MEMS SENSORS BASED ONLINE PHYSIOTHERAPY

**A PROJECT REPORT**

**Submitted by**

**PAVITHRAN PANDIYAN [REG NO: 1041210114]**

**T ANAND [REG NO : 1041210117]**

**KRISHNA VINAY [REG NO: 1041210131]**

Under the guidance of

**Mrs R. Vinolee**

(Assistant Professor, Department of Electronics and Communications Engineering)

***in partial fulfilment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

in

**ELECTRONICS AND COMMUNICATIONS ENGINEERING**

of

**FACULTY OF ENGINEERING AND TECHNOLOGY**



S.R.M. Nagar, Kattankulathur, Kancheepuram District

**MAY 2016**

**SRM UNIVERSITY**

(Under Section 3 of UGC Act, 1956)

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“ MEMS SENSOR BASED ONLINE PHYSIOTHERAPY”** is a bonafide work **“PAVITHRAN PANDIYAN [REG NO. 1041210114], T ANAND [REG NO. 1041210117], KRISHNA VINAY [REG NO. 1041210131]”** who carried out the work under my supervision. Certified further, that to the best of my knowledge the work reported herein that does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**SIGNATURE SIGNATURE**

Mrs. R. Vinolee Dr. T. Rama Rao

**GUIDE**  **HEAD OF DEPARTMENT**

Assistant Professor (O.G) Department of Electronics

Department of Electronics and Communication Enginnering

and Communication Engineering.

Signature of the Internal Examiner Signature of the External Examiner

**ABSTRACT**

MEMS inertial sensor based physical therapy application that guides patients to perform exercises and assists therapists to monitor as well as teach the specially designed exercises to patients. The application includes an automated exercise generator for therapists to record exercises, an automated posture recognition and tracking system to track and guide the patients while they perform the exercises and a visual feedback system for patients to correct the position and movement of their joints through direct and inverse kinematics approach. It enables continuous tracking of patients for pre-authored physiotherapy exercises.

This paper introduces a state-machine-based approach that tracks a patient's progress and provides continuous feedback indicating whether the patient is doing an exercise correctly or not. Patients seeking physiotherapy treatments are mostly aged and frail people who find it difficult to travel to hospitals and hence seek treatments by themselves.

The people seeking physiotherapy may be suffering from ailments as minor as a normal leg sprain to severe cases like cardio-respiratory dysfunction. MEMS sensors like MPU6050 which can be interfaced to our software suite that can recognize and record the exercises to be done by the patient. It can mimic the actions of patient subjected to, flexi-bands fastened around his joints to read his body movements and facilitates the system to understand the motion and hence the discrepancies between the optimal values prescribed by the physiotherapist and recorded observations of the patient respectively will be compared, checked for the regularity of the exercise performed, recovery pattern among different individuals for similar kinds of disorders would be studied.

**ACKNOWLEDGEMENTS**

We express our sincere gratitude and thanks to our Management for giving us a wonderful chance to show our attitude towards our career and gain knowledge. We extend our humble and sincere thanks to our **Head of the Department, Dr. T. Rama Rao**, Department of Electronics and communication for her moral support encouragement throughout the development of our project work. We express our heartfelt gratitude and thanks to our project coordinator, **Mr. S. Manikandaswamy, Assistant Professor (O.G)**, Electronics and Communication Department for providing moral support and motivation. We are indebted to our guide **Mrs. R. Vinolee, Assistant Professor (O.G)**, Department of Electronics and Communication for her valuable guidance, encouragement and timely help throughout our project work.

We extend our sincere thanks to all other faculty of our department for their kind co-operation and help in completing our project. Finally we would like to thank our parents and our friends whose kind encouragement, affection and moral support have helped us to complete the project successfully.

**TABLE OF CONTENTS**

**ABSTRACT**……………………………………………………………..................iii

**ACKNOWLEDGEMENT**……………………………………………....................iv

**LIST OF FIGURES**…………………………………………………......................viii

**LIST OF TABLES**……………………………………………………....................ix

**ABBREVIATIONS**……………………………………………………....................x

**1 Introduction**………………………………………………………...................1

* 1. Motivation…………………………………………………...............1
  2. Physiotherapy…………………………………………...............…...1
  3. MEMS Inertial Measurement Unit…………………….....................2
  4. Kinematics of Human arm..................................................................3
  5. Objectives............................................................................................5
     1. Integrate IMU Sensors with microcontrollers........................6
     2. Implement an algorithm to perceive motion of arm..............6

1. **Literature Review**..............................................................................................7
   1. Physiotherapy with e-Health services via cloud..................................7
   2. Physiotherapy using wireless body area network................................7
   3. Physiotherapy by means of NIST definition........................................8
   4. Understanding neuro engineering and rehabilitation techniques.........8
   5. Gesture recognition using accelerometer–microwave technology......9
   6. Gait and balance measurement using accelerometer...........................9
   7. Implementation of NFC technology for recovery of patients...............10
   8. Recovery tracking using accelerometer...............................................10
   9. Gesture acquisition using DTW...........................................................11
   10. Inertial measurement unit based gesture recognition..........................11
2. **Analysis of Existing system**................................................................................13
   1. Existing Physiotherapy system..............................................................13
   2. Comparison between existing and proposed system............................14
3. **Analysis of Proposed system**..............................................................................16
   1. System Model........................................................................................16
      1. Hardware Circuitry................................................................16
      2. Software Suite.......................................................................17
   2. Dynamic Time Warping Algorithm......................................................18
   3. Complimentary filter.............................................................................20
   4. Realistic constraints...............................................................................22
      1. Accuracy..............................................................................22
      2. Orientation..........................................................................22
      3. Processing time delay.........................................................22
      4. Arduino Memory Limitation..............................................22

4.5 Multidisciplinary tasks.............................................................................23

4.6 Engineering Standards..............................................................................23

5. **Results**....................................................................................................................24

5.1 Hardware Setup..........................................................................................24

5.2 ‘3-D’ sensor values.....................................................................................24

6. **Conclusion**...................................................................................................................27

6.1 Future Works..............................................................................................28

6.1.1 Hardware.......................................................................................28

6.1.2 Software........................................................................................28

**A – Arduino Board Development**.............................................................................29

A.1 Background................................................................................................29

A.2 Software.....................................................................................................30

A.3 Arduino Uno...............................................................................................31

A.3.1 Power............................................................................................32

A.3.2 Memory........................................................................................33

A.3.3 Input and Output...........................................................................33

A.3.4 Communication............................................................................34

A.3.5 Automatic Software Reset............................................................35

**B MEMS Inertial Sensor**........................................................................................36

**C Python**...................................................................................................................37

C.1 Python Programming....................................................................................37

C.2 VPython........................................................................................................37

C.3 pySerial.........................................................................................................38

**D Programming codes**..............................................................................................39

D.1 Arduino Codes..............................................................................................39

D.2 VPython Animation code.............................................................................43

D.3 GUI Code.....................................................................................................44

**REFERENCES**...........................................................................................................55

**LIST OF FIGURES**

* 1. Stable Platform IMU 3
  2. Inertial Navigation Algorithm 3
  3. Frames of individual joints linked together 4

3.1 Existing Physiotherapy System Based on Microsoft Kinect 13

4.1 Block Diagram showcasing data Retreivel operation 16

4.2 Working Methodology of DTW Algorithm 17

4.3 Graphical Representation of Comparison of two signals DTW Algorithm 18

4.4 Flowchart of DTW Algorithm Implemented 19

4.5 Block Diagram of First Order low pass and First order High Pass Filter 21

4.6 Complimentary filter Block Diagram 21

5.1 Hardware Setup Implemented in Human arm 24

5.2 Graphical plot of Acceleration vs time (x,y,z) 25

5.3 Graphical plot of Angular velocity vs time (x,y,z) 25

5.4 Graphical plot of Angular rotation vs time 25

5.5 Graphical representation of roll axis in one of the sensors 26

5.6 Animation of Limb that follows motion during exercise 26

A.1 Arduino Uno reference design 29

B.1 MPU 6050 36

**LIST OF TABLES**

* 1. Denavit- Hatenburg table 4
  2. Segment Lengths 5
  3. Angle Limits 5

3.1 Existing Physiotherapy vs Proposed Implementation 14

4.1 Multidisciplinary task involved in the project 23

4.2 Engineering Standards used in the Project 23

A.1 Arduino Uno Specifications 32

**ABBREVIATIONS**

MEMS Micro Electro-Mechanical Systems

IMU Inertial Measurement Unit

D-H Denavit-Hartenberg

WBAN Wireless Body Area Network

LAN Local Area Network

NFC Near Field Communication

I2C Inter-Integrated Circuit

SDA Serial Data

SCL Serial Clock

DTW Dynamic Time Warping

IDE Integrated Development Environment

PWM Pulse Width Modulation

DMP Digital Motion Processor

**CHAPTER 1**

**INTRODUCTION**

In the modern world the needs are growing so is the demand for developing novel techniques to transform the large distance barriers in our lives. This growing concern has to be addressed by making everyone self-sustainable. The modern era has witnessed a sudden ubiquitous growth in the bio medical engineering field which was neglected till a few decades back. The physically ailing members of our society need the assistance of others. Conventional researches in this field of motion sensing could be broadly classified as: data glove based and computer vision based. The proposed methodology of introducing high grade sensors would impart more precision to the various calculations.

* 1. **Motivation**

In this approach the frail patients can be easily be attended from home as the interfacing suite can accommodate auto scaling for multiple uploading with great ease. Over the past few decades many approaches for posture identification have been proposed, one such technique involves the determination of the recognition factor which often is dependent on the distance between the two points lying in the frame which further requires the prediction of path taken. These techniques have been implemented only to an extent of still and static documents and not completely accurate. As a dynamic picture conveys more information and is fairly challenging for the existing methodologies. In general conventional approaches takes an image and further breaks down it into several parts in a phase known as fragmentation. Hence the Direct and Inverse Kinematics approach was followed in this system, so as to provide the real-time analysis of each and every body parts.

**1.2 Physiotherapy**

Physiotherapy is a physical medicine and rehabilitation specialty that remediates impairments and promotes mobility, function, and quality of life through examination, diagnosis, prognosis, and physical intervention (therapy using mechanical force and movements). It is carried out by physiotherapists.

Physiotherapy involves the illnesses, or injuries that limit their abilities to move and perform functional activities as well as they would like in their daily lives. Physiotherapists use an individual's history and physical examination to arrive at a diagnosis and establish a management plan and, when necessary, incorporate the results of laboratory and imaging studies like X-rays, CT-scan, or MRI findings. Physiotherapy management commonly includes prescription of or assistance with specific exercises, manual therapy and manipulation, mechanical devices such as traction, education, physical agents which includes heat, cold, electricity, sound waves, radiation, rays, prescription of assistive devices, prostheses, orthoses and other interventions. Physiotherapy has many specialties including sports, neurology, wound care, EMG, cardiopulmonary, geriatrics, orthopaedic and paediatrics.

**1.3 MEMS Inertial Measurement Unit**

An inertial measurement unit (IMU) is an electronic device that measures and reports a body's specific force and angular rate using a combination of accelerometers and gyroscopes. Inertial measurement units (IMUs) typically contain three orthogonal rate-gyroscopes and three orthogonal accelerometers, measuring angular velocity and linear acceleration respectively. By processing signals from these devices it is possible to track the position and orientation of a device.

The inertial sensors are mounted on a platform which is isolated from any external rotational motion. In other words the platform is held in alignment with the global frame. This is achieved by mounting the platform using gimbals (frames) which allow the platform freedom in all three axes, as shown in Figure 1.1. The platform mounted gyroscopes detect any platform rotations. These signals are fed back to torque motors which rotate the gimbals in order to cancel out such rotations, hence keeping the platform aligned with the global frame. To track the orientation of the device the angles between adjacent gimbals can be read using angle pick-offs. To calculate the position of the device the signals from the platform mounted accelerometers are double integrated. Note that it is necessary to subtract acceleration due to gravity from the vertical channel before performing the integration.

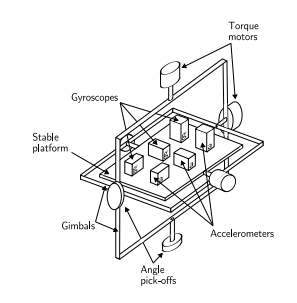


Figure 1.1: A stable platform IMU

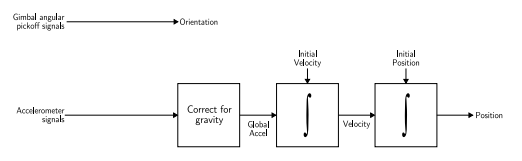


Figure 1.2: Inertial Navigation Algorithm

# **1.4 Kinematics of Human Arm**

Kinematics is known as the study of motion without considering the forces that create the motion. Direct Kinematics is the representation of the robot’s end-effector position and orientation through the geometries of robots i.e., joint and link parameters. Using Kinematics, the mathematical model is developed to compute the position and orientation of each fingertip (end-effector’s) based on the given human joint position. Each human joint is considered as a revolute joint. The homogenous transformation of the fingertip related to the base frame (arm) is formulated using Denavit-Hartenberg (D-H) method.

The human arm was modelled using the Direct Kinematics method. The problem was solved geometrically by attaching individual inertial frames to human arm joints with the reference frame being considered with its origin in the point situated at the middle of the scapular belt. . Then all the individual frames of joints are attached to links as shown in the Figure 1.

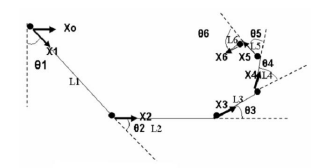


Fig 1.3: Frames of individual joints linked together.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **i** | **αi-1** | **ai-1** | **di** | **θi** |
| 1 | 0 | 0 | 0 | θ1 |
| 2 | 0 | L1 | 0 | θ2 |
| 3 | 0 | L2 | 0 | θ3 |
| 4 | 0 | L3 | 0 | θ4 |
| 5 | 0 | L4 | 0 | θ5 |
| 6 | 0 | L5 | 0 | θ6 |
| 7 | 0 | L6 | 0 | 0 |

TABLE 1.1: Denavit-Hartenberg table

Using the Denavit-Hartenberg table as shown in Table 1 and the general formula for determining the homogeneous transformation between each frame as shown in the below matrix (Fig 2),

(1.1)

a final overall transform is obtained by applying the set of transforms sequentially as shown in the below matrix.

(1.2)

We use the following segment lengths for study purpose based on average statistical data shown in table 2.

|  |  |  |  |
| --- | --- | --- | --- |
| **Subject** | **L1(Upper arm)** | **L2(Forearm)** | **L3(Hand)** |
| Adult Male | 0.315m | 0.287m | 0.105m |
| Adult Female | 0.272m | 0.252m | 0.091m |

Table 1.2: Segment Lengths

We use the following angle limits (degree same for both male and female) which is observed from table 3.

|  |  |  |
| --- | --- | --- |
| **Angle** | **Min** | **Max** |
| θ1 | -140 | 90 |
| θ2 | 0 | 145 |
| θ3 | -70 | 90 |

Table 1.3: Angle limits

We simulated Forward Pose Kinematics for the entire range of motion from the minimum to maximum angle on each joint simultaneously.

Given θ1= -135o, θ2= 90o, θ3=45o, the unique FPK solutions are:

Xmale = [xH yH θ] = [0.085 -0.426 0] (1.3)

Xfemale = [xH yH θ] = [0.077 -0.371 0] (1.4)

The above obtained are the results of the transform upon substituting the values for the usually observed limb length and the twist angle of it.

**1.5 Objectives**

Our objective is to make a standalone, low-cost online physiotherapy system which compares the motion of arm of the physiotherapist and the patient.

This would be achieved with the fulfilment of the following sub-objectives:

**1.5.1 Integrate IMU sensors with the microcontroller**

The Inertial measurement Unit(IMU) sensors are integrated with the microcontroller through I2C bus and data is transmitted and received serially.

**1.5.2 Implement an algorithm to perceive the motion of arm**

To implement inverse kinematics concepts and algorithms to perceive and record the motion of arm which can be extended to other body parts in future works. The algorithm should take in continuous values in real-time application and the motion of the arm should be recorded.

**1.5.3 Develop a software suite**

* + Graphical User Interface to facilitate ease of use.
  + Direct and Inverse Kinematics approaches to track the movement of human body parts.
  + Statistical analytics to track the regularity of therapy undertaken and patient’s performances and to recognize pattern of recovery among different patients for similar disorders.

**CHAPTER 2**

**LITERATURE REVIEW**

**2.1 Physiotherapy using eHealth services via Cloud**

Cloud Computing is an emerging commercial model which allows organizations to eliminate the need to maintain costly hardware, software and network infrastructures. There has been a significant advance in wearables and connected health monitoring devices constituting the so called Wireless Body Area Networks (WBANs). Patients or the medical healthcare provider subscribe to the HSP which provides its services on the Cloud. Peak loads may be introduced across different locations of the network affecting negatively the QoS. The Cloud Provider needs to optimize the propagation delay and processing time in order to respect the QoS requested by the HSP in case of violations. Auto scaling is the prominent technology introduced in this cloud version of WBANs. This auto-scaling

mechanism relies on vertical and horizontal provisioning with application-delay constraints.

**2.2 Physiotherapy using wireless Body Area Network using intelligent motion sensors**

Wearable health monitoring systems integrated into a telemedicine system in this technique of monitoring the sensing the motion sensors. In this method the various motion sensors connected to different positions of the body are controlled by the Zigbee protocol and further transferred to the Internet by LAN or GPRS. The caregiver emergency will be intimated using the personal server. Here a Telos wireless platform with intelligent signal processing

daughtercard ISPM is equipped with the microcontroller. The activity sensor is made of ISPM and Telos platform which has flat memory. The size and weight proves to be a pulling down factor for this technique.

Inclusion of continuous monitoring data into medical databases will allow integrated analysis of all data to optimize individualized care and provide knowledge discovery through integrated data mining.

**2.3 Physiotherapy by means of NIST definition in Cloud computing**

The various degrees of usages of Cloud computing is defined by NIST which is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets; but such standards and guidelines shall not apply to national security systems. It defines the five types of characteristics involved in cloud computing. It also explains the various service and deployment models.

**2.4 Understanding Neuro-Engineering and Rehabilitation Techniques**

This method complies to the standards laid by the international medical fraternity and also proposes a novel technique of cutting down the need of qualified human assistance for convalescing patient for their rehabilitative needs. The primary objective of this study was to compare time and motion measures during real life physical therapy with estimates of active time (i.e. the time during which a patient is active physically) obtained with a wireless body area network (WBAN) of 3D accelerometer modules positioned at the hip, wrist and ankle. Post-acute rehabilitation is a key component of the health care delivery system, yet we know little about the active ingredients of the rehabilitation process that produce the best outcomes. Methods more efficient than observation are needed to measure active time in rehabilitation. This technique can be applied to heart monitoring and other physical ailments it puts forth a strategy for observation. For which Kinematics has been suggested as a better alternative to estimate mobilization and active time in rehabilitation. Using the microcontrollers and an analog to digital converter the reading is taken. The variables are measured for this purpose which are measure of active time by TM observation and by WBAN acceleration and the finding reveal that the results are favouring the latter but only by using three modules.

**2.5 Gesture recognition using Accelerometer- uWave technology- (quantisation and time warping)**

uWave is an algorithm for interaction using a single three-axis accelerometer. It’s capable of user interaction and authentication. Gestures1 have recently become attractive for spontaneous interaction with consumer electronics and mobile devices in the context of pervasive computing. It uses gesture recognition

method based on a single accelerometer using dynamic time warping (DTW). The main functions of this algorithm is its continuous acceleration quantization, dynamic time warping (DTW), and template adaptation.

By which we can detect the minimum distance between the observed pattern and the desired pattern. It’s equipped with a Wii remote has built in three axis accelerometer .To the collected gestures the algorithm is employed, its functionality can be extended to light weight user authentication and 3D mobile application.

**2.6 Gait and balance measurements using accelerometer**

The central purpose of this experiment involves the gait and balance measurement, impairments in gait and balance are sometimes quite subtle and subject to individual interpretation. Accelerometers are an ideal choice for evaluating variability of movement and balance providing a non-invasive, portable method of measurement. The repeated patterns obtained with measures of acceleration contain information on the smoothness or variability of the walking pattern

(index of smoothness). The comparison of gait of adults and older people are studied. Using accelerometer

data a framework is proposed to standardise and define the events of the sit-stand-sit movement cycle.

The input data is observed using optical motion analysis and uses reflective sensors. Hence the gait measurements can give an insight into fallers and non fallers.

**2.7 Implementation of NFC technology for recovery of patients**

This system employs Near field communication and this is used for detecting Alzheimer,s disease affected patients. Using a computer to support daily

activities is not yet a reality nowadays. Alzheimer affected patients need intensive care and to track and serve these patients need. The different technologies for assisting these patients include Context Modelling through Ontology and Context Sensing and Phone-mediated natural interaction with NFC. It’s a short ranged communication technology that facilitates operation in active and passive mode which can be accessed by bringing the device close. This can be done by communication over the entire area at the centre is facilitated. This is achieved by a Bluetooth enabled NFC mobile phone. In order to keep control the patients’ health when they are at home or at the daycentres, it is necessary to monitor their vital signs, which are measurements of the basic functions of the body. To use a monitoring device, a person (the patient or assistant) should be able of just touching its NFC tag with the phone, in order to launch the mobile phone application.

**2.8 Recovery tracking using accelerometer feed image recognition technique**

Zolta´n PREKOPCSA´K describes a real-time hand gesture recognition system, which identifies relevant parts in the continuous sensor data stream, and classifies them to the most probable gesture. This research lies in the intersection of the technical and design perspectives.

It makes use of the built in Java environment of a smart phone and forwards the data to a computer. Only the segmented parts are forwarded to the next component. This is followed by segmentation and gesture recognition by markov models. To test the accuracy of the recognition, a dataset of gestures have been created. HMMs are able to filter movements that are not similar to predefined gestures.

**2.9 Gesture acquisition using DTW and excerpts on image processing**

The research by Iván Raso, Ramón Hervás, José Bravo elucidates a rehabilitation system based on new technological tendencies in the mobility and ubiquitous computing areas. As a pre-process procedure, raw data output by mobile device accelerometer is filtered, and then we use the technique called Dynamic Time Warping to train and recognize movements. The ubiquity of mobile devices has led to the emergence of personalized and adaptive services that are able to respond particular needs of each specific user. Accelerometry has become a powerful choice for evaluating variability of person’s movement using these kinds of sensor that provide a non-invasive method of measurement and have a successful accuracy. A phone with IMU is considered for this purpose. When a patient performs the exercise, it always starts at the same point approximately. By making the patient workout the different factors for measurement are determined. There are numerous methods employed for the pattern recognition and eventually the maximum DTW is obtained for the training purpose.

As soon as patients finish the exercise, the mobile device uses the last sample to recognize the end of the rehabilitation exercise.

**2.10 Inertial Measurement Unit based Gesture recognition**

This research describes an inertial gesture recognition framework composed of three parts. The first is a compact, wireless six-axis inertial measurement unit to fully capture three-dimensional motion. The second,

a gesture recognition algorithm, analyzes the data and categorizes it on an axis-by-axis basis as simple motions (straight line, twist, etc.) with magnitude and duration. The third allows an application designer to combine recognized gestures both concurrently and consecutively to create specific composite gestures can then be set to trigger output routines. IMU holds a number of advantages over other sensing technologies: they directly measure important parameters for human interaction and they can easily be embedded into mobile platforms.

It presents a solution in the form of a generalized framework for inertial gesture recognition. Its application is extended to expressive footwear which does much similar work. Since inertial sensors measure their own motion, the system will either have to be worn by the subjects or embedded in an object manipulated by them. For this reason atomic gesture recognition algorithm is used this is because of the human kinetic nature. While the activity detection algorithm is designed primarily to filter the data before it is passed on to the recognition system, other purposes can be served in annotating the data stream. It introduces constraints such that the system cannot track multi-dimensional gestures. This data is then analyzed with a windowed variance algorithm to find periods of activity, and a generalized gesture recognition algorithm is applied to those periods.

**CHAPTER 3**

**ANALYSIS OF EXISTING SYSTEM**

**3.1 Existing Physiotherapy System**

Existing physiotherapy systems operate on the principle of Digital Image Processing using the help of Microsoft Kinect motion sensor device. Microsoft Kinect is a motion-sensor device that allows users to interact with the given system through body movement and voice recognition. When paired with a brace worn by the user, program will deliver customized exercises submitted by the patient's physical therapist, as well as accurate correcting feedback and a physical therapy progress report.

The patients begin by meeting with their physical therapist for a diagnosis and rehabilitation plan. The therapist (or physical therapy assistant) would then lead the patient through the assigned workouts in front of the Kinect device in order to program the desired movements into the system. The patient would then follow the prompts from the program in order to complete the workout. As the patient executes their customized exercises, the program will measure the angle of completion (percent of exercise completed) for a given exercise and relay that information to the physical therapist. Through this progress report, the therapist will have access to quantitative data to determine the individual improvement as well as other significant therapy information that can be used for research and development.

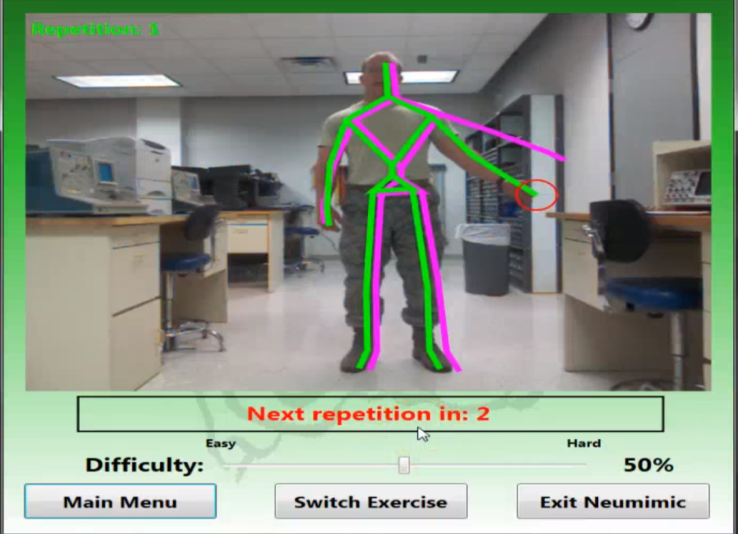


Figure 3.1: Existing physiotherapy system based on Microsoft Kinect.

Here the pink colour skeleton signifies the exercise performed and prescribed the physiotherapist. The green colour skeleton denotes the movement of the patient’s arm and the degree of closeness of the exercise performed is generated.

**3.2 Comparison between existing and proposed physiotherapy systems**

|  |  |  |
| --- | --- | --- |
|  | **Existing System** | **Proposed System** |
| **Operating Principle** | Digital Image Processing | Inverse Kinematics |
| **Software** | MATLAB | Python, VPython |
| **Processing Time** | 1s | 2s |
| **Cost** | Rs.12000 | Rs.1500 |
| **Portability** | No | Yes |

Table 3.1: Existing Physiotherapy versus proposed implementation

In table 3.1, we can see some common parameters which are being compared between the existing physiotherapy system and proposed physiotherapy system.

The existing system is a lot more complex one which requires a high computational capacity to run the software it uses whereas our proposed system needs very less computational capacity to run our software. The existing system uses a motion sensor camera in the form of Microsoft Kinect which captures the movement of our body parts whereas our system has sophisticated hardware components located in the area concerned i.e., limbs or arms and hence is more local towards the problem statement. This system with the help of Inertial Measurement Unit(IMU) tracks the movement of body parts and provides a more accurate result.

Also, our system has been made self-dependent and adaptable making it compatible and accessible from any system or computer and hence is portable, whereas the existing system needs specific softwares to run the Digital Image Processing tools and hence making it non-portable.

However, our system has lower computational capacity than the existing system and hence requires more processing time than the existing system. But considering the economic impact, the trade-off between cost and computational capacity is not a big issue as the difference in the processing time is not too high whereas the difference in cost of the system is high.

**CHAPTER 4**

**ANALYSIS OF PROPOSED SYSTEM**

**4.1 Design Methodology**

This section would broadly be classified into two parts each focussing on the hardware circuitry and the software suite implemented in the proposed project.

**4.1.1 Hardware Circuitry**

Online Physiotherapy can be achieved by using MEMS Inertial Measurement Units shortly abbreviated as MEMS in the figure 4.1. These MEMS Inertial devices are very small and compact devices having dimensions of 21x15.5x2.6mm hence can be attached to flex-bands. These flex-bands will be worn on the joint portions of the limbs or arms. The MEMS Inertial devices are interfaced with an ATMega 328 microcontroller through Inter-Integrated Circuit (I2C) interfacing protocol. The values of MEMS Inertial based sensors are passed to the ATMega328 microcontroller through Serial Data (SDA) and Serial Clock (SCL) links. The microcontroller passes the information to the software on the computer via Bluetooth transmission protocol as shown in figure 4.1. A HC-05 Bluetooth module is interfaced with ATMega 328 which facilitates transmission of data to the Bluetooth module attached with the computer. The Start/stop is a push button used to facilitate the user to start or stop recording values.

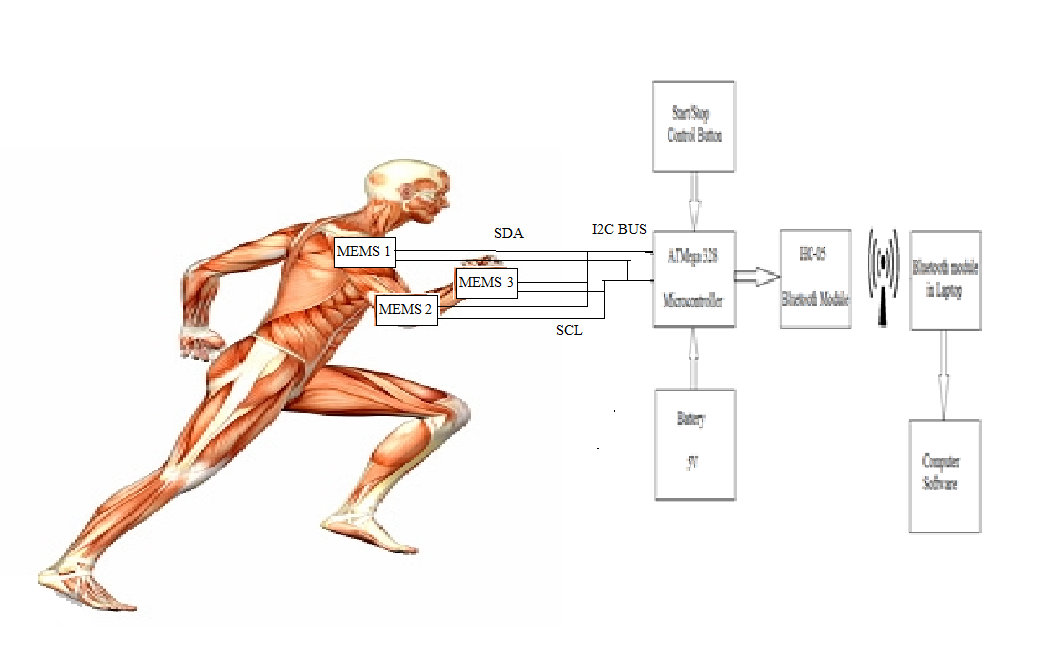


Figure 4.1: Block diagram showcasing data retrieval operation

**4.1.2 Software Suite**

A fully fledged Python environment Graphical User Interface is used in this system which is implemented within the software suite which provides the user with access to operate the system in an easier way apart from detecting the continuous changes in the signal and processing it further. The data acquired through ATMega 328 Microcontroller is read and serially input into the Python based GUI. The three axis data are acquired separately and their vector magnitude is found by calculating the root mean square of the data available at all the three axes. The software analyses the responses and drafts a report on patient’s performance. The performance of physiotherapist and patients’ limbs are then processed through direct and inverse kinematics approach. The software uses a reference model as prescribed by an expert physiotherapist for evaluation of the patient and compares them through Dynamic Time Warping Algorithm as detailed in the below sections. The software then compiles and records daily report of the patient. The above values are then used to perform a statistical analysis of the regularity of the exercise performed, recovery time taken, and the recovery pattern among different patients having similar kind of disorders. The flowchart of the algorithm is depicted in fig 5.2.

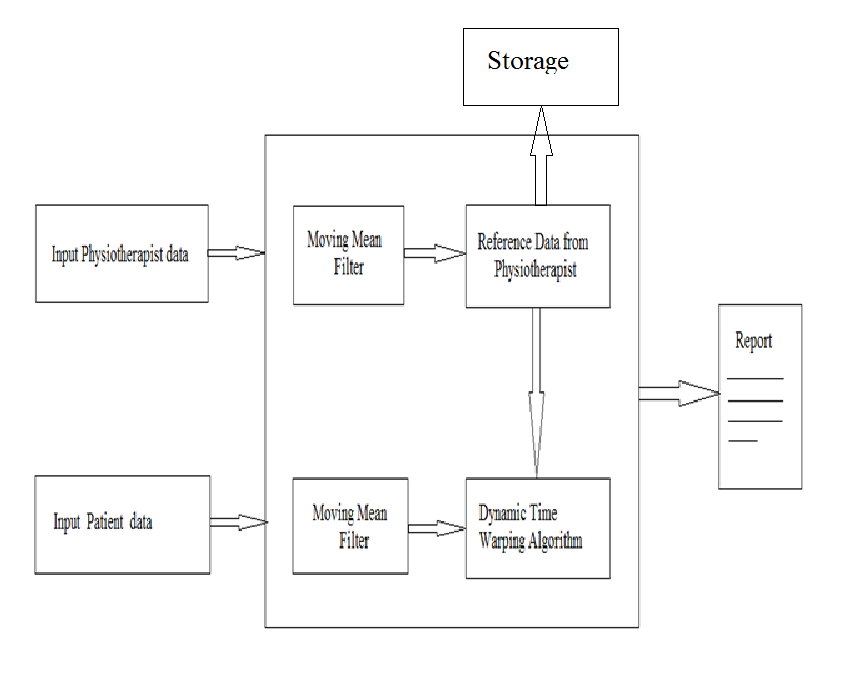


Figure 4.2: Working Methodology of Dynamic Time Warping Algorithm

**4.2 Dynamic Time Warping Algorithm**

In time series analysis, dynamic time warping (DTW) is an algorithm for measuring similarity between two temporal sequences which may vary in time or speed. Dynamic time warping (DTW) is a time series alignment algorithm developed originally for speech recognition. The graph generated shows the variation of amplitudes over the whole spectrum of samples as given in Fig 4.3.

Any data that can be passed on as a linear data can be analyzed by DTW such as audio, video and graphics data. It aims at aligning two sequences of feature vectors by warping the time axis iteratively until an optimal match (according to a suitable metrics) between the two sequences is found. Here, Signal 2 is the reference signal and Signal 1 is the real-time signal. DTW is very fast as it requires O(N2).

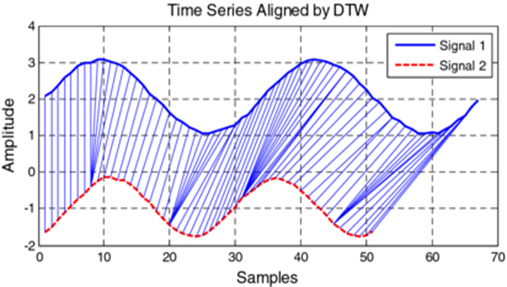


Figure 4.3: Graphical representation of comparison of two signals using Distance Time Warping Algorithm.

DTW is a method that calculates an optimal match between two given sequences (e.g. time series) with certain restrictions. The sequences are "warped" non-linearly in the time dimension to determine a measure of their similarity independent of certain non-linear variations in the time dimension. This sequence alignment method is often used in time series classification.

The exercise recorded by the physiotherapist is kept as a reference signal showcased as signal 2 in Fig 4, and a dynamic signal of patient’s values as he performs the exercise is delivered as an input signal showcased as signal 1 in Fig 4. The Dynamic Time Warping Algorithm measures the similarity between the two temporal sequences and the algorithm’s output is recorded and mapped to a percentage scale from 1-100 and recorded on a daily basis as exercise is been performed by the patient.

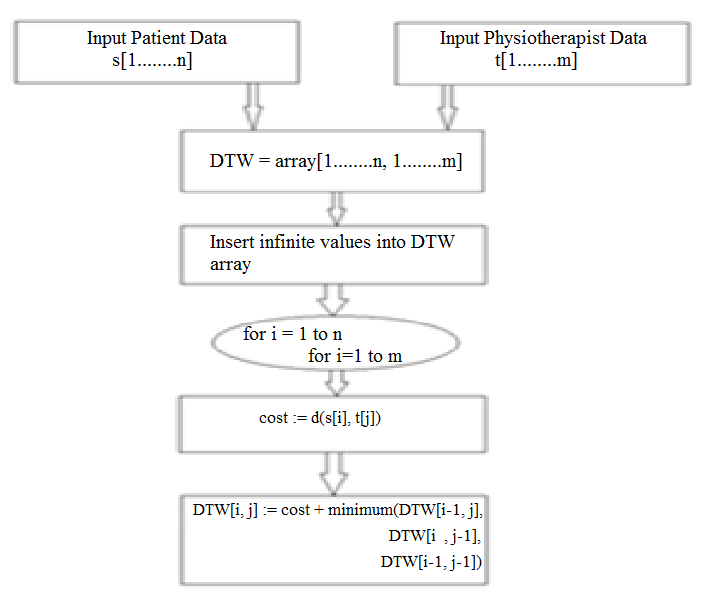


Figure 4.4: Flow-chart of Dynamic Time Warping algorithm implemented

Pseudo-code for Dynamic Time Warping Algorithm:

int DTWDistance(s: array [1..n], t: array [1..m]) {

DTW := array [0..n, 0..m]

for i := 1 to n

DTW[i, 0] := infinity

for i := 1 to m

DTW[0, i] := infinity

DTW[0, 0] := 0

for i := 1 to n

for j := 1 to m

cost := d(s[i], t[j])

DTW[i, j] := cost + minimum(DTW[i-1, j ],

DTW[i , j-1],

DTW[i-1, j-1])

return DTW[n, m]

}

Figure 4.4 and the above pseudo code explain the algorithmic implementation of Dynamic Time Warping algorithm. The pseudo code is implemented with the help of a function call with two parameters of type array s and t. As a first step, both the arrays are filled with values of zeroes. Then, cost function is used to obtain the distance between the two linear arrays. Then, the DTW array is filled in with the values about the distance between the two linear arrays. Hence, the final DTW array shows us the degree of closeness between the input physiotherapist data and the patient’s data.

The percentage of closeness is evaluated in the following manner:

**Formula:**

Root mean square of the values gives the degree of closeness of

X-axis= ∑(Acc\_x\_patient – Acc\_X\_physiotherapist)2 / (Ang\_X\_Physiotherapist) [1]

Y-axis= ∑(Acc\_y\_patient – Acc\_Y\_physiotherapist)2 / (Ang\_Y\_Physiotherapist) [2]

Z-axis= ∑(Acc\_z\_patient – Acc\_Z\_physiotherapist)2 / (Ang\_Z\_Physiotherapist) [3]

**Standard Deviation, SD= (X+Y+Z)/3**

Hence, the final degree of closeness = SD

**4.3 Complimentary Filter**

Complimentary filter is a digital algorithm that serves to “blend” or “fuse” similar or redundant data from different sensors to achieve a robust estimate of a single state variable.

The complementary filter is a frequency domain filter. In its strictest sense, the definition of a complementary filter refers to the use of two or more transfer functions, which are mathematical complements of one another. Thus, if the data from one sensor is operated on by G(s), then the data from the other sensor is operated on by I-G(s), and the sum of the transfer functions is I, the identity matrix. In the case of a one-dimensional filter as will be described in this paper, the identity matrix reduces to the scalar number one. In a typical two-input system, one input will provide information with high frequency noise, and is thus low-pass filtered. The other input provides information with low frequency noise, and is high-pass filtered. If the low-pass and high-pass filters are mathematical complements, then the output of the filter is the complete reconstruction of the variable being sensed, minus the noise associated with the sensors. A block diagram illustrating this process with “perfect” 1st -order low-pass and high-pass filters is shown below:

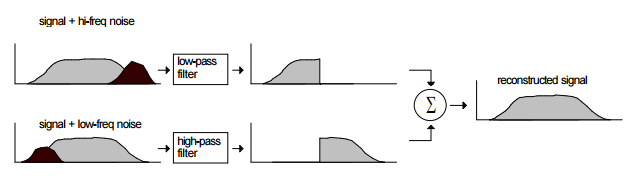


Figure 4.5: Block diagram with 1st order Low pass and High pass filters

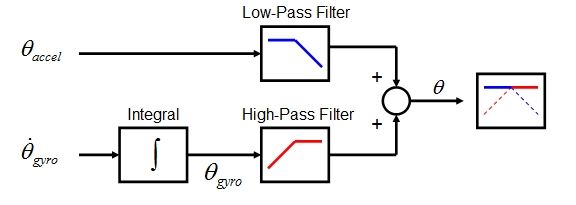


Figure 4.6: Complimentary Filter Block Diagram

Figure 4.6 depicts the use of Complimentary filter function in our system. The angular signals from accelerometer is passed to the Low-pass filter filters high frequency signals (such as the accelerometer in the case of vibration) and the angular signals from gyroscopes are integrated and passed through High-pass filter that filter low frequency signals (such as the drift of the gyroscope) which are later summed up to obtain accurate angular orientation value.

**4.4 Realistic Constraints**

Every system has its own realistic constraints. The following are the realistic constraints faced by our system:

**4.4.1 Accuracy**

Sincere effort has been taken to produce maximum accuracy out of the system. Various kinds of filters such as Low-pass filters, High-pass filters and Complimentary filters have been used in tandem to bring out the best result. Regardless of these there is still a possibility of few noises to creep into the system due to Electromagnetic interference and vibrations in real-time system. Hence a tested accuracy of 95% has been proved.

**4.4.2 Orientation**

Since, the system involves usage of inertial sensors such as accelerometers and gyroscopes, the orientation of the system must be maintained at all times to obtain best result out of the system. Usage of the system in other orientations than the prescribed would lead to errors and inefficiency.

**4.4.3 Processing time delay**

While the Arduino Uno allows for a compact setup for the system to run on, it has a very low processing compared to a traditional microcontroller. This leads to considerable delays in time taken for a sensor value to travel from sensor to the processing unit of the microcontroller. This constraint could be solved with development of low-cost high speed microcontrollers in the future.

**4.4.4 Arduino memory limitations**

The Arduino core code contains a nice little round robin data buffer where we can keep throwing data at it and the Arduino code will read the data and process it in order.

However, this data buffer is by default only 64 bytes in size. This value is hard coded in the Arduino core source code and applies to all Arduino boards, even those with a vast amount of RAM available. Increasing this value to allow for the Arduino to receive more data carries the risk of corrupting the bootloader and is beyond the scope of the project.

As a result of this limitation, values detected by the MEMS Inertial sensor cannot be sent to the Arduino at the same time. Hence, there is some delay in the order of a few seconds between each 64 block of values.

**4.5 Multidisciplinary Tasks**

Our system involves usage and implementation of technologies from various fields as listed below.

|  |  |  |
| --- | --- | --- |
| **Department** | **Utilized for** | **Remarks** |
| Micro Electro-Mechanical Systems (MEMS) | Minimizing the size of Inertial sensor used in the system | Tracking and recording movements of arms and limbs |
| Dynamic Time Warping Algorithm | Comparing two linear streams of data | Used to calculate the degree of closeness |
| Computational/IT | Arduino C | To code ATMega328 in Arduino platform |
| Software | Graphical User Interface Development | Provides ease of use for the user |
| Desktop Publications | Report | Presentation and review |

Table 4.1: Multidisciplinary tasks involved in the project

**4.6 Engineering Standards**

|  |  |
| --- | --- |
| **Standard** | **Description** |
| STK 500 | ATMega 328 Hardware programmer |
| IEEE 802.3 | Ethernet |
| RS 232 | Serial Communication |

Table 4.2 Engineering Standards used in project

**CHAPTER 5**

**RESULTS**

**5.1 Hardware Setup**

Figure 5.1 shows the completed hardware setup being implemented in a human arm. Two MEMS sensors are attached in flex-bands and worn in the arm. They connected by wires and data is passed serially to the ATMega328 which passes on the data through an USB port to the Software suite in the computer.

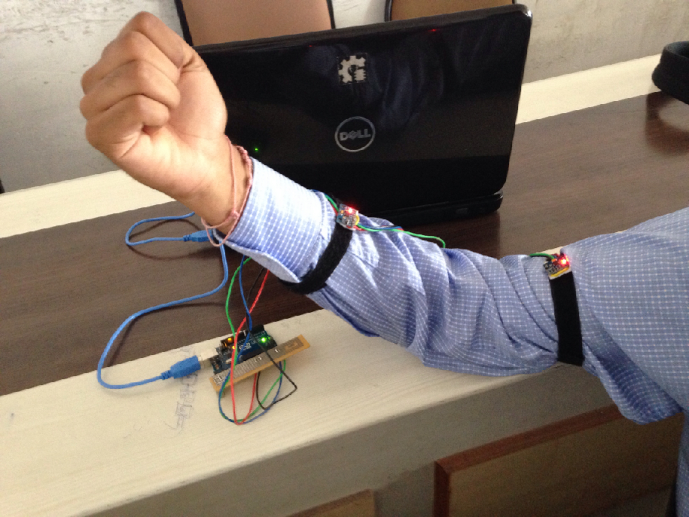


Figure 5.1 Hardware setup implemented in human arm

**5.2 Three-Dimensional Sensor Values**

In this paper we could evaluate performance of the patient by tracking the acceleration, angular velocity and angular positions of the movement of the limb as shown in Fig 5.2, 5.3 and 5.4 respectively. We will use both the accelerometer and gyroscope data for the same purpose: obtaining the angular position of the object. The gyroscope can do this by integrating the angular velocity over time. To obtain the angular position with the accelerometer, we are going to determine the position of the gravity vector (g-force) which is always visible on the accelerometer.

On the short term, we use the data from the gyroscope, because it is very precise and not susceptible to external forces. On the long term, we use the data from the accelerometer, as it does not drift. The complementary filter combines the data from accelerometer and gyroscope to provide a stable result as shown in fig 5.4.

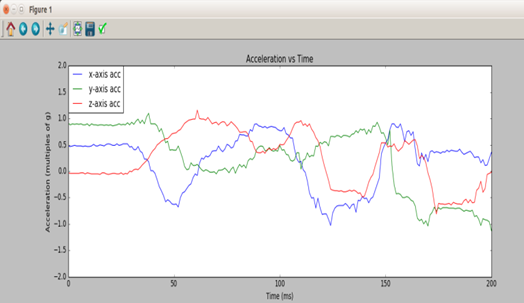


Fig 5.2: Graphical plot of Acceleration vs Time for x, y, z axis.



Fig 5.3: Graphical plot of Angular velocity vs Time for x, y, z axis.

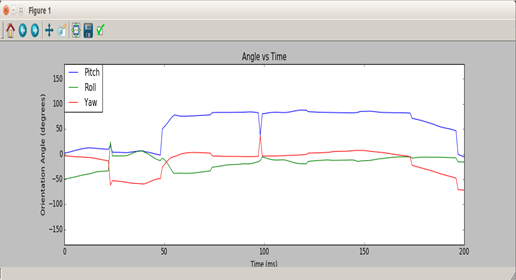


Fig 5.4: Graphical plot of Angular rotation vs Time for roll, pitch and yaw using complimentary filter.

The motion is then compared by Dynamic Time Warping algorithm by comparing the patient’s data with the prescribed physiotherapist’s exercise obtained. An iterative comparison is made with respect to the physiotherapist data for each discrete time sample using dynamic time warping algorithm. The graph shown in Fig 5.5 contains the deviations in roll axis of one of the MEMS sensors.

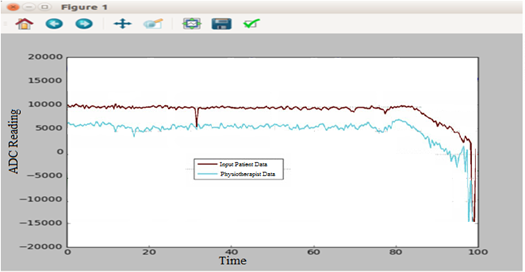


Fig 5.5: Graphical Representation of roll axis in one of the sensors.

The same concept was applied to pitch and yaw axes in other sensors and the degree of closeness of each MEMS sensor (joint) was obtained individually. This is also depicted in the form of an animation as shown in fig 5.6.

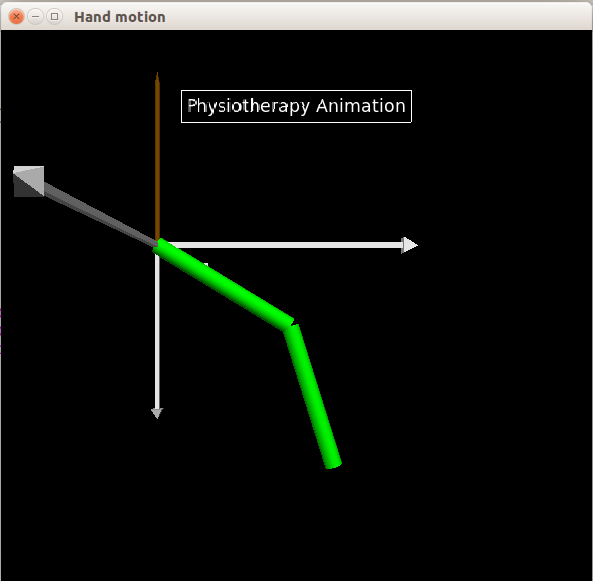


Fig 12: Animation of a limb that follows the motion during the exercise.

**CHAPTER 6**

**CONCLUSION**

This project here presented the basic concept and the synergy of methodology for the development of an efficient methodology capable of tracking the movements made by the test subject and results were drawn from comparisons between the optimal and obtained readings. In addition, the methodology has potential for commercial and scientific applications such as patient information retrieval, recovery, etc. This paper produced successful results considering one of the human limb (i.e. the arm) and hence can be extended to other limbs and body parts using the same technology. This paper also attempts to provide a new dimension on the inertial sensor readings’ processing. In particular, the methodology presented here deals with helping the patient convalescing and rate of recovery. Statistical and mathematical analysis has shown that the device is capable to track the patient’s progress and pattern of recovery among different patients with similar disorders.

The applicability of gesture recognition for physiotherapy is extensive. This project is aimed at the frail and weak building an application suite for dynamically uploading data. It will give quality service to millions of people who are in need of physiotherapy. There are a number of contributions in this paper firstly the power requirement and production cost are seemingly less. It has been proposed to use a long standing battery for optimization in this field. This requires no other additional accessories and only minimal hardware such as bracelet to the arms and legs. Incorporation to the web further enhances the value of this project manifold as cloud seems to conquer the IT industry use of virtual machines can cut down the hardware and makes it rather easy to operate. Physiotherapy industry is million dollar industry this project was designed eyeing the low income groups. The microcontroller can connect to the PC terminal via the software suite tailored as per user needs. In order to do this we need only bands or bracelet kind of devices for closely monitoring the patient without violating the terms of standard file transfer protocol. It has a robust device operation with greater leeway to incorporate changes.

**6.1 Future Works**

**6.1.1 Hardware**

As of now, the hardware setup is aimed at performing operations in an arm only. This could be extended to all body parts with improvement in the hardware setup according to specific customised needs of various body parts. Also, the system’s computational capacity can be increased with development of new low cost and high processing speed microcontrollers in the future.

As mentioned before, the inbuilt buffer memory size is set at the default of 64 bytes in the Arduino. This required us to send data in blocks which increases the time delay. An alternative more effective solution for this problem has to be researched upon. Options include setting a higher buffer size (though it risk damaging the microcontroller) or finding a more capable replacement microcontroller.

**6.1.2 Software**

Various improvements in aesthetics, flow control, stability, security and privacy protection of the software suite can be improved. As of now, the software setup is aimed at performing operations in an arm only. This could be extended to all body parts with improvement in the software setup according to specific customised needs of various body parts.

**APPENDIX A**

**ARDUINO BOARD DEVELOPMENT**

**A.1 Background**

Arduino is a very popular and easy to use programmable board. It consists of a simple hardware platform and a free source code editor with an easy one-click compile/upload feature. It has been used to make robots, home automation gadgets, automotive projects, for sensing and controlling lights, motors, locks and servos, sound and video, interactive objects like animated sculptures, toys and games, radio links and just about anything else you can dream up. It can even be a web server and connect projects to the Internet.

An Arduino board consists of an Atmel 8-bit AVR microcontroller with complementary components that facilitate programming and incorporation into other circuits. An important aspect of the Arduino is its standard connectors, which lets users connect the CPU board to a variety of interchangeable add-on modules known as shields. Some shields communicate with the Arduino board directly over various pins, but many shields are individually addressable via an I2C serial bus, so many shields can be stacked and used in parallel.

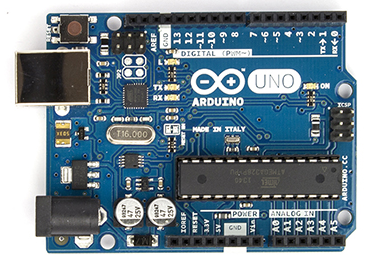


Figure A.1: Arduino UNO Reference Design

Official Arduinos have used the megaAVR series of chips, specifically the ATmega8, ATmega168, ATmega328, ATmega1280, and ATmega2560. Most boards include a 5 volt linear regulator and a 16 MHz crystal oscillator (or ceramic resonator in some variants), although some designs such as the LilyPad run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions.

An Arduino’s microcontroller is also pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory, compared with other devices that typically need an external programmer. This makes using an Arduino more straightforward by allowing the use of an ordinary computer as the programmer. At a conceptual level, when using the Arduino software stack, all boards are programmed over an RS-232 serial connection, but the way this is implemented varies by hardware version. Serial Arduino boards contain a level shifter circuit to convert between RS-232-level and TTL-level signals. Current Arduino boards are programmed via USB, implemented using USB-to-serial adapter chips such as the FTDI FT232. Some variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, Bluetooth or other methods.

**A.2 Software**

The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and derives from the IDE for the Processing programming language and the Wiring projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capableof compiling and uploading programs to the board with a single click. A program or code written for Arduino is called a "sketch" [13]. Arduino programs are written in C or C++. The Arduino IDE comes with a software library called "Wiring" from the original Wiring project, which makes many common input/output operations much easier. Users only need define two functions to make an executable cyclic executive program:

• setup(): a function run once at the start of a program that can initialize settings.

• loop(): a function called repeatedly until the board powers off.

A typical first program for a microcontroller simply blinks an LED on and off. In the Arduino environment, the user might write a program like this:

*#define LED\_PIN 13*

*void setup() {*

*pinMode(LED\_PIN, OUTPUT); // Enable pin 13 for digital output }*

*void loop() {*

*digitalWrite(LED\_PIN, HIGH); // Turn on the LED*

*delay(1000); // Wait one second (1000 milliseconds)*

*digitalWrite(LED\_PIN, LOW); // Turn off the LED*

*delay(1000); // Wait one second*

*}*

It is a feature of most Arduino boards that they have an LED and load resistor connected between pin 13 and ground; a convenient feature for many simple tests. The previous code would not be seen by a standard C++ compiler as a valid program, so when the user clicks the "Upload to I/O board" button in the IDE, a copy of the code is written to a temporary file with an extra include header at the top and a very simple main() function at the bottom, to make it a valid C++ program. The Arduino IDE uses the GNU toolchain and AVR Libc to compile programs, and uses avrdude to upload programs to the board. As the Arduino platform uses Atmel microcontrollers, Atmel’s development environment, AVR Studio or the newer Atmel Studio, may also be used to develop software for the Arduino.

**A.3 Arduino Uno**

The Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.

|  |  |
| --- | --- |
| Microcontroller | [ATmega328P](http://www.atmel.com/Images/doc8161.pdf) |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limit) | 6-20V |
| Digital I/O Pins | 14 (of which 6 provide PWM output) |
| PWM Digital I/O Pins | 6 |
| Analog Input Pins | 6 |
| DC Current per I/O Pin | 20 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (ATmega328P) of which 0.5 KB used by bootloader |
| SRAM | 2 KB (ATmega328P) |
| EEPROM | 1 KB (ATmega328P) |
| Clock Speed | 16 MHz |
| Length | 68.6 mm |
| Width | 53.4 mm |
| Weight | 25 g |

Table A.1 : Arduino Uno specifications

**A.3.1 Power**

The Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

* Vin. The input voltage to the Uno board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
* 5V.This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
* 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
* GND. Ground pins.
* IOREF. This pin on the Uno board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

**A.3.2 Memory**

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

**A.3.3 Input and Output**

Each of the 14 digital pins on the Uno can be used as an input or output, using [pinMode()](https://www.arduino.cc/en/Reference/PinMode), [digitalWrite()](https://www.arduino.cc/en/Reference/DigitalWrite), and [digitalRead()](https://www.arduino.cc/en/Reference/DigitalRead) functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

* Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
* External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
* PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
* SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
* LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it’s off.
* TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, abelled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.  
There are a couple of other pins on the board:

* AREF. Reference voltage for the analog inputs. Used with analogReference().
* Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

**A.3.4 Communication**

The Uno has a number of facilities for communicating with a computer, another Uno board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

**A.3.5 Automatic (Software) Reset**

Rather than requiring a physical press of the reset button before an upload, the Uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino Software (IDE) uses this capability to allow you to upload code by simply pressing the upload button in the interface toolbar. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data. The Uno board contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line.

**APPENDIX B**

**MEMS INERTIAL MEASUREMENT UNIT – MPU 6050**

The InvenSense MPU-6050 sensor contains a MEMS accelerometer and a MEMS gyro in a single chip. It is very accurate, as it contains 16-bits analog to digital conversion hardware for each channel. Therefor it captures the x, y, and z channel at the same time. The sensor uses the I2C-bus to interface with the Arduino. The MPU-6050 is not expensive, especially given the fact that it combines both an accelerometer and a gyro.

The sensor has a "Digital Motion Processor" (DMP), also called a "Digital Motion Processing Unit". This DMP can be programmed with firmware and is able to do complex calculations with the sensor values.The DMP can do fast calculations directy on the chip. This reduces the load for the microcontroller (like the Arduino). The DMP is even able to do calculations with the sensor values of another chip, for example a magnetometer connected to the second (sub)-I2C-bus.

In our project we use two sensors, For which,the pin "AD0" selects between I2C address 0x68 and 0x69. That makes it possible to have two of these sensors in a project. Most breakout boards have a pullup or pulldown resistor to make AD0 default low or high. Connect AD0 to GND or 3.3V for the other I2C address.



Figure B.1: MPU 6050

**APPENDIX C**

**PYTHON**

**C.1 Python Programming**

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language.Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library.

Python code can be packaged into stand-alone executable programs for some of the most popular operating systems, allowing the distribution of Python-based software for use on those environments without requiring the installation of a Python interpreter.

Python has a large standard library, commonly cited as one of Python's greatest strengths, providing tools suited to many tasks. This is deliberate and has been described as a "batteries included"[28] Python philosophy. For Internet-facing applications, a large number of standard formats and protocols (such as MIME and HTTP) are supported. Modules for creating graphical user interfaces, connecting to relational databases, pseudorandom number generators, arithmetic with arbitrary precision decimals, manipulating regular expressions, and doing unit testing are also included.

**C.2 VPython**

VPython is the Python programming language plus a 3D graphics module called Visual. VPython allows users to create objects such as spheres and cones in 3D space and displays these objects in a window. This makes it easy to create simple visualizations, allowing programmers to focus more on the computational aspect of their programs. The simplicity of VPython has made it a tool for the illustration of simple physics, especially in the educational environment.

VPython is a simple rendering tool for 3D objects and graphs. Its main use has been in education, but it has also been used in commercial or research settings.

**C.3 PySerial**

pySerial is a Python API module to access the serial port. pySerial provides a uniform API across multiple operating systems, including Windows, Linux, and BSD

[PySerial](http://www.python.org/pypi/pyserial) is a library which provides support for serial connections ("RS-232") over a variety of different devices: old-style serial ports, Bluetooth dongles, infra-red ports, and so on. It also supports remote serial ports via RFC 2217

Bluetooth connections can also be handled using the socket module, provided that support for Bluetooth is present in the underlying operating system and has been compiled into the Python distribution being used.

**APPENDIX D**

**Programming Codes**

**D.1 Arduino Uno Code**

*#include "I2Cdev.h"*

*#include "SPI.h"*

*#include "MPU6050\_6Axis\_MotionApps20.h"*

*#if I2CDEV\_IMPLEMENTATION == I2CDEV\_ARDUINO\_WIRE*

*#include "Wire.h"*

*#endif*

*MPU6050 mpu1;*

*MPU6050 mpu2(0x69);*

*#define OUTPUT\_READABLE\_YAWPITCHROLL*

*#define LED\_PIN 13*

*bool blinkState = false;*

*bool dmpReady1 = false;*

*uint8\_t mpuIntStatus1;*

*uint8\_t devStatus;*

*uint16\_t packetSize1;*

*uint16\_t packetSize2;*

*uint16\_t fifoCount1;*

*uint8\_t fifoBuffer1[64];*

*bool dmpReady2 = false;*

*uint8\_t mpuIntStatus2;*

*uint16\_t fifoCount2;*

*uint8\_t fifoBuffer2[64];*

*int zero\_detect1,zero\_detect2;*

*Quaternion q;*

*VectorInt16 aa;*

*VectorInt16 aaReal;*

*VectorInt16 aaWorld;*

*VectorFloat gravity;*

*float euler[3];*

*float ypr[3];*

*float ypr2[3];*

*uint8\_t teapotPacket[14] = { '$', 0x02, 0,0, 0,0, 0,0, 0,0, 0x00, 0x00, '\r', '\n' };*

*bool TurnOnZI1 = false;*

*volatile bool mpuInterrupt = false; // indicates whether MPU interrupt pin has gone high*

*void dmpDataReady() {*

*mpuInterrupt = true;*

*}*

*void setup() {*

*#if I2CDEV\_IMPLEMENTATION == I2CDEV\_ARDUINO\_WIRE*

*Wire.begin();*

*TWBR = 24;*

*#elif I2CDEV\_IMPLEMENTATION == I2CDEV\_BUILTIN\_FASTWIRE*

*Fastwire::setup(400, true);*

*#endif*

*Serial.begin(38400);*

*mpu1.initialize();*

*mpu2.initialize();*

*while (Serial.available() && Serial.read());*

*while (Serial.available() && Serial.read());*

*devStatus = mpu1.dmpInitialize();*

*devStatus = mpu2.dmpInitialize();*

*if (devStatus == 0) {*

*mpu1.setDMPEnabled(true);*

*attachInterrupt(0, dmpDataReady, RISING);*

*mpuIntStatus1 = mpu1.getIntStatus();*

*dmpReady1 = true;*

*packetSize1 = mpu1.dmpGetFIFOPacketSize();*

*mpu2.setDMPEnabled(true);*

*attachInterrupt(0, dmpDataReady, RISING);*

*mpuIntStatus2 = mpu2.getIntStatus();*

*dmpReady2 = true;*

*packetSize2 = mpu2.dmpGetFIFOPacketSize();*

*} else {*

*pinMode(LED\_PIN, OUTPUT);*

*}*

*void loop() {*

*if (!dmpReady1) return;*

*mpuInterrupt = false;*

*mpuIntStatus1 = mpu1.getIntStatus();*

*fifoCount1 = mpu1.getFIFOCount();*

*fifoCount2 = mpu2.getFIFOCount();*

*mpu1.resetFIFO();*

*} else if (mpuIntStatus1 & 0x02) {*

*while (fifoCount1 < packetSize1) fifoCount1 = mpu1.getFIFOCount();*

*mpu1.getFIFOBytes(fifoBuffer1, packetSize1);*

*fifoCount1 -= packetSize1;*

*#ifdef OUTPUT\_READABLE\_YAWPITCHROLL*

*mpu1.dmpGetQuaternion(&q, fifoBuffer1);*

*mpu1.dmpGetGravity(&gravity, &q);*

*mpu1.dmpGetYawPitchRoll(ypr, &q, &gravity);*

*#endif*

*blinkState = !blinkState;*

*digitalWrite(LED\_PIN, blinkState);*

*}*

*if (!dmpReady2) return;*

*mpuInterrupt = false;*

*mpuIntStatus2 = mpu2.getIntStatus();*

*fifoCount2 = mpu2.getFIFOCount();*

*if ((mpuIntStatus2 & 0x10) || fifoCount2 == 1024) {*

*mpu2.resetFIFO();*

*} else if (mpuIntStatus2 & 0x02) {*

*while (fifoCount2 < packetSize2) fifoCount2 = mpu2.getFIFOCount();*

*mpu2.getFIFOBytes(fifoBuffer2, packetSize2);*

*fifoCount2 -= packetSize2;*

*#ifdef OUTPUT\_READABLE\_YAWPITCHROLL*

*mpu2.dmpGetQuaternion(&q, fifoBuffer2);*

*mpu2.dmpGetGravity(&gravity, &q);*

*mpu2.dmpGetYawPitchRoll(ypr2, &q, &gravity);*

*Serial.print(ypr[0] \* 180/M\_PI);*

*Serial.print("\t");*

*Serial.print(ypr[1] \* 180/M\_PI);*

*Serial.print("\t");*

*Serial.print(ypr[2] \* 180/M\_PI);*

*Serial.print("\t");*

*Serial.print(ypr2[0] \* 180/M\_PI);*

*Serial.print("\t");*

*Serial.print(ypr2[1] \* 180/M\_PI);*

*Serial.print("\t");*

*Serial.println(ypr2[2] \* 180/M\_PI);*

*// Serial.print("\t");*

*// Serial.println(zero\_detect2);*

*#endif*

*// blink LED to indicate activity*

*blinkState = !blinkState;*

*digitalWrite(LED\_PIN, blinkState);*

*}*

*}*

**D.2 VPython Animation Code**

*from pylab import \**

*from visual import \**

*import math*

*import serial*

*theta1 = math.radians(-60)*

*phi1 = math.radians(10)*

*theta2 =math.radians(-45)*

*phi2 = math.radians(60)*

*theta\_dot = 0.0*

*phi\_dot = 0.1*

*g = 9.8*

*l = 1.0*

*m = 0.3*

*time = 0.0*

*dt = 0.0001*

*ser=serial.Serial('/dev/ttyACM0',38400)*

*ball1 = sphere(pos=vector(l\*sin(theta1)\*cos(theta2),-l\*cos(theta1)\*cos(theta2),l\*sin(theta2)), radius=0.12, color=color.blue)*

*ball2 = sphere(pos=vector(l\*sin(theta1)\*cos(theta2) + l\*sin(phi1)\*cos(phi2),-l\*cos(theta1)\*cos(theta2) - l\*cos(phi2),l\*sin(theta2) + l\*sin(phi2)), radius=0.12, color=color.blue)*

*arm1 = cylinder(pos=(0,0,0), axis=(l\*sin(theta1)\*cos(theta2),-l\*cos(theta1)\*cos(theta2),l\*sin(theta2)), radius=.03, color=color.cyan)*

*arm2 = cylinder(pos=(0,0,0), axis=(l\*sin(theta1)\*cos(theta2) + l\*cos(phi2), -l\*cos(theta1)\*cos(theta2) - l\*cos(phi2), l\*sin(theta2) + l\*sin(phi2)), radius=.03, color=color.cyan)*

*in\_x\_plane=arrow(pos=(0,0,0), axis=(2,0,0), shaftwidth=0.1, headwidth=0.1, color=color.orange, opacity=0.3)*

*in\_y\_plane=arrow(pos=(0,0,0), axis=(0,-2,0), shaftwidth=0.1, headwidth=0.1, color=color.orange, opacity=0.3)*

*in\_z\_plane=arrow(pos=(0,0,0), axis=(0,0,2), shaftwidth=0.1, headwidth=0.1, color=color.orange, opacity=0.3)*

*nub = sphere(pos=vector(0,0,0), radius=0.05, color=color.white)*

*scene.userspin = 1*

*scene.autoscale = 0*

*scene.range = (2.05,2.05,2.05)*

*timestep=0.01*

*while 1:*

*try:*

*s=ser.readline().split()*

*k=map(float,s)*

*print k*

*theta1=math.radians(k[4])*

*theta2=math.radians(k[5])*

*phi2 = math.radians(k[1])*

*ball1.pos.x = l\*sin(theta1)\*cos(theta2)*

*ball1.pos.y = -l\*cos(theta1)\*cos(theta2)*

*ball1.pos.z = l\*sin(theta2)*

*arm1.axis = (ball1.pos.x, ball1.pos.y, ball1.pos.z)*

*ball2.pos.x = l\*sin(theta1)\*cos(theta2) + l\*cos(phi2)*

*ball2.pos.y = -l\*cos(theta1)\*cos(theta2) - l\*cos(phi2)*

*ball2.pos.z = l\*sin(theta2) + l\*sin(phi2)*

*arm2.pos = (l\*sin(theta1)\*cos(theta2), -l\*cos(theta1)\*cos(theta2), l\*sin(theta2))*

*arm2.axis = (ball2.pos.x - ball1.pos.x, ball2.pos.y - ball1.pos.y, ball2.pos.z-ball1.pos.z)*

*except:*

*pass*

**D.3 Graphical User Interface Code**

*import wx*

*import os*

*import serial*

*import glob*

*'''ser=serial.Serial('/dev/ttyACM0',38400)*

*s=ser.readline().split()*

*k=map(float,s)*

*print k*

*'''*

*import time*

*flag=0*

*patient\_name=''*

*patient\_file=''*

*reference\_file='doctor.txt'*

*class Welcomepage(wx.Frame):*

*def \_\_init\_\_(self,parent,id):*

*wx.Frame.\_\_init\_\_(self,parent,id,'MESEBONPHY',size=(800,600))*

*panel=wx.Panel(self)*

*custom=wx.StaticText(panel, -1, " MEMS SENSOR BASED ONLINE PHYSIOTHERAPY ", (200,20),(360,-1), wx.ALIGN\_CENTER)*

*custom.SetForegroundColour('blue')*

*img1 = wx.Image('physio1.jpg', wx.BITMAP\_TYPE\_ANY)*

*sb1 = wx.StaticBitmap(panel, -1, wx.BitmapFromImage(img1),pos=(200,100))*

*button=wx.Button(panel,label='Therapist',pos=(120,400),size=(200,40))*

*butto=wx.Button(panel,label='Patient',pos=(500,400),size=(200,40))*

*self.Bind(wx.EVT\_BUTTON,self.Onbutton, button)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutto,butto)*

*self.Bind(wx.EVT\_CLOSE,self.closewindow)*

*def Onbutton(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Therapistpage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def Onbutto(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Patientpage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def closewindow(self,event):*

*self.Destroy()*

*class Therapistpage(wx.Frame):*

*def \_\_init\_\_(self,parent,id):*

*wx.Frame.\_\_init\_\_(self,parent,id,'MESEBONPHY',size=(800,600))*

*panel=wx.Panel(self)*

*custom=wx.StaticText(panel, -1, " MEMS SENSOR BASED ONLINE PHYSIOTHERAPY ", (200,20),(360,-1), wx.ALIGN\_CENTER)*

*custo=wx.StaticText(panel, -1, " Welcome Doctor ", (200,40),(360,-1), wx.ALIGN\_CENTER)*

*custom.SetForegroundColour('blue')*

*info=wx.StaticText(panel, -1, " Please select the patients file and doctor file before checking patients performance ", (100,550),(360,-1), wx.ALIGN\_CENTER)*

*info.SetForegroundColour('red')*

*custo.SetForegroundColour('blue')*

*img1 = wx.Image('therapy.png', wx.BITMAP\_TYPE\_ANY)*

*sb1 = wx.StaticBitmap(panel, -1, wx.BitmapFromImage(img1),pos=(200,100))*

*button=wx.Button(panel,label='Record Exercise',pos=(70,500),size=(200,40))*

*butto=wx.Button(panel,label='Exercise Files',pos=(500,500),size=(200,40))*

*butt=wx.Button(panel,label='Patient Performance',pos=(285,500),size=(200,40))*

*but=wx.Button(panel,label='Back',pos=(20,20),size=(80,40))*

*self.Bind(wx.EVT\_BUTTON,self.Onbutton, button)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutto,butto)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutt, butt)*

*self.Bind(wx.EVT\_BUTTON,self.Onbut,but)*

*self.Bind(wx.EVT\_CLOSE,self.closewindow)*

*def Onbutton(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=TherapistRecordingpage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def Onbutto(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Exercisepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def Onbutt(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Performancepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def Onbut(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Welcomepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def closewindow(self,event):*

*self.Destroy()*

*class Performancepage(wx.Frame):*

*def \_\_init\_\_(self,parent,id):*

*wx.Frame.\_\_init\_\_(self,parent,id,'MESEBONPHY',size=(800,600))*

*panel=wx.Panel(self)*

*global patient\_name*

*print patient\_name*

*custom=wx.StaticText(panel, -1, " ONLINE PHYSIOTHERAPY EVALUATION", (200,20),(360,-1), wx.ALIGN\_CENTER)*

*name=wx.StaticText(panel, -1, "Name : "+ patient\_name, (50,100),(360,-1), wx.ALIGN\_CENTER)*

*custom.SetForegroundColour('blue')*

*info=wx.StaticText(panel, -1, " ", (100,550),(360,-1), wx.ALIGN\_CENTER)*

*info.SetForegroundColour('red')*

*img1 = wx.Image('hands.jpg', wx.BITMAP\_TYPE\_ANY)*

*sb1 = wx.StaticBitmap(panel, -1, wx.BitmapFromImage(img1),pos=(500,70))*

*but=wx.Button(panel,label='Back',pos=(20,20),size=(80,40))*

*anim=wx.Button(panel,label='View animation',pos=(50,450),size=(80,40))*

*self.Bind(wx.EVT\_BUTTON,self.Onanim,anim)*

*self.Bind(wx.EVT\_BUTTON,self.Onbut,but)*

*self.Bind(wx.EVT\_CLOSE,self.closewindow)*

*f=open(patient\_name+'.txt','r')*

*g=open('doctor.txt','r')*

*fr=f.readline()*

*gr=g.readline()*

*shroll=0*

*shpitch=0*

*elroll=0*

*elpitch=0*

*sh\_mean\_deviation=0*

*el\_mean\_deviation=0*

*el\_roll\_variance=0*

*sh\_pitch\_variance=0*

*sh\_roll\_variance=0*

*el\_pitch\_variance=0*

*count=0*

*while fr !='' or gr !='':*

*fr=f.readline()*

*gr=g.readline()*

*frs=fr.split()*

*grs=gr.split()*

*if len(grs) ==6 and len(frs) ==6:*

*shroll= float(frs[1])-float(grs[1])*

*shpitch=float(frs[2])-float(grs[2])*

*elroll=float(frs[4])-float(grs[4])*

*elpitch=float(frs[5])-float(grs[5])*

*if abs(elroll)>30 or abs(shroll>30) or abs(shpitch)>30 or abs(elpitch)>30:*

*continue*

*else:*

*sh\_roll\_variance+=abs(shroll)*

*el\_roll\_variance+=abs(elroll)*

*sh\_pitch\_variance+=abs(shpitch)*

*el\_pitch\_variance+=abs(elpitch)*

*count+=1*

*print shroll,shpitch,elroll,elpitch,count*

*shpd=wx.StaticText(panel, -1, "shoulder pitch deviation : "+str(sh\_pitch\_variance/1000), (50,150),(360,-1), wx.ALIGN\_CENTER)*

*shrd=wx.StaticText(panel, -1, "shoulder roll deviation : "+str(sh\_roll\_variance/1000), (50,200),(360,-1), wx.ALIGN\_CENTER)*

*elpd=wx.StaticText(panel, -1, "elbow pitch deviation : "+str(el\_pitch\_variance/1000), (50,250),(360,-1), wx.ALIGN\_CENTER)*

*elrd=wx.StaticText(panel, -1, "elbow roll deivation : "+str(el\_roll\_variance/1000), (50,300),(360,-1), wx.ALIGN\_CENTER)*

*pcls=wx.StaticText(panel, -1, "no.of iterations : "+ "4", (50,350),(360,-1), wx.ALIGN\_CENTER)*

*timetaken=wx.StaticText(panel, -1, "Time taken : "+ "10", (50,400),(360,-1), wx.ALIGN\_CENTER)*

*print sh\_roll\_variance/count,sh\_pitch\_variance/count,el\_roll\_variance/count,el\_pitch\_variance/count*

*def Onanim(self,event):*

*global patient\_name*

*print patient\_name*

*self.Destroy()*

*os.system('python newm.py '+patient\_name)*

*def Onbut(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Therapistpage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def closewindow(self,event):*

*self.Destroy()*

*class Exercisepage(wx.Frame):*

*def \_\_init\_\_(self,parent,id):*

*wx.Frame.\_\_init\_\_(self,parent,id,'MESEBONPHY',size=(800,600))*

*self.panel=wx.Panel(self)*

*i=0*

*self.custom=wx.StaticText(self.panel, -1, " MEMS SENSOR BASED ONLINE PHYSIOTHERAPY ", (200,20),(360,-1), wx.ALIGN\_CENTER)*

*self.custo=wx.StaticText(self.panel, -1, " Welcome Doctor, Here are the patient files ", (200,40),(360,-1), wx.ALIGN\_CENTER)*

*self.dir=wx.StaticText(self.panel, -1, " Directory: /home/pavithran/pywk/... ", (200,80),(360,-1), wx.ALIGN\_CENTER)*

*self.custom.SetForegroundColour('blue')*

*self.custo.SetForegroundColour('red')*

*self.a=glob.glob("/home/pavithran/pywk/\*.txt")*

*self.lc = wx.ListCtrl(self.panel, -1, style=wx.LC\_REPORT,size=(293,len(self.a)\*30),pos=(200,100))*

*self.lc.InsertColumn(0, 'Patient Name')*

*self.lc.InsertColumn(1, 'Text file')*

*self.lc.SetColumnWidth(0, 140)*

*self.lc.SetColumnWidth(1, 153)*

*self.button=wx.Button(self.panel,label='Main Menu',pos=(100,450),size=(120,40))*

*self.butto=wx.Button(self.panel,label='Delete File',pos=(300,450),size=(120,40))*

*self.butt=wx.Button(self.panel,label='Select for report',pos=(500,450),size=(120,40))*

*self.but=wx.Button(self.panel,label='Back',pos=(20,20),size=(80,40))*

*for x in self.a:*

*x=x.split("/")*

*self.lc.InsertStringItem(i,x[4])*

*self.lc.SetStringItem(i, 1, x[4])*

*i=i+1*

*self.Bind(wx.EVT\_BUTTON,self.Onbutton, self.button)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutto,self.butto)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutt, self.butt)*

*self.Bind(wx.EVT\_BUTTON,self.Onbut,self.but)*

*self.Bind(wx.EVT\_CLOSE,self.closewindow)*

*def Onbutton(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Welcomepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def Onbutto(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Exercisepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def Onbutt(self,event):*

*global patient\_name*

*index = self.lc.GetFocusedItem()*

*print index*

*k= self.a[index].split('/')*

*n=k[4].split('.')*

*patient\_name=n[0]*

*self.indexlabel=wx.StaticText(self.panel, -1, k[4] + " was selected ", (200,480),(360,-1), wx.ALIGN\_CENTER)*

*def Onbut(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Therapistpage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def closewindow(self,event):*

*self.Destroy()*

*class Patientpage(wx.Frame):*

*def \_\_init\_\_(self,parent,id):*

*wx.Frame.\_\_init\_\_(self,parent,id,'MESEBONPHY',size=(800,600))*

*panel=wx.Panel(self)*

*global patient\_name*

*box=wx.TextEntryDialog(None, "Username ", "MESEBONPHY", "Enter the your Name")*

*if box.ShowModal()==wx.ID\_OK:*

*answer=box.GetValue()*

*patient\_name=answer*

*custom=wx.StaticText(panel, -1, " MEMS SENSOR BASED ONLINE PHYSIOTHERAPY ", (200,20),(360,-1), wx.ALIGN\_CENTER)*

*warn=wx.StaticText(panel, -1, " Please connect the device before continuing... ", (50,550),(360,-1), wx.ALIGN\_CENTER)*

*custo=wx.StaticText(panel, -1, " Welcome " + answer, (200,40),(360,-1), wx.ALIGN\_CENTER)*

*custom.SetForegroundColour('blue')*

*custo.SetForegroundColour('blue')*

*warn.SetForegroundColour('red')*

*img1 = wx.Image('physio.jpg', wx.BITMAP\_TYPE\_ANY)*

*sb1 = wx.StaticBitmap(panel, -1, wx.BitmapFromImage(img1),pos=(50,75))*

*button=wx.Button(panel,label='Main Menu',pos=(500,450),size=(120,40))*

*exe1=wx.Button(panel,label='Arms',pos=(550,100),size=(120,40))*

*exe2=wx.Button(panel,label='Legs',pos=(550,150),size=(120,40))*

*exe3=wx.Button(panel,label='Wrist',pos=(550,200),size=(120,40))*

*exe4=wx.Button(panel,label='Ankle',pos=(550,250),size=(120,40))*

*exe5=wx.Button(panel,label='Hip',pos=(550,300),size=(120,40))*

*exe6=wx.Button(panel,label='neck',pos=(550,350),size=(120,40))*

*but=wx.Button(panel,label='Back',pos=(20,450),size=(80,40))*

*self.Bind(wx.EVT\_BUTTON,self.Onbutton, button)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutto,exe1)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutto,exe2)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutto,exe3)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutto,exe4)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutto,exe5)*

*self.Bind(wx.EVT\_BUTTON,self.Onbutto,exe6)*

*self.Bind(wx.EVT\_BUTTON,self.Onbut,but)*

*self.Bind(wx.EVT\_CLOSE,self.closewindow)*

*def Onbutton(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Welcomepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def Onbutto(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=PatientRecordingpage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def Onbut(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Welcomepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def closewindow(self,event):*

*self.Destroy()*

*class PatientRecordingpage(wx.Frame):*

*def \_\_init\_\_(self,parent,id):*

*wx.Frame.\_\_init\_\_(self,parent,id,'MESEBONPHY',size=(800,600))*

*panel=wx.Panel(self)*

*global patient\_name*

*print patient\_name*

*custom=wx.StaticText(panel, -1, " MEMS SENSOR BASED ONLINE PHYSIOTHERAPY ", (200,20),(360,-1), wx.ALIGN\_CENTER)*

*custo=wx.StaticText(panel, -1, " Welcome " +patient\_name+" , Record your exercise ", (200,40),(360,-1), wx.ALIGN\_CENTER)*

*custom.SetForegroundColour('blue')*

*custo.SetForegroundColour('red')*

*img1 = wx.Image('arm.jpg', wx.BITMAP\_TYPE\_ANY)*

*sb1 = wx.StaticBitmap(panel, -1, wx.BitmapFromImage(img1),pos=(50,75))*

*img2 = wx.Image('should.jpg', wx.BITMAP\_TYPE\_ANY)*

*sb2 = wx.StaticBitmap(panel, -1, wx.BitmapFromImage(img2),pos=(50,275))*

*button=wx.Button(panel,label='Main Menu',pos=(100,500),size=(120,40))*

*start\_rec=wx.Button(panel,label='New Recording',pos=(550,500),size=(120,40))*

*but=wx.Button(panel,label='Back',pos=(20,20),size=(80,40))*

*self.Bind(wx.EVT\_BUTTON,self.Onbutton, button)*

*self.Bind(wx.EVT\_KEY\_DOWN,self.onKeyPress,start\_rec)*

*self.Bind(wx.EVT\_BUTTON,self.record, start\_rec)*

*self.Bind(wx.EVT\_BUTTON,self.Onbut,but)*

*self.Bind(wx.EVT\_CLOSE,self.closewindow)*

*def onKeyPress(self, event):*

*keycode = event.GetKeyCode()*

*print keycode*

*if keycode == wx.WXK\_SPACE:*

*print "you pressed the spacebar!"*

*event.Skip()*

*def Onbutton(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Welcomepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def record(self,event):*

*global flag*

*i=100*

*t=time.time()*

*s=time.time()*

*text\_file=open(patient\_name +'.txt','w')*

*while s-t <10:*

*try:*

*x=ser.readline()*

*print(x)*

*print t-s*

*text\_file.write(x)*

*if x=="\n":*

*text\_file.seek(0)*

*text\_file.truncate()*

*text\_file.flush()*

*s=time.time()*

*except:*

*pass*

*i=i-1*

*text\_file.close()*

*def Onbut(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Welcomepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def closewindow(self,event):*

*self.Destroy()*

*class TherapistRecordingpage(wx.Frame):*

*def \_\_init\_\_(self,parent,id):*

*wx.Frame.\_\_init\_\_(self,parent,id,'MESEBONPHY',size=(800,600))*

*panel=wx.Panel(self)*

*custom=wx.StaticText(panel, -1, " MEMS SENSOR BASED ONLINE PHYSIOTHERAPY ", (200,20),(360,-1), wx.ALIGN\_CENTER)*

*custo=wx.StaticText(panel, -1, " Welcome Doctor , Record your exercise ", (200,40),(360,-1), wx.ALIGN\_CENTER)*

*custom.SetForegroundColour('blue')*

*custo.SetForegroundColour('red')*

*img1 = wx.Image('arm.jpg', wx.BITMAP\_TYPE\_ANY)*

*sb1 = wx.StaticBitmap(panel, -1, wx.BitmapFromImage(img1),pos=(50,75))*

*img2 = wx.Image('should.jpg', wx.BITMAP\_TYPE\_ANY)*

*sb2 = wx.StaticBitmap(panel, -1, wx.BitmapFromImage(img2),pos=(50,275))*

*button=wx.Button(panel,label='Main Menu',pos=(100,500),size=(120,40))*

*start\_rec=wx.Button(panel,label='New Recording',pos=(550,500),size=(120,40))*

*but=wx.Button(panel,label='Back',pos=(20,20),size=(80,40))*

*self.Bind(wx.EVT\_BUTTON,self.Onbutton, button)*

*self.Bind(wx.EVT\_KEY\_DOWN,self.onKeyPress,start\_rec)*

*self.Bind(wx.EVT\_BUTTON,self.record, start\_rec)*

*#self.Bind(wx.EVT\_BUTTON,self.stop, stop\_rec)*

*self.Bind(wx.EVT\_BUTTON,self.Onbut,but)*

*self.Bind(wx.EVT\_CLOSE,self.closewindow)*

*def onKeyPress(self, event):*

*keycode = event.GetKeyCode()*

*print keycode*

*if keycode == wx.WXK\_SPACE:*

*print "you pressed the spacebar!"*

*event.Skip()*

*def Onbutton(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Welcomepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def record(self,event):*

*global flag*

*i=100*

*t=time.time()*

*s=time.time()*

*text\_file=open('doctor.txt','w')*

*while s-t <10:*

*try:*

*x=ser.readline()*

*print(x)*

*print t-s*

*text\_file.write(x)*

*if x=="\n":*

*text\_file.seek(0)*

*text\_file.truncate()*

*text\_file.flush()*

*s=time.time()*

*except:*

*pass*

*i=i-1*

*text\_file.close()*

*def Onbut(self,event):*

*self.Destroy()*

*app=wx.PySimpleApp()*

*frame=Therapistpage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

*def closewindow(self,event):*

*self.Destroy()*

*if \_\_name\_\_=='\_\_main\_\_':*

*app=wx.PySimpleApp()*

*frame=Welcomepage(parent=None,id=-1)*

*frame.Show()*

*app.MainLoop()*

**REFERENCES**

[1] K. M. Culhane, M. OConnor, D. Lyons, and G. M.Lyons, “Accelerometers in rehabilitation medicine for older adults,” *Age and Ageing, vol. 20, Oct. 2005, pp. 556–560.*

[2] S. Choquette, M. Hamel, and P Boissy. “Accelerometer- based wireless body area network to estimate intensity of therapy in postacute rehabilitation,” *Journal of Neuro* *Engineering and Rehabilitation vol. 5(20), 2008*.

[3] Jiyayang Liu, Lin Zhong, Jehan Wickramsuriya, Venu Vasudevan Department of Electrical Computer Engineering, Rice University, “uWave: Accelerometer-based personalized gesture recognition and its applications ISSN: 1574-1192, Elsevier Publishing Company*.*

[4] Elie Rachkidi, El Hadi Cherkaoui, Mustapha Aitidir and Nazim Agoulmine COSMO, IBISC Laboratory University France, “Towards Efficient Automatic Scaling and Adaptive cost-optimized eHealth Services in Cloud”, *2008 20th IEEE International Conference on* *Tools with Artificial Intelligence.*

[5] J. E and all., “A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation.” *J Neuroeng Rehabil., vol. 1,no.6, 2005.*

[6] P. Mell and T. Grance, “The nist definition of cloud computing,” *NIST Special Publication 800-145,* *September 2011.*

[7] Ivan Raso, Ramon Hervas, Jose Bravo Technologies and Information System Department Castilla – La Mancha University ,Spain m-physio: “Personalized Accelerometer Based Physical Rehabilitation Platform, UBICOMM 2010”: *The Fourth International Conference on Mobile Ubiquitous* *Computing, System, services and Technologies.*

[8] J. Bravo, D. Lopez de Ipiña, C. Fuentes, R. Hervás,R. Peña, M. Vergara and G. Casero. “Enabling NFC technology for supporting chronic diseases: A proposal for alzheimer caregivers,” *LNCS 5355.Springer Publishing Company.*

[9] Z. Prekopcsk, “Accelerometer based real-time gesture Recognition ” Proc. *International Student Conference* *on Electrical Engineering, Prague, Czech Republic.*

[10] K. Agarwal, Y. X. Guo and B. Salam, "Wearable AMC Backed Near-Endfire Antenna for On-Body Communications on Latex Substrate," in *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 6, no. 3, pp. 346-358, March 2016.

[11] C. C. Y. Poon, Yuan-Ting Zhang and Shu-Di Bao, "A novel biometrics method to secure wireless body area sensor networks for telemedicine and m-health," in *IEEE Communications Magazine*, vol. 44, no. 4, pp. 73-81, April 2006.

[12] Parak, Jakub, et al. "Rehabilitation of patients using accelerometer: First experiment." Lékař a technika 2 (2012): 65-68.

[13] Merrett, Geoff V., et al. "Energy managed reporting for wireless sensor networks." Sensors and Actuators A: Physical 142.1 (2008): 379-389.

[14] Benbasat, Ari Y., and Joseph A. Paradiso. "An inertial measurement framework for gesture recognition and applications." Gesture and Sign Language in Human-Computer Interaction. Springer Berlin Heidelberg, 2001. 9-20.

[15] Clark, J.H. (1976). Designing surfaces in 3D. Communications of the ACM, 19(8). p. 454-460.

[16] Cao, Xiang, and Ravin Balakrishnan. "VisionWand: interaction techniques for large displays using a passive wand tracked in 3D." Proceedings of the 16th annual ACM symposium on User interface software and technology. ACM, 2003.