland, MD., "The Development of Robotics for Interventional MRI", Neurosurgery Clinics of North America, Elsevier Inc., Vol. 20, pp. 193–206, 2009.

34. Mitsuhiro Oura, Yo Kobayashi, Jun Okamoto and Masakatsu G.Fujie, "Development of MRI Compatible Versatile Manipulator for Minimally Invasive Surgery", Biomedical Robotics and Biomechatronics, 2006, The First IEEE/RAS-EMBS International Conference, pp. 176 - 181, February 20-22, 2006.

35. Andreas Melzer, Bernd Gutmann, Thomas Remmele, Renate Wolf, Andreas Lukoscheck, Michael Bock, Hubert Bardenheuer and Harald Fischer, "INNOMOTION for Percutaneous Image-Guided Interventions", Engineering in Medicine and Biology Magazine, IEEE, Vol. 27, pp. 66 - 73, 2008.

36. Fujio Tajima, Kousuke Kishi, Kazutoshi Kan, Hiroshi Ishii, Kouji Nishizawa, Masakatsu G. Fujie, Takeyoshi Dohi, Ken-ichi Sudo, Shin-ichi Takamoto, "An MR-compatible master–slave manipulator with interchangeable surgical tools", International Congress Series, Computer Assisted Radiology and Surgery, Proceedings of the 17th International Congress and Exhibition, Elsevier Science B.V., Vol. 1256, pp. 529-537, 2003.

37. ABC News, "MRI Scanner Accidents on the Rise", Website: http://abcnews.go.com/GMA/OnCall/story?id=1057960&page=1.

38. Siemens, "The first Open-Bore MRI MAGNETOM Espree", Website: http://www.medical.siemens.com/siemens/it_IT/gg_mr_FBAs/files/brochures/Espree_Brochures/Espree_Brochure_Final_13103194_4.pdf.

39. Siemens, "MAGNETOM Espree 1.5T - Technical Specifications", Website: http://www.medical.siemens.com/webapp/wcs/stores/servlet/ProductDisplay~q_catalogId~e_-1~a_catTree~e_100010,1007660,12754,14330~a_langId~e_-1~a_productId~e_142246~a_storeId~e_10001~a_view~e_38.htm.

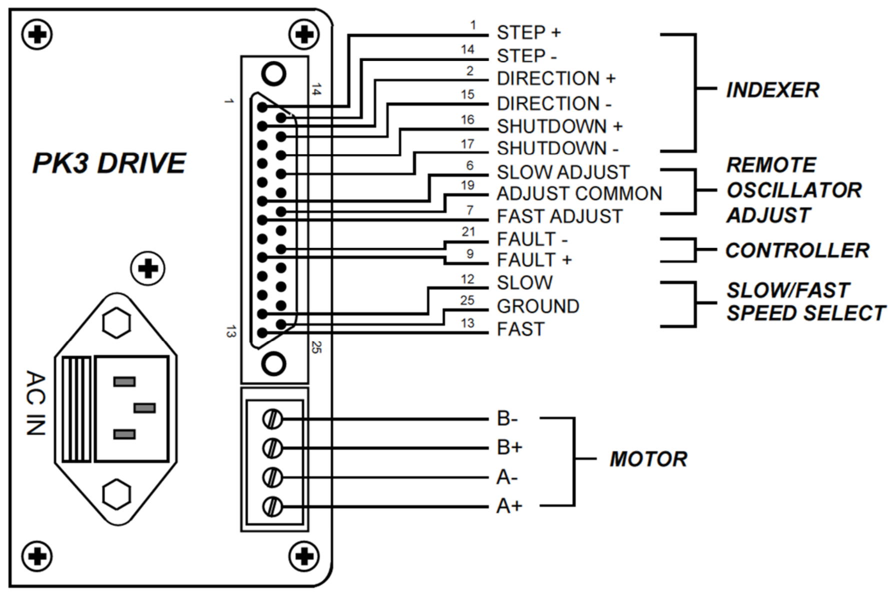
40. Yuji Wakasa, Masato Oka, Kanya Tanaka, Masami Fujii, Syuichi Yamauchi, and Kazuyuki Minami, "Development of a Needle-Insertion Robot for MRI-Guided Stereotactic Surgery", Journal of Robotics and Mechatronics, Vol. 18, pp. 643-649, 2006.

41. Craig, John J., "Introduction to Robotics: Mechanics and Control 3rd Edition", Upper Saddle River, NJ : Pearson Education International, 2005.

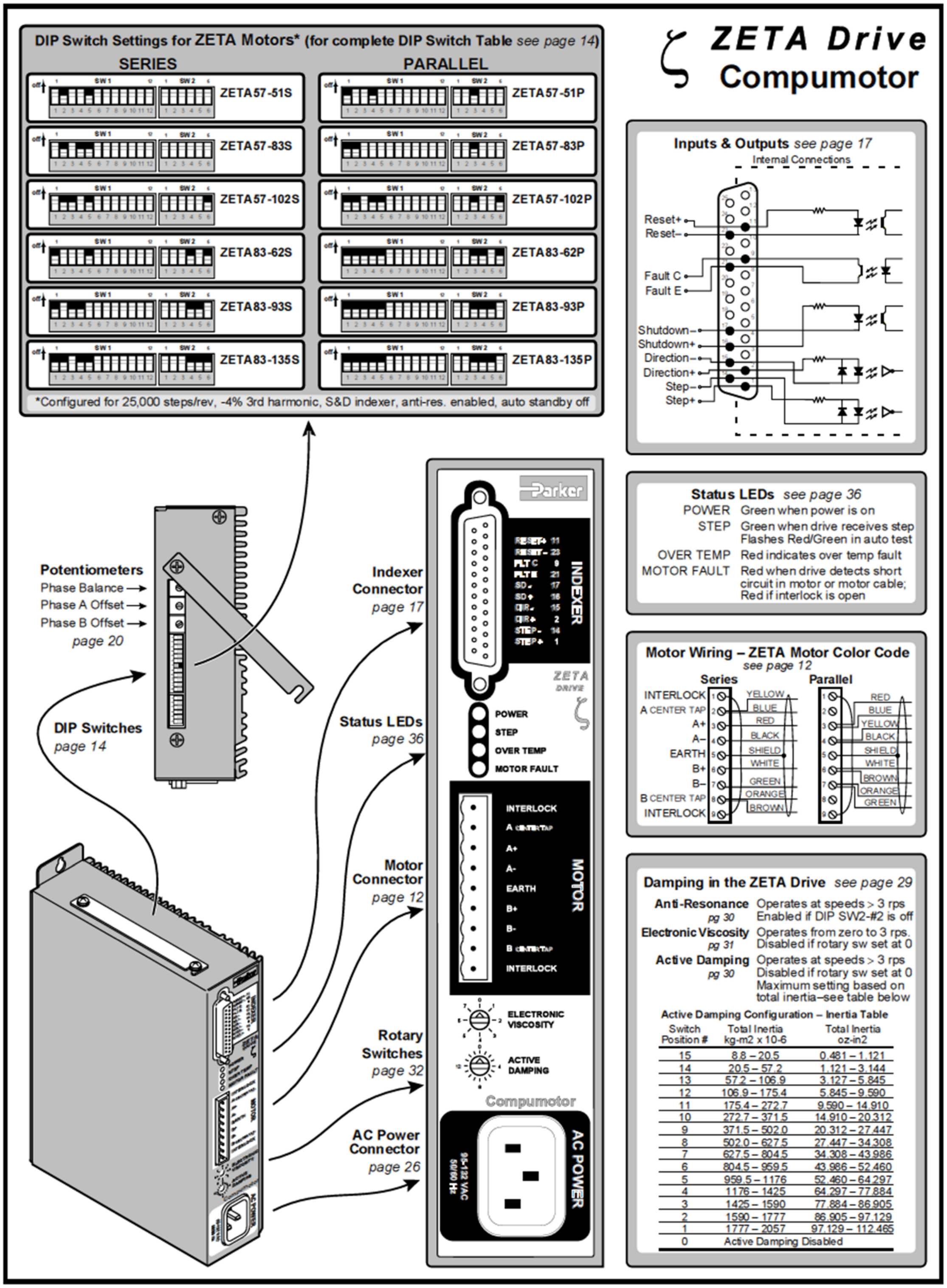
42. Nathoo, Narendra M.D., Ph.D., "In Touch with Robotics: Neurosurgery for the Future", Neurosurgery: the Congress of Neurological Surgeons, Vol. 56, pp. 421-433,2005.

APPENDIX

A.1 Stepper Motor Drives



The figure above illustrates the connectors on the PK3 front panel, and shows the pinouts for indexer and motor connections.



DIP switch settings and wirings for ZETA drive

A.2 Wirings for the break out board that connects to the controller card

Connector Pinout - *MFX-PCI1440*

Connector J1 (Analog Command Axes 1 & 2, Dig. inputs 1-4, Dig. outputs 1-4, A/D inputs 1 & 2)

VHDCI Pin #	*Digital I/O Channel #*	*Circuit Type*	*Description (default configuration)*	*Adam-3968 Pin #*
J1 - 1			Axis 1 Servo Command (+/- 10V)	1
J1 – 35			Axis 1 return / Analog Ground	35
J1 – 2			Axis 2 Servo Command (+/- 10V)	2
J1 – 36			Axis 2 return / Analog Ground	36
J1 – 3				3
J1 – 37				37
J1 – 4	53	Output - open collector driver	Axis 2 Amp. Enable output	4
J1 – 38			+5 VDC	38
J1 – 5	49	Output - open collector driver	Axis 1 Amp. Enable output	5
J1 – 39			+5 VDC	39
J1 – 6	51	Output - open collector driver	#1 PWM out / #1 Direction / #5 Full Current out	6
J1 – 40			+5 VDC	40
J1 – 7	55	Output - open collector driver	#2 PWM out / #2 Direction / #6 Full Current out	7
J1 – 41			+5 VDC	41
J1 – 8				8
J1 – 42				42
J1 – 9			+12 VDC	9
J1 – 43			+12 VDC	43
J1 – 10			Axis 1/2 Encoder Ref. (1.5V)	10
J1 – 44			Axis 1/2 Encoder Ref. (1.5V)	44
J1 – 11			Axis 1 Encoder Phase A+	11
J1 – 45			Axis 1 Encoder Phase A-	45
J1 – 12			Axis 1 Encoder Phase B+	12
J1 – 46			Axis 1 Encoder Phase B-	46
J1 – 13			Axis 1 Encoder Phase Z+	13
J1 – 47			Axis 1 Encoder Phase Z -	47
J1 – 14			Axis 2 Encoder Phase A+	14
J1 – 48			Axis 2 Encoder Phase A-	48
J1 – 15			Axis 2 Encoder Phase B+	15
J1 – 49			Axis 2 Encoder Phase B-	49
J1 – 16			Axis 2 Encoder Phase Z+	16
J1 – 50			Axis 2 Encoder Phase Z -	50
J1 – 17	20	Input - opto isolated (bi-	Axis 1 Amp. Fault input (shared by Axis 5 Drive	17
J1 – 51			Axis 1/5 Amp Fault supply / return	51
J1 – 18	24	Input - opto isolated (bi-	Axis 2 Amp. Fault input (shared by Axis 6 Drive	18
J1 – 52			Axis 2/6 Amp Fault supply / return	52
J1 – 19	33	Output - TTL	Digital Out #1 / Axis 1 - 4 Compare	19
J1 – 53			+5 VDC	53
J1 – 20	34	Output - TTL	Digital Output #2	20
J1 – 54			+5 VDC	54
J1 – 21	35	Output - TTL	Digital Output #3	21
J1 – 55			+5 VDC	55
J1 – 22	36	Output - TTL	Digital Output #4	22
J1 – 56			+5 VDC	56
J1 – 23	1	Input - TTL	Dig. In. #1 / Axis 1 & 2 Position Capture (Latch)	23
J1 – 57			Ground	57
J1 – 24	2	Input - TTL	Digital Input #2	24
J1 – 58			Ground	58
J1 – 25	3	Input - TTL	Digital Input #3	25
J1 – 59			Ground	59
J1 – 26	4	Input - TTL	Digital Input #4	26
J1 – 60			Ground	60
J1 – 27	17	Input - opto isolated (bi-	Axis 1 Coarse Home (shared by Axis 5 Home)	27
J1 – 61			Axis 1/5 Coarse Home / Home return / supply	61
J1 – 28	21	Input - opto isolated (bi-	Axis 2 Coarse Home (shared by Axis 6 Home)	28
J1 – 62			Axis 2/6 Coarse Home / Home return / supply	62
J1 – 29	18	Input - opto isolated (bi-	Axis 1Limit + (shared by Axis 5 Limit +)	29
J1 – 63			Axis 1/5 Limit + return / supply	63
J1 – 30	22	Input - opto isolated (bi-	Axis 2 Limit + (shared by Axis 6 Limit +)	30
J1 – 64			Axis 2/6 Limit + return / supply	64
J1 – 31	19	Input - opto isolated (bi-	Axis 1 Limit - (shared by Axis 5 Limit -)	31
J1 – 65			Axis 1/5 Limit - return / supply	65
J1 – 32	23	Input - opto isolated (bi-	Axis 2 Limit - (shared by Axis 6 Limit -)	32
J1 – 66			Axis 2/6 Limit - return / supply	66
J1 – 33			Analog Input #1 (option)	33
J1 – 67			Analog In #1 return / An. Ground	67
J1 – 34			Analog Input #2 (option)	34
J1 – 68			Analog In #2 return / An. Ground	68

Connector Pinout - *MFX-PCI1440 (continued)*

Connector J3 (Pulse Command Axes 5 & 6, Dig. inputs 9-12, Dig. outputs 9-12, A/D inputs 5 & 6)

VHDCI Pin #	Digital I/O Channel #	Circuit Type	Description (default configuration)	Adam-3968 Pin #
J3 - 1	50	Output - open collector driver	Axis 5 All Windings Off output	1
J3 – 35			+5 VDC	35
J3 – 2			Axis 5 Step / CCW Pulse	2
J3 – 36			+5 VDC	36
J3 – 3			Axis 5 Direction / CW Pulse	3
J3 – 37			+5 VDC	37
J3 – 4	51	Output - open collector driver	Axis 5 Full Current / Axis 1 Unipolar Direction output	4
J3 – 38			+5 VDC	38
J3 – 5	55	Output - open collector driver	Axis 6 Full Current / Axis 2 Unipolar Direction output	5
J3 – 39			+5 VDC	39
J3 – 6	54	Output - open collector driver	Axis 6 All Windings Off output	6
J3 – 40			+5 VDC	40
J3 – 7			Axis 6 Step / CCW Pulse	7
J3 – 41			+5 VDC	41
J3 – 8			Axis 6 Direction / CW Pulse	8
J3 – 42			+5 VDC	42
J3 – 9			+12 VDC	9
J3 – 43			+12 VDC	43
J3 – 10			Axis 5/6 Encoder Ref. (1.5V)	10
J3 – 44			Axis 5/6 Encoder Ref. (1.5V)	44
J3 – 11			Axis 5 Encoder Phase A+	11
J3 – 45			Axis 5 Encoder Phase A-	45
J3 – 12			Axis 5 Encoder Phase B+	12
J3 – 46			Axis 5 Encoder Phase B-	46
J3 – 13			Axis 5 Encoder Phase Z+	13
J3 – 47			Axis 5 Encoder Phase Z -	47
J3 – 14			Axis 6 Encoder Phase A+	14
J3 – 48			Axis 6 Encoder Phase A-	48
J3 – 15			Axis 6 Encoder Phase B+	15
J3 – 49			Axis 6 Encoder Phase B-	49
J3 – 16			Axis 6 Encoder Phase Z+	16
J3 – 50			Axis 6 Encoder Phase Z -	50
J3 – 17	20	Input - opto isolated (bi-directional)	Axis 5 Drive Fault input (shared by Axis 1 Amp	17
J3 – 51			Axis 1/5 Amp Fault supply / return	51
J3 – 18	24	Input - opto isolated (bi-directional)	Axis 6 Drive Fault input (shared by Axis 2 Amp	18
J3 – 52			Axis 2/6 Amp Fault supply / return	52
J3 – 19	41	Output - TTL	Digital Out #9 / Axis 5 - 8 Position Compare	19
J3 – 53			+5 VDC	53
J3 – 20	42	Output - TTL	Digital Output #10	20
J3 – 54			+5 VDC	54
J3 – 21	43	Output - TTL	Digital Output #11	21
J3 – 55			+5 VDC	55
J3 – 22	44	Output - TTL	Digital Output #12	22
J3 – 56			+5 VDC	56
J3 – 23	9	Input - TTL	Dig. In. #9 / Axis 5 & 6 Position Capture (Latch)	23
J3 – 57			Ground	57
J3 – 24	10	Input - TTL	Digital Input #10	24
J3 – 58			Ground	58
J3 – 25	11	Input - TTL	Digital Input #11	25
J3 – 59			Ground	59
J3 – 26	12	Input - TTL	Digital Input #12	26
J3 – 60			Ground	60
J3 – 27	17	Input - opto isolated (bi-directional)	Axis 5 Home (shared by Axis 1 Coarse Home)	27
J3 – 61			Axis 1/5 Coarse Home / Home return / supply	61
J3 – 28	21	Input - opto isolated (bi-directional)	Axis 6 Home (shared by Axis 6 Coarse Home)	28
J3 – 62			Axis 2/6 Coarse Home / Home return / supply	62
J3 – 29	18	Input - opto isolated (bi-directional)	Axis 5 Limit + (shared by Axis 1 Limit +)	29
J3 – 63			Axis 1/5 Limit + return / supply	63
J3 – 30	22	Input - opto isolated (bi-directional)	Axis 6 Limit + (shared by Axis 2 Limit +)	30
J3 – 64			Axis 2/6 Limit + return / supply	64
J3 – 31	19	Input - opto isolated (bi-directional)	Axis 5 Limit - (shared by Axis 1 Limit -)	31
J3 – 65			Axis 1/5 Limit - return / supply	65
J3 – 32	23	Input - opto isolated (bi-directional)	Axis 6 Limit - (shared by Axis 2 Limit -)	32
J3 – 66			Axis 2/6 Limit - return / supply	66
J3 – 33			Analog Input #5 (option)	33
J3 – 67			Analog In #5 return / An. Ground	67
J3 – 34			Analog Input #6 (option)	34
J3 – 68			Analog In #6 return / An. Ground	68

Connector Pinout - *MFX-PCI1440 (continued)*

Connector J4 (Pulse Command Axes 7 & 8, Dig. inputs 13-16, Dig. outputs 13-16, A/D inputs 7 & 8)

VHDCI Pin #	*Digital I/O Channel #*	*Circuit Type*	*Description (default configuration)*	*Adam-3968 Pin #*
J4 - 1	58	Output - open collector driver	Axis 7 All Windings Off output	1
J4 – 35			+5 VDC	35
J4 – 2			Axis 7 Step / CCW Pulse	2
J4 – 36			+5 VDC	36
J4 – 3			Axis 7 Direction / CW Pulse	3
J4 – 37			+5 VDC	37
J4 – 4	59	Output - open collector driver	Axis 7 Full Current / Axis 3 Unipolar Direction	4
J4 – 38			+5 VDC	38
J4 – 5	63	Output - open collector driver	Axis 8 Full Current / Axis 4 Unipolar Direction	5
J4 – 39			+5 VDC	39
J4 – 6	62	Output - open collector driver	Axis 8 All Windings Off output	6
J4 – 40			+5 VDC	40
J4 – 7			Axis 8 Step / CCW Pulse	7
J4 – 41			+5 VDC	41
J4 – 8			Axis 8 Direction / CW Pulse	8
J4 – 42			+5 VDC	42
J4 – 9			+12 VDC	9
J4 – 43			+12 VDC	43
J4 – 10			Axis 7/8 Encoder Ref. (1.5V)	10
J4 – 44			Axis 7/8 Encoder Ref. (1.5V)	44
J4 – 11			Axis 7 Encoder Phase A+	11
J4 – 45			Axis 7 Encoder Phase A-	45
J4 – 12			Axis 7 Encoder Phase B+	12
J4 – 46			Axis 7 Encoder Phase B-	46
J4 – 13			Axis 7 Encoder Phase Z+	13
J4 – 47			Axis 7 Encoder Phase Z -	47
J4 – 14			Axis 8 Encoder Phase A+	14
J4 – 48			Axis 8 Encoder Phase A-	48
J4 – 15			Axis 8 Encoder Phase B+	15
J4 – 49			Axis 8 Encoder Phase B-	49
J4 – 16			Axis 8 Encoder Phase Z+	16
J4 – 50			Axis 8 Encoder Phase Z -	50
J4 – 17	28	Input - opto isolated (bi-	Axis 7 Drive Fault input (shared by Axis 3 Amp	17
J4 – 51			Axis 3/7 Amp Fault supply / return	51
J4 – 18	32	Input - opto isolated (bi-	Axis 8 Amp. Fault input (shared by Axis 8 Amp	18
J4 – 52			Axis 4/8 Amp Fault supply / return	52
J4 – 19	45	Output - TTL	Digital Output #13	19
J4 – 53			+5 VDC	53
J4 – 20	46	Output - TTL	Digital Output #14	20
J4 – 54			+5 VDC	54
J4 – 21	47	Output - TTL	Digital Output #15	21
J4 – 55			+5 VDC	55
J4 – 22	48	Output - TTL	Digital Output #16	22
J4 – 56			+5 VDC	56
J4 – 23	13	Input - TTL	Dig. In. #13 / Axis 7 & 8 Position Capture (Latch)	23
J4 – 57			Ground	57
J4 – 24	14	Input - TTL	Digital Input #14	24
J4 – 58			Ground	58
J4 – 25	15	Input - TTL	Digital Input #15	25
J4 – 59			Ground	59
J4 – 26	16	Input - TTL	Digital Input #16	26
J4 – 60			Ground	60
J4 – 27	25	Input - opto isolated (bi-	Axis 7 Home (shared by Axis 3 Coarse Home)	27
J4 – 61			Axis 3/7 Coarse Home / Home return / supply	61
J4 – 28	29	Input - opto isolated (bi-	Axis 8 Home (shared by Axis 4 Coarse Home)	28
J4 – 62			Axis 4/8 Coarse Home / Home return / supply	62
J4 – 29	26	Input - opto isolated (bi-	Axis 7 Limit + (shared by Axis 3 Limit +)	29
J4 – 63			Axis 3/7 Limit + return / supply	63
J4 – 30	30	Input - opto isolated (bi-	Axis 8 Limit + (shared by Axis 4 Limit +)	30
J4 – 64			Axis 4/8 Limit + return / supply	64
J4 – 31	27	Input - opto isolated (bi-	Axis 7 Limit - (shared by Axis 7 Limit -)	31
J4 – 65			Axis 3/7 Limit - return / supply	65
J4 – 32	31	Input - opto isolated (bi-	Axis 8 Limit - (shared by Axis 8 Limit -)	32
J4 – 66			Axis 4/8 Limit - return / supply	66
J4 – 33			Analog Input #7 (option)	33
J4 – 67			Analog In #7 return / An. Ground	67
J4 – 34			Analog Input #8 (option)	34
J4 – 68			Analog In #8 return / An. Ground	68

B.1 C++ PROGRAM

Program for entering skin entry point and target point and calculating distances required to be moved by the robotic arm.

M1, M2, M5, M6, M7, M8 are motor movements required for respective Axes 1, 2, 5, 6, 7 and 8 to reach to the final target.

```
#include "stdafx.h"

#include <iostream>

#include <stdio.h>

#include <math.h>

#include <sstream>

#include <stdlib.h>

#include <fstream>

using std::cin;

using std::cout;

using std::endl;

using namespace std;

int main()

{

const double PI = 4.0*atan(1.0);

double d=0;

int M1=0, M2=0, M5=0, M6=0, M7=0, M8=0 ;

double D, _D, theta4=0;

double x1=0, y1=0, z1=0, x2=0, y2=0, z2=0, x3=0, y3=0, z3=0, x4=0, y4=0, z4=0, a=0, b=0;

double theta6=0, theta5=0, D1=0, D2=0, D3=0, d1=0, d2=0, d3=0;
```

```
ofstream outputFile;
cout<<"Distance of the needle from the point of entry dd = 2 cms\n";
cout<<"\nLength of the needle l = 10.671 cms\n";
cout<<"\nHeight of the robotic arm d4 = 73.818 cms\n";
cout<<"\nLength of the robotic arm a4+d7 = 75.514 cms\n";
cout<<"\nRange of the robot in X direction (downward) = 5 cms\n";
cout<<"\nRange of the robot in Y direction (+ve&-ve Y, sideways) = 3.5 cms\n";
cout<<"\nRange of the robot in Z direction (forward) = 5 cms\n";

    const double dd = 2;
    const double l = 10.671;
    const double R2 = 3.4;
    const double d4 = 73.819;
    const double a4 = 62.942;
    const double a7 = 10.671;
    const double d7 = 12.572;

    cout<< "\nPlease enter the coordinates of point of entry : \n";
    cout<< "\nX coordinate, x2= ";
    cin >> x2;
    cout<< "\nY coordinate, y2= ";
    cin >> y2;
    cout<< "\nZ coordinate, z2= ";
    cin >> z2;
```

```
cout<< "\nPlease enter the coordinates of final point the needle tip is supposed to reach : \n";

cout<< "\nX coordinate, x1= ";

cin >> x1;

cout<< "\nY coordinate, y1= ";

cin >> y1;

cout<< "\nZ coordinate, z1= ";

cin >> z1;


d = sqrt((x2-x1)*(x2-x1) + (y2-y1)*(y2-y1) + (z2-z1)*(z2-z1));

cout<< "\nValue of d = " ;

cout << d << endl;


D= d + dd;

_D= d + l + dd;

cout<< "\nValue of D = " ;

cout << D << endl;

cout<< "\nValue of _D = " ;

cout << _D << endl<< "\n";


x3= x1 + D*(x2-x1)/d;

y3= y1 + D*(y2-y1)/d;

z3= z1 + D*(z2-z1)/d;


x4= x1 + _D*(x2-x1)/d;

y4= y1 + _D*(y2-y1)/d;
```

```
z4= z1 + _D*(z2-z1)/d;

cout<< "\nValue of x3 = " ;
cout << x3 << endl;
cout<< "\nValue of y3 = " ;
cout << y3 << endl;
cout<< "\nValue of z3 = " ;
cout << z3 << endl;
cout<< "\nValue of x4 = " ;
cout << x4 << endl;
cout<< "\nValue of y4 = " ;
cout << y4 << endl;
cout<< "\nValue of z4 = " ;
cout << z4 << endl<< "\n";

if(abs(y4)<=R2)
{
        d3 = x4-d4;
        d2 = -y4;
        d1 = z4-a4-d7;

        cout<<"the value of y4=" << y4<<endl;
        cout<<"\nthe value of R2="<<R2<<endl;

        cout<< "\nValue of d1 = " ;
        cout << d1 << endl;
```

```
cout<< "\nValue of d2 = " ;
cout << d2 << endl;
cout<< "\nValue of d3 = " ;
cout << d3 << endl;
theta4 = 0;
theta6 = (180/PI)*asin((-d2-y3)/a7);
theta5 = (180/PI)*asin((a4+d7+d1-z3)/(a7*cos(theta6*PI/180)));


D1= z3 + a7*sin((theta5)*PI/180)*cos((theta6)*PI/180) - d7*cos((theta5)*PI/180)
- a4;
D2= -a7*sin((theta6)*PI/180) - y3;
D3= x3 + a7*cos((theta5)*PI/180)*cos((theta6)*PI/180) +
d7*sin((theta5)*PI/180) - d4;


cout.precision(4);
cout<< "\n\nValue of D1 = " ;
cout << D1 << endl<< "\n";


cout.precision(2);
cout<< "\nValue of D2 = " ;
cout << D2 << endl<< "\n";


cout.precision(4);
cout<< "\nValue of D3 = " ;
cout << D3 << endl<< "\n";


cout.precision(4);
```

```
cout<< "\nValue of theta4 = " ;
cout << theta4 << endl<< "\n";

cout.precision(4);
cout<< "\nValue of theta5 = " ;
cout << theta5 << endl<< "\n";

cout.precision(4);
cout<< "\nValue of theta6 = " ;
cout << theta6 << endl<< "\n";

M1 =  80*theta4;
cout<<"Motor 1 has to move by = "<<M1<<" units\n";
M2 =  -2218*D1;
cout<<"Motor 2 has to move by = "<<M2<<" units\n";
M5 =  131233*D2;
cout<<"Motor 5 has to move by = "<<M5<<" units\n";
M6 =  -4059*D3;
cout<<"Motor 6 has to move by = "<<M6<<" units\n";
M7 =  -1.11*theta6;
cout<<"Motor 7 has to move by = "<<M7<<" units\n";
M8 =  69.44*theta5;
cout<<"Motor 8 has to move by = "<<M8<<" units\n";
outputFile.open("C:\\OuputValues\\OutputData.txt");
outputFile << M1<< endl;
outputFile << M2 << endl;
```

```
		outputFile << M5<< endl;
		outputFile << M6<< endl;
		outputFile << M7<< endl;
		outputFile << M8<< endl;
		outputFile.close();
	}
	else
	{
		d3=x4-d4;
		cout<<"The value of d3 = ";
		cout<<d3<<endl;

		if(y4>R2)
			d2 = -3.4;
		else
			d2 = 3.4;
		cout<<"the value of d2 = ";
		cout<<d2<<endl;
		//theta4 = 0;
		theta4 = (180/PI)*asin((-y4-d2)/(d7+a4));
		//cout<<"the value of theta4 = ";
		//cout<<theta4<<endl;
		d1 = z4-(a4+d7)*(cos((theta4)*PI/180));
		cout<<"the value of d1";
		cout<<d1<<endl;
		a = -(y3+d2+(a4+d7)*(sin((theta4)*PI/180)))*(cos((theta4)*PI/180));
```

```
b = -(z3-d1-(a4+d7)*(cos((theta4)*PI/180)))*(sin((theta4)*PI/180));

cout<<"the value of a = ";

cout<<a<<endl;

cout<<"the value of b = ";

cout<<b<<endl;


theta6 = (180/PI)*(asin((a + b)/a7));

theta5 = (180/PI)*asin((((a4+d7)*cos((theta4)*PI/180))-z3 + d1-
a7*(sin((theta4)*PI/180))*(sin((theta6)*PI/180)))/(a7*(cos((theta4)*PI/180))*(cos((theta
6)*PI/180))));

D1= z3 + a7*(sin((theta5)*PI/180))*(cos((theta6)*PI/180))*(cos((theta4)*PI/180)) -
d7*(cos((theta5)*PI/180))*(cos((theta4)*PI/180)) - a4*(cos((theta4)*PI/180));

D2= a7*(sin((theta5)*PI/180))*(cos((theta6)*PI/180))*(sin((theta4)*PI/180)) -
a7*(sin((theta6)*PI/180))*(cos((theta4)*PI/180)) -
d7*(sin((theta4)*PI/180))*(cos((theta5)*PI/180)) - a4*sin((theta4)*PI/180) - y3;

D3= x3 + a7*(cos((theta5)*PI/180))*cos((theta6)*PI/180) + d7*sin((theta5)*PI/180) -
d4;


cout.precision(4);

cout<< "\nValue of D1 = " ;

cout << D1 << endl<< "\n";

cout.precision(4);

cout<< "\nValue of D2 = " ;

cout << D2 << endl<< "\n";

cout.precision(4);

cout<< "\nValue of D3 = " ;

//cout << setprecision(4) << showpoint << D3 << endl;

cout << D3 << endl<< "\n";
```

```
cout.precision(4);
cout<< "\nValue of theta4 = " ;
cout << theta4 << endl<< "\n";
cout.precision(4);
cout<< "\nValue of theta5 = " ;
cout << theta5 << endl<< "\n";
cout.precision(4);
cout<< "\nValue of theta6 = " ;
cout << theta6 << endl<< "\n";

//cout.precision(4);
M1 =  80*theta4;
cout<<"Motor 1 has to move by = "<<M1<<" units\n";
//cout.precision(4);
M2 =  -2218*D1;
cout<<"Motor 2 has to move by = "<<M2<<" units\n";
//cout.precision(4);
M5 =  131233*D2;
cout<<"Motor 5 has to move by = "<<M5<<" units\n";
//cout.precision(4);
M6 =  -4059*D3;
cout<<"Motor 6 has to move by = "<<M6<<" units\n";
//cout.precision(4);
M7 =  -1.11*theta6;
cout<<"Motor 7 has to move by = "<<M7<<" units\n";
//cout.precision(4);
```

```
            M8 =  69.44*theta5;
            cout<<"Motor 8 has to move by = "<<M8<<" units\n";
            outputFile.open("C:\\OuputValues\\OutputData.txt");
            outputFile << M1<< endl;
            outputFile << M2 << endl;
            outputFile << M5<< endl;
            outputFile << M6<< endl;
            outputFile << M7<< endl;
            outputFile << M8<< endl;
            outputFile.close();
      }
            system ("PAUSE");
      return 0;
}
```

B.2 C++ PROGRAM

Motion Control API Program to physically move the motors to the required amounts as obtained from the previous C++ program.

```
// DESCRIPTION
// This program implements a windows based user interface for a motion controller. The front panel displays position and trajectory generator settings, status info, and accepts numerical move data. Setup dialogs are provided for steppers, servos, scaling, and controller configuration.

#include <windows.h>
#include <windowsx.h>
```

```
#include <stdio.h>

#include <sstream>

#include <math.h>

#include "..\mcapi.h"

#include "..\mcdlg.h"

#include "cwdemo.h"

#include<fstream>

using namespace std;


//  Constants

const char gszAppName[] = "CWDemo";          // the name of our app

const char gszIniFile[] = "CWDemo.ini";      // our ini file


//  DESCRIPTION

//    Registers the window class (if no previous instances), creates a main window and all
the display controls. Then opens the Motion API and sets up a timer for the status
displays.


int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR
lpszCmdLine, int nCmdShow)

{

        DialogBox(hInstance, "MainDialog", NULL,
        reinterpret_cast<DLGPROC>(MainDlgProc));

        return 0;

}

//  DESCRIPTION

//    Main window is implemented as a dialog box to simplify the code.
```

```
BOOL CALLBACK MainDlgProc(HWND hDlg, UINT uMsg, WPARAM wParam,
LPARAM lParam)

{

const DWORD      stat[10] = { MC_STAT_MTR_ENABLE, MC_STAT_TRAJ,
MC_STAT_DIR,       MC_STAT_INP_HOME,  MC_STAT_INP_INDEX,
MC_STAT_ERROR, MC_STAT_PLIM_TRIP, MC_STAT_MLIM_TRIP,
MC_STAT_AMP_FAULT };

        static short int nID,                                   // current controller
ID

        Axis                    = 1,                    // current axis number

        OldAxis                 = -1;                   // last axis number

        static int  DisplayPrecision    = 0,                     // for formatting
readouts

        ModeButton              = -1,           // our current mode (Abs, Rel,
Cycle)

        Cycle                   = 0,                    // cycle mode flag

        SkipTimer               = false,        // skip timer flag (set when in
dialogs)

        OLStepper               = false;    // axis is open loop stepper (no
following error)

        static UINT      ErrorMessage;                  // pre-agreed upon error
message

        static HCTRLR    hCtlr;                         // controller handle

        int d1, d2,d3,d4,d5,d6;

        /*ifstream inputfile; inputfile.open( "C:\\OuputValues\\OutputData.txt");*/

        fstream file("C:\\OuputValues\\OutputData.txt");

                        int linecount = 0;

                        int valueArray[100];

                        int *finalArray;

                        while (!file.eof())
```

```
{
file >> valueArray[linecount];
linecount++;
}
finalArray = new int[linecount];
for (int i = 0; i < linecount; i++)
{
finalArray[i] = valueArray[i];
}
d1=valueArray[0];
d2=valueArray[1];
d3=valueArray[2];
d4=valueArray[3];
d5=valueArray[4];
d6=valueArray[5];

delete[] finalArray;

switch (uMsg)
{
    case WM_INITDIALOG:

        //
        //  Restore options, axis number, controller ID, distance, and mode
        //
        {
```

```
char buffer[32];

GetPrivateProfileString("Settings", "Distance", "0", buffer, sizeof(buffer),
gszIniFile);
Edit_SetText(GetDlgItem(hDlg, IDC_DISTANCE), buffer);

if (GetPrivateProfileInt("Settings", "AutoInit", 0, gszIniFile))
CheckMenuItem(GetMenu(hDlg), IDM_AUTO_INIT, MF_CHECKED);
else
CheckMenuItem(GetMenu(hDlg), IDM_AUTO_INIT,
MF_UNCHECKED);
nID = GetPrivateProfileInt("Settings", "ControllerID", 0, gszIniFile);
Axis = GetPrivateProfileInt("Settings", "Axis", 1, gszIniFile);
ModeButton = GetPrivateProfileInt("Settings", "Mode", IDC_BTN_REL,
gszIniFile);
CheckRadioButton(hDlg, IDC_BTN_ABS, IDC_BTN_CYCLE,
ModeButton);
}
//
//  Set default on color for LEDS - makes timer processing easy.
SendDlgItemMessage(hDlg, IDC_LED2, LEDM_SETCHECKCOLOR,
WPARAM(false), LPARAM(PALETTERGB(224, 224, 0)));
SendDlgItemMessage(hDlg, IDC_LED3, LEDM_SETCHECKCOLOR,
WPARAM(false), LPARAM(PALETTERGB(224, 224, 0)));
SendDlgItemMessage(hDlg, IDC_LED4, LEDM_SETCHECKCOLOR,
WPARAM(false), LPARAM(PALETTERGB(224, 224, 0)));
SendDlgItemMessage(hDlg, IDC_LED5, LEDM_SETCHECKCOLOR,
WPARAM(false), LPARAM(PALETTERGB(224,   0, 0)));
```

```
SendDlgItemMessage(hDlg, IDC_LED6, LEDM_SETCHECKCOLOR,
WPARAM(false), LPARAM(PALETTERGB(224,   0, 0)));

SendDlgItemMessage(hDlg, IDC_LED7, LEDM_SETCHECKCOLOR,
WPARAM(false), LPARAM(PALETTERGB(224,   0, 0)));

SendDlgItemMessage(hDlg, IDC_LED8, LEDM_SETCHECKCOLOR,
WPARAM(false), LPARAM(PALETTERGB(224,   0, 0)));

SendDlgItemMessage(hDlg, IDC_LED9, LEDM_SETCHECKCOLOR,
WPARAM(false), LPARAM(PALETTERGB(224, 224, 0)));


//  Open the default (last used) motion controller. If there is a problem offer the user a
chance to select a different controller.

while (true)

        {

        if ((hCtlr = OpenController(hDlg, nID, &Axis)) > 0)

                break;

        else

        {

        if ((nID = static_cast<short int>(MCDLG_SelectController(hDlg, -1, 0, NULL)))
>= 0)

                continue;

                EndDialog(hDlg, false);

                return true;

        }

        }

//  Register our private error message

        ErrorMessage = RegisterWindowMessage("MCErrorNotify");

//  Set up the timer for readout updates

                while (!SetTimer(hDlg, ID_UPDATE_TIMER, 110, NULL))
```

```
			if (IDCANCEL == MessageBox(hDlg, "Too many clocks or timers!",
gszAppName,

	MB_ICONEXCLAMATION | MB_RETRYCANCEL))
					{
						MCClose(hCtlr);
						EndDialog(hDlg, false);
						break;
					}
				SetFocus(GetDlgItem(hDlg, IDC_BTN_ON));
				return false;			// false so our SetFocus() will work


			case WM_TIMER:				// display current motor data on
timer msg
				{				// process timer unless suspended
			MCSTATUSEX	NewStatus;
			static MCSTATUSEX OldStatus;
			double		NewValue;
			char		buffer[32];


			NewStatus.cbSize = sizeof(NewStatus);
			OldStatus.cbSize = sizeof(OldStatus);
			if (SkipTimer)
				break;
// If axis number has changed get new scale factor, check operating mode, etc.
					if (Axis != OldAxis)
					{
```

```
                              DisplayPrecision = OpenAxis(hDlg, hCtlr, Axis,
&OLStepper);

                              OldAxis = Axis;

                              }

// Get status and update LEDs if status has changed.

                              MCGetStatusEx(hCtlr, Axis, &NewStatus);

                              if (memcmp(&NewStatus, &OldStatus,
sizeof(NewStatus)))           // has status changed?

                              {

                                      OldStatus = NewStatus;


        int MotorOn  = MCDecodeStatusEx(hCtlr, &NewStatus,
MC_STAT_MTR_ENABLE);

        int MotorErr = MCDecodeStatusEx(hCtlr, &NewStatus, MC_STAT_ERROR);

        EnableWindow(GetDlgItem(hDlg, IDC_BTN_OFF), MotorOn);

        EnableWindow(GetDlgItem(hDlg, IDC_BTN_MOVE_P), MotorOn &&
!MotorErr);

        EnableWindow(GetDlgItem(hDlg, IDC_BTN_MOVE_M), MotorOn &&
!MotorErr);

        EnableWindow(GetDlgItem(hDlg, IDC_BTN_STOP), MotorOn && !MotorErr);

        EnableWindow(GetDlgItem(hDlg, IDC_BTN_ABORT), MotorOn &&
!MotorErr);

        EnableWindow(GetDlgItem(hDlg, IDC_BTN_HOME), MotorOn &&
!MotorErr);

        EnableWindow(GetDlgItem(hDlg, IDC_Move_Robot), MotorOn && !MotorErr);

        for (int i = 0; i < 9; i++)

        CheckDlgButton(hDlg, IDC_LED0 + i, MCDecodeStatusEx(hCtlr, &NewStatus,
        stat[i]) ? BST_CHECKED : BST_UNCHECKED);

        WORD wPhase;
```

```
MCGetServoOutputPhase(hCtlr, Axis, &wPhase);

CheckDlgButton(hDlg, IDC_LED9, wPhase == MC_PHASE_REV ?
BST_CHECKED : BST_UNCHECKED);


// If we are in cycle mode check for end of travel generate next
move if ready

if (MotorOn && !MotorErr)

{

if (Cycle && MCIsStopped(hCtlr, Axis, 0.0))

{

if (Cycle == 1)

SendMessage(hDlg, WM_COMMAND,
WPARAM(IDC_BTN_MOVE_M), LPARAM(GetDlgItem(hDlg,
IDC_BTN_MOVE_M)));

else

//SendMessage(hDlg, WM_COMMAND,
WPARAM(IDC_Move_Robot), LPARAM(GetDlgItem(hDlg, IDC_Move_Robot)));

SendMessage(hDlg, WM_COMMAND,
WPARAM(IDC_BTN_MOVE_P), LPARAM(GetDlgItem(hDlg,
IDC_BTN_MOVE_P)));

}

}

else

Cycle = 0;        // cancel cycle mode if motor has been
disabled or error

}

// Update position readout. MCGetPositionEx return value is checked for errors.

if (MCGetPositionEx(hCtlr, Axis, &NewValue) ==
MCERR_NOERROR)
```

```
{
_snprintf(buffer, sizeof(buffer), "%.*f", DisplayPrecision,
NewValue);
SetDlgItemText(hDlg, IDC_POSITION, buffer);
}
else
break;                                  // error - abort timer
processing
// Update optimal position readout. MCGetOptimalEx return value is checked for
errors.
if (MCGetOptimalEx(hCtlr, Axis, &NewValue) ==
MCERR_NOERROR)
{
_snprintf(buffer, sizeof(buffer), "%.*f", DisplayPrecision,
NewValue);
SetDlgItemText(hDlg, IDC_OPTIMAL, buffer);
}
else
break;                                  // error - abort timer
processing
//  Update target position readout. MCGetTargetEx return value is checked for
errors.
if (MCGetTargetEx(hCtlr, Axis, &NewValue) == MCERR_NOERROR)
{
_snprintf(buffer, sizeof(buffer), "%.*f", DisplayPrecision, NewValue);
SetDlgItemText(hDlg, IDC_TARGET, buffer);
}
else
break;                                  // error - abort timer processing
```

```
//  Update following error readout. MCGetFollowingError return value is checked for errors.

                if (!OLStepper)

                {

                if (MCGetFollowingError(hCtlr, Axis, &NewValue) ==
MCERR_NOERROR)

                {

                _snprintf(buffer, sizeof(buffer), "%.*f", DisplayPrecision, NewValue);

                SetDlgItemText(hDlg, IDC_FOLLOW, buffer);

                }

                }

                else

                SetDlgItemText(hDlg, IDC_FOLLOW, "");

                }

                break;

                case WM_COMMAND:

if (GET_WM_COMMAND_HWND(wParam, lParam) == NULL) // check for main menu selection

                {

                switch (GET_WM_COMMAND_ID(wParam, lParam))

                {

                case IDM_CONFIGURE_AXIS:

                SkipTimer = true;

        MCDLG_ConfigureAxis(hDlg, hCtlr, Axis, MCDLG_PROMPT |
        MCDLG_CHECKACTIVE, NULL);

                Cycle = SkipTimer = false;

                break;
```

```
case IDM_SCALING:
SkipTimer = true;
MCDLG_Scaling(hDlg, hCtlr, Axis, MCDLG_PROMPT |
MCDLG_CHECKACTIVE, NULL);
Cycle = SkipTimer = false;
break;
case IDM_AUTO_INIT:
SkipTimer = true;
if (GetMenuState(GetMenu(hDlg), IDM_AUTO_INIT, MF_BYCOMMAND) &
MF_CHECKED)
CheckMenuItem(GetMenu(hDlg), IDM_AUTO_INIT,
MF_UNCHECKED);
else
{
CheckMenuItem(GetMenu(hDlg), IDM_AUTO_INIT, MF_CHECKED);
MCDLG_RestoreAxis(hCtlr, MC_ALL_AXES, MCDLG_PROMPT |
MCDLG_CHECKACTIVE, NULL);
}
Cycle = SkipTimer = false;
break;
case IDM_INIT:
SkipTimer = true;
MCDLG_RestoreAxis(hCtlr, MC_ALL_AXES, MCDLG_PROMPT |
MCDLG_CHECKACTIVE, NULL);
Cycle = SkipTimer = false;
break;
case IDM_SAVE_SETTINGS:
SkipTimer = true;
```

```
MCDLG_SaveAxis(hCtlr, MC_ALL_AXES, 0, NULL);
SkipTimer = false;
break;
case IDM_CONTROLLER_INFO:
SkipTimer = true;
MCDLG_ControllerInfo(hDlg, hCtlr, 0, NULL);
SkipTimer = false;
break;
case IDM_CONTROLLER_SELECT:
{
Cycle = 0;                  // cancel cycling mode
SkipTimer = true;
while (true)
{
short int id;
if ((id = MCDLG_SelectController(hDlg, nID, 0, NULL)) >= 0)
{
HCTRLR new_hCtlr;
if ((new_hCtlr = OpenController(hDlg, id, &Axis)) > 0)
{
MCClose(hCtlr);
nID = id;
hCtlr = new_hCtlr;
break;                      // done
}
}
```

```
else
break;                          // cancel
}
OldAxis = -1;                           // force recalculation of display
precision
SkipTimer = false;                      // turn timer back on
}
break;
case IDM_CONTROLLER_RESET:
{
HCURSOR hCursor = SetCursor(LoadCursor(NULL, IDC_WAIT));
Cycle = 0;                                      // cancel cycling mode
MCReset(hCtlr, MC_ALL_AXES);
        if (GetMenuState(GetMenu(hDlg), IDM_AUTO_INIT, MF_BYCOMMAND) &
MF_CHECKED)
        MCDLG_RestoreAxis(hCtlr, MC_ALL_AXES, MCDLG_PROMPT |
MCDLG_CHECKACTIVE, NULL);
SetCursor(hCursor);
}
break;
case IDM_ABOUT:
SkipTimer = true;
MCDLG_AboutBox(hDlg, "About CWDemo", 0);
SkipTimer = false;
break;
case IDM_EXIT:
PostMessage(hDlg, WM_CLOSE, WPARAM(0), LPARAM(0));
```

```
break;
}
}
else                    //  If not a menu selection, it may be a window control
{
switch(GET_WM_COMMAND_ID(wParam, lParam))
{
case IDC_COMBO_AXIS:
if (GET_WM_COMMAND_CMD(wParam, lParam) ==
LBN_SELCHANGE)
Axis = static_cast<short>(SendDlgItemMessage(hDlg, IDC_COMBO_AXIS,
CB_GETCURSEL, 0, 0l)) + 1;
Cycle = 0;                                  // cancel cycling
mode
break;
case IDC_BTN_ABS:
case IDC_BTN_REL:
case IDC_BTN_CYCLE:
ModeButton = static_cast<int>(wParam);
CheckRadioButton(hDlg, IDC_BTN_ABS, IDC_BTN_CYCLE,
ModeButton);
if (ModeButton == IDC_BTN_ABS)
SetDlgItemText(hDlg, IDC_LBL_TRAVEL, "Target");
else
SetDlgItemText(hDlg, IDC_LBL_TRAVEL, "Distance");
Cycle = 0;                                  // cancel cycling
mode
break;
```

```
case IDC_BTN_ON:
for (int Axis_Number =0; Axis_Number<7; Axis_Number++)
{
MCEnableAxis(hCtlr, Axis_Number, true);
}
Cycle = 0;                                  // cancel cycling mode
OldAxis = -1;                          // force recalculation of display
precision
break;
case IDC_BTN_OFF:
for (int Axis_Number =0; Axis_Number<7; Axis_Number++)
{
MCEnableAxis(hCtlr, Axis_Number, false);
}
Cycle = 0;                                  // cancel cycling
mode
break;
case IDC_BTN_MOVE_P:
case IDC_BTN_MOVE_M:
{
char   szBuffer[32];
double move;

GetWindowText(GetDlgItem(hDlg, IDC_DISTANCE), szBuffer,
sizeof(szBuffer));
move = strtod(szBuffer, NULL) * (wParam == IDC_BTN_MOVE_P ? 1 :
-1);
```

```
if (ModeButton == IDC_BTN_ABS)
MCMoveAbsolute(hCtlr, Axis, move);
else
{
MCMoveRelative(hCtlr, Axis, move);
if (ModeButton == IDC_BTN_CYCLE)
Cycle = wParam == IDC_BTN_MOVE_P ? 1 : -1;
}
}
break;
case IDC_Move_Robot:
char
buffer1[32],buffer2[32],buffer3[32],buffer4[32],buffer5[32],buffer6[32] ;
//GetWindowText(GetDlgItem(hDlg, IDC_D1), buffer1, sizeof(buffer1));
MCMoveAbsolute(hCtlr, 1, d1);
MCMoveAbsolute(hCtlr, 7, d5);
MCIsStopped(hCtlr, 7, 10.0);
MCMoveAbsolute(hCtlr, 8, d6);
MCIsStopped(hCtlr, 1, 10.0);
MCIsStopped(hCtlr, 8, 10.0);
MCMoveAbsolute(hCtlr, 2, d2);
MCMoveAbsolute(hCtlr, 5, d3);
MCMoveAbsolute(hCtlr, 6, d4);
//MCMoveAbsolute(hCtlr, Axis, move);
break;
case IDC_BTN_HOME:
```

```
//for (int Axis_Number =0; Axis_Number<7; Axis_Number++)
    //{
        MCGoHome(hCtlr, 6);
        Cycle = 0;
        MCIsStopped(hCtlr, 6, 15.0);
        MCGoHome(hCtlr, 1);
        Cycle = 0;
        MCGoHome(hCtlr, 2);
        Cycle = 0;
        MCGoHome(hCtlr, 5);
        Cycle = 0;
        //MCGoHome(hCtlr, 6);
        //Cycle = 0;
        MCGoHome(hCtlr, 7);
        Cycle = 0;
        MCGoHome(hCtlr, 8);
        Cycle = 0;
        //}
        // cancel cycling mode
        break;
    case IDC_BTN_STOP:
  // for MC1xx modules you should call MCEnableAxis()
for (int Axis_Number =0; Axis_Number<7; Axis_Number++)
        {
MCStop(hCtlr, Axis_Number);            //   after a call to
MCStop()
        }
```

```
                                Cycle = 0;
// cancel cycling mode
                                break;


case IDC_BTN_ABORT:                        // for MC1xx modules you should
call MCEnableAxis()
MCAbort(hCtlr, Axis);                      //   after a call to MCAbort()
Cycle = 0;                                     // cancel cycling mode
break;
case IDC_BTN_ZERO:
MCSetPosition(hCtlr, Axis, 0.0);
Cycle = 0;                                     // cancel cycling mode
break;
}
}
break;
case WM_CLOSE:
        KillTimer(hDlg, ID_UPDATE_TIMER);
        SaveState(hDlg, nID, Axis, ModeButton);
        MCClose(hCtlr);
        EndDialog(hDlg, true);
        break;


    default:
        if (uMsg == ErrorMessage)
        {
            SkipTimer = true;
```

```
                char buffer[128];
            MCTranslateErrorEx(static_cast<short int>(wParam), buffer,
sizeof(buffer));
            std::ostringstream message;
            message << "MCAPI Function '" <<
reinterpret_cast<LPSTR>(lParam) << "()' reported the following error:" << std::endl;
            message << ">> " << buffer << " <<" << std::endl << std::endl;
            message << "Do you want to quit CWDemo?";
            if (MessageBox(NULL, message.str().c_str(), gszAppName,
MB_ICONEXCLAMATION | MB_YESNO) == IDYES)
            PostMessage(hDlg, WM_CLOSE, WPARAM(0), LPARAM(0));
                else
                SkipTimer = false;
            }
                else
                return false;           // not our msg - pass to default dialog
proc
            }
            return true;
            }


//  OpenAxis - ready an axis for use
// DESCRIPTION
//  Readies an axis - makes certain axis is in position mode, changes front panel LED
labels based upon axis type, sets the number of decimal places in the display based upon
scale factors.
int OpenAxis(HWND hWnd, HCTRLR hCtlr, short int Axis, int* OLStepper)
```

```
{
	int mode;

	if (hCtlr > 0)
	{
		MCAXISCONFIG AxisConfig;
		AxisConfig.cbSize = sizeof(AxisConfig);
		MCGetAxisConfiguration(hCtlr, Axis, &AxisConfig);
		if (AxisConfig.MotorType & MC_TYPE_SERVO)
		{
			SetDlgItemText(hWnd, IDC_LED4, "Index");
			*OLStepper = false;
		}
		else
		{
			SetDlgItemText(hWnd, IDC_LED4, "Home");
			MCGetModuleInputMode(hCtlr, Axis, &mode);
			*OLStepper = mode == MC_IM_OPENLOOP;
		}
		MCGetOperatingMode(hCtlr, Axis, &mode);
		if (mode != MC_MODE_POSITION)
		MCSetOperatingMode(hCtlr, Axis, 0, MC_MODE_POSITION);
		MCSCALE Scaling;
		MCGetScale(hCtlr, Axis, &Scaling);
		return static_cast<int>(log10(fabs(Scaling.Scale)) + 0.99);
	}
```

```
	return 0;
}
//    OpenController - opens and initialize the selected controller
//  DESCRIPTION
```

Opens the controller specified by ID, checks to make certain it has at least one motion axis, restores the axis settings (if Auto Init is enabled), and configures the front panel controls.

```
HCTRLR OpenController(HWND hWnd, int nID, short int* Axis)
{
	HCTRLR hCtlr;
	if ((hCtlr = MCOpen(static_cast<short int>(nID), MC_OPEN_BINARY, "")) <= 0)
	{
		char buffer[128];
		MCTranslateErrorEx(hCtlr, buffer, sizeof(buffer));
		std::ostringstream message;
		message << "Unable to open controller for the following reason:\n\nID "
	<< nID << "  -  " << buffer << "\n";
	MessageBox(hWnd, message.str().c_str(), gszAppName, MB_ICONEXCLAMATION | MB_OK);
	return hCtlr;
	}
	//  Get the controller configuration data
	MCPARAM Param;
	MCGetConfiguration(hCtlr, &Param);
	if (Param.NumberAxes == 0)
	{
```

```
        MessageBox(hWnd, "No motor modules are installed on this controller. CWDemo\n"
                "requires that at least one servo or stepper module be installed.",
                gszAppName, MB_ICONEXCLAMATION | MB_OK);
        MCClose(hCtlr);
        return 0;
    }
    //  Initialize the axes
    if (GetMenuState(GetMenu(hWnd), IDM_AUTO_INIT, MF_BYCOMMAND) & MF_CHECKED)
    MCDLG_RestoreAxis(hCtlr, MC_ALL_AXES, MCDLG_PROMPT | MCDLG_CHECKACTIVE, NULL);
    if (*Axis > Param.NumberAxes)
        *Axis = 1;
    //  Initialize dialog controls
    SendDlgItemMessage(hWnd, IDC_COMBO_AXIS, CB_RESETCONTENT, WPARAM(0), LPARAM(0));
    for (int i = 1; i <= Param.NumberAxes; i++)
    {
        std::ostringstream text;
        text << "Axis " << i;
SendDlgItemMessage(hWnd, IDC_COMBO_AXIS, CB_ADDSTRING, WPARAM(0), LPARAM(text.str().c_str()));
    }
SendDlgItemMessage(hWnd, IDC_COMBO_AXIS, CB_SETCURSEL, WPARAM(*Axis - 1), LPARAM(0));
//  Hook the error message callback function. This way any error messages are neatly delivered to the window message loop.
```

```
    MCErrorNotify(hWnd, hCtlr, MCERRMASK_STANDARD);
    return hCtlr;
}
//  SaveState - updates INI file
// DESCRIPTION
Gets and saves main window position, axis number, controller ID and Auto Init state.
void SaveState(HWND hWnd, short int nID, WORD Axis, short int Mode)
{
    char buffer[32];
    // Auto-Init state
    if (GetMenuState(GetMenu(hWnd), IDM_AUTO_INIT, MF_BYCOMMAND) &
MF_CHECKED)
        WritePrivateProfileString("Settings", "AutoInit", "1", gszIniFile);
    else
        WritePrivateProfileString("Settings", "AutoInit", "0", gszIniFile);
    // Current controller, axis number, distance, and mode
    _snprintf(buffer, sizeof(buffer), "%d", nID);
    WritePrivateProfileString("Settings", "ControllerID", buffer, gszIniFile);
    _snprintf(buffer, sizeof(buffer), "%d", Axis);
    WritePrivateProfileString("Settings", "Axis", buffer, gszIniFile);
    GetWindowText(GetDlgItem(hWnd, IDC_DISTANCE), buffer, sizeof(buffer));
    WritePrivateProfileString("Settings", "Distance", buffer, gszIniFile);
    _snprintf(buffer, sizeof(buffer), "%d", Mode);
    WritePrivateProfileString("Settings", "Mode", buffer, gszIniFile);
    GetWindowText(GetDlgItem(hWnd, IDC_D1), buffer, sizeof(buffer));
}
```