



NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

DEVELOPMENT OF A 3-D PEN INPUT DEVICE

by

Deven A. Rhett

September 2008

Thesis Advisor:
Second Reader:

Xiaoping Yun
Roberto Cristi

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			<i>Form Approved OMB No. 0704-0188</i>	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE September 2008	3. REPORT TYPE AND DATES COVERED Master's Thesis	
4. TITLE AND SUBTITLE Development of a 3-D Pen Input Device			5. FUNDING NUMBERS	
6. AUTHOR(S) Deven A. Rhett, DOD Civilian				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited			12b. DISTRIBUTION CODE	
13. ABSTRACT (maximum 200 words) The principal objective of this research is to demonstrate the capability of obtaining the acceleration in the navigation frame of a unistroke which can be written on any surface or in the air while correcting integration errors from the measurements of the IMU (Inertial Measurement Unit) of the pen-type input devices. With the core topic of obtaining the acceleration while correcting integration errors, there are four subsidiary research questions relating to the pen-type input devices. First question is how to segment a stroke from the tip of the pen-type input device movement of the user. Second question is how to adjust the integration errors rapidly growing as time increases. Third question is how to reconstruct 2-D trajectory from 3-D trajectory in the air. Fourth question is how to project the trajectory onto the x-y plane.				
14. SUBJECT TERMS IMU, Inertial Measurement Unit, Pen-Type Input Device, Unistroke, Integration Errors, 2-D Trajectory, 3-D Trajectory			15. NUMBER OF PAGES 53	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UU	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

DEVELOPMENT OF A 3-D PEN INPUT DEVICE

Deven A. Rhett, Civilian, DoD, Washington, D.C.
B.A. and B.S., Gallaudet University, 2005

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

**NAVAL POSTGRADUATE SCHOOL
September 2008**

Author: Deven A. Rhett

Approved by: Xiaoping Yun
Thesis Advisor

Roberto Cristi
Second Reader

Jeffrey B. Knorr
Chairman, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The principal objective of this research is to demonstrate the capability of obtaining the acceleration in the navigation frame of a unistroke which can be written on any surface or in the air while correcting integration errors from the measurements of the IMU (Inertial Measurement Unit) of the pen-type input devices. With the core topic of obtaining the acceleration while correcting integration errors, there are four subsidiary research questions relating to the pen-type input devices. First question is how to segment a stroke from the tip of the pen-type input device movement of the user. Second question is how to adjust the integration errors rapidly growing as time increases. Third question is how to reconstruct 2-D trajectory from 3-D trajectory in the air. Fourth question is how to project the trajectory onto the x-y plane.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I.	FUNCTIONALITY OF 3-D PEN TYPE INPUT DEVICE	1
A.	PURPOSE OF PEN-TYPE INPUT DEVICE	1
B.	COMPARSION OF PEN-TYPE INPUT DEVICES.....	2
1.	EPOS Digital Pen & USB Flash Drive	2
2.	Logitech Pen	3
3.	Fly Fusion	4
4.	3-D Pen Input Device	4
II.	CRITERIA OF THE 3-D PEN INPUT DEVICE	7
A.	INTERFACE nIMU WITH A COMPUTER.....	8
B.	PROCESS RAW DATA THROUGH MATLAB.....	10
1.	Quaternion.....	11
a.	Quaternion Addition	11
b.	Quaternion Multiplication with Scalar	11
c.	Vector Cross Product	11
d.	Vector Dot Product.....	12
e.	Quaternion Multiplication with Two Quaternions.....	12
f.	Quaternion Norm.....	12
g.	Quaternion Complex Conjugate.....	12
h.	Quaternion Inverse	12
i.	Rotation Matrix From Quaternion	12
2.	Factored Quaternion Algorithm.....	13
a.	Elevation Quaternion.....	13
b.	Roll Quaternion.....	15
c.	Azimuth Quaternion	16
d.	Singularity Avoidance.....	18
III.	TRACKING ALGORITHM DEVELOPMENT	21
A.	SENSOR ORIENTATION.....	22
B.	EARTH COORDINATE SYSTEM	22
C.	VELOCITY OF SENSOR.....	23
D.	VELOCITY OF PEN BASED ON SENSOR	23
E.	DETECT PAUSE PHASES	25
F.	POSITION OF THE PEN TIP	25
IV.	RESULT OF RESEARCH.....	27
V.	CONCLUSION & RECOMMENDATIONS	31
A.	CONCLUSION	31
B.	RECOMMENDATIONS.....	31
	LIST OF REFERENCES.....	33
	INITIAL DISTRIBUTION LIST	35

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

Figure 1.	EPOS Digital Pen & USB Flash Memory From [1].	3
Figure 2.	Logitech Pen From [2].	3
Figure 3.	Fly Fusion Pen From [3].	4
Figure 4.	Power Supply, nIMU and USB interface board connectivity From [4].	8
Figure 5.	nIMU Data Console Main Menu.	9
Figure 6.	Raw Data obtained from nIMU sensor.	10
Figure 7.	Pitch, Roll, Yaw with nIMU sensor From [4].	13
Figure 8.	Earth-Fixed Coordinates $x_e y_e z_e$, Sensor Coordinates $x_s y_s z_s$, and Pen or marker Tip Coordinates $x_t y_t z_t$ From [10].	21
Figure 9.	MEMSense nIMU attach to a pen or marker From [10].	22
Figure 10.	MEMSense nIMU sensor and Compass set in North direction.	27
Figure 11.	Raw Data with Sensor point in North direction.	28
Figure 12.	Readability format of the Raw Data.	29
Figure 13.	Matlab Application of Quaternion Result.	29
Figure 14.	C++ Application of Quaternion result.	30

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

FQA	Factored Quaternion Algorithm
IMU	Inertial Measurement Unit
nIMU	nano Inertial Measurement Unit
NPS	Naval Postgraduate School
USB	Universal Serial Bus

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

The objective of this research is to develop a 3-D pen input device that is able to track a person's handwriting. With such a device, a person can write on a board, on a surface or in the air with no paper necessary and record the handwriting into a certain application which can be viewed through a graphical user interface. In order to accomplish this, a sensor will be attached to an existing writing instrument such as a pen, marker or a piece of stick. One application of such a device is for distance learning. When a teacher writes on the board in a classroom, his/her handwriting can be tracked in real time, and made available on the web to students at remote sites. Compared to other distance learning technologies such as video cameras, the 3-D pen input device would require much less bandwidth.

The sensor used for developing the 3-D pen input device is called the nano Inertial Measurement Unit (nIMU) from MEMSense. The nano Inertial Measurement Unit is small enough to be implemented into a pen-like input device. The first step is to interface a nIMU to a computer so that the computer can read data from the sensor in real time. Second, the sensor provides acceleration, angular rate, and a magnetic field measurement in the sensor coordinates. Before integrating acceleration to obtain velocity and position, the acceleration measurements must be transformed into a fixed earth-based coordinate system. To do so, it is necessary to estimate the orientation or attitude of the sensor at any given moment. Stroke segmentation, zero velocity compensation, imaginary writing plane and projection, and rotation transformation will be investigated. Third, in order to integrate the acceleration in an earth coordinate system it is necessary to determine the start and end time of the integration, i.e., determining the beginning and end of each writing stroke. Fourth, the computed hand trajectory from integration will contain errors, which should be removed as much as possible.

In order to approach this research, several steps are taken to accomplish its mission. First, a nIMU will be attached to any writing instrument. After securing the sensor onto a writing instrument, data will be collected from the sensor. The data

collected from the sensor are gyroscope, accelerometer, magnetometer, and counter for processing through three algorithms. The three algorithms are Quaternion, Factored Quaternion Algorithm (quaternion-based algorithm) and Tracking Algorithm Development. After processing and implementing the algorithms, the orientation of the pen will be estimated. The orientation represented in a quaternion will in turn be used to compute the position of the pen tip.

The objectives of the research are accomplished by processing and implementing the algorithms through Matlab and C++ applications. In order to validate the final result of this research, a Matlab application was used for offline processing. In the Matlab application, the Factored Quaternion Algorithm was tested in static configurations to verify the correct results. After obtaining the correct result of quaternion in static configurations, a C++ application was used for real-time processing. The quaternion result was verified in dynamic configurations and agrees with that produced by the Matlab application.

ACKNOWLEDGMENTS

The author would like to acknowledge the intellectual support, guidance and patience provided by Professor Xiaoping Yun, my thesis advisor, and James Calusdian, who is the lab director for servo control lab here at Naval Postgraduate School. They both have been contributing their knowledge with the nIMU and C++ programming. There are a number of people who I want to thank for all their help to ensure that I have available resources here at NPS to help my learning process here at NPS. They are Professor Jeffrey B. Knorr, who is the Chairman of Department of Electrical and Computer Engineering, Professor Roberto Cristi, who is my academic advisor, Nancy Weigle, the secretary of the Dean who provided essential help, and Danell Fanning, who is a wonderful sign language interpreter who gave her best effort to ensure that I understand in the classroom and outside the classroom. Finally, I would like to acknowledge my son, Kimane, who endured with his mother while studying at school even during the weekend and late evenings. Also last but not the least, my best friend Philip Njoku, my relatives Mary Page, Veronica Leurentop, and my parents Arthur & Sheila Rhett for the moral and psychological support. Thank you all for everything.

THIS PAGE INTENTIONALLY LEFT BLANK

I. FUNCTIONALITY OF 3-D PEN TYPE INPUT DEVICE

A. PURPOSE OF PEN-TYPE INPUT DEVICE

The purpose of using a pen-type input device is to improve communication with other people. Several types of people can benefit from the pen-type input device such as those in the defense community, distance-learning students, and/or hearing-impaired people. Here is an example of a scenario for each one of the three types of people that will use the pen-type input device.

With the defense community, for instance armed forces out in the battlefield, if one soldier is not able to communicate with one another or their command leader or vice versa, without giving away their location to their armed adversary, the pen-type input device can help. With the pen-type input device, the soldier can write in the air and be able to get the word across to another soldier or command leader and save himself and/or another from their armed adversary. It is important for soldiers and command leaders to be able to communicate efficiently with each other.

For distance-learning students who are not able to be present in the classroom, the pen-type input device can be quite handy for both the students and for the professors/teachers. In this case, the Naval Postgraduate School has distance-learning students from all across the country. With professors in the classroom, some of them will write vital information on the board to help the students understand the topic well. The distance-learning students would like to obtain the professor's notes from the board. In order for this to happen, the professors might have to copy/scan their notes and upload them into Blackboard or any other form of eLearning portal. However with a pen-type input device, the professors are saved the trouble of either rewriting their notes or copying and scanning the notes then uploading them into a form of eLearning, then broadcasting them for the distance learning students. Instead, the professors can use the device in the classroom at the same time as writing on the board for the distance-learning students and the students present in the classroom.

Hearing-impaired students who are profoundly deaf or hard of hearing, with sign language as their primary language, attend a university/school with hearing people who might not have an efficient way to communicate with them in the classroom. With the pen-type input device, this will be an advantage to both parties in the classroom. The student is required to take notes in class but at the same time, miss some valuable information because he/she is not observing the interpreter while the professor/teacher is speaking and writing on the board. For the hearing students this is not a problem, since they can at the same time, listen and take notes without looking at the professor/teacher. With the pen-type input devices, the deaf/hard of hearing student can observe the interpreter while the professor writes on the board and the software will record every stroke of the professor handwriting on the board onto the screen of the computer and save it.

B. COMPARSION OF PEN-TYPE INPUT DEVICES

After searching to see what other pen-type input devices are available on the market, we found three pen-type input devices, EPOS Digital Pen & USB Flash Drive, Logitech Digital Pen and Fly Fusion. In order to justify the need for another device, we will compare and contrast between our pen-type input device and the other three currently on the market.

1. EPOS Digital Pen & USB Flash Drive

With this digital pen, a user can write, sketch and/or draw on any type of paper and it is portable. However, the user is required to attach the USB Flash Drive on the top of the paper which captures and stores any written text and/or drawings in real-time to the Flash memory, as shown in Figure 1. In addition to the digital pen, it requires its own refill ink which can be quite expensive.



Figure 1. EPOS Digital Pen & USB Flash Memory From [1].

2. Logitech Pen

The Logitech pen, shown in Figure 2, works similar to the EPOS digital pen. It is portable and requires special ink. With the Logitech pen, the user is required to write on special paper, which is provided with the Logitech pen. However, it does not have a USB flash memory.



Figure 2. Logitech Pen From [2].

3. Fly Fusion

The Fly Fusion pen is a portable pen, which works similarly to the Logitech pen and requires writing on special paper. However, it does not have a USB flash memory to record the handwriting like the EPOS digital pen. The Fly Fusion pen has a tiny camera located at the tip of the pen, which reads the coordinates of the small dots that are located on the Fly Fusion paper. Figure 3 shows how the process works to accomplish its tasks.



Figure 3. Fly Fusion Pen From [3].

4. 3-D Pen Input Device

The proposed NPS 3-D pen input device used in this research contains some of the features described for the pen input devices above and important criteria which fit the user's needs for utilizing the 3-D pen input device. The NPS 3-D pen input device contains a gyroscope to record the movement of the handwriting stroke, and the ability to attach itself to any writing equipment. Thus, this 3-D pen input device does not need special ink, paper, or camera to accomplish its goals. The 3-D pen input device, meets the needs of the defense community. In order to achieve the 3-D pen input device, a sensor which is obtained from a company called MEMSense [4] was attached to existing writing

instruments such as a pen, marker or a piece of stick. The MEMSense sensor used in this research is called nano Inertial Measurement Unit (nIMU). With this nIMU sensor which is small enough to attach to a writing instrument, a person will be able to record handwriting when it is being used in the air, on the board or on any surface. Using the nIMU sensor, the capability of obtaining the acceleration in the navigation frame of a unistroke, which can be written on any surface or in the air while correcting integration errors from the measurements of the IMU (Inertial Measurement Unit), of the pen-type input devices was demonstrated? With the core topic of obtaining the acceleration while correcting integration errors, there are four subsidiary research questions relating to the pen-type input devices. First question is how to segment a stroke from the tip of the pen-type input device movement of the user. Second question is how to adjust the integration errors rapidly growing as time increases. Third question is how to reconstruct 2-D trajectory from 3-D trajectory in the air. Fourth question is how to project the trajectory onto the x-y plane.

THIS PAGE INTENTIONALLY LEFT BLANK

II. CRITERIA OF THE 3-D PEN INPUT DEVICE

In order for the 3-D pen input device to accomplish its goals, there are six steps that need to be taken before utilizing the device. The first step is to interface the nIMU with a computer so that it will be enabled to read raw data from the sensor in real time. The second step is to obtain the accelerometer, gyro and magnetometer measurements in the sensor coordinate system and estimate the sensor orientation in quaternion form. The third step is to obtain the acceleration in earth coordinate system then integrates. The fourth step is to compute hand trajectory from the integration and remove any errors as much as possible. The fifth step is to display the final hand trajectory on the computer screen in real time as one writes on the surface or in the air with the pen input device. The sixth step is to make the tracking results available for others through the web. However, in this research we will focus mainly on the first three steps of the criteria. Step four through six will be performed in subsequent research.

Within the first three steps, there are six sub steps to be performed before moving on to the fourth step. The first step is to obtain the accelerometer, gyro and magnetometer measurement in the sensor coordinate system, and to estimate the sensor orientation, is represented by a quaternion. The second step is to convert the accelerometer measurement from in the sensor coordinate system into the earth coordinate system using the quaternion rotation. The third step is integrating the accelerometer measurement which is in earth coordinate in order to obtain the velocity of the origin of the sensor coordinate system. The fourth step is to compute the velocity of the pen tip which is based on the origin of the sensor coordinate system and the angular velocity. The fifth step is to observe the pause phases in writing then apply the zero velocity correction. The sixth step is integrating the velocity of the pen tip in order to compute the

position of the pen tip. Before proceeding with this, it is necessary to become well acquainted with the background of quaternion mathematics using the software called Matlab. The purpose of using the software Matlab is to be familiar with the quaternion and how to apply it in an algorithm. This is followed by implementing an algorithm using quaternion arithmetic which will track the movement of the unistroke. After becoming well acquainted with the algorithm which is in Matlab code, it will be converted into C++ code using the Microsoft Visual Studio 2005.

A. INTERFACE nIMU WITH A COMPUTER

In order to interface the nIMU sensor with a computer, there are two other pieces of equipment required for hardware setup and software installation; a power supply which is set to 8.3 Volts and a USB which connects the nIMU sensor to the power supply and to the computer. Figure 4 shows the hardware setup and in Figure 5 the software interfaces to obtain raw data from the nIMU sensor.



Figure 4. Power Supply, nIMU and USB interface board connectivity From [4].

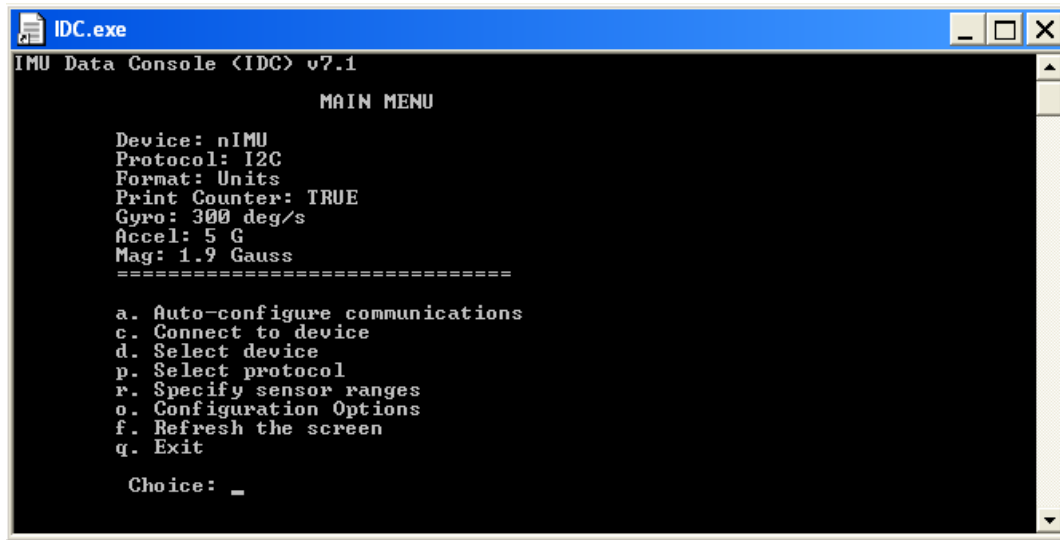


Figure 5. nIMU Data Console Main Menu.

In order to accomplish getting the right raw data from the right sensor, important details are needed. For instance, the sensor device which is used in this research is nIMU with the protocol I2C and set the data format in units along with the counter as shown in Figure 5.

After the hardware setup and the installation of the software to work with the nIMU sensor, the next step is to establish the auto-configuration communication (a) of the nIMU sensor. The purpose of auto-configuration communication between the nIMU sensor and the computer is to ensure that the IDC.exe is able to identify the device. After it has been identified, then the device is connected to the computer as stated in the data console menu (c). This is the part where the user obtains raw data from the nIMU sensor. An example of the raw data is shown in Figure 6.

0.00137329	-0.000640869	-0.00123596	-0.000457764	-0.00691223	0.19725	0.110596	-0.0271912
0.00183105	-0.00196838	-0.00109863	-0.000457764	-0.00682068	0.197159	0.109955	-0.0286102
0.00311279	-0.000640869	-0.000183105	-0.000411987	-0.00741577	0.197571	0.110229	-0.0273743
0.000732422	-0.00114441	0.00050354	-0.000686646	-0.00750732	0.197113	0.109772	-0.0284271
0.00210571	-0.00302124	-0.00050354	-0.000869751	-0.00695801	0.196976	0.111053	-0.0269165
0.00100708	-0.00141907	-0.00279236	-4.57764e-005	-0.00737	0.19693	0.110229	-0.0286102
0.00105286	-0.000869751	0.000228882	-0.000869751	-0.00714111	0.196518	0.110596	-0.0270081
0.00146484	-0.000823975	-0.000549316	-0.00123596	-0.00691223	0.196426	0.110504	-0.0286102
0.00178528	-0.00050354	-0.00260925	-0.000549316	-0.00741577	0.196518	0.110687	-0.026825
0.00114441	-0.00201416	-0.000320435	-0.000640869	-0.00654602	0.196426	0.110229	-0.0283356
0.00151062	7.15527e-005	-0.00279236	-0.000869751	-0.0067749	0.196014	0.110779	-0.0270996
0.00302124	-4.57764e-005	0.000823975	-0.00050354	-0.00764465	0.196198	0.110229	-0.0284271
0.00215149	-0.00151062	-0.000823975	-0.00100708	-0.00718689	0.195923	0.110687	-0.026825
0.00128174	-0.00100708	-0.00151062	-0.00123596	-0.00714111	0.196747	0.110504	-0.0283356
0.000549316	-0.00178528	0.000640869	-0.000778198	-0.00750732	0.19606	0.110321	-0.026825
0.00260925	-0.000366211	-0.000549316	-0.000778198	-0.00718689	0.195969	0.10968	-0.0289307
0.00279236	-0.000457764	0.000320435	-0.000686646	-0.00695801	0.196198	0.111053	-0.0271912
0.00187683	-0.00132751	-0.00196838	-0.000915527	-0.0072913	0.196381	0.110413	-0.0287018
0.00302124	-0.00247192	-0.00201416	-0.000686646	-0.00759888	0.196884	0.110229	-0.0270996
0.00192261	-0.000915527	-0.000595093	-0.000640869	-0.00723267	0.196518	0.110138	-0.0285187
0.00183105	-0.00151062	0.000549316	-0.000778198	-0.00723267	0.196655	0.110687	-0.0271912
0.00247192	-0.00109863	-0.00270081	-0.000457764	-0.00723267	0.197021	0.110413	-0.0284271
0.00128174	-0.000915527	-0.0015564	-0.00151062	-0.0067749	0.19725	0.111237	-0.0269165
0.00416565	-0.00151062	-0.000228882	-0.00105286	-0.00695801	0.197021	0.109863	-0.0290222
0.00215149	7.15527e-005	-0.000869751	-0.000183105	-0.00769043	0.197388	0.110596	-0.026825
0.000640869	-0.000915527	0.000686646	-0.000686646	-0.00732422	0.197845	0.110321	-0.0290222
0.0017395	-0.00228882	-0.000274658	-0.00141907	-0.0067749	0.197617	0.110779	-0.0272827
0.0019019	-0.00178528	-0.00215149	-0.00114441	-0.00714111	0.1978	0.110687	-0.0284271
0.00311279	-0.00128174	-0.00114441	-0.000320435	-0.00700378	0.198166	0.110504	-0.0274658
0.00238037	-0.00292969	0.00114441	-0.000595093	-0.00686646	0.197937	0.110229	-0.0285187
0.00123596	-0.00265503	-0.0015564	-0.00105286	-0.00691223	0.198074	0.111053	-0.0269165
0.000457764	-0.00196838	-0.000869751	-0.000778198	-0.00700378	0.198212	0.110138	-0.0287018
0.00357056	-0.00146484	-0.000823975	-0.000915527	-0.00709534	0.198303	0.110779	-0.0270996
0.00228882	-0.00192261	-0.000183105	-0.00119019	-0.00691223	0.198166	0.109772	-0.0287933
0.00114441	-0.00114441	0.000366211	-0.000457764	-0.00695801	0.198898	0.110962	-0.0273743
0.0015564	-0.0015564	-0.00128174	-0.000823975	-0.00672913	0.198303	0.110229	-0.0285187

Figure 6. Raw Data obtained from nIMU sensor.

Based on the IDC (IMU Data Console) manual of the nIMU sensor device, there are a total of 13 columns of raw data. The first column is the counter (time). The second to fourth columns are the gyroscope xyz coordinates, fifth to seventh columns are the acceleration xyz coordinates, and the eighth to tenth columns are the magnetometer xyz coordinates. The last three columns are the temperature of the xyz gyroscopes. This experiment does not use the gyroscope temperature; however, it will be available for future researchers. These raw data can be saved as a text file through the main menu option (o) configuration. The purpose of saving the raw data into text file was to use the algorithm with quaternion arithmetic and learn how to interact with it in offline processing.

B. PROCESS RAW DATA THROUGH MATLAB

After testing the nIMU sensor by moving the device by the movements of yaw, roll and pitch, all the raw data from the sensor was stored into a text file. The core of the experiment is being able to work with the raw data from the sensor and process the data through Matlab using an algorithm. In order for the user to use the raw data obtained

from the sensor of the nIMU, the user needs to implement an algorithm which is called the Factored Quaternion Algorithm that will interact with the data from the sensor. The algorithm, which will work with the raw data, will be based on a concept called quaternion.

1. Quaternion

Before going any further into the factored quaternion algorithm, an understanding of the quaternion is important. The quaternion is “a non-commutative extension of complex numbers” which was first invented by an Irish mathematician Sir William Rowan Hamilton [5]. In quaternion algebra we define three elements i, j, k with the properties $i^2 = j^2 = k^2 = ijk = -1$ [7]. Based on this, as an extension of complex members, a quaternion is defined as $a + bi + cj + dk$, with a, b, c , and d real numbers. Within the quaternion arithmetic, we define an addition, multiplication, product, norm, complex conjugate, inverse and the rotation matrix, which we use to express the location and motion of the sensor. Below are the general formulas to work with quaternion arithmetic:

a. Quaternion Addition

$$\left. \begin{array}{l} p = p_0 + ip_1 + jp_2 + kp_3 \\ q = q_0 + iq_1 + jq_2 + kq_3 \end{array} \right\} p + q = (p_0 + q_0) + i(p_1 + q_1) + j(p_2 + q_2) + k(p_3 + q_3)$$

b. Quaternion Multiplication with Scalar

$$\left. \begin{array}{l} q = q_0 + iq_1 + jq_2 + kq_3 \\ c = \text{scalar} \end{array} \right\} cq = (cq_0) + i(cq_1) + j(cq_2) + k(cq_3)$$

c. Vector Cross Product

$$p \times q = \begin{vmatrix} i & j & k \\ p_1 & p_2 & p_3 \\ q_1 & q_2 & q_3 \end{vmatrix} = i(p_2q_3 - p_3q_2) + j(p_3q_1 - p_1q_3) + k(p_1q_2 - p_2q_1)$$

d. Vector Dot Product

$$p \bullet q = p_1 q_1 + p_2 q_2 + p_3 q_3$$

e. Quaternion Multiplication with Two Quaternions

When multiplying two quaternions together, the dot product and cross product are used to obtain the product of two quaternions.

$$\text{Note : } p = ip_1 + jp_2 + kp_3 \quad \text{and} \quad q = iq_1 + jq_2 + kq_3$$

$$\left. \begin{array}{l} p = p_0 + \vec{p} \\ q = q_0 + \vec{q} \end{array} \right\} pq = p_0 q_0 - \vec{p} \bullet \vec{q} + p_0 \vec{q} + q_0 \vec{p} + \vec{p} \times \vec{q}$$

f. Quaternion Norm

$$|q| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$$

g. Quaternion Complex Conjugate

$$q = q_0 + iq_1 + jq_2 + kq_3$$

$$q^* = q_0 - iq_1 - jq_2 - kq_3$$

h. Quaternion Inverse

$$q^{-1} = \frac{q^*}{|q|^2}$$

i. Rotation Matrix From Quaternion

$$R = \begin{bmatrix} 2q_0^2(-1) + 2q_1^2 & 2q_1q_2 - 2q_0q_3 & 2q_1q_3 + 2q_0q_2 \\ 2q_1q_2 + 2q_0q_3 & 2q_0^2(-1) + 2q_2^2 & 2q_2q_3 - 2q_0q_1 \\ 2q_1q_3 - 2q_0q_2 & 2q_2q_3 + 2q_0q_1 & 2q_0^2(-1) + 2q_3^2 \end{bmatrix}$$

2. Factored Quaternion Algorithm

After understanding the background in quaternion arithmetic, we introduce the factored quaternion algorithm. Matlab software was used to implement the quaternion formulas in offline processing before moving on to C++ in real-time processing. In order to implement an algorithm which uses quaternion formulas, there are three positions that use the quaternion formulas; elevation quaternion, roll quaternion, and azimuth quaternion along with the singular avoidance. These positions apply the same concept as flight dynamics [6] using the orientation and control in three dimensions. Instead of using the flight itself, the usage of the nIMU sensor will be in replacement for the flight. There are three critical nIMU sensor dynamics parameters, which are the angles of the rotation in three dimensions; pitch, roll, and yaw. To help understand the whole picture of the usage of pitch, roll, and yaw in nIMU sensor, Figure 7 shows the coordinates of a nIMU sensor. The z-axis is the yaw, the y-axis is the pitch and the x-axis is the roll.

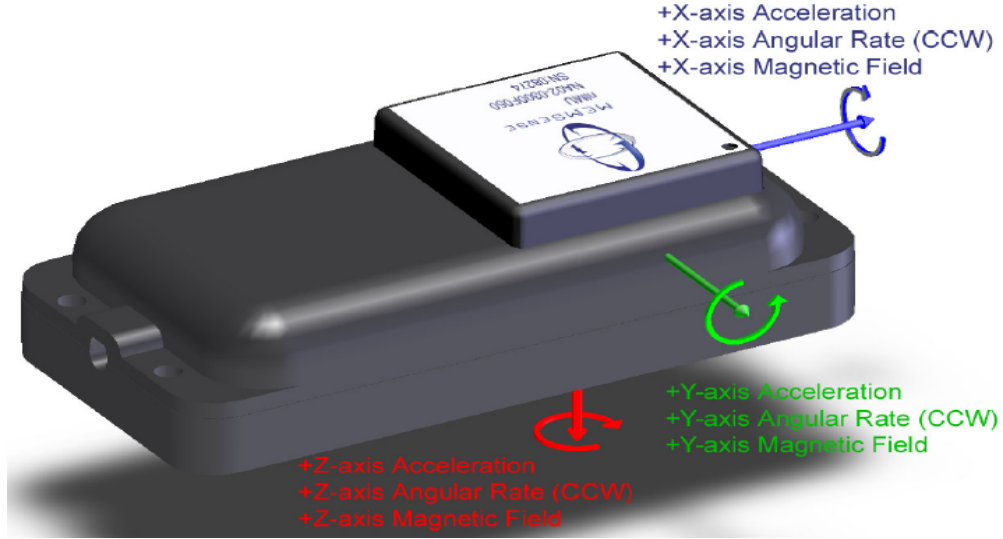


Figure 7. Pitch, Roll, Yaw with nIMU sensor From [4].

a. Elevation Quaternion

The elevation quaternion is a pitch rotation of a rigid body about the y-axis by an angle θ , which is aligned with the Earth coordinate system. Since the x-axis

accelerometer is perpendicular to gravity, it has zero acceleration at rest. Whereas the y-axis accelerometer also has zero acceleration, the z-axis accelerometer shows $-g$ (gravity). Including the elevation, the computation will be the following:

x-axis accelerometer:

$$a_x = g \sin \theta$$

z-axis accelerometer:

$$a_z = -g \sin \theta$$

$$g = -9.81 \text{ m/sec}^2$$

where g is the acceleration due to gravity.

The measured acceleration vector in the body coordinate system is

$$a = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

denoted as

The acceleration vector can be obtained from the raw data in column fourth to sixth of Figure 6. After obtaining the acceleration vector, it needs to be normalized to a unit vector. Therefore, the normalized vector of the acceleration measurements is:

$$\bar{a} = \frac{a}{|a|} = \begin{bmatrix} \bar{a}_x \\ \bar{a}_y \\ \bar{a}_z \end{bmatrix} \text{ whereas } |a| \text{ is the norm of the acceleration vector } a. \text{ Due to}$$

the convention restricted from the elevation angle θ , the range must be in the interval $-\pi/2 \leq \theta \leq \pi/2$ and $\cos \theta$ must be positive. To obtain the value of $\cos \theta$ and $\sin \theta$, the formulas below show the computation which are needed for the next computation.

$$\sin \theta = \bar{a}_x$$

$$\cos \theta = \sqrt{1 - \sin^2 \theta}$$

In order to obtain the elevation quaternion, there are two other values that are needed before processing the elevation quaternion. The two values, which are needed, are the half-angle formulas as shown below:

$$\sin \frac{\theta}{2} = \text{sign}(\sin \theta) \sqrt{(1 - \cos \theta) / 2}$$

where $\text{sign}()$ return the +1 for positive and -1 for negative

$$\cos \frac{\theta}{2} = \sqrt{(1 + \cos \theta) / 2}$$

After all the computation above, the final elevation quaternion formula is:

$$q_e = \cos \frac{\theta}{2} (1 \ 0 \ 0 \ 0) + \sin \frac{\theta}{2} (0 \ 0 \ 1 \ 0)$$

b. Roll Quaternion

The roll quaternion is the roll rotation of a rigid body about the x-axis by an angle ϕ . The acceleration that is measured by the z-axis accelerometer with roll angle ϕ is assigned to zero given by the equations of $a_z = -g \sin \theta$ [6]. The formula for the y-axis accelerometer is given due to the fact that the azimuth does not change the measurement. Therefore the equation for the y-axis accelerometer is $a_y = -g \cos \theta \sin \phi$ and the z-axis accelerometer is $a_z = -g \cos \theta \cos \phi$. After obtaining the results of the y and z-axis accelerometer, it needs to be normalized. Therefore the normalized acceleration measurements are:

$$\bar{a}_y = -\cos \theta \sin \phi$$

$$\bar{a}_z = -\cos \theta \cos \phi$$

The value of $\cos \theta$ comes from

$$\begin{aligned}\sin \theta &= \bar{a}_x \\ \cos \theta &= \sqrt{1 - \sin^2 \theta}.\end{aligned}$$

However, if the value of $\cos \theta$ is not zero, the value of $\sin \phi$ and $\cos \phi$ are computed from

$$\begin{aligned}\sin \phi &= -\bar{a}_y / \cos \theta \\ \cos \phi &= -\bar{a}_z / \cos \theta.\end{aligned}$$

If the value of $\cos \theta$ is zero then the x-axis of the body coordinates is vertically oriented [6]. Due to the convention restricted from the roll angle ϕ , the range must be in the interval $-\pi \leq \phi \leq \pi$. In obtaining the half angle values for ϕ , this can be computed in the same manner as θ :

$$\sin \frac{\phi}{2} = \text{sign}(\sin \phi) \sqrt{(1 - \cos \phi) / 2}$$

where $\text{sign}()$ return the +1 for positive and -1 for negative

$$\cos \frac{\phi}{2} = \sqrt{(1 + \cos \phi) / 2}$$

After obtaining all the values for ϕ , the roll quaternion equation is computed by:

$$q_r = \cos \frac{\phi}{2} (1 \ 0 \ 0 \ 0) + \sin \frac{\phi}{2} (0 \ 1 \ 0 \ 0)$$

c. Azimuth Quaternion

With azimuth rotation, it is the yaw rotation of the rigid body about the z-axis by an angle Ψ . Therefore, azimuth rotation has no effect on the roll and elevation quaternions; so in order to estimate the azimuth quaternion, the roll and elevation quaternions need to be computed first. They are then used to rotate the normalized magnetic field measurement vector. Before normalizing the magnetic field measurement, it is necessary to obtain the magnetic field measurement from the raw data in column

seventh to ninth. The normalized magnetic field is computed in the same way as the

normalized acceleration field: ${}^b m = \frac{m}{|m|} = \begin{bmatrix} {}^b m_x \\ {}^b m_y \\ {}^b m_z \end{bmatrix}$

With this measurement, the normalized magnetic field in body coordinates is computed in the Earth coordinate system as follows

$${}^e m = q_e \ q_r \ {}^b m \ q_r^{-1} q_e^{-1}$$

The final azimuth quaternion is then computed as

$$q_a = \cos \frac{\psi}{2} (1 \ 0 \ 0 \ 0) + \sin \frac{\psi}{2} (0 \ 0 \ 0 \ 1)$$

The magnetic field measurement vector in earth coordinate should agree with the known local normalized magnetic field vector with ψ the azimuth angle:

$$n_z = {}^e m_z$$

$$\begin{bmatrix} n_x \\ n_y \end{bmatrix} = \begin{bmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{bmatrix} \begin{bmatrix} {}^e m_x \\ {}^e m_y \end{bmatrix}$$

The normalized local magnetic

field N and the normalized magnetometer field measurement M are defined as

$$N = \begin{bmatrix} N_x \\ N_y \end{bmatrix} = \frac{1}{\sqrt{n_x^2 + n_y^2}} \begin{bmatrix} n_x \\ n_y \end{bmatrix}$$

$$M = \begin{bmatrix} M_x \\ M_y \end{bmatrix} = \frac{1}{\sqrt{{}^e m_x^2 + {}^e m_y^2}} \begin{bmatrix} {}^e m_x \\ {}^e m_y \end{bmatrix}.$$

From the definition of the magnetic field measurement vector in earth coordinate and the local normalized magnetic field vector with ψ the azimuth angle, we can relate N and M as

$$\begin{bmatrix} N_x \\ N_y \end{bmatrix} = \begin{bmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{bmatrix} \begin{bmatrix} M_x \\ M_y \end{bmatrix}.$$

From this we obtain $\cos \psi$ and $\sin \psi$ as

$$\begin{bmatrix} \cos \psi \\ \sin \psi \end{bmatrix} = \begin{bmatrix} M_x & M_y \\ -M_y & M_x \end{bmatrix} \begin{bmatrix} N_x \\ N_y \end{bmatrix}.$$

In order to obtain $\cos \frac{\psi}{2}$ and $\sin \frac{\psi}{2}$ before calculating the azimuth quaternion, we use the same equations as previously:

$$\sin \frac{\psi}{2} = \text{sign}(\sin \psi) \sqrt{(1 - \cos \psi) / 2}$$

where $\text{sign}()$ return the +1 for positive and -1 for negative

$$\cos \frac{\psi}{2} = \sqrt{(1 + \cos \psi) / 2}$$

From these equations, we compute the orientation of the rigid body as $\hat{q} = q_a \ q_e \ q_r$.

d. Singularity Avoidance

With the factored quaternion algorithm, we use three angles to derive the quaternion estimate of the orientation of the rigid body. However, with any three-parameter representation of a 3-D orientation, it is known that it is impossible to avoid singularity. Therefore the factored quaternion algorithm has a singularity and we need to implement a technique to avoid it. In order to do this we need to understand where the singularity occurs in the factored quaternion algorithm. The singularity occurs when the elevation angle is $\pm 90^\circ$ and this happens when the $\cos \theta = 0$ or $\bar{a}_z = 0$ in

$$\begin{aligned} \sin \phi &= -\bar{a}_y / \cos \theta \\ \cos \phi &= -\bar{a}_z / \cos \theta \end{aligned}$$

To avoid the singularity, we needed to check the value of $\bar{a}_z$. If the value of $\bar{a}_z \leq \varepsilon$, with ε a predefined constant such as $\varepsilon=0.1$, then the calculation of the

normalized acceleration measurement and the normalized magnetic field measurement vector are needed. The following is the calculation to be done:

$$\begin{aligned}\bar{a}_{offset} &= q_{\alpha} \bar{a} q_{\alpha}^{-1} \\ {}^b m_{offset} &= q_{\alpha} {}^b m q_{\alpha}^{-1}\end{aligned}$$

where q_{α} is the offset (rotation) quaternion defined as

$$q_{\alpha} = \cos \frac{\alpha}{2} (1 \ 0 \ 0 \ 0) + \sin \frac{\alpha}{2} (0 \ 0 \ 1 \ 0) .$$

THIS PAGE INTENTIONALLY LEFT BLANK

III. TRACKING ALGORITHM DEVELOPMENT

All of the important content from quaternion arithmetic and factored quaternion algorithm comes together into the tracking algorithm development. This is the final stage of obtaining the outcome of what is expected of the input device in offline processing through Matlab. However, it is not the final stage of the whole research plan because the main purpose of this research is able to obtain the outcome in real-time processing and this will be accomplished using C++. Within the tracking algorithm development, there are six steps which need to be accomplished in obtaining the outcome. After these steps, which will be explained shortly, have been accomplished, the outcome of the input device should be a graph of what the user has written with the sensor. Figure 8 shows where the sensor is supposed to be located on a pen or marker. As you can see there are three separate coordinates, one is for the sensor with x_s, y_s, z_s , second is for the pen or marker tip with x_t, y_t, z_t , and the third is the Earth-fixed coordinates with x_e, y_e, z_e . The purpose of having the Earth-fixed coordinates is to estimate the position of the pen or marker.

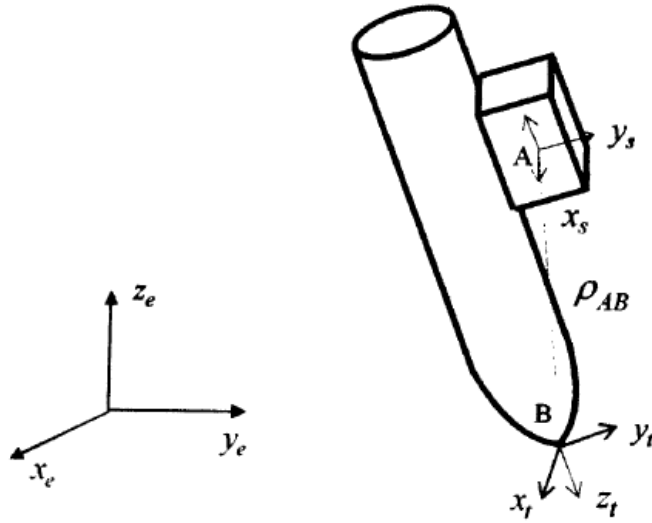


Figure 8. Earth-Fixed Coordinates x_e, y_e, z_e , Sensor Coordinates x_s, y_s, z_s , and Pen or marker Tip Coordinates x_t, y_t, z_t From [10].

A. SENSOR ORIENTATION

After obtaining the accelerometer and magnetometer measurements, we process them through the factored quaternion algorithm (FQA). However, the FQA does not include the gyro measurements which are obtained through the raw data of the nIMU sensor itself. All three measurements are in the sensor coordinate system and the next step is to estimate the sensor orientation. To estimate the sensor orientation, the orientation will be represented by a quaternion q . The sensor orientation will be the same as the writing instrument orientation because the sensor itself will be rigidly attached to the pen or marker as shown in Figure 9.

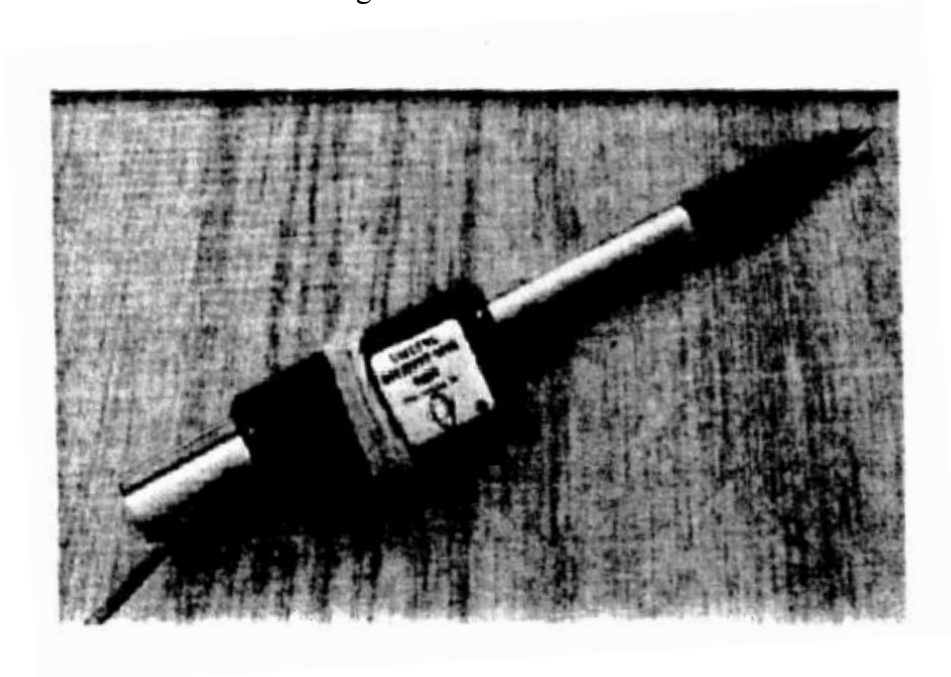


Figure 9. MEMSense nIMU attach to a pen or marker From [10].

B. EARTH COORDINATE SYSTEM

With the accelerometer measurement in the sensor coordinate system which was obtained through the FQA, the measurements are required to be converted into earth coordinate system to determine the location of the pen or marker and the sensor attached. This can be accomplished by the quaternion rotation operator, ${}^e a = q {}^s a q^*$ where ${}^e a$ is

the accelerometer measurement in earth coordinate system, ${}^s a$ is the accelerometer measurement in sensor coordinate system, q and q^* are the quaternion and quaternion conjugate which are obtained from the FQA.

C. VELOCITY OF SENSOR

In order to obtain the velocity of the sensor shown as point A in Figure 8, it is needed to integrate the accelerometer measurement in earth coordinate. This yields the velocity of the sensor at point A as the three-dimensional vector in earth coordinate system. The velocity of point A in earth coordinate is given by: ${}^e v_A = \int ({}^e a) dt$ whereas ${}^e a$ is obtained by the quaternion rotation operator.

D. VELOCITY OF PEN BASED ON SENSOR

After computing the velocity of the sensor in point A, the next step is to compute the velocity of the pen or marker based on the sensor in point A and the angular velocity. In order to accomplish this calculation, there are some components which need to be computed before reaching the final calculation of the equation ${}^e v_B = {}^e v_A + q ({}^s \omega \times {}^s \rho_{AB}) q^*$ which is the velocity of the pen or marker based on the sensor. Note the “s” which stands for “sensor” and “e” which stands for “earth”. The first step is to obtain the ${}^s \omega$ which is the angular rate measurement from the gyroscope in column two through four from the raw data of the sensor. The ρ_{AB} is the distance between the nIMU and the pen or marker tip as shown in Figure 8. In order to obtain the ρ_{AB} , there are two calculations which are needed. The first calculation is the S_ω which is the skew symmetric matrix associated with the ω , the angular rate measurement. The skew symmetric matrix S_ω is given by:

$$S_\omega = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

The skew symmetric matrix will be stored in the S array to go through enumerate of the raw data of the sensor for the gyroscope. The S array will be in a $3n \times 3$ matrix by

$$S = \begin{bmatrix} s_{\omega}(1) \\ \vdots \\ s_{\omega}(n) \end{bmatrix}$$

stacking n skew symmetric matrices given by: $\begin{bmatrix} s_{\omega}(1) \\ \vdots \\ s_{\omega}(n) \end{bmatrix}$. The second calculation is the ${}^s v_A$ which is the velocity of point A in the sensor coordinate, as shown in Figure 8 and it is given by

$${}^s v_A(i) = q^*(i) {}^e v_A(i) q(i)$$

where i = the number of row.

The term ${}^e v_A$ is obtained by integration as discussed earlier. The purpose of doing the calculation for ${}^s v_A$ through ${}^e v_A(i)$ is to convert from earth coordinate back to sensor coordinate. The ${}^s V_A$ will be a $3n \times 1$ vector by stacking the velocities of point A given by:

$${}^s V_A = \begin{bmatrix} {}^s v_A(1) \\ \vdots \\ {}^s v_A(n) \end{bmatrix}$$

After obtaining the calculation of both the S (skew symmetric matrix) and ${}^s V_A$ (velocities of point A), we obtain the distance between the nIMU and the pen or marker tip, ρ_{AB} . The equation is given by: $\rho_{AB} = -(S^T S)^{-1} S^T {}^s V_A$ where T is the transpose of the matrix S. With the final calculation for ρ_{AB} , we compute the vector cross product with ${}^s \omega$ and the result is in sensor coordinate system. To convert the result from sensor coordinate system to earth coordinate system, quaternion rotation operator is applied along with the final calculation of ${}^e v_B = {}^e v_A + q ({}^s \omega \times {}^s \rho_{AB}) q^*$.

E. DETECT PAUSE PHASES

When the user is writing something, the sensor is recording every unistroke of the data. However, the user is not always writing and there is a pause in between two unistrokes. The next step is to be able to detect this pause. If there is a pause phase, it is necessary to apply zero velocity correction to make sure that there is no data running during the pause phase.

F. POSITION OF THE PEN TIP

The final step of the tracking algorithm development is to compute the position of the pen tip. In order to compute the position of the pen tip, it is necessary to integrate the velocity of the pen tip,

$${}^e v_B = {}^e v_A + q ({}^s \omega \times {}^s \rho_{AB}) q^*$$

The equation for computing the position of the pen tip is given,

$${}^e p_B = \int ({}^e v_B) dt$$

However, the pen tip position, ${}^e p_B$, is computed in three dimensions. Because the user normally writes on surface such as a piece of paper on a table, or on a blackboard, the pen tip trajectory is actually in two dimensional plane but not always. For instance, the user might write in the air and there is no plane to be able to hold any hand motions in. With the calculation of the position of the pen tip, which is the result in three dimensional, it will give the final result of the handwriting tracking. However, the results will be projected onto a two-dimensional space in order to be displayed on a computer screen.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. RESULT OF RESEARCH

After coding all the algorithms for quaternion, factored quaternion algorithm, and tracking algorithm development in Matlab for offline processing, we tested the sensor's raw data to see the result of what is expected. In order to accomplish the result of what is expected, we ran the factored quaternion algorithm first. We set the sensor pointing to North direction and sitting on a flat surface as it is shown in Figure 10.



Figure 10. MEMSense nIMU sensor and Compass set in North direction.

The purpose of setting the sensor in North direction is due to the fact that we want the accelerometer xyz coordinates to be ideally 0 0 1 and magnetometer xyz coordinates to be ideally zero in y component and nonzero in x and z components. This is true if the sensor is motionless and in reference to the gravity of the earth pointing downward. While the sensor is in North direction, we obtain the raw data as shown in Figure 11.

Figure 12 is the same raw data obtain from the nIMU sensor but in a readable format. The bottom of Figure 12 is a row of data showing accelerometer and magnetometer xyz measurements.

12991	0.494185	-0.109863	-1.1261	-0.0025197	-0.03477	0.981734	0.102196
2.61082	0.307109	54.682	55.0634	55.0453			
4529	0.078906	-0.759843	-1.09963	-0.00189099	-0.032959	0.983724	0.102022
-2.61117	0.305109	54.5912	55.1179	54.9988			
0.1008	-1.55559	0.631714	0.109063	-0.00103105	-0.0375366	0.983724	0.101761
-2.61116	0.309487	54.5548	55.2451	54.7546			
53844	-0.521851	1.1261	-0.05919	-0.00165211	-0.0354767	0.98671	0.102892
-2.61178	0.304085	54.7183	54.9726	54.8273			
44629	-0.0549316	1.0437	0.32959	-0.00086646	-0.0389099	0.984879	0.101239
-2.61134	0.302586	54.6275	55.3984	54.8091			
36220	-0.261304	-1.2085	0.302124	-0.00205994	-0.03477	0.983724	0.101587
2.6116	0.304085	-0.4455	55.2888	54.8455			
27754	0.32959	-0.631714	-0.439453	-0.00434875	-0.0341034	0.985565	0.102544
-2.61178	0.306935	54.8455	55.1361	55.0271			
19287	-0.631714	0.439453	-0.00183105	-0.0405121	0.985565	0.102544	
2.6116	0.307005	54.4458	55.2814	54.9363			
10028	1.09863	-0.137329	-0.576782	-0.00274658	-0.0357056	0.986023	0.102109
-2.60883	0.307283	54.791	55.2451	54.9363			
2369	0.411987	-0.24792	0.931038	-0.00086646	-0.0375366	0.987854	0.101239
-2.6116	0.308588	54.5912	55.1724	54.9181			
57435	-0.767043	0.761304	1.26343	-0.00205994	-0.0375366	0.987854	0.101761
2.61195	0.306761	54.573	55.4268	54.8636			
50972	-0.85144	-1.09863	-0.32959	-0.00320435	-0.032959	0.987167	0.102718
-2.61178	0.307109	54.5548	55.2088	55.0453			
42501	0.631714	-0.795509	-0.109863	-0.00137329	-0.0359344	0.988312	0.102022
-2.61064	0.306761	54.6275	55.1179	55.0634			
34041	0.219727	-0.384521	0.219727	-0.00366211	-0.0393677	0.988998	0.10237
2.6116	0.307457	54.7728	55.1543	55.1724			
25575	-0.078906	-1.76528	0.576782	-0.00389099	-0.0359344	0.98671	0.101761
-2.61117	0.305891	54.7728	55.1724	55.1724			
17115	-1.26343	0.439453	-0.192361	-0.00160217	-0.0359344	0.990372	0.102022
-2.61134	0.307109	54.682	55.1543	55.1361			
0659	0.219727	0.411987	-0.576782	0.00114441	-0.036456	0.986252	0.102196
-2.61195	0.307457	54.7183	55.2814	55.009			
183	-0.731838	-0.521851	-1.0437	-0.00220882	-0.0357056	0.986481	0.102283
-2.60883	0.308086	54.8455	55.1724	55.0453			
57257	0.109863	0.192261	0.932038	-0.00137329	-0.0363922	0.988998	0.102022
-2.61117	0.308086	54.573	55.2996	54.9181			
40708	-0.384521	0	-0.494305	-0.00160217	-0.0366211	0.990143	0.102544
2.6116	0.307283	54.6893	55.1906	55.0016			
40328	-0.706372	-0.741577	-0.796509	-0.000915527	-0.0375366	0.987854	0.101239
-2.61212	0.30563	54.5185	55.1708	55.1361			
31865	-0.521851	0.384521	-1.2085	-0.00160217	-0.0359344	0.98671	0.102718
-2.61021	0.305457	54.4458	55.2541	55.0998			
22295	-1.0437	0.706372	-0.0274658	-0.00434875	-0.0354767	0.988998	0.102892
-2.61134	0.304085	54.7546	55.2633	55.0271			
14930	0.32959	0.214111	0.219727	0.00086646	-0.0341034	0.986252	0.102022
-2.6116	0.306239	54.5912	55.2451	55.0634			
6472	-0.109863	-0.219727	-0.109863	-0.000915527	-0.0366211	0.991287	0.102718
-2.61178	0.30653	54.7183	55.0816	55.1724			
63541	-1.75007	-1.09863	0.411987	-0.000915527	-0.0352478	0.990829	0.10237
-2.61134	0.30653	54.7546	55.2996	54.9363			
55074	0.137329	-0.439453	0.98877	0.000228882	-0.0354767	0.987396	0.101239
2.61178	0.306457	54.6893	55.2359	55.1906			
46683	-0.219727	-0.32959	0.0274658	-0.00183105	-0.0354767	0.987456	0.101935
2.60751	0.305457	54.8455	55.1543	55.2267			

Figure 11. Raw Data with Sensor point in North direction.

Counter	Gyroscope			Accelerometer		Magnetometer	Gyro Temperature					
3378	0.027466	0.686646	-0.54932	0.0139618	0.0151062	-1.00548	0.109415	-0.0163513	-0.0317459	56.2624	56.9172	56.4622
12488	-1.78528	0.302124	0.247192	0.0160217	0.0130463	-1.0112	0.110023	-0.0169601	-0.0304413	56.1534	56.117	56.771
21892	-1.56555	-0.24719	-0.13733	0.0160217	0.0144196	-1.0096	0.110806	-0.0177429	-0.0311371	56.226	56.0989	56.6983
33216	-0.63171	0.796509	-0.02747	0.0164795	0.0160217	-1.00525	0.10995	-0.0171341	-0.0307892	56.1534	56.0807	56.771
41004	-0.63171	0.384521	-0.30212	0.0162506	0.0109863	-1.0096	0.109328	-0.0177429	-0.0311371	56.1534	56.0625	56.6438
52032	0.164795	0.631714	-0.10986	0.017395	0.011673	-1.00708	0.110023	-0.0166992	-0.031572	56.2624	56.1897	56.6257
59809	-0.57678	0.32959	-0.27466	0.0178528	0.0130463	-1.00639	0.111154	-0.0171341	-0.0311371	56.1352	56.117	56.444
5311	1.31836	1.29089	-0.32959	0.0183105	0.0119019	-1.00777	0.108545	-0.0163513	-0.0307892	56.0807	55.9899	56.6257
13093	-1.23596	1.23596	0.054932	0.0196838	0.0100708	-1.00822	0.10985	-0.0169601	-0.0320938	56.0807	56.117	56.553
24120	-0.49439	0.576782	1.51062	0.0167084	0.0109863	-1.01051	0.109676	-0.0171341	-0.031311	56.0807	56.0444	56.5712
31910	-0.46692	0.054932	-0.38452	0.0196838	0.00961304	-1.01166	0.109676	-0.0171341	-0.0300934	55.9717	56.2987	56.3168
41024	-0.93384	1.09863	0.494385	0.017395	0.015564	-1.01006	0.110806	-0.0167862	-0.0322678	55.9354	56.2605	56.6257
52347	0.274658	1.1261	-0.19226	0.0164795	0.0121307	-1.00708	0.109502	-0.0169601	-0.031485	56.0262	56.2987	56.444
60136	0.109863	1.15356	0.686646	0.0171661	0.0130463	-1.01074	0.109676	-0.0163513	-0.0320938	56.0444	56.226	56.5167
3700	-0.98877	0.65918	-0.32959	0.0185394	0.0144196	-1.0112	0.109328	-0.0177429	-0.031485	56.0807	56.1715	56.5348
15031	-0.52185	1.07117	0	0.0151062	0.00961304	-1.01074	0.109328	-0.0165253	-0.0306152	56.1352	56.226	56.3895
22818	-0.30212	1.62048	-0.24719	0.0144196	0.0151062	-1.01074	0.109502	-0.0165253	-0.0307892	55.9354	56.1715	56.6438
31921	-0.43945	-0.79651	0.549316	0.0192261	0.013504	-1.0096	0.110023	-0.0176559	-0.031398	56.2442	56.1715	56.5893
43244	1.2085	-0.60425	-0.35706	0.0203705	0.00961304	-1.00937	0.109502	-0.017482	-0.0317459	56.0989	56.117	56.5348
51038	-0.98877	0.439453	-0.32959	0.0171661	0.0100708	-1.00937	0.110458	-0.0167862	-0.0317459	55.899	56.0625	56.5712
62065	-0.79651	1.01624	0.109863	0.0219727	0.0114441	-1.01234	0.108893	-0.0163513	-0.0306152	56.0262	56.2079	56.4077
4314	-0.68665	0.906372	-0.32959	0.0171661	0.0125885	-1.00891	0.109502	-0.0171341	-0.0306152	55.9172	56.1534	56.4985
15337	-0.79651	0.686646	0.302124	0.0183105	0.0130463	-1.00891	0.110197	-0.0161774	-0.031485	56.0989	56.3168	56.2987
23126	-0.68665	-0.10986	-0.74158	0.0153351	0.0107574	-1.00891	0.109502	-0.0163513	-0.0322678	56.0444	56.117	56.3713
32230	-0.68665	0.576782	0.604248	0.0206282	0.00801086	-1.00639	0.110632	-0.0165253	-0.0302673	56.008	56.2624	56.4622
0.0206282 0.00801086 -1.00639 0.110632 -0.0165253 -0.0302673												

Figure 12. Readability format of the Raw Data.

When the sensor's accelerometer and magnetometer xyz coordinates are what is expected, we implemented this data into the factored quaternion algorithm. Ideally, the result of the quaternion should be $1+0i+0j+0k$. After running the factored quaternion algorithm, the result of the quaternion is shown in Figure 13.

quaternion =
0.99707658623584 -0.00475008895120 0.01001560500480 0.07560030086997

Figure 13. Matlab Application of Quaternion Result.

With real time processing which is performed through C++ application, the algorithms are the same except for the coding style and how the result will be presented. With offline processing as mentioned before, the result will show after the calculation of the factored quaternion algorithm as shown in Figure 13. With real time processing,

while the sensor is running, at the same time the factored quaternion algorithm through the C++ application will obtain the raw data, calculate the quaternion and display the result on the application. If the sensor is moving in different direction, a different quaternion result will be displaying on the application. Figure 14 shows the result of the C++ application when the sensor is pointed to North direction. The result agrees with that produced by the Matlab program.

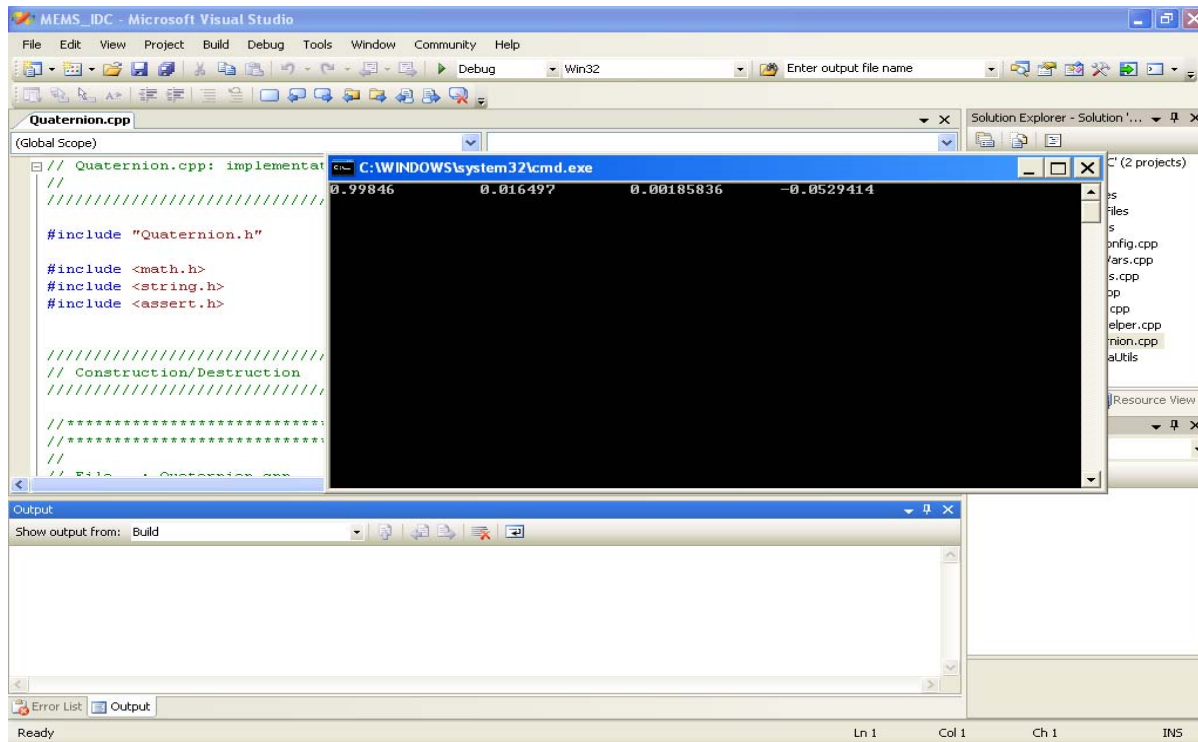


Figure 14. C++ Application of Quaternion result.

V. CONCLUSION & RECOMMENDATIONS

A. CONCLUSION

The objective of this research was to read the unistroke of handwriting by developing a 3-D pen input device. The emphasis was on implementing algorithms for tracking orientation of the 3-D device. First, a Matlab application was developed for offline processing. However, this is not the end result of the research plan for the development of the 3-D device. The main purpose for this research plan is to perform this experiment with real-time processing. In order to read the unistroke in real-time processing, a C++ version of the factored quaternion algorithm was implemented. With the C++ version of the algorithm of the 3-D input device, the user can write on any surface or in the air and have his/her handwriting recorded. As soon as the research is finalized, several types of people will be able to benefit from the development of the 3-D input device. Types of people include, but are not limited to, the defense community, distance-learning students, professors/teachers and hearing-impaired individuals.

B. RECOMMENDATIONS

Although orientation tracking of the 3-D input device was implemented and tested in this research, position tracking has not been completed. For future investigation relating to this research, it is recommended that a position tracking algorithm be developed and implemented. After the completion of the position tracking algorithm, it is possible to collect the data and exhibit the result through a graphic user interface (GUI) to display handwriting.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

- [1] EPOS Advanced Digital Positioning Technologies, Hod-Hasharon, Israel, “Digital Pen & USB Flash Drive,” 2002-2007, http://www.epos-ps.com/dwlfls/USB_spec_site_pdf.pdf. last accessed on 20 June 2008.
- [2] Logitech®, Fremont, CA, “Logitech io2 Pen Technology,” 2008, <http://www.logitech.com/index.cfm/66/459&cl=us,en>. last accessed on 20 June 2008.
- [3] LeapFrog Enterprises, Inc., Emeryville, CA, “FLY Fusion™ Pentop Computer,” 2007-2008, <http://www.flyworld.com>. last accessed on 20 June 2008.
- [4] MEMSense, LLC, Rapid City, SD, “nIMU Nano Inertial Measurement Unit Series Documentation,” July 2008, <http://www.memsense.com> and <http://www.memsense.com/images/downloads/57/IDC-Manual.pdf>. July 2008.
- [5] Wikipedia® Encyclopedia, San Francisco, CA, “Quaternion,” 2001–2008, <http://en.wikipedia.org/wiki/Quaternion>. last accessed on 16 May 2008.
- [6] Wikipedia® Encyclopedia, San Francisco, CA, “Flight Dynamics,” 2001-2008, http://en.wikipedia.org/wiki/Flight_dynamics. last accessed on 16 May 2008.
- [7] MathWorld—A Wolfram Web Resource, “Quaternion,” 1991-2008, <http://mathworld.wolfram.com/Quaternion.html>. last accessed on 16 May 2008.
- [8] X. Yun, E.R. Bachmann, and R.B. McGhee, “A Simplified Quaternion-Based Algorithm for Orientation Estimation From Earth Gravity and Magnetic Field Measurements,” *IEEE Transactions on Instrumentation and Measurement*, Vol. 57, No.3, pp.638-650, March 2008.
- [9] S. T. Karris, *Numerical Analysis Using MATLAB® and Excel*, Orchard Publications, pp. 631, March 2007.
- [10] X. Yun, “Kinematic Equations of a 3-D Input Device Using IMUs,” Research Notes, Naval Postgraduate School, July 2008.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
Ft. Belvoir, VA
2. Dudley Knox Library
Naval Postgraduate School
Monterey, CA
3. Xiaoping Yun
Department of Electrical Engineering
Naval Postgraduate School
Monterey, CA
4. Roberto Cristi
Department of Electrical Engineering
Naval Postgraduate School
Monterey, CA
5. James Calusdian
Department of Electrical Engineering
Naval Postgraduate School
Monterey, CA