Developing a prototype using IMUs to do 24 hours measurement of knee joint mobility in children with Cerebral Palsy

Anza Ahmed

Lund, 2019

Master's Thesis in

Biomedical Engineering

Supervisor: Nebojsa Malesevic

Assistant supervisor: Ingrid Svensson

Department of Biomedical Engineering

LUND UNIVERSITY

LTH FACULTY OF ENGINEERING

“I have not failed. I just found 10,000 ways that won't work.”

- *Thomas A. Edison*

Contents

Abstract

Cerebral Palsy (CP) is a disorder of movement that is caused before, during or soon after birth. CP affects movement and co-ordination caused by lack of muscle control by the brain [1], [2]. Stiff muscles, weak muscles, tremors and seizures as well as difficulties with swallowing are examples of symptoms in children with CP. Every year there are approximately 200 children born in Sweden that are affected by CP [3].

Children affected by CP must visit physicians or physiotherapists for an evaluation check up on their joints. Physicians use a goniometer to study the knee flexion/extension angle thus providing a plan for rehabilitation treatments. A challenge for the physiotherapists is that they are unable to measure the knee joint movement of children with CP for a longer period. There is a lack of research in the area of CP combined with long term measurement devices. This thesis project had a main purpose to create a prototype that was able to measure the knee joint angle in children with CP in their normal environment, for 24 hours.

Two BNO055 inertial measurement units (IMUs) were used to obtain the knee joint angle by using quaternions, which are a number system in four dimensions. The final prototype was tested on a robot and human subject. The results concluded that BNO055 is a good IMU when the measurements are performed for a short time with slow or linear movements. For longer periods, however, the sensors lose their calibration and provide drifted results, hence it is not suited for doing measurements for 24 hours. Another upgraded and high cost sensor BNO080 was then replaced that provided more accurate results for knee joint angles over a longer period of motion. BNO080 also provided short drifting results over time, but not as bad as BNO055. Due to its many operational modes, it might be possible to overcome drifting errors.

It was possible to develop a device for knee joint measurements with promising results for shorter periods, however, to continuously measure data for 24 hours on nonlinear human knee motions, a strong IMU with good autocalibration techniques is needed, and this is something that needs to be tested and studied further.

Sammanfattning

Cerebral Pares (CP) är en rörelsestörning som orsakas före, under eller strax efter födseln. CP påverkar rörelse och koordination orsakad av brist på muskelkontroll av hjärnan [1], [2]. Stela muskler, svaga muskler, skakningar och kramper samt svårigheter att svälja är exempel på symtom hos barn med CP. Varje år är det cirka 200 barn födda i Sverige som drabbas av CP [3].

Barn som drabbats av CP måste besöka läkare eller fysioterapeuter för att utvärdera sina leder. Läkare använder en goniometer för att studera knävinkeln och därmed ge en rehabiliteringsplan. En utmaning för sjukgymnasterna är att de inte kan mäta knäledsrörelsen hos barn med CP under en längre period. Det saknas forskning inom området CP i kombination med långsiktiga mätanordningar. Detta avhandlingsprojekt hade ett huvudsyfte att skapa en prototyp som kunde mäta knäledsvinkeln hos barn med CP i sin normala miljö under 24 timmar.

Två BNO055 inertial measurement units (IMU) användes för att erhålla knäledsvinkeln med hjälp av kvaternioner, som är ett talsystem i fyra dimensioner. Den slutliga prototypen testades på en robot och ett mänskligt ämne. Resultaten drog slutsatsen att BNO055 är en bra IMU när mätningarna utförs under en kort tid med långsamma eller linjära rörelser. Under längre perioder tappar sensorerna sin kalibrering och mätresultaten driftar, varför det inte är lämpligt att göra mätningar under 24 timmar. Försök gjordes också med en uppgraderad och dyrare sensor, BNO080, och den gav mer exakta resultat för knäledsvinklar under en längre rörelseperiod. BNO080 gav också korta driftresultat över tid, men inte så illa som BNO055. På grund av dess många driftslägen kan det vara möjligt att övervinna driftfel.

Det var möjligt att utveckla en prototyp för mätningar av knäleden med lovande resultat under kortare perioder, men att kontinuerligt mäta data under 24 timmar av olinjära mänskliga knärörelser kräver en sofistikerad IMU med bra rutiner for autokalibrering. Detta behöver undersökas och testas mer framöver.

Acknowledgements

I would like to thank and express my gratitude to my assistant supervisor Ingrid Svensson, who gave me the opportunity to work on this amazing project. This thesis would not be possible without your positivity and trust in me. Secondly, I would like to thank my supervisor Nebojsa who helped me achieve my goals from beginning to the end. I will always be grateful for your wisdom and support and I couldn’t ask for a better supervisor than you. I would also like to express my gratitude to Christian Antfolk, who helped me gain all the necessary electrical equipment needed for this project. Lastly, I would also like to thank Anders Robertsson from the robotics labs for allowing me to run some tests on the robot, and Pamela Svensson for providing support on 3D printing plastic cases for the IMUs.

Acronyms

CP	Cerebral Palsy
GND	Ground
MCU	Microcontroller
FSR	Force Sensitive Resistors
I2C	Inter Integrated Circuit
IMU	Inertial Measurement Unit
DC	Direct Current
AC	Alternating Current
GMFCS	Gross Motor Function Classification System
LED	Light-emitting diode
IDE	Integrated development environment
RMSE	Root mean square error

Preface

This master thesis was completed at the department of Biomedical Engineering at Lund University (LTH) under the supervision of Nebojsa Malesevic and Ingrid Svensson. It began on March 2019 and finished on October 2019 achieving 30 ECTS credits.

1. Introduction

Cerebral Palsey (CP) is a collective name of disability caused by brain injury that occurs before the age of two. Children with CP may have cognitive impairment, that can range from mild to severe, but can often be treated by a team of medical professionals. In Sweden, there are over 200 children that get CP every year [1]. The disabilities vary from those who have almost normal function to those who have a pronounced disability. Those who are affected by CP often have too high tension (spasticity) in some muscles while other muscles may be weakened. Since there is an imbalance between muscles that stretch and bend around a joint in the arms and legs, the muscles shortens and cause joint stiffness (contracture). In some people there is a risk for hip dislocation due to asymmetrical activity of the muscles surrounding the hip [4]. There are several methods to reduce muscle tension and prevent contracture, but it is important that the measures are taken in early stages [5].

The physicians and physiotherapists need to evaluate the joint movement of children with CP. This is done by estimating the flexion/extension angles through a goniometer. A problem is that they are unable to measure joint movements for a longer period, this would, however, give more accurate and realistic description of a child's movement and mobility.

1.1 Objective and problems

The main objective of this thesis is to create a prototype that can continuously measure the knee joint motion by estimating the angle for a longer period, in this case 24 hours. The idea is to mount one sensor on a shank, and the other one on a thigh, hence measure the absolute angle between them. This thesis will answer following problems:

- What efficient way can be used to create a smart device?
- Is the prototype comfortable for the children with minimal impact on their daily life?
- How accurate are the obtained results, are there any errors?
- What improvements can be done regarding the quality of measurements and comfort?

1.2 Disposition

This thesis consists of four main sections starting with a background section which will provide theory for children with CP (Cerebral Palsy). This section will make the readers understand what this project is about, hence why a measuring prototype is needed. The second section is about implementing the sensors using different software programs as well as some background on the electrical circuit layout.

The third section is about testing the prototype and analyzing the results to see what further improvements can be made to the sensors.

Lastly, this report will present results and conclusion with thoroughly discussions on improvements and future developments.

2. Background

This chapter gives an overall view on the subject Cerebral Palsy (CP), how a child’s quality of life can improve with either treatments or therapy, and an overview on the knee joint and how to measure its absolute angle. It will also cover a background on software and sensors that were applied for this project.

2.1 Cerebral palsy

Cerebral palsy is a disorder which leads to disability in young children. CP is a result of a brain injury sustained during fetal development or birth. Children with CP has lack of coordination and independent movement of muscles [6]. CP is not always diagnosed at once, parents usually notice that something is wrong when a child fails to reach developments such as rolling over, crawling or walking. CP is caused by damage to the motor cortex of the brain, which is responsible for muscle control and coordination. Out of approximately 100 000 children born in Sweden each year, over 200 are affected by CP [3].

2.1.1 Treatment of Cerebral palsy

There is no cure for CP, but there are treatments that can alleviate the symptoms and improve a child’s life quality. The symptoms change as the child grows older, based on these, the subsequent treatment is planned [3].

Children with CP require long-term care with a medical care team. There are different types of treatments depending on the condition of children. Medication is one way to treat pain and reduce muscles tightness, and to prevent epileptic seizures. The doctor might recommend injections of onabotulinumtoxinA (Botox, Dysport) to relax a specific muscle [2]. Injections have side effects such as pain at the injection site, mild flu-like symptoms or difficulty breathing and swallowing. Drugs such as diazepam (Valium), dantrolene (Dantrium), baclofen (Gablofen) and tizanidine (Zanaflex) are often used to relax muscles, however, they are not recommended for long-term use. Diazepam can cause side effects including drowsiness, change in blood pressure and risk of liver damage that requires monitoring [2].

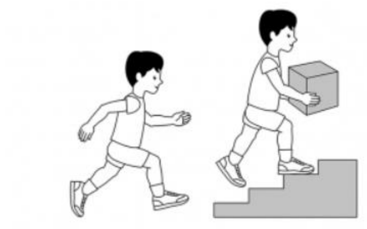
Therapies play a vital role in treating cerebral palsy. Physical therapy helps the children gain strength, flexibility, mobility and motor development. This is done by doing proper exercise and muscle training provided by therapists depending on a child's condition. Occupational therapy is another type that aims to improve children's ability to do everyday tasks [8].

2.1.2 CPUP

CPUP is a follow-up program for people with Cerebral Palsy. It started in 1994 as a collaborative project between pediatric orthopedics and habilitation in Skåne. The people behind CPUP met many children with CP who developed hip dislocation and joint stiffness. They wanted to create a system where all children with CP could be followed in a structured way. Their main goal is to increase the knowledge of CP and improve cooperation between different occupational categories around people with CP [9]. CPUP cooperates with several countries including England, Holland, Germany, Poland and more. More than 95% of parents in Sweden agreed to allow their children with CP to participate in this follow-up program. The program is also available in Norway, Denmark, Scotland, Iceland and Australia.

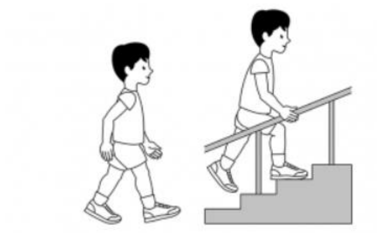
2.1.3 GMFCS

Gross Motor Function Classification System (GMFCS) is a system which provides families and clinicians a thorough description of a child's motor function and gives an idea of what equipment can aid a child in the future, e.g. wheelchairs and crutches. GMFCS is used for infants under 2 years of age and throughout their 18th birthday. A child over the age of 5 will not improve their GMFCS level, for example if a child is classified at a level 5 at the age of 6, then it is likely that they will need to use a mobility device throughout their life [10]. GMFCS can be categorized into 5 different levels:



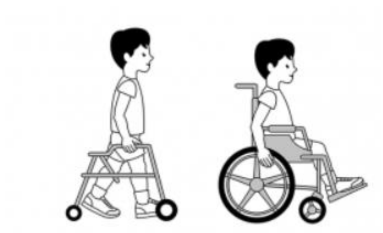
GMFCS Level 1:

Children walk and climb the stairs without the use of railing. They perform gross motor skills such as running and jumping, but speed, balance and coordination are limited.



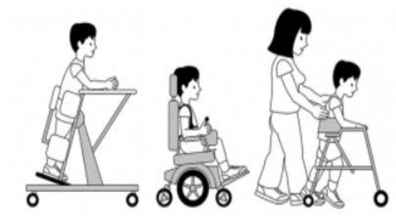
GMFCS Level 2:

Children walk and climb the stairs without the use of railing. They may experience difficulty walking long distances and balancing on uneven terrain or crowded areas. Children have minimal ability to perform skills such as running and jumping.



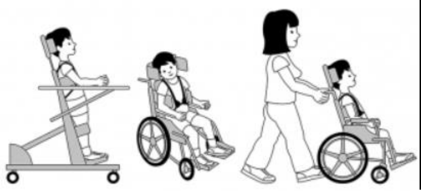
GMFCS Level 3:

Children climb stairs holding onto a railing with supervision assistance. In most indoor settings, they walk using a hand-held mobility device. Long distances require wheeled mobility.



GMFCS Level 4:

Children require physical assistance or powered mobility in most settings. At school or other outdoor environment, children are transported in a manual wheelchair.



GMFCS Level 5:

Children are transported in a manual wheelchair in all settings. Children have limited ability to maintain head and trunk postures as well as leg and arm movements.

Figure 1: Different GMFCS levels [10].

2.2 Knee joint

The knee joint is a hinge joint that consists of three bones: the femur, the tibia and the patella. Hinge joints can move along one axis to flex or extend and are formed between two or more bones [12]. It is said to be one of the strongest and most complicated joints in the human body [11]. It supports the body's weight and allows the lower leg to move relative to the thigh. Knee joint movements are essential to everyday activities, such as walking, running, sitting, providing stability and standing.

2.2.1 Knee anatomy

The knee consists of bones, ligaments, tendons and menisci. The knee joint keeps the femur (thigh bone), tibia (shin bone), and patella (kneecap) in place.

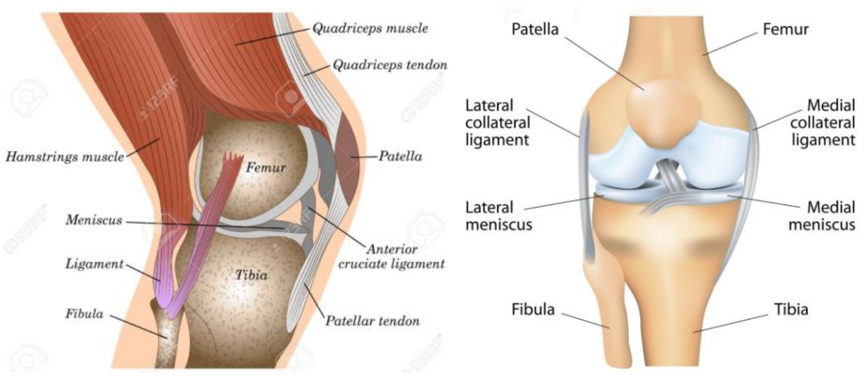


Figure 2: Knee anatomy [13].

Menisci are shaped like discs that act as a cushion or shock absorber so that the bones of the knee can move through without rubbing themselves against each other. The knee has two menisci: medial, on the inner side of the knee, and lateral, on the outer side of the knee [14].

Ligaments are tough tissues and connect bones to other bones. The knee has four ligaments:

- ACL (anterior cruciate ligament), which prevents the femur from sliding backward on tibia.
- PCL (posterior cruciate ligament), which prevents the femur from sliding forward on the tibia.

- MCL (medial collateral ligament), which prevents sideways movement of the femur.
- LCL (lateral collateral ligament), which prevents sideways movement of the femur.

Tendons are also tough tissues that provide stability to the joint. They connect bone to muscle unlike ligaments, which connects bones to bones. The largest tendon in the knee is the patellar tendon that runs up the thigh and attaches to the quadriceps [14]. Muscles around the knee are the hamstrings and quadriceps. They strengthen the leg and help flex and extend the knee. There is also a joint capsule (membrane bag) that surrounds the knee joint. It is filled with a liquid called synovial fluid which lubricates the joint.

2.2.2 Common knee injuries

Knees are often injured during sport activities, exercises or accidents. The most common symptoms are pain and swelling, instability or difficulty with weight bearing. Sprain and strains are injuries that occur to the ligaments. The most often injured ligaments are the ACL and MCL. These injuries are common in sports such as football and basketball where one might experience a sudden twist in motion or an incorrect landing from a jump [14]. Another injury is when a meniscal is teared apart. It also happens during sports where the knee twists or pivots.

A fracture can happen from motor vehicle accidents, falls or sports related accidents. The most common broken bone around the knee is the patella (kneecap).

2.2.3 Treatment of knee injuries

All knee injuries should be evaluated by a doctor. Physical therapy assists a person to recover from knee injuries. Such recovery programs include continuous home exercise programs where the main goal is to restore stability, strength and mobility. In some cases, a surgery is necessary to repair the damage [14].

2.2.4 Previous research on cerebral palsy and knee pain

Children with CP have an abnormal knee joint motion which is very common. Since children with CP have many health challenges, knee symptoms may not be given high priority and in some cases be overlooked [16]. Proper diagnosis and treatments are keys to maximize a child’s quality of life. Two common problems that CP patients have in the knee are increased knee flexion during stance phase and reduced knee flexion during the swing phase of gait. Some of the factors that lead to increased knee flexion during stance phase are hamstrings spasticity, quadriceps weakness, soleus weakness, and lever-arm dysfunction [38]. Gait-stiff knee interferes with ground clearance hence gait patterns result into esthetically poor gait with increased energy consumption of the knee, this may lead to pain in the knee. Rehabilitations or exercises might heal these gait abnormalities. Studying the knee joint mobility over a longer period might aid clinicians in providing better treatments for children with CP that will increase their knee gait pattern [38].

Previous studies have reported that the abnormal knee joint kinematics and kinetics are caused by spasticity and that the knee flexion deformity caused by hamstring contracture increases the contact force in the patellofemoral [15]. A study was performed by C. Tardieu in 1988, where several children with CP aged between 9 and 15 were studied for 24 hours. Tardieu measured the angle of the knee with minimal torque (fully extended knee) and maximal torque (fully flexed knee). He subtracted the first angle with the second and got the range of motion for passive soleus muscle stretch. His results provided new guidelines for the continuous treatment of children with cerebral palsy. Tardieu used a potentiometer to measure the knee angle on one leg. This thesis project is inspired by Tardieu’s research; however, the aim is to develop a wireless and user-friendly prototype using modern technology, something that lacks research on.

2.2.5 Knee range of motion and movements

As previously explained, the knee joint can be regarded as a hinge joint. The following movements involve the knee range of motion:

- Flexion
- Extension
- Medial rotation
- Lateral rotation

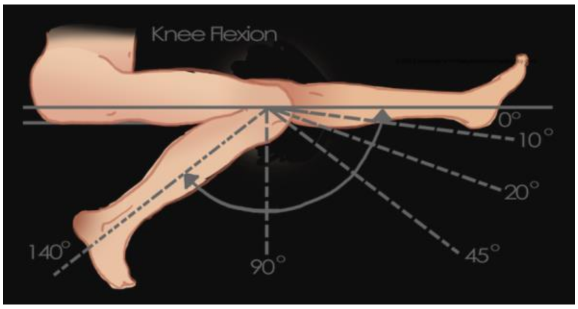


Figure 3: Flexion of the knee [17].

Rotatory movements at the knee have small range. These rotations take place around a vertical axis and occur in the lower compartment of the joint, below the menisci. They can be combined with flexion and extension, which are the chief movements.

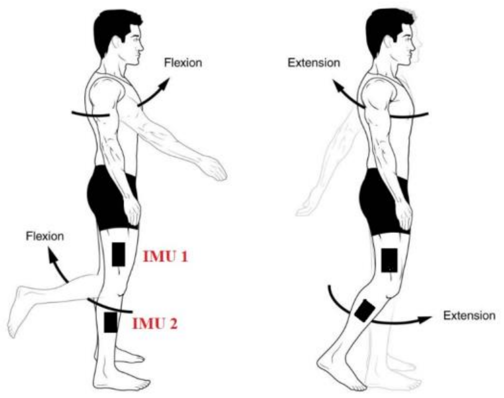


Figure 3: Flexion and extension of the knee [18].

Knee motion takes place in 3D space but is most considerable in the sagittal plane. The full extension is generally around 5-10 degrees, while the full flexion is somewhere between 120-150 degrees [17]. Moreover, there is also an active and passive motion when measuring the maximal flexion angle. The latter is usually 5-10 degrees larger. The active motion is when a subject applies muscles forces to reach a range of motion while the passive motion is when a measurer applies external forces to reach a certain level of flexion [19]. In some cases, the knee joint can bend the wrong way, also called hyperextension. It can occur to anyone and results in swelling, pain and tissue damage [39]. This project has not taken that into account since the IMUs are unable to measure the negative knee joint angle.

The inertial measurement unit (IMU) device will measure the flexion/extension angles in the knee joint from 0 – 180 degrees, since this is the relevant range for knee angles. The easiest way to measure the knee flexion/extension angle is to place one IMU on a subject’s thigh and another on the shank as shown in figure 3.

2.3 Software and sensors

This section will provide some background on the sensors and software that were used in order to develop the knee joint angle prototype.

2.3.1 Arduino

The Arduino Software (IDE) is an open-source electronics platform which allows a user to write program code and upload them on a board. It can be used to design and build devices that interact with the real world. An Arduino board contains a number of pins in two varieties: analog pins, which can read a range of values, and digital pins, which can read and write a single state.

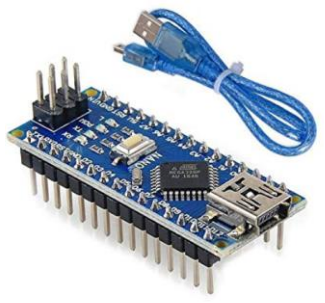


Figure 4: Arduino nano microcontroller board which was used for this thesis [20].

A user can connect different sensors, LEDs or buttons and program them accordingly to one’s needs. The Arduino IDE is a software that is written in Java and allows a user-friendly environment way to write code [20].

2.3.2 Inertial measurement unit (IMU)

Inertial measurement unit (IMU) is a self-contained system that uses gyroscopes, accelerometers and magnetometers to measures linear and angular motions. IMU sensors have gained a wide-spread use in mobile phones and gaming controllers. Keeping a quadcopter drone afloat despite pressure of winds, vacuuming house with a robot, or translating the movements using VR headset, are some of the many examples that use IMU sensors [21]. IMUs has a 3-axis accelerometer, a 3-axis gyroscope and an additional 3-axis magnetometer which makes it a 9-axis IMU. IMUs are often paired with other sensors which combines the data from multiple sensors.

An IMU provides 2 to 9 DOF (Degrees of Freedom) which basically refers to the number of different ways an object can move throughout the 3D space.

The accelerometer measures movement of the X, Y and Z axes which makes it possible to see how fast the sensor is ascending and descending as well as to derive speed over ground using acceleration data. They can even be used to detect earthquakes and are also used in medical devices such as bionic limbs or other artificial body parts [40].

The magnetometer can calculate which direction the sensor is facing with respect to magnetic north. It is possible to calculate the direction a sensor is facing and the angle at which it is leaning.

The gyroscope measures the orientation and angular velocity along the three axes, by integrating angular velocity over time, it can predict roll, pitch and yaw angles. This IMU module is useful since it allows us to know which way the sensor is facing at the time of data measurement [22]. The sensor used in this project is Adafruit's BNO055 and BNO080. Both sensors have 9-DOF and built in functions for Euler angles and quaternions. These topics will be described in the next section.

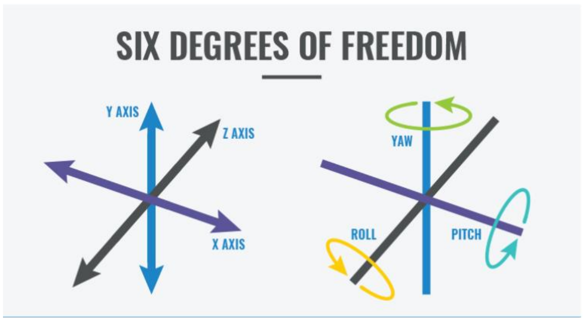


Figure 5: 6-DOF [21].

2.3.3 Euler angles vs. Quaternions

Euler angles allow a simple visualization of objects in 3D space but has a limited range of motion. This is where quaternions come in handy. A sensor which is using quaternions can orient anywhere in the space [24].

Euler angles

Euler's rotation theorem describes any rotation using three angles Φ, θ, and Ψ. If the rotation is described in terms of rotation matrices D, C, and B, then a general rotation A can be written as following:

$$A = BCD \qquad (1)$$

Euler angles are the three angles given the three rotation matrices. Matrix A can be written as

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \qquad (2)$$

Eq. 2 illustrates the "x-convention", which is the most common definition [25]. The rotation is given by Euler angles (Φ, θ, Ψ) where

Φ is the angle about the z-axis using D, which is the first rotation.

θ is the angle θ $\in$ $[0, \pi]$ given by the second rotation about the former x-axis (now x') using C.

Ψ is given by the third rotation about the former z-axis (now z') using B. These angles are often described as roll, pitch and yaw.

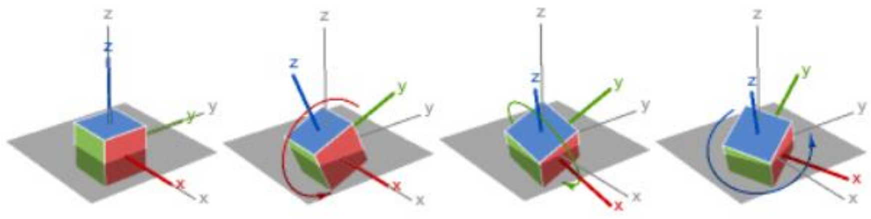


Figure 6: X-Y-Z rotations [24].

If the axis stays partially perpendicular, they are sufficient. However, a phenomenon called gimbal lock can occur as the axes rotate. An angle exists where two axes can describe the same rotation, it is then impossible to reorient without an external reference.

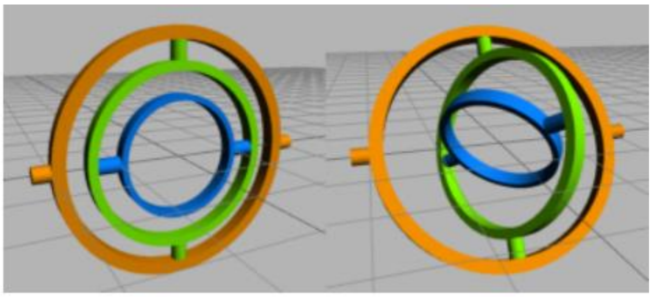


Figure 7: Gimbal lock (on the left) [24].

In fig. 7, the image on the left has blue and orange ring aligned. This means that the blue ring may rotate to allow pitch, the green ring will allow yaw, but the orange ring does not allow roll since it is aligned with the blue ring. The system is now rotating in 2D instead of 3D. This is one of the main reasons this project used quaternions instead of Euler angles to measure the absolute knee flexion/extension angle.

Quaternions

Quaternions is a system of numbers in 4 dimensions, where each quaternion consists of 4 scalar numbers, 3 imaginary and 1 real. Each of these imaginary dimensions has a unit value of the square root of -1. A quaternion can be shown as follows [23]:

$$q = w + ix + jy + kz \qquad (3)$$

Where w, x, y and z are real numbers and i, j, k quaternion units. Typically, w,x,y and z and kept in a range between -1 and 1 and also

$$1 = \sqrt{w^2 + x^2 + y^2 + z^2} \qquad (4)$$

These four numbers reorient vectors in a single rotation, without changes in length. Quaternions are free from gimbal lock and a lot simpler to compose. They are more compact compared to rotation matrices, more efficient and more numerically stable. Eq. 5 shows the formula for calculating the absolute angle θ between two quaternions, which was used to calculate the knee flexion/extension angle:

$$\theta = 2 * arccos(|q_w * q_{w_2} + q_x * q_{x_2} + q_y * q_{y_2} + q_z * q_{z_2}|) \qquad (5)$$

Where (q_w, q_x, q_y, q_z) are the quaternion values of the first IMU sensor and (q_{w_2}, q_{x_2}, q_{y_2}, q_{z_2}) the quaternion values of the second IMU sensor (section 3.1).

2.3.4 Sampling rate

Frequency rate will be mentioned throughout this report. It is the number of recordings every second and the desired frequency rate for this project is around 100 Hz. This is selected to capture dynamics of the human motion.

2.3.5 IMU comparison with other techniques

A literature study was performed to compare IMUs with other methods such as electro goniometer (potentiometer). A study was performed by Lousie Wrange and Nadia Wåhlin in 2018 to measure the knee joint angle using a potentiometer. According to that study, it was easy to implement a potentiometer instead of IMUs. A problem was that the sample rate was too low (2 Hz) and therefore provided less accurate gait patterns for dynamic motions such as walking or cycling [19]. IMUs can provide a sample rate over 100 Hz.

Another study by Giovanni Legnani "*A model of an electro-goniometer and its calibration for biomechanical applications*" was performed where several experimental tests were run to measure the knee joint angle. Two goniometers were combined to obtain the relative 3-D angular position. The study concluded that if the attached wires between the goniometers had twists or inaccurate positioning, the errors increased up to 3-4° [47].

A study from the Indian Institute of Technology Delhi used a Force Sensitive Resistor (FSR) to calculate the join angle of the knee. The system consisted two modules: a knee brace slider crank and a signal processing unit with wireless communication. FSR was used to calculate the force while walking whereas the bar slider crank system calculated the circular motion to reciprocating linear motion. The obtained results seemed accurate; however, the tests were run on a defined platform for short periods and the device was bigger compared to IMUs. It is hard to conclude how this module would perform during longer measurements in normal environments [48].

3. Implementation of the measurement system

This section presents a step by step description on how the knee angle measurement system was implemented with a general description of all the electrical equipment and material used.

3.1 Implementing the sensors (BNO055)

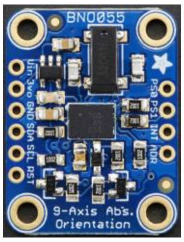


Figure 8: Adafruit's BNO055 Absolute Orientation Sensor [26].

Fig. 8 shows the 9-DOF sensor which can output various sensor data such as absolute orientation with Euler vectors (100 Hz), quaternion (100 Hz), angular velocity, temperature and much more [26]. The good thing about this sensor is that it already has built in algorithms that can provide data in minutes, instead of spending weeks or months fiddling with complex algorithms. It is possible to get the quaternion values from this sensor in 3D-space by calling the method getQuat() that was obtained from the BNO055 library from the GitHub repository [41].

At first, the BNO055 sensor was connected to the Arduino nano. BNO055 has a 3VO pin (3,3V output) which was connected directly into nano's 3,3 V. Ground (GND) pin was connected to GND on nano, SDA (I2C clock pin) to nano's A4 channel and SCL (I2C data pin) to A5. I2C is a serial protocol for two-wire interface to connect low-speed devices [28].

Second sensor was initialized on the same Arduino nano. It is done in the same way as the first sensor; except that the ADR pin also needs to be connected. ADR changes the default I2C address to avoid a crash with the first sensor. It was connected to 3,3 V pin, while Vin was connected to nano's 3,3 V. See Fig.10.

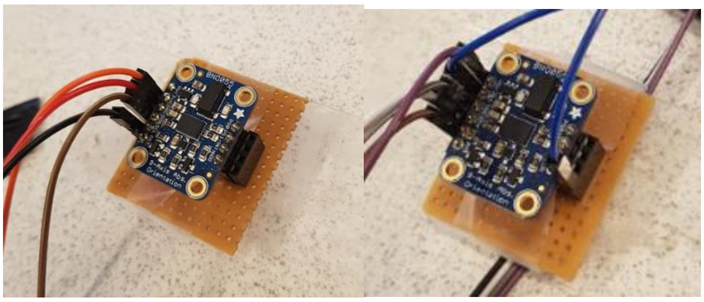


Figure 10: Connection of the first sensor (left) and second (right). The second sensor has an additional pin connected to ADR.

The next step was to implement the code for the sensors. Adafruit has provided example code on their website which was easy to follow. Two BNO055 objects were created and initialized (see appendix):

In the void loop, two new sensor events were created followed by getting the quaternion values of each sensor in IMU mode: The formula for the angle between two quaternions (eq. 5) was coded and thereafter printed out. The sensors were placed on a goniometer to see if the angle values were accurate.

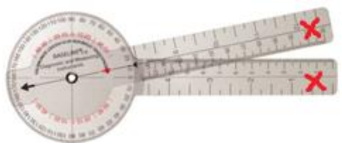
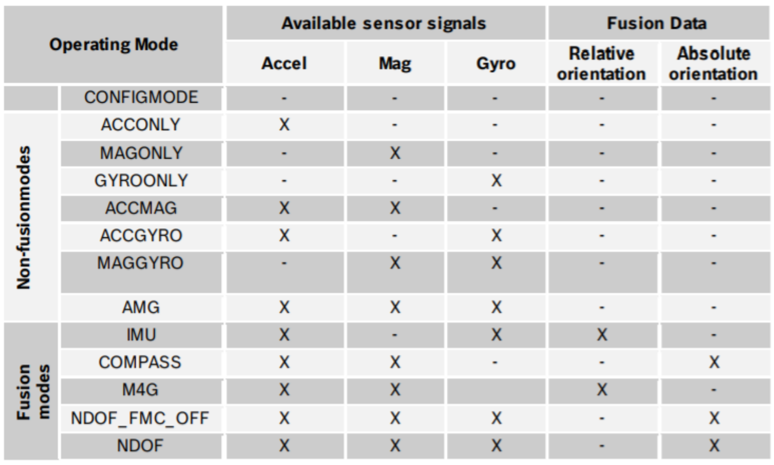


Figure 11: A goniometer. A BNO055 sensor was taped on each side (marked with red cross) to see if the calculated angle values were correct.

3.1.1 Relative and absolute orientation

BNO055 has two options for the orientation: absolute and relative. The absolute measurement provides a rotation relative to magnetic north, hence it uses all three sensors: accelerometer, magnetometer and gyro. The relative measurement is the orientation relative to the orientation the sensors had when the acquisition started. Both settings work by integrating the rotation, and applying a correction factor from the accelerometer, including a detection of the gravity vector. The relative orientation suspends the magnetometer. BNO055 has a built-in operation mode for IMU, where magnetometer is turned off. The magnetometer needs to calibrate each time the sensor is turned on, so does gyroscope and accelerometer. These can provide inaccurate values if not properly calibrated. Fig 9. represents the datasheet from Bosch that shows all available operating mode. Both relative and absolute orientations were tested on human subject and will be discussed in the upcoming sections.



Operating Mode		Available sensor signals			Fusion Data	
		Accel	Mag	Gyro	Relative orientation	Absolute orientation
	CONFIGMODE	-	-	-	-	-
Non-fusionmodes	ACCONLY	X	-	-	-	-
	MAGONLY	-	X	-	-	-
	GYROONLY	-	-	X	-	-
	ACCMAG	X	X	-	-	-
	ACCGYRO	X	-	X	-	-
	MAGGYRO	-	X	X	-	-
	AMG	X	X	X	-	-
Fusion modes	IMU	X	-	X	X	-
	COMPASS	X	X	-	-	X
	M4G	X	X		X	-
	NDOF_FMC_OFF	X	X	X	-	X
	NDOF	X	X	X	-	X

Figure 9: Datasheet of BNO055 Absolute Orientation Sensor which shows different operating modes [27].

3.1.2 Calibration procedure

To measure the relative angle, it is important for the sensors to start while they are aligned in a 0-degree position, i.e. the starting position (initial condition) must be 0°. The easiest way to do so is to place both sensors in the same position in a straight horizontal or vertical line before powering the nano as shown in fig. 12. It is also important to calibrate the gyroscope by leaving the sensors on the table (stable position).

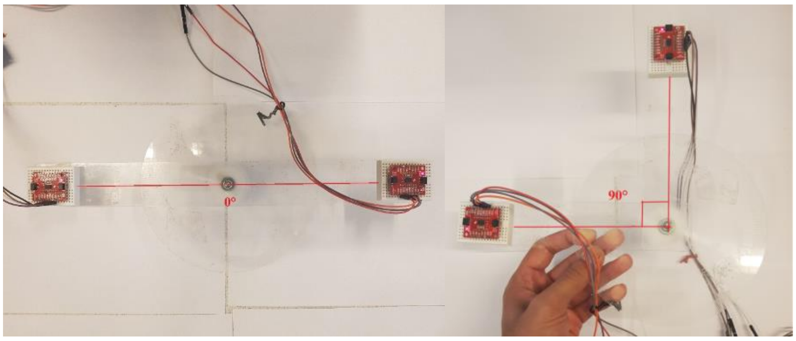


Figure 12: Sensors taped on a goniometer with the initial condition of aligning them in a 0-degree position before turning on the system (on left).

If for example the sensors are powered on in 90° (fig.12, right screenshot), the angle values will be inaccurate since as you move sensor 2 back into 0° position, the angle will be shown as 90° instead of 0°.

To measure the absolute orientation with NDOF operating setting, there is no need to place the sensors in a 0° position, but a proper calibration is needed. Adafruit provides a code to verify the calibration (see appendix). Gyroscope is calibrated when the sensors are stable. Calibrating accelerometer is done by rotation the sensors 45°, and magnetometer by drawing a figure “8” horizontally in space. This will allow the magnetometer to find the magnetic north [34]. According to the datasheet from Bosch, the fusion mode has a built-in auto calibration function that calibrates the sensors while using them [27], however, calibration in the beginning is still necessary to get the best possible results.

3.2 Implementing real time plots of data using LabVIEW

To visualize all the data from Arduino, a real time chart was implemented using LabVIEW (appendix 3). LabVIEW stands for Laboratory Virtual Instrument Engineering Workbench and is a visual programming language developed by National Instruments [29]. LabVIEW has built in functions that can read serial data from Arduino to LabVIEW’s VI (Virtual Instrument). A way to do so is to use VISA resource to get data serially.

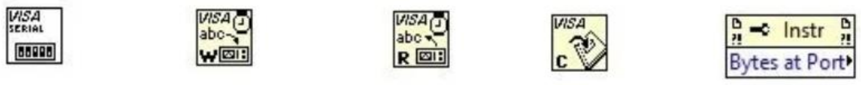


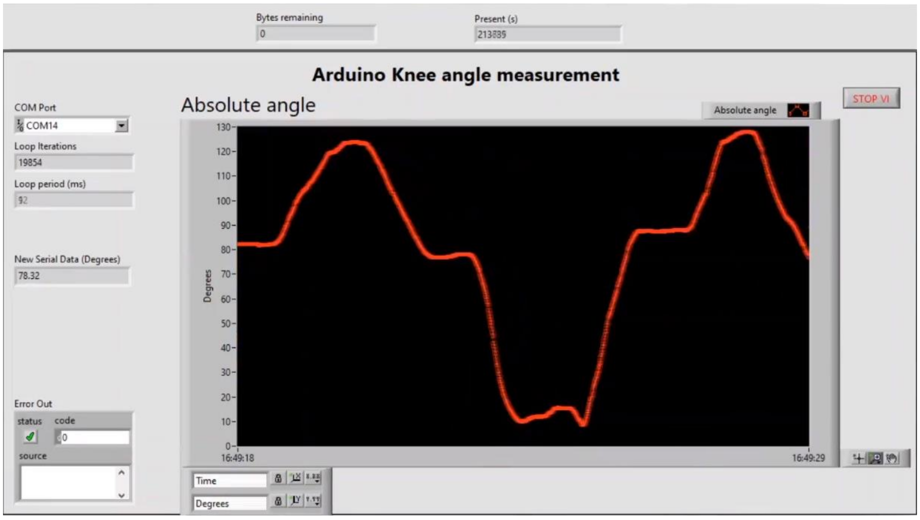
Figure 14: VISA blocks that will transfer data from Arduino to LabVIEW [29].

VISA serial is where the settings from Arduino are entered. The data has a baud rate of 115200 and is transferred in 8 bits. Baud rate refers to the number of signal or symbol changes that occur per second [30]

A while loop is created where strings are read from the COM port where serial communication is taking place. The strings are then converted into numbers with decimals and sent into an array which is displayed as a plot with angle values as Y-values and time as X-values.

It is important to close the COM port outside the while loop to avoid error on the next run. A TDMS file was created to save the measured data. It stores the data in a hierarchically organized file with high streaming speed. TDMS files can easily be converted into MATLAB.

The results after the LabVIEW implementation can be seen in fig.17.



Bytes remaining: 0
Present (s): 213889

Arduino Knee angle measurement

COM Port: COM14
Loop Iterations: 19854
Loop period (ms): 92
New Serial Data (Degrees): 78.32
Error Out
status: ✓ code: 0
source:

Figure 17: LabVIEW VI after its implementation.

It was vital to have a loop iteration less than 10 ms i.e. make the data run at 100 Hz. This was done by making some adjustments in the block diagram to shorten the time complexity. One of the adjustments was to insert a case structure with a check for a “true” condition as shown in fig.16. This avoided unnecessary loop iterations for the TDMS file, which slowed down the readings.

3.3 Testing of IMUs on an IRB 120 robot using LabVIEW

The IMUs were tested on ABB’s IRB 120 robot at the robotics lab at M section in LTH. The robot has motors that provide fast acceleration and can deliver accuracy and agility in any application [33]. Three different speed settings were tested: slow (37,5 deg/s), medium (125 deg/s) and fast (250 deg/s) [45]. These speeds were selected to resemble some extreme conditions that could be found in human gait, except that the movements of the robot were linear. Plots and parameters were obtained such as errors (RMSE), linearity, accuracy as well as results if one of the IMUs were tilted 10-15 degrees and will be presented in the results section.

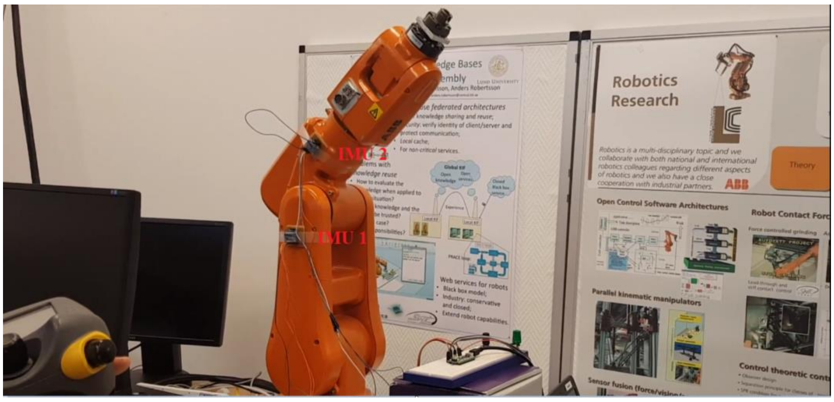


Figure 18: IMUs tested on a robot. Upper part of the robot arm was moving on different speeds, the results are presented in section 4.

3.4 Adding a Micro-SD breakout board into the system

After successful tests from the robot, a Micro-SD breakout board was implemented into the system. Data logging requires a bigger storage, at least several gigabytes of storage. Since this project requires a continuous 24 hours data logging, it is essential to use a microSD card reader. All data will be logged into the SD card and there are built in functions in Arduino that can be used to easily implement it.

SanDisk extreme microSD cards were used for this project. They have writing speeds up to 90 MB/S.

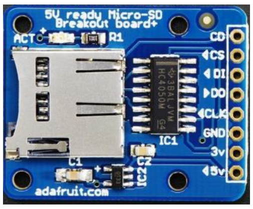


Figure 19: Adafruit's Micro-SD breakout board [31].

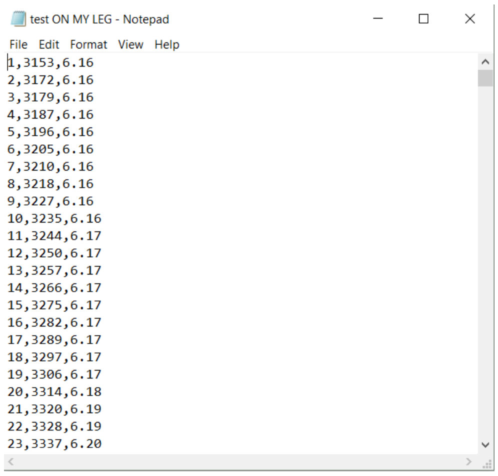
Adafruit provide Micro-SD breakout board (fig. 19) that can be connected to Arduino nano. Arduino IDE has a library called SD [42] which contains all the necessary methods for datalogging. The breakout board can be connected to either 5 V or 3,3 V. It was convenient to connect the breakout board to 3,3V, the same as IMUs. CLK pin was connected to nano's D13 (digital 13) pin, DO to D12, DI to D11, and CS to D10 on nano. SD cards require a lot of data transfer; therefore, they give the best performance when connected to the hardware SPI pins on a microcontroller. These pins are usually the digital pins depending on what kind of microcontroller one has.

After some trial and error, it was noted that the SD library was not so effective since the datalogging had a frequency rate of 50 Hz. The main goal was to achieve a rate around 100 Hz. This is when another updated SD library was discovered on Adafruit's website. This library is called SdFat [43], hence it was installed and tried instead.

After initializing the SD-card, a while loop was created that basically creates a "txt" file into SD-card memory, where all the data will be stored. This code can create data files from digits 00 up to 99 depending on if the file already exists or not. In the void loop, a string was created that decides the structure of the "txt" file. A counter "Id" was created to know the amount of measurements recorded into the text file (see appendix). ElapsedTime is the time since Arduino nano started, which is called by the millis() method. Lastly, relDeg is the measured angle values. This string is stored in the buffer of the SD-card each time a new data string is recorded.

The text file must close for the logging to be saved. Using the close() method stores the data into the "txt" file (from the buffer). However, this method costs a lot of time since the file is opened and closed each time it iterates through the loop. The frequency reached around 66 Hz, which was still not fast enough. This problem was fixed by storing the data every 10 seconds.

There is a method called flush(), which is the same as close(), except that after a close(), no further writes() can be done unless the file is opened again, but with flush(), it is still possible. Flush() ensures that any bytes written in the buffer are physically saved to the SD card, hence it is more effective than close(). This made the frequency reach 122 Hz, which fitted the desired sample rate. The SD-card has a storage of 32 GB.



```
1,3153,6.16
2,3172,6.16
3,3179,6.16
4,3187,6.16
5,3196,6.16
6,3205,6.16
7,3210,6.16
8,3218,6.16
9,3227,6.16
10,3235,6.16
11,3244,6.17
12,3250,6.17
13,3257,6.17
14,3266,6.17
15,3275,6.17
16,3282,6.17
17,3289,6.17
18,3297,6.17
19,3306,6.17
20,3314,6.18
21,3320,6.19
22,3328,6.19
23,3337,6.20
```

Figure 20: A screenshot of the text file which stored all the data, where first column is the number of measurements, second is the time in milliseconds since Arduino started, and third represents the angle in degrees [31].

3.5 Adding a button and LED

To make a more user-friendly device, a button and LED was added into the system. The idea is to start recording data and turning on the LED if the button is pressed. If it is turned off, the LED will turn off and the device will not further log any data into the SD card. This idea makes it easier for the users to control when they want to log data. It also makes it possible for the users to take breaks in-between from the recordings. This button makes sure that the nano is still running without logging any data, hence no starting calibration is needed as explained in section 3.1. A 1 kΩ resistor was connected in series with the LED.

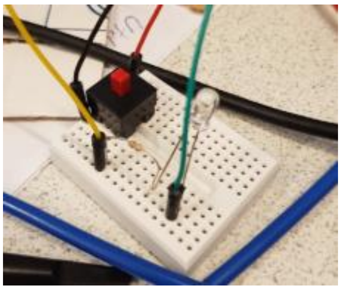


Figure 21: Light Emitting Diode and button.

3.6 Final circuit and design

After implementing all the above-mentioned parts, the final circuit looked as shown in figure 22 and 23.

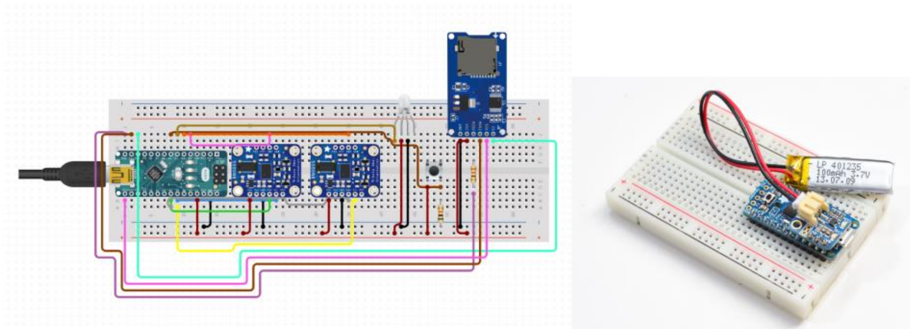


Figure 22: Circuit for the IMUs, nano, Micro-SD breakout board, button, LED, and Pro Trinket with a LiPo battery.

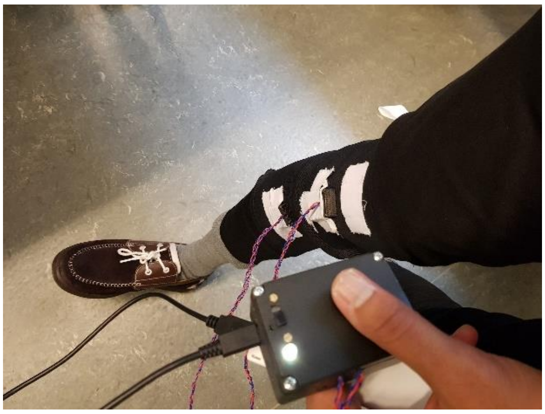


Figure 23: A screenshot of the final prototype, packed in a plastic box.

A plastic box was used to pack all the electronics. The slide switch button turns on and off datalogging with the LED. IMUs are packed inside plastic boxes created by a 3D-printer. The nano was replaced by a 3 V Pro Trinket. It works in the same way as Arduino nano, except that it has more pins and that it can run on 3V, compared to nano which runs at 5V. Pro trinket is perfect for portable projects. It also comes with a LiPoly/Lilon backpack which makes it possible to connect a LiPoly battery to the system. A 1500mAh LiPo battery and a power bank was used for this project.

3.7 Implementing the BNO080 IMU

BNO080 is the successor of BNO055 IMU also developed by Bosch. With its 9 DOF, it runs on powerful algorithms and produces accurate rotation vectors with quaternions. This IMU is better suited for motion tracking unlike BNO055 that comes with heavy drifting errors on motion tracking (section 5).

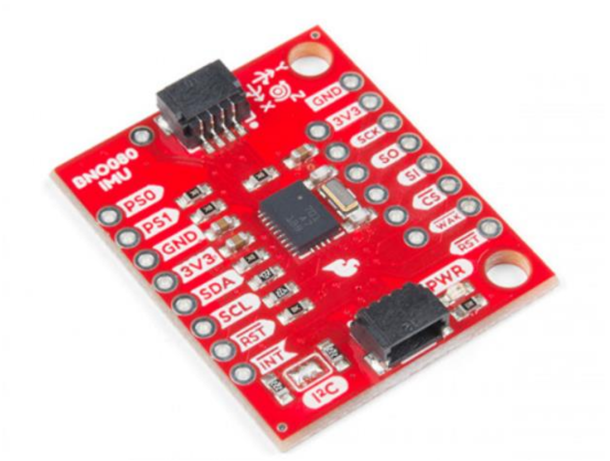


Figure 24: BNO080 IMU.

BNO080 is more complex than BNO055 and provides more functions such as dynamic and static calibration [37]. Dynamic calibration, which is the recommended setting, is the correction of the data returned from the sensors that varies with either time or change of environment. BNO080 calculates the adjustments and applies the correction factors as necessary in real time. Static calibration is the correction of non-varying parameters to the data returned from the sensors.

Implementation of BNO080 was done is the same way as BNO055 except for some differences. BNO080 has a built in I2C “jumper” as shown in figure 25. The jumper allows one to change the address of the BNO080 from 0x4B (default) to 0x4A. This allows one to connect two BNO080 sensors to a microcontroller.

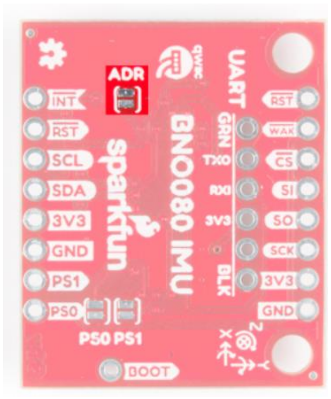


Figure 25: ADR jumper behind BNO080.

Soldering the ADR jumper allows the IMU to change its address. SDA (I2C data signal) and SCL (I2C clock signal) were connected to Arduino Nano’s A4, respectively A5 channels. BNO080’s power supply should be between 1.65 – 3,6 V therefore it shouldn’t be connected to Arduino’s 5 V input, but 3,3 V. Library for BNO080 was obtained from GitHub [44]. The code and functions for BNO080 can be found in appendix 2. According to the datasheet from Bosch, it is recommended to use 9-axis sensor fusion output (Rotation vector) for BNO080 [37]. This will provide the best possible data for motion tracking.

Same experiments as BNO055 were carried out by measuring the angle using quaternions on a goniometer (fig. 12) before running some measurements on a human subject.

4. Results and discussion on BNO055 - robot tests

This section will cover some of the results and errors from the testing phase, i.e. tests from the robot and goniometer. The purpose of this section is to see how accurate the sensors are as well as how their linearity is, which was done by creating plots in MATLAB.

4.1 Results from the goniometer

The results were very accurate with a maximum error of 1-2° as compared to the values from goniometer. The sensors were tilted in different directions to see if the angle values were unchanged, surprisingly, they were still accurate.

4.2 Results from robotics lab

The angle measurements tests on a robot were done at the M section in LTH with supervision of Anders Robertsson. As previously mentioned, the IMUs were tested on different speeds: slow, medium and fast. Robot flexion/extension angle was tested in a range between 0-130°.

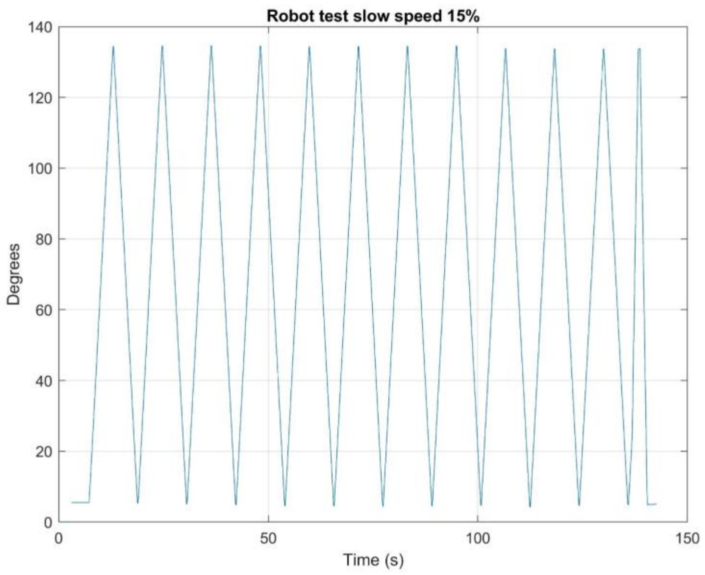


Figure 26: Results from the robotics lab, slow speed (15% of the total speed).

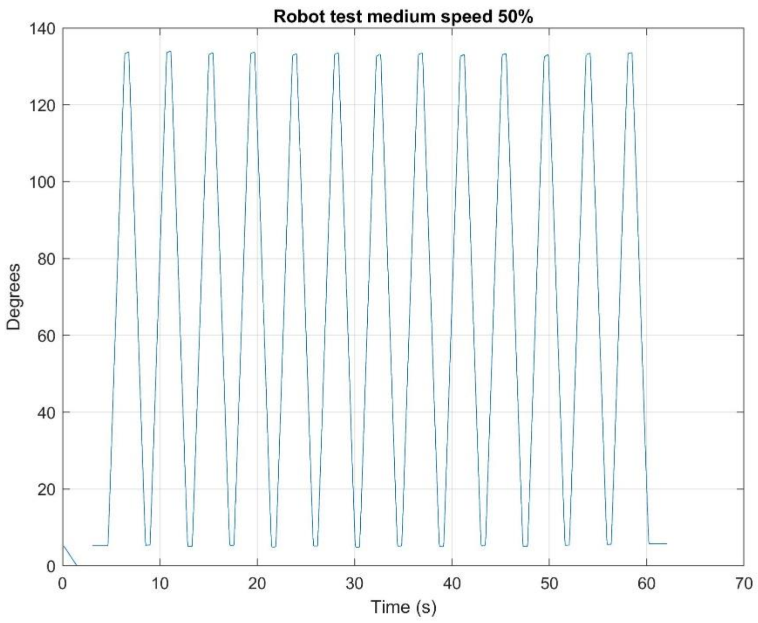


Figure 27: Results from the robotics lab, medium speed (50% of the total speed).

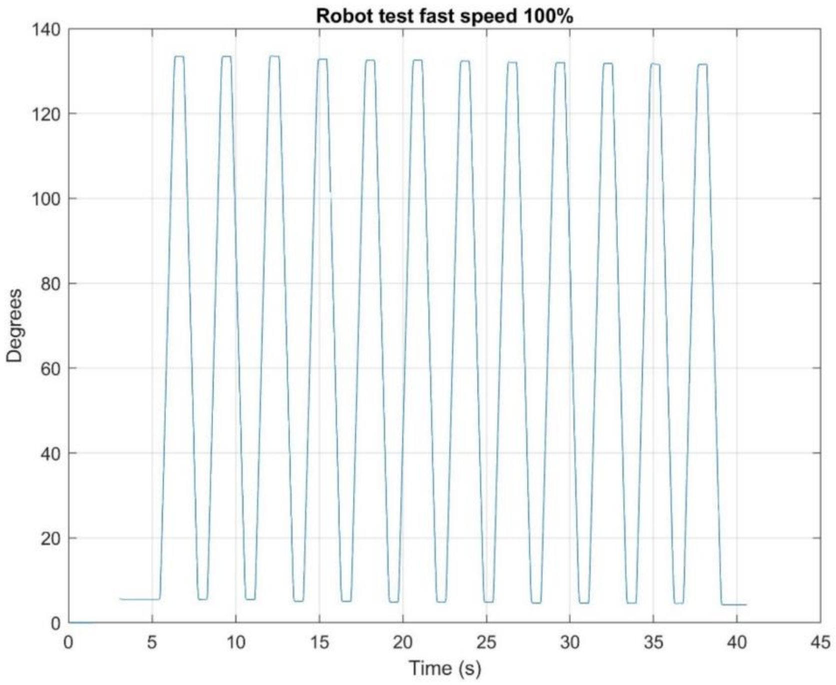


Figure 28: Results from the robotics lab, fast speed (100%).

It can be noted from fig. 23-25 that the plots are very linear with similar peaks. This is to be expected since the movements of the robot arm are linear. The real maximum angle of the robot was 130° which can be compared to the measured angle (tab. 1).

Real angle of robot	**Slow speed**	**Medium speed**	**Fast speed**
130°	134 – 5,27 = **129°**	133,62 – 5,27 = **128°**	133,5 – 5,54= **128°**

Table 1: Real angle of the robot arm compared with the measured values.

Even here the error is 1-2° compared with the real angle values.

When humans walk or run, a possible leg twist or turn can arise. This can make the IMUs tilt a certain degree. Upper IMU was tilted 10° off from its horizontal plane and was again tested in different speeds. The results were same as fig. 26-28.

Real angle of robot	**Slow speed - tilt**	**Medium speed - tilt**	**Fast speed - tilt**
130°	133,82 – 4,35 = **129°**	133 – 4,42 = **129°**	133 – 4,6= **128°**

Table 2: Real angle of the robot arm compared with the measured values with a tilt of 10°.

The angle is 1-2 degrees off from the real angle.

4.3 Root Mean Square Error (RMSE) of the tests

RMSE is the standard deviation of the residuals (prediction errors). How far away the data points are from the regression, is what residuals are a measure of [32]. RMSE is always between 0 and 1. If for example all points lie exactly on a line with positive slope, then RMSE will be 0, this means that there is no spread in the values of y around the regression line.

MATLAB was used to calculate the RMSE errors for the data presented in section 4.2 using Curve Fitting Toolbox [46]. The toolbox provides methods to assess goodness of fit for both linear and nonlinear parametric fits. Linear model Poly1, a first-degree polynomial was used for regression: $f(x) = p1 * x + p2$. The purpose of this calculation was to study how accurate the linearity is for the measured values if they are fitted with f(x).

Speed of robot arm	**RMSE**	**R-square**
15% (low)	0,1 – 0,2	1
50% (medium)	0,1- 0,2	1
100% (fast)	0,3 – 0,6	1

Table 3: RMSE errors for different velocity.

Fig. 30 illustrates some plots of how RMSE was determined. It was measured for both increasing and decreasing slopes for different peaks.

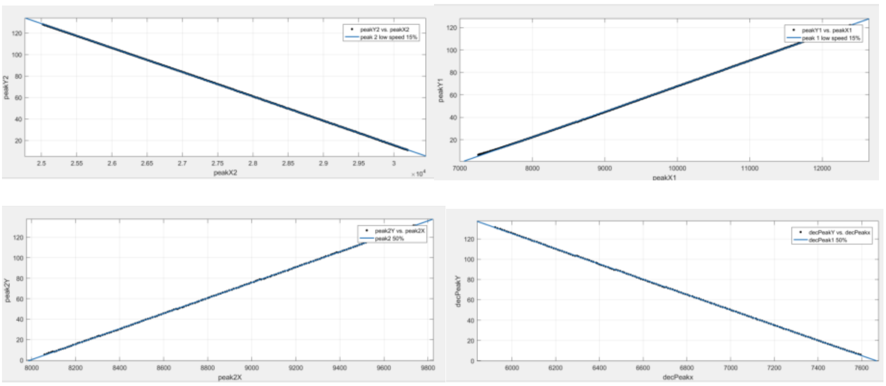


Figure 30: RMSE values for increasing and decreasing slopes.

Black dots are the measured values with the blue line being the linear regression.

Fig. 30 illustrate can be noted that the error is close to 0, and all the data points fit well on the line. Data points from 100% speed has a little higher RMSE value (0,3 – 0,6).

5. Results and discussion on BNO055 - human tests

Fig. 31-37 illustrates the acquired data from human knee joint measurements tested on the right leg of the author of this project. The results are from different operating modes with short and long periods.

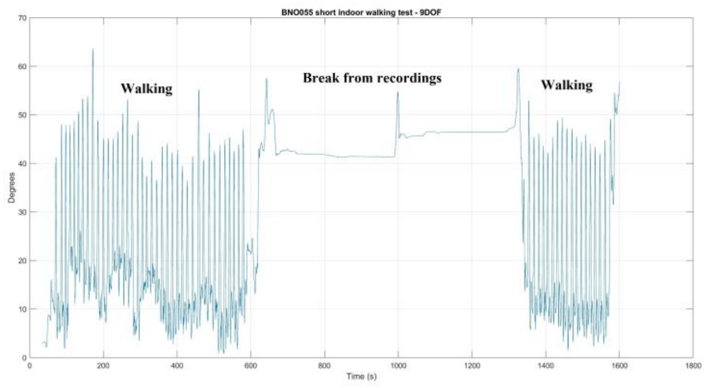


Figure 31: Angle measurement on human knee for short period of times using absolute orientation mode NDOF.

Fig. 31 illustrates that this experiment provided good results, since there are no drifting errors. The horizontal lines in the middle shows the breaks that were taken in between measurements. All sensors were calibrated before the measurements took place.

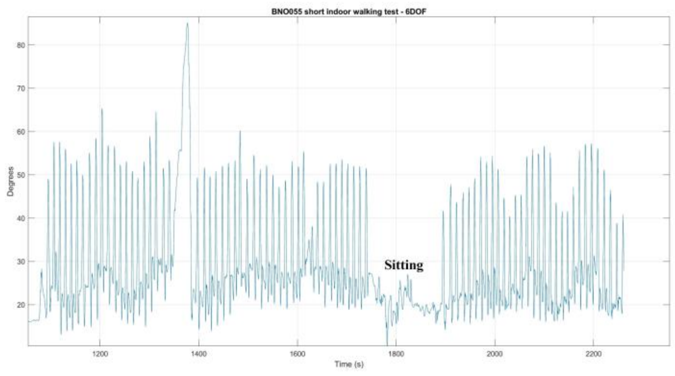


Figure 32: Angle measurement on human knee for short period using relative orientation mode IMU.

Fig. 32 represents obtained results using the relative orientation mode IMU. Only the gyroscope was calibrated with the initial angle close to 0°.

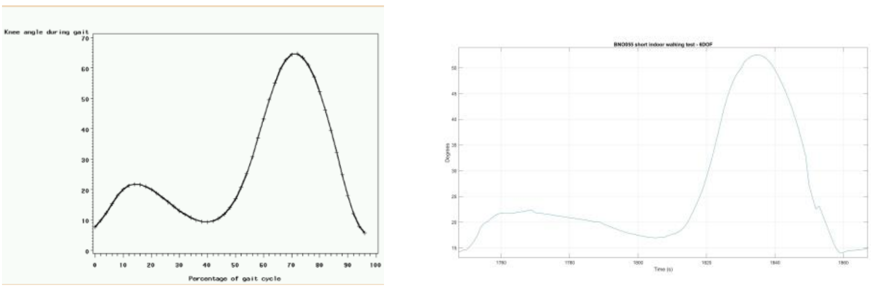


Figure 33: Knee angle during gait from the literature on left, and measured knee angle (right) from IMU (fig. 32).

Figure 33 illustrates how the knee joint angle is supposed to be during a gait cycle (left side) from a study [36]. It can be noticed that the measured angle from IMUs seem to be like that of the gait.

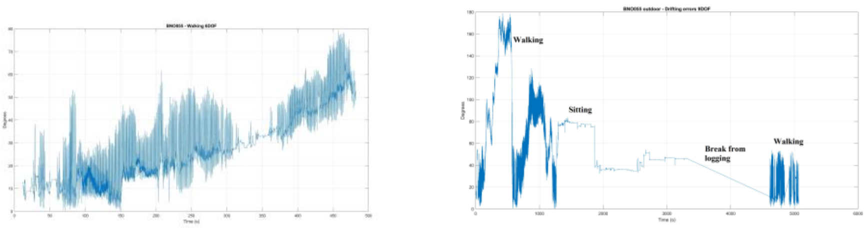


Figure 34: Angle measurement on human knee for a longer period 6-DOF (left) and 9-DOF (right) with drifting errors.

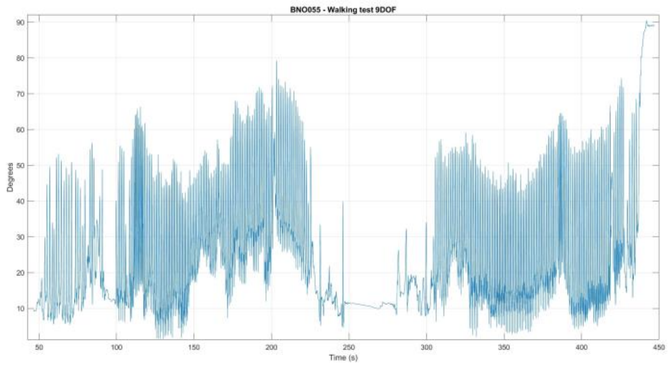


Figure 35: Angle measurement on human knee for 8 minutes using NDOF with slower movements.

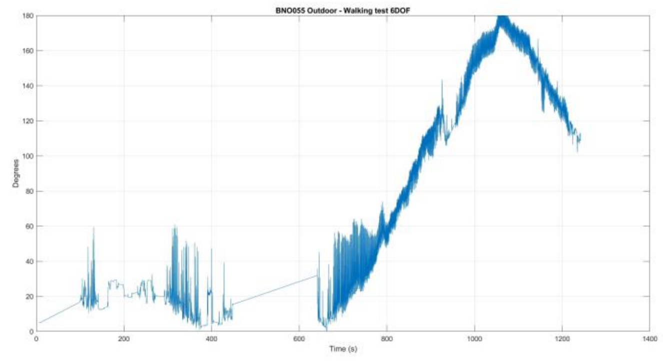


Figure 36: Walking experiment on human knee for 18 minutes with, relative orientation (IMU).

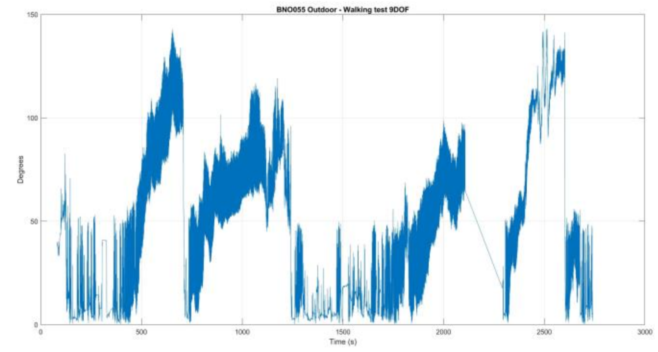


Figure 37: Angle measurement on human knee for 1 hour using absolute orientation.

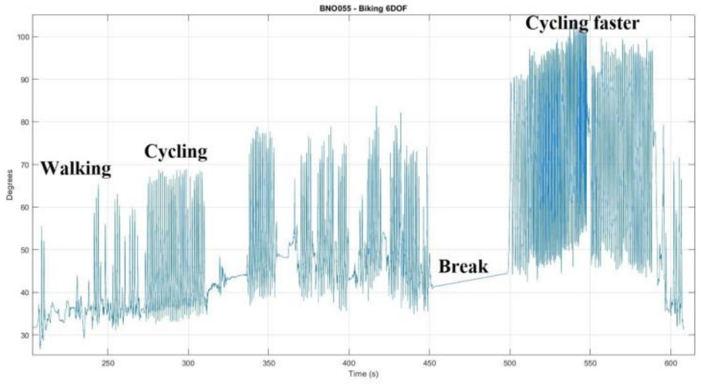


Figure 38: Angle measurement on human knee for 10 minutes on a bike using IMU.

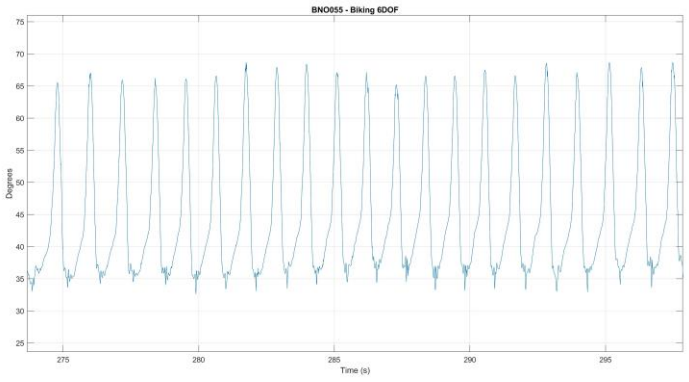


Figure 39: Angle measurement during cycling.

It can be noticed from fig. 34 – 38 that both settings provide drifting values if the measurements are done over a longer time span with continuous movements of the leg. Some drifting errors are > 20° making the IMU unusable. Measurements with slow movements over a shorter time span provide better results as shown in fig. 31-32. One possible reason can be that the BNO055 have an auto calibration function that cannot keep up with the leg movements over a longer period. The drifting values seem to be occurring from gyro and magnetometer. These were calibrated before the measurements started but get drifted off at some point. The tests were run for several days in different settings without any success. A contact with Bosh SensorTec was obtained but the only conclusion they provided was to properly calibrate the sensors before doing measurements, this was, however, done properly in all settings.

Overall, it seems like the relative orientation (IMU) is better than the absolute orientation (NDOF). The reason is mainly because there is no magnetometer that can be disturbed from other magnetic materials or ferrous. Fig. 37 illustatres that the drifts from absolute orientation are far worse than the ones from relative orientation (fig. 36). It also seems like it is easier to stabilize the calibration in IMU mode as compared to NDOF. The sensors were also tested in stationary positions and the results showed no drifting; hence the sensors work fine in stable positions.

According to Adafruit's forum, several people had problems with BNO055 calibration in forward motion. The unpredictable auto calibration running in the background of BNO055 sensors cannot be manually controlled, thus might be a cause for invalid results.

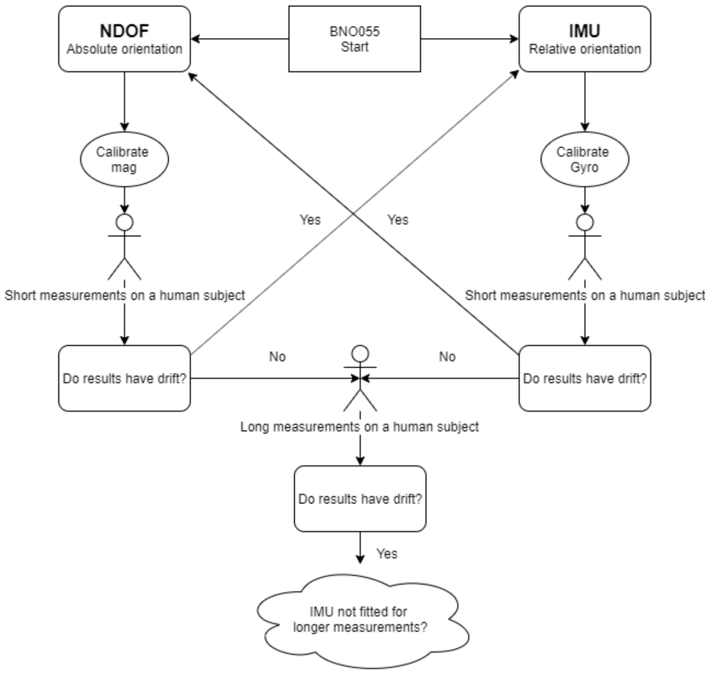


Figure 40: A flowchart on how measurements were tested.

Figure 35 describes the process of trial and errors that occurred when running some human knee joint tests. No operating mode gave proper results for longer periods on BNO055. After not fully succeeding in removing the drifting errors, another IMU (BNO080) was ordered and tested.

6. Results and discussion on BNO080 - human tests

After failing in getting drift-free values from BNO055 during longer measurements, it was replaced by BNO080, a more complex and accurate IMU. Due to lack of time, it was not possible to run as many tests with all the different operational modes as it was done on BNO055. BNO080 had a sampling frequency above 600 Hz, that was decreased to 100 Hz by adding some delay functions in the code. A few walking tests were performed on two main settings: Game Rotation Vector and Rotation vector. The first one is like BNO055's IMU (relative orientation) mode, i.e. the magnetometer is suspended whereas the latter is like BNO055's NDOF mode, where magnetometer provides support to reduce drift from gyroscope. Game rotation vector is recommended if the application requires an orientation estimate in an unstable magnetic field, while rotation vector, that uses 9-axis sensor fusion outputs is recommended for motion tracking in a relatively stable magnetic field. Figure 41-42 illustrates the obtained results from both operational settings.

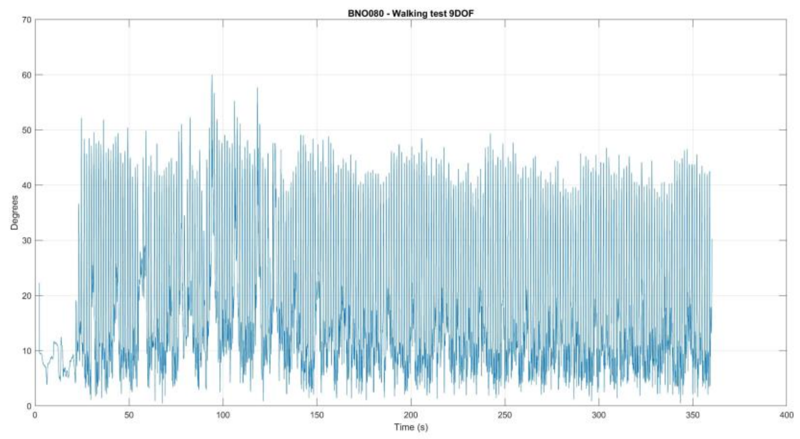


Figure 41: Indoor walking with 9-axis sensor fusion for several minutes (Rotation Vector).

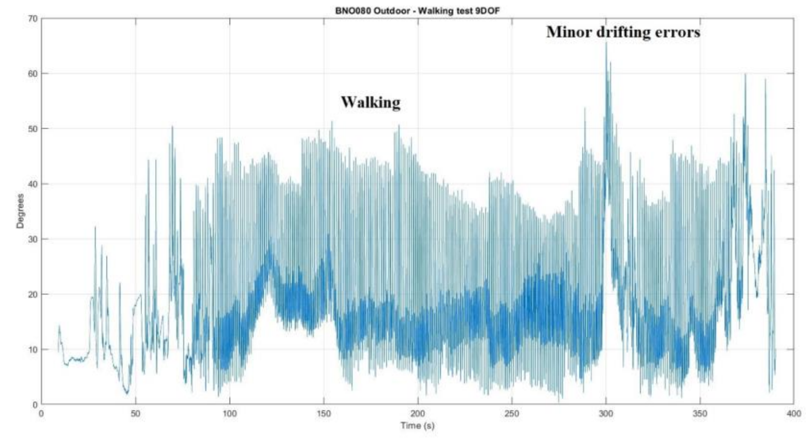


Figure 42: Outdoor walking for several minutes with 9-axis sensor fusion (Rotation Vector).

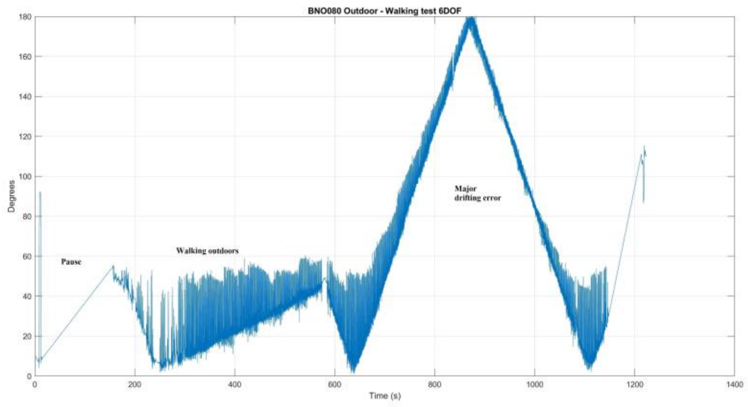


Figure 43: Outdoor walking for several minutes with 6-axis sensor fusion (Game rotation vector – no magnetometer).

The sensors used dynamic calibration in both settings. Even here, it shall be noted that the drifts occurred over longer measurements. In fig.43 they tend to reach up to 180° (very unstable). On the other side, rotation vector mode provided a lot of better results in both outdoor and indoor settings. Fig. 42 illustrates some small occurrences of drifting in some periods, however, BNO080 gets stable quicker than BNO055 after some time. BNO80 has room for improvement due to its several operational settings, that should be tested further in future over a longer period. BNO080 is not perfect, but a lot more improved compared to BNO055. It is hard to draw any conclusions now, but this is something that needs to be tested further in future with the different operational settings that fits the requirements to measure children with CP for 24 hours.

7. Final discussion

Measurements using BNO055 from the IRB 120 robot provided good results without any drifts. Since the movement of the robot arm is linear, it is easier for the calibration of the sensors to be stable compared with human leg motion. Robot tests were done in relative orientation which seem to be working very accurate.

When it comes to using a BNO055 sensor on humans, it might not be the best approach if the measurements are supposed to log data for several hours, that in this case is 24 hours. The sensors give accurate results when human knee motions are slow or done for a short period of time, in that case up to a minute or more. The auto calibration of the sensors cannot keep up with the long nonlinear movements. Relative orientation (IMU) mode provides more accurate results compared to NDOF (with magnetometer). IMU mode is recommended if BNO055 is used for measurements. Different experiments were carried out, such as walking indoors and outdoors. Indoor results provided fewer drifting errors compared to outdoor experiments, so did linear experiments (robot tests) or measurements for short periods that required minimal motion.

Due to lack of time, it was not possible to fully test BNO080 IMU, but the obtained results show that they have potential in providing accurate results for measurements that require a lot of motion tracking. They seem to handle forward motions with more accuracy and less drifting compared to BNO055, but unlike BNO055, the best possible setting is when the magnetometer is fused with the gyro and accelerometer. BNO080 is a successor of the BNO055 that offers more features and a significantly improved performance, but it comes with a complex interface.

To measure knee joint angle for 24 hours requires a high quality IMU that can keep its calibration stable in different environments. This is not an easy task since there are several motions and disturbances involved. It is, however, possible to get good results for shorter periods using both IMUs. One way to reduce the drifting is to shield the sensors. A layer of insulation and then a layer of aluminum foil around the sensors will lessen the disturbances, isolating the sensors from vibrations might also help in reducing drifting errors. BNO055 sensors were shielded in a plastic box with silicon around the box provided by Orteopedteknik in Lund, this, however, did not stop drifting errors from gyro. Shielding should be tested on BNO080 sensors since the obtained results from section 6 were without it.

Another important point is to properly calibrate the sensors before recording data. This will provide accurate results until the sensors become off-calibrated due to motion. BNO080 has functions that can auto calibrate the sensors after a certain period, or if it detects a change of environment, it is also possible to turn off dynamic calibration unlike BNO055. This is also a setting that should be tested in future. Even if the values get drifted off at some point, the sensors will recalibrate itself until the data is stable. When doing measurements on children for a longer time, one possible solution could be to study the intervals where no drifting occurs and ignore the parts that where drifting errors occurred.

8. Future work and developments

It was not possible to do 24 hours measurement on children with CP since the IMUs needs to be tested further before creating a final prototype.

BNO080 needs to be tested more for longer periods on human subjects. It is common that IMUs have drifting problems during motion tracking. To fix this problem is time consuming since the IMUs needs to be tested in all different operational settings until the best solution is found. It also depends on what kind of environment the measurements are taking place, i.e. if it is free from magnetic disturbances or not. Something that should be tested in future is to use EMC (electromagnetic compatibility) shielding. It is used to protect a sensitive signal from external electromagnetic signals, or to prevent stronger signal from leaking out and interfering with surrounding electronics. A way to do so is to use a metallic foil braid to shield the electrical wires. A plastic case might not have been the best shield against interferences, something that was used in this project.

Doing measurements on children can be a challenging task since they need to wear the sensors for 24 hours. It might be tiring or boring. A possible solution for future can be to create a more child-friendly device. For example, adding an entertainment function where they can get points depending on how long they wear the device for. Once they are done measuring for 24 hours, a reward can be given in form of some music or sound that congratulates them. Another thing to consider is what will happen if they accidentally press the on/off button. The sensors need to recalibrate again for the measurements to be correct, if this is done unproperly, the recorded data will thus be inaccurate. A possible solution can be to attach on/off button somewhere it is not easy for the kids to reach. There will be wires attached to their legs. Is there any way to hide them? This is also something that needs to be considered for the future. 24 hours of data is a lot. More work needs to be put into how to analyze the results and conclusions. Are certain intervals more interesting than others? A deeper understanding of the results is required by someone who has a more depth of knowledge on children with CP.

9. Conclusion

The required solution can measure the knee joint angle in children with CP for a short time (a few minutes) using IMUs, but not for 24 hours if a lot of movements/vibrations are involved. IMUs provide accurate knee joint mobility results, but lack in providing good results for continuous movements over longer periods. BNO080 has the potential to solve such tasks since it is designed for motion tracking over a longer period with high precision and quality, however, this is something that needs to be tested further. There are no research or scientific articles where they have been tested on humans over a longer period.

Appendix 1

Arduino code for BNO055:

```
#include <Wire.h>

#include <Adafruit_Sensor.h>

#include <Adafruit_BNO055.h>

#include <utility/imumaths.h>

#include <math.h>

//#include <SD.h>

#include "SdFat.h"

#include <SPI.h>

#define ACTIVATED LOW

#define FILE_BASE_NAME "Data"

Adafruit_BNO055 bno = Adafruit_BNO055(55, BNO055_ADDRESS_A);

Adafruit_BNO055 bno1 = Adafruit_BNO055(55,BNO055_ADDRESS_B);

SdFat SD;

//SdFile myFile;

File myFile;

unsigned long id = 1;

const int chipSelect = SS;

#define SD_CS_PIN SS

unsigned long time;

long timeInterval = 10000;

long previousTime = 0;

const int buttonPin = 9;

int buttonState = 0;

long previousMillis = 0;

unsigned long StartTime = millis();

int LED = 6;

const uint8_t BASE_NAME_SIZE = sizeof(FILE_BASE_NAME) -1;

char fileName[] = FILE_BASE_NAME "00.txt";
```

```
void setup(void)

{

  //115200

  Serial.begin(115200);

  Serial.println(F("Relative angle measurement")); Serial.println("");

  Serial.print(F("Initializing SD card..."));

  /* Initialise the sensor */

  if(!bno.begin())

  {

    /* There was a problem detecting the BNO055 ... check your connections
*/

    Serial.print(F("Ooops, no BNO055 detected ... Check your wiring or I2C
ADDR!"));

    while(1);

  }

    if(!bno1.begin())

  {

    /* There was a problem detecting the BNO055 ... check your connections

    Serial.print(F("Ooops, no BNO055 second detected ... Check your wiring
or I2C ADDR!"));

    while(1);

  }

//Calibrating the sensors

  uint8_t system, gyro, accel, mag = 0;

//Gyro != 3 is calibration for IMU, and System!=3 is calibration for NDOF

 while(gyro!=3) {

   bno.getCalibration(&system, &gyro, &accel, &mag);

   bno1.getCalibration(&system, &gyro, &accel, &mag);

 }

  pinMode(SS, OUTPUT);

//Installing the SD microchip

if (!SD.begin(SD_CS_PIN)) {

    Serial.println(F("initialization failed!"));

   exit(0);
  }
  Serial.println(F("initialization done."));
// Creating a text file in SD card.
  while(SD.exists(fileName)) {
    if(fileName[BASE_NAME_SIZE +1] != '9'){
      fileName[BASE_NAME_SIZE +1]++;
    } else if (fileName[BASE_NAME_SIZE] != '9') {
      fileName[BASE_NAME_SIZE +1] = '0';
      fileName[BASE_NAME_SIZE]++;
    } else {
      Serial.println(F("Can't create file name"));
      return;
    }
  }
  myFile = SD.open(fileName, O_CREAT | O_WRITE);
  if(!myFile) {
    Serial.println(F("Open failed"));
    return;
  }
  Serial.print(F("Opened: "));
  Serial.println(fileName);
//Set the operation mode for BNO055
bno.setMode(bno.OPERATION_MODE_IMUPLUS);
bno1.setMode(bno.OPERATION_MODE_IMUPLUS);
  bno.setExtCrystalUse(true);
  //bno.begin(bno.OPERATION_MODE_IMUPLUS);
  bno1.setExtCrystalUse(true);
//Code for the button.
pinMode(buttonPin, INPUT);
digitalWrite(buttonPin,HIGH);
pinMode(LED, OUTPUT);
//When all the initialization is complete, the LED will start blinking
while(!digitalRead(9)) {
  //Serial.println(F("Press the button to start recording data"));
  digitalWrite(LED, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(100);                       // wait for a second
  digitalWrite(LED, LOW);    // turn the LED off by making the voltage LOW
  delay(100);
  }
}
void loop(void)
{
 unsigned long CurrentTime = millis();
 unsigned long ElapsedTime = CurrentTime - StartTime;
  buttonState = digitalRead(buttonPin);
  //digitalWrite(LED, HIGH);

  /* Get a new sensor event */
  sensors_event_t event;
  bno.getEvent(&event);
  bno1.getEvent(&event);

//Get quaternion values.
imu::Quaternion quat = bno.getQuat();
imu::Quaternion quat2 = bno1.getQuat();

quat.normalize();
quat2.normalize();
//imu::Quaternion res = quat.conjugate()*quat2;
// Read the state of the pushbutton value
buttonState = digitalRead(buttonPin);

//Formula for the absolute angle between two quaternions.
double relDeg = 57.2958*2*acos(abs(quat.w()*quat2.w() + quat.x()*quat2.x() +
quat.y()*quat2.y()+quat.z()*quat2.z()));
//double invRes = 57.2958*2*acos(res.w());
//Creates a string when the button is pressed. The string is saved into the
text file.
 if(buttonState == HIGH){
  digitalWrite(LED, HIGH);

String  dataString  =  String(id)  +  ","  +  String(ElapsedTime)  +  ","  +
String(relDeg);

myFile.println(dataString);

//Serial.println(dataString);

  id++;

//Run flush every 10 seconds to avoid slow readings.

if(CurrentTime - previousTime >= timeInterval){

myFile.flush();

previousTime = CurrentTime;

//Serial.println(F("Now flushing"));

}

  Serial.println(dataString);

 } else {

  digitalWrite(LED, LOW);

//  Serial.println(F("Not recording"));

 }
```

Appendix 2

Arduino code for BNO080:

```
#include <Wire.h>

#include "SparkFun_BNO080_Arduino_Library.h"

#include <Adafruit_Sensor.h>

//#include <utility/imumaths.h>

#include <math.h>

//#include <SD.h>

#include "SdFat.h"

#include <SPI.h>

#define ACTIVATED LOW

#define FILE_BASE_NAME "Data"


BNO080 myIMU; //Open I2C ADR jumper - Goes to Address 0x4B

BNO080 myIMU2; //Closed I2c ADR jumper - Goes to Address 0x4A

SdFat SD;

//SdFile myFile;

File myFile;

unsigned long id = 1;

#define SD_CS_PIN SS

long timeInterval = 10000;

long previousTime = 0;

const int buttonPin = 9;

int buttonState = 0;

unsigned long StartTime = millis();

int LED = 6;

const uint8_t BASE_NAME_SIZE = sizeof(FILE_BASE_NAME) -1;

char fileName[] = FILE_BASE_NAME "00.txt";


void setup(void)

{

  //115200 baud rate
  Serial.begin(115200);

  Serial.println(F("Relative angle measurement")); Serial.println("");

  Serial.print(F("Initializing SD card..."));

  Wire.begin();

  //Wire.setClock(0.1);

  /* Initialise the sensor */

  if(myIMU2.begin(0x4B) == false)

  {

    /* There was a problem detecting the BNO080 ... check your connections
*/

    Serial.print(F("Ooops, no BNO080 main sensor detected ... Check your
wiring or I2C ADR JUMPER!"));

    while(1);

  }

    if(myIMU.begin(0x4A) == false)

  {

    /* There was a problem detecting the BNO080 ... check your connections
*/

    Serial.print(F("Ooops, no BNO080 second sensor detected ... Check your
wiring or I2C ADR JUMPER!"));

    while(1);

  }

//Use this setting if we want the magnetometer to be suspended.

//myIMU.enableGameRotationVector(50); //Send data update every 50ms.

  //myIMU2.enableGameRotationVector(50); //Send data update every 50ms.

//This uses all 9DOF fusion mode with magnetometer.

  myIMU.enableRotationVector(50);

  myIMU2.enableRotationVector(50);

pinMode(SS, OUTPUT);

if (!SD.begin(SD_CS_PIN)) {

    Serial.println(F("initialization failed!"));

   exit(0);

  }

  Serial.println(F("initialization done."));
```

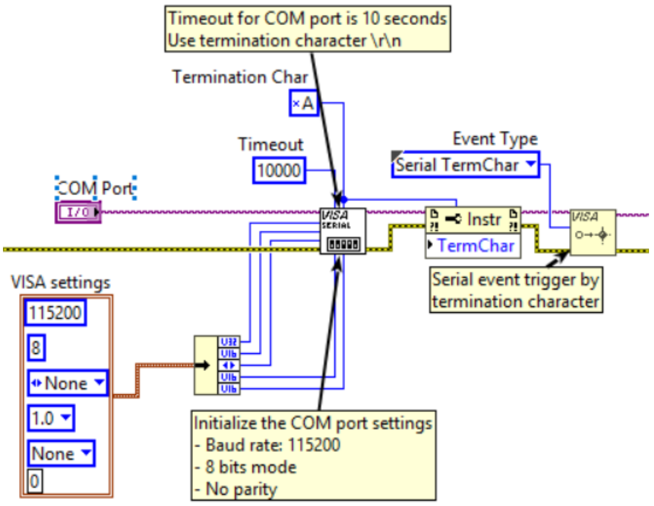
```
//For creating a text file in SD card.
  while(SD.exists(fileName)) {
    if(fileName[BASE_NAME_SIZE +1] != '9'){
      fileName[BASE_NAME_SIZE +1]++;
    } else if (fileName[BASE_NAME_SIZE] != '9') {
      fileName[BASE_NAME_SIZE +1] = '0';
      fileName[BASE_NAME_SIZE]++;
    } else {
      Serial.println(F("Can't create file name"));
      return;
    }
  }
  myFile = SD.open(fileName, O_CREAT | O_WRITE);
  if(!myFile) {
    Serial.println(F("Open failed"));
    return;
  }
  Serial.print(F("Opened: "));
  Serial.println(fileName);
  //myFile.close();

//enable dynamic calibration for accel, gyro and mag
myIMU.calibrateAll();
myIMU2.calibrateAll();

//Code for the button.
pinMode(buttonPin, INPUT);
digitalWrite(buttonPin,HIGH);
pinMode(LED, OUTPUT);

while(!digitalRead(9)) {
  //Serial.println(F("Press the button to start recording data"));
  digitalWrite(LED, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(100);                       // wait for a second
  digitalWrite(LED, LOW);    // turn the LED off by making the voltage LOW

  delay(100);

  }

}

void loop(void)

{

 unsigned long CurrentTime = millis();

 unsigned long ElapsedTime = CurrentTime - StartTime;

  buttonState = digitalRead(buttonPin);

  //digitalWrite(LED, HIGH);

double relAngle;

//This must be written to check if the data exists, otherwise there won’t be
any data.

  if (myIMU.dataAvailable() == true && myIMU2.dataAvailable() == true)

    {

//Normalizing the quaternions

Qnormalize();

Q2normalize();

//Formula for finding the angle in degrees.

relAngle  =  57.2958*2*acos(abs(myIMU.getQuatReal()*myIMU2.getQuatReal()  +
myIMU.getQuatI()*myIMU2.getQuatI()                                         +
myIMU.getQuatJ()*myIMU2.getQuatJ()+myIMU.getQuatK()*myIMU2.getQuatK()));

    }

// Read the state of the pushbutton value

buttonState = digitalRead(buttonPin);

//Creates a string when the button is pressed. The string is saved into the
text file.

 if(buttonState == HIGH){

  digitalWrite(LED, HIGH);

String  dataString  =  String(id)  +  ","  +  String(ElapsedTime)  +  ","  +
String(relAngle);

myFile.println(dataString);



//Serial.println(dataString);

  id++;
```

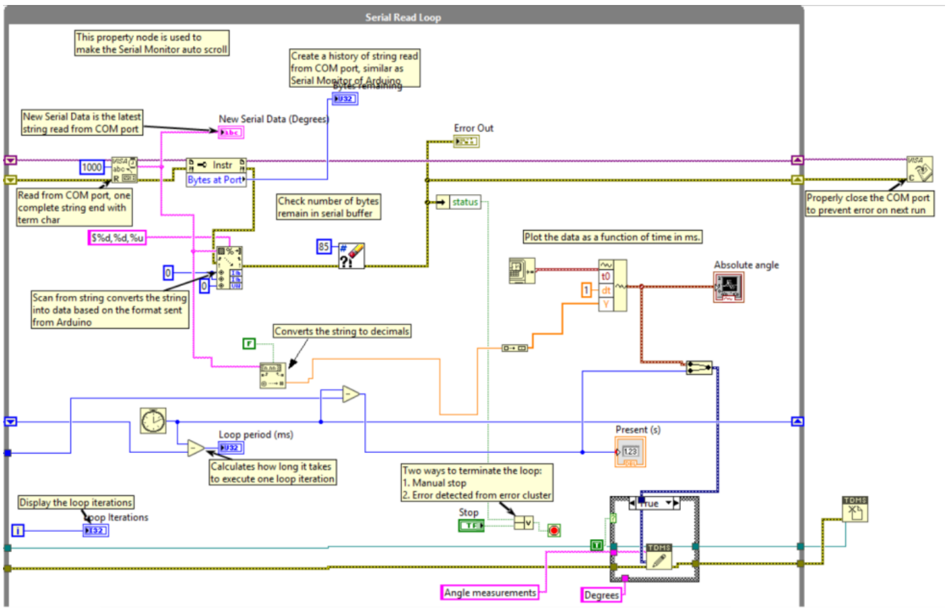
```
//Run flush every 10 seconds to avoid slow readings.

if(CurrentTime - previousTime >= timeInterval){

myFile.flush();

previousTime = CurrentTime;

}

 Serial.println(dataString);

  delay(5);

 } else {

  digitalWrite(LED, LOW);

//  Serial.println(F("Not recording"));

 }

}

//Methods to normalize the quaternions.

void Qnormalize() {

double n;

n          =          sqrt(myIMU.getQuatReal()*myIMU.getQuatReal()          +
myIMU.getQuatI()*myIMU.getQuatI()   +   myIMU.getQuatJ()*myIMU.getQuatJ()   +
myIMU.getQuatK()*myIMU.getQuatK());


double w = myIMU.getQuatReal();

double I = myIMU.getQuatI();

double J = myIMU.getQuatJ();

double K = myIMU.getQuatK();


w /= n;

I /= n;

J /= n;

K /= n;

}


void Q2normalize() {

double n;

n         =         sqrt(myIMU2.getQuatReal()*myIMU2.getQuatReal()         +
myIMU2.getQuatI()*myIMU2.getQuatI()  + myIMU2.getQuatJ()*myIMU2.getQuatJ()  +
myIMU2.getQuatK()*myIMU2.getQuatK());
```

```
double w = myIMU2.getQuatReal();

double I = myIMU2.getQuatI();

double J = myIMU2.getQuatJ();

double K = myIMU2.getQuatK();

w /= n;

I /= n;

J /= n;

K /= n;

}
```

Appendix 3

LabVIEW front panel and Block Diagram




Serial Read Loop
This property node is used to make the Serial Monitor auto scroll
Create a history of string read from COM port, similar as Serial Monitor of Arduino
Bytes remaining
U32
New Serial Data is the latest string read from COM port
New Serial Data (Degrees)
abc
Error Out
1000
Instr
Bytes at Port
Read from COM port, one complete string end with term char
Check number of bytes remain in serial buffer
status
$%d,%d,%u
85
0
0
Scan from string converts the string into data based on the format sent from Arduino
Plot the data as a function of time in ms.
t0
dt
Y
1
Absolute angle
Converts the string to decimals
F
Loop period (ms)
U32
Calculates how long it takes to execute one loop iteration
Present (s)
123
Two ways to terminate the loop:
1. Manual stop
2. Error detected from error cluster
Stop
TF
True
T
TDMS
Display the loop iterations
Loop Iterations
i
I32
Angle measurements
Degrees
Properly close the COM port to prevent error on next run
VISA

References

[1] Nation Health Service, NHS, ‘’Overview: Cerebral palsy”, [online], last access: 15.03.2017. link: https://www.nhs.uk/conditions/cerebral-palsy/

[2] Mayo Clinic, ‘’Cerebral Palsy’’, [online], last access: 2019, link: https://www.mayoclinic.org/diseases-conditions/cerebral-palsy/symptoms-causes/syc-20353999

[3] Hjärnfonden, ‘’Cerebral Pares: Visste du att”, [online], last access: 2019. link: https://www.hjarnfonden.se/om-hjarnan/diagnoser/cp-skada/#

[4] Teresa Pountney, E.M Green, ‘’Hip dislocation in cerebral palsy”, 2006, NCBI, link: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1420759/

[5] CPUP, ‘’Information för dig med CP”, 2019, Uppföljningsprogram för cerebral pares, link:
http://cpup.se/vad-ar-cp/#CP

[6] Dr. Nicolas Gutierrez, Cerebral Palsy Group ‘’What is cerebral palsy?”, 2019 [online], link: https://cerebralpalsygroup.com/cerebral-palsy/

[7] Neurorapporten 2019 ,Neuro, ‘’Vård och behandling av cerebral pares (cp)”, 2019 [online], link:
https://neuro.se/diagnoser/cerebral-pares-cp/vaard-och-behandling-av-cerebral-pares-cp/

[8] NHS ,”Occupational therapy”, 2017 [online], link:
https://www.nhs.uk/conditions/occupational-therapy/

[9] CPUP ,‘’Uppföljningsprogram för Cerebral Pares” 2019 [online], link:
https://cpup.se/vad-ar-cpup/

[10] Cerebral Palsy Alliance, ‘’Gross Motor Function Classification System (GMFCS)” 2018 [online], link:
https://cerebralpalsy.org.au/our-research/about-cerebral-palsy/what-is-cerebral-palsy/severity-of-cerebral-palsy/gross-motor-function-classification-system/

[11] Anatomy Next,‘’Knee Joint” 2019 [online], link:
https://www.anatomynext.com/knee-joint/

[12] Tim Barclay, PhD, innerbody,‘’Hinge Joint” 2018 [online], link:
https://www.innerbody.com/image_skel07/skel31.html

[13] Elina, Vector, 123rf,‘’Knee anatmoy” 2019 [online], link:
https://www.123rf.com/photo_95643732_stock-vector-anatomy-of-the-knee-joint-side-view-template-for-training-a-medical-surgical-poster-traumatology-pag.html

[14] Kathleen Davis FNP, MedicalNewsToday,‘’How to prevent and treat knee injuries” 2017 [online], link:
https://www.medicalnewstoday.com/articles/299204.php

[15] Young Choi, MD, CiOS,”Anterior Knee Pain in Patients with Cerebral Palsy” 2014 [online], link:
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4233222/

[16] Shalmali Pal, lermagazine,”Cerebral palsy and knee pain: management tips” 2015 [online], link:
https://lermagazine.com/issues/february/cerebral-palsy-and-knee-pain-management-tips

[17] Arun Pal Singh, Bone and Spine,”Knee range of motion and movements” 2019 [online], link:
https://boneandspine.com/knee-range-of-motion/

[18] Teachmeanatomy,” Anatomical Terms of Movement” 2019 [online], link:
https://teachmeanatomy.info/the-basics/anatomical-terminology/terms-of-movement/

[19] Louise Wrange, Nadia Wåhlin, BME, “Development of techniques for measuring the mobility of knee joints in children with Cerebral Palsy” 2018.

[20] Arduino,” Getting started with Arduino products” 2019 [online], link:
https://www.arduino.cc/en/Guide/HomePage

[21] hillcrestlabs, CEVA,” What is an IMU sensor” 2018 [online], link:
https://www.hillcrestlabs.com/posts/what-is-an-imu-sensor

[22] Xinabox wiki,” 9DOF- Nine Degrees of Freedom” 2017 [online], link:
https://wiki.xinabox.cc/9DOF_-_Nine_Degrees_of_Freedom

[23] Euclidean space,” Maths - Quaternions” 2017 [online], link:

https://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/

[24] All about circuits, Mark Hughes, " Capturing IMU Data with a BNO055 absolute orientation Sensor" 2017 [online], link: https://www.allaboutcircuits.com/projects/bosch-absolute-orientation-sensor-bno055/

[25] MathWorld, Wolfram, " Euler Angles" 2019 [online], link: http://mathworld.wolfram.com/EulerAngles.html

[26] Adafruit, BNO055 Absolute Orientation Sensor, "Overview" 2019 [online], link: https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview

[27] Bosch, BNO055 datasheet Absolute Orientation Sensor, section 3.3, p.20 2019 [online], link: https://cdn-shop.adafruit.com/datasheets/BST_BNO055_DS000_12.pdf

[28] I2C info "Interface and Protocol", 2019 [online], link: https://i2c.info/

[29] instructables circuits "Arduino and LabVIEW", 2019 [online], link: https://www.instructables.com/id/Arduino-and-LabVIEW/

[30] ElectronicDesign "Baud Rate", 2019 [online], link: https://www.electronicdesign.com/communications/what-s-difference-between-bit-rate-and-baud-rate

[31] Adafruit, "Micro SD card Breakout Board Tutorial", 2019 [online], link: https://learn.adafruit.com/adafruit-micro-sd-breakout-board-card-tutorial/introduction

[32] Statistics how to, "RMSE: Root Mean Square Error", 2019 [online], link: https://www.statisticshowto.datasciencecentral.com/rmse/

[33] ABB, "IRB 120", 2019 [online], link: https://new.abb.com/products/robotics/industrial-robots/irb-120

[34] Bosch Sensortec, "BNO055", 2019 [online], link: https://www.bosch-sensortec.com/bst/products/all_products/bno055

[35] Adafruit forum, “BNO055 calibration & stops recording in forward motion”, 2017 [online], link: https://forums.adafruit.com/viewtopic.php?f=19&t=78459&p=603120&hilit=auto+calibration#p603120

[36] Dave Thompson, “Knee movement during gait“, 2003 [online], link: https://ouhsc.edu/bserdac/dthompso/web/pk/kneegraf.htm

[37] hillcrestlabs, “BNO080 SiP Datasheet “, 2017 [online], link: https://www.digikey.com/en/datasheets/hillcrestlaboratoriesinc/hillcrest-laboratories-inc5a05f340566d07c196001ec1#pf39

[38] Dhiren Ganjwala, Hitesh Shah, Indian Journal of Orthopaedics “Management of knee problems in spastic cerebral palsy “, 2017 [online], link: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6394172/

[39] Natasha Freutel, “Hyperextended knee: Symptoms, treatment, recovery time “, 2016 [online], link: https://www.healthline.com/health/fitness-exercise/hyperextended-knee

[40] Ryan Goodrich, Live Science “Accelerometers: What they are & How they work “, 2013 [online], link: https://livescience.com/40102-accelerometers.html

[41] adafruit/Adafruit_BNO055, GitHub [online], link: https://github.com/adafruit/Adafruit_BNO055

[42] Arduino-libraries/SD, GitHub, [online], link: https://github.com/arduino-libraries/SD

[43] greiman/SdFat, GitHub, [online], link: https://github.com/greiman/SdFat

[44] sparkfun/SparkFun_BNO080_Arduino_Library, GitHub, [online], link: https://github.com/sparkfun/SparkFun_BNO080_Arduino_Library

[45] ABB, Robotics, IRB 120 datasheet [online], link: https://search-ext.abb.com/library/Download.aspx?DocumentID=ROBO149EN_D&LanguageCode=en&DocumentPartId=2&Action=Launch

[46] Mathworks, Evaluating goodness of fit[online], link:
https://se.mathworks.com/help/curvefit/evaluating-goodness-of-fit.html

[47] Giovanni Legnani, Medical Engineering & Physics, “A model of an electro-goniometer and its calibration for biomechanical applications “, 2001 [online], link:
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.876.281&rep=rep1&type=pdf

[48] Ramandeep Singh, Indian Institute of Technology Delhi, “Wearable Knee Joing Angle Measurement System based on Force Sensitive Resistors“, 2018 [online], link:
https://www.researchgate.net/publication/325702224_Wearable_knee_joint_angle_measurement_system_based_on_force_sensitive_resistors