Robot Kinematics:
Applications in Virtual Reality Based Pedagogy and Sensor Calibration

Megan Flanders

November 22, 2016

A thesis submitted for the degree of

Doctor of Philosophy

to the
School of Engineering
National University of Ireland, Cork

Supervisor: Dr. Richard C. Kavanagh

Head of School: Prof. William P. Marnane

Take to kinematics. It will repay you.
It is more fecund than geometry; it adds a fourth dimension to space.

– Chebyshev, 1873

Abstract

Conventions exist to describe the kinematics of a robot concisely, providing information about both its form and pose (position *and* orientation). Although mathematically convenient, the physical correlation between the parameters of these conventions and the robot that they represent is not necessarily intuitively obvious. Those who are new to the field of robotics may find it especially difficult to visualize these relationships.

After presenting relevant background information on kinematics, robotics, virtual reality, and inertial sensors, this thesis investigates the effectiveness of using desktop virtual reality tools to help university-level students with the visualization of fundamental concepts in robot kinematics. Specifically, it examines how the new “Rotation Tool” assists students in the visualization of fixed and mobile frame compound rotations while verifying their non-commutative nature. It also explains how the new “Build-A-Robot” aids students in identifying the role that each of the Denavit-Hartenberg parameters plays in the description of the position and orientation of a serial manipulator’s component links. To enable flexible, real-time user interaction, Build-A-Robot employed a novel approach wherein MATLAB was used to directly manipulate the fundamental geometry of Virtual Reality Modeling Language (VRML) objects. Survey feedback and examination results are presented which indicate the students’ increased understanding that resulted after using both of these tools. This improvement was especially apparent among students who struggled to understand the concepts when traditional teaching methods alone were used.

Tolerances in the manufacturing and assembly of robot arms introduce errors to the nominal kinematic models specified by manufacturers. This thesis also considers the impact of non-ideal kinematic parameters on the motion of the end-effector of a SCARA robot, which was used to calibrate an attached dual-axis accelerometer. Two novel, in-place calibration routines that employ dynamic accelerations are presented and validated using experimental data.

Declaration

I, Megan Flanders, hereby certify that this thesis is my own work, and except where acknowledged in the text, the contents of this submission are original. This work has not been submitted, in whole or in part, for consideration for any other degree in this university or elsewhere.

Megan Flanders

Acknowledgements

It feels a bit surreal to finally be at the stage where I get to write my acknowledgements section. It is the section that I have been thinking about for years – every time someone helped in some small way to make my path smoother. But now that it is time to assemble those somewhat abstract thoughts into concrete sentences, I am sure that I will accidentally leave someone important out, or be unable to accurately express my appreciation for both the large and small gestures. So please know that this list is not exhaustive, but more of a sampling of the gratitude that I have felt throughout my PhD.

The first thanks go to my supervisor, Dr. Richard Kavanagh, who has expertly guided me through the challenges of academia, knowing exactly when I needed focused advice, and when I needed a general pep talk. I value and respect your knowledge and opinions, and count myself lucky that you answered what must have seemed like a completely random e-mail from Canada so many years ago. Thank you also to my examiners, Dr. Saaj and Dr. Guangbo Hao, for the genuine interest that you displayed in my research, and for the thoughtful comments and suggestions regarding this thesis.

Sincere thanks to the Irish Research Council and the Natural Sciences and Engineering Research Council of Canada, who funded the work described in this thesis and provided me with this life-changing opportunity. Thank you also to the Tyndall National Institute for providing the wireless inertial measurement unit used in the final work chapter, with special thanks to Dr. Michael Walsh for his assistance in the early stages of its use.

Thank you to all of the staff in the Electrical and Electronic Engineering department at University College Cork who have helped me during my studies, whether it was assisting with administrative tasks, answering technical questions, or letting me build LEGO robots (err, I mean mentor the next generation of engineers and scientists). Special thanks to Hilary and James for all of your work in getting the SCARA robot up and running initially (despite not having the correct manuals or software), and to Mick and Tim for carefully leveling the robot and building a custom mounting platform that saved me much time and frustration. And of course, a huge thank you to Ralph, who I count as a good friend, and who has an almost magical way of keeping the department running smoothly (and convincing a finicky photocopier to work when I have waited until the very last minute to print student tutorials and surveys).

Thank you to all of the friends – new and old, near and far – who have provided words of encouragement and support over the years. Special thanks go to Carolyn, who was not only able to explain statistics in a way that made sense to this engineer, but also has an uncanny way of knowing just when I need an e-mail from the other side of the world, or a chocolate bar. To the lads and ladies of the postgraduate lab, past and present, thanks for the long tea breaks and the precious memories – you have been my family in Ireland, and you will always hold a special place in my heart. Finally, to the friends that I have made outside of the lab, thank you for giving me a reason to venture beyond the walls of the Electrical and Electronic Engineering building, so that I could admire early tulips and bluebell-covered forest floors. You have displayed strength and bravery when helping me deal with some of the unanticipated bumps in the road while living in Ireland, and I am thankful to call you all friends.

Love and thanks to my family, who have always encouraged me to follow my dreams and embrace adventure, even when it takes me far from home. Not even the whole of the Atlantic ocean and most of the land mass of Canada has been too great a distance for your love and support to find its way to me, and I can never express how much it continues to mean to me. I would also like to mention that I won the Editor's award for the paper that my mom proofread; never underestimate the importance of a mom's stamp of approval.

Finally, to Mark – thank you for keeping me (mostly) sane and balanced, and for your empathetic support through the most trying phases of the PhD. But most of all, thank you for being my reason to keep coming back to Ireland so that I could finish.

Contents

Acronyms

2D	two-dimensional
3D	three-dimensional
ADS	application development system
AI Lab	artificial intelligence laboratory
API	application programming interface
ARCNET	Attached Resource Computer NETwork
CAD	computer-aided design
CAVE	CAVE Automatic Virtual Environment
D-H	Denavit-Hartenberg
DOF	degree of freedom
EAI	External Authoring Interface
GUI	graphical user interface
HMD	head mounted display
iDeMRT	interactive Desktop Mental Rotation Trainer
IIR	infinite impulse response
IMU	inertial measurement unit
IRKA	Initial Rotation Knowledge Assessment
MDI	Manual Data Input

MEMS	microelectromechanical systems
LSB	least significant bit
NURBS	non-uniform rational Bézier spline
PSVT:R	Purdue Spatial Visualization Test: rotation section
PUMA	Programmable Universal Machine for Assembly
SCARA	Selective Compliance Assembly Robot Arm
SSL/E	Sankyo Structured Language/Enhanced
VRML	Virtual Reality Modeling Language
WIMU	wireless inertial measurement unit
X3D	eXtensible 3D
XML	eXtensible Markup Language

Academic Institutions, Companies, and Professional Associations:

AMF	American Machine and Foundry Corporation
BARA	British Automation and Robot Association
IEEE	Institute of Electrical and Electronics Engineers
JARA	Japan Robot Association
MIT	Massachusetts Institute of Technology
RIA	Robotic Industries Association
SAIL	Stanford Artificial Intelligence Laboratory
UCC	University College Cork
Unimation	Universal Automation

Symbols

a_λ	general acceleration along the axis indicated by the optional λ (m/s^2)
$a_{c,\lambda}$	calibrated acceleration, for the accelerometer axis indicated by λ (m/s^2)
$a_{m,\lambda}$	uncalibrated measured acceleration, for the accelerometer axis indicated by λ (LSB)
a_n	normal acceleration that is associated with a curved path (m/s^2)
$\tilde{a}_k$	D-H parameter: link length, measured along the x_k-axis (m)
$a_{s,\lambda}$	partially calibrated (scaled) acceleration, for the accelerometer axis indicated by λ (m/s^2)
b_λ	bias of an accelerometer, corresponding to the accelerometer axis indicated by λ (LSB)
$\tilde{d}_k$	D-H parameter: link offset, measured along the z_{k-1}-axis (m)
F	general force (N)
g	gravity (m/s^2)
K	spring constant (N/m)
L_k	origin of the D-H frame that is assigned to the k^{th} link (unitless)
m	mass (kg)
n	number of links in a serial robot chain (unitless)
N_λ	number of position peaks considered, in the direction described by λ (unitless)
p	statistical significance measure, used in null hypothesis testing (unitless)

q_k	value of the k^{th} variable D-H parameter corresponding to the chosen home position of the robot (units according to $\tilde{\theta}_k$ or $\tilde{d}_k$)
R_λ	rotation matrix corresponding to a rotation about the axis described by λ (unitless)
r	general radius of a circle (m)
r_c	nominal radius of the calibration circle (m)
r_{ee}	distance between the origin of the frame of the accelerometer and the center of the end-effector (m)
$r_{ee,sub}$	distance between the origin of the frame of the accelerometer and the center of the end-effector, as determined by the subset of data corresponding to sub (m)
$r_{m,\lambda}$	mean measured radius of circular path, in the direction of the λ-axis (m)
S_λ	sensitivity of an accelerometer, corresponding to the accelerometer axis indicated by λ (LSB/(m·s^{-2}))
t	statistical measure, relating to Student's t-test (unitless)
T_{k-1}^{k}	homogeneous transformation matrix relating the (k-1)$^{\text{th}}$ and k^{th} frames (unitless)
U	statistical measure, relating to the Mann-Whitney U test (unitless)
v	general velocity (m/s)
$x_{(\_)}$	position along the x-axis of the frame described by the subscript (m)
$x_{p,i}$	the i^{th} x_m-position peak, found by double integrating $a_{m,x}$ (m)
$y_{(\_)}$	position along the y-axis of the frame described by the subscript (m)
$y_{p,i}$	the i^{th} y_m-position peak, found by double integrating $a_{m,y}$ (m)
$z_{(\_)}$	position along the z-axis of the frame described by the subscript (m)
$\tilde{\alpha}_k$	D-H parameter: link twist describing a rotation about the x_k-axis (deg or rad)
β	general rotation angle about the y-axis (deg or rad)

δ	spring displacement (m)
Δ	change; used in conjunction with another symbol (unitless)
ϕ	general rotation angle about the x-axis (deg or rad)
φ_{ee}	angle relating r_{ee} and the y_b-axis (deg or rad)
$\varphi_{ee,sub}$	angle relating r_{ee} and the y_b-axis, as determined by the subset of data corresponding to sub (deg or rad)
γ	general rotation angle about the z-axis (deg or rad)
γ_a	angle between the x_s-axis and the direction of a_n when the starting roll position is at zero (deg or rad)
$\gamma_{a,j}$	angle between the x_s-axis and the direction of a_n for the j^{th} offset of the starting roll position (deg or rad)
γ_c	phase offset in the x_b-y_b plane, relating the frames of the partially calibrated accelerometer and the base of the SCARA (deg or rad)
$\gamma_{c,sub}$	phase offset in the x_b-y_b plane, relating the frames of the partially calibrated accelerometer and the base of the SCARA, as determined by the subset of data corresponding to sub (deg or rad)
η_λ	non-idealities of an accelerometer, corresponding to the axis indicated by λ (m/s^2)
$\tilde{\theta}_k$	D-H parameter: joint angle describing a rotation around the z_{k-1}-axis (deg or rad)
τ	time (s)
ω	general angular velocity (rad/s)
ω_0	fundamental angular velocity around the calibration circle (rad/s)
ψ	angle between the x_b-axis and the direction of normal acceleration (deg or rad)

Subscripts

b	relating to the base frame of the SCARA
i	describes the index of a position peak
j	describes the index of an offset starting roll joint position
k	describes the link number in a serial robot chain, where k=1 describes the link connected to the base joint
m	relating to the frame of the uncalibrated accelerometer
s	relating to the frame of the partially calibrated accelerometer
λ	describes the axis under consideration: x, y, or z

Chapter 1

Introduction

The geometry of pure motion: such is the scope of kinematics in its study of the linear and angular displacements, velocities, and accelerations of a body, without consideration or reference to the forces that may have caused that body's motion. When the body in question is a robot arm, the relative pose and motion of the individual links making up that arm can be examined to determine the cumulative effect on the pose and motion of the robot's hand, more precisely known as its end-effector. Conversely, the pose and motion of the robot's end-effector can be measured and used to determine a set of possible poses and movements undertaken by the intermediate links. Collectively, this is referred to as robot kinematics, and this thesis focuses on the visualization and experimental application of the principles of robot kinematics.

1.1 Kinematics

Kinematics was officially named and recognized as a new and independent branch of science by Ampère in 1834 [1], but its fundamental principles were developed much earlier in an effort to better understand and design machines and mechanisms [2]. Euler's work, performed more than a half a century before Ampère officially proposed the new science, was particularly important in laying the foundation for kinematic analysis. Euler was among the first to describe general motion as being made up of combinations of translations and

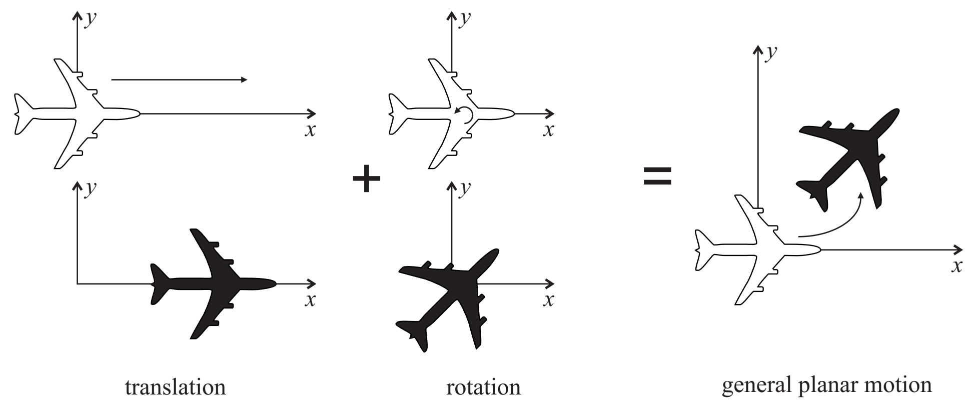


Figure 1.1 The general planar motion of an airplane is shown in terms of a translation and a rotation. The initial poses are shown in white, the final in black.

rotations [3], as illustrated in Figure 1.1, and his manner of measuring motion relative to a Cartesian reference frame [4] is still popular practice today.

However, it is Euler's theorem on rotations that is widely regarded as his most important contribution to kinematics. This theorem states that for every three-dimensional (3D) rotation, it is always possible to find a fixed axis about which that rotation occurred [5]. The rotation axis provides a consistent reference for identifying other properties of a given rotation, such as the angle through which the rotation has traveled. It also allows for the effect of combining successive rotations to be analyzed. Much like 3D translations can be broken down into a series of translations along known axes, so too can a set of rotations made about known axes be found to represent a single 3D rotation made about any rotation axis. For simplicity, these known axes are often chosen to be the axes of the assigned Cartesian reference frame. An important caveat to the combining of rotations is that, unlike translations, they are non-commutative, and therefore the order in which compound rotations are performed affects the final orientation, as illustrated in Figure 1.2.

Early advances in kinematic synthesis tended to ignore the formal analytical techniques being derived at the time, and were instead driven largely by inventors designing mechanisms which were capable of transforming a given input motion into a desired output motion in new ways. For example, the sun and planet gear set [6], depicted in Figure 1.3, was designed as a means to transform linear, reciprocating motion to rotary motion, without infringing on the crank patents of Wasbrough [7] and Pickard [8]. Often, additional benefits of the new mechanism only became obvious once designed and built. In the

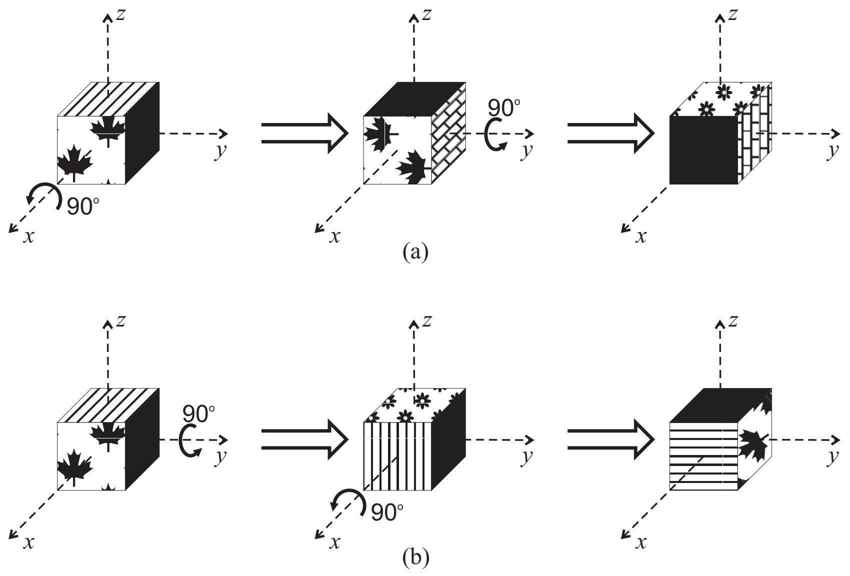


Figure 1.2 a) The progression of poses of a cube undergoing 90° rotations about the x- and y-axes. b) The non-commutative nature of compound rotations is evident from the different final orientation that results when the order of the rotations in (a) is reversed.

case of the sun and planet gear, it was found that the gear ratio could be manipulated to control the number of revolutions made by one complete stroke of the connecting rod [9], unlike the one-to-one transfer of motion performed by the simple crank.

Perhaps one of the most noteworthy advances in early kinematic synthesis came when Watt designed his four-bar linkage [10] to be used in the push-pull action of his famous double-action steam engine, shown in Figure 1.4. Instead of trying to manipulate the motion of the end links of the linkage, as was the

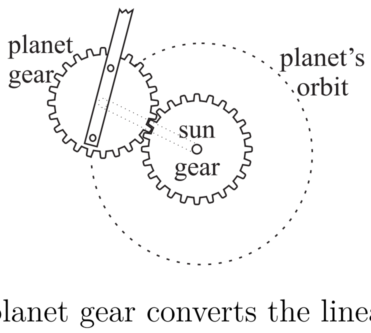


Figure 1.3 The sun and planet gear converts the linear motion of a connecting rod to the rotating motion of an attached flywheel. When an equal number of teeth are present on both gears, the sun gear and attached flywheel each complete two revolutions for a single orbit of the planet gear (corresponding to a single up-down stroke of the connecting rod).

popular approach at that time, Watt focused his attention on the motion of the intermediate link. He observed that an intermediate point on this intermediate link traversed an approximately straight line when the end points of that link made an arc motion, enabling him to transform the linear motion of the steam engine's piston to the rocking motion of the working beam without placing lateral stress on the piston. As a result, Watt's steam engine not only served to power the Industrial Revolution, but it also inspired the likes of Chebyshev [11], Peaucellier [12], and other mathematicians, to examine the kinematic curves that could be produced from the motion of the intermediate links in multi-bar linkages.

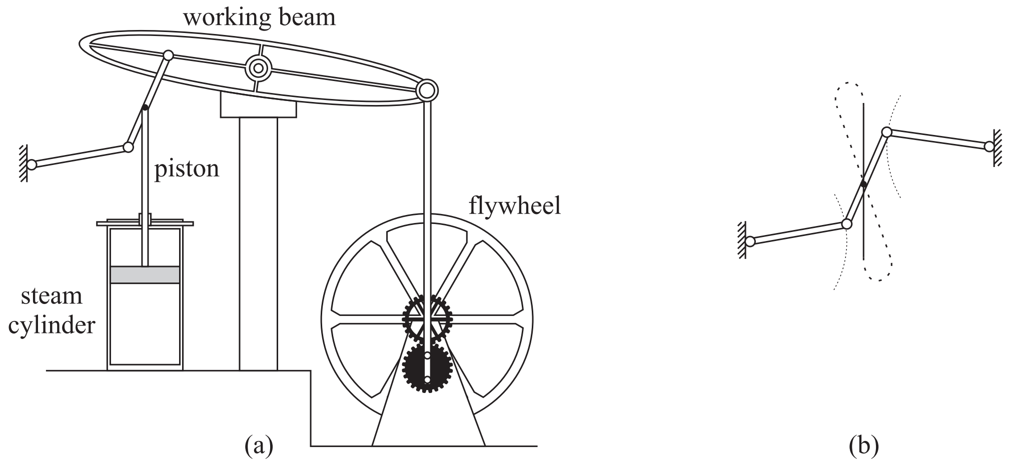


Figure 1.4 a) A simplified representation of Watt's double-action steam engine, and b) the figure-eight path that could be traveled by the intermediate point of his four-bar linkage. Motion of this four-bar linkage was restricted so that the intermediate point of the intermediate link only moved along the approximately linear section that is highlighted.

The age of modern kinematics is considered to have started when Reuleaux observed that the relative motion between two parts was dependent on the physical form of the surfaces connecting those parts [13]. He called each set of adjacent surfaces a kinematic pair, the properties of which could be uniquely identified based on the type of motion that it produced. For example, the working surfaces of the kinematic pair shown in Figure 1.5a constrain the motion to that of a rotation about a single axis. On the other hand, working surfaces resembling that of a prism, like the kinematic pair shown in Figure 1.5b, restrict the motion to that of a linear translation along one axis.

Reuleaux was able to show that kinematic pairs that produced identical motion should be treated identically, and he even introduced a set of symbols to represent the types of kinematic pairs that he had identified. Although his symbolic notation was an incomplete kinematic description, it did allow him to prove that certain mechanisms were kinematically identical, despite having very different physical appearances.

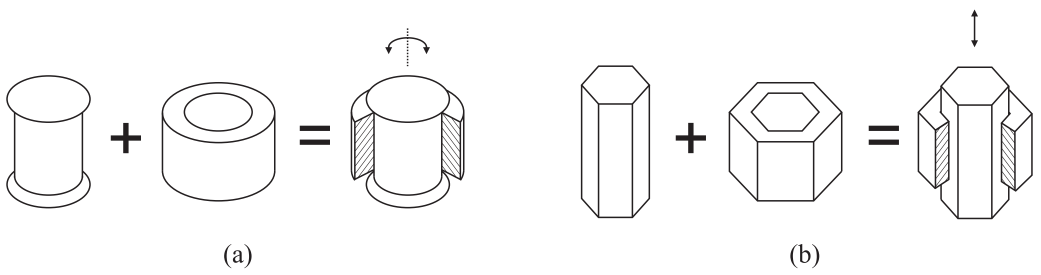


Figure 1.5 a) The solid and hollow cylinders exemplify working surfaces that form a revolute joint. b) The enclosed prism exemplifies a pair of working surfaces that form a prismatic joint. (Based on Figures 40 and 41 of [13].)

1.2 Robot Kinematics

Current methods of analyzing the kinematics of a robot arm make extensive use of Reuleaux's kinematic pairs to construct the arm's corresponding open kinematic chain. Such chains can be viewed as a simple series of connections, with the kinematic pairs – or joints – separating the chain's links. The most widely employed convention for creating and analyzing these kinematic chains was introduced by Denavit and Hartenberg in 1955 [14], in which all robot arm joints are represented as a revolute joint or a prismatic joint, so as to constrain the motion of the intermediate links to one degree of freedom (DOF). Joints that allow for higher DOFs are represented as a combination of revolute and prismatic joints; for example, a spherical joint is represented by three mutually orthogonal revolute joints, separated by links of zero length, and a cylindrical joint is represented by a prismatic joint and a revolute joint, separated by links of zero length.

The Denavit-Hartenberg (D-H) convention systematically assigns reference frames and four parameters to each one DOF joint, making use of the common perpendicular axis between adjacent joints to do so. These four parameters provide geometric information about the relative pose and structure of each link, effectively describing the translations and rotations required to move from one joint to the next. When used to build the homogeneous matrices corresponding to these translations and rotations, the D-H parameters add the mathematical description that was missing from Reuleaux's symbolic kinematic notation, and provide a method to determine the motion and pose of the robot's end-effector, in the case of forward kinematics, or of the intermediate links, in the case of inverse kinematics.

1.3 Thesis Motivation and Aims

Although there are significant mathematical advantages to representing a robot's kinematics concisely, there are also disadvantages to such representations. The main disadvantage of using a model as concise as that introduced by the D-H convention is that the physical correlation between the kinematic parameters and the structure and pose of the actual robot is not intuitively obvious, especially to those who are new to the field of robotics. This can make it difficult for students to effectively visualize the resultant 3D robot structures, and in turn, learn the principles of robot kinematics. This is evidenced by relatively poor past performance on relevant examination questions in robotics courses offered by the Department of Electrical and Electronic Engineering at University College Cork (UCC). Thus, the major aim of the first part of this thesis is to investigate the impact of introducing visualization tools to aid in the teaching of fundamental robot kinematics concepts. Specifically, virtual reality based tools are introduced to help students visualize fixed and mobile frame compound rotations, and to visualize robots defined solely by their D-H parameters.

Ultimately, the main objective of learning about robot kinematics is to apply the theory to the planning and programming of a robot's motion. However, mechanical inconsistencies of real robot arms limit the practicality of applying kinematic models that are based on a manufacturer's set of specified kinematic parameters. Component manufacturing tolerances, arm assembly discrepancies, the maintenance and replacement of parts, and variable joint compliance due to part wear and tear, all contribute to physical kinematic parameters that are unique to a given robot at a given time, and which are bound to differ from those specified by the manufacturer. Even small differences in individual parameters can contribute to significant errors in the accuracy of the placement of the robot's end-effector, depending on the motion. As such, external sensors are often used to track the true motion of the end-effector, either to create a more accurate model of the robot, or to be used as feedback in the direct control of the robot. Due to their small size and low cost, accelerometers are a popular sensor to use for such tasks, either on their own or in combination with other sensors. As with all sensors, the accuracy of the information provided by an accelerometer depends directly on the accuracy of its calibration. The accuracy of the calibration is, in turn, affected by how the accelerometer is attached to the robot. Thus, the major aim of the second part of this thesis is to design and validate an accelerometer calibration routine that can be performed with the accelerometer already attached to the robot.

1.4 Thesis Layout

This thesis is organized into seven chapters, the remainder of which are dedicated to providing details on the common materials and individual methodologies employed to meet each of the aims identified in this introductory chapter.

Specifically, **Chapter 2** presents the reader with relevant background information on robots, starting with a detailed definition of what a robot is considered to be for the purposes of this thesis. A summary of significant milestones in the development of the field of robotics is then given, in order to place the subsequent review of the current state of the field in its proper context.

Chapter 3 introduces the major hardware and software components used throughout this thesis. In particular, it discusses the virtual reality software and the wireless inertial measurement units (WIMUs) used, as well as details of the Sankyo SR8408 SCARA robot and its relevant software. Because virtual reality and inertial sensors are expansive fields of research in and of themselves, this chapter provides only the details required to understand the work in this thesis. Any relevant operational theory that is not included elsewhere in the text is included in this chapter.

The visualization of concepts that are critical to the understanding of robot kinematics is considered in the following two chapters. **Chapter 4** introduces the theory associated with fixed-axes and mobile-axes compound rotations, and explores the difficulties that students in university-level robotics courses have in visualizing and applying the main principles of compound rotations. It describes the Rotation Tool, a desktop virtual reality tool developed by the author to help students visualize the effects of compound rotations. The chapter also discusses the impact on student confidence and understanding that the introduction of this tool has had on students at UCC.

Compound rotations are combined with 3D translations in **Chapter 5**, as the theory of forward kinematics according to the D-H convention is presented in detail. The difficulty of visualizing exactly what each D-H parameter represents when using traditional teaching methods is examined, and the value of introducing Build-A-Robot, a second desktop virtual reality tool developed by the author, is explored. The use of MATLAB and the Simulink 3D Animation Toolbox to directly manipulate the geometric dimensions of virtual shapes used in Build-A-Robot's virtual scene is emphasized as an ability which, to the best of the author's knowledge, has not been formally documented by others.

Chapter 6 applies the principles of robot kinematics that are visualized in Chapters 4 and 5 to plan the motion of a real robot arm. This motion incorporates dynamic accelerations which can be used to calibrate a dual-axis accelerometer that is attached to the robot's end-effector. Because the kinematic model of the real robot is likely to deviate from the model specified by the manufacturer, the calibrating motion is designed in such a way as to minimize the effect of those deviations on the path of the end-effector. This, in turn, minimizes the errors introduced to the measured calibration parameters. Two novel accelerometer calibration routines are discussed and analyzed using experimental data.

The thesis concludes with a final discussion in **Chapter 7**, highlighting the novel contributions presented in this work. It also suggests directions for future work.

1.5 List of Publications

The following journal papers were published as a result of the research described in this thesis:

1. Flanders, M. & Kavanagh, R. C. (2013) Visualizing compound rotations with virtual reality, *Engineering Design Graphics Journal, 77*(3), 15-30.*

2. Flanders, M. & Kavanagh, R. C. (2015) Build-A-Robot: Using virtual reality to visualize the Denavit-Hartenberg parameters, *Computer Applications in Engineering Education, 23*(6), 846-853.

*Recipient of the 2013 Editor's Award for the Engineering Design Graphics Journal.

Chapter 2

Robots

It has been said that there can be no progress without a solid understanding of the past. The objective of this chapter is to impart a fundamental knowledge of how robots have evolved to their current state, thereby providing background information that will be helpful in understanding the later chapters of this thesis. To do so, the term "robot" is first defined, after which a brief history of significant milestones in the development of the field of robotics – from its inception to current day – is given, with a particular emphasis on the contributions to robot kinematics.

2.1 What is a Robot?

The word "robot" is derived from the Czech word *robota*, meaning *forced labor*, and was introduced to audiences for the first time in the 1921 premiere of Karel Čapek's play, Rossum's Universal Robots [15]. Since then, the idea of what a robot is has fluctuated and evolved, and still varies today depending on who is asked. Even Joseph Engelberger, deemed by many to be the father of robotics, reportedly stated that he could not define a robot, but would certainly know one when he saw one. Although many experts in the field of robotics may nod their heads in sage agreement with such statements, it is hardly satisfying for those who are new to the field.

One possible way to define a robot is to do so based on its physical form. For instance, some experts focus only on robots that are fixed in place, like

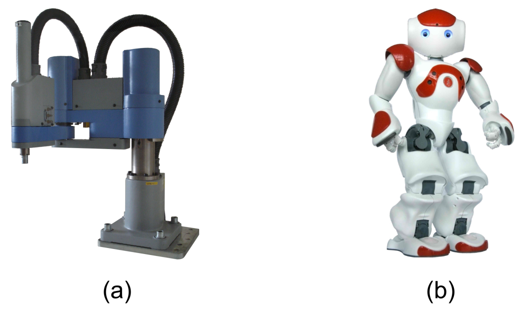


Figure 2.1 a) SCARA robot arm, produced by Sankyo. b) NAO humanoid robot [22], developed by Aldebaran Robotics.

the robot arm shown in Figure 2.1a, while others only consider robots that are capable of moving around the environment freely, like the humanoid robot shown in Figure 2.1b. In both cases, robots can consist of chains of serial links or loops of parallel ones, and these links can be rigid or compliant. Mobile robots can move using wheels, tracks, legs, wings, or some other mechanism. Whether fixed or mobile, a robot's mechanical form may mimic, in part or in whole, that of an insect [16, 17], a snake [18], a bird [19], a four legged mammal [20], a human [21], or something that exists only in the imagination of its creator. Given the breadth of forms that a robot can assume, it quickly becomes clear that using such criteria alone makes it very difficult to pinpoint exactly what a robot is.

Another way to define a robot is in terms of its function. A robot is generally classed as an industrial robot if it is used for manufacturing-related tasks and industrial automation applications. Both the Robotic Industries Association (RIA) [23] of the United States and the British Automation and Robot Association (BARA) [24] only consider industrial robots in their official definitions of a robot, since it was the industrial robot that sparked the creation of the robot industry. Conversely, the Japan Robot Association (JARA) [25], the world's oldest national robot association, takes a broader scope in its definition and also includes service robots. Service robots have more direct interaction with humans in their daily lives than industrial robots do, with duties as diverse as performing household chores [26, 27], serving as an educational companion to children [28], providing personalized motivation to help individuals reach targeted goals [29], and assisting the elderly in hospital or care home settings [30]. Besides the lack of consensus between leading robot authorities, the ability to reprogram and repurpose robots adds to the difficulty of using such criteria to come up with an absolute definition.

How a robot is controlled seems to be the most universally emphasized factor in determining what is and is not a robot, although the bar distinguishing robot from machine tends to move with the advance of technology. In the past, teleoperated manipulators were deemed by many to be robots. The Canadarms and other such teleoperated space manipulators [31] are certainly tremendous feats of engineering, and many would consider them to be examples of the ultimate robot arms. However, based on input from the standards councils of 21 countries, the most recent international standard [32] relegates all teleoperated manipulators to the class of robotic device rather than actual robot. According to this standard, an actuated mechanism must be programmable in two or more axes, and move with a degree of autonomy within its environment while performing tasks, in order to be considered a robot. Therefore, pick-and-place manipulators that rely on mechanical stops [33] are no longer considered to be robots, despite their role in the evolution of robots. On the other hand, all robot authorities agree that machines that can be programmed to follow a trajectory by calculating its inverse kinematics, or decide on its own course of action based on information from sensors and adaptive learning algorithms, are truly robots.

In truth, all three of the above factors play some role in determining whether a machine is deemed to be a robot or not, making it clear as to why the definition is neither absolute nor agreed upon universally. For the purposes of this thesis, only robot arms and their kinematics are considered in detail, and all further use of the term “robot” refers to this small category of what is an ever-growing family of programmable, actuated machines.

2.2 An Historical Overview of Robotics

2.2.1 From Science Fiction to Reality

From the moment that robots first took over the world in Čapek’s renowned play, people have been fascinated by the possibilities and repercussions of introducing robots into the real world. Robots have been the topic of myriad science fiction stories and have made multiple appearances on the big screen, both in the role of hero and villain. Perhaps some of the most famous robot stories are those that were written by Isaac Asimov, the man credited for inventing [34] and popularizing the term “robotics,” and for introducing the three laws of robotics [35]:

1. A robot may not injure a human being, or, through inaction, allow a human being to come to harm.
2. A robot must obey the orders given to it by human beings, except where such orders would conflict with the First Law.
3. A robot must protect its own existence, as long as such protection does not conflict with the First or Second Laws.

Although originally conceived as a story aid to avoid the usual dark theme of evil robots destroying their creators (and later expanded to include a zeroth law [36] when required for subsequent plots), these laws have sparked much discussion about their applicability in real life, and the importance of ensuring safe human-robot interactions. Such debates will likely become only more heated as robots and artificial intelligence grow evermore sophisticated, and as robots become more integrated into people's lives.

Besides sparking philosophical debates about a robot's purpose and operation, Asimov's stories have served to inspire the imagination of countless engineers and scientists. They certainly caught the attention of American physicist and engineer, Joseph Engelberger [37]. It was Asimov's stories that were in the back of his mind when he met George Devol Jr., an accomplished American inventor, at a cocktail party in 1956; and it was the arm of one of Asimov's robots that sprang to the fore of his mind when Devol began to describe his latest patent application [38]. Devol's patent for a programmed article transfer device [39] described a general purpose machine that could perform repetitive tasks automatically. But unlike the specialized, purpose-built, numerical control manufacturing systems that were common at the time, his device was capable of universal automation (or "unimation," as he called it). That is, his device could carry out different tasks without requiring any changes to its mechanical form; all it required was a different program to control it.

The prospect of using a real, programmable robot arm in industrial automation processes was revolutionary in the United States at the time, and this fateful meeting between Devol and Engelberger laid the foundation for the formation of the world's first robot company, Unimation Inc. Under the leadership of Engelberger as the company's first president, they installed what is recognized as the first industrial robot arm, the Unimate, in 1961 to handle hot die-cast pieces in a General Motors plant in New Jersey [40]. Unlike the mobile, Cartesian robot that was described in Devol's original patent, this first

Unimate was a fixed, spherical robot arm, as shown in Figure 2.2, with five hydraulically-powered axes and a magnetic drum memory to store sequences of programmed positions and motions. Although unpopular and more expensive at the time, the Unimate also employed solid-state transistors instead of the more common analog vacuum tubes, a decision that Engelberger has said greatly contributed to the early success of the Unimate over many of its competitors when the cost of these components began to decrease [41]. This first Unimate robot performed various jobs tirelessly for more than a decade, working for 100,000 hours before retiring to the Smithsonian Institution's National Museum of American History, demonstrating the versatility and reliability of robots even in their earliest days.

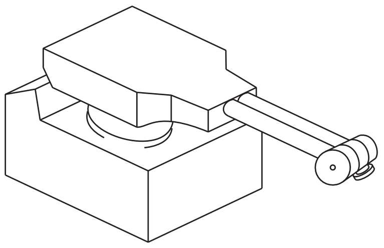


Figure 2.2 The Unimate was the world's first industrial robot arm. (Drawing based on Figure 2.4 in [37].)

2.2.2 Early Robotic Research

The arrival of computers in academic research laboratories prompted the creation and exploration of the brave new world of artificial intelligence, and it was only a matter of time before the ideas in this field were applied to robot control. Some of the earliest research combining computers and robot arms was the work of Heinrich Ernst at the Massachusetts Institute of Technology (MIT). Beginning in 1960, Ernst built a computer-controlled mechanical hand, named MH-1 [42]. The MH-1 system consisted of a motorized master-slave manipulator purchased from American Machine and Foundry Corporation (AMF), and a gripper with multiple touch sensors attached to it. The slave arm, normally controlled by a human operator by moving the master arm, was transformed into a programmable robot by connecting it to a TX-O computer instead. Receiving feedback from the touch sensors, MH-1 could sense objects in its environment and decide what to do with these objects. For example, it could differentiate between small wooden blocks and larger boxes, and could determine the location of each in order to place the blocks into the box. For the first time, a robot arm could interpret the environment that it was interacting with! Although the capabilities of MH-1 were groundbreaking, they were not without error. Ernst admitted that MH-1 reacted unexpectedly approximately 20% of the time, though that only seemed to add to onlookers' excitement when MH-1 completed a task as expected. MH-1 was also not capable of interpreting a dynamic world, assuming that everything in its workspace was static unless it moved an object. Certainly MH-1 represented a monumental first step towards giving robots intelligence, but clearly much work had yet to be done.

Ernst's work encouraged other researchers to conduct their own experiments with computers and mechanical arms. In 1963, an orthotic arm developed by a group at the Rancho Los Amigos Hospital in California was purchased by the recently formed Stanford Artificial Intelligence Laboratory (SAIL). Known as the Rancho Arm, this orthosis was a seven DOF motor-driven exoskeleton that supported and moved a user's arm based on input from a bank of tongue switches [43]. Early experimentation with the arm involved connecting it to a computer and feeding in the joint positions as inputs to various programs [44]. Ultimately, it was used in Stanford's hand-eye project to study the incorporation of vision and obstacle avoidance techniques, where it gained the nickname, "Butterfinger" [45]. However, because the motion of the orthosis was meant to imitate that of a human arm, it had extremely complex kinematics associated with it, and finding a closed-form solution for the inverse kinematics was impossible. Instead, its inverse kinematics had to be solved iteratively [46], a computationally intensive and time-consuming chore for the computers of that time. This, in addition to errors caused when the arm's links were close to their end-points [47], made it difficult to move the arm accurately.

Based on what they had learned from working with the Rancho Arm, researchers at Stanford endeavored to design and build their own robot arms. The Orm, designed by Larry Leifer and Victor Scheinman in 1965, was the first attempt. Named after the Norwegian word for "snake" due to its serpentine structure and motion, the pneumatically-powered Orm consisted of 24 inflatable sacks arranged between seven metal disks, as depicted in Figure 2.3a. A digital approach was taken, and it was decided that each sack could represent one of two states: fully inflated or completely deflated. Limiting the number of states of each joint was intended to simplify the interface to a digital computer while still allowing the arm's end-effector to reach a finite, but large, number of positions. However, it quickly became clear that the position of the arm was undefined while the sacks changed state, causing the arm to thrash about wildly until steady-state was again attained [48]. In addition, a simple digital pattern could not be found that related the states of the sacks to the end-effector's position, because positions that were close in Cartesian space often had very different joint configurations [47]. Such issues made it difficult to produce repeatable motion with the Orm, and work on the arm was abandoned in favor of an analog hydraulic design.

The Hydraulic Arm designed at Stanford between 1966 and 1968 was developed with the primary goal of creating a manipulator that could move as fast as the computer could send it instructions, and was used to study the dynamics of robot arms [46]. It had six DOFs, not including the end-effector, and was both fast and powerful. In fact, it moved with such force that it shook the whole room that it was housed in when accelerating, and the floor had to be reinforced with heavy steel I-beams [52]. As was common practice in research laboratories at that time, the computer that it was connected to was timeshared between multiple projects. However, the Hydraulic Arm often required full computational power to perform coordinated trajectory and motion planning in real-time, and would lock out other users to operate in what was dubbed, "spacewar mode." Despite the speed that this arm offered, the mess created by leaking hydraulic fluid, and the danger it posed to bystanders when making unexpected movements, made it impractical for long term use.

With the intention of overcoming some of the major difficulties encountered with the Orm and Stanford's Hydraulic Arm, an electrically-powered robot arm was developed by Scheinman at Stanford in 1969 [48]. Known simply as the Stanford Arm, this was the first electric arm to be controlled by a computer. The Stanford Arm had six DOFs, and consisted of five revolute

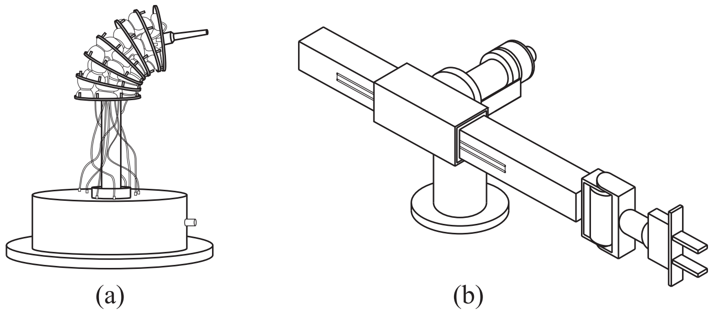


Figure 2.3 Two of the early robot arms developed at Stanford: a) the Orm, and b) the Stanford Arm. (Drawings based on photographs [49, 50] and Figure 3 in [51].)

joints and one prismatic one, as can be seen in Figure 2.3b, making it unlike any of the other arms that had been studied at Stanford thus far. Scheinman designed the geometry of this robot with the specific goal of having well-defined kinematics, ensuring that closed-form inverse kinematics solutions could be calculated quickly and precisely by the computer that it was attached to [46]. The Stanford Arm also had brakes installed on all of its joints so that a particular position could be held when a program was paused or power to the arm was removed, making it ideal for study. In fact, the success and popularity of the Stanford Arm among researchers was such that over ten copies of it were made for use in various university, government, and industrial research laboratories. Two Stanford arms were used as part of Stanford's hand-eye project for more than 20 years to perform tasks, such as the assembling of a Model-T Ford water pump based on vision and touch feedback [53].

Of course, the work initiated by Ernst at MIT also continued in that institution's artificial intelligence laboratory (AI Lab). A Versatran (versatile transfer robot) was purchased from AMF for use as an early experimental platform, but its simple kinematics hindered efforts to develop flexible obstacle avoidance techniques [54]. Frustrated by the limitations imposed by existing manipulators, Marvin Minsky, co-founder of MIT's AI Lab, set out to design a more flexible arm. By 1966, he had designed an arm with eight DOFs [55] that moved like an octopus' tentacle. This tentacle arm was modular in structure, and grew over the years to an impressive 14 DOFs [56]. It was hydraulically-powered and strong enough to lift an adult, but could also be controlled by a flexible joystick with enough precision to gingerly embrace a child [57]. Such precise control was a huge improvement over the dangerous power of Stanford's Hydraulic Arm and the wildly unpredictable motion of the Orm. With so many DOFs, however, solving the inverse kinematics for Minsky's tentacle arm was not trivial. In fact, the flexibility of the arm relied on the fact that there was often more than one solution to the inverse kinematics, thus allowing the arm to avoid obstacles and approach a target position from a variety of directions. Taking advantage of the symmetry of the arm allowed for the simplification of the calculations. By considering the two halves of the arm separately – one with the known shoulder position, and one with the desired hand position – two sets of valid central joint positions could be found. The intersection of these sets provided a valid solution set for the entire arm, and the most appropriate overall position could be chosen from this set based on factors such as initial position and location of obstacles [55].

Research combining vision and computer-controlled robot arms was also conducted at MIT, with efforts in the 1970s focusing on the control of miniature systems that could assemble, inspect, test, and repair electronic circuit boards. In the autumn of 1972, Scheinman visited MIT for a few months to design a miniature arm that would be suitable for such applications [58]. Standing at just over 27 cm high [59], this mini-arm had six revolute joints and a workspace radius of approximately 40 cm [60]. It could carry payloads of up to 1.5 kg, and complete simple movements in less than a second. Like the Stanford Arm before it, Scheinman designed this arm to have relatively simple kinematics, with the axes of the final three joints intersecting. All of these features made it an ideal research platform, and multiple copies were built for various research centers.

By 1973, David Silver had also built a robot system capable of small-scale assembly at MIT [61]. The Silver Arm had four DOFs, two of which were provided by a small platform that could move in the x-y plane. The other two DOFs were on the arm itself, allowing it to travel vertically along the z-axis, and to rotate the gripper. It also incorporated eight different force feedback mechanisms, enabling it to execute fine motions with a repeatability of ± 0.08 mm, making it ideal for electronic assembly applications.

Meanwhile, on the other side of the Atlantic Ocean, great strides were being made in artificial intelligence and robotics in the University of Edinburgh's Department of Machine Intelligence. Prototypes for robotic systems were built in this research laboratory starting in 1969 [62], allowing researchers to experiment with different mechanical designs before building Freddy II in 1972 [63]. Freddy II was a five DOF robot system, and like the system that Silver would build a year later, it consisted of a robot arm suspended over a mobile platform. Freddy II was more than 300 times larger than Silver's system, however, and had an extra DOF in the robot's wrist that allowed it to turn objects over. As large as it was, it could still lift objects as small as 2 mm in diameter. Given a random heap of wooden parts, Freddy II could pick out the parts it recognized and use these parts to assemble a toy car or a toy boat [64]. It incorporated vision to identify and separate individual pieces from the heap, but relied on touch alone when assembling the structures. This approach meant that researchers had to explicitly determine and program end-points that the robot was to move to in order to carry out the assembly process, a tedious and time-consuming process that had to compensate for positioning inaccuracies. This encouraged the creation of RAPT [65], an object-level robot programming language that allowed the robot to use relative spatial relationships between the different wooden parts to determine its trajectory, instead of manually entered positions [66]. This approach relied on determining the position and orientation of a part relative to a fixed world frame, making extensive use of calculations involving homogeneous transformations, the theory of which is discussed in further detail in Chapter 5.

2.2.3 Collaboration Between Academia and Industry: The SCARA

As University researchers developed the science of robotics, leaders in the growing robot industry began to realize the potential of incorporating the ideas developed in academic laboratories into the robots that they built and sold. One prominent example of academic researchers collaborating with members of industry to create a revolutionary robot design occurred in Japan during the late 1970s. Hiroshi Makino at Yamanashi University was experimenting with providing force feedback to numerical control machines, in an attempt to improve the accuracy of automated assembly processes. He was especially interested in the problem of accurately inserting a peg into a hole, something that Freddy II at the University of Edinburgh was able to accomplish by employing a spiral search method that incorporated force feedback [64]. While using his numerical control machine to stack LEGO pieces, Makino observed that the force feedback was not always necessary to successfully join two pieces. In fact, he observed that simply allowing a small amount of compliance in the horizontal plane while pushing down firmly in the vertical direction could accommodate any slight positioning inaccuracy, resulting in the successful joining of the building blocks [67].

Based on this observation, Makino designed the Selective Compliance Assembly Robot Arm (SCARA), completing the set of assembly drawings for the first prototype in 1978. In that same year, Makino also established a workshop to discuss his findings with companies in industry, and to explore options for building physical prototypes to study in real factory settings. Such prototypes were impossible to build with the limited funds available for Makino's research alone, and the collaboration encouraged by the consortium resulted in a wider range of ideas and applications, as well as faster acceptance by industry. The consortium continued for three years, yielding two prototypes for the group to experiment with, and by 1981, both Sankyo Seiki and Nitto Seiko had developed commercial SCARA models.

Although the exact layout of joints of a SCARA-type robot may vary, as shown in Figure 2.4, the general four DOF SCARA structure typically includes three revolute joints and one prismatic joint, with the axes of motion all directed along the vertical z-axis. This configuration ensures that vertical forces, including gravity, do not affect the angular position of the revolute joints, and negates the need for brakes to constantly hold these revolute joints in place [68]. This, in turn, means that compliance in the x-y plane can be incorporated while still maintaining rigidity along the z-axis.

The SCARA was the first robot to be designed completely in Japan, and its ability to place objects precisely at high speeds soon made it a popular choice on assembly lines in that country. Its relatively large workspace compared to its small footprint makes it ideal for factories and research laboratories where floor space is limited. And although Makino laments the lack of a master patent for the SCARA [67], it was the ability to use the design freely that likely contributed to the proliferation of the SCARA in industry and academia.

As a testament to the strengths of its design, the SCARA configuration is still used extensively in industry and research laboratories today, more than three decades later. Its theoretical kinematics have been well studied, and it is often used as an example to illustrate important concepts in University robot courses. Further, the high repeatability of the SCARA makes it an ideal candidate to perform the work discussed in Chapter 6.

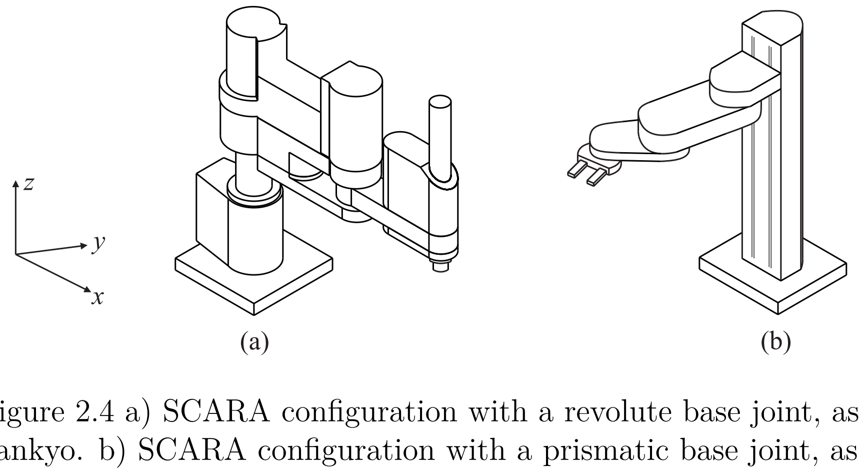


Figure 2.4 a) SCARA configuration with a revolute base joint, as produced by Sankyo. b) SCARA configuration with a prismatic base joint, as produced by Precise Automation.

2.3 Current State of Robotics

The robot industry has undergone considerable change since its inception in 1961, especially in terms of its most influential participants. Early industry leaders like Unimation, although so crucial to the establishment of the industry, have long since had their robot lines taken over by competing companies [69, 70]. Other new leaders have emerged over the years, with many of the parent companies located in Japan. In fact, the vast majority of the world's robots have long been produced and used in Japan, followed by the United States, Germany, and the Republic of Korea. However, the demand for robots is rising in other countries, as exemplified by the fact that, for the first time, the number of robots purchased for use in China in 2013 exceeded the number of new installations in Japan in that year [71].

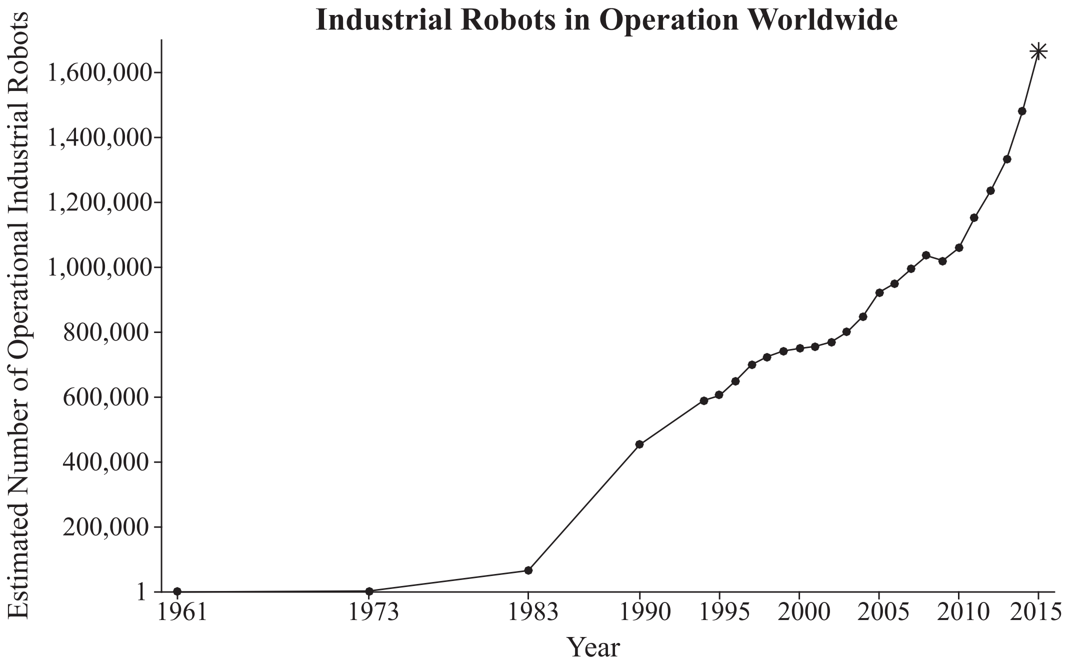


Figure 2.5 All confirmed data points (•) and projected data points (∗) are supplied by World Robotics [71–83]

Despite the rise and fall of numerous robot manufacturers, the industry as a whole has continued to grow for more than half a century. As Figure 2.5 indicates, the growth in operational robots throughout the world has risen substantially in the last four years, and it is projected that the number of robots operating worldwide at the end of 2018 will be well over two million [83]. Although the majority of robots are still involved in jobs like handling die-castings, spot welding, and spray painting in the automotive industry, a growing number are also being employed in the electronics, food preparation, and pharmaceutical industries. In these industries, robots are tasked with handling a wide variety of materials while inspecting, packaging, and palletizing the final products for shipping. In the electronics industry, robots are now involved in all aspects of electronic circuit board production and testing, as envisioned by researchers at MIT in the 1970s. Of course, today's robots must possess much greater resolution than those designed forty years ago, in order to properly place and test the exponentially smaller [84] electronic components within the circuit.

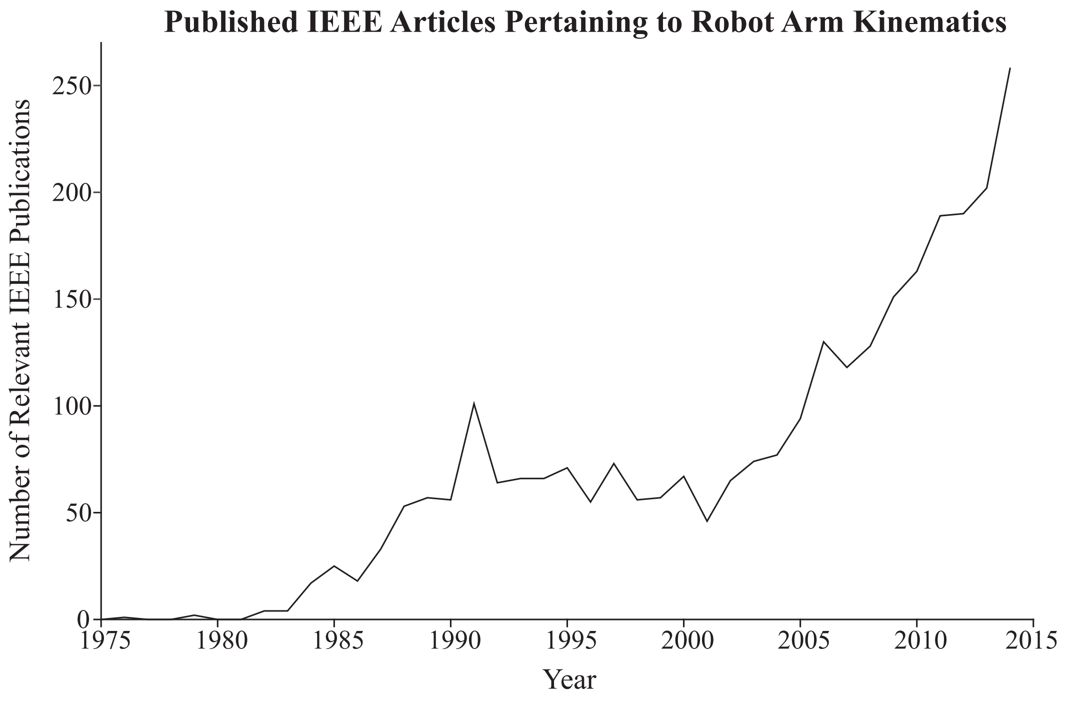


Figure 2.6 This plot of the yearly IEEE publication count relating to robot arm kinematics was generated by the keyword search ("robot" and "arm" and "kinematic") of metadata in IEEE *Xplore*.

Robot-related research continues to play an important role in the expansion of the industry. For example, the development of sophisticated vision and machine learning algorithms has recently allowed robots to efficiently identify and sort overlapping and irregularly shaped construction and demolition waste for recycling [85]. Such developments have also allowed for the creation of robot arms capable of locating and picking ripe fruit, vegetables, and ornamental cut flowers [86]. Research concentrating specifically on robot arm kinematics has been on the rise during the last decade, as illustrated in Figure 2.6, demonstrating the importance of this topic as robot systems and applications become more complex and require higher accuracy. Advances in robot accuracy mean safer interaction with humans, and nowhere is this more apparent than in the growing use of autonomous robots in the medical field [87]. Whether used to precisely deliver radiation to a patient's tumor without damaging surrounding healthy tissue [88], to perform delicate biopsies [89], or to execute exact orthopedic surgeries [90], the accurate positioning of a robot's end-effector is critical to its safe and effective use in medical procedures.

Regardless of the end application, robots are increasingly being required to move with a higher level of resolution and accuracy than humans are capable of accomplishing on their own, making it impossible to employ traditional, interactive programming techniques that involve manually leading the robot through taught points. Instead, the robot must calculate appropriate trajectories to move to the end-effector positions that will satisfy high-level programming commands. Of course, the accuracy of these calculations can only be as accurate as the underlying kinematic model used by the robot, as explored further throughout this thesis.

Chapter 3

Hardware and Software

Having presented a comprehensive summary of relevant background information on robotics in the previous chapter, the objective of this chapter is to do the same, albeit on a smaller scale, for the three main components used to carry out the work described in this thesis. The field of virtual reality and the software options for authoring virtual reality based educational applications are considered first, with important terms defined, and the choice of software used for the work in Chapters 4 and 5 justified. A description of the fundamental concepts of the chosen software, complete with a simple working example, is also provided. The second section considers the WIMU hardware, and explains the theory of how its triaxial accelerometer operates. Finally, the chapter concludes with a description of the hardware and software components of the SCARA system, which is used in the accelerometer calibration routine described in Chapter 6.

3.1 Virtual Reality

The field of virtual reality is currently experiencing a surge in renewed interest, thanks in no small part to the publicity generated by the release of products like Nintendo's Wii, Microsoft's Kinect, Google's Cardboard, and the Oculus Rift head mounted display (HMD). The marketing for such products largely target entertainment applications – for example, interactive 3D video games and stereoscopic personal theaters – but the increasing accessibility of virtual reality technology means that other applications are becoming more prevalent. For example, virtual reality technology has been used in the rehabilitation of patients with neurological diseases [91] and patients who have suffered strokes [92]. And although engineers at the Walt Disney Company do use virtual reality technology to entertain their theme park guests, they also use it as a design tool to assess the performance and safety of new theme park attractions [93]. Similarly, architects use virtual reality as a means to collaborate with colleagues [94] in the design of new buildings, and to identify potential problems [95] before construction begins.

Educational and training applications are areas that have benefited, and continue to benefit, from the application of virtual reality. Flight simulators served as one of the earliest applications in training, allowing pilots to learn how to use the complex instrumentation in a model cockpit and to respond to a variety of potentially dangerous scenarios in a safe environment [96]. Archaeological and anthropological students can gain unique insights into the cultural heritage of a region by exploring the streets of the past in virtual constructions of ancient settlements [97]. Virtual models of damaged or fragile artifacts provide a means for archaeological students and tourists alike to examine objects that would otherwise by inaccessible [98]. Medical students can interact with 3D virtual anatomical models, not only to learn about anatomy [99, 100], but also to rehearse surgical procedures [101]. Virtual scientific visualizations can help students to understand the role of invisible forces, such as those between molecules [102]. Virtual laboratories [103] provide science and engineering students with hands-on experience in distance learning applications. Robotic virtual laboratories, in particular, allow students to safely access sensitive and potentially dangerous machinery, while eliminating the prohibitively expensive cost of attaining, housing, and maintaining such specialized hardware.

It is clear from the above examples that virtual reality is being incorporated into a growing number of diverse applications. The increasing popularity of the field means that the phrase “virtual reality” is often used, and misused, leading many to wonder what, exactly, virtual reality is.

3.1.1 Definitions

It can be argued that virtual reality has its roots in the theater [104], where actors have endeavored to create believable alternate worlds for audiences to escape to for centuries. In fact, some of the earliest technological innovations relevant to the field of virtual reality were aimed at improving this sense of cinematic immersion. For example, the stereoscopic television apparatus patented by Henry McCollum [105] in 1945 allowed users to view 3D scenes that differed from the real scene in front of them by using cathode ray tubes to project images onto the lenses of a pair of glasses worn by the user. Morton Heilig's telesphere mask [106] and Sensorama simulator [107] also displayed stereoscopic scenes, but added auditory and olfactory cues to provide users with an even greater sense of immersion.

Although such inventions created compelling immersive experiences, the true potential of virtual reality was not considered until Ivan Sutherland wrote of an ultimate display in 1965 that was capable of responding to user inputs [108]. He argued that in order to create a convincing 3D illusion, a computer-generated scene must maintain a perspective that is consistent with the user's point of view. His HMD [109], known as the Sword of Damocles, accounted for this "kinetic depth effect" by tracking the position and orientation of the user's head, and updating the displayed wireframe scene accordingly.

Despite these early advances, it was not until the 1980s that the term "virtual reality" gained popularity, when VPL Research released their DataGlove [110] and EyePhone [111] – the first commercially available sensor glove and HMD. Jaron Lanier of VPL used the term to describe the interactions that were possible between multiple individuals in a computer-generated world [112] using technology like that created by his company, but the phrase quickly became associated with the technology itself. This association persists today, with prominent dictionaries specifically citing sensor glove and HMD technology in their definitions [113]. However, as the field of virtual reality has grown, so too has the availability of alternative technology for viewing and interacting with virtual scenes. For example, desktop computer screens and CAVE Automatic Virtual Environments (CAVEs) [114] – virtual reality theaters that surround users with seamless projections of images on the walls, floor, and in some setups, the ceiling – can be used instead of HMDs to achieve varying levels of immersion. Sensor gloves [115] can be replaced with full-body inertial suits [116], omnidirectional treadmills [117], cameras [118, 119], ultrasonic sensors [120], or a fusion of these technologies to capture motion and allow users to interact with virtual worlds in natural ways.

Instead of focusing on the evolving technology associated with the field, Steuer [121] argues that virtual reality should be defined by the type of human experience offered. This experience is determined by the level of presence achieved by a communication medium, or the feeling of actually being in the virtual world. Both immersion and interaction play a role in achieving presence. The level of immersion achieved depends on how sensorially vivid a virtual world is, taking into account both the number of senses stimulated and the degree to which each is stimulated. Therefore, virtual reality systems that rely on low quality visual displays are not as immersive as systems with high quality visual displays, or ones that also present auditory and haptic feedback. The level of interaction achieved depends on the speed at which a user's input affects the virtual environment, the number of different input options available to a user at a given time, and how natural these input options feel to the user. Thus, virtual reality systems with slow response times and what seem to be unpredictable responses to a user's actions result in frustrated users and implausible virtual environments. Finally, the inherent willingness of a user to believe in the virtual world also plays a role in the level of presence achieved, but since this varies widely between individuals, only the technical aspects of incorporating immersion and interaction will be considered in this thesis.

3.1.2 3D Software Options

When designing virtual reality based educational tools that are meant to be used by an entire class of students, the most practical and affordable solution, in most cases, is to implement a desktop application. Without the specialized hardware found in CAVEs and HMD implementations, desktop virtual reality applications must rely heavily on visual effects to provide users with a sense of immersion. In order to maximize affordability and accessibility, desktop virtual reality applications must also use the input from common devices like a standard mouse and keyboard to enable interaction that feels natural. Fortunately, advances in the field of computer graphics mean that a plethora of software options exist for authoring high quality, interactive 3D images on desktop screens [122]. A search of the relevant literature reveals five popular, freely-available, platform-independent software options that can be used to create interactive virtual worlds: OpenGL, Java 3D, Virtual Reality Modeling Language (VRML), eXtensible 3D (X3D), and WebGL. The relative popularity of each over the last twenty years is indicated in Figure 3.1.

OpenGL [123] is a low-level application programming interface (API) that provides standardized access to graphics hardware, and as such, can be used

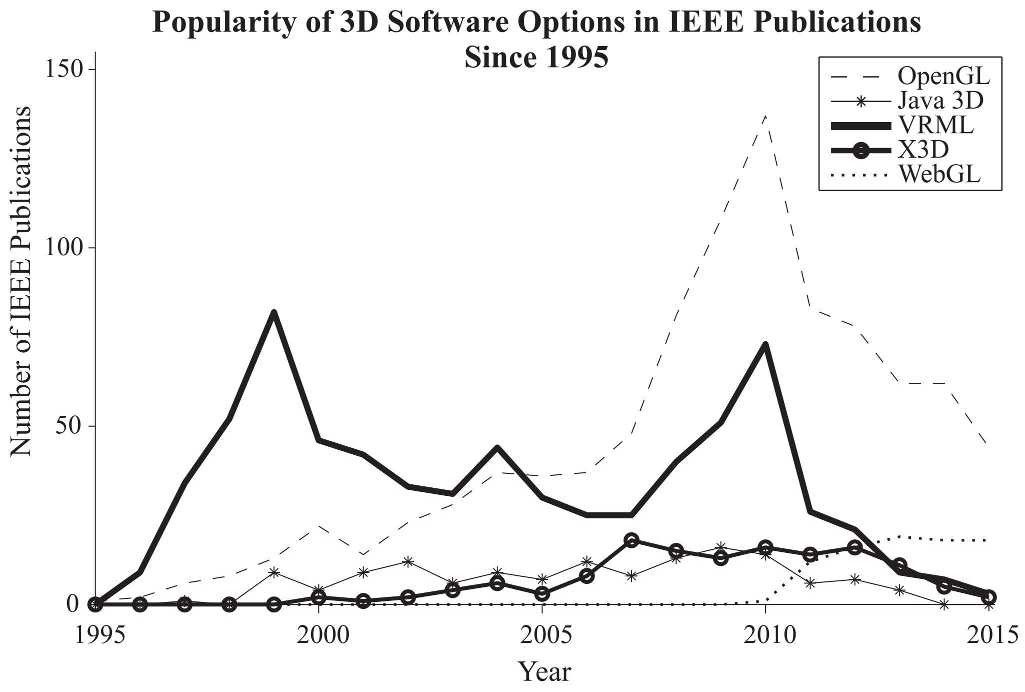


Figure 3.1 This plot of the yearly IEEE publication count relating to the different 3D software options was generated by relevant keyword searches ("OpenGL"; "Java 3D" or "java3d"; "VRML"; "X3D"; and "WebGL") of metadata in IEEE *Xplore*.

on its own or as an interface to higher-level APIs. This, combined with the fact that OpenGL can be used to render two-dimensional (2D) scenes as well as 3D, may partially explain the popularity of OpenGL that is indicated in Figure 3.1. First released in 1992 by Silicon Graphics Inc. [124] as an open version of their proprietary graphics library, OpenGL allows 3D objects, defined as a collection of points, lines, and triangles, to be rendered consistently across multiple platforms, without having to consider the details of windowing or interfacing with specific input devices. OpenGL is shader-based, meaning that programmers must write custom shader programs (for example, vertex and fragment shaders) to explicitly define how individual pixels are displayed on a screen, in addition to an application program that describes the higher-level geometry and behavior of objects. Although this gives programmers complete control over how a virtual object is rendered on the screen, it also requires an in-depth knowledge of the underlying theory and mathematics of computer graphics that beginners typically do not possess. Unlike other low-level APIs, such as Microsoft's proprietary DirectX, OpenGL is platform independent, and its application programs can be written in a number of high-level programming languages in order to incorporate logic and control. C/C++ is a popular choice, which means that the application program must ultimately be compiled to run on a specific platform, largely erasing the benefits of OpenGL's platform independence from an end-user's perspective.

Java 3D [125] is a high-level, object-oriented API that enables programmers to build a scene graph using Java classes and methods. This scene graph is a hierarchical tree-like structure that connects individual objects to create an interactive 3D virtual scene that can be viewed on any Java-enabled platform as either a Java application or a Java applet. Logic and control can be added to this virtual world by utilizing additional Java APIs. The details of how the scene is to be rendered is typically left to Java 3D to optimize, before passing on instructions to a low-level API like OpenGL. However, experienced programmers are able to manually control aspects of the scene rendering, should they choose to do so. First released in 1997 by Sun Microsystems, Figure 3.1 shows that Java 3D did not gain the same level of popular usage that OpenGL and VRML did, and although it is currently being maintained by JogAmp, Sun Microsystems (now Oracle) has stopped the active development of Java 3D in order to concentrate on newer projects.

Unlike OpenGL and Java 3D, VRML is not an API. Instead, it describes a standard file format that is recognized by modern browsers, and was proposed in 1994 as a means to bring interactive 3D visualizations to the Internet, much like HTML was developed to bring text and 2D media to the Internet. Accepted as an international standard in 1997 [126], VRML 2.0 (also known as VRML97) employs an acyclic scene graph to describe virtual worlds, which are most typically rendered in a 2D web browser, with the help of a plug-in. VRML objects have limited abilities to drive events and make a scene dynamic, but are able to interface with high-level programming languages, like JavaScript and Java, to incorporate logical control into virtual worlds. As is evident from Figure 3.1, VRML gained early widespread acceptance among researchers, and although its relative popularity has declined in recent years, it remains a viable option that is well supported by computer-aided design (CAD) modeling packages like SolidWorks and Pro/ENGINEER, and scientific simulation packages like MATLAB's Simulink 3D Animation toolbox.

X3D [127] is also a standard file format, and was created by the same consortium that created VRML to become VRML's successor. As such, it supports additional functionality compared to VRML, including the ability to specify geospatial positions, include programmable shaders, and generate curved lines and surfaces with non-uniform rational Bézier splines (NURBS). X3D files can be encoded using an eXtensible Markup Language (XML) format [128] that can be validated and allows for interaction with other web languages. Alternatively, X3D can be encoded using a classic VRML format [129] to allow for backwards compatibility with VRML97, or using a compressed binary format [130] that minimizes the size of files to decrease transmission and loading times. The X3D international standard was first accepted in 2004, but as is indicated in Figure 3.1, has yet to gain the widespread popularity achieved by VRML. This may partially be because VRML remains so well supported by proprietary products, providing little incentive for users to switch to X3D. For example, the Simulink 3D Animation toolbox for MATLAB did not support X3D until 2015, and VRML is still currently the default file format accepted by that toolbox.

WebGL [131] was first released in 2011, and as Figure 3.1 shows, has been steadily gaining in popularity since that time. Although it was not a viable option when the work described in this thesis began, it is included here for completeness and to demonstrate its potential for use in future virtual reality based educational applications. WebGL is a JavaScript implementation of a subset of OpenGL, and like OpenGL, is shader-based. However, unlike OpenGL, WebGL applications are generally HTML files that initiate the run-time compilation of its shader programs in real-time. This allows a WebGL application program that is located on a remote server to be rendered in any compliant web browser without a plug-in, making it truly platform independent.

Clearly, any of the five options described above are capable of producing interactive 3D visualizations, albeit through different means. However, the simple structure of VRML makes it the easiest for beginners to learn, and the most conducive to fast application development. Furthermore, the ongoing support of VRML in a wide range of proprietary software packages provides educators with the necessary flexibility to meet the specific requirements of their individual circumstances. Such factors make VRML more attractive than the other options described when designing educational applications, and as is illustrated in Figure 3.2, is mentioned more often than the other software options in IEEE educational publications and other prominent educational journals as a result.

In regards to the work related to this thesis, it was determined that developing VRML applications that could be launched inside of MATLAB would best suit the needs of the engineering students at UCC, as well as at other

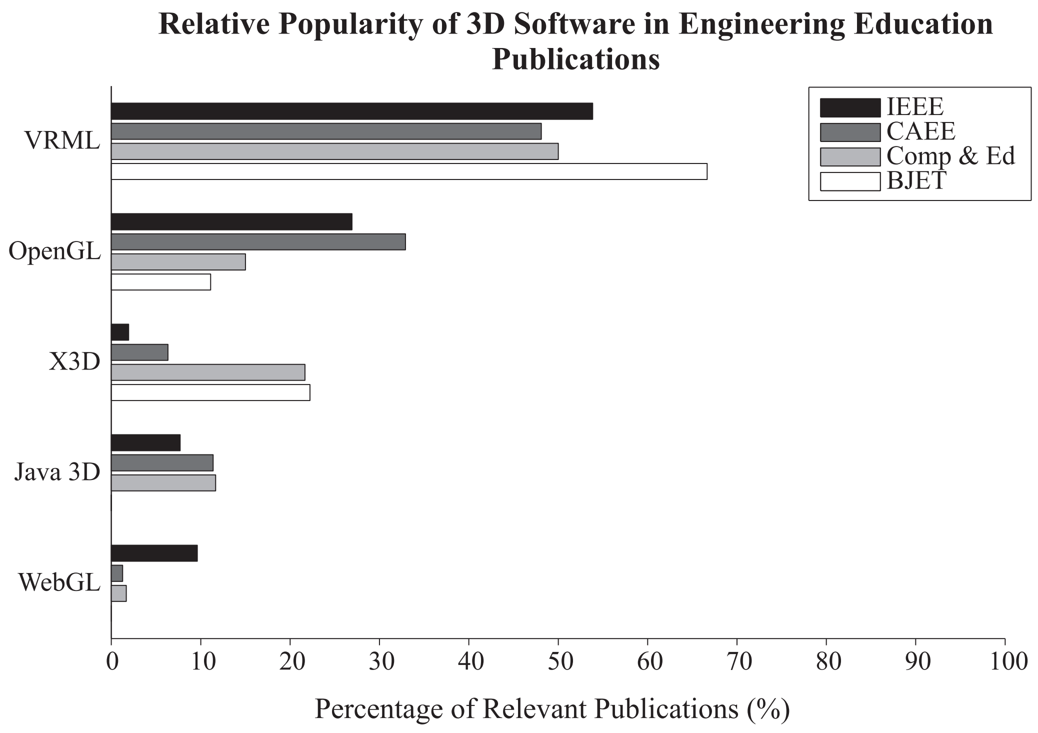


Figure 3.2 VRML is mentioned more often than the other 3D software options in IEEE educational conferences and journals, Computer Applications in Engineering Education (CAEE), Computers & Education (Comp & Ed), and the British Journal of Educational Technology (BJET).

institutions with engineering programs, largely because the students are already familiar with the MATLAB environment from use in prerequisite courses. It was thought that such an application would allow students to focus all of their cognitive effort on understanding the visualization, rather than having to learn how to use a new tool. As an additional benefit, students are able to view the uncompiled VRML and MATLAB source code associated with the educational tools, to verify the related underlying mathematical theory taught in their lectures. Motivated students can even use the source code to learn how to build their own virtual reality applications. A similar “learn by example” approach is followed in the next section, as a way of introducing the fundamental concepts of VRML that are used to create the tools described in Chapters 4 and 5 in detail.

3.1.3 VRML Technical Details

Every VRML file can be broken down into, at most, four main sections: the header, the scene graph, the prototype definitions, and the event routes. The header contains information regarding the version of VRML used in that particular file, and is the only section that must always be included. It can be assumed that all work in this thesis uses the latest version, VRML 2.0 [126]. The scene graph contains a hierarchical collection of built-in and custom nodes that define the virtual scene. Nodes are the smallest functional element in VRML, and are made up of fields and events. The values associated with these fields and events determine the properties and behaviors of each node. Prototypes define custom nodes, created using a combination of the 54 built-in nodes and any other previously defined prototypes. Prototype definitions allow the programmer to reuse node combinations effectively, without having to copy long sections of code multiple times. Finally, event routes map the output of one node to the input of another when events occur, and are used as a way of propagating change in the virtual world. Event routes can be between nodes in the scene graph, or within a prototype definition.

An example of a relatively simple scene graph is depicted in Figure 3.3, along with the resulting rendering of the virtual scene. In this example, a cylinder and cone are arranged in such a way as to form a 3D arrow. This particular scene graph has four distinct hierarchical levels and contains instances of all four major types of nodes: world state nodes, grouping nodes, child nodes, and field nodes.

The world state nodes define properties of the virtual world in which virtual objects may reside. The visual details of the background and the default angle from which a user will view the world are defined in this example, but other world state nodes include ones that contain information on the world's default navigation scheme and how objects in the world are lit.

Grouping nodes act as parent nodes, and each such node defines a relative coordinate space for its children. This coordinate space may be translated, rotated, and even scaled compared to the fixed world coordinate system, and indeed, compared to higher-level relative coordinate systems. This concept is illustrated in Figure 3.4.

Child nodes are located one level down in the hierarchy from their parent, and may themselves be parents if of the grouping node type. It should be noted, however, that a scene graph is meant to be acyclic, and infinite loops must be avoided. Perhaps the most commonly used child node is the Shape node, the fields of which define the appearance and geometry of an object. Other child nodes of note include environmental sensing nodes and pointing-device sensing

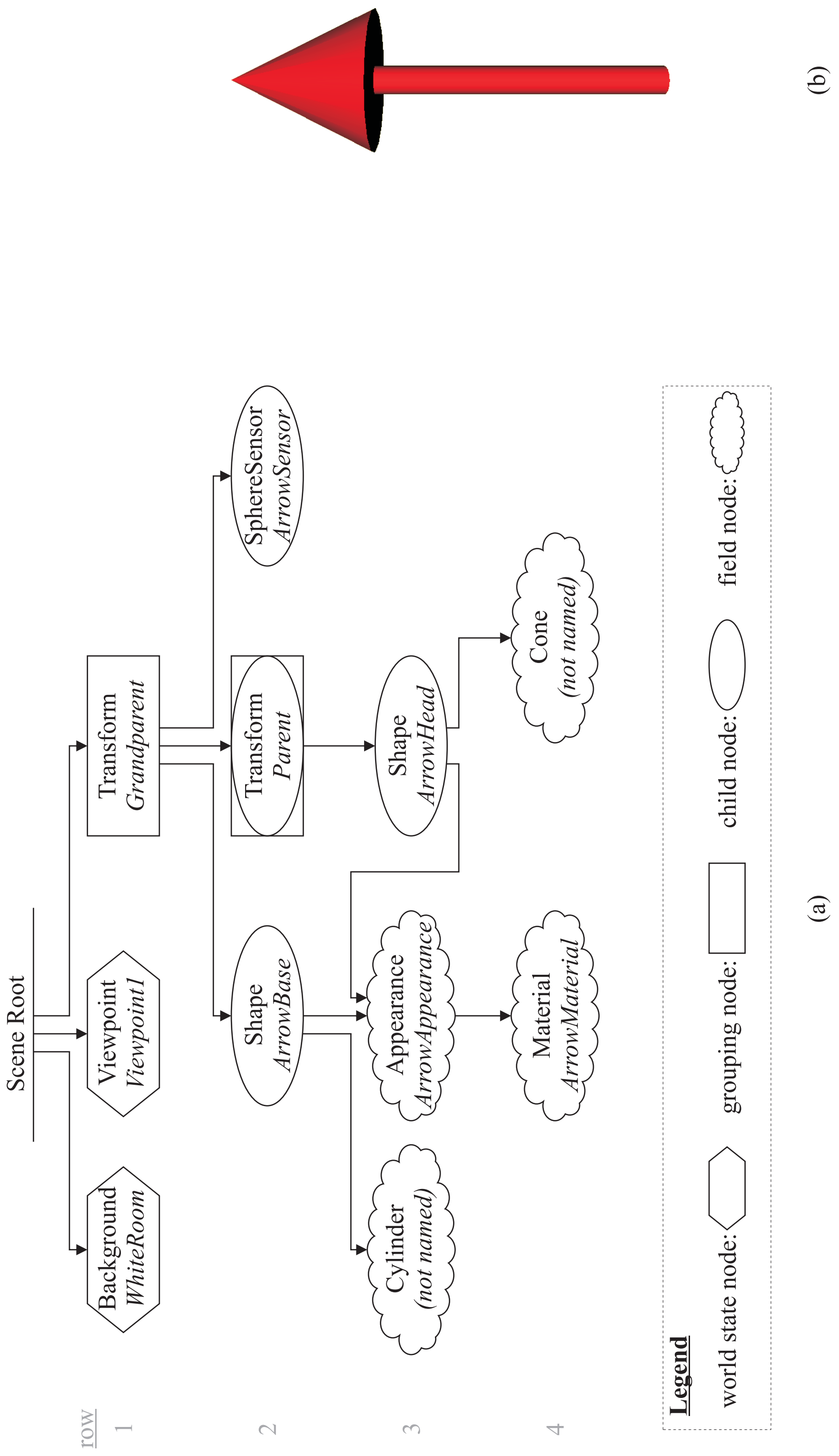


Figure 3.3 a) Scene graph and b) corresponding rendering of a 3D arrow.

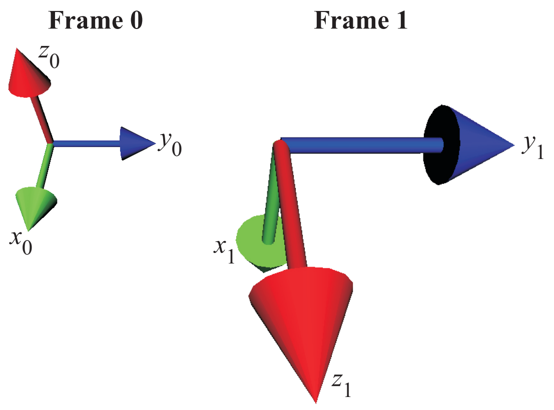


Figure 3.4 Objects in coordinate frame 1 are translated to the right of the fixed world coordinate frame (frame 0). They are also rotated by 90° about the y_0-axis, and are double the scale of what they would be in frame 0.

nodes, such as the SphereSensor included in the example scene graph, which allow a user to interact with the virtual world.

The lowest levels of the hierarchy in Figure 3.3a consist of field nodes; that is, nodes that define the properties of other nodes. For example, the geometry field of a Shape node can have a Box, Cone, Cylinder, or Sphere node as its value. These geometry node types, in turn, contain fields to describe the physical geometry of their particular shape, such as the length, width, and height dimensions of a box, or the height and radius of a cylinder. Field nodes like the Appearance node require further field nodes as values to its fields. In the case of the Appearance node, these nodes define properties associated with the material and texture of a given shape. Specifically, the Material node defines the color and opaqueness of the shape, as well as how the object reflects any light in the scene, whereas the texture node types apply images or movies onto the shape's surface.

The VRML code used to realize the 3D arrow shown in Figure 3.3 is included as Figure 3.5, with the header, scene graph, and event routing sections each identified. In this particular example, there are no prototype definitions. However, a few slight modifications could turn the entire scene graph into a prototype definition, thereby creating a custom Arrow node. Such an Arrow node was used to create the coordinate frames depicted in Figure 3.4, the specific details of which are contained in Appendix A.

When considering the code in Figure 3.5, the format of the header line should be noted. Normally, the inclusion of a # character outside of a quoted string indicates that the remainder of the line should be treated as a comment,

```
Line
 1  #VRML V2.0 utf8
 2
 3  #----------------------------------------
 4  # arrow.wrl
 5  #
 6  # Simple example that draws a 3D arrow by
 7  # combining a cylinder and cone.
 8  #----------------------------------------
 9
10  DEF WhiteRoom Background {
11    groundColor  1 1 1  #white (RGB)
12    skyColor     1 1 1  #white (RGB)
13  }
14
15  DEF Viewpoint1 Viewpoint {
16    fieldOfView 0.785398
17    orientation 0 0 1 0
18    position 0 0 10
19    description "Default View"
20  }
21
22  DEF Grandparent Transform {
23    translation 0 1 0, rotation 0 0 1 0, scale 1 1 1
24    children [
25      DEF ArrowBase Shape {
26        appearance DEF ArrowAppearance Appearance {
27          material DEF ArrowMaterial Material {
28            diffuseColor 1 0 0  #red (RGB)
29            transparency 0      #opaque
30          }
31        }
32        geometry Cylinder {radius 0.1, height 2}
33      }
34      DEF Parent Transform {
35        translation 0 1.5 0 #(ArrowBase+ArrowHead heights)/2
36        rotation 0 0 1 0    #no rotation (x, y, z, angle)
37        scale 1 1 1         #no scaling (Sx, Sy, Sz)
38        children [
39          DEF ArrowHead Shape {
40            appearance USE ArrowAppearance
41            geometry Cone {bottomRadius 0.5, height 1}
42          }
43        ]
44      }
45      DEF ArrowSensor SphereSensor{}
46    ]
47  }
48
49  ROUTE ArrowSensor.rotation_changed TO Grandparent.rotation
50  # Note that the above is functionally equivalent to:
51  # ROUTE ArrowSensor.rotation_changed TO Grandparent.set_rotation
```

Header (lines 1)

Scene Graph (lines 10–47)

Event Routing (lines 49–51)

Figure 3.5 VRML code to realize the scene graph depicted in Figure 3.3a.

and hence should be ignored by the browser when rendering the virtual world. The header line, on the other hand, conveys crucial information to the browser regarding the version of VRML and type of character encoding used, and as mentioned previously, must always be present. The # character preceding this header information is simply part of the expected information package, and indicates that this line is not directly adding nodes to the scene or creating route maps to control the behavior of said nodes.

Also of note is the use of the keywords DEF and USE in the sample code. The DEF keyword is used to assign a name to a node, allowing it to be referred to at a later point in the VRML file or by external applications. Every node type can be named in this way, although it is not necessary to name every node. For instance, the Cylinder and Cone nodes are not named in the example in Figure 3.5. The USE keyword inserts a new instantiation of the node it refers to. This is not like inserting a prototyped node into the scene graph; it is not a new copy of the node, but rather acts like a pointer to the existing node, as illustrated in Figure 3.3a. In this way, a node may have multiple parents. However, care must be taken to avoid creating a loop of node references. For example, the Parent node in Figure 3.5 cannot use USE to include the Grandparent node as one of its children. Doing so would create an infinite loop, and would create the undesired scenario of making a node its own grandparent[1].

Once a virtual world is rendered by the browser, the standard way of driving change in this virtual world is via VRML's event handling system. This applies to both time-dependant animation sequences as well as changes that result from the user interacting with the scene. VRML events can be either incoming (an eventIn) or outgoing (an eventOut). An incoming event acts as a message to a node, instructing it to change some state within that node. An outgoing event, on the other hand, is a message from a node indicating that some state within that node has changed. A route map connects an eventOut of one node to an eventIn of another node, allowing for the propagation of events between nodes. Custom connections can be added between events by using the ROUTE keyword. For example, in Figure 3.5 the ArrowSensor detects the dragging motion of a user's mouse and interprets this motion as a 3D rotation. This rotation information is sent to the Grandparent node, where the coordinate space for the entire 3D arrow is rotated, resulting in the visual rotation of the arrow on the screen. In this way, the user is able to effectively control the orientation of

[1]This is unlike the types of convoluted human relationships that are possible, as identified in the song, "I'm my own grandpa," written by Dwight Latham and Moe Jaffe in 1947.

the 3D arrow. Notice that in this example the rotation field of the Grandparent Transform node is an exposedField. Unlike the normal fields, exposedFields have an eventIn ("set_*fieldname*") and an eventOut ("*fieldname*_changed") implicitly associated with them, and the correct event will automatically be applied depending on the context of how the exposedField is used in the route map. This means that the value of an exposedField can be changed via events, whereas normal field values cannot.

The ability to create custom route maps is powerful, but it does have limitations. Routes can only be established between events that are of the same type (i.e. both the eventIn and eventOut must be boolean, integer, float, 2D vector, 3D vector, rotation, color, time, string, or image types), and the changes effected rely on the existing behavior of the built-in VRML nodes. This severely restricts the type of logic and control that can be incorporated into the virtual scene. To circumvent this limitation, Script nodes can be included in the scene graph. A Script node contains programmer-defined fields, eventIns, and eventOuts. It cannot, however, contain programmer-defined exposedFields, meaning that all events must be defined and controlled explicitly within the node. A Script node also contains custom code (i.e. the script) that determines the behavior of the node. This script can be written in any language that the browser can interpret and execute, with JavaScript being one of the more popular choices. In this way, the full functionality of the chosen programming language is available to use within VRML. Just as with other VRML nodes, the eventOuts of a Script node can be routed to the eventIns of other nodes in the scene graph as a way to propagate change throughout the virtual world.

A second way of adding custom logic and control algorithms to a virtual scene is to connect it to an external application via the External Authoring Interface (EAI) [132]. Whereas the Script node enables the inclusion of custom programming within the scene graph, the EAI allows for the addition of custom logic and control from outside of the VRML file. This is beneficial, for instance, when providing users with custom graphical user interfaces (GUIs) as a way of interacting with the virtual world. It also allows for a wider range of programming languages to be used, because the browser does not have to interpret and execute the code directly. Instead, the EAI provides a means for the external application to get information from, and send information to, the virtual world through the browser. Common languages used with the EAI include Java and C/C++. External applications can use the EAI to access named nodes and the events of these named nodes, enabling them to drive change in the virtual world through VRML's event handling system. In this way, external applications are able to effectively modify the values of exposedFields. However, just as is the case with route maps and Script nodes, the EAI does not allow the values of normal fields to be changed, the repercussions of which are explored further in Chapter 5.

3.2 WIMU

One of the main pieces of hardware used in the second part of this thesis is the third generation WIMU developed by the Tyndall National Institute in Cork, Ireland [133]. Generally speaking, inertial measurement units (IMUs) are devices used to measure an object's motion; more specifically, they measure an object's relative change in position and orientation. They typically employ accelerometers and gyroscopes to measure this relative motion, although the inclusion of other sensors (for example, magnetometers) to provide additional data is becoming more prevalent.

Mechanical gimbaled IMUs were the first IMUs to be developed, and were used in precision-sensitive projects such as the original strategic missile guidance systems [134] and as part of the Apollo spacecrafts' inertial navigation system [135]. Such configurations are still used in expensive, marine-grade inertial navigation systems for applications where accuracy is of paramount importance. However, advances in microelectromechanical systems (MEMS) technology in recent years has resulted in dramatic improvements in the accuracy of electronic strapdown IMUs, and they now offer a feasible alternative to the gimbaled configuration. The ability to manufacture MEMS-based IMUs at a fraction of the size and price of their gimbaled counterparts has resulted in an explosion of new applications using inertial sensors. From airbag deployment, to interactive video game control, to determining the screen orientation of smartphones, inertial sensors have become highly integrated into the average person's everyday life.

Tyndall's WIMU, shown in Figure 3.6, is an example of a miniature strapdown IMU, and is based on their modular 25 mm mote platform [136]. It transmits data from its triaxial accelerometer, its single and dual-axis gyroscopes, and its triaxial magnetometer via a wireless protocol to a base station connected to a nearby computer. Its wireless communication abilities and compact size allow it to be used in a wide variety of environments and appli-

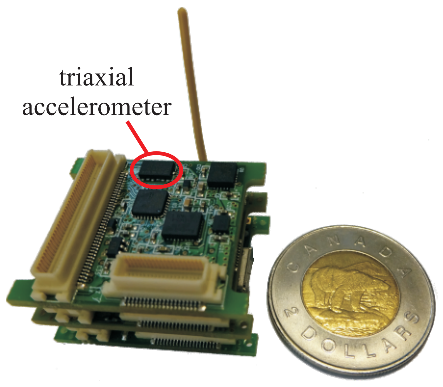


Figure 3.6 WIMU provided by the Tyndall National Institute, Cork, Ireland.

cations, including ocean wave monitoring [137], health monitoring [138, 139], sports analysis and training [140–142], and gesture recognition for gaming and multimedia control [143, 144].

The work described in this thesis made use of the accelerometer on Tyndall's WIMU. In order to understand the work in Chapter 6, the general theory of how accelerometers operate is explained in the following subsection.

3.2.1 Accelerometer

Tyndall's WIMU uses Analog Devices' ADXL345 triaxial accelerometer [145] to measure linear acceleration along three mutually orthogonal axes: the WIMU's x-axis, y-axis, and z-axis. This triaxial accelerometer consists of a polysilicon micromachined mass that is suspended above a silicon wafer with polysilicon springs. Although the actual design and fabrication of such a structure involve proprietary, state-of-the-art MEMS techniques that continue to be developed and refined, each sensing axis can be modeled as a simple mass-spring system, as shown in Figure 3.7, and the triaxial accelerometer can be thought of as three such mass-spring systems, arranged so that they are mutually orthogonal to one another.

As with all objects undergoing acceleration, Newton's second law of motion applies to the proof mass attached to the end of the spring in this mass-spring system, so that the acceleration, a, of the proof mass is proportional to the net force, F, acting on it, and is inversely proportional to its mass, m:

$$F = ma. \tag{3.1}$$

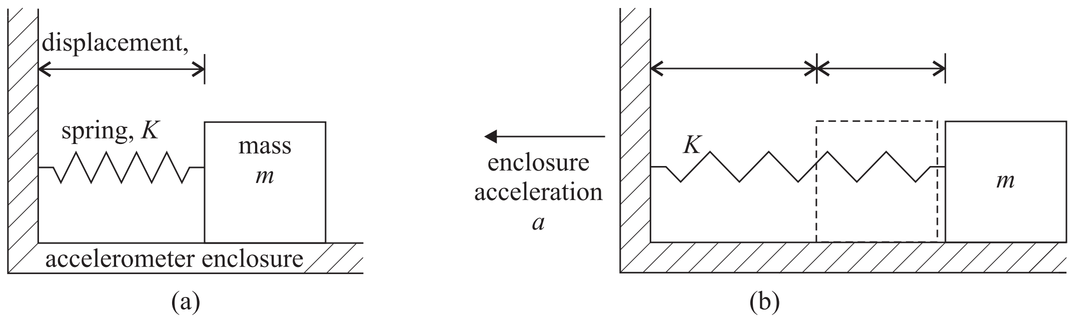


Figure 3.7 A mass-spring system representing a single-axis accelerometer a) at rest, and b) undergoing linear acceleration in the direction of the accelerometer's axis of measurement.

Hooke's law also applies to this mass-spring system, so that when the system is accelerated, the resultant force on the proof mass acts to compress or extend the spring – depending on the direction of acceleration – by a distance, $\Delta\delta$. Assuming that the spring deformation is relatively small and does not exceed the elastic limits of the spring, this distance is linearly proportional to the force, so that

$$F = K\Delta\delta, \tag{3.2}$$

where K is the stiffness of the spring under consideration. Thus, the acceleration of the proof mass can be found by measuring its change in displacement:

$$a = \frac{K}{m}\Delta\delta. \tag{3.3}$$

Numerous methods can be used to measure this displacement change in MEMS applications, including the use of piezoelectric components, piezoresistive components, or capacitive components. The ADXL345 triaxial accelerometer uses differential capacitive sensing.

When interpreting the data from an accelerometer, it is important to keep in mind that, for earth-bound applications, gravity will act on the proof mass at all times. The acceleration caused by gravity is known as static acceleration because it is present even when the accelerometer is at rest. The effects of this static acceleration will have to be subtracted from the accelerometer measurements to recover any dynamic acceleration components. In applications where the accelerometer is moved with constant velocity, the static acceleration caused by gravity will be the only component contributing to the magnitudes measured on each of the accelerometer's axes. These magnitudes will vary as

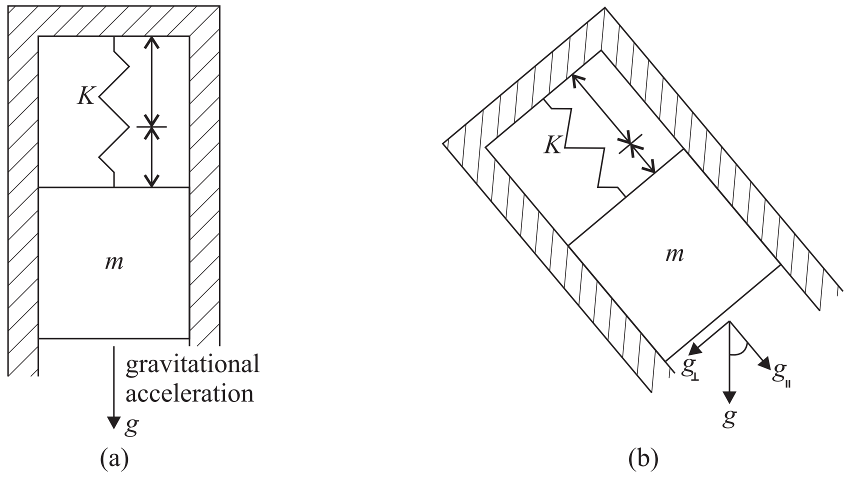


Figure 3.8 Static acceleration when the accelerometer is a) aligned with gravity, and b) rotated about an axis pointing out of the plane of the page.

the accelerometer is rotated, as illustrated in Figure 3.8, which can be used to determine the sensor's orientation.

3.3 SCARA

Before the WIMU can be used to measure arbitrary motion, it must first be calibrated according to a set of known reference positions and movements. As will be discussed further in Chapter 6, a robot arm is an ideal platform on which to calibrate the WIMU, due to the flexibility in the 3D paths that its end-effector can traverse, and because it can be programmed to move its end-effector at specified velocities and accelerations over these 3D paths. Furthermore, a robot's motion is repeatable, ensuring consistency between experimental trials. The relevant details of the hardware and software components of the SCARA system used to execute the calibration routine detailed in this thesis are described in this section.

3.3.1 SCARA Hardware

As illustrated in Figure 3.9, the SCARA system used for the work in this thesis consists of four main hardware components: a host computer, the Sankyo SC3150 controller, the Sankyo teaching pendant, and of course, the robot arm itself – a Sankyo SR8408 SCARA.

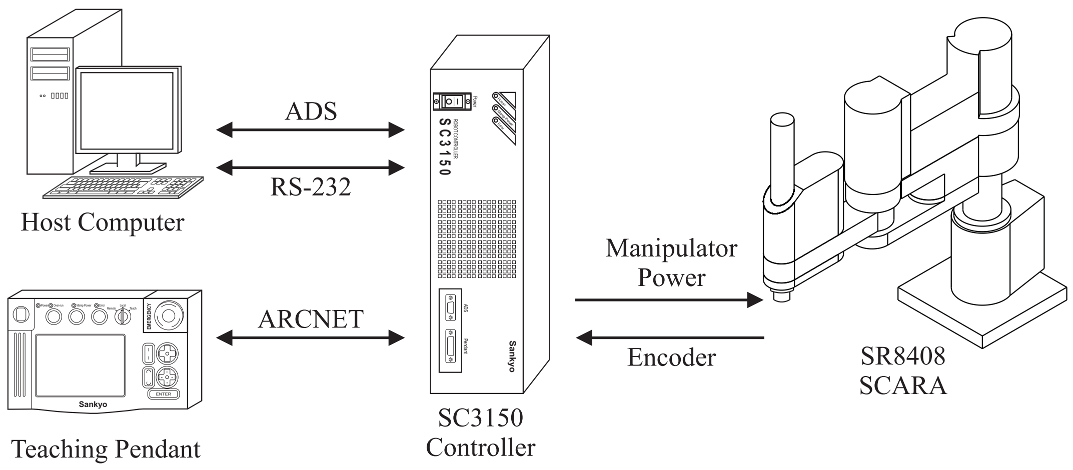


Figure 3.9 An overview of the SCARA system in UCC's mechatronics laboratory.

Host Computer

The host computer is a Dell personal computer running Windows 7 Enterprise, with a 3 GHz Intel Core 2 Duo processor and 4 GB of RAM. This computer is used to create and debug robot programs, as well as to post-process experimental data.

SC3150 Controller

The controller [146] functions as the brain of the SCARA system, driving and monitoring the robot's motion according to the programmed task that is currently running. By interpreting the encoder data associated with the four joints of the SCARA in conjunction with the stored homing position data [147], the controller can periodically determine the robot's pose, and adjust the drive signals to the joint servomotors to achieve the desired overall motion of the robot.

Teaching Pendant

The teaching pendant [148] is a handheld device, intended to allow a human operator to monitor and control certain SCARA actions without having to access the host computer. The main purpose of the teaching pendant is to allow an operator to manually determine and save a set of robot poses, thereby "teaching" a robot where to place its end-effector. The Sankyo SCARA can be taught in three different ways: directly, remotely, or through a Manual Data Input (MDI) process. Direct teaching involves turning off the power to the robot's servomotors and physically moving the robot's links to the desired positions. Similar hands-on teaching methods have been popular since the earliest Unimate [41], and have been described as being akin to taking the robot by the hand and leading it. Remote teaching requires the robot's servomotors to be powered on while the operator uses jog buttons on the pendant to move the robot, much as one would use buttons on a remote control when directing the motion of a character in a video game, or when driving a remote control car. Finally, MDI teaching offers a more quantitative approach by enabling the operator to enter or modify numerical sets of position data using a number pad on the pendant's touchscreen. Of course, the accuracy that can be achieved in positioning the end-effector using this final approach depends on the accuracy of the underlying kinematic model.

SR8408 SCARA

As described in Chapter 2, a SCARA's joints may be configured in a variety of ways so as to yield the robot's characteristic compliance in the horizontal plane. The Sankyo SR8408 SCARA – illustrated in Figure 3.10 – is of the revolute base joint type of configuration, with a prismatic third joint. According to the manufacturer's specifications, this robot has a total arm length of 550 mm in the x-y plane, and can move its joints according to the ranges summarized in Table 3.1. These ranges of motion result in the reachable workspace shown in Figure 3.11.

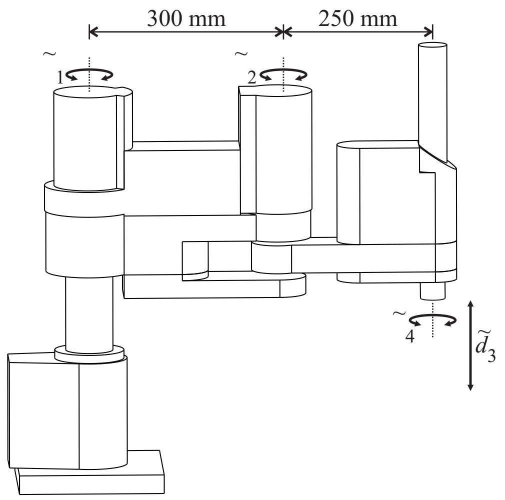


Figure 3.10 Side view of the SR8408 SCARA, with dimensions and axes of motion shown.

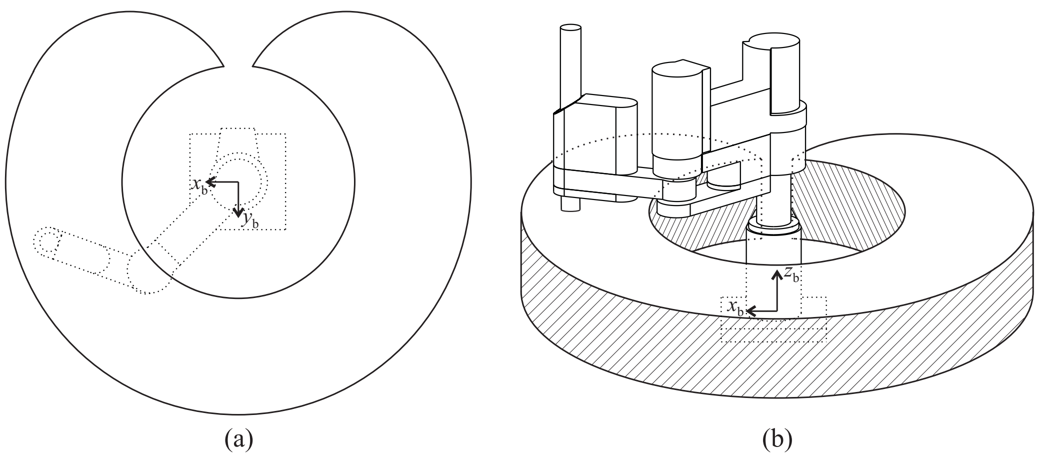


Figure 3.11 a) Top view of SR8408 SCARA's workspace in the x-y plane, and b) 3D view of the workspace, with the base frame of the SCARA shown.

Each of the SCARA's four joints has a dedicated servomotor associated with it to move that joint, employing harmonic drives and belt-pulley systems to transfer the servomotors' motion, as required. Each servomotor is equipped with a magnetic absolute encoder disk to provide information regarding the position of the servomotor, and by extension, the associated joint. When interpreted by the controller, the data from each encoder can be used to identify 8192 unique positions within a complete motor revolution [149]. Thus, the position of the revolute joints can be determined with a resolution of better than 0.05°, and the position of the prismatic joint can be determined with a resolution of better than 0.02 mm.

The resolution of each joint limits the overall repeatability and accuracy with which a robot can place its end-effector. As illustrated in Figure 3.12, repeatability and accuracy, although often incorrectly used interchangeably, refer to two different measurements [150]. A robot's repeatability measurement refers to the variation in the pose of the end-effector when the robot is programmed to move to the same point multiple times. In addition to joint

Table 3.1 Operational Range of the SR8408 SCARA

Joint	Minimum	Maximum	Resolution
$\tilde{\theta}_1$	-120°	120°	0.05°
$\tilde{\theta}_2$	-120°	120°	0.05°
$\tilde{d}_3$	0 mm	150 mm	0.02 mm
$\tilde{\theta}_4$	-360°	360°	0.05°

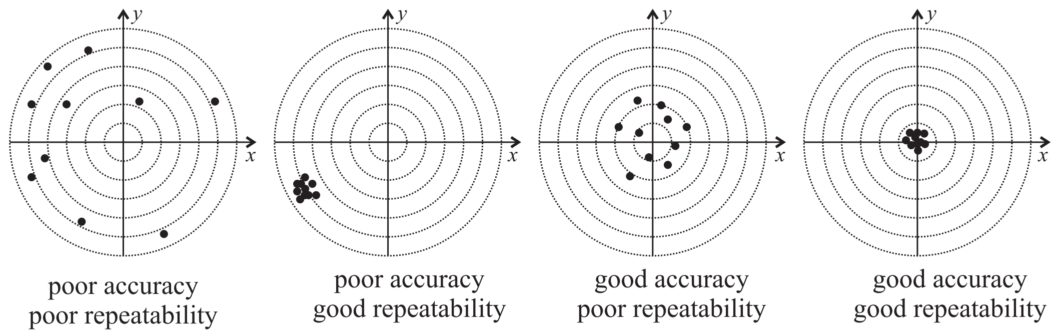


Figure 3.12 Accuracy and repeatability when a robot moves its end-effector to the origin of the x-y plane ten times.

resolution, factors such as joint and link compliance can have a significant effect on a robot's repeatability. On the other hand, the accuracy of a robot refers to how close it can move its end-effector to a specified pose, as measured relative to a known world frame. The absolute accuracy of a robot depends on a precise homing procedure [147], as well as correctly modeled kinematic parameters.

WIMU Mounting Platform

A custom mounting platform was built to allow for the secure and simple attachment of the WIMU to the SCARA's end-effector while carrying out the work described in Chapter 6.

This mounting platform, pictured in Figure 3.13, was designed so that the inertial sensor circuit board of the WIMU could be secured to the platform's base in a way that would not interfere with the antenna of the WIMU. The mounting platform has open sides to prevent the battery and WIMU from overheating, and also to facilitate easy access to the WIMU when recharging its battery. Although not strictly necessary, the sides of the platform can also be adjusted in order to level the WIMU after it has been attached to the SCARA. Most importantly, the mounting platform attaches the WIMU to the SCARA in such as way as not to interfere with the motion of the SCARA itself – all four joints of the SCARA are free to move through the entirety of their operational ranges.

3.3.2 SCARA Software

A single SCARA application makes use of multiple software components to coordinate the many aspects of the SCARA system. Aside from the system-

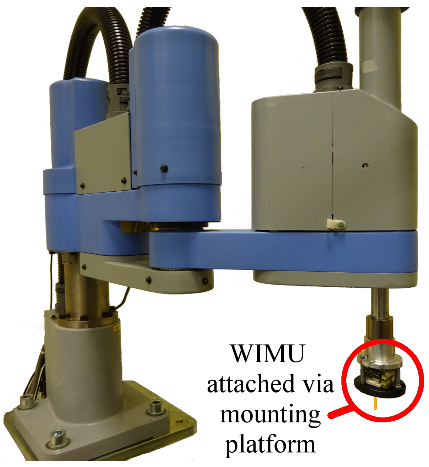


Figure 3.13 The WIMU is securely attached to the end-effector of the SCARA via a custom-built mounting platform. Notice that the antenna of the WIMU is pointing downward.

level software that always runs on the host computer, controller, and teaching pendant, custom code must also be developed for each unique application. This custom code relates both to the tasks executed by the controller, and the programs running on the host computer to establish communication with the robot.

Establishing Communication

The source files used to define robot tasks are programmed using the proprietary Sankyo Structured Language/Enhanced (SSL/E) [151], a high-level programming language that provides access to specialized built-in functions related to the motion of the robot. SSL/E also has built-in functions that allow the SCARA controller to communicate over its serial port using the RS-232 protocol. These functions can be incorporated into custom robot tasks that listen for real-time commands sent by the host computer via RS-232 communication, which the robot arm then carries out [152]. Conversely, a custom program can be created so that real-time data from the robot, such as that pertaining to the robot's position at a given time, may be communicated and saved to the host computer via the RS-232 connection for later analysis. A simplified flowchart showing the steps required in such a program is provided in Figure 3.14. In this program, the controller sends information regarding the robot's current position, as well as a sign-off message to indicate the end of transmission.

Meanwhile, the host computer runs a custom MATLAB function, *saveRobot-Data.m*, to receive, check, and save the robot position data that is transmitted by the SCARA controller. Because MATLAB, like SSL/E, comes with built-in serial port communication functions, it is straightforward to configure the host computer's serial port to match that established by the controller. As illustrated in Figure 3.15, it is imperative that *saveRobotData.m* starts executing before the controller begins its task, to ensure that no messages are lost.

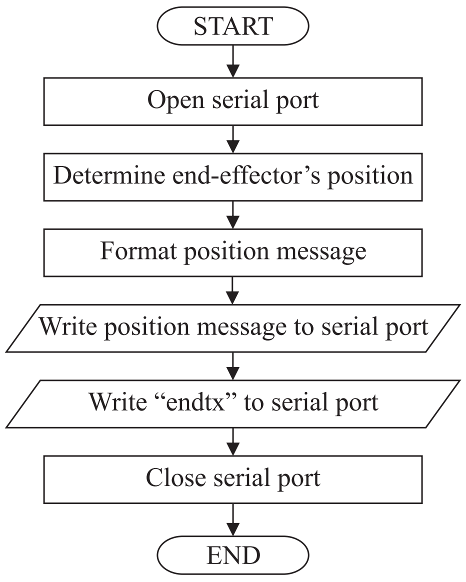

START
Open serial port
Determine end-effector's position
Format position message
Write position message to serial port
Write "endtx" to serial port
Close serial port
END

Figure 3.14 Flowchart demonstrating the transmission of robot position data by the controller.

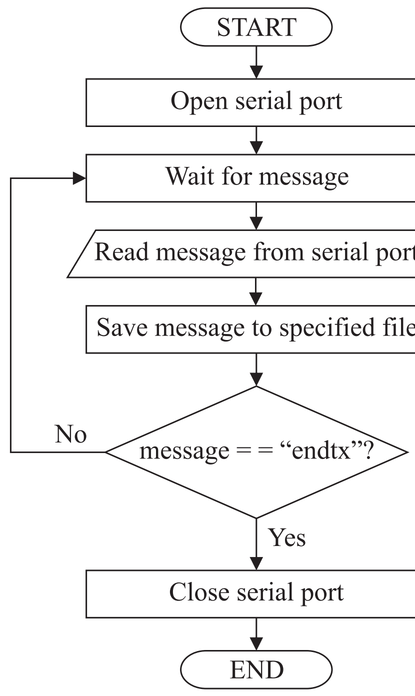


Figure 3.15 Flowchart demonstrating the receipt and saving of robot position data on the host computer.

Chapter 4

Visualizing Compound Rotations

Mental rotations are among the most difficult of all spatial tasks to perform, and even those with high levels of spatial ability can struggle to visualize the result of compound rotations. Yet visualizing and understanding the principles of compound rotations are critical in understanding robot kinematics. As discussed in the previous chapter, VRML and MATLAB can be used together to create virtual reality based educational tools to aid in the visualization of difficult concepts. The Rotation Tool, created by the author, is such a tool that aids in the visualization of compound rotations. The objective of this chapter is to investigate the effectiveness of the Rotation Tool in assisting engineering students with the visualization of fixed and mobile frame rotations, and in the process, providing those students with a better understanding of the fundamental principles of compound rotations. The chapter begins by identifying the need for such a virtual reality based tool, followed by a review of comparable software-based solutions. A detailed description of the Rotation Tool's GUI and underlying algorithms is provided in the third section. The methodology used to study the tool's effectiveness during the 2012/13 academic year is then outlined, followed by a comprehensive analysis of the most relevant results.

4.1 Why Use Virtual Reality?

In mechanically-complex engineering courses, such as those specializing in robotics and mechatronics, it is imperative that students be able to accurately visualize 3D objects, both as a sum of interconnected parts and as a whole. Moreover, students must also be able to visualize the motion of these objects in 3D space. These tasks require students to conduct complex geometric transformations consisting of translations and compound rotations, as described in Chapter 1. Although students typically demonstrate little difficulty in visualizing the effect of linear translations, it has been observed that some have more difficulty visualizing the effect of compound rotations. In fact, of the three categories of spatial ability identified by Linn and Peterson, students have repeatedly demonstrated the largest discrepancy in their ability to perform mental rotations [153], with rotations about multiple axes among the most difficult types of rotations to visualize [154].

Traditionally, isometric drawings have been used as compound rotation visualization aids, depicting the starting, intermediate, and final orientations of a rotated object. They are often used in lecture notes and textbooks to demonstrate the non-commutative nature of compound rotations, as well as the difference between rotations made about a fixed frame of orthonormal axes and those made about a mobile frame that rotates with the object. Such static drawings encourage an analytic approach to breaking down compound rotations [155], but assume that students can envision the correct motion associated with each step. In addition, examples must be chosen carefully to avoid the interpretation of the isometric views as 2D patterns instead of the intended 3D representation [156].

Animated presentations can depict the action of the rotation more clearly than static images, but still make students passive observers rather than active participants. Sketching employs a more active approach and can be beneficial in developing weak spatial abilities [157], but generally involves drawing isometric views on 2D media that instructors must manually verify. Rotating physical manipulatives allows for a true visualization in 3D space, but has limited accuracy when demonstrating rotation angles that are not integer multiples of 90°. It has also been argued that students rotate physical manipulatives instinctively and too quickly to encourage proper examination of the motion [158]. Instead, a software visualization aid that places limitations on the rotations is advocated, to encourage students to predict the outcome of the motion and force a deeper analysis. Virtual reality, with its interactive capabilities and its accurate 3D depictions, can be incorporated into a rotation visualization tool that does just that. Such a software tool will also be able to automatically verify results, providing students with immediate feedback and the ability to use the tool outside of the classroom.

4.2 Comparable Software-Based Solutions

The importance of strong spatial skills – including mental rotation skills – in the areas of science, technology, engineering, and mathematics has long been argued [159], garnering the attention of engineering educators hoping to improve student performance and retention with targeted spatial training [160]. Sorby and her colleagues have conducted extensive research in the field [160, 161], developing and using multimedia software and a workbook [162] to train students in multiple aspects of spatial ability. Visualizing single and multiple rotations about a fixed frame are among the topics considered, and the non-commutative nature of compound rotations is emphasized. The modules developed by Sorby and Wysocki, along with a separate spatial assessment and training website [163], have been successfully incorporated into training courses at multiple institutions [164]. Sorby [165] cautions against using these modules with an audience of mixed spatial abilities, however, as they are intended to target those with weak spatial skills.

Other examples of targeted mental rotation training include the exercises on single rotations developed for handheld touch screen devices by Martin-Dorta et al. [166]. Rafi and Samsudin [167] consider compound rotations with their interactive Desktop Mental Rotation Trainer (iDeMRT), prompting students to choose the set of consecutive rotations that will result in the overall change in orientation displayed. The interactive version of iDeMRT was found to be more effective than the animated version [168], demonstrating the importance of active participation. Display Object [169] and the Physical Model Rotator [170] pair the rotation of concrete objects with rotating computer images, allowing students to associate the motion seen on the screen with the corresponding real-world motion.

C.-Y. Wang et al. [171] use stereoscopic technology to help students visualize 3D rotations, finding that incorporating interactive control generates higher levels of student enthusiasm than more passive displays. Price and Lee [172] also use stereoscopic technology to display rotations in 3D, finding that interpreting such displays increases the cognitive load of students, compared to the visualization of rotations displayed on paper. Such findings underscore the importance of using new technology effectively, rather than relying on the technology itself to be the answer.

Desktop virtual reality tools capable of demonstrating important compound rotation principles include VRMath, a virtual learning environment that Yeh [173] uses to demonstrate the non-commutative nature of compound rotations to disbelieving primary school students. Manseur includes two virtual reality based tools with his textbook [174] that demonstrate the motion of fixed and mobile frame rotations separately, each about a different set of non-configurable rotation axes. Bruder and Wedeward's [155] virtual reality rotation tool, on the other hand, compares the fixed and mobile frame rotations that result from the same set of three user-defined rotation angles and axes.

4.3 Rotation Tool

In order to break down the yaw-pitch-roll motion of a robot wrist, or keep track of the effects of joint angles and link twists when considering the kinematics of a robot arm, students must have more than strong visualization skills. They must also possess a solid understanding of the fundamental principles of compound rotations. The Rotation Tool, described below, allows students to explore these fundamental principles before having to apply them to such mechanically-complex examples.

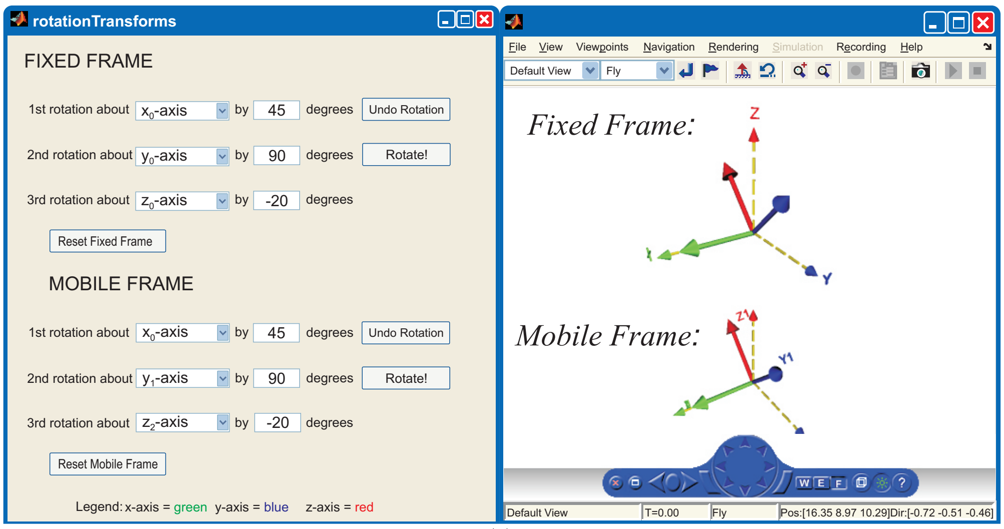
4.3.1 Tool Description

The Rotation Tool is an interactive educational tool that consists of a custom MATLAB GUI and a virtual reality window, as shown in Figure 4.1. Students use the GUI to enter up to three consecutive rotations about the fixed and/or mobile reference frames, with complete freedom in their choice of rotation angles and axes. This freedom allows students to vary the order in which sets of rotations are executed, and thereby confirm the non-commutative nature of compound rotations. This is illustrated by comparing the resultant orientations for the mobile frame in Figures 4.1c and 4.2c. The GUI is designed to also provide separate control of rotations made about the fixed and mobile frames. This allows students to explore the conditions that lead to different and matching final orientations when making the two types of rotations, as exemplified in Figures 4.1b and 4.2c, thereby verifying the relationship between the two types of compound rotations.

The 3D animation of each rotation in the virtual world is initiated by pressing the "Rotate!" buttons on the GUI, giving students complete, real-time control of the visualization. Undo options are also available to allow students to step back and forth between orientations, as needed, to gain a clear understanding of the rotation action. To preserve the order of the rotations, only the relevant rotation and undo buttons are enabled at any time. This built-in software constraint also serves as a way to encourage deeper analysis and understanding of the effect of each rotation on the overall orientation. Each frame can be reset at any time, and a new set of rotations can be entered and visualized.

Every effort has been made to ensure that the Rotation Tool is a user-friendly and engaging learning tool. It was designed with features intended to offset the increased cognitive load demands that virtual reality based tools can place on students, and it incorporates feedback from students who used the tool in years prior to this study. For example, visual cues normally present in the virtual world's background (removed in Figures 4.1 and 4.2 for clearer printing) allow students to navigate the virtual world with ease, enabling them to change their viewpoint when occlusion occurs and gain a real sense of the 3D space being represented. The inclusion of both rotation frame types in the same virtual window guarantees that they are viewed from the same perspective, ensuring an accurate comparison of resulting orientations. Simple 3D graphics are used to represent the rotating frames, placing the focus on the rotations themselves, rather than on the visualization of geometrically-complex rotation objects. Similarly, a simple color scheme and clear axis labels ensure that minimal mental effort is required to correctly interpret the scene. Finally, smooth, animated transitions between initial and final orientations give students an unambiguous sense of the motion associated with each rotation.

The most important features of the Rotation Tool, from a usability perspective, are summarized in Table 4.1. Compared to similar desktop virtual reality tools, the Rotation Tool offers clear advantages in its ability to demonstrate *all* of the fundamental principles of compound rotations. That is, the Rotation Tool allows users to verify the non-commutative nature of compound rotations, and also the difference between fixed and mobile frame rotations. Unlike any


rotationTransforms

FIXED FRAME

1st rotation about x_0-axis by 45 degrees Undo Rotation

2nd rotation about y_0-axis by 90 degrees Rotate!

3rd rotation about z_0-axis by -20 degrees

Reset Fixed Frame

MOBILE FRAME

1st rotation about x_0-axis by 45 degrees Undo Rotation

2nd rotation about y_1-axis by 90 degrees Rotate!

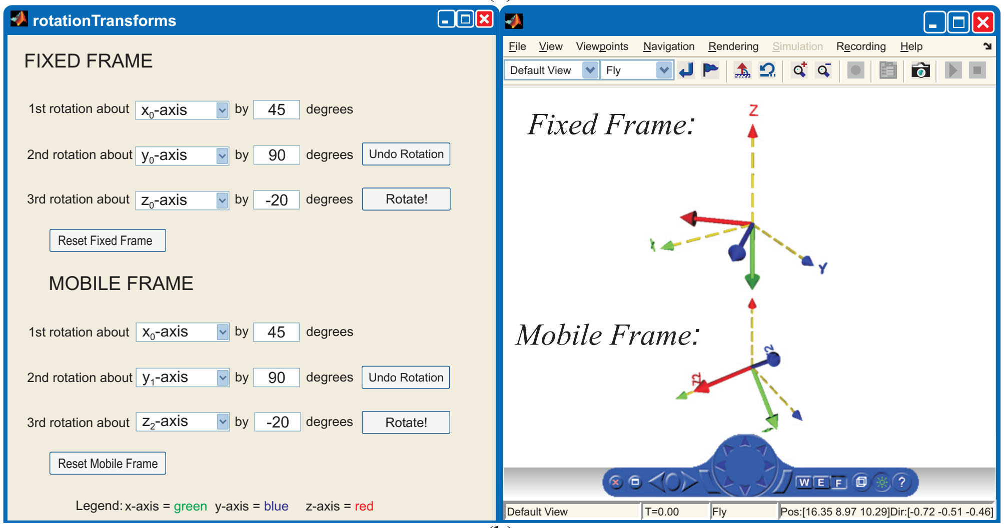
3rd rotation about z_2-axis by -20 degrees

Reset Mobile Frame

Legend: x-axis = green y-axis = blue z-axis = red

File View Viewpoints Navigation Rendering Simulation Recording Help
Default View
Fly
Fixed Frame:
Z
Y
Mobile Frame:
Z1
Y1
Default View
T=0.00
Fly
Pos:[16.35 8.97 10.29]Dir:[-0.72 -0.51 -0.46]

(a)


rotationTransforms

FIXED FRAME

1st rotation about x_0-axis by 45 degrees

2nd rotation about y_0-axis by 90 degrees Undo Rotation

3rd rotation about z_0-axis by -20 degrees Rotate!

Reset Fixed Frame

MOBILE FRAME

1st rotation about x_0-axis by 45 degrees

2nd rotation about y_1-axis by 90 degrees Undo Rotation

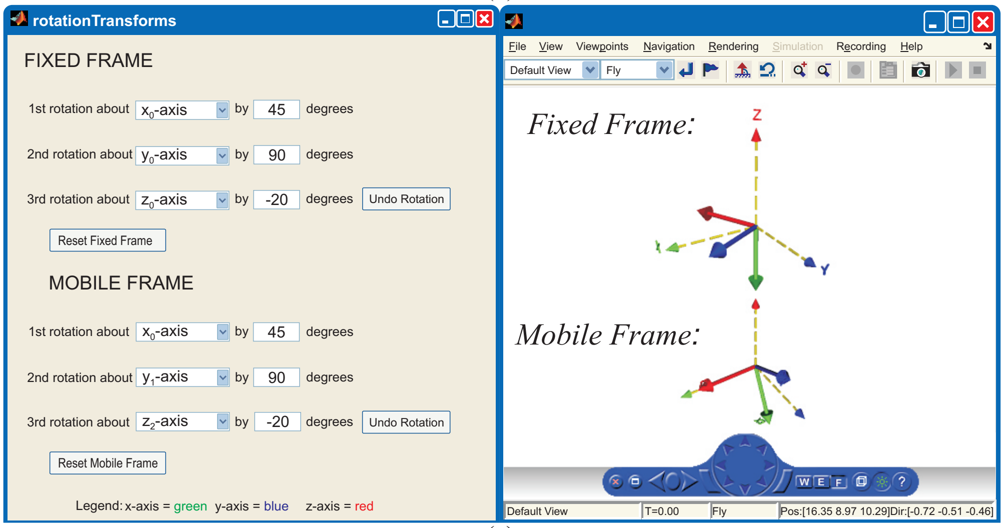
3rd rotation about z_2-axis by -20 degrees Rotate!

Reset Mobile Frame

Legend: x-axis = green y-axis = blue z-axis = red

File View Viewpoints Navigation Rendering Simulation Recording Help
Default View
Fly
Fixed Frame:
Z
Y
Mobile Frame:
Z2
Y2
Default View
T=0.00
Fly
Pos:[16.35 8.97 10.29]Dir:[-0.72 -0.51 -0.46]

(b)


rotationTransforms

FIXED FRAME

1st rotation about x_0-axis by 45 degrees

2nd rotation about y_0-axis by 90 degrees

3rd rotation about z_0-axis by -20 degrees Undo Rotation

Reset Fixed Frame

MOBILE FRAME

1st rotation about x_0-axis by 45 degrees

2nd rotation about y_1-axis by 90 degrees

3rd rotation about z_2-axis by -20 degrees Undo Rotation

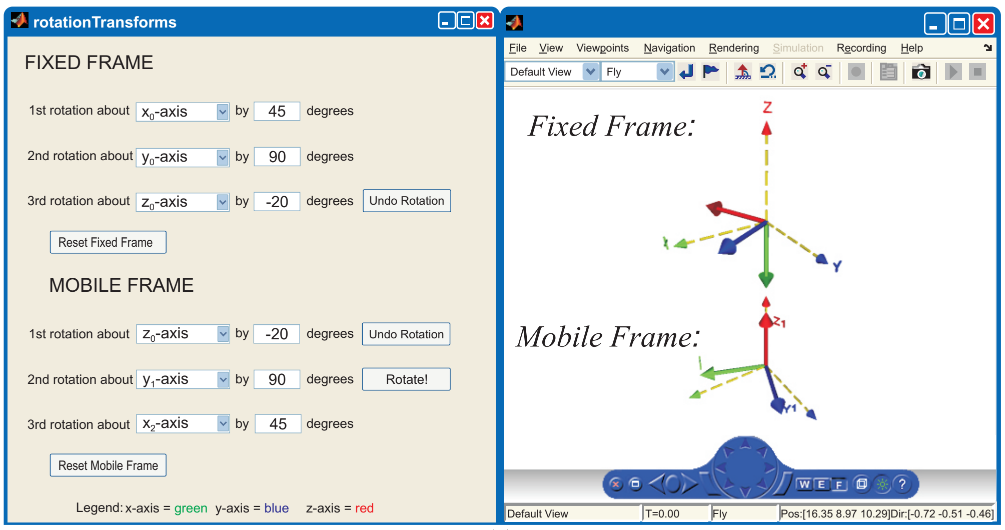
Reset Mobile Frame

Legend: x-axis = green y-axis = blue z-axis = red

File View Viewpoints Navigation Rendering Simulation Recording Help
Default View
Fly
Fixed Frame:
Z
Y
Mobile Frame:
Default View
T=0.00
Fly
Pos:[16.35 8.97 10.29]Dir:[-0.72 -0.51 -0.46]

(c)

Figure 4.1 Rotation tool: visualizing the progression of the same three rotations made about the fixed and mobile frames. Notice that the final orientations in (c) are different for the two frame types.



rotationTransforms

FIXED FRAME

1st rotation about x_0-axis by 45 degrees

2nd rotation about y_0-axis by 90 degrees

3rd rotation about z_0-axis by -20 degrees Undo Rotation

Reset Fixed Frame

MOBILE FRAME

1st rotation about z_0-axis by -20 degrees Undo Rotation

2nd rotation about y_1-axis by 90 degrees Rotate!

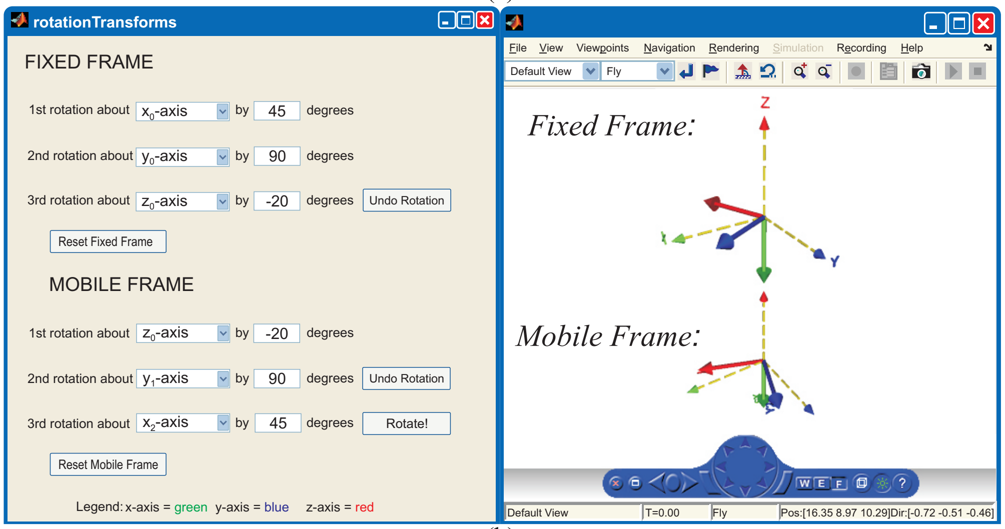
3rd rotation about x_2-axis by 45 degrees

Reset Mobile Frame

Legend: x-axis = green y-axis = blue z-axis = red

File View Viewpoints Navigation Rendering Simulation Recording Help
Default View
Fly
Fixed Frame:
Z
Y
Mobile Frame:
Z1
Y1
W E F
Default View T=0.00 Fly Pos:[16.35 8.97 10.29]Dir:[-0.72 -0.51 -0.46]

(a)



rotationTransforms

FIXED FRAME

1st rotation about x_0-axis by 45 degrees

2nd rotation about y_0-axis by 90 degrees

3rd rotation about z_0-axis by -20 degrees Undo Rotation

Reset Fixed Frame

MOBILE FRAME

1st rotation about z_0-axis by -20 degrees

2nd rotation about y_1-axis by 90 degrees Undo Rotation

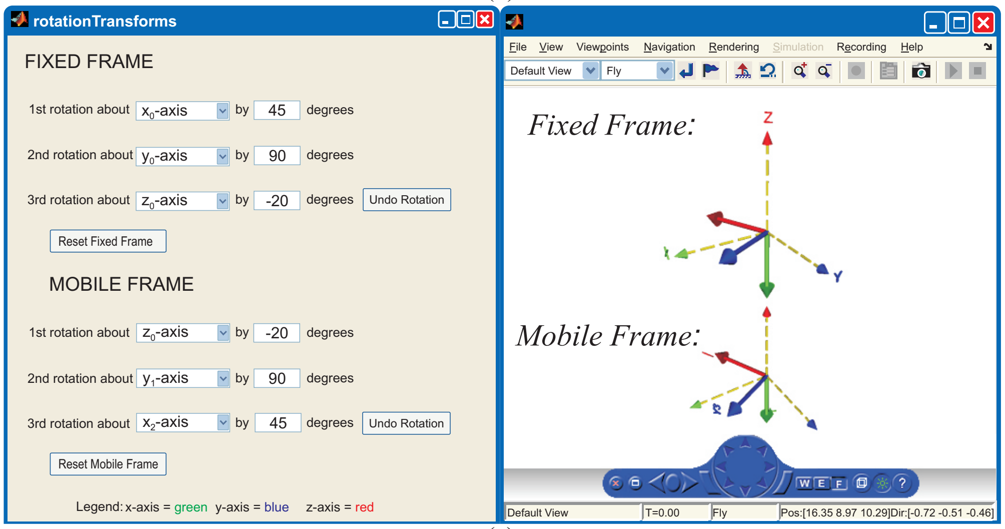
3rd rotation about x_2-axis by 45 degrees Rotate!

Reset Mobile Frame

Legend: x-axis = green y-axis = blue z-axis = red

File View Viewpoints Navigation Rendering Simulation Recording Help
Default View
Fly
Fixed Frame:
Z
Y
Mobile Frame:
W E F
Default View T=0.00 Fly Pos:[16.35 8.97 10.29]Dir:[-0.72 -0.51 -0.46]

(b)



rotationTransforms

FIXED FRAME

1st rotation about x_0-axis by 45 degrees

2nd rotation about y_0-axis by 90 degrees

3rd rotation about z_0-axis by -20 degrees Undo Rotation

Reset Fixed Frame

MOBILE FRAME

1st rotation about z_0-axis by -20 degrees

2nd rotation about y_1-axis by 90 degrees

3rd rotation about x_2-axis by 45 degrees Undo Rotation

Reset Mobile Frame

Legend: x-axis = green y-axis = blue z-axis = red

File View Viewpoints Navigation Rendering Simulation Recording Help
Default View
Fly
Fixed Frame:
Z
Y
Mobile Frame:
W E F
Default View T=0.00 Fly Pos:[16.35 8.97 10.29]Dir:[-0.72 -0.51 -0.46]

(c)

Figure 4.2 Rotation tool: visualizing the progression of mobile frame rotations when executed in the reverse order of the fixed frame rotations. Notice that the final orientations in (c) are identical for the two frame types.

Table 4.1 Comparison of Desktop Virtual Reality Rotation Tools

	RPYsim [174]	ZYZsim [174]	Bruder & Wedeward [155]	Rotation Tool
Visualize compound rotations about:				
(a) fixed frame	●		●	●
(b) mobile frame		●	●	●
User enters rotation angles	●	●	●	●
User selects rotation axes			●	●
Rotations about fixed & mobile frames are controlled separately	N/A	N/A		●
User interface prevents rotations from being initiated out of order				●
User can reverse individual rotations				●

of the other tools, the Rotation Tool also allows users to easily explore the relationship between fixed and mobile frame compound rotations. The Rotation Tool's simple user interface prevents users from performing actions which may confuse and distract them from the visualization, and the ability to move forward and backward through an individual rotation is unique.

4.3.2 Technical Implementation Details

The Rotation Tool's virtual world is defined by a VRML file that contains a scene graph which is very similar to that described in Figure A.2. This virtual world is displayed in a custom browser provided by MATLAB's Simulink 3D Animation Toolbox. The orientation of each of the frames is calculated in MATLAB according to the values entered in the GUI, and the virtual world is updated via a custom interface provided by the Simulink 3D Animation Toolbox. This custom interface is similar to the EAI described in Chapter 3, in that it allows external MATLAB programs to drive events in the virtual world. However, as will be discussed further in Chapter 5, the two interfaces differ in how they drive those events.

Within the MATLAB program, the rotation matrices R_x, R_y, and R_z are used to represent the rotations made about the x-, y-, and z-axes through the angles ϕ, β, and γ, respectively:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix}, \tag{4.1}$$

$$R_y(\beta) = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix}, \tag{4.2}$$

$$R_z(\gamma) = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.3}$$

Consecutive rotation matrices are pre- or post-multiplied for fixed or mobile frame rotations, as appropriate. A conscious decision was made to use rotation matrices instead of other rotation representations, such as quaternions, to make the underlying code more readable for students, and to reinforce the mathematical difference between fixed and mobile rotations that is taught in the lectures. However, the resultant rotation matrices must be converted to their equivalent axis-angle representations in order to be displayed in VRML, and Paul's [175] approach is used to avoid singularities when the net rotation angle is an integer multiple of 180°. In the case of no net rotation, the rotation axis is arbitrarily set to (1, 0, 0) to ensure that the VRML rotation axis remains well defined at all times.

4.4 Methodology

There were two robotics courses offered by the Department of Electrical and Electronic Engineering at UCC during the 2012/13 academic year, when this study took place: "Mechatronics and Industrial Automation" offered to fourth year Electrical Engineering undergraduates, and "Mechatronics and Robotics" offered to students in the taught Masters in Mechanical Engineering program. Although these were separate courses, the classes were combined for the lectures on compound rotations.

4.4.1 Before the Rotation Lectures

To avoid making assumptions regarding students' spatial abilities and pre-existing knowledge of compound rotations, the students were given an Initial Rotation Knowledge Assessment (IRKA) before attending any lectures on rotations. The complete IRKA is included in Appendix B. The IRKA contained eight questions to test students' mental rotation skills, carefully chosen from the rotation section of the Purdue Spatial Visualization Test PSVT:R [176]. As summarized in Table 4.2, these eight questions included both positive and negative rotations about one or two axes. They also encompassed all of the geometries identified by Onyancha et al. [154], and special care was taken to correct any errors present in the original questions [177]. Although a number of standardized mental rotation tests exist [178–180], the PSVT:R questions were chosen as the most appropriate because they do not contain impossible rotations or questions that rely on pattern recognition.

Table 4.2 Subset of Questions Selected from PSVT:R for IRKA

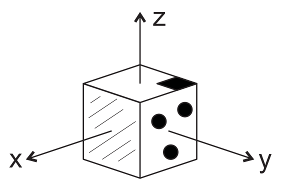
Number of Questions	Rotation Type	Rotation Angle
2	Single	±90°
2	Single	180°
2	Double	±90°
2	Double	±90°, 180°

The IRKA also contained a question to test the students' knowledge of the non-commutative nature of compound rotations, shown in Figure 4.3. The term "non-commutative" was strictly avoided in the question, in case the students had been taught the terminology in the past without really understanding it. A cube with a regular geometry and uniquely patterned faces was used to simplify the visualization, in order to ensure that the focus remained on determining the non-commutative property of compound rotations. Rotation angles of 45° made sketching the intermediate and final orientations difficult, forcing students to answer based on their existing knowledge or intuitive beliefs.

Students were given approximately 15 minutes to complete the IRKA, after being told that participation was voluntary and would have no impact on their grades for the course. It was determined that this time interval would remove the time pressure normally associated with the PSVT:R, and allow students to solve the visualization questions either holistically or analytically, as suited their preference and visualization abilities. To avoid influencing the students'

True or False?

You are given a cube (with 6 different faces) centered on the orthogonal frame $\{x, y, z\}$:



Rotating the cube 45° about the z-axis followed by a rotation of 45° about the y-axis gives a **different** final orientation than rotating the cube 45° about the y-axis followed by a rotation of 45° about the z-axis.

Figure 4.3 Non-commutative question on the IRKA.

choice of strategy, coordinate axes were not added to the PSVT:R questions [181]; however, students were allowed to add their own axes or make sketches directly on the IRKA, if desired. Students were also encouraged to provide feedback at the end of the IRKA in the form of open comments.

4.4.2 Tutorial After the Rotation Lectures

After the relevant material on rotations had been covered in the traditional lecture format, the students were invited to attend a voluntary tutorial in the computer laboratory that would use a virtual reality based tool to reinforce the topics covered in class. The tutorial was held during regularly scheduled class time so as not to interfere with the students' schedules. The students were again assured that participating would have no impact on their grades for the course.

The tutorial was formatted to follow a time-interrupted series method of evaluation appropriate for small populations, and consisted of a pre-test, treatment, and a post-test. In this case, the treatment was the use of the Rotation Tool to complete a worksheet. It was hypothesized that the students would have a better understanding of the fundamental principles of compound rotations after the treatment, as measured by the pre- and post-tests. It was further expected that the treatment would increase the students' confidence in their ability to understand fundamental compound rotation principles, as measured by survey feedback.

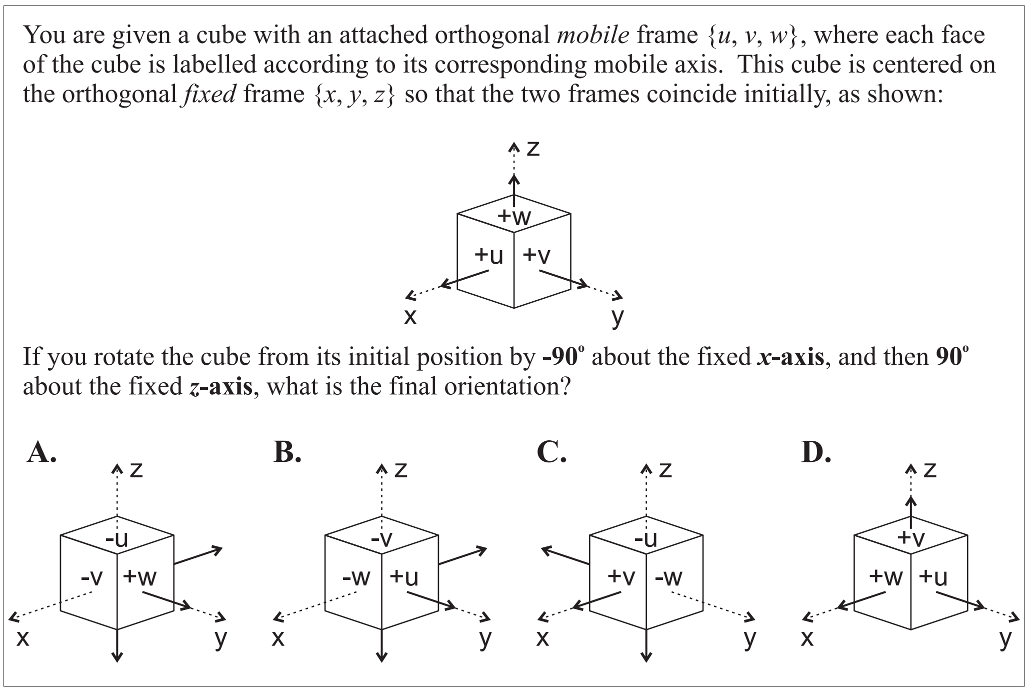
Pre-Test

The pre-test, included in Appendix B, was designed to determine the students' level of understanding after being taught about compound rotations in a traditional lecture setting. Rather than testing students' abilities to perform mental rotations, as was already tested in the IRKA, this test was designed to focus on how well they could apply the fundamental principles of compound rotations. The first page contained detailed instructions on how to complete the test, along with two examples to ensure that there was no ambiguity or confusion over nomenclature or what was being asked. This was followed by six questions on the visualization of compound rotations and the application of their principles, as summarized in Table 4.3, as well as one Likert-style survey question asking students to rate their understanding of the difference between fixed and mobile frame rotations based on what they had learned in the lectures.

Table 4.3 Pre-Test Questions

Question	Task Required
1	Visualize double rotation about fixed frame.
2	Apply knowledge of non-commutative nature of compound rotations.
3	Apply knowledge of relationship between fixed and mobile frame rotations.
4	Visualize triple rotation about mobile frame.
5	Apply knowledge of non-commutative nature of compound rotations.
6	Apply knowledge of relationship between fixed and mobile frame rotations.

The visualization questions involved the rotation of a cube by ±90° or 180°, about either a fixed frame or a mobile frame, as illustrated in Figure 4.4. A cube was selected as the rotation object to simplify the visualization task, concentrating instead on testing each student's ability to make rotations about the specified frame type and to correctly apply the right-hand rule. Each pre-test question was designed to build on the previous question, thereby also minimizing the visualization effort required. For example, Questions 2 and 3 are simply permutations of the rotations done in Question 1, and do not require students to visualize the rotations if they understand the underlying compound rotation principles. Similarly, the visualization required in Question 4 is actually just one rotation added to the result of Question 1, and Questions 5 and 6 are permutations of Question 4. All of the questions that did not specifically require



You are given a cube with an attached orthogonal *mobile* frame $\{u, v, w\}$, where each face of the cube is labelled according to its corresponding mobile axis. This cube is centered on the orthogonal *fixed* frame $\{x, y, z\}$ so that the two frames coincide initially, as shown:

If you rotate the cube from its initial position by **-90°** about the fixed ***x*-axis**, and then **90°** about the fixed ***z*-axis**, what is the final orientation?

Figure 4.4 Question 1 from the pre-test. The incorrect solutions include the results of rotations made about the wrong rotation frame type and/or in the wrong direction.

visualization included a choice of "Don't know" as an answer to discourage students from simply guessing.

The students were given approximately 20 minutes to complete the pre-test, an adequate amount of time for students who realized the relationship between the questions and possessed a good understanding of the underlying theory.

Treatment

Once the pre-tests were collected, a brief demonstration of the main features of the Rotation Tool was given. During this demonstration, two examples were completed that emphasized the main principles of compound rotations. Detailed written instructions on the Rotation Tool's use and main features, as well as the examples covered in the demonstration, were provided during the tutorial and are included in Appendix C. They were also made available online to all students registered in the course, for easy reference throughout the term.

After the demonstration, the students were asked to use the Rotation Tool to complete a worksheet, included in Appendix B, encouraging the students to explore the capabilities of the Rotation Tool with purpose. The students used the Rotation Tool to visualize different sets of compound rotations made about both fixed and mobile frames, and identified which conditions yielded the same, or different, final orientations. They were then asked to generalize their findings, thereby deriving the universal principles of compound rotations.

The students were given approximately 20 minutes to complete the worksheet using the Rotation Tool, either on their own or in pairs. They were encouraged to discuss results with each other and ask questions of the instructors, as needed.

Post-Test

The general format of the post-test, included in Appendix B, matched that of the pre-test, with six multiple choice questions testing the students' abilities to visualize different sets of compound rotations and their understanding of the compound rotation principles. A survey consisting of five Likert-style questions and two open comment questions was also included. The students were given approximately 20 minutes to complete the survey and post-test, without the aid of the Rotation Tool.

4.5 Results and Analysis

4.5.1 Before the Rotation Lectures

In the 2012/13 academic year, a total of 28 students completed the IRKA. The percentage of students who correctly answered each question is shown in Figure 4.5.

Mental Rotation Skills

The mean score for the subset of PSVT:R questions was 89.7%, with a standard deviation of 10.8%. Although a direct comparison with previous studies using the same subset of questions is not possible, such a high score clearly demonstrates that the students entered the course with high spatial skills and a good ability to perform mental rotations. Therefore, poor spatial abilities can be definitively ruled out in this study as a contributing factor to students' difficulties in visualizing the result of compound rotations.

As Figure 4.5 shows, Questions 7 and 8 each received a lower percentage of correct responses than all of the other PSVT:R questions. This indicates that

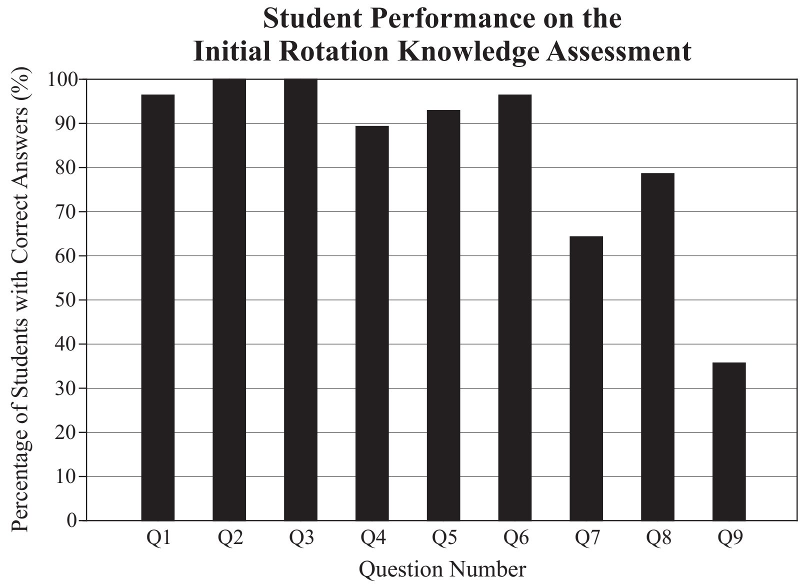


Figure 4.5 The breakdown of correct answers on the IRKA in 2012/13, where Q1-Q8 are the PSVT:R questions outlined in Table 4.2, and Q9 is the question on the non-commutative nature of compound rotations that is illustrated in Figure 4.3.

double rotations with rotation angles of ±90° and 180° are the most difficult rotation types on the PSVT:R for students to perform, consistent with the findings of Onyancha et al. [154]. In turn, this suggests that regardless of a student's ability to perform mental rotations, compound rotations become increasingly more difficult to visualize as the number of rotations and unique rotation angles increase. This could be because of increased spatial memory requirements. Regardless of the underlying reason, a visualization tool like the Rotation Tool will remove the ambiguity and difficulty in visualizing the orientation, allowing students to focus on examining the underlying compound rotation principles.

It should be noted that no distinction is made between the performance of female and male students on the PSVT:R questions, as is done in other studies. The small class size and small percentage of female students, typical of many high-level engineering courses, made such a distinction impractical. However, the large discrepancy in scores seen in other studies was not present in this study. This could be because the group of students tested here were well advanced in their engineering studies, and either had naturally strong spatial abilities that enabled them to remain in engineering, or had developed them during their studies. It could also be because a subset of the PSVT:R questions was used, and measures were taken to remove biases favoring holistic strategies over analytic ones.

Non-Commutative Nature of Rotations

Contrary to the high performance on the PSVT:R questions, only 35.7% of students answered the question on the non-commutative nature of compound rotations correctly. Such a low percentage indicates that the intuitive belief that rotations are commutative held by primary students [173] persists into higher levels of learning. Although assumed to be a straightforward fact to teach students, this could be a significant contribution to students' inability to effectively visualize compound rotations and understand their underlying principles.

Student Feedback

Seven out of the 28 participating students chose to leave comments at the end of the IRKA. Most seemed to enjoy working through the IRKA, leaving comments along the lines of, "that was fun," and a couple of students commented on the increasing difficulty of the questions as the test went on. One student did comment specifically on having difficulty visualizing the 3D shape in Question 8, stating that the "shape can be interpreted in different ways." Although questions were carefully selected in an effort to avoid these types of problems, this does highlight the difficulty of visualizing 3D shapes from isometric drawings on a 2D medium.

One student "didn't really understand [the] last question [Q9]" and thought that a "sample would help." Although all 28 students answered the question and only one student commented on the difficulty of it, it is possible that other students also found the change in format for the last question confusing. On closer examination of the returned IRKAs, it was found that nine students made deliberate marks on their paper, indicating a good understanding of what the question asked. Of these nine, only 33.3% answered the question correctly. This corresponds closely with the overall percentage of students who answered the question correctly. Thus, despite any possible confusion with the change in question format, it can be confidently concluded that the majority of the students did not understand that rotations are non-commutative.

4.5.2 Tutorial After the Rotation Lectures

During the tutorial, the students were asked to return three items: the pre-test, the worksheet, and the post-test. Of the documents that were returned in the 2012/13 tutorial, 21 complete sets could be matched up based on anonymous student identification numbers. The results for these 21 students are reported below.

Worksheet

In general, the students used the Rotation Tool to complete the worksheet without difficulty. There were very few questions asked of the instructors, and all students were observed to be eager to try the tool out for themselves, even those who were working in small groups.

Overall, students performed very well on the worksheet, further demonstrating their ability to use the Rotation Tool correctly and easily. The mean score on the worksheet was 86.2%, with a standard deviation of 19.6%. Of particular note, all 21 students correctly identified that compound rotations are non-commutative because the order of the rotations affects the final orientation. In addition, 19 of the 21 students correctly identified the relationship between fixed and mobile frame rotations.

Pre-Test vs. Post-Test

The mean pre-test score of 42.9% (standard deviation of 25.0%) was lower than expected, given that the students had already attended lectures on compound rotations. Such a low result clearly demonstrates the need for additional, non-traditional teaching methods to be included in the curriculum. The mean post-test score was 70.6% (standard deviation of 29.3%), resulting in a mean gain of 27.8% (standard deviation of 31.8%). Despite the small sample size, the distribution of gains in test scores follows a normal distribution, as determined by visual inspection and tested using the Kolmogorov-Smirnov test ($p = 0.6512$), the Shapiro-Wilk test ($p = 0.5374$), and the Anderson-Darling test ($p = 0.4034$). The improvement in test scores after the treatment is highly significant ($t = 4.01$, $p < 0.001$), with seven students earning perfect post-test scores, compared to just one earning a perfect pre-test score.

Despite the improvements in the post-test performance over that of the pre-test, it was observed that many students struggled to finish both the pre-test and post-test in the allotted time. In fact, only 11 students completed the pre-test, whereas 15 students finished the post-test. Although the Wilcoxon signed-ranks test indicates that the median increase in the number of questions answered after the treatment is statistically significant ($z = 1.91$, $p < 0.05$), it is clear that some students still had trouble finishing all six questions on the post-test. Closer examination of the returned tests reveals that a number of students made sketches for the questions that did not strictly require visualization, with a higher percentage of students making sketches for every question of the post-test than for the pre-test. Although it is encouraging to see the students take an analytical approach similar to the breakdown of rotations advocated by the Rotation Tool, it is clear that the students either did not realize the relationships between the questions or they did not trust their understanding of the relationships. Instead of applying the compound rotation principles that most were able to correctly identify when completing the worksheet just moments before, they relied on the concrete visualizations provided by sketches to answer all of the post-test questions.

Student Feedback

The students were asked to rate how well they understood the difference between fixed and mobile frame rotations, both before and after using the Rotation Tool. These ratings are compared in Figure 4.6, and clearly illustrate the students' increased confidence in their understanding after using the Rotation Tool. This corresponds with the increase in general rotation knowledge that the large majority of students felt resulted from using the Rotation Tool, as shown in Figure 4.7.

The written comments received from students at the end of the tutorial were overwhelmingly positive, with multiple comments along the lines of "this was cool" and "I'd [like] to have more classe[s] like this." The value of the Rotation Tool as a visualization aid was made particularly clear from the enthusiastic comments left by students who felt that they had struggled to understand the concepts taught in the lectures, with one student stating, "I will use it after this because I don't get this stuff at all but this will make it [a] lot easier." Even students entering the tutorial with confidence in their knowledge and ability to visualize compound rotations saw the benefit of the Rotation Tool, as evidenced by the student who commented, "Feel as if I may already have

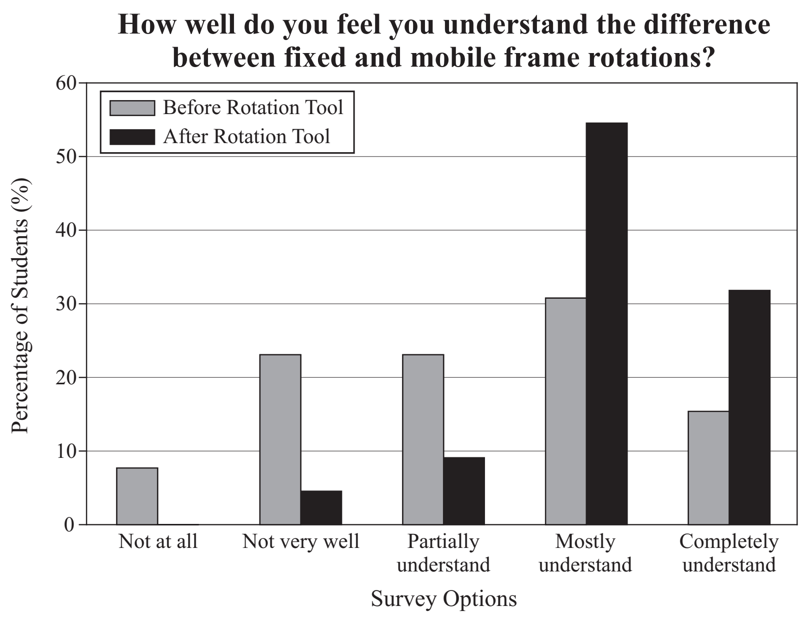


Figure 4.6 Comparison of students' self-rated level of understanding of the difference between fixed and mobile frame rotations, before and after using the Rotation Tool.

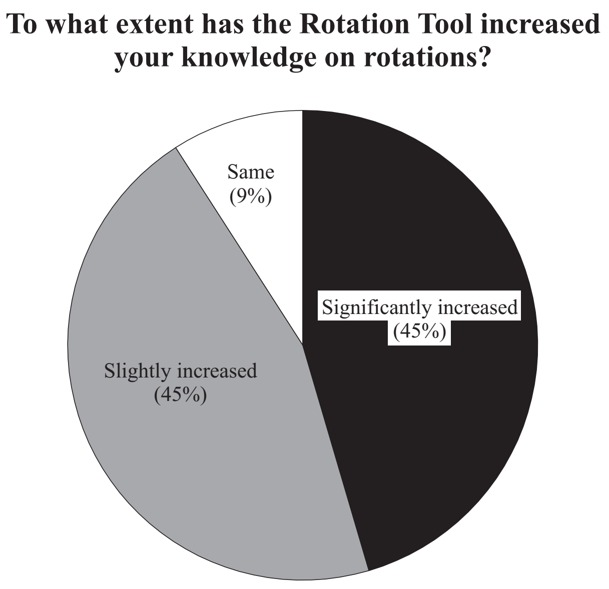


Figure 4.7 Student feedback after using the Rotation Tool. Note that no students felt that the Rotation Tool had significantly or slightly decreased their knowledge of rotations.

some degree of spa[t]ial awareness; however this tool definitely allows one to visualize this better."

4.6 Conclusions

Although the students in this study demonstrated strong mental rotation abilities, as measured by the IRKA, many exhibited relatively low self-confidence in their understanding of fundamental compound rotation principles when taught using traditional methods. This could be because traditional visualization aids are unable to overcome incorrect intuitive beliefs held by a large portion of students, such as the belief that compound rotations are commutative. Focused use of the Rotation Tool yielded significant improvements in test results and an increase in students' self-rated confidence in their understanding of compound rotation principles. As a direct result of the findings of this study, the Rotation Tool has been incorporated into the regular curriculum of the relevant robotics courses taught at UCC.

Despite the improvements seen, it is clear from the number of sketches on the returned post-tests that the students were still not completely confident in applying their knowledge of compound rotation principles, preferring to verify their answers with concrete visualizations. Although a perfectly acceptable method of solving the questions, it does suggest that there is room for improvement in how the Rotation Tool is introduced to students. For example, when the Rotation Tool is first demonstrated, the instructor can ask students to predict what they will see before executing the rotations. Similarly, the worksheet can be modified to include more probing questions as to why certain results are observed.

Chapter 5

Visualizing the Denavit-Hartenberg Parameters

It is clear from the findings of the previous chapter that virtual reality based educational tools have the potential to improve students' understanding of difficult concepts, especially when those concepts involve complex 3D visualizations. The objective of this chapter is to examine whether or not virtual reality can also aid students in their understanding of the forward kinematics of serial robot arms and the D-H parameters. The chapter begins with a discussion of why virtual reality is appropriate for this application, followed by a brief overview of the theory of forward kinematics according to the D-H convention. Existing software tools that are suitable for use in forward kinematics education are then reviewed and compared with Build-A-Robot, the virtual reality based tool created by the author using VRML, MATLAB, and the Simulink 3D Animation Toolbox. Build-A-Robot's GUI and underlying algorithms are described in detail, and the advantage of using MATLAB to directly manipulate the geometric dimensions of VRML objects is discussed. Finally, Build-A-Robot's effectiveness as a learning aid is assessed based on student feedback received after the tool was used in dedicated tutorial sessions, as well as on student performance on relevant final examination questions.

5.1 Another Case for Virtual Reality

As indicated in Chapter 4, virtual reality is playing an ever-increasing role in educational tools as a way to make abstract concepts more tangible for students. Its use aids in the visualization of 3D objects, providing realistic models that may be examined from an infinite number of viewpoints. Such visualization capabilities offer an improvement over those provided by even the most detailed 2D representations of an object on paper. With its built-in interactive capabilities, virtual reality can also be used to inspire students to go beyond the theory and static examples presented in lecture notes and textbooks, encouraging them to explore realistic scenarios safely, and to study cause and effect without worrying about damaging sensitive equipment or irreversibly affecting critical results.

Courses on robotics and mechatronics are prime candidates for incorporating virtual reality based tools into the curriculum. The mechanical structure of a serial robot arm and its position within its workspace become progressively more difficult to visualize in 3D as the DOFs increase. Moreover, certain robot positions may result in the occlusion of part of the robot, making it necessary to adjust the viewpoint. Animated multimedia presentations offer a method of illustrating a predefined motion of the robot from predefined viewpoints, but such presentations cannot be modified easily during a demonstration when responding to unanticipated questions. Using one's hand and arm to mimic a robot arm's orientation and position offers real-time flexibility, but this approach quickly reaches its limit when the considered robot configurations and workspaces differ from that of a human arm. Only virtual reality puts complete, interactive control in the students' hands and, in the absence of expensive real robots, provides the most complete form of visualization.

The multidisciplinary nature of courses in robotics and mechatronics means that a large volume of theory must be covered within the allotted lecture time, not leaving enough time to solve multiple examples on any one concept in class, let alone different nuances of the same example. This puts the responsibility on the students to solve a more comprehensive collection of examples outside of the classroom. A difficulty with this approach is that multiple, equally correct, solutions often exist for problems in robotics, giving rise to student frustration and self-doubt when their solution does not match a given sample solution. An automated tool that provides students with instant visual verification allows them to gain confidence in their problem-solving approach, and also to investigate the existence of alternative solutions.

5.2 Forward Kinematics and the Denavit-Hartenberg Parameters

As was discussed briefly in Chapter 1, forward kinematics, when applied to a robot arm, involves determining the position and orientation of a robot's end-effector relative to its base, given the position and orientation of all of the links in between. The convention introduced by Denavit and Hartenberg [14] to simplify and standardize the expression of kinematic information for an n-link serial robotic manipulator is the most popular convention used to teach university students about forward kinematics, and the D-H parameters are used to form homogeneous transformation matrices that mathematically express the relative position and orientation of reference frames corresponding to the ends of the robot's rigid links. As noted by Corke [182], there are two generally accepted versions of the D-H methodology. The work in this thesis follows the original methodology of assigning reference frames distally, as shown in Figure 5.1a, and according to the algorithm clearly described by Schilling [183]. The final reference frame assigned to the end-effector adheres to the convention described by Spong et al. [184], as illustrated in Figure 5.2. Despite the standardized rules offered by these conventions and algorithms, the assignment of reference frames is still not unique, adding to the complexity of teaching students about the D-H convention. Because it is impractical for instructors to provide students with solutions for every possible permutation of valid frame assignments, a computerized kinematics tool such as Build-A-Robot is useful because it allows students to verify the validity of solutions for themselves.

Once the frames have been assigned, the D-H parameters used to describe the robotic manipulator can be determined. Students are presented with the following definitions of the four D-H parameters identified in Figure 5.1b:

- $\tilde{d}_k$ is the link offset, and is the distance along the z_{k-1}-axis between the origin of the L_{k-1} reference frame and the intersection point of the x_k- and z_{k-1}-axes;

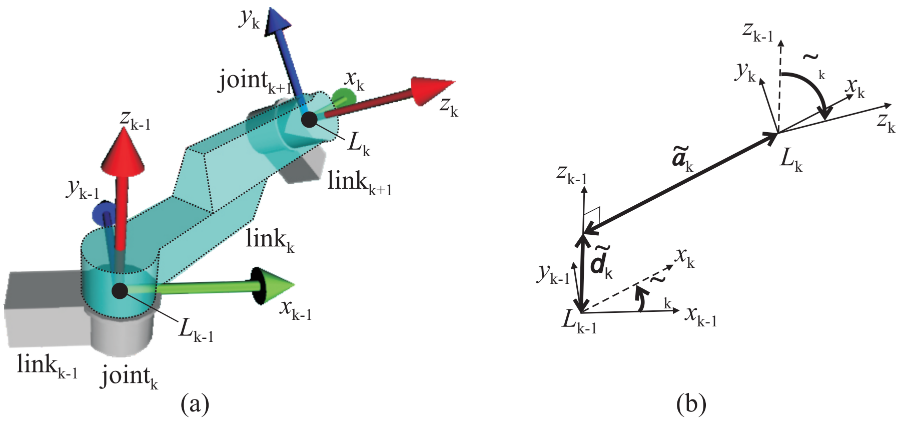


Figure 5.1 a) Distal frame assignment, and b) D-H parameters for the k^{th} link.

- $\tilde{\theta}_k$ is the joint angle, and is the rotation angle from the x_{k-1}-axis to the x_k-axis, about the z_{k-1}-axis (with the right-hand rule determining the direction of positive rotation);
- $\tilde{a}_k$ is the link length, and is the distance along the x_k-axis between the origin of the L_k reference frame and the intersection point of the x_k- and z_{k-1}-axes; and
- $\tilde{\alpha}_k$ is the link twist, and is the rotation angle from the z_{k-1}-axis to the z_k-axis, about the x_k-axis (again, with the right-hand rule determining the direction of positive rotation).

To provide students with a mathematical appreciation of how these parameters are used to relate the orientation and position of frame L_k to frame L_{k-1}, they are presented with the following corresponding homogeneous transforma-

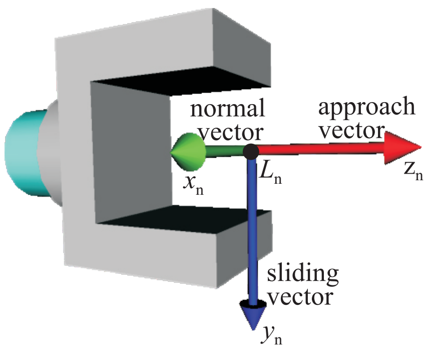


Figure 5.2 n^{th} D-H frame associated with the end-effector.

tion derivation:

$$\begin{aligned} T_{k-1}^{k} &= R_z(\tilde{\theta}_k) Trans_z(\tilde{d}_k) Trans_x(\tilde{a}_k) R_x(\tilde{\alpha}_k) \\ &= \begin{bmatrix} \cos(\tilde{\theta}_k) & -\sin(\tilde{\theta}_k)\cos(\tilde{\alpha}_k) & \sin(\tilde{\theta}_k)\sin(\tilde{\alpha}_k) & \tilde{a}_k\cos(\tilde{\theta}_k) \\ \sin(\tilde{\theta}_k) & \cos(\tilde{\theta}_k)\cos(\tilde{\alpha}_k) & -\cos(\tilde{\theta}_k)\sin(\tilde{\alpha}_k) & \tilde{a}_k\sin(\tilde{\theta}_k) \\ 0 & \sin(\tilde{\alpha}_k) & \cos(\tilde{\alpha}_k) & \tilde{d}_k \\ 0 & 0 & 0 & 1 \end{bmatrix}, \end{aligned} \tag{5.1}$$

which is derived from the appropriate translation vectors, and the rotation matrices from Equations 4.1 and 4.3. It is important that students are comfortable with the fact that $\tilde{\theta}_k$ will be variable and $\tilde{d}_k$ will be constant in Equation 5.1 when the k^{th} joint is revolute, whereas $\tilde{\theta}_k$ will be constant and $\tilde{d}_k$ will be variable when it is a prismatic joint.

Conventional methods of teaching and testing forward kinematics require that students select a soft home position by selecting an appropriate value for each variable parameter, $\tilde{\theta}_k$ or $\tilde{d}_k$. It has been observed that this causes considerable confusion for many students at UCC: it is difficult to convince students through static examples in a lecture that the non-variable D-H parameters for a given structure and frame assignment will remain the same regardless of what soft home position is chosen, even if it does not match the position used in the sample solution. An interactive tool that allows students to visualize the effect of changing the soft home position will alleviate this confusion and allow students to gain a deeper appreciation of what each D-H parameter represents.

5.3 Existing Forward Kinematics Educational Software

A number of non-commercial software tools have been developed over the years to model the kinematics of serial robot manipulators. Robotica [185] for Mathematica and Corke's Robotic Toolbox [186] for MATLAB/Simulink are two prominent examples of software packages with textual interfaces, with each containing a comprehensive set of functions related to forward kinematics and other topics in robotics. Active learning is encouraged by such tools through the process of programming, which has been shown to provide insight into mathematically complex topics, like robot kinematics, when used by students with strong programming skills [187]. For students who are less comfortable with textual programming, RobotiCad [188] and RIO [189] add a GUI to the functions in Corke's toolbox to improve accessibility. RIO's GUI allows students to experiment with changing the pose of pre-built virtual robots created with VRML.

RobUALab [190], the UJI online robot [191], and the VRML 2.0 Robot [192] provide detailed virtual reality models of real robots for students to experiment with in safe, virtual environments before sending instructions to move the real robots via teleoperation. As with RIO, all robots that are studied must have custom models pre-programmed into the tools, allowing students to investigate the role of the variable D-H parameters for a limited set of robots. The UJI online robot and the VRML 2.0 Robot provide natural means of interacting with their virtual scenes, such as through spoken commands and intuitive mouse movements. Such high-level interactivity options decrease the cognitive load associated with using the tools, making it easier for students to focus on the concepts being learned [193].

Rather than studying the forward kinematics of virtual robot models already created, the EjsRL library [194] and ARTE [195] allow students to specify all of the D-H parameters of a chosen robot and associate their model with custom 3D graphics files. Similarly, RobotScene [196] provides a sophisticated GUI in which students can create and assemble 3D virtual robot links according to the D-H parameters that they have determined. These tools require students to conduct a detailed analysis of one robot arm, resulting in an in-depth understanding of the forward kinematics for that particular robot.

Cakir and Butun's tool [197] and ROBOLAB [198], both based in MATLAB, emphasize the investigation of robot kinematics for general robot configurations rather than exploring the properties of specific robots. Cakir and Butun's tool allows for the visualization of robots with up to six revolute joints, whereas ROBOLAB allows for the study of six DOF robots with both revolute and prismatic joints, based on a library of 16 pre-programmed combinations of these joints. 3D-RAS [199], based in LabView, is similar to these tools in that it allows students to visualize, in real-time, the effects of changes to any of the D-H parameters for general anthropomorphic robot configurations with up to five DOFs.

Robot-Draw [200] uses a Perl program to automatically generate VRML models for any robot arm with four to six DOFs, giving users complete flexibility over the D-H parameters entered and the type of each joint. A new virtual model is generated for each set of D-H parameters studied. RModelo, available with the purchase of Manseur's textbook [174], is Robot-Draw's successor. This tool allows users to enter the D-H parameters for up to 20 prismatic or revolute joints, all of which can be animated to move from an initial position to a final one, as specified by the user before the virtual model is generated.

RoboAnalyzer [201] allows students to enter and modify all of the D-H parameters of a serial robot arm, with any changes reflected immediately in the virtual model. Thus, students can interact with the tool to change the frame assignments, the soft home position, and even the structure of the robot in real-time. As of version V7.1 of this tool [202], 19 pre-programmed general robot configurations are available for students to choose from. Of these configurations, prismatic joints can be selected for general robots of five DOFs or less, whereas robots containing only revolute joints may have up to seven DOFs.

SnAM [203] simulates the kinematics of robot arms with n DOFs and, like RoboAnalyzer, can be used to investigate the effect of modifying any of the D-H parameters in real-time. This flexibility makes it an excellent prototyping tool for those already familiar with the theory of forward kinematics.

5.4 Build-A-Robot

5.4.1 Tool Description

Build-A-Robot is an interactive forward kinematics tool that combines the most important features identified in the previous section into a single tool, as summarized in Table 5.1. It consists of a MATLAB GUI and a virtual reality window that displays the robot's 3D link-coordinate representation, as shown in Figure 5.3. It is controlled by a MATLAB program interacting with a single, universal VRML file that contains the virtual reality specifications for Build-A-Robot's generic link-coordinate model. Students use the GUI to select the desired DOF for their robot, up to a maximum of seven degrees, allowing them to visualize both kinematically simple and redundant robot arms. Editable fields where students enter the D-H parameters are displayed in the GUI according to the number of DOFs selected, along with an option to make each joint either revolute or prismatic. Once all parameters are input into the GUI, the corresponding link-coordinate model is updated and displayed by pressing the *Build Robot* button. In the virtual model, revolute joints are depicted by cylinders, whereas prismatic joints are portrayed by rectangular

Table 5.1 Comparison of Forward Kinematics Tools

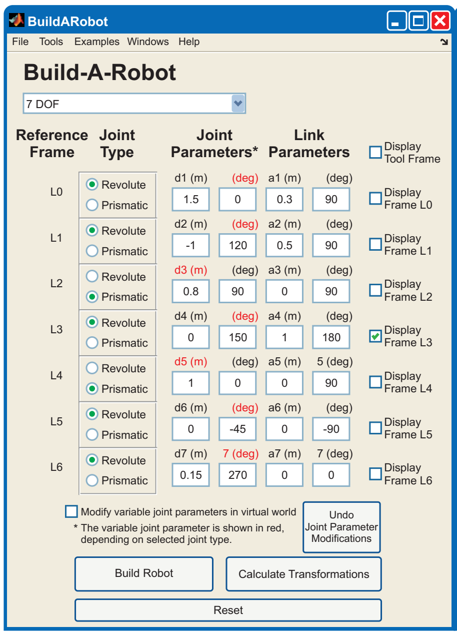
	RModelo [174]	Cakir & Butun [197]	ROBOLAB [198]	3D-RAS [199]	SnAM [203]	RoboAnalyzer [202]	Build-A-Robot
Underlying programming language(s)	Visual Basic, VRML	MATLAB	MATLAB	LabView	C++, OpenGL	Visual C#, OpenGL	MATLAB, VRML
DOF	up to 20	up to 6	6	up to 5	n	up to 7	up to 7
Joint permutations	all	limited	limited	limited	all	limited	all
Virtual model's modifiable D-H parameters	variable $\tilde{\theta}$ or $\tilde{d}$	$\tilde{\theta}$	variable $\tilde{\theta}$ or $\tilde{d}$	$\tilde{\theta}$, $\tilde{d}$, $\tilde{\alpha}$, $\tilde{a}$	$\tilde{\theta}$, $\tilde{d}$, $\tilde{\alpha}$, $\tilde{a}$	$\tilde{\theta}$, $\tilde{d}$, $\tilde{\alpha}$, $\tilde{a}$	$\tilde{\theta}$, $\tilde{d}$, $\tilde{\alpha}$, $\tilde{a}$
Educational focus	●	●	●	●		●	●
Modify D-H parameters:							
(a) in real-time		●	●	●	●	●	●
(b) via GUI	●	●	●	●	●	●	●
(c) via virtual scene							●

parallelepipeds. To aid with visualization, each link offset, $\tilde{d}_k$, is shown in blue, and each link length, $\tilde{a}_k$, is shown in purple.

Once built, users can navigate the virtual world and view the model from any angle. Any of the D-H parameters can be changed in the GUI, and this change will instantly be reflected in the existing virtual model. Users can also click and drag on the virtual robot's links to change the soft home position directly in the virtual environment, and view the resulting change in the numeric value of each variable joint parameter, $\tilde{\theta}_k$ or $\tilde{d}_k$, in the GUI.

As can be seen from the contents of Build-A-Robot's menus in Figure 5.4, the data entered into the GUI can be saved at any time as a text-based **.br* file, allowing students to save specific robot configurations to study in further detail at a later time. The ability to generate **.br* configuration files also allows instructors to save and distribute sample solutions to in-class examples and past examination questions. When opened, the contents of these **.br* files are used to automatically populate Build-A-Robot's GUI, which can then be used to update and display the virtual link-coordinate model, as normal.

Reference frames can be added to the virtual display so that students can gain a better understanding of the rotation parameters, $\tilde{\theta}_k$ and $\tilde{\alpha}_k$, and also to confirm the different sets of D-H parameters that result from different frame assignments. The frames can similarly be removed at any point, so as not to


BuildARobot

File Tools Examples Windows Help

Build-A-Robot

7 DOF

Reference Frame	Joint Type	Joint Parameters*		Link Parameters		Display Tool Frame ☐
L0	[✓ Revolute] Prismatic	d1 (m) 1.5	(deg) 0	a1 (m) 0.3	(deg) 90	☐ Display Frame L0
L1	[✓ Revolute] Prismatic	d2 (m) -1	(deg) 120	a2 (m) 0.5	(deg) 90	☐ Display Frame L1
L2	Revolute [✓ Prismatic]	d3 (m) 0.8	(deg) 90	a3 (m) 0	(deg) 90	☐ Display Frame L2
L3	[✓ Revolute] Prismatic	d4 (m) 0	(deg) 150	a4 (m) 1	(deg) 180	✓ Display Frame L3
L4	Revolute [✓ Prismatic]	d5 (m) 1	(deg) 0	a5 (m) 0	5 (deg) 90	☐ Display Frame L4
L5	[✓ Revolute] Prismatic	d6 (m) 0	(deg) -45	a6 (m) 0	(deg) -90	☐ Display Frame L5
L6	[✓ Revolute] Prismatic	d7 (m) 0.15	7 (deg) 270	a7 (m) 0	7 (deg) 0	☐ Display Frame L6

☐ Modify variable joint parameters in virtual world

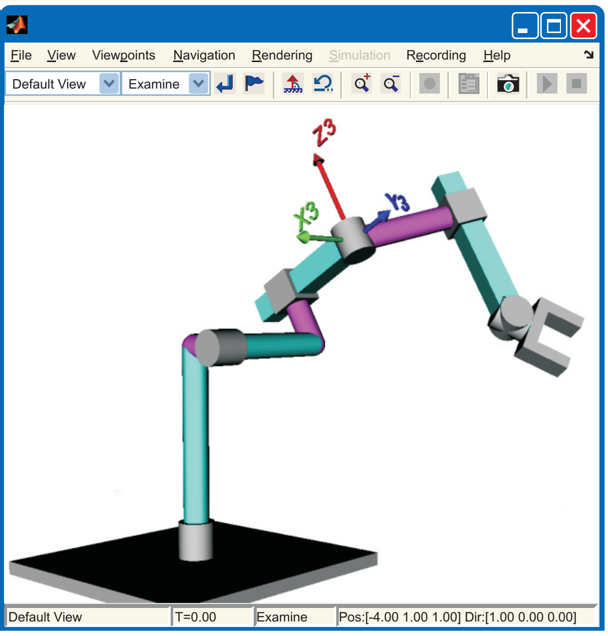
* The variable joint parameter is shown in red, depending on selected joint type.

Undo Joint Parameter Modifications

Build Robot

Calculate Transformations

Reset


File View Viewpoints Navigation Rendering Simulation Recording Help
Default View
Examine
Z_3
X_3
Y_3
Default View
T=0.00
Examine
Pos:[-4.00 1.00 1.00] Dir:[1.00 0.00 0.00]

(a) (b)

Figure 5.3 Build-A-Robot's a) GUI, and b) corresponding virtual reality 3D link-coordinate model. The scene background has been removed for clarity when printed.

clutter the model unnecessarily, either via the individual checkboxes on the GUI, or by deselecting "Display All Reference Frames" in the Tools menu.

To reinforce the concept of homogeneous transformations, the overall transformation relating the end-effector's coordinate frame to the base coordinate frame can be calculated and displayed by pressing the *Calculate Transformations* button. The homogeneous transformations relating each coordinate frame, L_k, to the previous frame, L_{k-1}, are also displayed, allowing students to see the progression of transformations. If the window containing the homogeneous transformations is closed prematurely, it can be reopened by again pressing the *Calculate Transformations* button, or by selecting the "Transformation Matrices" option in the Windows menu. The Windows menu also contains an option for reopening the virtual model window, in case it is accidentally closed.

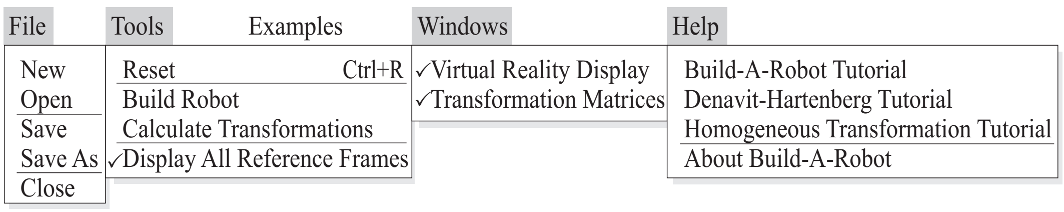

File	Tools	Examples	Windows	Help
New	Reset Ctrl+R		✓Virtual Reality Display	Build-A-Robot Tutorial
Open	Build Robot		✓Transformation Matrices	Denavit-Hartenberg Tutorial
Save	Calculate Transformations			Homogeneous Transformation Tutorial
Save As	✓Display All Reference Frames			About Build-A-Robot
Close				

Figure 5.4 Build-A-Robot's menu options.

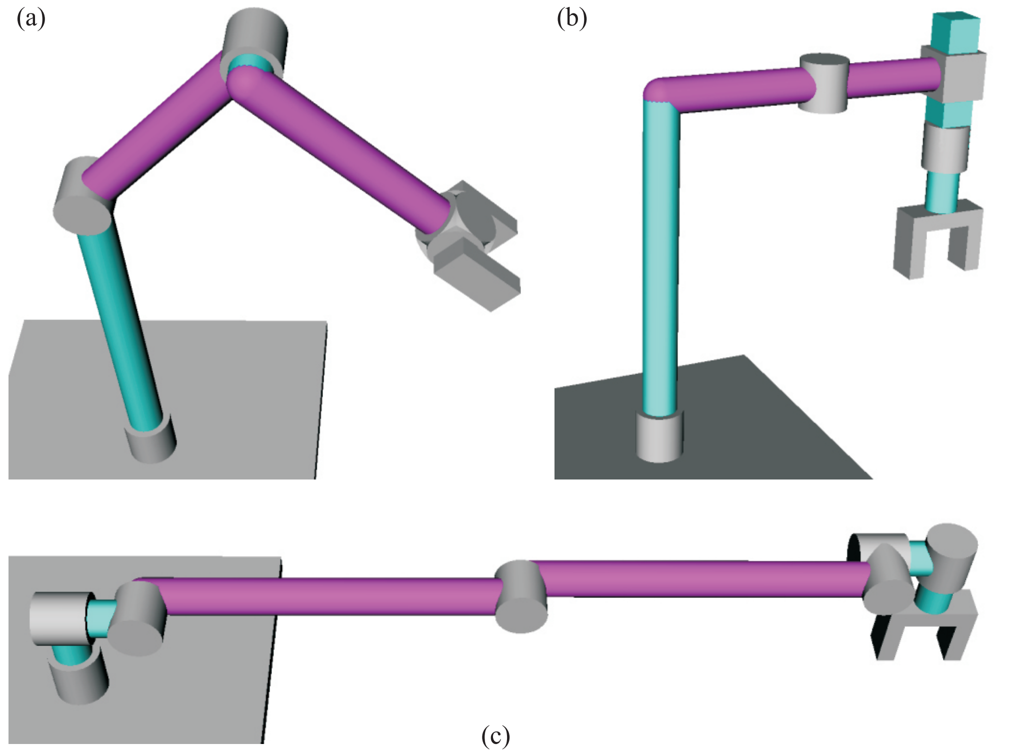


Figure 5.5 Preconfigured models of real robots included with Build-A-Robot: a) PUMA 560, b) Sankyo SR8408 SCARA, and c) Canadarm2.

Some of the additional features of the tool include the ability to view preconfigured example models that are based on real-world robotic arms, some of which are illustrated in Figure 5.5, and to access a Help menu that contains tutorials both on using Build-A-Robot and on forward kinematics theory. Such features ensure that the tool is easy for students to use on their own, outside of the classroom.

5.4.2 Technical Details

As mentioned previously, a single VRML file defines the generic virtual robot used in Build-A-Robot, along with its virtual world. The general structure of this VRML file is shown in Figure 5.6. Notice that each joint and link segment in the generic robot model is made up of both a cylinder and box. The transparency of each segment is controlled by MATLAB in real-time, according to what is entered in Build-A-Robot's GUI. This makes it unnecessary to pre-program libraries of specific permutations of joint combinations from which the user can choose; all 254 possible configurations for robots with up to seven

DOFs are automatically incorporated in Build-A-Robot's universal VRML file, but only the relevant links and joints are displayed.

Each link in the link-coordinate model is a child of the previous link, thereby automatically incorporating all of the previous rotations and translations, and minimizing the number of calculations that MATLAB must perform behind the scenes. Even with this built-in efficiency, each rotation requires two calculations: one to determine the nine-element rotation matrix based on the relevant joint angle or link twist entered, and another to convert this rotation matrix into its equivalent, VRML recognized, axis-angle representation. As with the Rotation Tool, Paul's [175] approach is used to carry out this conversion, thus avoiding display problems at singularities.

In Build-A-Robot's link-coordinate model, CylinderSensor and PlaneSensor nodes are enabled for all visible revolute and prismatic joints, respectively. When one of these sensor nodes detects a mouse press on its associated robot link, it transforms any subsequent mouse dragging motions into either a joint rotation or translation, as appropriate for that joint type. This offers students an intuitive way of interacting with the virtual robot, which may feel more natural and immersive for some than using a separate GUI to change parameter values.

Perhaps the most important feature of VRML, as it relates to Build-A-Robot, is the ability to name nodes with the DEF keyword. These uniquely named nodes can be referenced by external programs, either through the EAI [132], or through a custom interface. As described in Chapter 3, when accessed through the EAI, the values of *exposed fields* can be changed by the external program, whereas *field* values cannot. This restriction limits the capabilities of external programs when using the EAI to interface with VRML worlds [204]. The nodes in Build-A-Robot's VRML file are not accessed by the EAI, however; instead, they are accessed through a custom interface provided by MATLAB's Simulink 3D Animation Toolbox. Because the Simulink 3D Animation Toolbox uses a custom browser to render its VRML scenes, it is able to offer extended functionality in its custom interface over the standard EAI. One such extension is the ability to access and modify *all* fields from MATLAB (J. Houska, personal communication, 2011). This extended capability means that fundamental geometric properties such as a cylinder's height and a box's size may be changed directly by MATLAB. This goes beyond the ability to simply scale VRML objects, and it means that the dimensions of a shape can be set to zero instead of just very close to zero. It also allows a parent

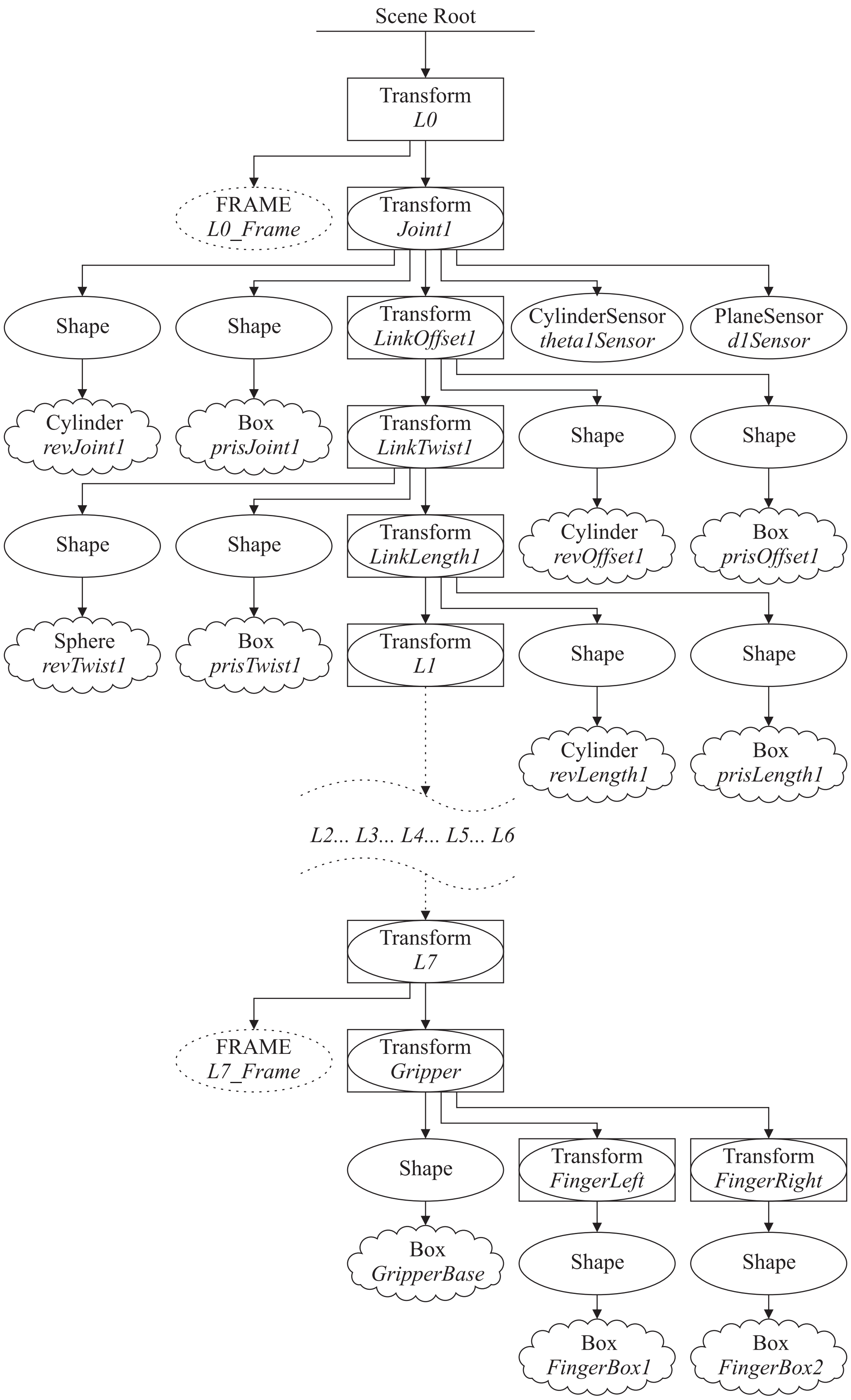


Figure 5.6 Key geometric components of Build-A-Robot's VRML hierarchy.

link's length to be changed without affecting the length of any of its children links. Thus, one universal VRML file can be created to describe a completely generic robot, which can, in turn, be configured by an external, interfacing MATLAB program. In this case, the link lengths and link offsets in Build-A-Robot's universal VRML file are initially set to zero, and MATLAB changes the geometry of each link according to the joint types chosen and the D-H parameters entered by the student after the virtual world is created. Such direct manipulation by MATLAB offers real-time interactivity that is not possible with tools that rely on the creation of unique virtual models for each set of D-H parameters studied.

5.5 Tool Assessment

Two robotics courses were offered by the Department of Electrical and Electronic Engineering at UCC during this study: "Mechatronics and Industrial Automation" offered to fourth-year Electrical Engineering undergraduates, and "Mechatronics and Robotics" offered to students in the taught Masters in Mechanical Engineering program. In each of the 2010/11, 2011/12, and 2012/13 academic years, Build-A-Robot was introduced to both classes as part of a non-compulsory tutorial session on forward kinematics. During this time, over 100 students were given the opportunity to use Build-A-Robot.

In each tutorial, students were given a brief demonstration of the main features of the tool while the instructor used it to consider a past final examination question on forward kinematics. The students were also given written instructions on these main tool features as they pertained to the example, which they could refer to during the tutorial and when studying independently afterwards. These instructions are included in Appendix D. After the demonstration, the students were encouraged to use the tool, either on their own or in pairs, asking questions of the instructor as required. Students were provided with continued access to Build-A-Robot after the tutorial, along with a number of Build-A-Robot configuration files that gave sample solutions to previous years' final examination questions, thereby encouraging students to use the tool as a study-aid throughout the semester.

Table 5.2 Summary of Students' Opinions Regarding Build-A-Robot

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
This would be a useful addition to the course. (Asked in 2011 only.)	0%	9%	0%	36%	**55%**
	Very Difficult	Somewhat Difficult	Neutral	Somewhat Easy	Very Easy
How easy is this tool to use?	0%	2%	0%	38%	**61%**
	Much More Difficult	More Difficult	Same	Somewhat Easier	Much Easier
How much easier is it to understand the interactive 3D models than the 2D drawings given in class?	0%	2%	0%	37%	**61%**
	Significantly Decreased	Slightly Decreased	Same	Slightly Increased	Significantly Increased
To what extent has this tool increased your knowledge on forward kinematics? (Asked in 2012 and 2013.)	0%	0%	4%	**74%**	22%

5.5.1 Student Feedback

After using Build-A-Robot in the tutorial, the students were asked to give their opinions on the tool by filling out a survey with Likert-style questions. The most relevant results are summarized in Tables 5.2 and 5.3. As can be seen, the feedback from the students has been overwhelmingly positive, with over 90% of respondents from the first offering of the tutorial either agreeing or strongly agreeing that Build-A-Robot would make a useful addition to the course. Every year, a majority of students have consistently responded that Build-A-Robot is very easy to use, and that it is much easier to understand the 3D models produced by the tool than the 2D drawings given in class.

In the second and third offerings of the tutorial, additional questions were included on the survey to gain a deeper insight into the students' confidence levels regarding their knowledge of forward kinematics after using Build-A-Robot. This modified survey is included in Appendix D. In both years that the modified survey was used, the majority of students felt that the tool had increased their knowledge of forward kinematics and rated their understanding of specific forward kinematics principles highly, as summarized in Table 5.3.

Table 5.3 Students' Self-Rated Understanding of Forward Kinematics Principles After Using Build-A-Robot

After using the tool, how well do you feel you understand the following: (1 = Not at all, 5 = Completely understand)	1	2	3	4	5
a. The D-H parameters and what they represent.	2%	0%	29%	**44%**	24%
b. Changing the soft home position **never** has an effect on parameters $\tilde{a}$ and $\tilde{\alpha}$.	2%	4%	24%	**49%**	20%
c. When assigning reference frames using the D-H algorithm, there are multiple correct solutions.	2%	2%	13%	39%	**43%**
d. Different frame assignments will affect the D-H parameters and intermediate transformations, but the **overall** homogeneous transformation is always the same for the same soft home position.	2%	4%	22%	**52%**	20%

In addition to the positive survey scores, there were multiple written comments regarding students' positive experiences with Build-A-Robot, especially from those who struggled with the concepts of the D-H convention when taught using more traditional methods in class. For example, one student remarked that he or she "didn't know much about [the D-H] parameters in the first place. [Build-A-Robot] is very helpful in understanding forward kinematics." Yet another student commented that the tool "clarified in one hour, months of confusion with Denavit Hartenberg principles." A majority of students indicated their intention to use Build-A-Robot outside of the tutorial, with one student stating that "this tool has been very helpful for me. I'll [be] back to practice more to understand better."

5.5.2 Student Examination Performance

As part of the final examination for "Mechatronics and Industrial Automation" every year, the students are asked one question on forward kinematics. The median ratios comparing students' performance on the forward kinematics question relative to their performance on the remainder of the examination, in the years before and after Build-A-Robot was introduced, are listed in Table 5.4. A ratio of 1.00 signifies that the students performed as well on the forward kinematics question as on the remainder of the examination, whereas a score greater than 1.00 signifies better relative performance on the forward kinematics question. As can be seen from Table 5.4, the median performance ratios in each

Table 5.4 Relative Performance on Forward Kinematics Final Examination Question

Year	Number of Students	Median Performance Ratio	U Value*	Significance Level*
Before Build-A-Robot**	123	1.02	–	–
2010/11	34	1.34	1212.0	$p < 0.0001$
2011/12	31	1.20	1674.5	$p > 0.05$
2012/13	32	1.21	1368.0	$p < 0.005$

* Based on the one-tailed Mann-Whitney U test, compared to the performance in the years before Build-A-Robot's introduction.

**Four years of combined results

of the three years that Build-A-Robot was used were higher than the median ratio prior to the introduction of the tool.

Due to the non-normality of the data, typical of high-level engineering courses with small class sizes, the Mann-Whitney U test was used to further compare the distributions of performance ratios for before and after Build-A-Robot was introduced. In 2010/11, the first year that Build-A-Robot was used, the Mann-Whitney U test indicates that the distribution of performance ratios was significantly higher than that from the years prior to Build-A-Robot's introduction. In 2011/12, the forward kinematics question involved a robot arm with greater kinematic complexity than the robot arms used on examinations in previous years. It was anticipated that this additional difficulty would not present a problem for students, and although the increase in the median performance ratio supports this supposition, the Mann-Whitney U test indicates that the increase in the distribution of performance ratios in 2011/12 was not statistically significant. Closer examination of the returned answers to the 2011/12 forward kinematics question revealed a clear divide between those with a solid understanding of the D-H principles and those without, and highlighted ways to improve the tutorial and presentation of Build-A-Robot in subsequent years. The tutorial in 2012/13 was modified accordingly, providing students with more time to use Build-A-Robot to focus on specific principles of forward kinematics. According to the Mann-Whitney U test, the distribution of students' performance ratios in 2012/13 again saw a significant increase over that from the years prior to Build-A-Robot's introduction.

The combined distributions of students' relative performance on the forward kinematics question is shown in Figure 5.7, and clearly demonstrates the overall shift in understanding of forward kinematics since Build-A-Robot was intro-

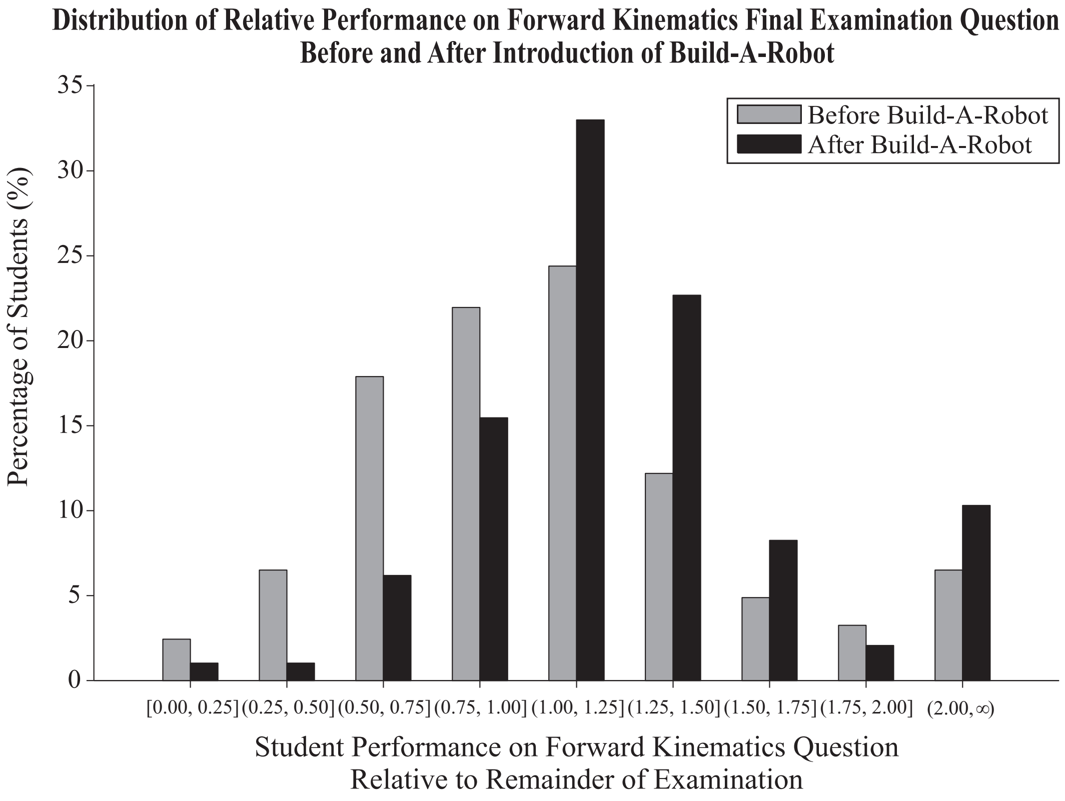


Figure 5.7 Distribution of student performance on the forward kinematics question relative to performance on the remainder of the final examination, in the years before and after Build-A-Robot was introduced.

duced, especially among those with the poorest levels of relative understanding. In the years before Build-A-Robot was used, almost half of the students (44%) performed worse on the forward kinematics question than they did on the other questions on the final examination. After Build-A-Robot was introduced, this proportion dropped to less than a quarter of the students (23%).

5.6 Conclusions

As a forward kinematics learning tool, Build-A-Robot is second to none in the freedom it offers to students in exploring the effect that each D-H parameter has on frame assignment, robot arm pose, and even the structure of serial robot arms. It provides familiar, real-world examples for the students to investigate, and also allows them to create robot arms with uncommon configurations, without the limitations imposed by pre-programmed libraries of configuration options. The implementation of one universal VRML file provides real-time interactivity not available from tools that must generate a unique virtual model for every set of D-H parameters studied, and provides students with the option to build, and modify, a virtual robot by updating one D-H parameter at a time.

The enthusiastic feedback received from students after using Build-A-Robot demonstrates the merit of incorporating interactive visualization tools into the curriculum. Students' high self-rated confidence in their knowledge of forward kinematics, as well as significant improvements in their relative performance on the final examination, indicate the effectiveness of the tool when used to demonstrate specific D-H principles. The positive effects of using Build-A-Robot as a visualization aid were particularly noticeable among the students who struggled to learn the concepts of the D-H convention using traditional means alone. As a direct result of the findings of this study, Build-A-Robot has been incorporated into the regular curriculum of the relevant robotics courses taught at UCC.

In addition to assisting students to learn about forward kinematics, Build-A-Robot also demonstrates the power of using MATLAB, in combination with the Simulink 3D Animation Toolbox, to directly manipulate fields in a VRML file without the restrictions imposed by the EAI. As a result, the fundamental geometric dimensions of virtual objects in generic VRML files, and indeed *any* field of any named node, can be changed in real-time. Such flexibility, especially the ability to set those dimensions to zero, offers great potential in the development of engaging virtual reality based educational tools in many areas of engineering education.

Chapter 6

Calibrating a Dual-Axis Accelerometer with Dynamic Accelerations

Having established a firm understanding of forward kinematics according to the D-H convention in the previous chapter, this chapter aims to apply these principles to plan the motion of an accelerometer attached to the end-effector of a real SCARA, and in doing so, calibrate the x- and y-axes of the accelerometer. The chapter begins with a discussion of accelerometer sensor calibration techniques that rely on static accelerations, and the limitations of this approach when used in conjunction with a SCARA. Two novel accelerometer calibration methods that use dynamic accelerations are introduced to overcome these limitations, with consideration given to the likelihood that the SCARA's true kinematic parameters will deviate from the ideal D-H model specified by the manufacturer. The chapter concludes by analyzing the validity of these new calibration routines, as verified by experimental results.

6.1 The Need for a New Calibration Routine

In addition to the diverse list of example applications provided in Section 3.2, accelerometers – like the one included on Tyndall's WIMU – are also widely used in robotics. They are most often utilized to provide information regarding the current position and/or orientation of a robot's intermediate joints [205–207] or of its end-effector, often with the intention of using that information in the control system of the robot in order to improve its accuracy [208]. Unlike motion-detecting systems that include cameras, lasers, or ultrasonic sensors, accelerometers do not suffer from occlusion while tracking the motion of a robot. This, combined with the low cost and small size of MEMS accelerometers, makes them a popular choice to include as part of a robot's overall system. Accelerometers have also been used to determine the true kinematic parameters of a specific robot at a specific time [209–213], which as described in Chapter 1, are likely to deviate from the ideal parameters specified by the manufacturer.

Before accelerometers can be used for such applications, however, they must be properly calibrated. Each axis of an accelerometer is most commonly calibrated according to the following linear model [214, 215]:

$$a_{s,\lambda} = (a_{m,\lambda} - b_\lambda)/S_\lambda - \eta_\lambda, \tag{6.1}$$

where $a_{s,\lambda}$ is the partially calibrated sensor data for the λ-axis of the accelerometer being considered (it has not yet been adjusted for the misalignment of its axes relative to a specific reference frame), $a_{m,\lambda}$ is the raw sensor measurement, b_λ is the bias, S_λ is the sensitivity, and η_λ represents the collective non-idealities in the system. As can be seen from Equation 6.1, the sensitivity parameter is effectively a scaling factor that relates the measured and calibrated accelerations, and in this thesis it converts the accelerometer measurements in least significant bits (LSBs) to the corresponding physical acceleration in m/s^2. The non-idealities component includes the effects of MEMS sensor nonlinearities, cross-axis effects, and white noise [216, 217].

Although typical bias and sensitivity parameters are specified for MEMS accelerometers by their manufacturers, these specifications cannot be guaranteed for individual chips due to manufacturing tolerances during the mass production of MEMS sensors, and to the mechanical stresses placed on individual chips when soldered into a circuit. Indeed, simply using an accelerometer places mechanical stress on the internal structure of the sensor, especially if the accelerometer is exposed to accelerations that are larger than its rated maximum acceleration. Thus, each individual sensor must be calibrated to find its unique set of calibration parameters. This calibration should be performed just before it is used, because the calibration parameters of a given accelerometer change with the age of the sensor, the ambient temperature, and the power levels supplying the accelerometer's circuit. Indeed, accelerometers should be recalibrated on a regular basis, but especially when there are significant changes to any of these factors.

The simplest and most popular accelerometer calibration routine relies on static acceleration measurements made when the accelerometer is resting in different orientations. This can most easily be done by manually rotating the accelerometer on a level surface so that each of the accelerometer's axes, in turn, points in the direction of, or against, gravity [218]. However, additional methods have been developed that do not require such careful selection of poses [219–221]. These calibration routines are typically performed prior to the attachment of the accelerometer to the robot, but this introduces an unknown transformation between the sensor and the robot when the accelerometer is eventually mounted to the robot. Even with the most careful of attachments, this will introduce errors to the measurements of the robot's motion.

Instead, it is preferred to calibrate the accelerometer in-place, after it has been attached to the robot. Calibration using static accelerations can be achieved if the robot is able to change the orientation of the accelerometer with respect to the direction of gravity [207, 208, 222, 223]. However, such motion is not always possible for accelerometers that are attached to intermediate joints, or for accelerometers attached to the end-effector of low DOF robots. Such is the case with the SR8408 SCARA, whose nominally parallel revolute joints all have axes that are approximately aligned with the direction of gravity when the base is level. This results in the SCARA only being able to change the orientation of the accelerometer within the plane that is orthogonal to gravity. Thus, an accelerometer calibration routine performed by a SCARA must instead rely on the introduction of known dynamic accelerations, as described in the next section.

6.2 Calibrating with Dynamic Accelerations

Many motions that incorporate dynamic acceleration were considered for the calibration of the accelerometer. The simplest involved accelerating the end-effector of the SR8408 SCARA along two straight lines in the directions of each of the x- and y-axes of the SCARA's base frame: the x_b- and y_b-axes. However, with maximum straight-line paths of 0.474 m possible along both the x_b- and y_b-axes, the accelerations and decelerations would necessarily be small and would have to occur over a very short time. Such transient accelerations would be difficult to measure accurately with the accelerometer.

Curved paths in the x_b-y_b plane provide longer paths, with arc lengths of 2.303 m possible. More importantly, the curved motion introduces a normal acceleration, a_n, in addition to the tangential acceleration seen in straight-line accelerations:

$$a_n = \omega^2 r, \tag{6.2}$$

where ω is the angular velocity, and r is the radius of the circle or partial circle being traversed. This normal acceleration is constant for constant angular velocities, resulting in accelerations that are much easier to measure over a longer period of time than the transient tangential accelerations. Such an approach is similar to that described in [224], where an accelerometer is placed on a turntable at a known radius from the turntable's center, and rotated at a known angular velocity over multiple periods. The cyclical motion of the turntable allows for an arbitrarily long sample of accelerometer measurements to be taken, which can be used to find the average normal acceleration. The advantage of this approach is clear, especially when dealing with accelerometer measurements that are, by their nature, noisy. Unfortunately, as can be determined from the operational range of the SR8408 SCARA, summarized in Table 3.1, the SCARA's base and elbow joints are not able to rotate in a complete circle; they can only rotate in a partial arc. The SCARA's roll joint can rotate through two complete circles, but since the roll joint is aligned with the end-effector, the resulting radius of those circles is zero. Even when taking into account any displacement of the origin of the accelerometer's frame from the center of the end-effector when the WIMU is mounted to the SCARA, it is clear that the radius of such a circle is too small to generate appreciable accelerations.

However, analysis of the kinematics of the SCARA reveals that a combination of simultaneous, co-operative joint movements can allow the end-effector to travel in a circle with a radius of up to 135.8 mm within the workspace of the SR8408 SCARA. Moreover, if the orientation of the end-effector and the attached accelerometer remains fixed relative to the base frame of the SCARA, this circle can be traversed as many times as desired, allowing for an arbitrarily large number of measurements to be taken by the accelerometer. Such an approach was deemed to be the most suitable for calibrating the WIMU's accelerometer, and is illustrated in Figure 6.1.

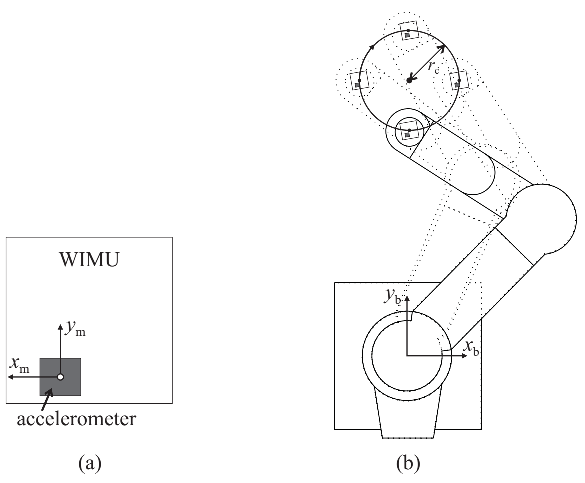


Figure 6.1 Top views of a) the WIMU and its accelerometer when mounted to the SCARA, and b) the circular path that can be traversed an unlimited number of times by the end-effector of the SCARA.

6.2.1 Two Circle-Based Calibration Routines

The properties of the calibration circle were chosen so as to maximize the normal accelerations seen by the accelerometer, while limiting vibrations and other sources of noise in the system. This was done because acceleration measurements at the higher end of the measurable range of the accelerometer are relatively less effected by the errors introduced by quantization and electrical noise. From Equation 6.2, it is clear that in order to maximize the normal acceleration, both the radius and the angular velocity should be maximized.

Although circles with a maximum radius of 135.8 mm are possible in the x_b-y_b plane of the SCARA's workspace – the plane of interest in this work due to the fact that the motion of the SCARA's end-effector in the x_b- and y_b-directions will benefit most from being tracked with an accelerometer – a radius of 75 mm was initially selected because it is the largest possible circle that can be traversed by the end-effector in all three planes of the workspace of the SR8408 SCARA. It was reasoned that this will allow the dual-axis accelerometer calibration routine to be extended to the x_b-z_b and y_b-z_b planes in the future. This selection turned out to be a fortuitous one, as explained further in Section 6.3.

The maximum tangential speed with which the SR8408 SCARA can move its end-effector in a planar circle is 1000 mm/s. For a circle of radius 75 mm, this corresponds to an angular velocity of 13.3 rad/s. However, it was found that the force of such high-speed motions cause the table on which the SCARA is bolted to shake violently, introducing unwanted vibrations and accelerations to the entire system. Instead, an angular velocity of 5.33 rad/s was used, as this was found to be the maximum angular velocity that did not cause the SCARA's workspace to visibly vibrate when the end-effector is moved around the calibration circle.

The nuances of how the circular motion adhering to the above properties was used to determine each of the accelerometer's calibration parameters are described, in turn, below.

Bias

The accelerometer's x- and y-biases, b_x and b_y respectively, are taken to be the mean resting values measured by each of the respective axes of the accelerometer, for 30 seconds before the calibration routine and 30 seconds after. Note that any tilt introduced by the mounting of the WIMU onto the end-effector will result in the inclusion of a small gravitational-based acceleration in these measurements, as explained in Section 3.2. Therefore, removing the resting mean bias will not only remove the sensor's intrinsic bias, but will also remove any gravitational effects specific to motion in that plane.

Sensitivity

Because the orientation of the accelerometer remains fixed while traveling around the calibration circle, the normal acceleration alternates between the x- and y-axes of the accelerometer. This produces measured acceleration signals that are sinusoidal, rather than the constant signals seen in turntable types of applications. Ideally, these sinusoidal signals will be a perfect sine and cosine wave:

$$\begin{bmatrix} a_x \\ a_y \end{bmatrix} = \omega^2 r \begin{bmatrix} \cos(\omega\tau) \\ -\sin(\omega\tau) \end{bmatrix}, \tag{6.3}$$

where a_x and a_y are the ideal accelerations along the x_b- and y_b-axes, respectively, and τ is time.

It should be noted that although the exact placement of the accelerometer's frame origin within the chip is unknown, it is known that the accelerometer chip is not centered on the WIMU, nor is it likely to be centered on the SCARA's end-effector. However, as illustrated in Figure 6.2, any offset of the accelerometer from the center of the end-effector only results in a corresponding

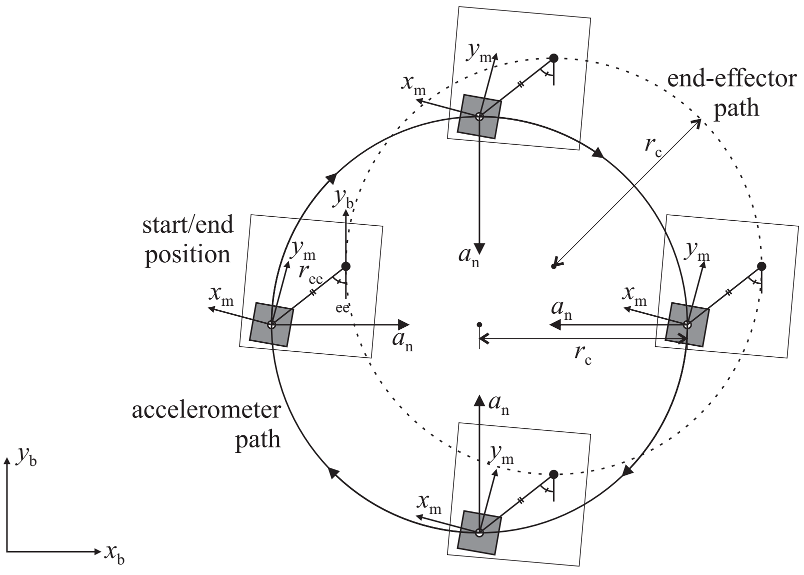


Figure 6.2 The WIMU, shown as a white box, and its accelerometer, shown as a shaded box, are mounted with an offset from the center of the end-effector. The path of the accelerometer is a shifted version of the end-effector's path, of identical size and shape, when the accelerometer maintains a fixed orientation.

shift in the circular path that is seen by the accelerometer; the size and shape of the circle are unaffected.

While the SCARA can repeat motions with a high precision, these motions will not be ideal. Although it may look to the naked eye that the end-effector is traveling smoothly around a perfect circle at a constant speed, the SCARA's controller is constantly adjusting the torque to the motor at each of the robot's joints according to the feedback received from its position encoders. In its efforts to follow the programmed circular path, the motion of the end-effector actually moves in many small, approximately linear sections that, when combined, approximate the reference circle. But when considering a given smaller interval of the circle, the path may deviate slightly, introducing a jitter to the radius and a variation in angular velocity. For this reason, Equation 6.3 can only be used to approximate the mean x- and y-accelerations over a long period of time. They should not be used directly to determine the sensitivity of the axes of the accelerometer.

Instead, the measured signals are integrated twice, and the resultant signals describing the position of the end-effector as it moves around the calibration circle are used:

$$\begin{bmatrix} \iint a_x d\tau d\tau \\ \iint a_y d\tau d\tau \end{bmatrix} = r \begin{bmatrix} -\cos(\omega\tau) \\ \sin(\omega\tau) \end{bmatrix} \tag{6.4}$$

As shown in Equation 6.4, the double integration effectively removes the squared effect that any variation in the angular velocity has on the magnitude of the signals. Although the magnitudes of the position signals are still dependent on the radius, the effect of small variations in the radius will be removed by using the mean peak position values from a large number of circles. It was determined that moving the accelerometer through 200 circles provides an adequately large data set, even if the WIMU drops some data packets, while still only taking four minutes to complete at the chosen angular velocity. If the non-idealities in Equation 6.1 are filtered out, that equation can be rearranged and integrated twice to derive the following relationship for the sensitivity parameter:

$$S_\lambda = \frac{\iint (a_{m,\lambda} - b_\lambda)}{\iint a_{s,\lambda}} = \frac{r_{m,\lambda}}{r_c}, \tag{6.5}$$

where $r_{m,\lambda}$ is the mean measured radius of the circular path in the direction of the λ-axis, obtained by double integrating the accelerometer's measurements after its bias has been removed, and r_c is the calibration circle's nominal radius of 75 mm. The sensitivity of each accelerometer axis is then simply:

$$S_x = \frac{-(\sum_{i=1}^{N_x} |x_{p,i}|)/N_x}{r_c} \tag{6.6}$$

and

$$S_y = \frac{(\sum_{i=1}^{N_y} |y_{p,i}|)/N_y}{r_c}, \tag{6.7}$$

where $x_{p,i}$ and $y_{p,i}$ are the i^{th} position peaks found by double integrating the measured accelerations, $a_{m,x}$ and $a_{m,y}$, respectively, and N_x and N_y are the number of position peaks considered for each axis. Note that S_x is multiplied by -1 to account for the x_m-axis of the accelerometer pointing in approximately the opposite direction of the x_b-axis, due to the physical orientation of the WIMU when attached to the end-effector of the SCARA via the mounting platform.

At this stage of the calibration procedure, enough data has been collected to partially calibrate the accelerometer, according to the linear model presented in Equation 6.1. This yields the properly scaled accelerations $a_{s,x}$ and $a_{s,y}$. Although the directions of $a_{s,x}$ and $a_{s,y}$ do not yet necessarily align with the x_b- and y_b-axes, depending on how the accelerometer is attached to the end-effector of the SCARA, these partially calibrated signals can be used to determine the absolute magnitude of a calibrated acceleration, a_c, in the x_b-y_b plane:

$$|a_c| = \sqrt{a_{s,x}^2 + a_{s,y}^2}. \tag{6.8}$$

Depending on the sensing application, the magnitude of acceleration in the x_b-y_b plane may provide adequate information without a description of the direction of the acceleration. For example, the radius of an arc can be determined from $|a_c|$, and can be used to find the link lengths of a robot arm, as described by [209–212].

The concise SCARA routine for partially calibrating a dual-axis accelerometer is summarized in Algorithm 6.1. This entire short calibration routine takes five minutes to complete.

Algorithm 6.1 Short Accelerometer Calibration Routine

```
1: The SCARA remains at rest in the start position of the calibration circle for 30 s.
2: The SCARA's end-effector is accelerated to the desired angular velocity, $\omega_0$, as it travels clockwise around one calibration circle; the orientation of the accelerometer remains fixed relative to the base frame.
3: The SCARA maintains a constant angular velocity, $\omega_0$, as it moves around 200 calibration circles, keeping the orientation of the accelerometer fixed.
4: The SCARA moves in one final calibration circle with the orientation of the accelerometer fixed, as the robot decelerates and comes to a stop.
5: The SCARA remains at rest in the start/end position of the calibration circle for 30 s.
```

Phase Offset

For applications that require knowledge of an acceleration's direction, a more complete calibration routine can be executed. Using the partially calibrated accelerometer signals, $a_{s,x}$ and $a_{s,y}$, the angle relating the scaled accelerometer frame to the direction of the the normal acceleration can be determined:

$$\gamma_a = \arctan(a_{s,y}/a_{s,x}), \tag{6.9}$$

where γ_a is the angle between the direction of the normal acceleration and the x_s-axis, as illustrated in Figure 6.3.

In order to create a measurable acceleration whose direction remains constant relative to the accelerometer's scaled axes, the SCARA rotates the accelerometer

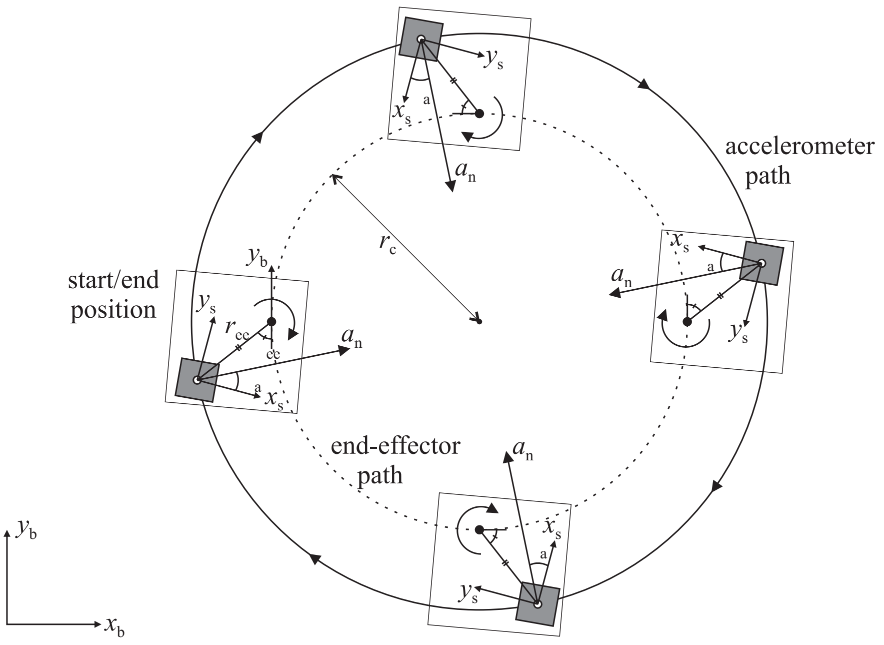


Figure 6.3 The circular path of the accelerometer will have a different radius to that of the end-effector when the accelerometer is rotated about the roll joint.

around its roll joint while traveling around the calibration circle. The direction and magnitude of the angular velocity of the roll joint rotation match that of the larger calibration circle during this motion. In this case, the accelerometer's offset from the center of the end-effector does make a difference to the radius of the circle, as shown in Figure 6.3. However, the relative proportion of acceleration along the x_s- and y_s-axes remains the same regardless of radius length, which is all that is required to determine γ_a.

Close examination of the geometry in Figure 6.3 shows that γ_a is not actually the phase offset required to align the accelerometer's scaled frame with the fixed base frame. Indeed, the angle γ_c, shown in Figure 6.4 is the phase offset, which is related to γ_a by:

$$\gamma_c = \gamma_a - \psi, \tag{6.10}$$

where ψ is the angle between the x_b-axis and the direction of normal acceleration.

The relationship between ψ and the offset between the origins of the frames of the accelerometer and the end-effector, represented by the distance r_{ee} and the angle φ_{ee}, can be found by examining the right triangles in Figure 6.4b:

$$\tan(\psi) = r_{ee}\cos(\varphi_{ee})/(r_c + r_{ee}\sin(\varphi_{ee})). \tag{6.11}$$

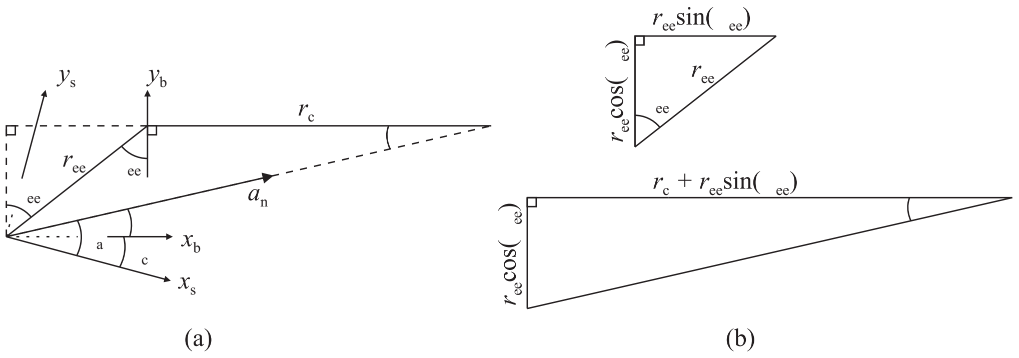


Figure 6.4 The geometry relating the frame of the partially calibrated accelerometer and the center of the end-effector to the direction of normal acceleration, when the starting roll position is at zero.

Although there are too many unknowns in Equations 6.10 and 6.11 to be solved with one γ_a phase measurement, multiple distinct measurements can be made if the starting roll position is changed before each phase measurement. This results in the geometry shown in Figure 6.5 for the j^{th} measurement, and the following general relation:

$$\gamma_{a,j} = \gamma_c + \Delta\tilde{\theta}_4 + \arctan(\frac{r_{ee}\cos(\varphi_{ee} + \Delta\tilde{\theta}_4)}{r_c + r_{ee}\sin(\varphi_{ee} + \Delta\tilde{\theta}_4)}). \tag{6.12}$$

If the starting roll joint is incremented by one degree between phase measurements, a set of 360 unique $\gamma_{a,j}$ can be measured and used to find the least squared error solution to Equation 6.12. This is reflected in Algorithm 6.2, which summarizes the complete SCARA routine for calibrating a dual-axis accelerometer. This extended routine takes 47.5 minutes to complete.

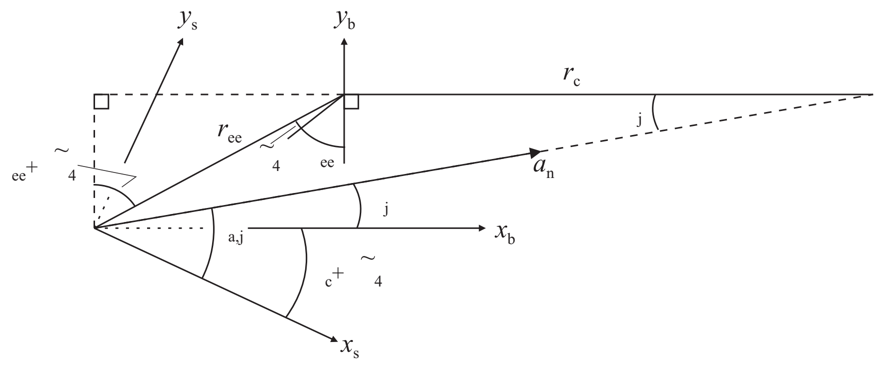


Figure 6.5 The geometry relating the frame of the partially calibrated accelerometer and the center of the end-effector to the direction of normal acceleration, when the starting roll position is offset by $\Delta\tilde{\theta}_4$.

Once γ_c is found, the accelerometer can be completely calibrated by combining the linear calibration relation, described in Equation 6.1, with a rotation around the z-axis, as given in Equation 4.3:

$$\begin{bmatrix} a_{c,x} \\ a_{c,y} \\ 1 \end{bmatrix} = \begin{bmatrix} S_x \cos(-\gamma_c) & -S_y \sin(-\gamma_c) & b_x \\ S_x \sin(-\gamma_c) & S_y \cos(-\gamma_c) & b_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} a_{m,x} \\ a_{m,y} \\ 1 \end{bmatrix}. \tag{6.13}$$

Algorithm 6.2 Extended Accelerometer Calibration Routine

```
1: The SCARA remains at rest in the start position of the calibration circle for 30 s.
2: The SCARA's end-effector is accelerated to the desired angular velocity, $\omega_0$, as it travels clockwise around one calibration circle; the orientation of the accelerometer remains fixed relative to the base frame.
3: The SCARA maintains a constant angular velocity, $\omega_0$, as it moves around 200 calibration circles, keeping the orientation of the accelerometer fixed.
4: The SCARA moves in one calibration circle at $\omega_0$, while also rotating the accelerometer clockwise with an angular velocity of $\omega_0$.
5: The SCARA moves in two calibration circles at $\omega_0$, again keeping the orientation of the accelerometer fixed.
6: The SCARA moves in one calibration circle at $\omega_0$, while also rotating the accelerometer counterclockwise by 359 degrees.
7: Step 5 is repeated.
8: Steps 4-7 are repeated 359 times.
9: The SCARA moves in one final calibration circle with the orientation of the accelerometer fixed, as it decelerates and comes to a stop.
10: The SCARA remains at rest in the start/end position of the calibration circle for 30 s.
```

The main features of the two circle-based calibration routines are summarized in Table 6.1, and compared to other prominent accelerometer calibration routines that are described in the literature. It is clear from this comparison that the short circle-based calibration routine is the only routine that does not require a change in the orientation of the accelerometer. This makes it particularly suitable for calibrating accelerometers in-place on low DOF robots or in other mechanical systems in which rotary motion is limited. The extended circle-based calibration routine requires that the accelerometer is rotated about only one axis, and unlike routines that employ static acceleration alone, that rotation axis can be parallel to the direction of gravity.

Table 6.1 Comparison of Accelerometer Calibration Routines

	Manually rotate [218–221]	Anthropomorphic robot used [207, 208, 222, 223]	Turntable used [224]	Short circle-based routine	Extended circle-based routine
Measured acceleration:					
(a) static	●	●			
(b) dynamic			●	●	●
Requires rotation about axis that is	perpendicular to gravity	perpendicular to gravity	any direction	accelerometer orientation is unchanged	any direction
Number of accelerometer axes calibrated	3	up to 3	1	2*	2*
Performed in-place		●		●	●
Robot DOF	-	6	-	4	4

* The extension to three axes is theoretically possible if circular motion in a second plane is measured. However, as this is outside the scope of this thesis, it is instead left for future work.

6.3 Non-Ideal Kinematics Considerations

6.3.1 Non-Ideal Link Twists

The nominal D-H model for the Sankyo SR8408 SCARA, as specified in Figure 6.6 and Table 6.2, dictates that the axes of the three revolute joints are all parallel to one another. However, it is unlikely that the joints of an actual SR8408 SCARA, such as that utilized in this work, are perfectly aligned. This is due to the factors listed in Chapter 1, as well as the effect of the physical weight of the links acting to pull the elbow and roll joints out of alignment over time.

The deviations from the ideal model are expected to be small, and not necessarily directly observable. However, imperfect alignment can be detected from accelerometer measurements, even before the accelerometer is calibrated. As described in [209], the direction of a revolute joint's axis relative to that of

Table 6.2 D-H Parameters of SR8408 SCARA

Link	$\tilde{\theta}$ (deg)	$\tilde{d}$ (mm)	$\tilde{\alpha}$ (deg)	$\tilde{a}$ (mm)
1	q_1=0	310	180	300
2	q_2=0	0	0	250
3	0	q_3=75	0	0
4	q_4=0	0	0	0

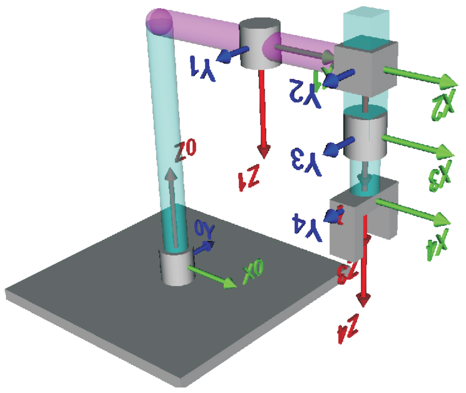
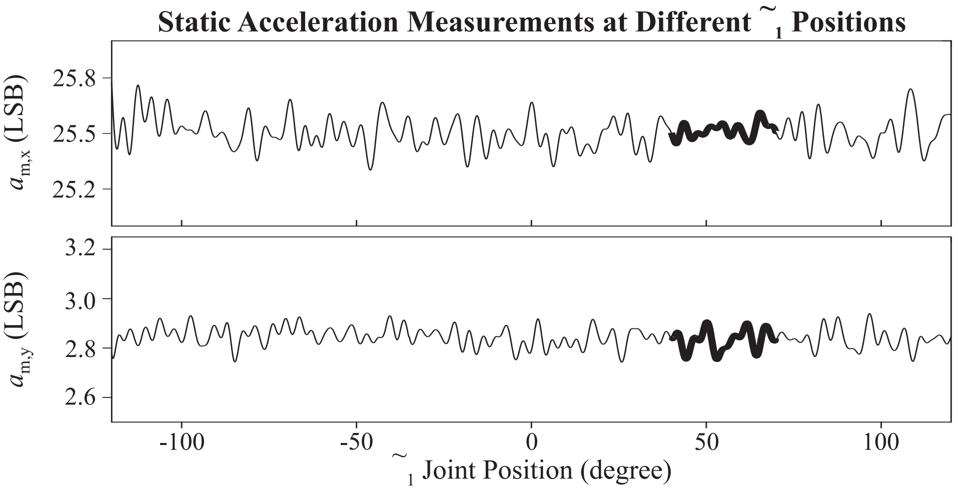


Figure 6.6 D-H model of the SR8408 SCARA.

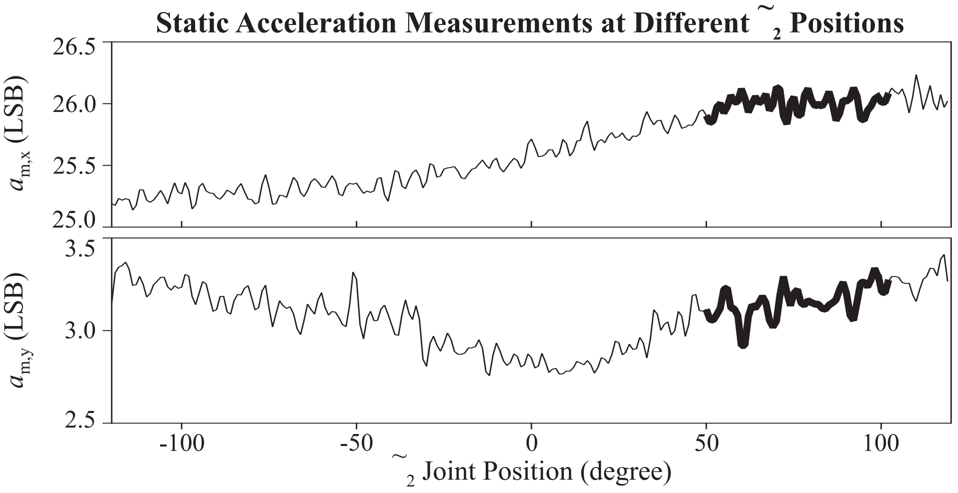
gravity can be determined by rotating just that joint by small intervals, and measuring the static acceleration when the robot is at rest between rotations. This procedure was used to measure the static acceleration at one degree intervals for the entire range of the base and elbow joints, and for 360 degrees around the roll joint. The mean results of these uncalibrated measurements are shown in Figure 6.7, from which a number of qualitative conclusions can be made.

The x_m- and y_m-static accelerations measured when the arm is rotated about the base joint show very little variation, indicating that the axis of that joint is aligned with gravity. This validates the leveling process that the SCARA underwent before beginning the experiments discussed in this chapter. Conversely, the x_m- and y_m-static accelerations appear sinusoidal when the arm is rotated about the elbow and roll joints, in turn, indicating that these joint axes are not aligned with gravity. Therefore, these joints are clearly not aligned with the base joint.

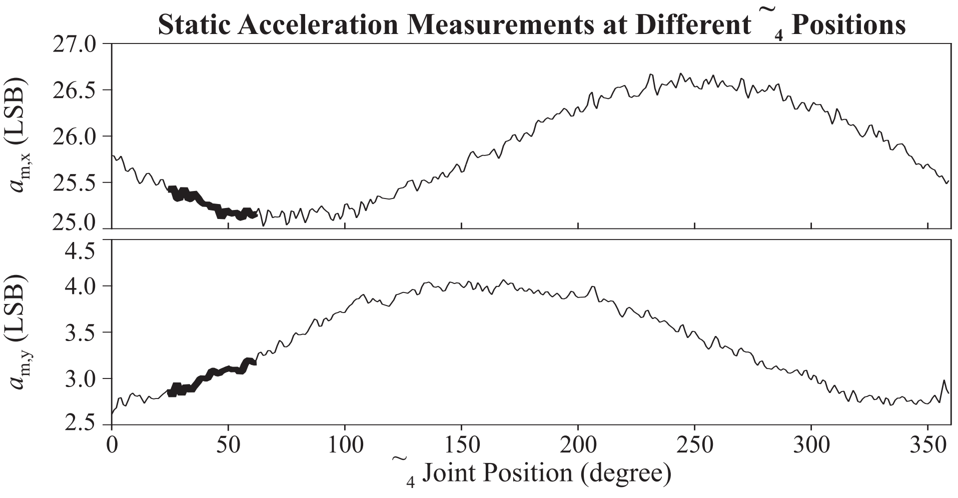
The consequences of the joint misalignment to the calibration routines must be considered. Rotations about the misaligned elbow and/or roll joint(s) result in a change in the tilt of the WIMU, thus changing the effective plane in which $a_{m,x}$ and $a_{m,y}$ are being measured. This can be thought of as introducing a rotation about the x_m- and/or y_m-axes of the accelerometer. It is clear from Figure 6.7 that such a rotation has an effect on the resting bias parameter, as explained in Section 6.2. Applying the rotation matrices in Equations 4.1 and 4.2 to the calibration matrix described in Equation 6.13 reveals that a change in tilt will also have an effect on the apparent sensitivity parameters. Thus, in order to minimize the error in the measured calibration parameters,



(a)



(b)



(c)

Figure 6.7 Uncalibrated static acceleration measurements made when the accelerometer's position is changed relative to a) the base joint, b) the elbow joint, and c) the roll joint. The ranges of joint positions used in the calibration routines are shown in bold.

the calibration routine must attempt to minimize motion about the misaligned joints.

In the short routine, the WIMU is held in a fixed orientation with respect to the base frame as it is moved around the calibration circle. In order to accomplish this motion, all three revolute joints must move in co-operation. The range of motion that is required of each of the joints is highlighted in Figure 6.7. As can be observed, the deviation in static accelerations within each of these ranges is small, keeping the change in accelerometer tilt small. This, in turn, minimizes the error introduced to the sensitivity parameters. A larger calibration circle would require motion over larger ranges of all three joints, thereby introducing larger errors to the sensitivity parameters. A smaller calibration circle, on the other hand, would require motion over slightly smaller ranges of the joints, resulting in the introduction of smaller errors to the sensitivity parameters. However, the advantage of any reductions in absolute error in the acceleration measurements is negated by the increase in relative errors introduced by the measurement of low accelerations. Thus, using a calibration circle with a radius of 75 mm is a good design compromise.

In the extended routine, the roll joint must rotate over a much larger range than in the short routine. This introduces a small sinusoidal component to the ideally constant x_m- and y_m-accelerations. Although this introduces a small error to the measured phase offset parameter, this error can be minimized by using large normal accelerations in the calibration circle. This is because the absolute deviation in the static acceleration component remains the same regardless of the dynamic acceleration applied, so the relative impact of varying the accelerometer tilt will be less when larger dynamic accelerations are used. Thus, using a large angular velocity – the largest that does not introduce visible workstation vibrations – helps to minimize the error in the measured phase offset parameter.

6.3.2 Non-Ideal Link Lengths

The physical link lengths of the SCARA may also deviate from those specified in Table 6.2. When compounded with the fact that compliance in the joints may make the apparent link lengths vary, depending on the motion of the robot, it is clear that the path of the end-effector will be affected. As shown in Figure 6.8, deviations in the link lengths result in the shifting of the end-effector's path, as well as a deformation from that of the perfect circle that the SCARA "thinks"

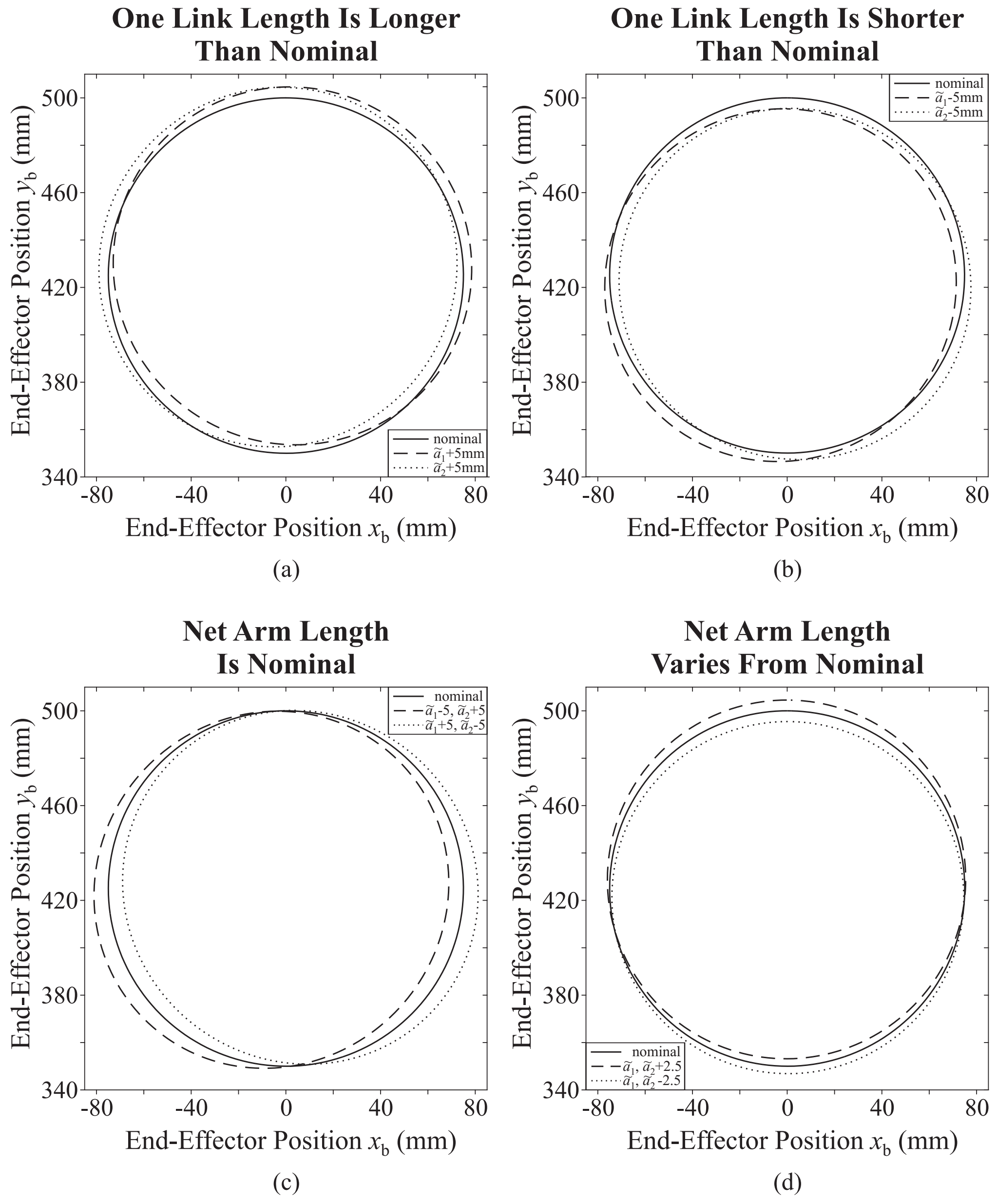


Figure 6.8 Deviations in the end-effector's path for different scenarios of non-ideal link lengths.

that it is moving around, based on the nominal D-H model. In fact, the paths become more elliptical in shape.

This deviation in the shape of the path results in the introduction of non-zero angular accelerations and corresponding changes in the angular velocity, so that the simple relationship described in Equation 6.2 no longer exists. However, by double integrating the accelerometer measurements, the error introduced by such deviations is limited to that of the error in the mean radius of the elliptical path, or equivalently, to the error in the radius of the best-fit circle of the elliptical path.

The best-fit ellipses [225] were found for the end-effector paths that result from the example link length deviations shown in Figure 6.8. These example deviations were selected as an upper bound on the true deviations expected to exist for the physical SCARA, as confirmed by the manual coarse measurements of the link lengths, performed with a ruler. The semi-major and semi-minor radii for the best-fit ellipses are listed in Table 6.3, along with the error in the best-fit circle radii. As can be seen, even in the worst case situation where the net arm length of the SCARA deviates from its nominal length by 5 mm, the mean radius of the calibration path only deviates from the nominal 75 mm radius by 1%. Thus, the maximum error which is expected to be introduced to the calibration sensitivities by non-ideal link lengths is limited to 1%.

Table 6.3 Best-Fit Elliptical Paths for Sample Link Length Deviations

$\Delta\tilde{a}_1$ (mm)	$\Delta\tilde{a}_2$ (mm)	Semi-major Radius (mm)	Semi-minor Radius (mm)	Radius Error
+5	0	76.1	75.1	0.83%
0	+5	76.3	75.2	1.00%
-5	0	75.0	73.7	0.83%
0	-5	75.0	73.5	1.00%
-5	+5	76.2	74.0	0.18%
+5	-5	75.9	73.8	0.15%
+2.5	+2.5	75.7	75.6	0.91%
-2.5	-2.5	74.4	74.2	0.92%

6.4 Experimental Results and Analysis

To evaluate the calibration routines described by Algorithms 6.1 and 6.2, experimental data was collected with Tyndall's WIMU attached to the end-effector of the SR8408 SCARA. The WIMU was configured to collect data from both the accelerometer and the gyroscope at a sampling rate of 355 Hz, the maximum sampling rate allowed by the WIMU in that configuration. The smallest measurement ranges were selected for both sensors: $\pm 2g$ for the accelerometer, and ± 440 deg/s for the gyroscope. Although these experiments focus on the calibration of the accelerometer rather than the gyroscope, it was observed that different WIMU configurations – for example, the sampling frequency used and the sensor combination selected – changed the measured calibration parameters. Therefore, to allow for future applications which may involve tracking the motion of the SCARA based on information provided by the calibrated accelerometer and gyroscope, both sensors were activated during these tests.

6.4.1 Short Calibration Routine

The validity of the short calibration routine was evaluated by analyzing the data collected according to Algorithm 6.1 in MATLAB. A sample portion of that data from one representative test run is plotted in Figure 6.9, along with the corresponding lowpass filtered signals. The filtered acceleration signals show a clear sinusoidal trend when the WIMU is moved around the calibration circle with a fixed orientation relative to the base frame, as expected.

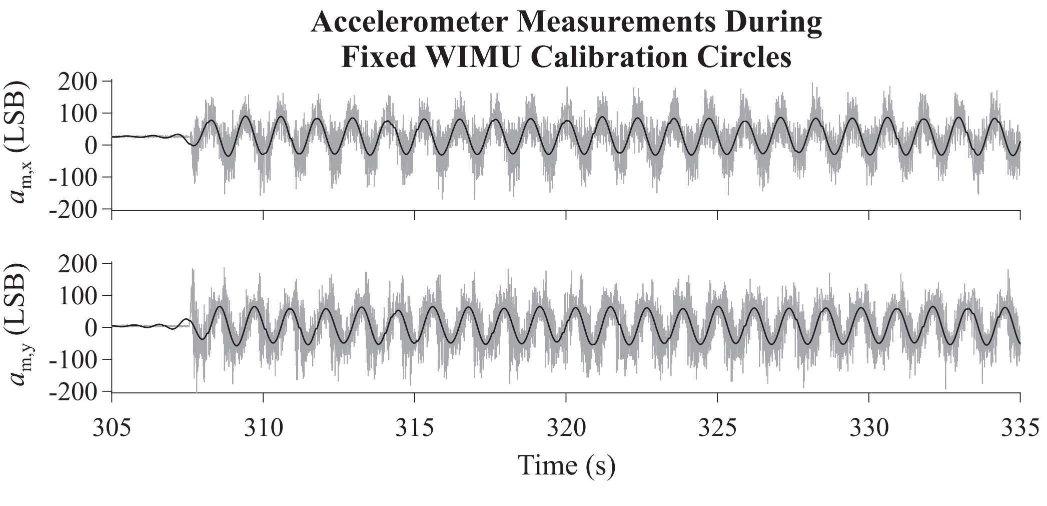


Figure 6.9 Sample raw accelerometer data collected during the short calibration routine, along with the corresponding lowpass filtered signals.

The bias of each accelerometer axis was determined by finding the mean raw acceleration measurement during the intervals of rest at the beginning and end of the calibration routine. The bias parameters from the test run shown in Figure 6.9 are included in Table 6.4.

The sensitivity of each accelerometer axis was determined by first applying a finely tuned, bandpass filter to the raw acceleration measurements. This bandpass filter was created using MATLAB's *designfilt* function, which automatically designs a filter of the smallest order required to meet the specifications entered. In this case, an infinite impulse response (IIR) Butterworth filter was designed, with lower stopband and passband frequencies of 0.5 Hz and 0.55 Hz, respectively, and upper passband and stopband frequencies of 1.2 Hz and 1.5 Hz, respectively. Zero-phase filtering was achieved by using MATLAB's *filtfilt* function. Rather than applying a lowpass filter to the bias-corrected measurements, this bandpass filter was applied to the raw accelerometer measurements to remove any additional low frequency noise caused by system non-idealities.

The filtered acceleration signals were integrated to obtain the corresponding velocity signals. A highpass filter was applied to each resultant velocity signal, thereby removing any drift caused by the apparent jumps in the acceleration signal due to intervals of missing data. This highpass filter was an IIR Butterworth filter, again designed using MATLAB's *designfilt* function, with stopband and passband frequencies of 0.5 Hz and 0.55 Hz, respectively, to match the characteristics of the bandpass filter. It was possible to use a simple highpass filter to correct for integration drift, rather than a predictive Kalman filter that would require information from additional sources, because the motion of the end-effector was known to be cyclical.

A subset of the data relating to the filtered velocity signal was then integrated to obtain the corresponding position signals. The smaller subset was chosen to exclude ringing at the beginning and end of the velocity signals, caused by filtering. The highpass filter was applied again to correct for integration drift caused by missing data points. An interval of position peaks in the middle of the resultant position data set was selected, avoiding large sections of missing data, and the mean absolute peak value was determined. This information was used to find the sensitivity parameter of each axis of the accelerometer, as determined by Equations 6.6 and 6.7. The sensitivity parameters for the test run illustrated in Figure 6.9 are included in Table 6.4.

Table 6.4 Measured Accelerometer Calibration Parameters

Calibration Routine	b_x (LSB)	b_y (LSB)	S_x (LSB/(m·s^{-2}))	S_y (LSB/(m·s^{-2}))
Short	26.48	3.97	26.62	26.83
Extended	26.53	3.84	26.59	26.92

6.4.2 Validation of Short Calibration Routine

Additional data was collected according to Algorithm 6.1, but for circles of different radii. This data was collected during the same test run in which the calibrating data shown in Figure 6.9 was collected, without adjusting or powering off the WIMU between data collection routines. For simpler comparison during the analysis, the same angular velocity was used for each circle. Seven sets of data were collected, corresponding to seven different circle sizes.

The Lomb-Scargle method [226, 227] was used to determine the fundamental frequency, and thus the fundamental angular velocity, of the measured accelerations for each data set via MATLAB's *plomb* function. This method is appropriate for use with non-uniformly spaced data, as is the case when data points are dropped by the WIMU. Due to computer memory restrictions, a smaller interval of data had to be selected for each data set for analysis. These intervals were chosen so as to exclude large sections of missing data, but were otherwise chosen from approximately the middle of the measured acceleration signals. In order to ensure an accurate comparison between all of the data sets, only intervals yielding a fundamental frequency within 0.0006 Hz of the nominal 0.8488 Hz were used.

Using the parameters from the short calibration routine in Table 6.4 to partially calibrate all seven sets of data, the magnitudes of acceleration during the chosen intervals were found according to Equation 6.8. This information was used in conjunction with the measured angular velocities to determine the mean radius of each of the seven circles, as estimated by Equation 6.2. The difference between the mean measured radius and its corresponding nominal radius is plotted in Figure 6.10. As can be seen, the absolute error in the mean measured radius is 0.5 mm, or less, for each of the seven circles. This corresponds to errors of between 0.1% to 1.4%, indicating that accurate bias and sensitivity parameters were used to calibrate the accelerometer. Other

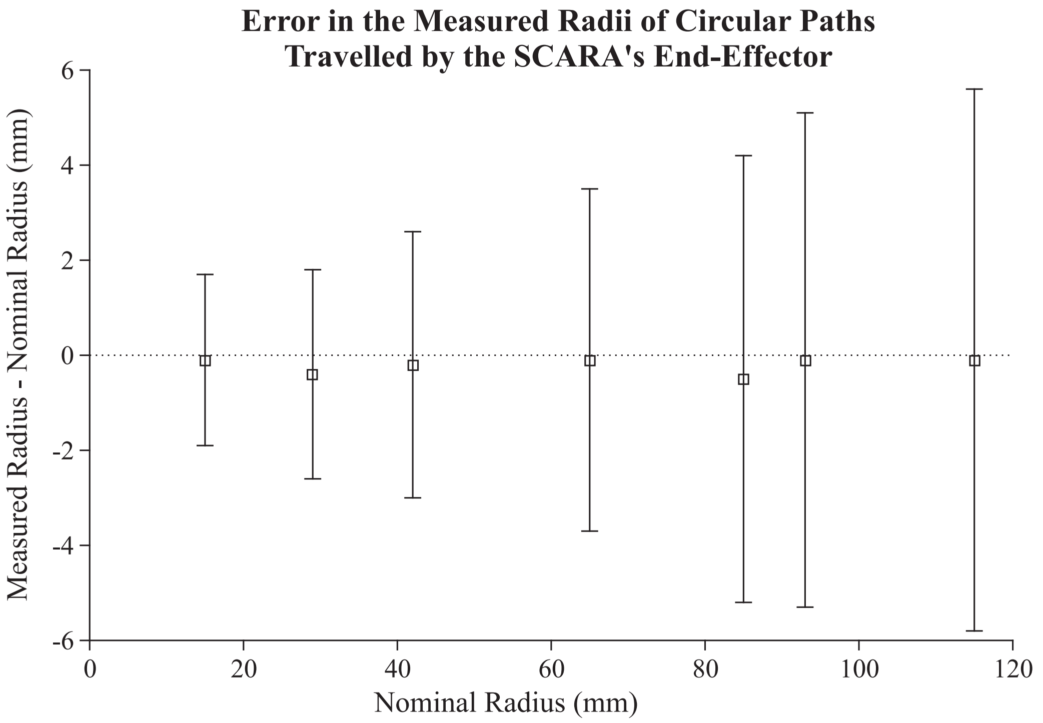


Figure 6.10 Error in the radii, as measured by the accelerometer, of different circular paths traveled by the end-effector. The mean error is shown as a square, and one standard deviation is shown as an error bar.

comparable studies report errors of between 3.3% and 11% when using an accelerometer to measure the radii of a robot's arc motion [209, 210, 212]. The improved performance in this study is likely due to the larger number of acceleration measurements permitted by traversing multiple complete circles, and also because the accelerometer was calibrated in-place rather than prior to its attachment to the robot.

It should be noted that similar absolute levels of accuracy were found for circles larger and smaller than the calibration circle, which required motions over larger and smaller joint ranges. This indicates that any errors introduced by rotating around the misaligned joint axes were inconsequential for the circular paths tested.

One standard deviation of each measured radius is also shown in Figure 6.10, as error bars. These deviation measurements clearly increase with the increase in the nominal radius of the circular path. The most likely reason for this is that the SCARA itself varied more in its positioning accuracy as the circle radius and tangential speed of its end-effector increased. This demonstrates the limited acceleration with which the SCARA can reliably move its end-effector before significant errors are introduced. The largest acceleration measured was for the circle with the largest radius, and even that was only 3.3 m/s^2; a relatively small acceleration in the $\pm 2g$ accelerometer measurement range. The accelerometer calibration parameters found using the SR8408 SCARA may be more precise if an accelerometer with a lower measurement range is used instead of the ADXL345 accelerometer that is on the WIMU under test.

6.4.3 Extended Calibration Routine

After validating the accuracy of the short calibration routine, new data was collected according to Algorithm 6.2. The biases and sensitivities were determined as described for the short calibration routine, and the measured parameters for one representative test run are included in Table 6.4.

Using these parameters, the accelerometer measurements for the example test run were partially calibrated, and the phase between the resulting $a_{s,x}$ and $a_{s,y}$ signals was determined according to Equation 6.9. As expected, during the intervals when the WIMU is rotated with the same angular velocity and direction as the calibration circle, the phase is relatively constant, as shown in Figure 6.11. A small ripple does exist, which could be the result of the rotation about the misaligned roll joint. It could also be the result of the filter trying to smooth out any small variations in the measurements of the accelerometer due to other system non-idealities. Regardless of the cause(s), the mean phase can easily be determined by considering data within the interval marked in Figure 6.11.

This approach was applied to find the mean phase, $\gamma_{a,j}$, for 355 starting roll positions, the results of which are plotted in Figure 6.12. Key data pertaining to the missing five $\gamma_{a,j}$ was either dropped completely by the WIMU or given a wrong timestamp, so that the data was compromised and could not be used. Even with these missing phase measurements, a best-fit curve could be found that matched the form of Equation 6.12, with a root mean squared error of 2.07 degrees. From this curve, it was determined that r_{ee} = 11.9 mm, φ_{ee} = 0.34 degrees, and γ_c = -0.015 degrees.

To determine the relative accuracy of these values, the physical layout of the WIMU is considered. When mounted so that the WIMU is approximately centered on the SCARA's end-effector, the accelerometer is located just to the left of the center of the WIMU, along its back edge, as depicted in the exaggerated illustration in Figure 6.3. Thus, it is reasonable to assume that

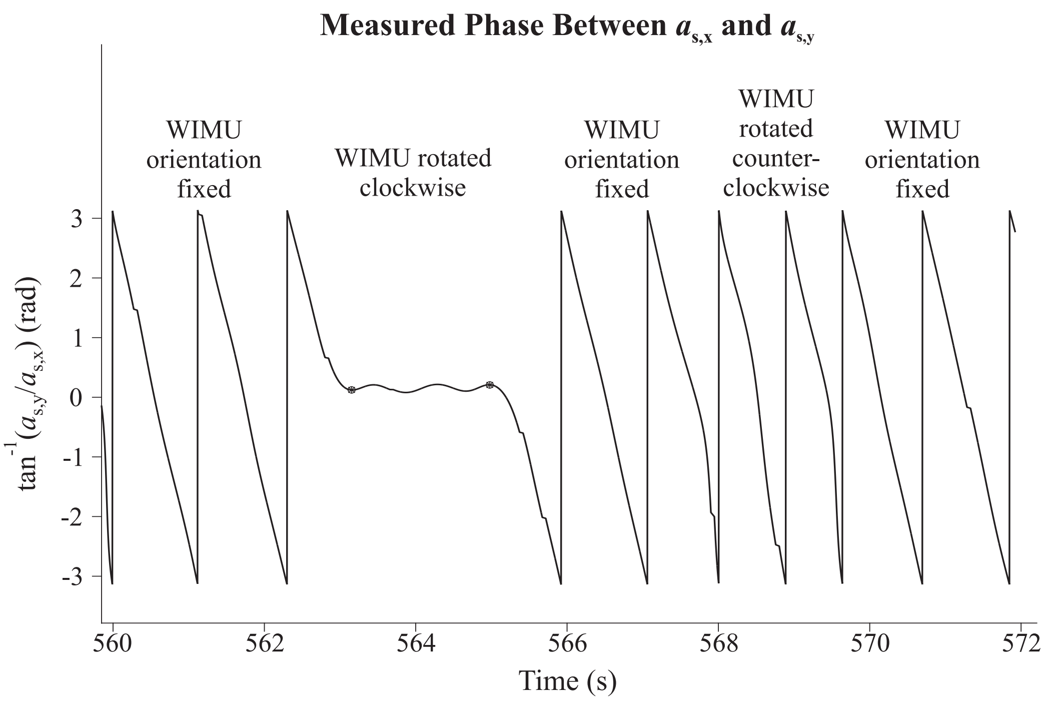


Figure 6.11 Sample phase plot with the different phases of the extended calibration routine labeled. Note that both WIMU rotations occur over 2 calibration circles instead of the 1 described in Algorithm 6.2, for improved plot clarity. This can be performed for $\Delta\tilde{\theta}_4$=0 only.

the distance between the origin of the accelerometer's frame and the center of the end-effector should approximate to just less than half the width of the circuit board, and that the angle relating that distance vector to the y_b-axis should be small. Since the WIMU's inertial sensor circuit board measures 25 mm by 25 mm, it is reasonable to estimate that r_{ee} should be slightly less than 12.5 mm. Such reasoning provides a qualitative validation of both the measured r_{ee} and φ_{ee} values. Furthermore, efforts were made to approximately align the accelerometer's y_m-axis with the y_b-axis, so that a γ_c that is close to zero was expected.

6.4.4 Shortening the Extended Calibration Routine

The main disadvantage of the extended calibration routine, as described by Algorithm 6.2, is the length of time that it takes to complete. The 47.5 minute accelerometer calibration routine accounts for approximately one fifth of the WIMU's four hour usable battery life, leaving just over three hours to use the calibrated accelerometer to track the motion of the SCARA in some way.

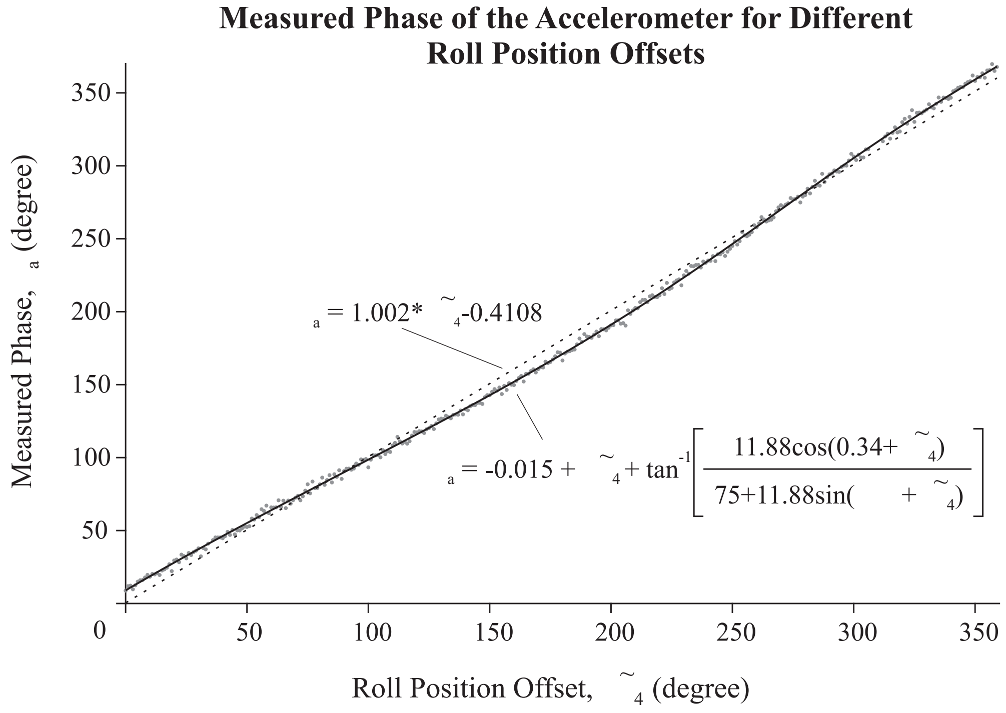


Figure 6.12 The measured angle between the x_s-axis and the normal acceleration, for different starting roll position offsets. Note the data's deviation from the best-fit linear relation.

Once the battery power is depleted, the WIMU must be removed from the SCARA to be recharged, and remounted to the SCARA for further motion-tracking applications. However, before the accelerometer can be used for such applications, it must be recalibrated in its new mounting position.

To determine whether the extended calibration routine can be shortened, the $\gamma_{a,j}$ data set was divided into two subsets: one including every other data point, and the second including the remaining offset data points. Curves adhering to Equation 6.12 were fit to both data subsets independently, and values for $r_{ee,sub}$, $\varphi_{ee,sub}$, and $\gamma_{c,sub}$ were found for each. These results are summarized in Table 6.5, along with the absolute difference of each parameter from the values found from the original data set. All of the measured parameters for each of the two separate data subsets are within an acceptable deviation from the parameters found when utilizing the complete data set, indicating that the number of data points gathered by the extended calibration routine can be cut in half without a significant loss in accuracy. This brings the duration of the extended calibration routine down to 26.2 minutes, which is slightly less than a comparable 27.74 minute calibration routine that utilizes static acceleration measurements and a six DOF anthropomorphic robot [222].

Table 6.5 Extended Calibration Parameters Found from Smaller Data Subsets

Data Subset	$r_{ee,sub}$ (mm)	$\lvert r_{ee,sub} - r_{ee}\rvert$ (mm)	$\varphi_{ee,sub}$ (deg)	$\lvert\varphi_{ee,sub} - \varphi_{ee}\rvert$ (deg)	$\gamma_{c,sub}$ (deg)	$\lvert\gamma_{c,sub} - \gamma_c\rvert$ (deg)
1	11.83	0.05	0.85	0.5	-0.037	0.02
2	11.93	0.05	-0.17	0.5	0.0070	0.02

6.5 Conclusions

Two novel in-place accelerometer calibration routines have been presented which utilize the dynamic accelerations of a SCARA. The first is a short routine that yields correct accelerometer magnitudes in the x_b-y_b plane; the second is an extended routine that also corrects for any phase misalignment about the z_b-axis that may have been introduced while attaching the accelerometer to the robot. The short routine can be completed in a mere five minutes, while the extended routine can be completed in 26.2 minutes. The duration of the extended routine is slightly less than the duration of a comparable accelerometer calibration method that employs a six DOF robot arm and static accelerations.

Both of the new calibration routines described in this chapter have been validated using experimental data, with relatively small errors found in the measured radii of the different curved paths traversed by the end-effector. This indicates the benefit of calibrating the accelerometer in-place, rather than prior to attaching it to the robot, before using it in applications that track the motion of the robot in some way.

The properties of the calibration path were carefully selected so as to minimize the errors introduced by non-ideal kinematic parameters. Based on experimental results, it can be concluded that the effects of such non-idealities were indeed minimal.

The advantage of using dynamic accelerations to calibrate an accelerometer is clear when using a low DOF system that cannot change the orientation of the accelerometer relative to the direction of gravity. Similar accelerometer calibration routines may also be of benefit in micro-g environments, like those encountered in space, where gravity cannot be used as a reference acceleration.

Chapter 7

Conclusions and Future Work

Through its examination of the geometry of the forms and motions of serial robot arms, this thesis has considered the practicalities of visualizing and applying the principles of robot kinematics. It has described how desktop virtual reality tools can be used to effectively aid students in the visualization of 3D rotations and translations, and the subsequent improvement in understanding of fundamental concepts in robot kinematics that results. This thesis has also described the practical application of the principles of robot kinematics in the design of end-effector motions that employ dynamic accelerations, which can be used to calibrate an attached dual-axis accelerometer. The errors introduced to this motion were minimized by considering how the true kinematics of the robot deviate from the nominal kinematic model specified by the manufacturer. The objective of this final chapter is to summarize and highlight the novel contributions described in this work, and to identify potential areas of future related work.

7.1 Contributions of This Work

7.1.1 Rotation Tool

After providing a comprehensive background of relevant information in the fields of kinematics, robotics, virtual reality, and inertial sensors, this thesis investigated the use of the virtual reality based Rotation Tool to aid university-level students with the visualization of compound rotations.

The Rotation Tool provides novel advantages in its user interface over comparable tools, including:

- the freedom to select the angles *and* axes describing compound rotations;
- the freedom to independently execute different rotation sets for the two compound rotation types; and
- the ability to view the forward and backward motion of a given rotation without limitation.

Such novel features ensure that students are able to explore and understand all of the fundamental principles of compound rotations, rather than just focusing on one particular principle.

One of the most important findings of this study was that the incorrect intuitive belief that compound rotations are commutative persists for a large number of university-level students. Such a finding is in direct opposition to the assumption made by some educators that students at higher levels of learning, particularly in engineering, either have a good existing understanding of the non-commutative nature of compound rotations, or that it is a principle that can be taught quickly and understood easily.

7.1.2 Build-A-Robot

Having established the usefulness of desktop virtual reality tools in the visualization of 3D rotations, this thesis expanded the scope of its investigation to examine the impact that Build-A-Robot had on students' understanding of the D-H convention and its parameters.

Build-A-Robot provides novel advantages in its user interface over comparable tools, including:

- the freedom to create a robot with any of the 254 possible permutations of joint combinations that are automatically incorporated into the tool's single virtual world;
- the freedom to modify the D-H parameters in real-time, after the robot is built; and

- the freedom to change the pose of the robot directly in the virtual scene.

Build-A-Robot has also advanced the state-of-the-art of virtual reality based educational tools in general with its novel manipulation of the geometry of virtual objects. Rather than being limited by the knock-on effects to child nodes that occur when a parent node is scaled in VRML, Build-A-Robot is able to directly manipulate the value of *any* field in any named node, allowing the geometric dimensions of parent and child nodes to be set independently from one another. These dimensions can also be set to zero, instead of just very close to zero, allowing for the use of a single generic virtual world file, instead of having to create unique world files that are dependent on the number and type of robot joints to be visualized.

Build-A-Robot improved students' confidence in their understanding of what the D-H parameters physically represent, as evidenced by enthusiastic student feedback. This increase in confidence and understanding was particularly evident amongst students who felt that they struggled to understand the principles of forward kinematics when taught using more traditional methods. The increase in understanding was also reflected in improved student performance when answering relevant final examination questions.

7.1.3 In-Place Accelerometer Calibration Routines

Finally, the principles of robot kinematics that were described and visualized in Chapters 4 and 5 were applied in planning the motion of a real SCARA in Chapter 6. Keeping in mind the non-idealities that were likely to be present in the true kinematic model of the SCARA, two novel calibration routines were designed to calibrate a dual-axis accelerometer that was attached to the SCARA's end-effector:

- the short calibration routine; and
- the extended calibration routine.

Whereas most accelerometer calibration routines typically rely on the static acceleration of gravity, both of the circle-based calibration routines presented in this work also employ dynamic accelerations. The main benefit of the short calibration routine is that it does not require the orientation of the accelerometer to change, making it particularly suitable for the in-place calibration of accelerometers attached to low DOF robots, such as the SCARA. On the other hand, the extended calibration routine does require that the accelerometer's orientation be changed about one axis. However, unlike calibration routines that rely on static acceleration alone, that rotation axis can be parallel to the direction of gravity. Therefore, both calibration routines are appropriate for calibrating an accelerometer in-place when the accelerometer's orientation cannot be adjusted in a way that enables the conventional static acceleration calibration approach to be applied, or when it is used in micro-g environments where gravity cannot be used as a reference acceleration.

The short calibration routine, requiring a mere five minutes to complete, was used to experimentally determine the accelerometer's biases and sensitivities that, in turn, could be used to partially calibrate the accelerometer. Such an approach is suitable for applications that only require measurements of the acceleration's magnitude, but not its direction. The short calibration routine was validated by using the partially calibrated accelerometer to determine the radii of different circles, with all of the measured radii being within 0.5 mm of the nominal radii.

The extended calibration routine was used to determine the phase offset in the x-y plane of the accelerometer's frame relative to the base frame, in addition to the accelerometer's biases and sensitivities. Such an approach allows for the direction *and* magnitude of measured accelerations to be determined. The extended calibration routine also provided information regarding the positional offset of the accelerometer's frame origin relative to the center of the end-effector that, in the absence of precise nominal values to compare it with, was determined to be at least approximately correct. The extended calibration routine requires 26.2 minutes to complete when 180 offset phases are measured, a time that is slightly less than a notable comparable calibration routine employing a robot arm and static acceleration measurements.

7.2 Future Work

The work in this thesis involved research in a number of different fields, and as such, myriad options exist to expand upon the findings that have been presented here. This thesis concludes with a discussion of some of the more relevant ideas for future work.

7.2.1 Robots and Virtual Reality

As discussed in Chapter 3, new virtual reality software options have emerged in recent years that provide viable alternatives to using VRML in educational applications. As VRML's successor, X3D offers an extended range of capabilities that may be useful, depending on the nature of the application. Through the diligent work of the Web3D Consortium, it also seems to have finally gained acceptance amongst commercial software product developers as the new standard, as exemplified by the fact that it is now supported by the MATLAB Simulink 3D Animation Toolbox. Going forward, it is anticipated that many software programs will transition from VRML to X3D, thus making it prudent to develop any future educational desktop virtual reality tools in X3D. Future work could investigate the mechanism through which external programs – such as those created in MATLAB – interface with X3D files, to determine if the same flexibility offered by Build-A-Robot can be recreated in X3D-based virtual worlds. If such flexibility is not offered by existing open-source or commercial products, a custom X3D browser that supports a custom mechanism for interacting with external applications can be built.

WebGL also offers enormous potential as a software option in which to develop virtual worlds, and the fact that its applications can be run in browsers without the need for a plug-in means that WebGL-based educational tools could be accessed by students from all over the world, without any extra software requirements. Although learning how to create virtual worlds in WebGL presents a steeper learning curve than learning how to use VRML or X3D, the potential in the extension of applications and control possible with WebGL could justify the extra effort required.

In terms of applications, university-level courses in parallel robotics may benefit from the introduction of virtual reality based tools to visualize the kinematics of closed-chain robot arms. To this end, it may be beneficial to modify Build-A-Robot's cylindrical representation of revolute joints to avoid any possible confusion with the higher DOF cylindrical joint. The addition of an extended lip to the top and bottom of the cylinder that currently represents the revolute joint, similar to that shown in Figure 1.5, may be suitable. Alternatively, future work could focus on the extension of Build-A-Robot to create a general specification to fully model the kinematics and dynamics of robots in virtual reality, similar to what the Humanoid Animation specification [228] accomplishes in defining how humanoid figures should be modeled in VRML and X3D. Such a specification would provide a framework for developing robots for educational visualization tools, as well as for designing and analyzing specific properties of robot arms in robotics research. Most importantly, it would enable the simple exchange of robot models between educational institutions and research groups.

7.2.2 Robots and Inertial Sensors

Having successfully calibrated a dual-axis accelerometer using the dynamic accelerations of a SCARA's end-effector, the accelerometer can be used in future work to track the motions made by the end-effector in the x-y plane. As in [209–212], such motions can be used to experimentally determine the true link lengths of the SCARA. However, just as in the accelerometer calibration routines described in Chapter 6, steps must be taken to try to minimize the errors introduced to the acceleration measurements by the SCARA's non-ideal kinematic parameters. Therefore, instead of moving the end-effector of the SCARA in arcs about the base and elbow joints in turn, only motions involving rotations about the gravity-aligned base joint should be considered. The elbow joint could be locked in slightly different positions during the base joint's arcs – within the range of $\tilde{\theta}_2$ values used in the accelerometer calibration circle – so that the links of the SCARA and the radius of each base joint's arc create a triangle with unique corner angles. The length of the radii of the base joint's arcs can be determined by measuring the normal accelerations with the partially calibrated accelerometer, and the angular velocities with a gyroscope like the one included on Tyndall's WIMU. The direction of the radii can be determined by measuring the direction of the normal acceleration with the fully calibrated accelerometer, if required. By analyzing the geometry of a large number of unique triangles, it should be possible to determine the lengths of the SCARA's two links with a high level of confidence.

In order to achieve higher levels of repeatable accuracy when tracking the motion of a robot's end-effector than has been previously reported by others, an accelerometer that offers a higher level of precision at lower accelerations can be used. An example of such an accelerometer is Analog Devices' ADXL313 triaxial accelerometer [229], which offers 10 bits of resolution in the $\pm 0.5g$ measurement range. The use of a low-range accelerometer would allow the SCARA to move its end-effector with low accelerations and tangential speeds, thus improving its positioning accuracy while introducing fewer and smaller vibrations into the system.

Another possible avenue for future work could be to extend the dual-axis accelerometer calibration routines presented in Chapter 6 into a routine that is capable of calibrating all three axes of a triaxial accelerometer. The SCARA is able to move its end-effector in circles in the x-z and y-z planes by incorporating motion along its prismatic joint, and the dynamic accelerations introduced by such motions could be used to find the sensitivity parameter for the third accelerometer axis. However, this sensitivity parameter must reflect the accelerometer's tilt already incorporated in the x- and y-sensitivities. The full calibration of a triaxial accelerometer would allow for the 3D tracking of the SCARA's end-effector, and could provide feedback to the control system of a robot, as discussed in [208]. This could ultimately improve the positioning accuracy of the end-effector, something that researchers in robotics have been working towards since the first Unimate handled its first die-cast piece so many years ago.

Appendix A

Arrow Node Prototype

This appendix demonstrates how a custom Arrow node can be prototyped in VRML, and then used to create the frames depicted in Figure 3.4.

The prototyped node has its own scene graph, as depicted in Figure A.1. Notice its similarity to the scene graph in Figure 3.3a. It is, in fact, just a simplified version of that scene graph, without the world and sensor nodes.

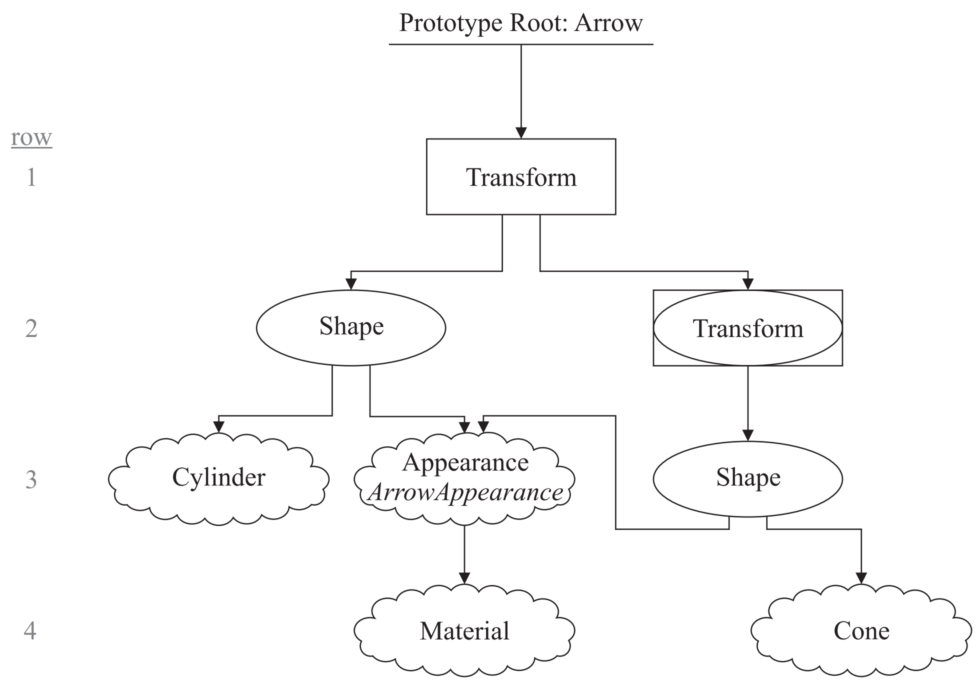


Figure A.1 Scene graph for the prototyped Arrow node.

Once created, the new Arrow node can be used in other scene graphs. That is, it can be used in a virtual world's main scene graph, as shown in Figure A.2, or it can be incorporated into other prototype definitions. In this example, three Arrow nodes are used to create frame0.

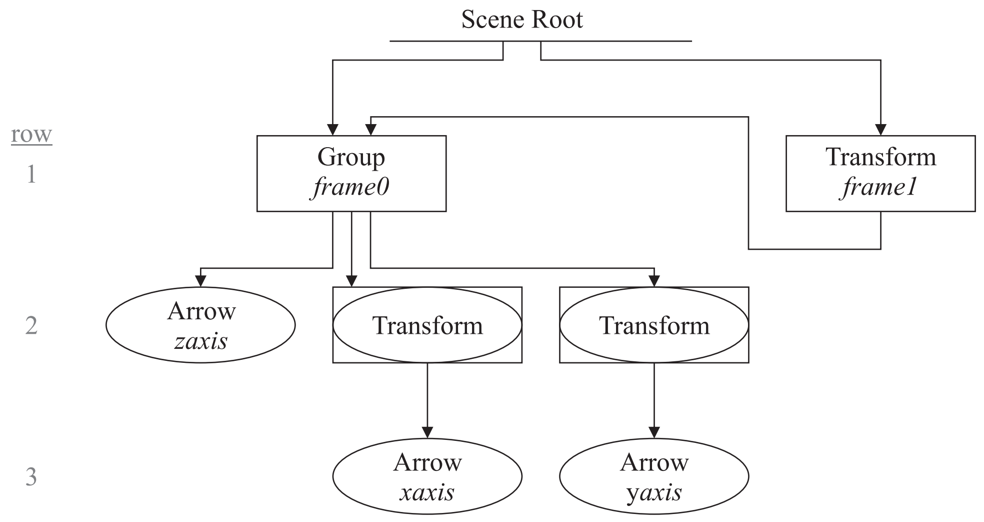


Figure A.2 World scene graph for the frames shown in Figure 3.4.

The code used to realize the prototyped node and the world scene graph is shown in Figure A.3. The PROTO keyword is used to identify the start of a prototype definition. This is followed by the name of the new node type, and any associated fields, exposedFields, eventIns, and eventOuts, along with their default values. In the case of the Arrow node, only one field is defined. This field is named arrowColor, and has a default value of red. The value of this field is passed on to the Material node within the prototype definition, which is what actually controls the color of the cylinder and cone that make up the arrow. When the Arrow node is used in the world scene graph, the default color is used for the z-axis, but different colors are specified for the x- and y-axes.

```
#VRML V2.0 utf8
#------------------------------------------------------------
# frames.wrl
#
# Simple example that defines and uses an arrow prototype node.
#------------------------------------------------------------

#---------
#Prototype definition
#---------
PROTO Arrow [field SFColor arrowColor 1 0 0]
{
 Transform {
  translation 0 1 0, rotation 0 0 1 0, scale 1 1 1
  children [
   Shape {
    appearance DEF ArrowAppearance Appearance {
     material Material {diffuseColor IS arrowColor}
    }
    geometry Cylinder {radius 0.1, height 2}
   }
   Transform {
    translation 0 1.5 0, rotation 0 0 1 0, scale 1 1 1
    children [
     Shape {
      appearance USE ArrowAppearance
      geometry Cone {bottomRadius 0.5, height 1}
     }
    ]
   }
  ]
 }
}

#---------
#Scene Graph
#---------
DEF frame0 Group {
 children [
  DEF zaxis Arrow {}  #default arrowColor: red
  Transform {
   rotation 1 0 0 1.57
   children [
    DEF xaxis Arrow {arrowColor 0 1 0}  #green
   ]
  }
  Transform {
   rotation 0 0 1 -1.57
   children [
    DEF yaxiz Arrow {arrowColor 0 0 1}  #blue
   ]
  }
 ]
}
DEF frame1 Transform {
 translation 5 0 0
 rotation 1 0 0 1.57
 scale 2 2 2
 children [
  USE frame0
 ]
}
```

Figure A.3 VRML code to realize the scene graphs depicted in Figures A.1 and A.2.

Appendix B

Rotation Tool Tests & Surveys

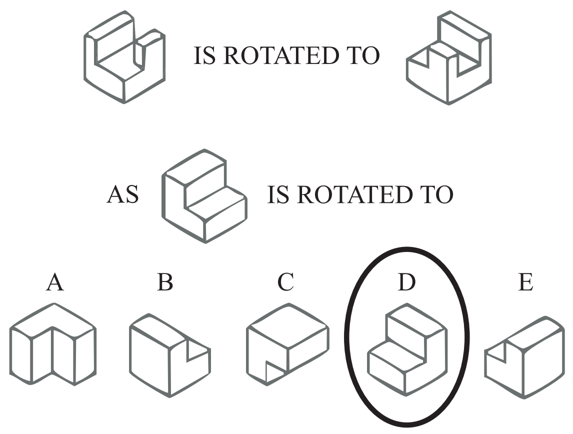
This appendix contains the tests, worksheet, and surveys given to students to determine the effectiveness of the Rotation Tool. The formats of the pre-test, worksheet, and post-test have been modified slightly from the original formats used in the 2012/13 study, in an attempt to clarify and emphasize the relationships between questions. These modified formats should be used for any future work. If there is limited time for the tutorial, the pre-test and post-test can be shortened so that only the first three questions of each are used. Although the answers to the tests and worksheet are not provided here so that these questions can be used in future tutorials, they may be attained by contacting the author directly.

B.1 IRKA

Student ID: Class:

Initial Rotation Knowledge Assessment:
Instructions

This exercise consists of **8** multiple choice questions and **1** true/false question. The questions are designed to see how well you can visualize the rotation of 3D objects. The following is an example of the type of multiple choice question that you will be asked to answer:

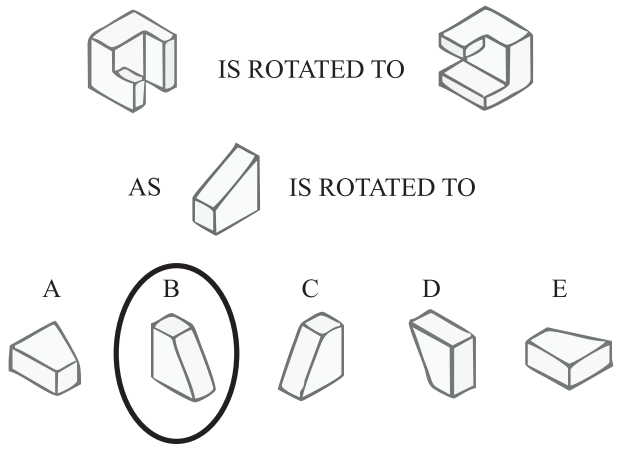


Procedure:

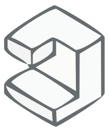
1) Study how the object in the top line of the question is rotated.
2) Picture in your mind what the object shown in the middle line of the question looks like when rotated in exactly the same manner as in the top line.
3) Select from among the five drawings given in the bottom line of the question (A, B, C, D, or E) the one that looks like the middle object rotated into the correct orientation. Feel free to make any marks and/or sketches on the page if you need to. Circle your answer, as shown above. (In this example, D is the correct answer.)

Each question has exactly ONE correct answer.

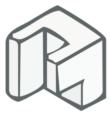
A second example shows a more complex rotation. In this case, the correct answer is B (as shown).



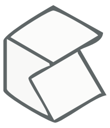
1.



IS ROTATED TO

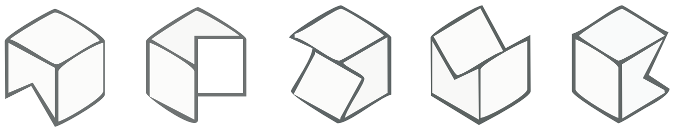


AS

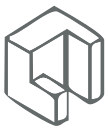


IS ROTATED TO

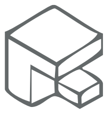
A B C D E



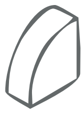
2.



IS ROTATED TO

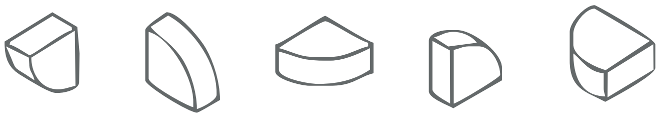


AS

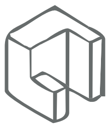


IS ROTATED TO

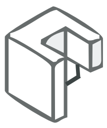
A B C D E



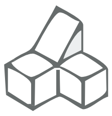
3.



IS ROTATED TO

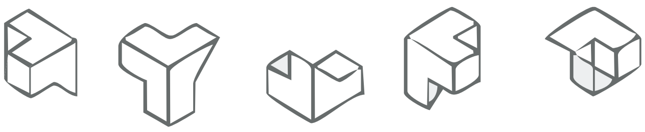


AS

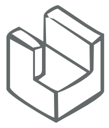


IS ROTATED TO

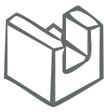
A B C D E



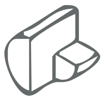
4.



IS ROTATED TO

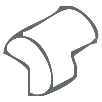


AS

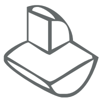


IS ROTATED TO

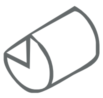
A



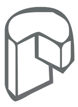
B



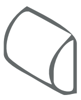
C



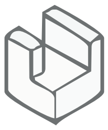
D



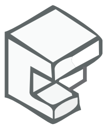
E



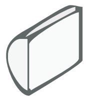
5.



IS ROTATED TO

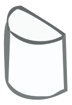


AS

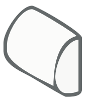


IS ROTATED TO

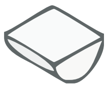
A



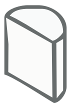
B



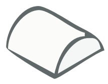
C



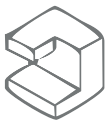
D



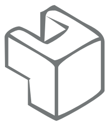
E



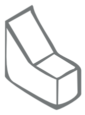
6.



IS ROTATED TO

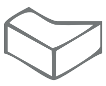


AS

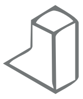


IS ROTATED TO

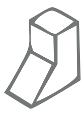
A



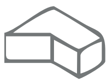
B



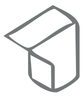
C



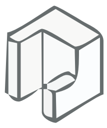
D



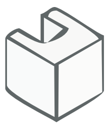
E



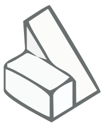
7.



IS ROTATED TO

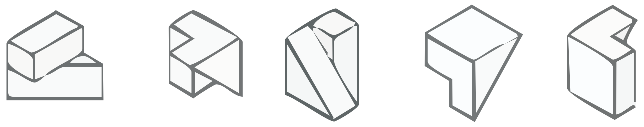


AS

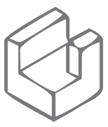


IS ROTATED TO

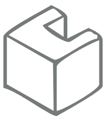
A B C D E



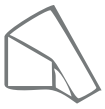
8.



IS ROTATED TO

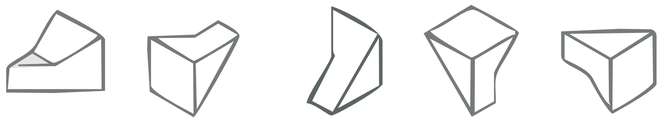


AS



IS ROTATED TO

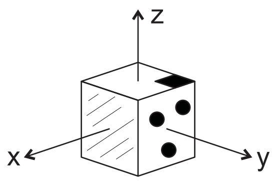
A B C D E



---

9. **True or False?**

You are given a cube (with 6 different faces) centered on the orthogonal frame $\{x, y, z\}$:



Rotating the cube 45° about the *z*-axis followed by a rotation of 45° about the *y*-axis gives a **different** final orientation than rotating the cube 45° about the *y*-axis followed by a rotation of 45° about the *z*-axis.

True False

Any feedback/comments on these questions:

B.2 Pre-Test

Student ID:
Class:

Before Rotation Tool:
Instructions

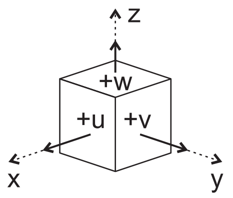
This exercise consists of **6** multiple choice questions. The questions are designed to see how well you understand rotations about fixed axes {*x*, *y*, *z*} and mobile axes {*u*, *v*, *w*}, based on the material presented in the lectures. The following are examples of the types of multiple choice questions that you will be asked to answer:

NOTE:

For all questions, use the right-hand rule to determine the direction of positive angle rotations.

Example

You are given a cube with an attached orthogonal *mobile* frame {*u*, *v*, *w*}, where each face of the cube is labelled according to its corresponding mobile axis. This cube is centered on the orthogonal *fixed* frame {*x*, *y*, *z*} so that the two frames coincide initially, as shown:



Example 1:

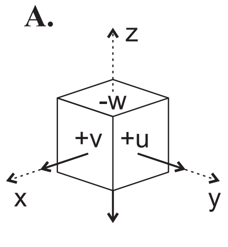
What is the final orientation if the following rotations are made from the initial position?

Fixed Frame:

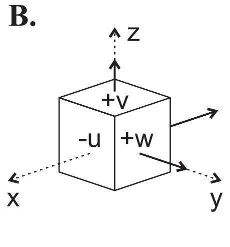
1st rotation about **z-axis** by **90°**

2nd rotation about **x-axis** by **180°**

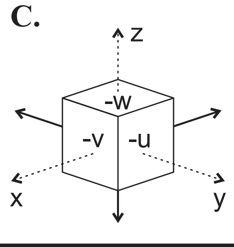
A.



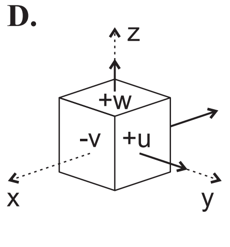
B.



[✓ **C.**]



D.



Example 2:

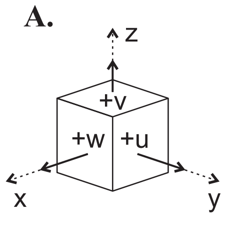
What is the final orientation if the following rotations are made from the initial position?

Mobile Frame:

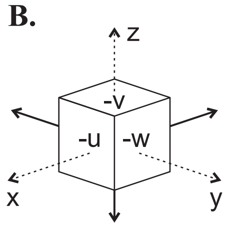
1st rotation about **u-axis** by **180°**

2nd rotation about **v-axis** by **90°**

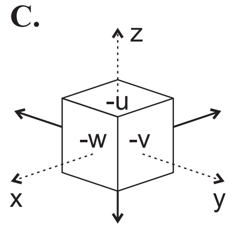
A.



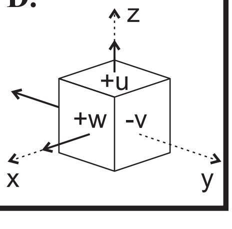
B.



C.

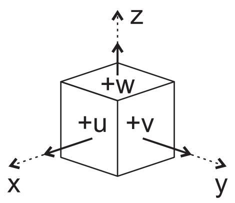


[✓ **D.**]

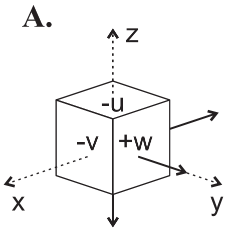


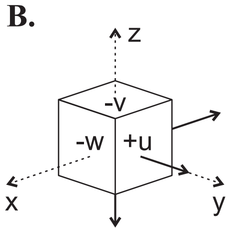
Questions

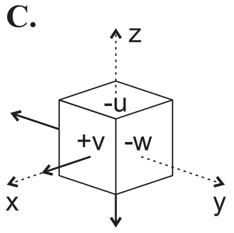
You are given a cube with an attached orthogonal *mobile* frame {*u*, *v*, *w*}, where each face of the cube is labelled according to its corresponding mobile axis. This cube is centered on the orthogonal *fixed* frame {*x*, *y*, *z*} so that the two frames coincide initially, as shown:

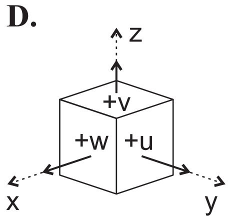


1. What is the final orientation if the following rotations are made from the initial position?
   Fixed Frame:
   1st rotation about ***x*-axis** by **-90°**
   2nd rotation about ***z*-axis** by **90°**

A. 

B. 

C. 

D. 

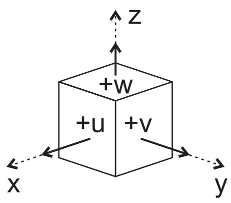
2. What is the final orientation if the following rotations are made from the initial position?
   Fixed Frame:
   1st rotation about ***z*-axis** by **90°**
   2nd rotation about ***x*-axis** by **-90°**

   A. Same as final orientation in #1
   B. Different from final orientation in #1
   C. Don't know

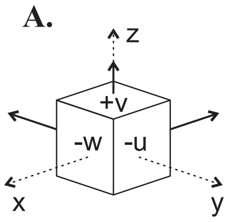
3. What is the final orientation if the following rotations are made from the initial position?
   Mobile Frame:
   1st rotation about ***u*-axis** by **-90°**
   2nd rotation about ***w*-axis** by **90°**

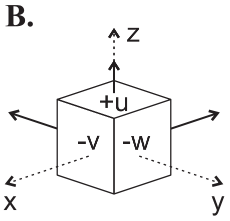
   A. Same as final orientation in #1
   B. Same as final orientation in #2
   C. Same as final orientation in both #1 and #2
   D. Different from final orientation in both #1 and #2
   E. Don't know

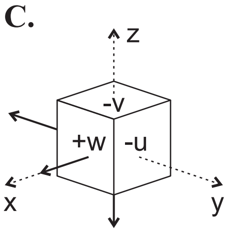
You are given a cube with an attached orthogonal *mobile* frame {*u*, *v*, *w*}, where each face of the cube is labelled according to its corresponding mobile axis. This cube is centered on the orthogonal *fixed* frame {*x*, *y*, *z*} so that the two frames coincide initially, as shown:

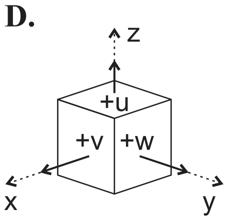


4. What is the final orientation if the following rotations are made from the initial position?
   Mobile Frame:
   1st rotation about ***w*-axis** by **90°**
   2nd rotation about ***u*-axis** by **-90°**
   3rd rotation about ***v*-axis** by **180°**

A.


B.


C.


D.


5. What is the final orientation if the following rotations are made from the initial position?
   Mobile Frame:
   1st rotation about ***v*-axis** by **180°**
   2nd rotation about ***u*-axis** by **-90°**
   3rd rotation about ***w*-axis** by **90°**

   A. Same as final orientation in #4
   B. Different from final orientation in #4
   C. Don't know

6. What is the final orientation if the following rotations are made from the initial position?
   Fixed Frame:
   1st rotation about ***y*-axis** by **180°**
   2nd rotation about ***x*-axis** by **-90°**
   3rd rotation about ***z*-axis** by **90°**

   A. Same as final orientation in #4
   B. Same as final orientation in #5
   C. Same as final orientation in both #4 and #5
   D. Different from final orientation in both #4 and #5
   E. Don't know

Survey

	1	2	3	4	5
1. Based on the lectures, how well do you feel you understand the difference between fixed axis rotations and mobile axis rotations? (1 = Not at all, 2 = Not really, 3 = Partially understand, 4 = Mostly understand, 5 = Completely understand)	☐	☐	☐	☐	☐

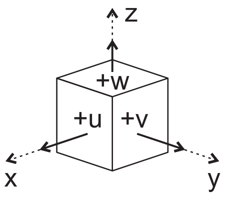
B.3 Worksheet

Student ID:
Class:

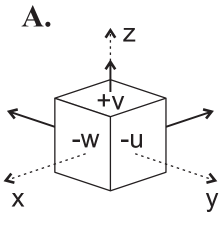
Rotation Tool:
Worksheet

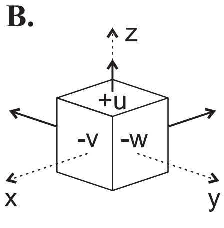
Use the Rotation Tool to help you work through the following questions.

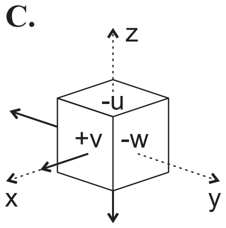
You are given a cube with an attached orthogonal *mobile* frame $\{u, v, w\}$, where each face of the cube is labelled according to its corresponding mobile axis. This cube is centered on the orthogonal *fixed* frame $\{x, y, z\}$ so that the two frames coincide initially, as shown:

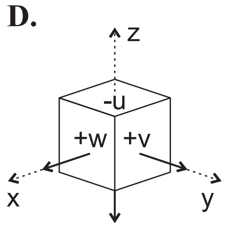


1. What is the final orientation if the following rotations are made from the initial position?
   Fixed Frame:
   1st rotation about $\boldsymbol{x_0}$**-axis** by **45°**
   2nd rotation about $\boldsymbol{y_0}$**-axis** by **90°**
   3rd rotation about $\boldsymbol{z_0}$**-axis** by **-45°**

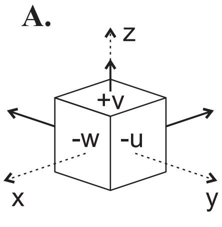
A.


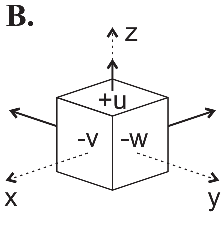
B.


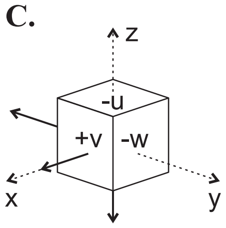
C.


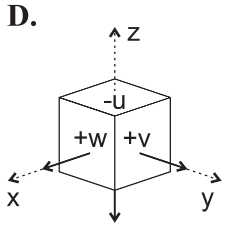
D.


2. a) If the same rotations are made from the initial position about the mobile axes, what is the final orientation?
   Mobile Frame:
   1st rotation about $\boldsymbol{x_0}$**-axis (*u*-axis)** by **45°**
   2nd rotation about $\boldsymbol{y_1}$**-axis (*v*-axis)** by **90°**
   3rd rotation about $\boldsymbol{z_2}$**-axis (*w*-axis)** by **-45°**

A.


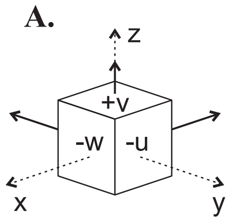
B.


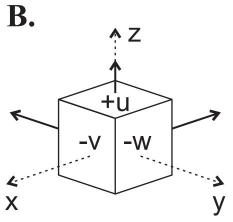
C.


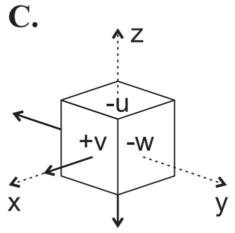
D.


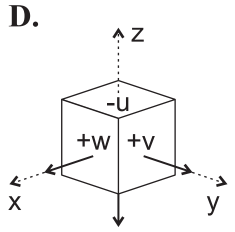
   b) Is this the same or different from the final orientation in #1?
      A. Same
      B. Different

3. a) If the rotations about the fixed axes are reversed, what is the final orientation?
   Fixed Frame:
   1st rotation about **z_0-axis** by **-45°**
   2nd rotation about **y_0-axis** by **90°**
   3rd rotation about **x_0-axis** by **45°**

A.


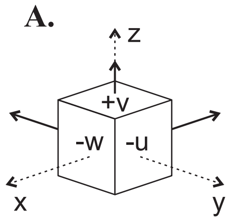
B.


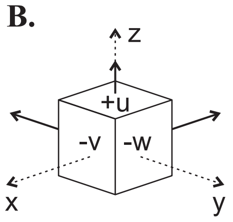
C.


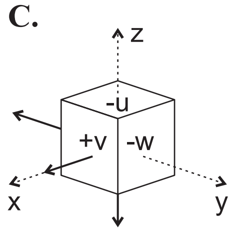
D.


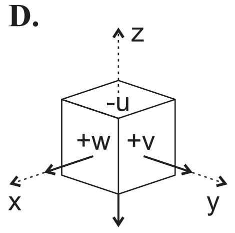
   b) Is this the same or different from the final orientation in #1?
      A. Same **B.** Different

4. a) If the rotations about the mobile axes are reversed, what is the final orientation?
   Mobile Frame:
   1st rotation about **z_0-axis (*w*-axis)** by **-45°**
   2nd rotation about **y_1-axis (*v*-axis)** by **90°**
   3rd rotation about **x_2-axis (*u*-axis)** by **45°**

A.


B.


C.


D.


   b) Is this the same or different from the final orientation in #1?
      A. Same **B.** Different

   c) Is this the same or different from the final orientation in #2?
      A. Same **B.** Different

5. Based on your answers above, does the order in which the rotations are made about the same kind of frame make a difference to the final orientation?
   A. No: rotations are commutative. **B.** Yes: rotations are non-commutative.

6. Based on your answers above, what is the relationship between multiple rotations about the fixed frame and multiple rotations about the mobile frame?
   A. If the same rotations are made in the <u>same</u> order in both frames, the final orientations will be the <u>same</u>.
   B. If the same rotations are made in the <u>exact opposite</u> order in the mobile frame as compared to in the fixed frame, the final orientations will be the <u>same</u>.
   C. There is no relationship .

B.4 Post-Test

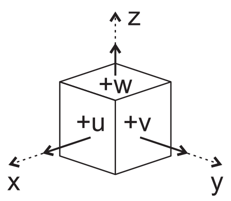
Student ID:
Class:

After Rotation Tool:
Instructions

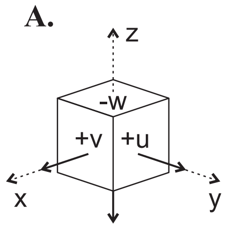
This exercise consists of **6** multiple choice questions. The questions are designed to see how well you understand rotations about fixed axes {*x*, *y*, *z*} and mobile axes {*u*, *v*, *w*}, after having used the Rotation Tool. The following are examples of the types of multiple choice questions that you will be asked to answer (same format as the exercises done before using the Rotation Tool):

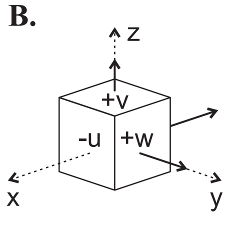
NOTE:
For all questions, use the right-hand rule to determine the direction of positive angle rotations.

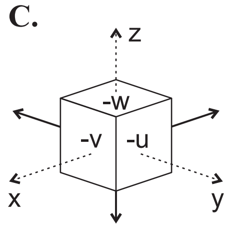
Example
You are given a cube with an attached orthogonal *mobile* frame {*u*, *v*, *w*}, where each face of the cube is labelled according to its corresponding mobile axis. This cube is centered on the orthogonal *fixed* frame {*x*, *y*, *z*} so that the two frames coincide initially, as shown:

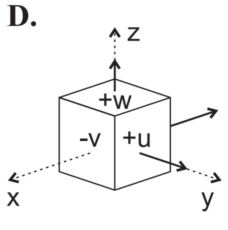


Example 1:
What is the final orientation if the following rotations are made from the initial position?
Fixed Frame:
1st rotation about ***z*-axis** by **90°**
2nd rotation about ***x*-axis** by **180°**

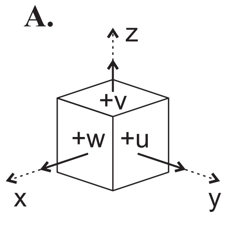
A.


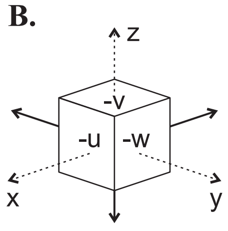
B.


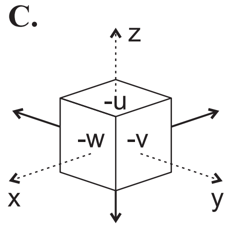
[✓ **C.**]


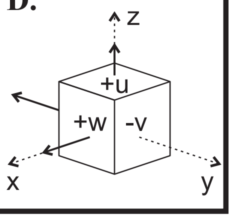
D.


Example 2:
What is the final orientation if the following rotations are made from the initial position?
Mobile Frame:
1st rotation about ***u*-axis** by **180°**
2nd rotation about ***v*-axis** by **90°**

A.


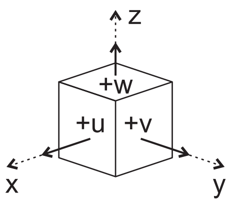
B.


C.


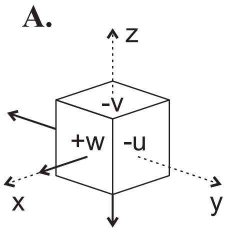
[✓ **D.**]


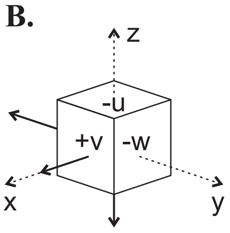
Questions

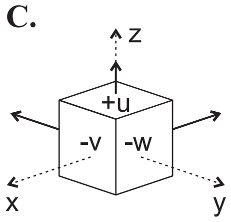
You are given a cube with an attached orthogonal *mobile* frame {*u*, *v*, *w*}, where each face of the cube is labelled according to its corresponding mobile axis. This cube is centered on the orthogonal *fixed* frame {*x*, *y*, *z*} so that the two frames coincide initially, as shown:

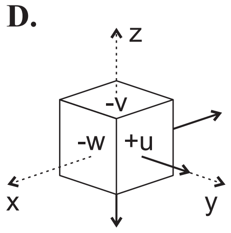


1. What is the final orientation if the following rotations are made from the initial position?
Mobile Frame:
1st rotation about ***w*-axis** by **90°**
2nd rotation about ***v*-axis** by **-90°**

A.


B.


C.


D.


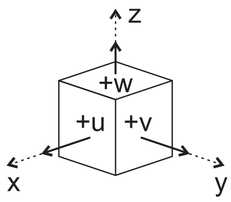
2. What is the final orientation if the following rotations are made from the initial position?
Fixed Frame:
1st rotation about ***y*-axis** by **-90°**
2nd rotation about ***z*-axis** by **90°**

A. Same as final orientation in #1
B. Different from final orientation in #1
C. Don't know

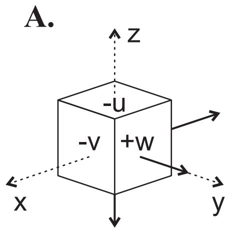
3. What is the final orientation if the following rotations are made from the initial position?
Mobile Frame:
1st rotation about ***v*-axis** by **-90°**
2nd rotation about ***w*-axis** by **90°**

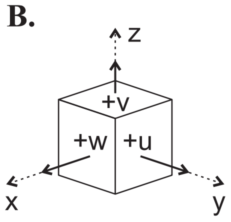
A. Same as final orientation in #1
B. Same as final orientation in #2
C. Same as final orientation in both #1 and #2
D. Different from final orientation in both #1 and #2
E. Don't know

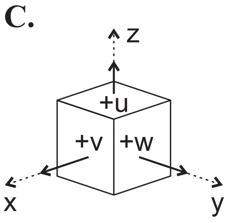
You are given a cube with an attached orthogonal *mobile* frame $\{u, v, w\}$, where each face of the cube is labelled according to its corresponding mobile axis. This cube is centered on the orthogonal *fixed* frame $\{x, y, z\}$ so that the two frames coincide initially, as shown:

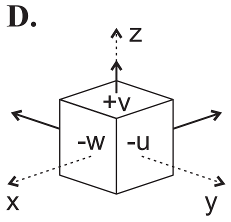


4. What is the final orientation if the following rotations are made from the initial position?
Fixed Frame:
1st rotation about ***y*-axis** by **-90°**
2nd rotation about ***z*-axis** by **90°**
3rd rotation about ***x*-axis** by **180°**

A.


B.


C.


D.


5. What is the final orientation if the following rotations are made from the initial position?
Mobile Frame:
1st rotation about ***v*-axis** by **-90°**
2nd rotation about ***w*-axis** by **90°**
3rd rotation about ***u*-axis** by **180°**

A. Same as final orientation in #4
B. Different from final orientation in #4
C. Don't know

6. What is the final orientation if the following rotations are made from the initial position?
Fixed Frame:
1st rotation about ***x*-axis** by **180°**
2nd rotation about ***z*-axis** by **90°**
3rd rotation about ***y*-axis** by **-90°**

A. Same as final orientation in #4
B. Same as final orientation in #5
C. Same as final orientation in both #4 and #5
D. Different from final orientation in both #4 and #5
E. Don't know

Rotation Tool:
Survey

		1	2	3	4	5
1.	After using the tool, how well do you feel you understand the difference between fixed axes rotations and mobile axes rotations? (1 = Not at all, 2 = Not really, 3 = Partially understand, 4 = Mostly understand, 5 = Completely understand)	☐	☐	☐	☐	☐
2.	To what extent has this tool increased your knowledge on rotations? (1 = Significantly decreased, 2 = Slightly decreased, 3 = Same, 4 = Slightly increased, 5 = Significantly increased)	☐	☐	☐	☐	☐
3.	How easy is this tool to use? (1 = Very difficult, 2 = Somewhat difficult, 3 = Neutral, 4 = Somewhat easy, 5 = Very easy)	☐	☐	☐	☐	☐
4.	How well does this tool demonstrate the difference between fixed axes and mobile axes rotations compared to 2D illustrations in class notes? (1 = Much worse, 2 = Slightly worse, 3 = Same, 4 = Slightly better, 5 = Much better)	☐	☐	☐	☐	☐
5.	How likely are you to use this tool to study for your final exam? (1 = Definitely not, 2 = Probably not, 3 = Neutral, 4 = Probably will, 5 = Definitely will)	☐	☐	☐	☐	☐

Any additional comments on this tool:

Any additional comments on this seminar:

Thank you very much for taking the time to participate in this seminar
and for trying out the Rotation Tool. ☺

Appendix C

Rotation Tool Instructions

This appendix contains the written set of instructions provided to students at UCC regarding the use of the Rotation Tool.

C.1 Before You Begin

1. Copy the **VRML Fwd Kin** folder from *denovo\Public\EE_4015* to the MATLAB folder on your personal U drive (*U:\MATLAB*).
2. Open MATLAB.
3. Change MATLAB's current directory to **U:\MATLAB\VRML Fwd Kin**

C.2 Instructions and Examples

1. To open the Rotation Tool, type in the following at the command prompt in MATLAB's command window:
   rotationTransforms

   **Don't try to double click on rotationTransforms.m in the current folder window. This will not open the Rotation Tool; it will only open MATLAB's editor to show you the code used to create the tool.*

2. Enter the following values into the graphical user interface (GUI) and observe what happens in the virtual reality world each time you press the **Rotate!** button.

Fixed Frame:	Mobile Frame:
1st rotation about $\boldsymbol{x_0}$**-axis** by **45** degrees	1st rotation about $\boldsymbol{x_0}$**-axis** by **45** degrees
2nd rotation about $\boldsymbol{y_0}$**-axis** by **90** degrees	2nd rotation about $\boldsymbol{y_1}$**-axis** by **90** degrees
3rd rotation about $\boldsymbol{z_0}$**-axis** by **-20** degrees	3rd rotation about $\boldsymbol{z_2}$**-axis** by **-20** degrees

Notice that after the first rotation in each system, both frames look identical. However, the second and third orientations are *different* in the fixed and mobile frames, illustrating the effect of compound rotations in each system.

***NOTE:** The right-hand rule was used as the convention to determine positive rotations: when your right thumb points along the positive axis of rotation (towards positive infinity), your fingers will curl around the axis and point in the direction of positive rotation. Put another way, when looking from the origin toward positive infinity along the axis of rotation, the positive direction of rotation is clockwise.

3. Press the **Reset** button for both frames.

4. Enter the following values into the GUI and observe what happens in the virtual reality world when you press the **Rotate!** button.

Fixed Frame:	Mobile Frame:
1st rotation about $\boldsymbol{x_0}$**-axis** by **45** degrees	1st rotation about $\boldsymbol{z_0}$**-axis** by **-20** degrees
2nd rotation about $\boldsymbol{y_0}$**-axis** by **90** degrees	2nd rotation about $\boldsymbol{y_1}$**-axis** by **90** degrees
3rd rotation about $\boldsymbol{z_0}$**-axis** by **-20** degrees	3rd rotation about $\boldsymbol{x_2}$**-axis** by **45** degrees

Notice that in this case the third (final) orientations are the same in the fixed and mobile frames. This is because the final fixed frame transformation is calculated by **pre**-multiplying all the rotation matrices, while the final mobile frame transformation is calculated by **post**-multiplying the rotation matrices - which means the final overall transformation is the same for both.

5. Try adding in your own values and confirming for yourself the difference between fixed frame rotations and mobile frame rotations. Use the tool to complete the worksheet given in class.

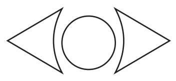
C.3 Navigation Tips

Sometimes it may be beneficial to change the view of the 3D frames in the virtual world. You can easily navigate around the virtual world and view the frames from different perspectives as follows.

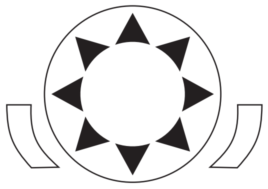
If using the keyboard:

$+$	Press + to zoom in.
$-$	Press - to zoom out.
$\rightarrow$	Press the right arrow to move the camera angle right (effectively moving the frames left).
$\leftarrow$	Press the left arrow to move the camera angle left (effectively moving the frames right).
$\uparrow$	Press the up arrow to move the camera angle up (effectively moving the frames down).
$\downarrow$	Press the down arrow to move the camera angle down (effectively moving the frames up).

If using the navigation panel on the bottom of the window:


Press the circle in between the two arrows to return to the default view.

Select **W** to **w**alk, **E** to **e**xamine, or **F** to **f**ly around the virtual scene.


Use the eight arrows on the navigation wheel to adjust the camera view. Press the buttons on the left and right of the navigation wheel to slide the view left and right.

For more information on navigating around the virtual world, please refer to **Help** → **Browse Help** in the virtual reality window and click on the Navigation Panel link.

Appendix D

Build-A-Robot Seminar Handouts

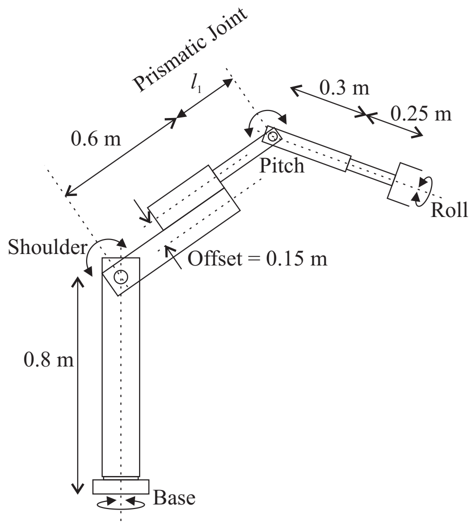
This appendix contains the written set of instructions provided to students at UCC regarding the use of Build-A-Robot to study a past examination question on forward kinematics. Note that this sample examination question was not created by the author, but by Dr. Richard Kavanagh. This appendix also contains the survey given to students at the end of the 2011/12 and 2012/13 Build-A-Robot seminars.

D.1 Before You Begin

1. Copy the **VRML Fwd Kin** folder from *denovo\Public\EE_4015* to the MATLAB folder on your personal U drive (*U:\MATLAB*).
2. Open MATLAB.
3. Change MATLAB's current directory to **U:\MATLAB\VRML Fwd Kin**

D.2 Forward Kinematics Sample Question

A five axis robot is of a modified spherical structure, where the prismatic joint is offset by 0.15 m, as shown in the diagram. The offset is in the plane of the paper only. The minor (pitch and roll) joints are of standard type, the pitch



joint of the wrist having a horizontal axis. You can assume that the joints also have no offsets in a plane perpendicular to the paper. A gripper acts as the end-effector, as shown. The prismatic length l_1 can vary from 0.1 m to 0.5 m. If the dimensions of the system are as on the diagram, use the Denavit-Hartenberg algorithm and matrix to:

(a) Assign suitable frames to the robot, using a link-coordinate diagram. [10 marks]

(b) Tabulate the kinematic parameters associated with the robot. [5 marks]

(c) Write down an expression for T_{base}^{tool} for this robot, based on your frame assignments. Your answer can take the form of a product of a number of matrices, i.e. you are *not* required to perform the multiplication. [3 marks]

(d) Perform a sanity check on T_1^2. [2 marks]

One possible solution* (part (b)):

Link	$\tilde{d}$	$\tilde{\theta}$	$\tilde{a}$	$\tilde{\alpha}$	Home q
1	0.8 m	q_1	0 m	90°	0°
2	0 m	q_2	0.15 m	90°	90°
3	q_3	180°	0 m	90°	$0.6 + l_1$
4	0 m	q_4	0 m	-90°	0°
5	0.55 m	q_5	0 m	0°	-90°

* Based on the frame assignment chosen in the powerpoint slides: *denovo\Public\EE_4015\1998 Summer Q2 Forward KinematicsMF.ppt*

D.3 Build-A-Robot Instructions

1. To open Build-A-Robot, type in the following at the command prompt in MATLAB's command window:

   runBuildARobot

   **Don't try to double click on runBuildARobot.m in the current folder window. This will not open Build-A-Robot; it will only open MATLAB's editor to show you the code used to create the tool.*

2. Try modeling the 5-axis robot from the Summer 1998 final exam. To do so, enter the following values in the Build-A-Robot GUI:

   Degrees of Freedom: 5 DOF

Frame	Joint Type	$\tilde{d}$	$\tilde{\theta}$	$\tilde{a}$	$\tilde{\alpha}$
L0	Revolute	0.8	0	0	90
L1	Revolute	0	90	0.15	90
L2	Prismatic	0.7	180	0	90
L3	Revolute	0	0	0	-90
L4	Revolute	0.55	-90	0	0

   These are the Denavit-Hartenberg (D-H) parameters for the 5-axis robot, based on one possible frame assignment. (Note that the frame assignment and solution are not unique.) The soft home position used assumes that the prismatic joint is at minimum extension (i.e. $l_1 = 0.1$ m).

3. Press the **Build Robot** button. A 3D link-coordinate representation of the 5-axis robot should be created in the virtual reality window. Notice that the $\tilde{d}$ lengths are shown in light blue, the $\tilde{a}$ lengths are shown in purple, the revolute joints are shown as cylinders, and the prismatic joints are shown as cubes.

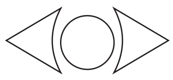
4. Navigate around the virtual world and view the robot from different angles.

   If using the keyboard:

+	Press + to zoom in.
−	Press - to zoom out.

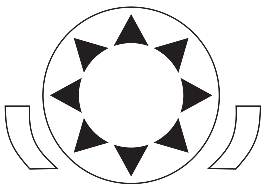
$\rightarrow$	Press the right arrow to move the camera angle right (effectively moving the frames left).
$\leftarrow$	Press the left arrow to move the camera angle left (effectively moving the frames right).
$\uparrow$	Press the up arrow to move the camera angle up (effectively moving the frames down).
$\downarrow$	Press the down arrow to move the camera angle down (effectively moving the frames up).

If using the navigation panel on the bottom of the window:



Press the circle in between the two arrows to return to the default view.

Select **W** to **w**alk, **E** to **e**xamine, or **F** to **f**ly around the virtual scene.



Use the eight arrows on the navigation wheel to adjust the camera view. Press the buttons on the left and right of the navigation wheel to slide the view left and right.

For more information on navigating around the virtual world, please refer to **Help** → **Browse Help** in the virtual reality window and click on the Navigation Panel link.

5. You can change the variable joint parameters ($\tilde{\theta}$ if a revolute joint, $\tilde{d}$ if prismatic) by interacting directly with the virtual reality robot representation. To do so, check the box beside **Modify variable joint parameters in virtual world**. Then in the virtual world, hover over one of the robot's joint or links; when the cursor changes to a hand, click the joint or link and hold the mouse button down as you drag the cursor around the screen. Notice that for revolute joints, this will result in a change to the corresponding $\tilde{\theta}$ value (as seen in the GUI and the rotation of the link-coordinate representation). Likewise, for prismatic joints you can change the $\tilde{d}$ value (as seen in the GUI and in the length of the link in the virtual world).

   To undo any changes made while the checkbox has been checked, simply press the **Undo Joint Parameter Modifications** button. To prevent any unintentional changes to the variable joint parameters, simply uncheck the checkbox.

6. You can also make changes to any of the parameters directly in the GUI. Try changing $\tilde{a}_3$ from 0 m to 0.25 m and notice what happens to the link-coordinate representation.

7. You can examine any of the frames by clicking the appropriate **Display Frame Ln** or the **Display Tool Frame** checkbox. If you want to view all of the frames at once, simply go to the **Tools** menu and select **Display All Reference Frames** (this menu item should have a checkmark next to it when all frames are displayed). To hide all reference frames, go to the Tools $\rightarrow$ Display All Reference Frames again (this should remove the checkmark in the menu dropdown) or uncheck the Display Frame Ln and Display Tool Frame checkboxes individually.

8. Press the **Calculate Transforms** button. This opens a window displaying all of the T_{k-1}^{k} matrices (as sanity checks), as well as the overall transformation from gripper to base. If the original D-H parameters (as given in step 2) are used, you will see that

$$T_1^2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0.15 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

   Note that T_1^2 was asked for on the Summer 1998 final exam, as a sanity check. From the first three columns in the matrix above, it can be seen that frame 2 is rotated from frame 1 so that the x_2-axis is pointing in the y_1-axis direction, the y_2-axis is pointing in the z_1-axis direction, and the z_2-axis is pointing in the x_1-axis direction (which agrees with the frames displayed in the virtual world). The final column of the homogeneous transformation matrix tells us that the origin of frame 2 is translated 0.15 m in the positive y_1-axis direction from the origin of frame 1. Similar evaluations can be carried out on all of the "sanity check" homogeneous transformations, as well as T_0^5 (the homogeneous transformation that relates the position of the gripper to the base in this case).

9. At any time, your work can be saved as a *.br file. Simply go to **File** $\rightarrow$ **Save As**.

10. You can open other *.br files by going to **File** → **Open**. Sample solutions to past exam questions (exam_summer2009, exam_autumn2010, exam_summer2010) have been included in the Build-A-Robot package to help you study on your own.

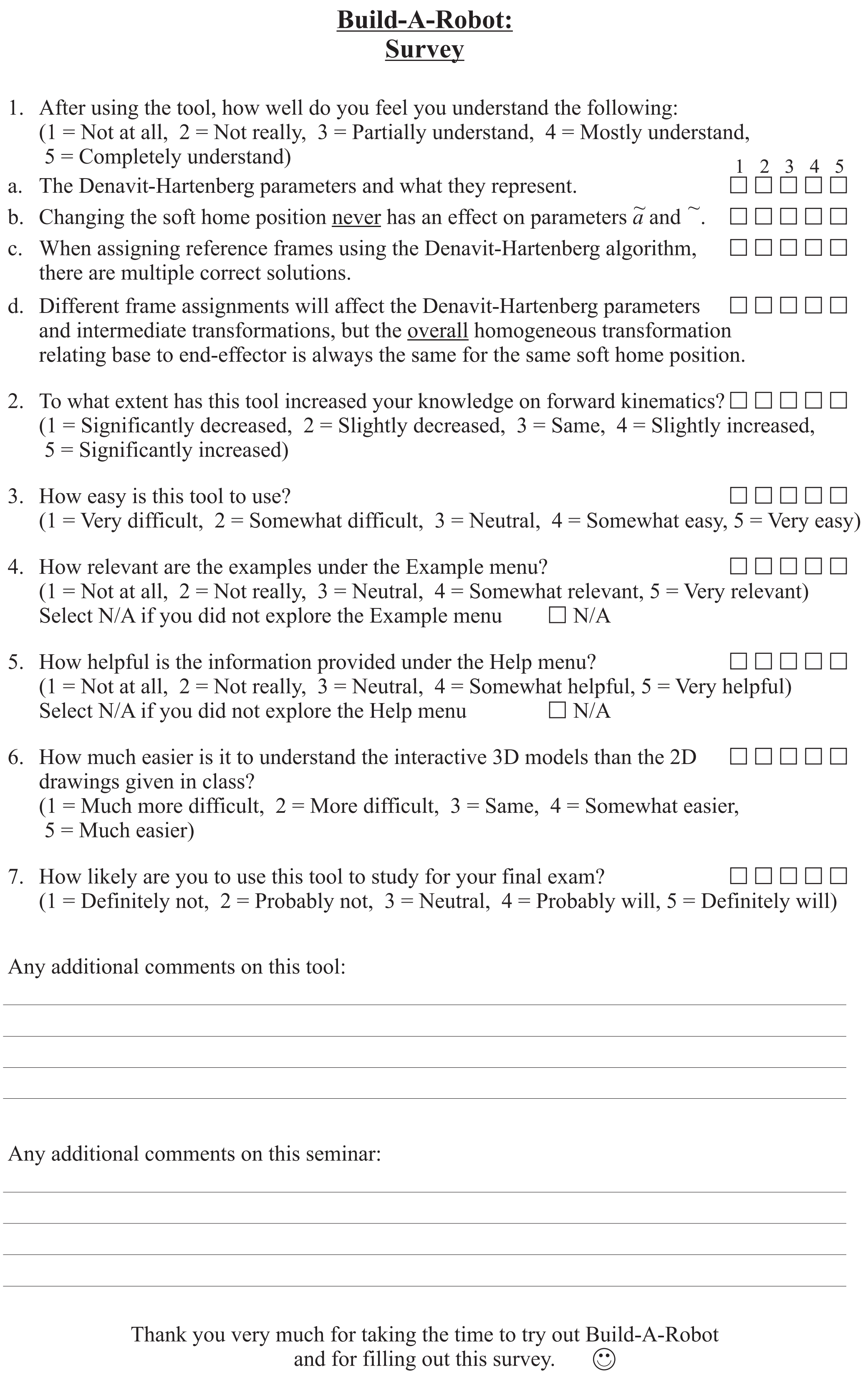
11. You can also open up models of real world robots by looking under the **Examples** menu.

12. Refer to the **Help** menu at any time for detailed tutorials on the Denavit-Hartenberg parameters, the right-hand rule, homogeneous transformations, and on how to use the Build-A-Robot program itself.

13. Press the **Reset** button to clear the fields and start again on your own. The tool can be used to see a 3D view of the examples provided in class, as a check for homework assignments, as well as preparation for the final exam.

14. Please be sure to complete the survey on what you thought of Build-A-Robot.

D.4 Survey



<u>Build-A-Robot:</u>
<u>Survey</u>

1. After using the tool, how well do you feel you understand the following:
   (1 = Not at all, 2 = Not really, 3 = Partially understand, 4 = Mostly understand, 5 = Completely understand)

		1	2	3	4	5
a.	The Denavit-Hartenberg parameters and what they represent.	☐	☐	☐	☐	☐
b.	Changing the soft home position <u>never</u> has an effect on parameters $\widetilde{a}$ and $\widetilde{}$.	☐	☐	☐	☐	☐
c.	When assigning reference frames using the Denavit-Hartenberg algorithm, there are multiple correct solutions.	☐	☐	☐	☐	☐
d.	Different frame assignments will affect the Denavit-Hartenberg parameters and intermediate transformations, but the <u>overall</u> homogeneous transformation relating base to end-effector is always the same for the same soft home position.	☐	☐	☐	☐	☐

2. To what extent has this tool increased your knowledge on forward kinematics? ☐ ☐ ☐ ☐ ☐
   (1 = Significantly decreased, 2 = Slightly decreased, 3 = Same, 4 = Slightly increased, 5 = Significantly increased)

3. How easy is this tool to use? ☐ ☐ ☐ ☐ ☐
   (1 = Very difficult, 2 = Somewhat difficult, 3 = Neutral, 4 = Somewhat easy, 5 = Very easy)

4. How relevant are the examples under the Example menu? ☐ ☐ ☐ ☐ ☐
   (1 = Not at all, 2 = Not really, 3 = Neutral, 4 = Somewhat relevant, 5 = Very relevant)
   Select N/A if you did not explore the Example menu ☐ N/A

5. How helpful is the information provided under the Help menu? ☐ ☐ ☐ ☐ ☐
   (1 = Not at all, 2 = Not really, 3 = Neutral, 4 = Somewhat helpful, 5 = Very helpful)
   Select N/A if you did not explore the Help menu ☐ N/A

6. How much easier is it to understand the interactive 3D models than the 2D drawings given in class? ☐ ☐ ☐ ☐ ☐
   (1 = Much more difficult, 2 = More difficult, 3 = Same, 4 = Somewhat easier, 5 = Much easier)

7. How likely are you to use this tool to study for your final exam? ☐ ☐ ☐ ☐ ☐
   (1 = Definitely not, 2 = Probably not, 3 = Neutral, 4 = Probably will, 5 = Definitely will)

Any additional comments on this tool:

Any additional comments on this seminar:

Thank you very much for taking the time to try out Build-A-Robot
and for filling out this survey. ☺

References

[1] Ampère, A.-M. (1834). *Essai sur la philosophie des sciences, ou Exposition analytique d'une classification naturelle de toutes les connaissances humaines.* Paris: Bachelier.

[2] Hartenberg, R. S. & Denavit, J. (1964). *Kinematic synthesis of linkages.* New York: McGraw-Hill.

[3] Euler, L. (1736). *Mechanica, volume 1* (I. Bruce, Trans.).

[4] Euler, L. (1749). Recherches sur le mouvement des corps célestes en général. *Mémoires de l'académie des sciences de Berlin*, *3*, 93–143.

[5] Euler, L. (1776). General formulas for the translation of arbitrary rigid bodies (J. Sten, Trans.) *Novi Commentarii academiae scientiarum Petropolitanae*, *20*, 189–207.

[6] Watt, J. (1781). British Patent No. 1306, issued 25 October 1781.

[7] Wasbrough, M. (1779). British Patent No. 1213, issued 10 March 1779.

[8] Pickard, J. (1780). British Patent No. 1263, issued 23 August 1780.

[9] Ferguson, E. S. (1962). Paper 27: Kinematics of mechanisms from the time of Watt. In *Bulletin 228: Contributions from the museum of history and technology* (pp. 185–230). Smithsonian Institution.

[10] Watt, J. (1784). British Patent No. 1432, issued 28 April 1784.

[11] Chebyshev, P. L. (1854). Théorie des mécanismes connus sous le nom de parallélogrammes. *Mémoires des savants étrangers présentés á l'Académie de Saint-Pétersbourg*, *7*, 539–586.

[12] Peaucellier, C.-N. (1873). Note sur une question de géométrie de compas. *Nouvelles annales de mathématiques, journal des candidats aux écoles polytechnique et normale*. 2nd ser., *12*, 71–78.

[13] Reuleaux, F. (1876). *The kinematics of machinery: Outlines of a theory of machines* (A. B. W. Kennedy, Trans.). London: Macmillan and Co.

[14] Denavit, J. & Hartenberg, R. S. (1955). A kinematic notation for lower-pair mechanisms based on matrices. *J Appl Mech*, *22*, 215–221.

[15] Čapek, K. (2014). *R.U.R. (Rossum's Universal Robots): A play in introductory scene and three acts* (D. Wyllie, Trans.). Adelaide, Australia: eBooks@Adelaide. Retrieved from https://ebooks.adelaide.edu.au

[16] Quinn, R. D., Nelson, G. M., Bachmann, R. J., Kingsley, D. A., Offi, J. & Ritzmann, R. E. (2001). Insect designs for improved robot mobility. In *Climbing and walking robots: From biology to industrial applications (CLAWAR 2001)* (pp. 69–76).

[17] Wood, R. J. (2008). The first takeoff of a biologically inspired at-scale robotic insect. *IEEE Trans. on Robotics*, *24*(2), 341–347.

[18] Hopkins, J. K., Spranklin, B. W. & Gupta, S. K. (2009). A survey of snake-inspired robot designs. *Bioinspiration & Biomimetics*, *4*(2), 021001.

[19] Send, W., Fischer, M., Jebens, K., Mugrauer, R., Nagarathinam, A. & Scharstein, F. (2012). Artificial hinged-wing bird with active torsion and partially linear kinematics. In *Proc. 28th Int. Congress of the Aeronautical Sciences (ICAS)* (Vol. 2, pp. 1148–1157).

[20] Li, Y. [Yibin], Li, B., Ruan, J. & Rong, X. (2011). Research of mammal bionic quadruped robots: A review. In *Proc. IEEE 5th Int. Conf. Robotics, Automation and Mechatronics (RAM)* (pp. 166–171).

[21] Wang, J. & Li, Y. [Yangmin]. (2008). A survey on the structures of current mobile humanoid robots. In *Proc. IEEE Asia Pacific Conf. Circuits and Systems (APCCAS 2008)* (pp. 1826–1829).

[22] Gouaillier, D., Hugel, V., Blazevic, P., Kilner, C., Monceaux, J., Lafourcade, P., . . . Maisonnier, B. (2009). Mechatronic design of NAO humanoid. In *IEEE Int. Conf. Robotics and Automation (ICRA '09)* (pp. 769–774).

[23] RIA. (1999). *ANSI/RIA R15.06-1999: American national standard for industrial robots and robot systems - Safety requirements.*

[24] BARA. (n.d.). Definition of robots and robot types. Retrieved from http://www.bara.org.uk/definition-of-robots.html

[25] JARA. (n.d.). About JARA: Outline. Retrieved from http://www.jara.jp/e/h/jara01.html

[26] Prassler, E., Ritter, A., Schaeffer, C. & Fiorini, P. (2000). A short history of cleaning robots. *Autonomous Robots*, *9*(3), 211–226.

[27] Prassler, E. & Kosuge, K. (2008). Domestic robots. In B. Siciliano & O. Khatib (Eds.), *Springer handbook of robotics* (pp. 1253–1281). Berlin, Germany: Springer-Verlag.

[28] Leite, I., Martinho, C. & Paiva, A. (2013). Social robots for long-term interaction: A survey. *International Journal of Social Robotics*, *5*(2), 291–308.

[29] Kidd, C. D. & Breazeal, C. (2008). Robots at home: Understanding long-term human-robot interaction. In *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS 2008)* (pp. 3230–3235).

[30] Dahl, T. S. & Kamel Boulos, M. N. (2014). Robots in health and social care: A complementary technology to home care and telehealthcare? *Robotics*, *3*(1), 1–21.

[31] Patten, L., Evans, L., Oshinowo, L., Ochisor, M., Kazuharu, N., Lodewijk, A. & Tabarah, E. (2002). International space station robotics: A comparative study of ERA, JEMRMS and MSS. In *Proc. 7th ESA Workshop on Advanced Space Technologies for Robotics and Automation (ASTRA 2002)* (pp. 1.3–1).

[32] International Organization for Standardization. (2012). *ISO 8373:2012 Robots and robotics devices -- Vocabulary.*

[33] Scott, P. B. (1984). *The robotics revolution: The complete guide.* Oxford, England: Basil Blackwell Ltd.

[34] Asimov, I. (1983). The word I invented. In *Counting the eons* (pp. 30–39). New York, NY: Doubleday.

[35] Asimov, I. (2008). Runaround. In *I, robot* (pp. 25–45). New York, NY: Bantam.

[36] Asimov, I. (1985). *Robots and empire.* London, England: Granada.

[37] Engelberger, J. F. (1982). *Robotics in practice: Management and applications of industrial robots.* Great Britain: AMACOM.

[38] Waurzyniak, P. (2006). Masters of manufacturing: Joseph F. Engelberger. *Manufacturing Engineering*, *137*(1), 65–75.

[39] Devol, G. C., Jr. (1961). *Programmed article transfer.* US Patent 2988237, issued 13 June 1961.

[40] Engelberger, J. F. (1989). *Robotics in service.* Cambridge, MA: MIT Press.

[41] Saveriano, J. W. (1981). An interview with Joseph Engelberger. *Robotics Age*, *3*(1), 10–23, 38.

[42] Ernst, H. A. (1962). MH-1, a computer-operated mechanical hand. In *Proc. of the May 1-3, 1962, Spring Joint Computer Conf.* (pp. 39–51). AIEE-IRE '62. San Francisco, CA: ACM.

[43] Corker, K., Lyman, J. H. & Sheredos, S. (1979). A preliminary evaluation of remote medical manipulators. *Bulletin of Prosthetics Research*, *16*(2), 107–134.

[44] Leifer, L. (1981). Rehabilitative robots. *Robotics Age*, *3*(3), 4–15.

[45] Feldman, G. (1968). *Butterfinger* [16mm film]. Computer Science Department Film Collection, Stanford University Archives. Retrieved from https://lib.stanford.edu/ai-media/film-05

[46] Roth, B. (2008). Foreword. In B. Siciliano & O. Khatib (Eds.), *Springer handbook of robotics* (pp. v–ix). Berlin, Germany: Springer-Verlag.

[47] Pieper, D. L. (1968). *The kinematics of manipulators under computer control* (Ph.D. Thesis in Mechanical Engineering, Stanford University).

[48] Scheinman, V. D. (2010). *An oral history conducted by P. Asaro and S. Šabanović* [Interview transcript]. Robotics History Project. Retrieved from http://roboticshistory.indiana.edu/content/vic-scheinman

[49] Richards, M. (2007). *The Orm.* Computer History Museum. Retrieved from http://www.computerhistory.org/revolution/artificial-intelligence-robotics/13/293/1279

[50] Garcia-Molina, H. (2005). *Scheinman Golden Arm cabinet.* Stanford Computer History Exhibits Photographs 2. Retrieved from http://infolab.stanford.edu/pub/voy/museum/pictures/DisplayHector/112ArmCabintet.jpg

[51] Scheinman, V. D. (1969). *Design of computer controlled manipulator* (Eng. Thesis in Mechanical Engineering, Stanford University).

[52] Scheinman, V. D. (2011). *Stanford Hydraulic Arm.* Computer History Museum. Retrieved from http://www.computerhistory.org/collections/catalog/102689927

[53] Bolles, R. & Paul, R. (1973). *The use of sensory feedback in a programmable assembly system* [Memo AIM-220]. SAIL, Stanford University.

[54] Minsky, M. & Papert, S. (1970). *1968-1969 progress report* [AI Memo No. 200]. Massachusetts Institute of Technology.

[55] Gresser, J.-Y. (1968). *Description and control of manipulation by computer-controlled arm* [AI Memo No. 165]. Massachusetts Institute of Technology.

[56] Minsky, M. (1979). *Toward a remotely-manned energy and production economy* [AI Memo No. 544]. Massachusetts Institute of Technology.

[57] MIT AI Lab. (1968). *AI history: Minsky tentacle arm* [Film]. Retrieved from http://video.mit.edu/watch/ai-history-minsky-tentacle-arm-12074/

[58] Minsky, M. & Papert, S. (1973). *Proposal to ARPA for continuation of micro-automation development* [AI Memo No. 274]. Massachusetts Institute of Technology.

[59] Horn, B. K. P. & Inoue, H. (1974). *Kinematics of the MIT-AI-Vicarm manipulator* [Working Paper 69]. Massachusetts Institute of Technology.

[60] Nilsson, N. J., Barrow, H. G., Coles, L. S., Gleason, G., Meyer, B. & Nitzan, D. (1973). *Study of equipment needs for artificial intelligence research* [Interim report]. SRI, Stanford University.

[61] Silver, D. (1973). *The little robot system* [AI Memo No. 273]. Massachusetts Institute of Technology.

[62] Barrow, H. G. & Salter, S. H. (1969). Design of low-cost equipment for cognitive robot research. In B. Meltzer & D. Michie (Eds.), *Machine intelligence 5* (pp. 555–566). Edinburgh University Press.

[63] Barrow, H. G. & Crawford, G. F. (1972). The Mark 1.5 Edinburgh robot facility. In B. Meltzer & D. Michie (Eds.), *Machine intelligence 7* (pp. 465–480). Edinburgh University Press.

[64] Ambler, A. P., Barrow, H. G., Brown, C. M., Burstall, R. M. & Popplestone, R. J. (1975). A versatile system for computer-controlled assembly. *Artificial Intelligence*, *6*(2), 129–156.

[65] Yin, B. (1984). *Combining vision verification with a high level robot programming language* (PhD Thesis, University of Edinburgh).

[66] Ambler, A. P. & Popplestone, R. J. (1975). Inferring the positions of bodies from specified spatial relationships. *Artificial Intelligence*, *6*(2), 157–174.

[67] Makino, H., Kato, A. & Yamazaki, Y. (2007). Review: Research and commercialization of SCARA robot - The case of industry-university joint research and development. *Int. J. of Automation Technology*, *1*(1), 61–67. [English translation] Original Japanese article appeared in (2005) *J. of the Robotics Society of Japan, 23*(2), 148-154.

[68] Lasky, T. A. & Hsia, T. C. (2006). Robot configuration. In R. C. Dorf (Ed.), *Systems, controls, embedded systems, energy, and machines.* Boca Raton, FL: CRC Press.

[69] Associated Press. (1982). Sale of Unimation to Westinghouse. *New York Times.* Retrieved from www.nytimes.com

[70] Associated Press. (1988). Company news; Westinghouse to sell Unimation to Staubli. *New York Times.* Retrieved from www.nytimes.com

[71] International Federation of Robotics. (2014). Executive summary. In *World robotics - Industrial robots 2014: Statistics, market analysis, forecasts and case studies* (pp. 11–24).

[72] United Nations Economic Commission for Europe. (2000). *The boom in robot investment continues - 900,000 industrial robots by 2003: UN/ECE issues its 2000 world robotics survey* [Press release]. Retrieved from http://www.unece.org/press/pr2000/00stat10e.html

[73] International Federation of Robotics & United Nations Economic Commission for Europe. (2003). Executive summary. In *World robotics 2003: Statistics, market analysis, forecasts, case studies and profitability of robot investment* (pp. 1–10). Geneva, Switzerland: United Nations.

[74] International Federation of Robotics & United Nations Economic Commission for Europe. (2004). Executive summary. In *World robotics 2004: Statistics, market analysis, forecasts, case studies and profitability of robot investment* (pp. 1–12). Geneva, Switzerland: United Nations.

[75] International Federation of Robotics & United Nations Economic Commission for Europe. (2005). Executive summary. In *World robotics 2005: Statistics, market analysis, forecasts, case studies and profitability of robot investment* (pp. 1–14). Geneva, Switzerland: United Nations.

[76] International Federation of Robotics. (2006). Executive summary. In *World robotics 2006: Statistics, market analysis, forecasts and case studies and profitability of robot investment* (pp. iii–x).

[77] International Federation of Robotics. (2007). Executive summary. In *World robotics 2007: Statistics, market analysis, forecasts and case studies and profitability of robot investment* (pp. ix–xv).

[78] International Federation of Robotics. (2008). *World robotics 2008: Welcome to the press conference* [Presentation slides].

[79] International Federation of Robotics. (2009). Executive summary. In *World robotics - Industrial robots 2009: Statistics, market analysis, forecasts and case studies* (pp. ix–xvi).

[80] International Federation of Robotics. (2010). Executive summary. In *World robotics - Industrial robots 2010: Statistics, market analysis, forecasts and case studies* (pp. vii–xiv).

[81] International Federation of Robotics. (2012). Executive summary. In *World robotics - Industrial robots 2012: Statistics, market analysis, forecasts and case studies* (pp. 8–18).

[82] International Federation of Robotics. (2013). Executive summary. In *World robotics - Industrial robots 2013: Statistics, market analysis, forecasts and case studies* (pp. 10–21).

[83] International Federation of Robotics. (2015). Executive summary. In *World robotics - Industrial robots 2015: Statistics, market analysis, forecasts and case studies* (pp. 13–26).

[84] Moore, G. E. (1965). Cramming more components onto integrated circuits. *Electronics*, *38*(8), 114–117.

[85] Lukka, T. J., Tossavainen, T., Kujala, J. V. & Raiko, T. (2014). ZenRobotics Recycler - Robotic sorting using machine learning. In *Proc. Sensor-Based Sorting 2014 Conf.* (pp. 169–176).

[86] Bac, C. W., van Henten, E. J., Hemming, J. & Edan, Y. (2014). Harvesting robots for high-value crops: State-of-the-art review and challenges ahead. *Journal of Field Robotics*, *31*(6), 888–911.

[87] Moustris, G. P., Hiridis, S. C., Deliparaschos, K. M. & Konstantinidis, K. M. (2011). Evolution of autonomous and semi-autonomous robotic surgical systems: A review of the literature. *The International Journal of Medical Robotics and Computer Assisted Surgery*, *7*(4), 375–392.

[88] Kilby, W., Dooley, J. R., Kuduvalli, G., Sayeh, S. & Maurer, C. R. (2010). The CyberKnife® robotic radiosurgery system in 2010. *Technology in Cancer Research & Treatment*, *9*(5), 433–452.

[89] Stoianovici, D., Kim, C., Srimathveeravalli, G., Sebrecht, P., Petrisor, D., Coleman, J., … Hricak, H. (2014). MRI-safe robot for endorectal prostate biopsy. *IEEE/ASME Trans. Mechatronics*, *19*(4), 1289–1299.

[90] Kazanzides, P., Mittelstadt, B. D., Musits, B. L., Bargar, W. L., Zuhars, J. F., Williamson, B., … Carbone, E. J. (1995). An integrated system for cementless hip replacement. *IEEE Engineering in Medicine and Biology Mag. 14*(3), 307–313.

[91] Meldrum, D., Glennon, A., Herdman, S., Murray, D. & McConn-Walsh, R. (2012). Virtual reality rehabilitation of balance: Assessment of the usability of the Nintendo Wii® Fit Plus. *Disability and Rehabilitation: Assistive Technology*, *7*(3), 205–210.

[92] Laver, K. E., George, S., Thomas, S., Deutsch, J. E. & Crotty, M. (2015). Virtual reality for stroke rehabilitation. *Cochrane Database of Systematic Reviews*, (2), 1–84.

[93] Mine, M. (2003). Towards virtual reality for the masses: 10 years of research at Disney's VR studio. In *Proceedings of the workshop on virtual environments* (pp. 11–17, 329–330). EGVE '03.

[94] Gu, N., Kim, M. J. & Maher, M. L. (2011). Technological advancements in synchronous collaboration: The effect of 3D virtual worlds and tangible user interfaces on architectural design. *Automation in Construction*, *20*(3), 270–278.

[95] Sacks, R., Whyte, J., Swissa, D., Raviv, G., Zhou, W. & Shapira, A. (2015). Safety by design: Dialogues between designers and builders using virtual reality. *Construction Management and Economics*, *33*(1), 55–72.

[96] Pausch, R., Crea, T. & Conway, M. (1992). A literature survey for virtual environments: Military flight simulator visual systems and simulator

sickness. *Presence: Teleoperators and Virtual Environments*, *1*(3), 344–363.

[97] Slator, B. M., Clark, J. T., Landrum, J., III, Bergstrom, A., Hawley, J., Johnston, E. & Fisher, S. (2001). Teaching with immersive virtual archaeology. In *Proceedings of the 7th international conference on virtual systems and multimedia* (pp. 253–262).

[98] Guttentag, D. A. (2010). Virtual reality: Applications and implications for tourism. *Tourism Management*, *31*(5), 637–651.

[99] Nicholson, D. T., Chalk, C., Funnell, W. R. J. & Daniel, S. J. (2006). Can virtual reality improve anatomy education? A randomised controlled study of a computer-generated three-dimensional anatomical ear model. *Medical Education*, *40*(11), 1081–1087.

[100] Yammine, K. & Violato, C. (2015). A meta-analysis of the educational effectiveness of three-dimensional visualization technologies in teaching anatomy. *Anatomical Sciences Education*, *8*(6), 525–538.

[101] Gurusamy, K., Aggarwal, R., Palanivelu, L. & Davidson, B. R. (2008). Systematic review of randomized controlled trials on the effectiveness of virtual reality training for laparoscopic surgery. *British Journal of Surgery*, *95*(9), 1088–1097.

[102] Brooks, F. P., Jr., Ouh-Young, M., Batter, J. J. & Kilpatrick, P. J. (1990). Project GROPE - Haptic displays for scientific visualization. *ACM SIGGRAPH Computer Graphics*, *24*(4), 177–185.

[103] Potkonjak, V., Gardner, M., Callaghan, V., Mattila, P., Guetl, C., Petrović, V. M. & Jovanović, K. (2016). Virtual laboratories for education in science, technology, and engineering: A review. *Computers & Education*, *95*, 309–327.

[104] Artaud, A. (1958). *The theater and its double* (M. C. Richards, Trans.). New York, NY: Grove.

[105] McCollum, H. J. D. N. (1945). *Stereoscopic television apparatus.* US Patent 2388170, issued 30 October 1945.

[106] Heilig, M. L. (1960). *Stereoscopic-television apparatus for individual use.* US Patent 2955156, issued 4 October 1960.

[107] Heilig, M. L. (1962). *Sensorama simulator.* US Patent 3050870, issued 28 August 1962.

[108] Sutherland, I. E. (1965). The ultimate display. In *Information processing 1965: Proceedings of IFIP congress* (Vol. 2, pp. 506–508).

[109] Sutherland, I. E. (1968). A head-mounted three dimensional display. In *AFIPS Fall joint computer conference proceedings, Part I* (Vol. 33, pp. 757–764).

[110] Zimmerman, T. G., Lanier, J., Blanchard, C., Bryson, S. & Harvill, Y. (1987). A hand gesture interface device. *ACM SIGCHI Bulletin*, *18*(4), 189–192.

[111] Blanchard, C., Burgess, S., Harvill, Y., Lanier, J., Lasko, A., Oberman, M. & Teitel, M. (1990). Reality built for two: A virtual reality tool. *ACM SIGGRAPH Computer Graphics*, *24*(2), 35–36.

[112] Lanier, J. (2001). Virtually there. *Scientific American*, *284*(4), 66–75.

[113] Oxford English Dictionary. (2014). virtual reality, n. Retrieved from http://www.oed.com

[114] Cruz-Neira, C., Sandin, D. J. & DeFanti, T. A. (1993). Surround-screen projection-based virtual reality: The design and implementation of the CAVE. In *Proceedings of the 20th annual conference on computer graphics and interactive techniques* (pp. 135–142). SIGGRAPH '93.

[115] Sturman, D. J. & Zeltzer, D. (1994). A survey of glove-based input. *IEEE Computer Graphics and Applications*, *14*(1), 30–39.

[116] Roetenberg, D., Luinge, H. & Slycke, P. (2009). *Xsens MVN: Full 6DOF human motion tracking using miniature inertial sensors* [Technical report]. Xsens Technologies BV.

[117] Souman, J. L., Robuffo Giordano, P., Schwaiger, M., Frissen, I., Thümmel, T., Ulbrich, H., … Ernst, M. O. (2011). CyberWalk: Enabling unconstrained omnidirectional walking through virtual environments. *ACM Transactions on Applied Perception*, *8*(4), 25:1–25:22.

[118] Moeslund, T. B., Hilton, A. & Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, *104*(2-3), 90–126.

[119] Shiratori, T., Park, H. S., Sigal, L., Sheikh, Y. & Hodgins, J. K. (2011). Motion capture from body-mounted cameras. *ACM Transactions on Graphics*, *30*(4), 31:1–31:10.

[120] Ijaz, F., Yang, H. K., Ahmad, A. W. & Lee, C. (2013). Indoor positioning: A review of indoor ultrasonic positioning systems. In *15th international conference on advanced communication technology (ICACT)* (pp. 1146–1150).

[121] Steuer, J. (1992). Defining virtual reality: Dimensions determining telepresence. *Journal of Communication*, *42*(4), 73–93.

[122] Evans, A., Romeo, M., Bahrehmand, A., Agenjo, J. & Blat, J. (2014). 3D graphics on the web: A survey. *Computers & Graphics*, *41*, 43–61.

[123] Segal, M. & Akeley, K. (2015). *The OpenGL® graphics system: A specification (version 4.5 (core profile) - May 28, 2015).* Khronos Group.

[124] Segal, M. & Akeley, K. (1992). *The OpenGL^{TM} graphics system: A specification (version 1.0).* Silicon Graphics Inc.

[125] Sowizral, H., Rushforth, K. & Deering, M. (2000). *The Java 3D^{TM} API specification* (2nd). Boston, MA: Addison-Wesley.

[126] International Organization for Standardization. (1997). *ISO/IEC 14772-1:1997 Information technology – Computer graphics and image processing – The Virtual Reality Modeling Language (VRML) – Part 1: Functional specification and UTF-8 encoding.*

[127] International Organization for Standardization. (2013). *ISO/IEC 19775-1:2013 Information technology – Computer graphics, image processing and environmental data representation – Extensible 3D (X3D) – Part 1: Architecture and base components.*

[128] International Organization for Standardization. (2009). *ISO/IEC 19776-1.2:2009 Information technology – Computer graphics and image processing – Extensible 3D (X3D) encodings – Part 1: Extensible Markup Language (XML) encoding.*

[129] International Organization for Standardization. (2015a). *ISO/IEC 19776-2:2015 Information technology – Computer graphics, image processing and environmental data representation – Extensible 3D (X3D) encodings – Part 2: Classic VRML encoding.*

[130] International Organization for Standardization. (2015b). *ISO/IEC 19776-3:2015 Information technology – Computer graphics, image processing and environmental data representation – Extensible 3D (X3D) encodings – Part 3: Compressed binary encoding.*

[131] Khronos Group. (2014). *WebGL specification, version 1.0.3, 27 October 2014*. Retrieved from https://www.khronos.org/registry/webgl/specs/1.0.3/

[132] International Organization for Standardization. (2004). *ISO/IEC 14772-2:2004 Information technology – Computer graphics and image processing – The Virtual Reality Modeling Language (VRML) – Part 2: External authoring interface (EAI).*

[133] Gaffney, M., Walsh, M., O'Connell, S., Wang, B., O'Flynn, B. & Ó Mathúna, C. (2011). A smart wireless inertial measurement unit system: Simplifying & encouraging usage of WIMU technology. In *5th interna-*

tional conference on pervasive computing technologies for healthcare and workshops (pp. 198–199).

[134] Schmidt, G. T. (2010). *INS/GPS technology trends* [RTO-EN-SET-116 Lecture Series].

[135] Holley, M. D., Swingle, W. L., Bachman, S. L., LeBlanc, C. J., Howard, H. T. & Biggs, H. M. (1976). *Apollo experience report - guidance and control systems: Primary guidance, navigation, and control system development* [NASA Technical Note].

[136] Bellis, S. J., Delaney, K., O'Flynn, B., Barton, J., Razeeb, K. M. & Ó Mathúna, C. (2005). Development of field programmable modular wireless sensor network nodes for ambient systems. *Computer Communications*, *28*(13), 1531–1544.

[137] Kennedy, D., Walsh, M. & O'Flynn, B. (2014). Development of a miniature, low-cost wave measurement solution. In *MTS/IEEE oceans - St. John's* (pp. 1–6).

[138] Marinković, S., Puppo, R., Pan, R. L. C. & Popovici, E. (2010). Implementation and testing of a secure fall detection system for body area networks. In *Proceedings of 27th international conference on microelectronics* (pp. 315–318).

[139] Walsh, M., O'Flynn, B., Ó Mathúna, C., Hickey, A. & Kellett, J. (2013). Evolving ambient intelligence. In M. J. O'Grady, H. Vahdat-Nejad, K.-H. Wolf, M. Dragone, J. Ye, C. Röcker & G. O'Hare (Eds.), (Chap. Correlating Average Cumulative Movement and Barthel Index in Acute Elderly Care, pp. 54–63). Communications in Computer and Information Science. Cham, Switzerland: Springer.

[140] Walsh, M., Barton, J., O'Flynn, B., Ó Mathúna, C. & Tyndyk, M. (2011). Capturing the overarm throw in darts employing wireless inertial measurement. In *IEEE sensors* (pp. 1441–1444).

[141] Connaghan, D., Kelly, P., O'Connor, N. E., Gaffney, M., Walsh, M. & Ó Mathúna, C. (2011). Multi-sensor classification of tennis strokes. In *IEEE sensors* (pp. 1437–1440).

[142] Gaffney, M., Walsh, M., O'Flynn, B. & Ó Mathúna, C. (2015). A highly automated, wireless inertial measurement unit based system for monitoring gym-based push-start training sessions by bob-skeleton athletes. *Sensors & Transducers*, *184*(1), 26–38.

[143] O'Flynn, B., Torre, G., Fernstrom, M., Winkler, T., Lynch, A., Barton, J., . . . Ó Mathúna, S. C. (2007). 4th international workshop on wearable and implantable body sensor networks. In S. Leonhardt, T. Falck & P. Mähönen (Eds.), (Chap. Celeritas – A Wearable Sensor System for Interactive Digital Dance Theatre, pp. 161–165). IFMBE Proceedings. Berlin, Germany: Springer.

[144] Bacigalupo, A. & Ricci, S. (2012). *Body area networks for gesture recognition using wireless inertial measurement unit* (M.Sc. Thesis in Electronic Engineering, University of Genoa and University College Cork).

[145] Analog Devices. (2011). *ADXL345: 3-axis, ±2g/±4g/±8g/±16g, digital accelerometer (rev. c).*

[146] Nidec Sankyo. (2005a). *SC3150 controller hardware manual: SC3000 series robot systems.*

[147] Nidec Sankyo. (2005b). *Installation manual (rev. 3): SC3000 series robot systems.*

[148] Nidec Sankyo. (2005c). *Pendant operation manual (rev. 3): SC3000 series robot systems.*

[149] Sankyo Seiki Manufacturing. (2000). *SCARA series hardware manual (version 2.00): SC3000 robot systems.*

[150] Shiakolas, P. S., Conrad, K. L. & Yih, T. C. (2002). On the accuracy, repeatability, and degree of influence of kinematics parameters for industrial robots. *International Journal of Modelling and Simulation*, *22*(4), 245–254.

[151] Nidec Sankyo. (2005d). *SSL/E language reference manual: SC3000 robot systems.*

[152] Ferrari, L. E. (2014). *Matlab-based control of a SCARA robot* (M.Sc. Thesis in Industrial Engineering, University of Padua and University College Cork).

[153] Linn, M. C. & Petersen, A. C. (1985). Emergence and characterization of sex differences in spatial ability: A meta-analysis. *Society for Research in Child Development*, *56*(6), 1479–1498.

[154] Onyancha, R. M., Derov, M. & Kinsey, B. L. (2009). Improvements in spatial ability as a result of targeted training and computer-aided design software use: Analyses of object geometries and rotation types. *Journal of Engineering Education*, *98*(2), 157–167.

[155] Bruder, S. & Wedeward, K. (2007). An interactive online robotics course. *Intelligent Automation & Soft Computing*, *13*(1), 105–116.

[156] Branoff, T. J. (2000). Spatial visualization measurement: A modification of the Purdue spatial visualization test - visualization of rotations. *Engineering Design Graphics Journal*, *64*(2), 14–22.

[157] Sorby, S. A. (1999). Developing 3D spatial visualization skills. *Engineering Design Graphics Journal*, *63*(2), 21–32.

[158] Gutiérrez, A. (1996). Visualization in 3-dimensional geometry: In search of a framework. In L. Puig & A. Gutiérrez (Eds.), *Proc. 20th conference of the international group for the psychology of mathematics education* (Vol. 1, pp. 3–19).

[159] Wai, J., Lubinski, D. & Benbow, C. P. (2009). Spatial ability for STEM domains: Aligning over 50 years of cumulative psychological knowledge solidifies its importance. *Journal of Educational Psychology*, *101*(4), 817–835.

[160] Sorby, S. A. (2009). Educational research in developing 3-D spatial skills for engineering students. *International Journal of Science Education*, *31*(3), 459–480.

[161] Sorby, S. A., Casey, B., Veurink, N. & Dulaney, A. (2013). The role of spatial training in improving spatial and calculus performance in engineering students. *Learning and Individual Differences*, *26*, 20–29.

[162] Sorby, S. A. & Wysocki, A. F. (2003). *Introduction to 3D spatial visualization: An active approach.* Clifton Park, NY: Thomson Delmar Learning.

[163] Blasko, D. G., Holliday-Darr, K., Mace, D. & Blasko-Drabik, H. (2004). VIZ: The visualization assessment and training web site. *Behavior Research Methods, Instruments, & Computers*, *36*(2), 256–260.

[164] Veurink, N. L., Hamlin, A. J., Kampe, J. C. M., Sorby, S. A., Blasko, D. G., Holliday-Darr, K. A., . . . Knott, T. W. (2009). Enhancing Visualization Skills-Improving Options aNd Success (EnViSIONS) of engineering and technology students. *Engineering Design Graphics Journal*, *73*(2), 1–17.

[165] Sorby, S. A. (2011). *Improving spatial visualization skills* [PowerPoint Slides]. http://www.engageengineering.org/associations/11559/files/ENGAGE Improving Spatial Visualization Skills Webinar JAN 27 2011.pdf.

[166] Martin-Dorta, N., Saorin, J. L. & Contero, M. (2011). Web-based spatial training using handheld touch screen devices. *Educational Technology & Society*, *14*(3), 163–177.

[167] Rafi, A. & Samsudin, K. (2009). Practising mental rotation using interactive Desktop Mental Rotation Trainer (iDeMRT). *British Journal of Educational Technology*, *40*(5), 889–900.

[168] Samsudin, K., Rafi, A. & Hanif, A. S. (2011). Training in mental rotation and spatial visualization and its impact on orthographic drawing performance. *Educational Technology & Society*, *14*(1), 179–186.

[169] Hsi, S., Linn, M. C. & Bell, J. E. (1997). The role of spatial reasoning in engineering and the design of spatial instruction. *Journal of Engineering Education*, *86*(2), 151–158.

[170] Kinsey, B. L., Towle, E. & Onyancha, R. M. (2008). Improvement of spatial ability using innovative tools: Alternative view screen and physical model rotator. *Engineering Design Graphics Journal*, *72*(1), 1–8.

[171] Wang, C.-Y., Yeh, S.-C., Wang, J.-L., Yang, S.-C. & Rizzo, A. (2011). A 3D motion controlled method for the training of mental rotation. In I. Aedo, N.-S. Chen, D. G. Sampson, J. M. Spector & Kinshuk (Eds.), *Proc. 11th IEEE international conference on advanced learning technologies (ICALT)* (pp. 1–3).

[172] Price, A. & Lee, H.-S. (2010). The effect of two-dimensional and stereoscopic presentation on middle school students' performance of spatial cognition tasks. *Journal of Science Education and Technology*, *19*(1), 90–103.

[173] Yeh, A. (2010). Three primary school students' cognition about 3D rotation in a virtual reality learning environment. In L. Sparrow, B. Kissane & C. Hurst (Eds.), *Shaping the future of mathematics education: Proceedings of the 33rd annual conference of the mathematics education research group of australasia (MERGA)* (pp. 690–697).

[174] Manseur, R. (2006). *Robot modeling and kinematics*. Boston, MA: Charles River Media.

[175] Paul, R. P. (1981). *Robot manipulators: Mathematics, programming, and control.* Cambridge, MA: MIT press.

[176] Guay, R. B. (1976). Purdue spatial visualization tests: Visualization of rotations. West Lafayette, IN: Purdue Research Foundation.

[177] Yue, J. (2007). Spatial visualization by isometric view. *Engineering Design Graphics Journal*, *71*(2), 5–19.

[178] Ekstrom, R. B., French, J. W., Harman, H. H. & Dermen, D. (1976). *Manual for kit of factor-referenced cognitive tests.* Princeton, NJ: Educational Testing Service.

[179] Gittler, G. & Glück, J. (1998). Differential transfer of learning: Effects of instruction in descriptive geometry on spatial test performance. *Journal for Geometry and Graphics*, *2*(1), 71–84.

[180] Vandenberg, S. G. & Kuse, A. R. (1978). Mental rotations, A group test of three-dimensional spatial visualization. *Perceptual and Motor Skills*, *47*(2), 599–604.

[181] Branoff, T. J. (1998). The effects of adding coordinate axes to a mental rotations task in measuring spatial visualization ability in introductory undergraduate technical graphics courses. *Engineering Design Graphics Journal*, *62*(2), 16–34.

[182] Corke, P. I. (2007). A simple and systematic approach to assigning Denavit-Hartenberg parameters. *IEEE Trans. Robotics*, *23*(3), 590–594.

[183] Schilling, R. J. (1990). *Fundamentals of robotics: Analysis and control.* Englewood Cliffs, NJ: Prentice Hall.

[184] Spong, M. W., Hutchinson, S. & Vidyasagar, M. (2006). *Robot modeling and control.* New York, NY: John Wiley & Sons.

[185] Nethery, J. F. & Spong, M. W. (1994). Robotica: A Mathematica package for robot analysis. *IEEE Robotics & Automation Magazine*, *1*(1), 13–20.

[186] Corke, P. I. (1996). A robotics toolbox for MATLAB. *IEEE Robotics & Automation Magazine*, *3*(1), 24–32.

[187] Aliane, N. (2011). Teaching fundamentals of robotics to computer scientists. *Computer Applications in Engineering Education*, *19*(3), 615–620.

[188] Falconi, R. & Melchiorri, C. (2008). RobotiCad: An educational tool for robotics. In *Proc. 17th international federation of automatic control world congress* (pp. 9111–9116).

[189] Lobov, A., Lastra, J. L. M. & Tuokko, R. (2003). A collaborative framework for learning robot mechanics: RIO-Robotics Illustrative sOftware. In *Proc. 33rd ASEE/IEEE annual frontiers in education* (Vol. 2, F4E:12–16).

[190] Jara, C. A., Candelas, F. A., Puente, S. T. & Torres, F. (2011). Hands-on experiences of undergraduate students in automatics and robotics using a virtual and remote laboratory. *Computers & Education*, *57*(4), 2451–2461.

[191] Marín, R., Sanz, P. J. & del Pobil, A. P. (2003). The UJI online robot: An education and training experience. *Autonomous Robots*, *15*(3), 283–297.

[192] Rohrmeier, M. (2000). Web based robot simulation using VRML. In *Proc. 32nd winter simulation conference* (Vol. 2, pp. 1525–1528).

[193] Violante, M. G. & Vezzetti, E. (2015). Virtual interactive e-learning application: An evaluation of the student satisfaction. *Computer Applications in Engineering Education*, *23*(1), 72–91.

[194] Jara, C. A., Candelas, F. A., Pomares, J. & Torres, F. (2013). Java software platform for the development of advanced robotic virtual laboratories. *Computer Applications in Engineering Education*, *21*(S1), E14–E30.

[195] Gil, A., Reinoso, O., Marin, J. M., Paya, L. & Ruiz, J. (2015). Development and deployment of a new robotics toolbox for education. *Computer Applications in Engineering Education*, *23*(3), 443–454.

[196] López-Nicolás, G., Romeo, A. & Guerrero, J. J. (2014). Active learning in robotics based on simulation tools. *Computer Applications in Engineering Education*, *22*(3), 509–515.

[197] Cakir, M. & Butun, E. (2007). An educational tool for 6-DOF industrial robots with quaternion algebra. *Computer Applications in Engineering Education*, *15*(2), 143–154.

[198] Kucuk, S. & Bingul, Z. (2010). An off-line robot simulation toolbox. *Computer Applications in Engineering Education*, *18*(1), 41–52.

[199] Mateo Sanguino, T. J. & Andújar Márquez, J. M. (2012). Simulation tool for teaching and learning 3D kinematics workspaces of serial robotic arms with up to 5-DOF. *Computer Applications in Engineering Education*, *20*(4), 750–761.

[200] Robinette, M. F. & Manseur, R. (2001). ROBOT-DRAW, an Internet-based visualization tool for robotics education. *IEEE Transactions on Education*, *44*(1), 29–34.

[201] Bahuguna, J., Chittawadigi, R. G. & Saha, S. K. (2013). Teaching and learning of robot kinematics using RoboAnalyzer software. In *Proc. of conference on advances in robotics* (pp. 1–6).

[202] *RoboAnalyzer*. (2014). Last accessed in March 2015. Retrieved from http://www.roboanalyzer.com/downloads.html

[203] González-Palacios, M. A., González-Barbosa, E. A. & Aguilera-Cortés, L. A. (2013). SnAM: A simulation software on serial manipulators. *Engineering with Computers*, *29*(1), 87–94.

[204] Afshari, K. E. & Payandeh, S. (1999). *Toward implementation of Java/VRML environment for planning, training and tele-operation of*

robotic systems. Presented at 3rd World Multiconference on Systemics, Cybernetics and Informatics.

[205] Cheng, P. & Oelmann, B. (2010). Joint-angle measurement using accelerometers and gyroscopes – A survey. *IEEE Transactions on Instrumentation and Measurement*, *59*(2), 404–414.

[206] Quigley, M., Brewer, R., Soundararaj, S. P., Pradeep, V., Le, Q. & Ng, A. Y. (2010). Low-cost accelerometers for robotic manipulator perception. In *2010 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 6168–6174).

[207] Cantelli, L., Muscato, G., Nunnari, M. & Spina, D. (2015). A joint-angle estimation method for industrial manipulators using inertial sensors. *IEEE/ASME Transactions on Mechatronics*, *20*(5), 2486–2495.

[208] Axelsson, P. & Norrlöf, M. (2012). Method to estimate the position and orientation of a triaxial accelerometer mounted to an industrial manipulator. In *10th IFAC symposium on robot control* (pp. 283–288).

[209] Canepa, G., Hollerbach, J. M. & Boelen, A. J. M. A. (1994). Kinematic calibration by means of a triaxial accelerometer. In *1994 IEEE international conference on robotics and automation (ICRA)* (Vol. 4, pp. 2776–2782).

[210] Croxell, J. R., Weinberg, J. B., Krauss, R. W. & Smith, S. R. (2008). Robotic limb calibration: Accelerometer based discovery of kinematic constants. In *Technical report (WS-08-08) of 2008 Association for the Advancement of Artificial Intelligence (AAAI-08) workshop on mobile robotics: Mobility and manipulation* (pp. 1–4).

[211] Ciarleglio, C. (2013). *Kinematic determination of an unmodeled serial manipulator by means of an IMU* (Master's Thesis in Aerospace Engineering, University of Maryland).

[212] D'Amore, N., Ciarleglio, C. & Akin, D. L. (2015). IMU-based manipulator kinematic identification. In *2015 IEEE international conference on robotics and automation (ICRA)* (pp. 1437–1441).

[213] Wieser, E., Mittendorfer, P. & Cheng, G. (2011). Accelerometer based robotic joint orientation estimation. In *11th IEEE-RAS international conference on humanoid robots (humanoids)* (pp. 67–74).

[214] Fisher, C. J. (2010). *AN-1057: Using an accelerometer for inclination sensing*. Analog Devices Application Note.

[215] Skog, I. & Händel, P. (2006). Calibration of a MEMS inertial measurement unit. In *XVII International Measurement Federation (IMEKO) World Congress: Metrology for a sustainable development* (Vol. 2, pp. 1445–1450).

[216] Ang, W. T., Khosla, P. K. & Riviere, C. N. (2007). Nonlinear regression model of a low-g MEMS accelerometer. *IEEE Sensors Journal*, *7*(1), 81–88.

[217] Łuczak, S. (2015). Experimental studies of hysteresis in MEMS accelerometers: A commentary. *IEEE Sensors Journal*, *15*(6), 3492–3499.

[218] Aggarwal, P., Syed, Z., Niu, X. & El-Sheimy, N. (2008). A standard testing and calibration procedure for low cost MEMS inertial sensors and units. *The Journal of Navigation*, *61*(2), 323–336.

[219] Syed, Z. F., Aggarwal, P., Goodall, C., Niu, X. & El-Sheimy, N. (2007). A new multi-position calibration method for MEMS inertial navigation systems. *Measurement Science and Technology*, *18*(7), 1897–1907.

[220] Won, S.-h. P. & Golnaraghi, F. (2010). A triaxial accelerometer calibration method using a mathematical model. *IEEE Transactions on Instrumentation and Measurement*, *59*(8), 2144–2153.

[221] Tedaldi, D., Pretto, A. & Menegatti, E. (2014). A robust and easy to implement method for IMU calibration without external equipments. In

2014 IEEE international conference on robotics and automation (ICRA) (pp. 3042–3049).

[222] Renk, E. L., Collins, W., Rizzo, M., Lee, F. & Bernstein, D. S. (2005). Calibrating a triaxial accelerometer-magnetometer: Using robotic actuation for sensor reorientation during data collection. *IEEE Control Systems*, *25*(6), 86–95.

[223] Beravs, T., Podobnik, J. & Munih, M. (2012). Three-axial accelerometer calibration using Kalman filter covariance matrix for online estimation of optimal sensor orientation. *IEEE Transactions on Instrumentation and Measurement*, *61*(9), 2501–2511.

[224] Gaffney, M., Walsh, M., O'Flynn, B. & Ó Mathúna, C. (2011). An automated calibration tool for high performance wireless inertial measurement in professional sports. In *IEEE sensors* (pp. 262–265).

[225] Hendel, T. (2008). *Ellipse fit.* MATLAB Central File Exchange. Retrieved from https://uk.mathworks.com/matlabcentral/fileexchange/22423-ellipse-fit

[226] Lomb, N. R. (1976). Least-squares frequency analysis of unequally spaced data. *Astrophysics and Space Science*, *39*(2), 447–462.

[227] Scargle, J. D. (1982). Studies in astronomical time series analysis. II-Statistical aspects of spectral analysis of unevenly spaced data. *Astrophysical Journal*, *263*, 835–853.

[228] International Organization for Standardization. (2005). *ISO/IEC 19774: 2005 Information technology – Computer graphics and image processing – Humanoid animation (H-Anim).*

[229] Analog Devices. (2016). *ADXL313: 3-axis, ±0.5g/±1g/±2g/±4g, digital accelerometer (rev. b).*