DOI:10.15774/PPKE.ITK.2017.002

Pázmány Péter Catholic University


Doctoral Thesis

# Kinematic measurement and analysis of human arm movements

*Author:*
Bence József Borbély

*Thesis Advisors:*
Prof. Dr. Péter Szolgay, D.Sc.
Dr. József Laczkó, Ph.D

*A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy*

*in the*

Roska Tamás Doctoral School of Sciences and Technology
Faculty of Information Technology and Bionics

Budapest, 2017

DOI:10.15774/PPKE.ITK.2017.002

*"Hope is not the conviction that something will turn out well but the certainty that something makes sense, regardless of how it turns out."*

Vaclav Havel

DOI:10.15774/PPKE.ITK.2017.002

# *Abstract*

Quantitative measurement and analysis of human motion is a key concept in understanding processes of our movement system. High precision measurement devices have advanced research activity in movement rehabilitation, performance analysis of athletes and the general understanding of the motor system during the last decades by making objective movement pattern comparison possible. This advancement was further accelerated by model-based analysis approaches that enabled explicit characterization of the studied movement patterns.

From behavioral aspects, manual and visual target tracking represents an important part of human movements and show strong predictive behavior. Studies investigating predictive manual tracking so far focused on the explanation of finger acceleration as a function of the 2D-tracking error and on the relation between the 3D-tracking error and path curvature and spatial depth but did not consider the control of shoulder, elbow and wrist joints. In the first part of the thesis an experimental setup and procedure is presented to investigate how motor synergies (involving the aforementioned joints) differ between predictive and non-predictive movements. During the analysis, motor synergies are evaluated by applying the uncontrolled manifold method to the joint angle variance during 2D tracking of a target on a graphics tablet, where the 2D pen position is used as the hypothetical task variable by the method. It is investigated whether the synergy index – defined by variance ratios affecting and being irrelevant for the task variable – drops during predictive, internally driven tracking movements compared to visually, externally driven tracking movements.

In the second part of the thesis the development of a custom wearable measurement device for arm movements is presented. The prototype incorporates inertial sensors for movement recording to overcome issues accompanying measurements with line of sight methods and enables evaluation and analysis of various sensor calibration, filtering and sensor fusion algorithms in a fully customizable manner. In the last part, a novel kinematic algorithm is introduced that utilizes orientation information of arm segments (directly measurable with inertial sensors) to perform joint angle reconstruction in real-time.

DOI:10.15774/PPKE.ITK.2017.002

# *Acknowledgements*

First of all, I would like to thank my scientific advisors *Péter Szolgay* and *József Laczkó* for their guidance, patience and valuable support during my studies.

I am very grateful to *Tamás Roska* and *Árpád Csurgay* for the inspiring discussions and their unbroken enthusiasm and encouragement.

I acknowledge the collaboration with *József Takács*, *Gábor Fazekas* and *Györgyi Stefanik* during the first year of my studies.

I thank my former and current colleagues *Dóra, Zsolt, Endre, Norbi, Zoli, János, Csaba, István, András, Balázs, Antal, Vamsi, Dani, Máté, Ádám, Zsolt, Gábor, Tamás, Miklós, Kálmán, András* among others for the chats at lunch, their advices and for discussions about my ideas.

I would like to acknowledge the financial support from the Pázmány Péter Catholic University, Faculty of Information Technology and Bionics through the following grants: TÁMOP-4.2.1.B-11/2/KMR-2011-0002, TÁMOP-4.2.2/B-10/1-2010-0014, KAP-1.1-14/030 and KAP15-055-1.1-ITK.

I thank *Katinka Tivadarné Vida* for her always kind help and patience to make the administrative side of life easier, the work of the Dean's Office, the Financial Department and the IT Department.

I spent over a year at the Research Training Group 1091 "Orientation and motion in space" of the German Research Foundation (DFG) being part of the University Hospital of Ludwig-Maximilians-Universität München where I met a lot of great minds. I would like to thank *Andreas Straube* for his supervision and generosity and *Thomas Eggert* for his guidance and tireless help during my stay, I learned a lot from both of them. Furthermore, I am grateful to *Maj-Catherine Botheroyd* for her help in administrative and personal manners and all of the former colleagues for making this period a great experience.

I am very grateful to my mother and father and to my whole family who always believed in me and supported me in all possible ways.

Finally, I am especially grateful to my wife *Margaréta* for all of her love, patience, support and encouragement that gave me the strength to go on even in the hardest periods of this journey.

DOI:10.15774/PPKE.ITK.2017.002

# Contents

DOI:10.15774/PPKE.ITK.2017.002

DOI:10.15774/PPKE.ITK.2017.002

DOI:10.15774/PPKE.ITK.2017.002

# List of Figures

DOI:10.15774/PPKE.ITK.2017.002

# List of Tables

DOI:10.15774/PPKE.ITK.2017.002

# Abbreviations

| | |
|---|---|
| **fMRI** | **f**unctional **M**agnetic **R**esonance **I**maging |
| **2D** | **2**-**D**imensional |
| **3D** | **3**-**D**imensional |
| **UCM** | **U**n**C**ontrolled **M**anifold method |
| **DoF** | **D**egrees **o**f **F**reedom |
| **REX** | **R**eal-time **EX**perimentation system |
| **TRU** | $^{2}$**U**npredictable $^{1}$**TR**ajectory |
| **TR**$n$ | **TR**ajectory $n$, where $n \in \{1, 2, 3, 4\}$ |
| **ANOVA** | **AN**analysis **O**f **VA**riances |
| **HSD** | **H**onest **S**ignificant **D**ifference |
| **LQG** | **L**inear **Q**uadratic **G**aussian |
| **EMG** | **E**lectro**M**yo**G**ram |
| **MCU** | **M**icro**C**ontroller **U**nit |
| **CPU** | **C**entral **P**rocessing **U**nit |
| **SRAM** | **S**tatic **R**andom **A**ccess **M**emory |
| **MSPS** | **M**ega **S**amples **P**er **S**econd |
| **ADC** | **A**nalog to **D**igital **C**onverter |
| **DAC** | **D**igital to **A**nalog **C**onverter |
| **I$^2$C** | **I**nter **I**ntegrated **C**ircuit |
| **SPI** | **S**erial **P**eripheral **I**nterface |
| **UART** | **U**niversal **A**synchronous **T**ransmitter / **R**eceiver |
| **SD** card | **S**ecure **D**igital card |
| **SDIO** | **S**ecure **D**igital **I**nput **O**utpout |
| **DMA** | **D**irect **M**emory **A**ccess |
| **BLE** | **B**luetooth **L**ow **E**nergy |

DOI:10.15774/PPKE.ITK.2017.002



| | |
|---|---|
| **RTOS** | **R**eal-**T**ime **O**perating **S**ystem |
| **FPU** | **F**loating **P**oint **U**nit |
| **MEMS** | **M**icro-**E**lectro-**M**echanical **S**ystem |
| **IMU** | **I**nertial **M**easurement **U**nit |
| **DMP** | **D**igital **M**otion **P**rocessor |
| **FIFO** | **F**irst **I**n **F**irst **O**ut |
| **DC** | **D**irect **C**urrent |
| **AAF** | **A**nti-**A**liasing **F**ilter |
| **LPF** | **L**ow **P**ass **F**ilter |
| **HPF** | **H**igh **P**ass **F**ilter |
| **FIR** | **F**inite **I**mpulse **R**esponse |
| **SAR** | **S**uccessive **A**pproximation **R**egister |
| **DSP** | **D**igital **S**ignal **P**rocessing |
| **SPP** | **S**erial **P**ort **P**rofile |
| **GUI** | **G**raphical **U**ser **I**nterface |
| **DPS** | **D**egrees **P**er **S**econd |
| **PCB** | **P**rinted **C**ircuit **B**oard |
| **PC** | **P**ersonal **C**omputer |
| **PCA** | **P**rincipal **C**omponent **A**nalysis |
| **LoS** | **L**ine **o**f **S**ight |
| **API** | **A**pplication **P**rogramming **I**nterface |
| **IK** | **I**nverse **K**inematics |
| **XML** | E**X**tensible **M**arkup **L**anguage |
| **PMx** | **P**rototype **M**arker x |
| **VMx** | **V**irtual **M**arker x |
| **ANSI** | **A**merican **N**ational **S**tandards **I**nstitute |
| **RMS** | **R**oot **M**ean **S**quare |

DOI:10.15774/PPKE.ITK.2017.002

# Chapter 1

# Introduction

## 1.1 Motivation and scope

Movement is an essential part of our daily lives. It is so essential that we tend to forget its importance and take it for granted until we have to experience restrictions in our movement capabilities because of one of surprisingly many possible reasons. The human movement system is extremely complex from both anatomical and functional aspects, having centralized and distributed nature at the same time that assures adequate operation in very diverse situations.

As a simple example let us consider the case when someone touches a hot surface without any prior knowledge about its temperature: the movement starts with a voluntary part where motion planning, initiation and timing (among other functions) are performed in the interconnected neural structures of the cerebrum, cerebellum and brainstem, followed by the transmission of the execution commands to intermediate gateways in the spinal cord controlling the coordinated operation of different muscle groups through their $\alpha$-motoneuron pools to make the arm extend and touch the surface. But simultaneously, the arm contains additional automatic built-in safety mechanisms (reflexes) that serve protecting purposes against various damaging factors – like burning the skin in our example – functioning independently of the central nervous system that results in far shorter reaction times than we could achieve voluntarily (generating an evasive movement even before we start noticing the pain). These reflexes are independent of cognitive processes as their functions are realized by direct wiring of neural pathways between the

DOI:10.15774/PPKE.ITK.2017.002



sensor elements (muscle spindles, Golgi-tendons and various receptors of the skin) and the $\alpha$-motoneuron pools in the spinal cord responsible for direct muscle control. Without going into further details (a comprehensive material on the topic can be found in [1]), even this simple example shows that the complexity and systematic structure of our movement system is an exciting, yet to be fully explored field of science.

Human movement science and selected fields of biomedical engineering aim to develop methodologies for proper examination of various aspects of our movement system. By elaborating quantitative measurement and analysis techniques of human movements, advancements in these fields have contributed to movement rehabilitation techniques [2], performance analysis of athletes [3] and general understanding of the motor system [4] during the last decades by making objective movement pattern comparison possible. This advancement was further accelerated by model-based analysis approaches that enabled explicit characterization of the studied movement patterns [5].

It should be noted however that the possibilities for quantitative examination of the movement system in its whole are always constrained by the technology available to date. This is particularly true in the case of analysis of complex scenarios involving higher functional levels of the movement system (possibly including cognitive processes) that cannot be measured and tracked directly today. On the other hand, direct measurement of movement kinematics has became a standard process that can be performed with various movement analyzer systems using electromagnetic[1], mechanical[2], ultrasound[3], optical[4] or inertial[5] technology. These measurements combined with additional recorded modalities (e.g. biopotential, force or (in rare cases) fMRI data) form the observation space in the wast majority of human movement analysis studies. As a consequence, conclusions about the underlying processes often have to be drawn incorporating simplified mathematical models or previous behavioral observations into these studies, in addition to appropriate experimental design that excludes as many uncontrolled factors as possible.

As a subfield of movement science, understanding the nature and internal workings of human arm movements is of particularly high interest because we use our arms and hands in every situation when object manipulation is needed at any complexity level.

[1] Polhemus' product portfolio – http://polhemus.com/motion-tracking/overview/
[2] Gypsy 7 – http://metamotion.com/gypsy/gypsy-motion-capture-system.htm
[3] Zebris' CMS systems – http://www.zebris.de
[4] Vicon – https://www.vicon.com/; OptiTrack – http://www.optitrack.com/
[5] Xsens – https://www.xsens.com/

DOI:10.15774/PPKE.ITK.2017.002



As a consequence, analysis of the underlying control methods in specific movement tasks and environmental conditions is a key aspect to gain more detailed knowledge of neural processes of our manual interaction with the environment.

In this thesis, three main topics are presented as an effort to contribute to the field of human movement science. The first topic covers an experimental study investigating details of target tracking arm movements while the other two introduce engineering contributions to the field of measurement techniques by presenting the design and implementation of a wearable measurement system along with an algorithm that establishes the connection between inertial measurements and model based analysis of movement kinematics.

## 1.2 Thesis outline

Chapter 2 introduces an experimental setup and procedure that was designed to investigate how motor synergies differ between predictive and non-predictive movements. Motor synergies were evaluated by applying the UCM method to the joint angle variance during 2D tracking of a target on a graphics tablet, where the 2D pen position was used as the hypothetical task variable. It was investigated whether the synergy index drops during predictive, internally driven tracking movements compared to visually, externally driven tracking movements. To address this question, tracking movements between periodic (and pre-trained) and non-periodic presentation modes were compared, which are known to challenge predictive and visually driven tracking modes respectively.

Chapter 3 describes the engineering prototype development of a custom wearable measurement device based on my experiences with a movement analyzer system using ultrasound technology. The prototype incorporates inertial sensors for movement recording to overcome issues accompanying measurements with the previous system (i.e. bulky setup, highly constrained measurement volume and low sampling rate) and enables evaluation and analysis of various sensor calibration, filtering and sensor fusion algorithms in a fully customizable manner.

An additional goal of this thesis is to extend the measurement and analysis workflow of human arm movements with a method that allows accurate and real-time calculation of anatomical joint angles for a widely used SIMM/OpenSim upper limb model when

DOI:10.15774/PPKE.ITK.2017.002



measurements are performed with the developed prototype. For this purpose a custom kinematic algorithm is introduced in Chapter 4 that utilizes orientation information of arm segments (directly measurable with inertial sensors) to perform joint angle reconstruction in real-time.

Chapter 5 summarizes the results and concludes the thesis.

DOI:10.15774/PPKE.ITK.2017.002

# Chapter 2

# Synergistic Control in Manual Tracking

The current chapter is based on the author's articlce entitled *"Motor synergies during manual tracking differ between familiar and unfamiliar trajectories"*. [J1]

## 2.1 Background

Manual and visual target tracking represents an important part of human movements and shows strong predictive behavior. This becomes most obvious when comparing tracking onset delay with phase delays during pursuit of periodic movements [6] or when the movement continues after disappearance of the target [7]. Oculomotor and manual tracking responses affect each other [8, 9] and seem to share predictive mechanisms [10]. Previous studies investigating predictive manual tracking focused on the explanation of finger acceleration as a function of the 2D-tracking error [11] and on the relation between the 3D-tracking error and path curvature and spatial depth [12] but did not consider the control of shoulder, elbow and wrist joints.

### The Uncontrolled Manifold Method

The analysis of joint angle variability, especially its structural decomposition into task-relevant and task-irrelevant components with respect to hypothesized task variables, is

DOI:10.15774/PPKE.ITK.2017.002



used to address redundancy in movement control mechanisms and was proposed by Scholz and Schöner [13] as the "Uncontrolled Manifold Method" ($UCM$). In this context the term "task variable" does not imply that it was explicitly addressed in the instructions to the subject, but that the covariation in the effector space is optimized to stabilize this variable. The main concept of the $UCM$ is to divide the total variance of the joint angles into two orthogonal sub-components that do and do not affect the proposed task variable. The variance in the component which does not influence the task variable is called the "uncontrolled variance" ($V_{\mathrm{UCM}}$) and can be used as an indicator of flexibility of the control system, while variance in its orthogonal component is called the "controlled" or "orthogonal variance" ($V_{\mathrm{ORT}}$). The relative size of $V_{\mathrm{UCM}}$ with respect to $V_{\mathrm{ORT}}$, quantified by the so-called synergy index, can be used to characterize the stability of the task variable [14].

In more detail, after selecting a hypothetical task variable (e.g. the endpoint of the arm), the Jacobian matrix ($\mathbf{J}$) can be obtained by the linearization of the task variable components expressed as a function of joint angles at the mean arm configuration. $\mathbf{J}$ expresses the linear mapping of differential changes in the 7-dimensional joint-space ($\Delta\underline{\varphi}$) to differential changes of the task variable ($\Delta\underline{\nu}$) as shown in (2.1).

$$\Delta\underline{\nu} = \mathbf{J}\,\Delta\underline{\varphi} \tag{2.1}$$

The basis of the uncontrolled manifold is named $\mathbf{B}_{\mathrm{UCM}}$, that of the orthogonal subspace is called $\mathbf{B}_{\mathrm{ORT}}$. $\mathbf{B}_{\mathrm{ORT}}$ and $\mathbf{B}_{\mathrm{UCM}}$ can be obtained from the $\mathbf{Q}$ matrix produced by the QR-decomposition of the transposed Jacobian matrix: $[\mathbf{Q}, \mathbf{R}] = \mathrm{qr}\left(\mathbf{J}^T\right)$; $\mathbf{Q} = [\mathbf{B}_{\mathrm{ORT}}, \mathbf{B}_{\mathrm{UCM}}]$, where $\mathbf{Q}$ is orthogonal, $\mathbf{B}_{\mathrm{ORT}}$ contains the first three and $\mathbf{B}_{\mathrm{UCM}}$ the last four columns of $\mathbf{Q}$. The normalized variances of the projections on the two subspaces can be computed as shown in (2.2) and (2.3), where $DoF_{\mathrm{ORT}}$ and $DoF_{\mathrm{UCM}}$ denote the dimensions of $\mathbf{B}_{\mathrm{ORT}}$ and $\mathbf{B}_{\mathrm{UCM}}$, respectively. The synergy index is defined as $si = \dfrac{V_{\mathrm{UCM}}}{V_{\mathrm{ORT}}}$.

$$V_{\mathrm{ORT}} = \frac{\mathrm{trace}\left(\mathbf{B}_{\mathrm{ORT}}^T\,\mathbf{\Sigma}\,\mathbf{B}_{\mathrm{ORT}}\right)}{DoF_{\mathrm{ORT}}} \tag{2.2}$$

$$V_{\mathrm{UCM}} = \frac{\mathrm{trace}\left(\mathbf{B}_{\mathrm{UCM}}^T\,\mathbf{\Sigma}\,\mathbf{B}_{\mathrm{UCM}}\right)}{DoF_{\mathrm{UCM}}} \tag{2.3}$$

DOI:10.15774/PPKE.ITK.2017.002



The normalization of the variances in the two subspaces to their respective dimensions is needed to ensure that, for a spherical distribution of the effector variables, the synergy index has the expected value of one, independent of the dimension of the task variable.

A large synergy index of the joint angle variance with respect to the task variable indicates that the "bad" variance (affecting the task variable) is relatively small compared to "good" variance (not affecting the task variable). It is important to note that this synergy index is specific for the chosen task variable and is not a general measure of covariation. The $UCM$ method has been used to show the synergistic properties of the motor control system involving reaching [15, J3], finger coordination [16, 17, 18] and bimanual pointing tasks [19, 20].

### Optimal feedback control theory

According to the theory of optimal feedback control [21], motor synergies can be explained by a feedback controller minimizing costs expressed as the sum of two terms, a so called task-error depending on system states (joint angles and velocities), and a penalty on the control signals. This theory predicts that the synergy index decreases under open loop conditions, increases with increasing motor noise, and decreases with increasing sensory noise. Therefore, it can be expected that predictive and non-predictive movements which differ in precision (noise) and in the contribution of feedforward commands, show different motor synergies. However, such expectations of the theory are based on the assumption that the feedback controller is optimally adjusted to the actual conditions.

This chapter investigates how motor synergies differ between predictive and non-predictive movements. Motor synergies were evaluated by applying the $UCM$ method to the joint angle variance during 2D tracking of a target on a graphics tablet, where the 2D pen position was used as the hypothetical task variable. It was investigated whether the synergy index drops during predictive, internally driven tracking movements compared to visually, externally driven tracking movements. To address this question, the presented work compares tracking movements between periodic (and pre-trained) and non-periodic presentation modes, which are known to challenge predictive or visually driven tracking modes respectively.

DOI:10.15774/PPKE.ITK.2017.002



## 2.2 Methods

### 2.2.1 Subjects

Seven healthy subjects participated in the study (6 males, 1 female, age: 33.4 ± 12.4 years, mean ± standard deviation). All subjects had normal or corrected-to-normal vision. Five subjects had right hand dominance and 2 subjects had left hand dominance according to their preferential hand use during writing. All subjects performed the movements with their dominant hand. Because of marker measurement errors that could not be corrected, data from one of the right-handed male subjects were excluded from kinematic analysis of joint angle variances, but not from tracking performance analysis. Subjects had given informed consent prior to participation in the experiment. The experimental procedure was in accordance with the Declaration of Helsinki and approved by the local Ethics Committee.

### 2.2.2 Experimental setup

The subjects sat in front of a table which was mounted with a graphic tablet that featured an integrated display (Figure 2.1, WACOM Cintiq 21UX, 43.2 × 32.4 cm, frame rate: 60 Hz) used for presentation of the target. The target was a white colored disk (diameter 1 cm) and moved in front of a gray background. The luminous sterance of the background luminance was about 20 cd/m$^2$ and the target had a luminance of about 130 cd/m$^2$. The sitting position of each subject was adjusted in the following way: (1) the body midline of the subject was aligned with the vertical midline of the graphic tablet, (2) the distance between the table on which the tablet was mounted and the subject's chair, and the height of the chair were aligned to get the subject facing the graphic tablet's midpoint orthogonally, (3) all possible target positions were within the anatomical range of motion of the subject's measured arm, but not at extreme positions, and (4) to prevent trunk movements and to fixate the head, a supporting chin-rest with adjustable height was used. Post-hoc analysis revealed that the average movement amplitude of the acromion, defined as the largest distance between any pair of time samples of its 3D trajectory, was 1.48 ± 0.25 cm (N=6). The viewing distance of the display was 40 cm.

DOI:10.15774/PPKE.ITK.2017.002



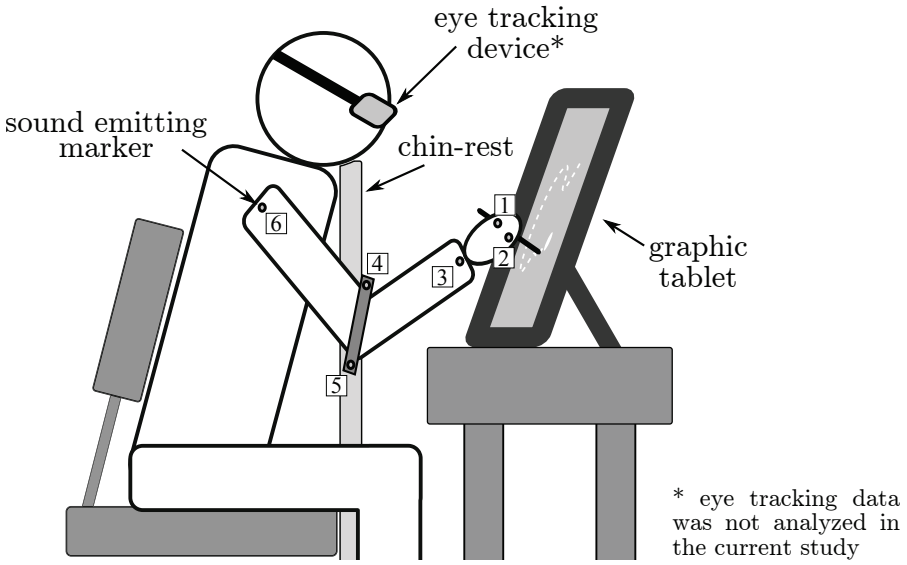


FIGURE 2.1: **Experimental setup.** Numbers 1-6 indicate ultrasound markers of the motion analysis system. The elbow bracelet that held the fourth and fifth markers was not attached to the chin-rest (as the figure may give an incorrect intuition), just to the subject's arm. Subjects sat on a chair that was adjusted to provide stable fixation for their trunk. The target is depicted on the graphic tablet's screen as a white disk. The dashed line was never presented on the screen, it is only used in the figure to imply target movement. The eye tracking device is depicted in the figure because the measurement was part of an extended study that involved the analysis of eye movement data in addition to movement kinematics, however this modality was not included in the analysis presented in the thesis.

Subjects were asked to track the target as accurately as possible with the pen of the graphic tablet using their dominant hand. Tracking performance was analyzed based on the pen data, recorded as 2D coordinates in the tablet's reference frame. Arm movements were recorded by an ultrasound-based movement analyzer system (Zebris Medical, Isny, Germany) running at 33 Hz. The markers were attached to the following positions: (1) the metacarpophalangeal joint of the index finger, (2) the metacarpophalangeal joint of the little finger, (3) the center of the wrist. Markers (4) and (5) were attached to a bracelet directly above the elbow at the medial and lateral ends of the bracelet. Marker (6) was attached to the acromion. From these marker positions, a geometrical model of the arm with 7 degrees of freedom (DoF) was reconstructed in the tablet's reference frame using a method described previously [J3]. The origin of this reference frame was the center of the tablet's screen, its $x$ and $y$ axes coincided with the screen's horizontal and vertical axes, while the $z$-axis was perpendicular to the screen pointing towards the subject (forming a right-hand coordinate system).

DOI:10.15774/PPKE.ITK.2017.002



All data (target position, pen position and marker positions) were recorded by a computer running the real-time REX system [22], a QNX Neutrino RTOS-based software environment that is widely used to control neuroscience experiments[1] and mapped at the common sampling rate of 1 kHz.

### 2.2.3 Design and procedure

As described in the introduction, the aim of the study was to investigate the effects of predictive and non-predictive tracking modes on movement performance and control. To achieve this, various 2D target trajectories were generated with a pseudo-random shape. One of these trajectories (TR1) was only presented in periodic repetitions, whereas the other trajectories (TR2, TR3, TR4) were presented in a random order without repetitions. Moreover, in order to further stimulate predictive tracking of TR1, subjects were familiarized with this trajectory during an initial training block (see Measurement blocks for further details of the experimental design).

#### Trajectories

The 2D-velocities of the generated trajectories were based on sums of 5 harmonics with random phase and a base frequency corresponding to a period of 4 s. The 5 harmonics had frequencies of 0.25, 1, 1.75, 2.5, and 3.25 Hz. The peak velocities of the components were proportional to a Gaussian envelope with a value 1 at 0 Hz and decayed for higher frequencies. The decay of 3 dB was reached at 0.75 Hz. The frequency and magnitude values of the summed components were selected after visual inspection of various parameter combinations. Both $x$ and $y$ velocities were generated independently. Integrating these velocities yielded the 2D trajectories. These trajectories were then resampled nonlinearly to adjust the sampling distance ($sd$) according to (2.4), where $V$ denotes the tangential velocity, $r$ is the radius of curvature, and $K$ and $\alpha$ are two free parameters [23]. This method is known as the two-thirds power law that describes an empirical relationship between the shape and the kinematics of free-hand and manual tracking movements [24, 25], and it was applied here to assure that the subjects perceive the generated trajectories as natural as possible to avoid trajectory-induced error factors during the analyzed tracking movements.

[1] http://www.qnx.com/products/neutrino-rtos/neutrino-rtos.html?lang=en

DOI:10.15774/PPKE.ITK.2017.002



$$\frac{sd}{\Delta t} = V = K\left(\frac{r}{1+\alpha\cdot r}\right)^{\frac{1}{3}} \tag{2.4}$$

The variable $\Delta t$ specifies the sampling interval for the generated movement trace. The parameters $K$ and $\alpha$ were adjusted to achieve a mean tangential velocity of 10 cm/s and a ratio between the maximum and minimum tangential velocity of 2.

In this way 14 different random trajectories were generated independently from each other. Figure 2.2 shows the 4 trajectories (TR1 to TR4) used for repeated presentation, as described in the next section. The remaining 10 trajectories (uniformly denoted as TRU) were used to introduce "unpredictable" sections, as described below.

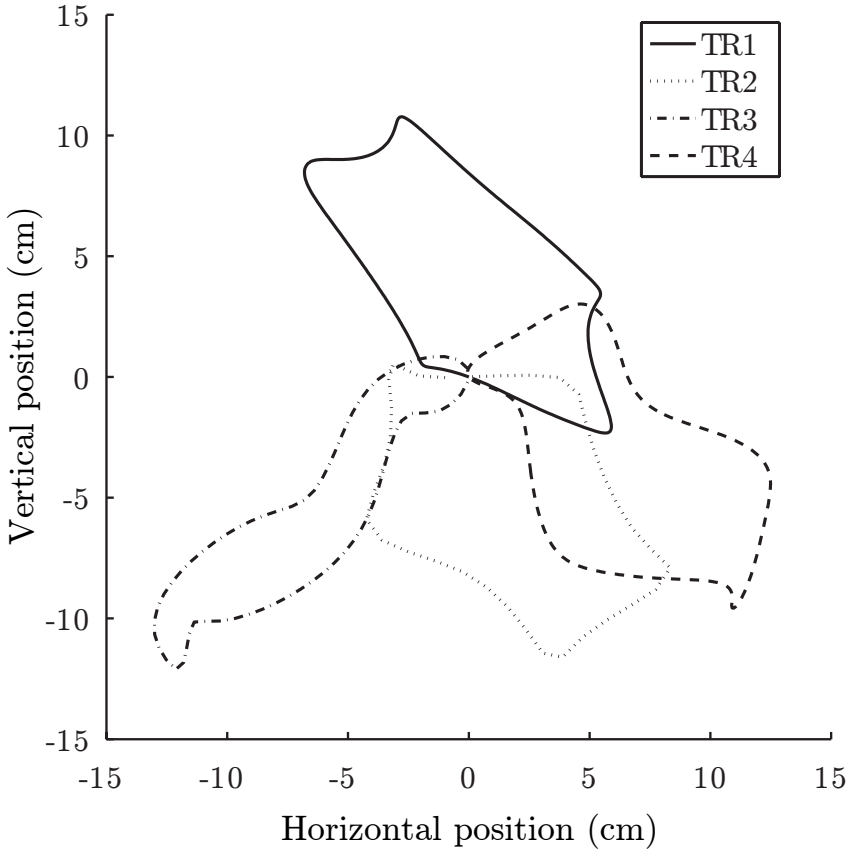


FIGURE 2.2: **The presented target trajectories.** Closed traces were generated by integrating sums of 5 harmonics with a random phase and a base frequency corresponding to a period of 4 s. Labels (TR1-TR4) were assigned randomly to the generated trajectories.

DOI:10.15774/PPKE.ITK.2017.002



**Measurement blocks**

The smallest unit of the design was one presentation of a generated trajectory, which is referred to as a "trial". Trials were grouped into so-called "sub-blocks", followed by a pause of 4 s. The initial trials of these sub-blocks were not included in the analysis because they differed from the other continuation trials in the movement initiation required after the 4 s pause (see also Data exclusion). The main experiment was composed of 6 blocks, each consisting of several sub-blocks as described below (for a graphical representation, see Figure 2.3). Blocks were separated by a break of about 5 minutes.

(1) In the first block, only the trajectory TR1 was presented in the so-called *periodic training* presentation mode as follows. The block consisted of 10 sub-blocks each containing 4 trials with periodic presentation of TR1. The purpose of this block was to make the subject familiar with the selected trajectory without introducing unwanted fatigue effects (pauses between sub-block executions). From the 40 presented trials, 30 continuation trials were analyzed.

(2) The *periodic training* block was followed by 5 test blocks, each presenting 12 sub-blocks. Six of these sub-blocks showed the *non-periodic test* presentation mode and contained the three trajectories TR2, TR3 and TR4 in a random order led by one of the unpredictable sections (TRU). Each of these 6 sub-blocks contained one of 6 possible permutations of TR2, TR3 and TR4. Alternating with the *non-periodic test* sub-blocks, 6 sub-blocks were inserted with TR1 in the so-called *periodic test* presentation mode. This presentation mode was – apart from the vicinity to the *non-periodic test* sub-blocks – identical to the *periodic training* mode. After exclusion of the initial trials of the *periodic test* sub-blocks and the initial, unpredictable sections of the *non-periodic test* sub-blocks, the five test blocks provided in total 90 trials (5 blocks $\times$ 6 sub-blocks $\times$ 3 trials) of *periodic test* trials (TR1), and 90 trials (5 blocks $\times$ 6 sub-blocks $\times$ 3 trials) of *non-periodic test* trials (TR2, TR3 and TR4).

The specific structure of the non-periodic sub-blocks kept the subjects under the illusion of path randomness despite repetitive presentations of TR2-TR4. These repetitions were necessary to calculate joint angle variance-covariance which is the basis for the Uncontrolled Manifold Method.

DOI:10.15774/PPKE.ITK.2017.002



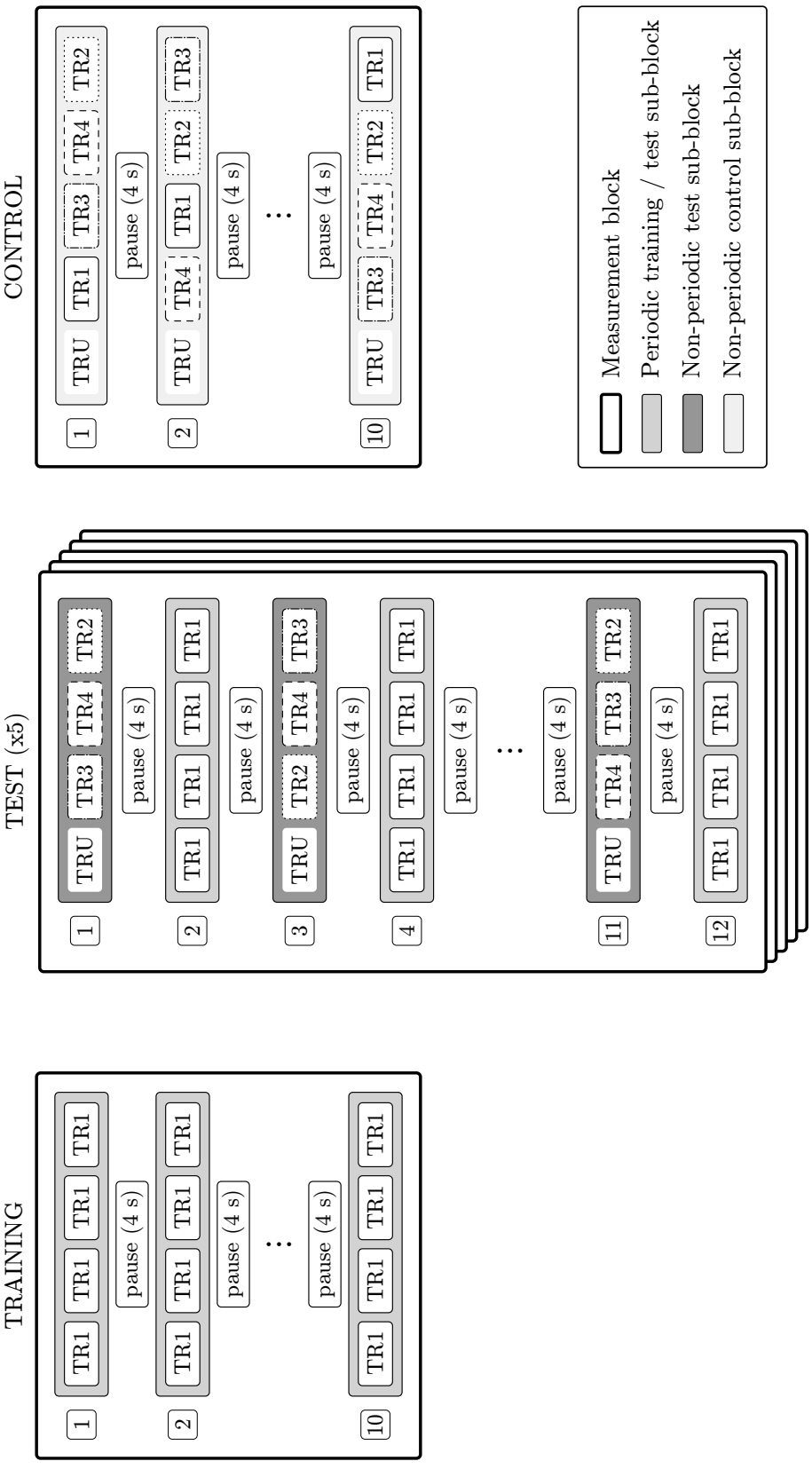


FIGURE 2.3: **Measurement blocks of the experiment.** The smallest unit of the design was one presentation of a generated trajectory, which is referred to as a "trial". Trials were grouped into so-called "sub-blocks", followed by a pause of 4 s. The main experiment was composed of 6 blocks (TRAINING + 5 x TEST), each consisting of several sub-blocks of selected trajectories. The measurement with the CONTROL block was performed after the main experiment to test whether differences between periodic and non-periodic presentations were related to the presentation modes and not to differences between the trajectories. The borders of the graphical boxes denoting TR1-TR4 refer to the corresponding trajectories shown in Figure 2.2

DOI:10.15774/PPKE.ITK.2017.002



To test whether effects of periodic or non-periodic presentations were related to differences between the trajectories rather than to the presentation modes a control experiment was performed on a different day, at least five weeks after the main experiment. This control consisted of a single "non-periodic" block with 10 sub-blocks, each starting with one of the "unpredictable" sections (TRU) followed by TR1, TR2, TR3 and TR4 in a random order. Thus, each trajectory was presented 10 times.

### 2.2.4 Data analysis

All data analysis was performed using MATLAB 7.9.0 (Mathworks, Natick, USA).

#### Tracking delay

Reduced tracking delay is the primary indicator for predictive versus visually driven tracking modes. Therefore, tracking delay was assessed by the time lag (ms) of pen position, evaluated by an algorithm used in previous studies [12]. In this algorithm, the hand-target distance was computed between the current hand position and the target position at any sampling point between 500 ms before and 100 ms after the current time. The lag was defined as the time point at which the hand-target distance was minimal. The average tracking delay was computed separately for each *subject*, *block*, *presentation mode*, and was averaged across all respective sampling points.

#### Data exclusion

The initial trial of each sub-block showed a tracking delay which started at a large value and rapidly decreased during the first half-cycle (2 s), due to movement initialization. For example, in the initial trial of the first sub-block of the training the tracking delay decreased during the first 2 s from 292 $\pm$ 67 ms to 120 $\pm$ 53 ms. For that reason, all initial trials of all sub-blocks were excluded from the analysis. As a further step of preprocessing, the time courses of the tracking delays were checked for the occurrence of discontinuities (related to discontinuous mapping between hand position and target position). Since these discontinuities occurred only near to the start and the end of the trials, all data from a time window of 1 s around the trial start were excluded from the analysis.

DOI:10.15774/PPKE.ITK.2017.002



### Application of the uncontrolled manifold method

Like the average tracking delay, the uncontrolled manifold method was applied separately for each combination of the factors *subject*, *block*, and *presentation mode*, averaged across all respective sampling points. Following the method described in Section 2.1, the synergy index was calculated with the pen position as the hypothetical task variable. Because of this, the arm configuration was constrained to 6 DoF since the $z$-component of the pen was constrained to the surface of the screen. Thus, the normalization factors were $DoF_{\mathrm{ORT}} = 2$ and $DoF_{\mathrm{UCM}} = 4$. The total variance ($V_{total}$) of the joint angles was computed as the trace of the variance-covariance matrix ($V_{total} = \mathrm{trace}(\mathbf{\Sigma})$).

### Statistical analysis

To test whether tracking performance differed between the trajectories (TR1 - TR4), each of the dependent variables *tracking delay*, the *total variance* and the *synergy index* of the control experiment was submitted to a repeated measures ANOVA with the factor trajectory (4 levels). For the main experiment each of these dependent variables was submitted to two repeated measures ANOVAs, one for the periodic training block and one for the test blocks. To analyze potential learning effects during the training consecutive pairs of the 10 sub-blocks were pooled to form a repeated factor *block number* with 5 levels. To analyze the differences between periodic and non-periodic presentation modes and potential training effects in the test blocks, the two repeated measures factors *presentation mode* (2 levels) and *block number* (5 levels) were used. Post-hoc tests were performed using Tukey's HSD test. Effects were considered significant for $\alpha$-errors $p < 0.05$. $\alpha$-errors different from this value are reported in the results to give a better intuition about the particular effect (e.g. $p = 0.06$ is a non-significant but marginal effect). Normality of the analyzed variables was checked with the Lilliefors test. Data sphericity was tested using Mauchly's sphericity test. Wilks' lambda multivariate test was applied if sphericity was not fulfilled. Descriptives of normally distributed variables are given as mean ± standard deviation and as median [interquartile range] otherwise. Statistical results are reported in the following standard forms:

One-sided and paired T-test: $\mathbf{T(df) = T\text{-}value,\ p < p\text{-}value}$

Repeated measures ANOVA: $\mathbf{F(df_{levels}, df_{error}) = F\text{-}value,\ p\ \{<, =, >\}\ p\text{-}value}$

DOI:10.15774/PPKE.ITK.2017.002



where df means degrees of freedom and T-value and F-value are the values of the corresponding T and F statistics (for short descriptions of the methods and reporting standards, please follow the links in the footnotes [2] [3]).

## 2.3 Results

Subjects reported that they felt familiar with the trajectory TR1 after the pure periodic training block of the main experiment, and that they also recognized it easily in the periodic test sub-blocks. In contrast, none of the subjects noticed that the trajectories TR2-TR4 were repeatedly presented. In the control experiment the tracking delay was 160 ± 20 ms, averaged across the 7 subjects, and showed a marginal effect of the factor *trajectory* (main effect $F(3,18) = 2.94$; $p = 0.06$). However, none of the post-hoc tests reached significance (Tukey: $p > 0.1$), indicating that the different shapes of the trajectories did not have a major effect on tracking mode. This was further supported by the observation that neither the *total variance* (48.8 ± 21.6 $deg^2$) of the joint angles nor the *synergy index* (17.0 ± 11.5) differed between the four trajectories (main effect of *trajectory* ($F(3,15) < 0.43$; $p > 0.73$).

### 2.3.1 Training block

In the training block the tracking delay decreased from 116 ± 41 ms during the first block to 94 ± 33 ms in the last block (Figure 2.4). This decrease was significant, as confirmed by the main effect of the factor block ($F(4,24) = 3.68$; $p < 0.02$). In the post-hoc test the tracking delay turned out to be longer in the first blocks than in the last three blocks (Tukey: $p < 0.05$).

The overall mean of the synergy index during the training was 3.8 ± 2.0, significantly larger than 1 (one sided T-test: $T(5) = 3.4$; $p < 0.01$), indicating that most of the joint angle variance was irrelevant for the pen position. There was also a significant main effect of the factor *block number* on the synergy index ($F(4,2) = 50.9$; $p < 0.02$). The third block showed a larger synergy index (5.77 ± 4.71) than the last block (1.93 ± 0.98;

[2] https://statistics.laerd.com/statistical-guides/dependent-t-test-statistical-guide.php
[3] https://statistics.laerd.com/statistical-guides/repeated-measures-anova-statistical-guide.php

DOI:10.15774/PPKE.ITK.2017.002



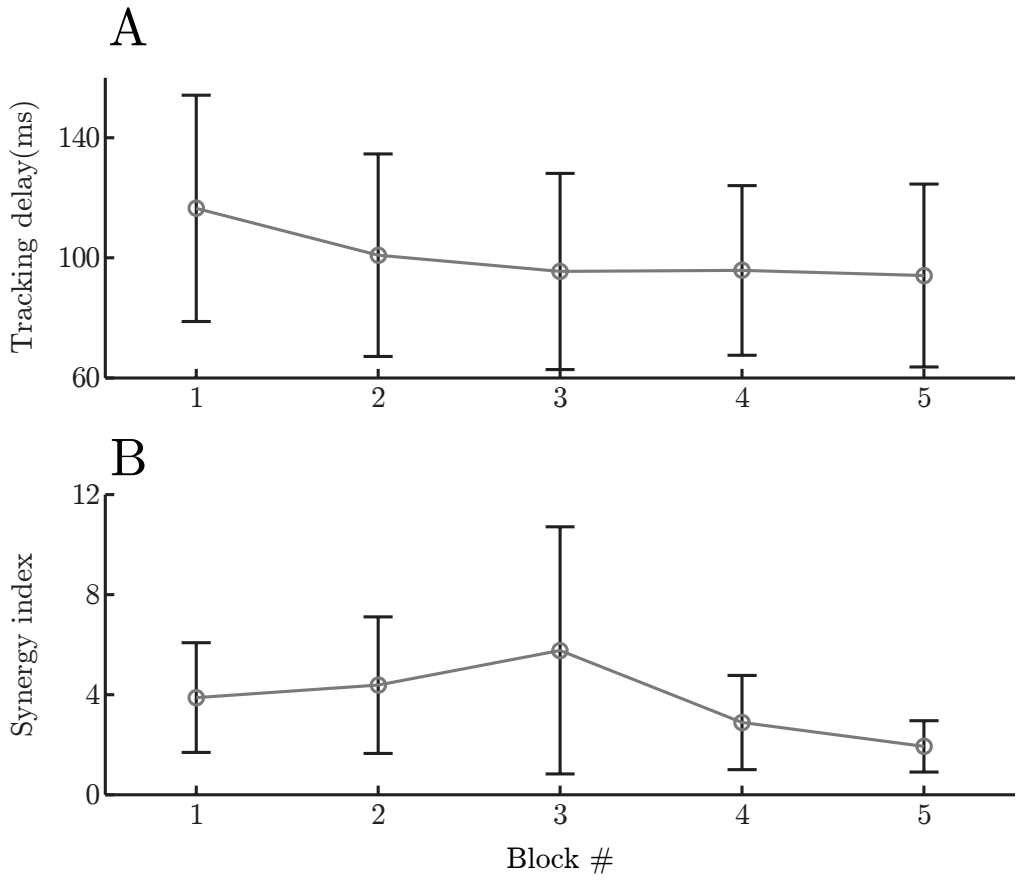


FIGURE 2.4: **Time course of the tracking delay (A) and of the synergy index (B) in the training block.** Consecutive pairs of the 10 sub-blocks were pooled to form the factor block number with 5 levels. Circles indicate the mean across subjects, and the whiskers indicate the 95% confidence interval of the mean. The occurrence of learning is suggested by the decrease of the tracking delay and of the synergy index across the blocks.

Tukey: $p < 0.04$). Thus, the time course of both the synergy index and the tracking delay showed a decrease during training.

The overall mean of the total variance of the joint angles remained relatively low during training ($5.64 \pm 1.78$ deg$^2$) and did not show a significant main effect on the factor *block number*.

### 2.3.2 Test blocks

The tracking delay during the test block was smaller (main effect of *presentation mode* $F(1,6) = 84.3; p < 0.001$) in the periodic presentation mode ($91 \pm 24$ ms; N=7) compared to the non-periodic presentation mode ($145 \pm 27$ ms; N=7). No main effect or interaction

DOI:10.15774/PPKE.ITK.2017.002



involving the factor *block number* was observed, indicating that the tracking delay was stable throughout the test blocks.

Figure 2.5 A-C shows the effect of the presentation mode on the total variance of the joint angles and their two normalized projections. The normalized variance in the uncontrolled manifold ($V_{\mathrm{UCM}}$) was 16.3 ± 12.0 deg$^2$ during the non-periodic presentation mode and strongly decreased in the periodic presentation mode by 75% to 4.0 ± 2.7 deg$^2$ (Figure 2.5 B). The normalized variance in the orthogonal subspace ($V_{\mathrm{ORT}}$) decreased by 42% from 1.2 ± 0.7 deg$^2$ (non-periodic) to 0.7 ± 0.4 deg$^2$ (Figure 2.5 C, periodic). Figure 2.5 D shows that the stronger decrease of $V_{\mathrm{UCM}}$ than of $V_{\mathrm{ORT}}$ also caused the synergy index to be smaller (main effect of *presentation mode*: $F(1,5) = 6.63$; $p < 0.05$) in the periodic presentation mode (9.07.6; N=6) compared to the non-periodic presentation mode (15.2 ± 10.4; N=6). Like for the tracking delay, also for $V_{\mathrm{UCM}}$ or $V_{\mathrm{ORT}}$ no main effect or interaction involving the factor *block* was observed.

Unsurprisingly, because of the decrease of $V_{\mathrm{UCM}}$ and $V_{\mathrm{ORT}}$, the total variance of the joint angles was also smaller (paired T-test: $T(5) = 2.57$; $p < 0.05$) during the periodic (17.46 ± 10.50 deg$^2$) than during the non-periodic presentation mode (67.82 ± 49.34 deg$^2$, Figure 2.5 A). Interestingly, when the very same trajectory TR1 was tracked periodically during the training block, the total variance (5.64 ± 1.78 deg$^2$) was even smaller than during the test blocks (paired T-test: $T(5) = 2.91$; $p < 0.05$).

In the inter-trial standard deviation of the 2D-pen position, the decrease of $V_{\mathrm{ORT}}$ during the periodic presentation was only reflected in an insignificant difference between the non-periodic (9.64 [6.78] mm) and the periodic presentation mode (7.42 [2.01] mm).

## 2.4 Discussion

The purpose of this study was to investigate the effects of prediction on joint angle variability in manual tracking movements. The results of the test blocks showed that both task-relevant and task-irrelevant variance decreased during tracking of familiar compared to unfamiliar trajectories. Since this decrease was stronger for the irrelevant than for the relevant variance, the synergy index also decreased during periodic tracking. The synergy index, as well as the total variance of the joint angles was smallest during periodic training, intermediate during the periodic test, and largest during the non-periodic test.

DOI:10.15774/PPKE.ITK.2017.002



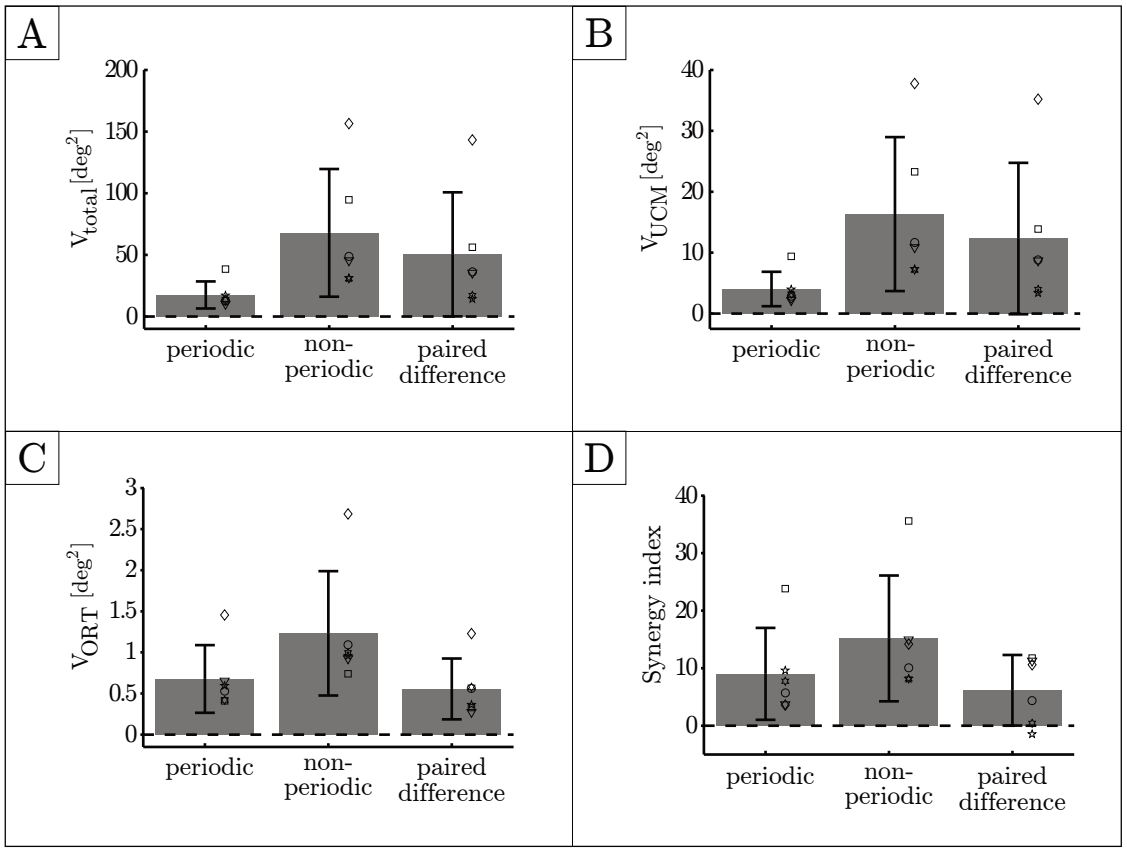


FIGURE 2.5: **Effects of the presentation mode on joint angle variances.** Small open symbols show the total variance of the joint angles (A: $V_{total} = trace(\Sigma)$) and the normalized projections of the variance on the subspaces irrelevant (B: $V_{\mathrm{UCM}}$) or relevant (C: $V_{\mathrm{ORT}}$ , see Eq. 2) for each of the 6 subjects. Note the different scaling of the ordinates. The variances were acquired during the test block in the periodic presentation mode (*periodic*) and the non-periodic presentation mode (*non-periodic*). The symbols labeled *paired difference* show the difference in the variance between non-periodic and periodic presentation modes for each subject. Symbols in panel D) show the synergy index. Bars indicate the mean across subjects, and the whiskers indicate the 95% confidence interval of the mean. All variances ($V_{\mathrm{total}}$, $V_{\mathrm{UCM}}$, $V_{\mathrm{ORT}}$) decreased in the periodic compared to the non-periodic presentation mode. The decrease was larger for $V_{\mathrm{UCM}}$ than for $V_{\mathrm{ORT}}$, leading also to a decreased synergy index during periodic presentation.

The control experiment showed that these differences were indeed due to the presentation mode and not just an artifact due to the different shapes of the trajectories.

### 2.4.1 Predictive tracking

The observation that subjects recognized the trajectory TR1, but none of the other trajectories, shows that knowledge about TR1 acquired during training was used for cognitive processes. It also suggests that this knowledge could be used for movement control. That predictive command components played a larger role during tracking

DOI:10.15774/PPKE.ITK.2017.002



of the familiar than of the unfamiliar trajectories is mostly supported by the reduced tracking delay on TR1 that gradually decreased during training. Similar developments of predictive command components during repetitive manual tracking of the same trajectory were observed previously [12, 26].

Since the trajectories TR2-TR4 were also repeatedly presented during the test blocks, and because they were, like TR1, a superposition of only a few harmonic components, it is rather likely that predictive strategies were also used on the trajectories TR2-TR4. However, the reduced tracking delay on TR1 suggests that the observed differences between the "familiar" and the "unfamiliar" trajectories were most likely related to the amount of prior knowledge used for movement control.

Remarkably, the tracking delay of the very first training block (116 $\pm$ 41 ms) was already smaller than the average tracking delay of the control experiment (160 $\pm$ 20 ms). This was due to the rapid decrease of the tracking delay during the first half-cycle of the initial training trial. In the periodic presentation mode of TR1 (training and test blocks) the tracking delay stayed small during the periodic continuation trials (only those were analyzed). This points to a fast buildup of predictive command components in manual tracking and is consistent with smooth pursuit which is known to develop predictive components even before the end of the first period of target motion [27, 28]. Differences in the tracking delay between the continuation trials of the training and of the control most likely result from differences in these fast developing predictive components between periodic and non-periodic presentation modes.

Thus, the smaller tracking delay on the familiar than on the unfamiliar trajectories during the test blocks may be due to the differences between periodic and non-periodic presentation modes as well as to the previous experience with TR1 during the training block. In both cases the differences reflect the use of prior knowledge about the trajectory used for movement control.

### 2.4.2 Adjustments of the synergy index

During the test blocks, not only the tracking delay, but also the synergy index was smaller while tracking TR1, when more prior knowledge about the trajectory was available. A decrease of the synergy index during learning was also observed with a bi-manual pointing

DOI:10.15774/PPKE.ITK.2017.002



task [19]. In this experiment, similar to the current study, an improvement in precision was associated with a greater decline of $V_{\mathrm{UCM}}$ than of $V_{\mathrm{ORT}}$.

Other adjustments of motor synergies were previously observed in a series of studies investigating task variables that changed suddenly after they were kept stationary for some time. Immediately before such a change, an anticipatory drop in the motor synergies was found for finger forces [29, 30, 31] as well as for muscle modes during posture control [32]. These synergy adjustments are viewed as feed-forward support for destabilizing a task variable in preparation for its sudden change, and demonstrate fundamental differences in the motor control of static posture and movement. In contrast, the synergy adjustment reported in the current study represents differences between movement execution modes.

Latash et al. [14] suggested that a decrease of the synergy index may be a specific outcome of learning. This is also an attractive hypothesis to explain the presented experimental data since a decrease of the synergy index was observed concurrent with the acquisition of prior knowledge about the target trajectory. However, so far it is not clear whether this change in the structure of the variance indicates a change of the underlying movement goal (the cost function which is minimized). Alternatively, changes of the structure of the effector variance may be a direct consequence of using prior knowledge to achieve the same movement goal. To discuss this question, it is necessary to concern the respective predictions of motor control theory.

Two different basic mechanisms have been proposed to explain synergy indices larger than one: specific minimization of task-relevant motor noise by optimal feedback control [33, 21], or planning noise systematically corrected for task errors [34]. Concerning planning noise, the model of van Beers et al. [34] predicts that the synergy index converges to a limit determined by the size of error corrections in the task-relevant and task-irrelevant subspace (see Appendix). If the system has no information about the task relevance of different noise components, explorations of the motor space should be homogeneous and all noise components should be corrected by the same percentage. The synergy index expected in that case is one. If learning does not only concern the prior knowledge of the 2D trajectory but also the knowledge about the task relevance of movement plan corrections, one might expect the synergy index to increase with learning. However, our experimental findings show the opposite result. Therefore, explaining this finding by

DOI:10.15774/PPKE.ITK.2017.002



changes on the planning level means assuming a specific change in the strategy of selective error correction. From this perspective, this change reflects a true modification of the movement goal and not just an automatic consequence of the acquisition of knowledge about the trajectory or about the task structure.

Concerning the effects of optimal feedback control on the synergy index, it is less clear whether the reported results also suggest a change in the underlying movement goal (whose achievement is represented by the task error). The next section focuses on this question.

### 2.4.3 The coice of cost function to model changing prior knowledge about the trajectory

The predictions of optimal feedback control for the synergy index during tracking with more or less precise prior knowledge about the trajectory have not been well investigated. It is discussed in the following whether this theory predicts the observed decrease of the synergy index with larger prior knowledge. For simplicity the discussion is restricted to the case of linear dynamic systems and signal-independent noise.

According to Todorov and Jordan [21] optimal feedback control reduces the variance of effectors in the task-relevant dimensions and explains the synergy index being larger than one. However, this study did not consider tracking movements but goal-directed movement with endpoint costs. Yüksel et al [35] presented an optimal feedback control law for tracking, developed within the linear-quadratic-gaussian (LQG)-framework [36] but did not analyze the synergy index predicted by such a controller. In the current study the algorithm of Yüksel et al. [35] was applied to a simplistic plant in which a lowpass filtered 2D-position signal was used to stabilize the 2D-system state on a moving 1D-subspace. It resembles manual tracking in that the plant and the motor control signal have a larger DoF than the trajectory. To analyze the dependence of the synergy index predicted by optimal control theory, two different approaches were adopted. The first investigates the effects of changing noise parameters while keeping the cost function unchanged, whereas the second presents a specific change of the cost function explaining the experimental results. The details of these simulations are shown in the Appendix.

DOI:10.15774/PPKE.ITK.2017.002



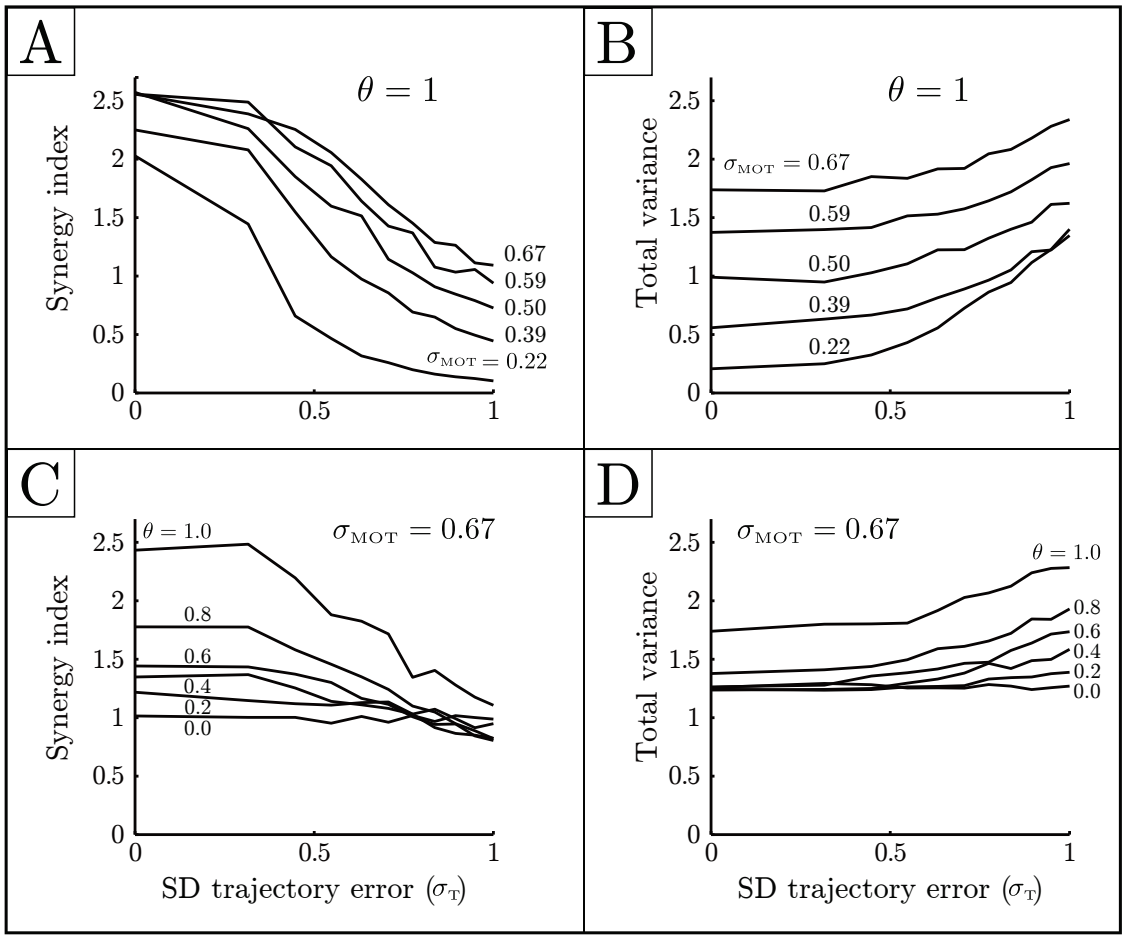


FIGURE 2.6: **Simulation result of optimal feedback, applied to a simplistic tracking system with a 2D-effector space and a 1D-target trajectory.** The task error is computed as a weighted average between a tracking error in the target space (weight $\theta$) and in the effector space (weight $(1-\theta)$). $\sigma_{\mathrm{T}}$ : standard deviation of the process noise of the additional system state *trajectory error*. $\sigma_{\mathrm{MOT}}$ : standard deviation of the process noise of the 2D effector space (motor noise). A/B: invariant control law ($\theta = 1$) C/D: variable control law ($0 \leq \theta \leq 1$) A/C: Synergy index computed as the ratio between the variance of the 2D-motor states irrelevant and relevant for the tracking error in the 1D-target space. B/D: Total variance of system in the effector space. Results are averages across 100 periods, each simulated with 101 discrete time samples.

### Invariant cost function

Yüksel et al. [35] studied trajectory planning in a task-space which is a linear function (**C**) of the effector space with reduced DoF. They minimized the outcome of a cost function ($\varepsilon^2$) that is the sum of task error and control costs, where the task error term is a quadratic form of the difference between the actual position of the system and the planned trajectory ($\underline{y}_t$), both expressed in the task space as shown in (2.5).

DOI:10.15774/PPKE.ITK.2017.002



$$\varepsilon^2 = \sum_{t=0}^{T} \left\| \mathbf{C}\underline{x}_t - \underline{y}_t \right\|^2 + \left\| \underline{u}_t^T \mathbf{R} \underline{u}_t \right\|^2 \tag{2.5}$$

The vector $\underline{x}_t$ denotes the position of the system in the effector space, i.e. the system state, and $\mathbf{C}\underline{x}_t$ its projection on the task space. The control costs are expressed as a quadratic form of the control signals $\underline{u}_t$. In this section it is assumed that both the cost function and the system dynamics are invariant with respect to prior knowledge of the target trajectory. For optimal feedback control within the LQG-framework, this assumption has the important implication that the control law generating the control signal on the basis of the current state estimate stays invariant as well [37]. To incorporate the target predictability the system analyzed by Yüksel et al. was extended by an additional state representing the difference between the actual and the expected trajectory that is called the *trajectory error*. The *trajectory error* affects the actual task error, it is observed by visual input, but it is not subject to control. The process noise (i.e. the random components of the input driving the system states) of the trajectory error is used to model the uncertainty about the trajectory: It is set to zero to mimic complete prior knowledge about the trajectory, and is increased with increasing uncertainty (see Appendix).

Simulations of this system with invariant control law (Figure 2.6 A) show that its synergy index decreases with increasing process noise of the trajectory error. This is because the state changes induced by the optimal control law are constrained to the task-relevant subspace. Consequently, increasing process noise of the trajectory error leads to an increase of task-relevant variance and a decrease of the synergy index. Thus, optimal feedback predicts the opposite effect to that observed experimentally. In contrast, decreasing motor noise leads to a decrease of the synergy index and of the total variance (Figure 2.6 A/B) as reported by Todorov and Jordan [21]. This is caused by decreased optimal estimator gain of the motor states induced by decreasing motor noise (Wiener filter, see Appendix). Even though this change is in the direction of the change observed in the current experiment, no further support was found for a direct link between a systematic decrease of peripheral motor noise of the arm and the acquisition of prior knowledge about the trajectory.

DOI:10.15774/PPKE.ITK.2017.002



**Variable cost function**

The basic assumption of the last section of an invariant cost function seems to be incompatible with the results. The next question to discuss is which dependencies of the cost function on target predictability can explain the data. Changes of the cost function might concern control costs or the task error.

First, the control costs are considered. Increase of motor control costs leads in general to a decrease of the control signals, but will not systematically affect the constraint of the induced state changes on the task-relevant subspace. Consequently, increasing control costs results in a reduction of the control gain, which reduces the synergy index and increases the total variance in the effector space. Increasing the motor cost of our simplistic motor plant by a factor of 10 resulted in a decrease of the synergy index from 2.5 to 1.7, and an increase of the total variance from 1.7 to 1.9. The directions of these changes are again not compatible with the experimentally observed decrease of the synergy index together with a decrease of the total variance.

Finally, changes of the cost function were investigated related to changes of the task error. It was hypothesized that without prior knowledge of the trajectory, the task error is expressed in the coordinates of the low-dimensional, external target space, whereas, with improving prior knowledge, the control strategy changes towards minimization of a task error that reflects the differences between the actual and the planned trajectory in the effector space. This cost function can be expressed as shown in (2.6), where $\mathbf{F}$ denotes the projection of the extended states (including the trajectory error) on the effector space.

$$\varepsilon(\theta)^2 = \sum_{t=0}^{T} \theta \cdot \left\| \mathbf{C}\underline{x}_t - \underline{y}_t \right\|^2 + (1-\theta) \cdot \frac{\dim(\underline{y}_t)}{\dim(\underline{x}_t)} \cdot \left\| \mathbf{F}\left(\underline{x}_t - \underline{x}_t^*\right) \right\|^2 + \left\| \underline{u}_t^T \mathbf{R} \underline{u}_t \right\|^2 \qquad (2.6)$$

The task error is expressed in the external target space for $\theta = 1$, and in the effector space for $\theta = 0$. The planned trajectory $\underline{x}_t^*$ was assumed to be identical with the optimal feedforward solution minimizing $\varepsilon(\theta = 1)^2$. Figure 2.6 C shows that the synergy index converges to 1 as the task error converges towards the tracking error in the effector space ($\theta = 0$). In this case, the total variance (Figure 2.6 D) converges to 1.27. Consistent with

DOI:10.15774/PPKE.ITK.2017.002



the experimental result, both synergy index and total variance decrease with decreasing $\theta$ (Figure 2.6 C/D). It is important to note that this hypothetical change in the task error is not a consequence of the acquired knowledge about the target trajectory but reflects a strategic change of the movement goal (parameterized by the additional parameter $\theta$). This is also reflected by the fact that, for $\theta = 0$, both synergy index and total variance become independent of the uncertainty of the expected trajectory in the target space ($\sigma_T$) because the cost function $\varepsilon(\theta = 0)^2$ is independent of trajectory error (Figure 2.6 C/D).

Experimental support for a strategic change of the movement goal is provided by the observation that the synergy index during periodic presentation of TR1 was larger during the test block ($9.0 \pm 7.6$) compared to the end of the training block ($1.93 \pm 0.98$) immediately before. At the same time such a difference was not observed for the tracking delay (end of training block: $94 \pm 33$ ms; test block $91 \pm 24$ ms). This suggests that the drop of the synergy index is not directly coupled to the efficiency of motor prediction, nor to the amount of available knowledge about the target trajectory. Both reduced synergy index and reduced effector variance seem to be features of a particular movement control mode which characterizes largely automated movements from visually driven tracking movements. For sequential pointing movements, such different movement execution modes were proposed in [38] describing a gradual shift of the movement goal defined in target space to one defined in motor coordinates. The variable cost function proposed here for tracking corresponds to such a gradual shift of the control strategy between a task error defined in target coordinates ($\theta = 1$) or motor coordinates ($\theta = 0$).

### 2.4.4 Conclusion

In summary, the experimental finding shows a smaller synergy index during tracking of familiar compared to unfamiliar trajectories. In contrast, motor control theory predicts that minimizing the tracking error in the target space implies that the synergy index decreases with impaired prior knowledge about the target trajectory. This prediction is independent of whether the synergy index is explained by control mechanisms for the compensation of planning noise or peripheral motor noise. Consequently, the opposite experimental finding suggests that the movement goal (formalized by the cost function, and achieved by the control strategy) differs between tracking of familiar and unfamiliar

DOI:10.15774/PPKE.ITK.2017.002



trajectories. The difference can be characterized by a modification of the task error being minimized. For visually driven tracking of unfamiliar trajectories, the task error seems to be defined in target coordinates, whereas for familiar trajectories it seems to be defined in motor coordinates. This strategic shift between visually driven and automated tracking movements explains the observed decrease of the synergy index on familiar trajectories.

DOI:10.15774/PPKE.ITK.2017.002

# Chapter 3

# A Development Framework for Arm Movement Measurements

Subsections 3.1 and 3.2 in the current chapter are based on the author's article entitled *"A measurement system for wrist movements in biomedical applications"*. [C1]

## 3.1 Background

The area of health assisting technology have been more and more active in the last few years in the field of medical instrumentation, movement rehabilitation, prosthetic devices and fitness accessories, just to name a few. This process resulted in a wider spread of devices addressing different areas on the border of life sciences and engineering, from simple commercial products (e.g. small pulse monitors) to very complex research projects like the Modular Prosthetic Limb[1] and its control interfaces.

Human movement recording – a complementary area to these fields – however, did not show this level of activity. The presumable reason for this is that the measurement methods in motion tracking applications are standardized, tested and validated since decades and manufacturers keep providing high level laboratory systems to meet these requirements. The most widely used measurement systems apply line-of-sight (LoS) methods (optical or ultrasound-based) that require a fixed *marker-sensor* structure. In these cases

[1] http://www.jhuapl.edu/prosthetics/scientists/mpl.asp

DOI:10.15774/PPKE.ITK.2017.002



passive (optical) or active (ultrasound) markers are placed on anatomically relevant locations of the studied subject. Having the markers in place, measurements have to be performed in a specialized laboratory environment where locations and orientations of the sensor elements (i.e. cameras or microphones) are known and invariant (at least across trials). This means that even though the spatial positions of the markers can be determined with good accuracy – especially with optical systems – the possible range of motion will always be constrained by the actual measurement volume covered by the sensors of the system. Although this property is not an issue for many movement analysis scenarios, there are cases when a measurement method allowing unconstrained free space movement would be more beneficial (e.g. various outdoor activities or ergonomic assessment of work environments, to name a few).

Advancements in the field of inertial sensor technology have given rise to new development directions in laboratory-free movement analysis methods. The main difference between LoS and inertial systems is the recorded modality: while LoS methods determine the *spatial locations* of markers based on planar position (optical) or timing (ultrasound) information, inertial sensors give their *orientation* in space by measuring physical quantities acting on them directly. These quantities are linear acceleration and angular velocity in most cases while they are supplemented with magnetic field measurements in more complete setups. To obtain orientation from raw inertial measurements, various sensor fusion algorithms have been developed utilizing Kalman-filters [39, 40], gradient descent methods [41], complementary filters [42, 43] and other techniques [44], most of them being capable for real-time operation in embedded systems. In addition, the recent evolution of chip-scale inertial sensors based on MEMS technology further widened the possibilities of wearable measurement device development by making the core sensing elements available for better integration. While there is no gold standard among fusion algorithms and sensor chips as compromises have to be made in aspects of accuracy, system complexity and computational demand of the fusion algorithm, it can be stated that inertial sensor technology is taking a more and more growing part in human movement measurements (a good example for this progress is Xsens' product portfolio).

A further aspect with regard to biomedical applications is the possibility of recording other modalities, most importantly bioelectric signals like muscle activities (electromyogram, EMG). Measuring EMG is a key aspect to get deeper insight into the dynamics of movements because it gives closer information about muscle activation patterns. While

DOI:10.15774/PPKE.ITK.2017.002



there are many approaches in the literature for individual EMG recording [45, 46, 47, 48], research efforts towards integrated systems utilizing kinematic measurements and muscle activity recording in the same package has only been started recently [49, 50] [2]. As to the best of the author's knowledge, there is only one commercially available EMG+IMU system on the market (TRIGNO$^{TM}$ IM, provided by Delsys from 2016Q3 [3]), that incorporates many small sensor units, each containing a single-channel EMG electrode and an inertial sensor. This system, while seeming to be a good overall solution for whole-body performance assessment in various situations, has some major differences from the system presented in the current chapter, e.g. it may use a proprietary and closed source kinematic model for movement reconstruction that needs to deal with the unique placement of the sensors. In particular, sensor placement seems to be optimized for EMG measurements (i.e. targeting muscles) and not for IMU operation.

The work described in this chapter is an effort towards the development of a research oriented, fully customizable, integrated and wearable measurement system specifically targeting arm movements in order to record and analyze movement patterns for biomedical applications. The system aims to provide a framework for development and testing of inertial sensor based movement measurement and analysis techniques including custom hardware setup, calibration routines, sensor fusion and device control from various platforms. While the complete system design includes EMG measurement capability, this chapter focuses mainly on the practical considerations and implementation of kinematic movement recording.

## 3.2 Measurement device

The concept of the system is depicted in Figure 3.1. The core of the design is a Base Unit responsible for controlling the measurements, collecting sensor data, performing pre-processing tasks and sending and/or storing the output data. The EMG sensors and the Base Unit itself are planned to be integrated into a sensor ring around the lower arm to reduce measurement noise and make usage more practical compared to standard wet-electrode EMG systems using long wires. There are four inertial sensors in the design which are placed (1) on the back of the hand, (2) at the distal end of the lower arm right

[2] In fact, almost the same time as the work presented in this chapter was started.
[3] http://www.delsys.com/products/emg-auxiliary-sensors/trigno-im/

DOI:10.15774/PPKE.ITK.2017.002



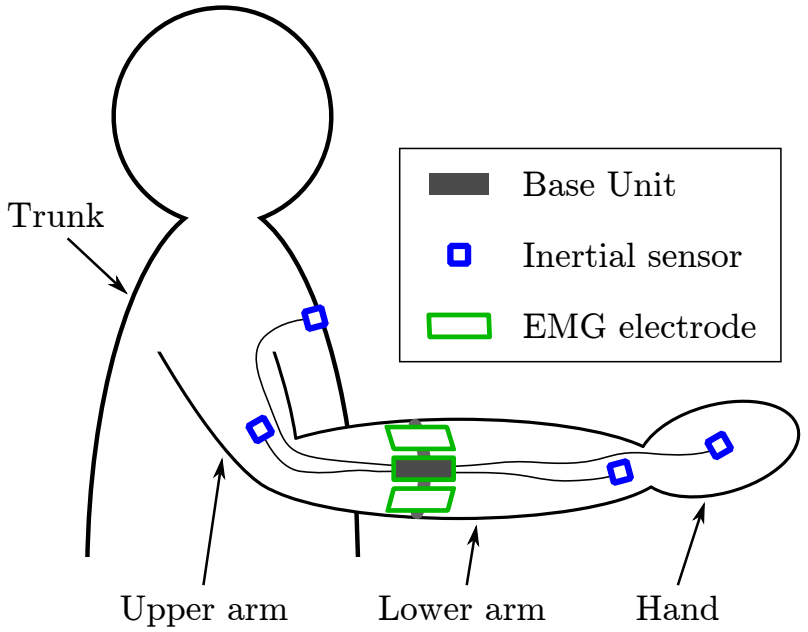


FIGURE 3.1: **Concept drawing of the measurement system.** The Base Unit is planned to be integrated into an EMG sensor ring around the lower arm. Inertial sensors are placed at the distal end of the lower arm right before the wrist joint and on the back of the hand, allowing the measurement of the hand's relative orientation with respect to the lower arm's orientation.

before the wrist joint, (3) on the distal end of the humerus right before the elbow joint and (4) on the lower end of the sternum. This arrangement allows the measurement of arm segment orientations relative to the trunk that is essential for movement analysis and a generic issue in free moving environments (i.e. the subject's trunk is often fixed during laboratory measurements).

The concept and placement of the EMG sensor ring is similar to Thalmic Labs' commercial product called Myo, however the proposed setup addresses a more research oriented scenario providing a higher EMG data rate (the Myo has only 200 Hz EMG output rate based on the company's website[4]) and external inertial sensors for the kinematic measurement of the whole arm. Considering the planned target applications, the main system requirements are the following:

- At least 100 Hz recording of arm orientation.
- Up to 6 channel, 1 kHz recording of forearm surface EMG.
- Real-time data streaming and on-board data storage capability.
- Low power consumption and self-contained, wireless operation.

[4]http://developerblog.myo.com/raw-uncut-drops-today/

DOI:10.15774/PPKE.ITK.2017.002



### 3.2.1 Base unit

The block diagram of the Base Unit is depicted in Figure 3.2. It is designed with an STM32F407VG microcontroller unit (MCU) as its central element which is a high performance ARM Cortex-M4 core running at up to 168 MHz. The MCU has 1 MB flash and 192 kB SRAM, built-in 12-bit 2.4 MSPS ADCs, various serial peripherals (including $I^2C$, SPI and UART), a dedicated SDIO interface for high speed SD card control and a 16-stream DMA controller. In the initial phase of development an STM32F4 Discovery board was used as the central hardware element of the system which provides access to almost all pins of this MCU and a debugger unit in the same package.
Because the system is designed to run from battery, a dedicated power management unit is needed that generates stable 3.3V digital supply from battery input ranging between 1.8V to 5.5V using a buck-boost converter (TPS63001). This allows the selection of battery type best fitting for the actual application from the available supply of various sizes, nominal voltages and capacities. The ±5V analog supply required for the developed EMG frontend (described in Section 3.2.3) is provided using a dual-output charge pump (MAX865) and ultralow-noise positive and negative low-dropout linear regulators (TPS7A4901 and TPS7A3001, respectively). Wireless device control and data streaming is performed by a Bluetooth 2.1 transceiver connected to the MCU through UART interface. Bluetooth 2.1 was chosen instead of Bluetooth 4 low energy (BLE) because based on test measurements the total transmission speed required for raw data streaming including EMG (230.4 kbaud) is higher than the reliable limit of BLE. Although this decision highly affects power consumption of the total system (Bluetooth current consumption: about 40mA ↔ about 8mA), raw data transmission is needed to keep maximal flexibility of the system.

Device firmware was implemented in C using the *Eclipse* IDE (version Kepler) and the *GNU Tools for ARM Embedded Processors* package on an Ubuntu 12.04 LTS system. Device programming and debugging was performed with *OpenOCD* (versoin 0.8.0). As written earlier, the central element of the system is an STM32F407VG ARM Cortex-M4 high performance 32-bit microcontroller from STMicroelectronics. As this device provides enough horsepower to easily handle the flash and RAM overhead of an operating system, the Base Unit's firmware was designed and implemented using *FreeRTOS*$^{TM}$

DOI:10.15774/PPKE.ITK.2017.002



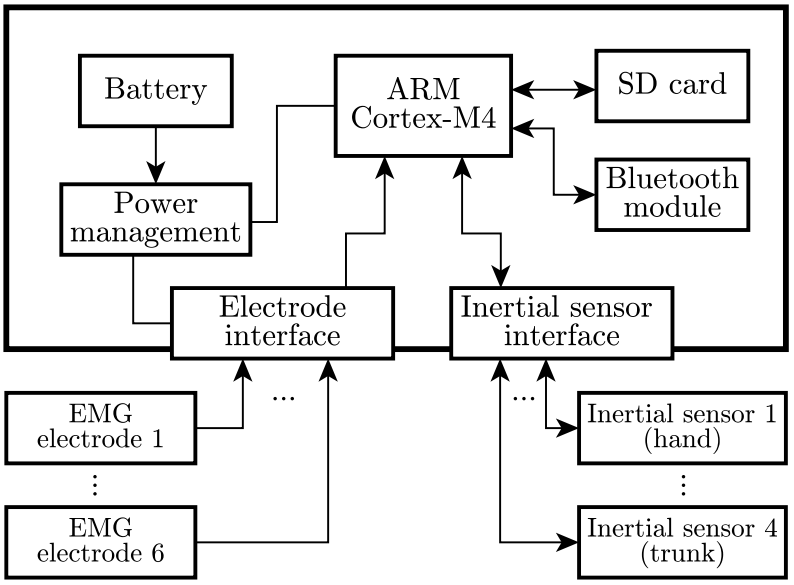


FIGURE 3.2: **Block diagram of the Base Unit.** All measurement and processing is performed by an ARM Cortex-M4 core running at 168 MHz. The Base Unit contains a Power Management Unit to provide the digital and analog supplies, data streaming (Bluetooth) and data storage (SD card) modules. Inertial sensors and EMG electrodes are handled through corresponding hardware interfaces.

[5], a free and industry standard real-time operating system for embedded applications. FreeRTOS' Task, Queue and Semaphore structure allowed designing system functionality at a higher abstraction level and with straightforward execution scheduling. For low-level device driver implementation, ST's *Standard Peripheral Library* for the STM32F4 Discovery kit (version 1.1.0) was used. The MCU's DMA controller was utilized in each scenario where it was applicable to further improve execution parallelism.

### 3.2.2 Inertial sensors

To perform measurements of joint kinematics, single chip 9 degrees-of-freedom MEMS inertial sensors were used (MPU-9250, InvenSense Inc.). Each sensor is a multi-chip module consisting of two dies integrated into a single package. One die houses the 3-axis accelerometer and the 3-axis gyroscope, while the other die houses the 3-axis magnetometer that is internally connected to the chip's control electronics via I2C bus [51]. Individual sensor component properties are shown in Table 3.1.

[5] http://www.freertos.org/

DOI:10.15774/PPKE.ITK.2017.002



TABLE 3.1: **Inertial sensor properties (MPU-9250)**

| **Sensor type** | **Full-scale range** | **Resolution** | **Sampling rate** |
|---|---|---|---|
| Accelerometer | ± 2, 4, 8, 16 G | 16-bit | up to 1kHz |
| Gyroscope | ± 250, 500, 1000, 2000 °/sec | 16-bit | up to 8kHz |
| Magnetometer | ± 4800$\mu T$ | 14 or 16-bit | up to 100Hz |

Based on these properties the sensors are able to provide enough flexibility to measure most common human movement tasks without saturation. Considering the planned movement tasks (low to moderate speed arm movements) the sensors were used as follows:

- Accelerometer: ± 2G, 16-bit, 200Hz
- Gyroscope: ± 500°/sec, 16-bit, 200Hz
- Magnetometer: ± 4800$\mu T$/sec, 16-bit, 100Hz

In addition to the base unit's firmware, a dedicated driver was developed for the inertial sensors to utilize control, calibration and measurement processes over the $I^2C$ bus. Zero motion calibration of the accelerometer and the gyroscope was implemented as part of the sensor initialization process, while hard and soft iron calibration of the magnetometer was realized in the control software as described in Section 3.3.2. However there is a proprietary on-chip Digital Motion Processor (DMP) in each sensor package, it was not used because it is only capable of performing 6-axis (accelerometer + gyroscope) sensor fusion, and there was no publicly available documentation for this unit at the time of development. Instead, a computationally efficient open source orientation filter [41] was used to provide sensor orientations in software using a gradient descent based 9-axis fusion algorithm. The applied method provides direct quaternion output (avoiding the phenomenon of gimbal lock) and is easily able to provide stable output rates above 100 Hz enabling the system to meet and exceed the requirements for orientation data measurement (for details, see Section 3.3.4).

**Sensor synchronization** High precision clock sources are essential for the synchronized operation of any sampled system. This becomes more obvious when multiple sensors are applied and required to keep the designed time frames during device operation. The used inertial sensors apply selectable on-chip internal clock sources that either offer low power consumption in specific cases by using a relaxation oscillator when the

DOI:10.15774/PPKE.ITK.2017.002



gyroscope is off or higher accuracy by using any of the X, Y or Z axis gyroscopes at the expense of increased current need. One issue is however that even when the higher precision option is enabled – because of the lack of any external synchronization or clocking option – small deviations between individual sensors will result in different sample intervals among the used chips.

As this phenomenon is inevitable in this case, the problem was handled by configuring each sensor to signal an interrupt when new measurement data is ready to be read from its internal FIFO storage[6]. At system initialization, a short timing measurement is performed by starting the IMUs in the order they are attached to the base unit. The measurement runs until 10 interrupts from each sensor is noticed and the time intervals between consecutive interrupt pulses are stored in RAM. This is followed by averaging the time interval samples of the individual sensors and reordering the sensor start sequence to assure fastest first, slowest last operation. The synchronization problem is solved by periodically restarting the sensors in every 500 ms to avoid too much timing drift and measurement frame overlapping. This induces an additional issue however: at every restart, the first measurement data in each sensor shows a glitch that would have an impact on the reconstructed orientation thus they are filtered out with a simple linear interpolation technique in the base unit's firmware.

### 3.2.3 EMG frontend

The schematic of the prototype EMG electrode frontend is depicted in Figure 3.3. Considering differential signal recording for each channel ($EMG_-$ and $EMG_+$), the central element of the design is the INA128 instrumentation amplifier (InAmp) with adjustable gain and dual power supply operation. The gain can be set with an external resistor ($R_G$) between 1 and 10,000. The device has wide power supply range (±2.25V to ±18V), low quiescent current ($700\mu A$) and high common-mode rejection ratio even at lower gains ($\sim$ 90dB at f = 1kHz, G = 10V/V) that all make it suitable for portable EMG applications. To allow amplification tuning during development, the gain resistor section of

[6]This means increased system load in general because the host processor needs to handle every new data sample instead of a buffered operation. There is no other option in this case however, because even the sensors have a 512-byte FIFO (able to store up to 28 samples of 9-axis data), only Data Ready and FIFO Overflow interrupts are available, rendering the sensor practically unusable for buffered operation.

DOI:10.15774/PPKE.ITK.2017.002



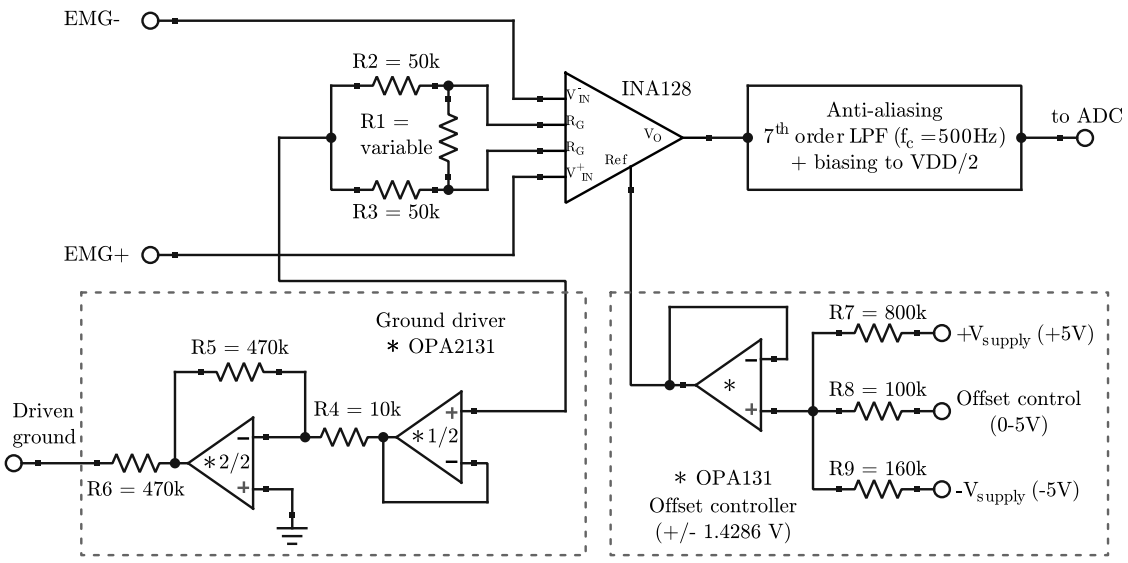


FIGURE 3.3: **Schematic drawing of the active EMG electrode frontend.** The design incorporates differential electrode setup with active body ground driver and an offset controller unit.

the frontend ($R1$ to $R3$) is designed to be adjustable over the full gain range of the amplifier ($R1$ = 100kΩ trimmer, $R_G$ = 50kΩ - 5Ω resulting in 2 - 10,000 V/V gain), while providing the average of input potentials to the active ground driver circuit.

The ground driver feeds back the inverted and amplified average input voltage to the body through a high impedance connection to reduce the amount of common-mode offset and noise (including 50Hz power line noise) in the differential signal. Besides amplification, offset control is an other essential part of signal acquisition and conditioning which is realized by a three resistor ($R7$ to $R9$) passive voltage averaging circuit connected to the InAmp's reference pin through a unity-gain buffer. Using this solution the DC offset of amplifier output can be altered by about ±1.43V which is suitable to compensate offsets introduced by the combination of the small amount of constant inter-electrode voltage (between $EMG_-$ and $EMG_+$) and higher gains. For testing purposes offset control is performed manually using a trimmer resistor on the prototype circuit, however in the final design this functionality will be integrated with one of the MCU's 12-bit DAC's for automated offset compensation even on longer terms (in this case considering 3.3V operation, $R7$ to $R9$ resistor values need to be adjusted).

Figure 3.4 shows test output from the InAmp with the gain kept at a lower level to avoid output saturation and noise amplification. During the measurement a single channel differential recording from the area of wrist flexion muscles on the forearm was performed

DOI:10.15774/PPKE.ITK.2017.002



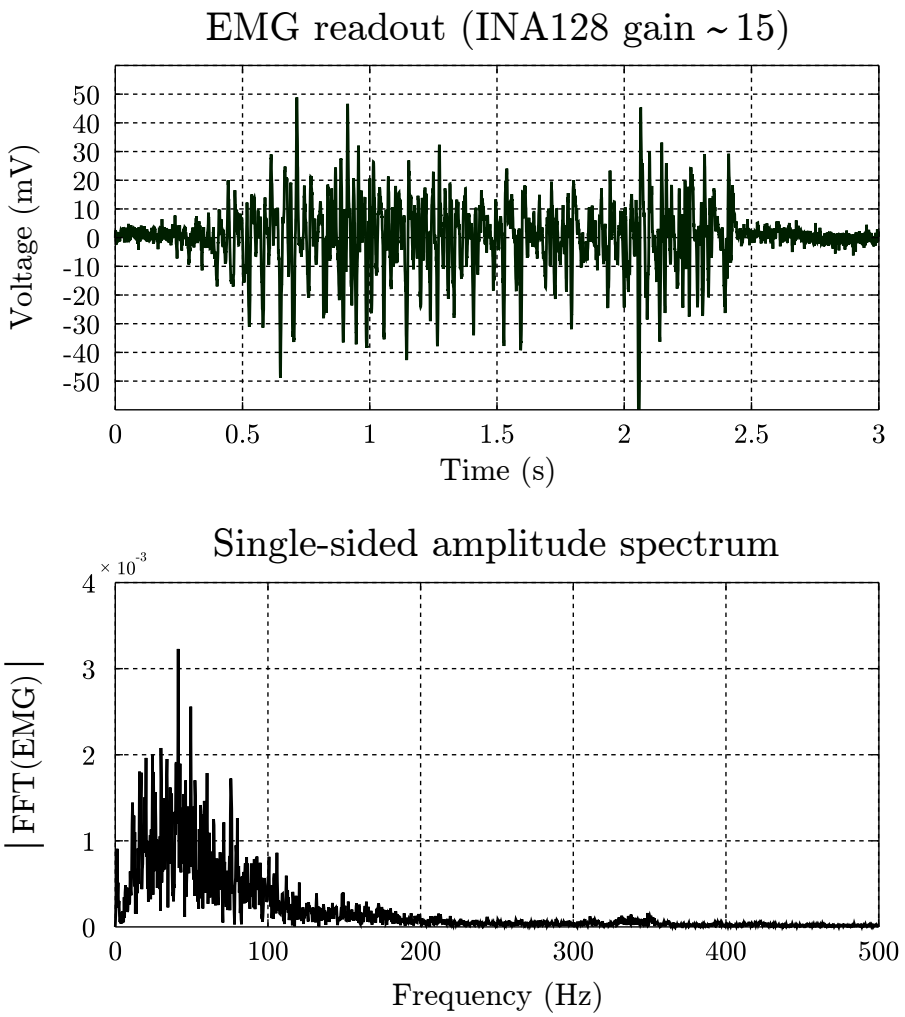


FIGURE 3.4: **Test EMG measurement.** A representative output of the designed EMG frontend circuit during a single channel differential recording from the area of wrist flexion muscles on the forearm.

using standard Ag/AgCl electrodes with an inter-electrode distance of about 3cm (effects from motion artefacts were minimised using self adhesive electrodes). The test circuit was assembled on a breadboard including power isolation (THB 3-0511), ±5V analog supply (using MAX865, TPS7A4901 and TPS7A3001) and the frontend itself and its output was measured directly with an oscilloscope. The recorded data shows satisfying EMG characteristics [52] with a spectral distribution between 10 and 200Hz.

At the current state of the work the final output stage of the frontend is still being developed. This will integrate a high order low-pass filter designed for further signal amplification and anti-aliasing purposes (AAF), and a biasing circuit that offsets the InAmp's zero-symmetrical output to half of the analog reference voltage in order to utilize the whole range of the used ADC. Considering the practical bandwidth of EMG

DOI:10.15774/PPKE.ITK.2017.002



signals of up to 500Hz [52] the AAF's cutoff frequency will be set to this value while the stopband attenuation remains dependent of ADC sampling frequency and resolution. For example, running the integrated 12-bit ADC of the selected MCU at 8 kHz, the AAF must provide -73 dB gain at 4 kHz stopband frequency which is achievable with a $5^{th}$ order linear phase active low-pass filter. If finer resolution is needed for the selected application and a 16-bit ADC is considered, the stopband gain of the AAF change to -97 dB which means a $7^{th}$ order linear phase filter with a 4 kHz stopband frequency. Because of the anti-aliasing stage, the oversampled data can be digitally filtered and decimated in the MCU's firmware using a digital FIR filter to get proper 1kHz output.

Even though the analog circuitry for EMG measurements has not been finalized yet, firmware for EMG recording was implemented by using the MCU's built-in 12-bit SAR-ADC. Considering the realization aspects of the AAF after the analog EMG frontend, a sampling frequency of 8 kHz was implemented for 6 input channels. The sampling is followed by a digital decimating low-pass filter ($F_c = 500$ Hz, decimation factor $= 8$) using ARM's *CMSIS DSP library*. To assure minimal lag and jitter during the measurement, the ADC is controlled in scanning mode using DMA with double buffering. This setup assures aliasing-free recording of EMG signals at 1 kHz output rate for each channel.

### 3.2.4 Hardware prototype

The prototype of the measurement device was constructed using ready-to-use development tools and hardware modules to make the first implementation more convenient and flexible for iterative development as it is shown in Figure 3.5. The depicted device contains the full set of hardware and firmware elements for kinematic movement recording and reconstruction (including sensor measurements and fusion), full firmware support for EMG measurements described above and all supporting hardware except the analog frontend.

## 3.3 Control software

At the first stage of development, the PC-side control application was implemented using MATLAB. The software's main goal is to complete system functionality by providing a control interface and supplementary algorithms for the measurement unit (i.e. graphical

DOI:10.15774/PPKE.ITK.2017.002



interfaces for different device control scenarios, various calibration routines and flexible testing of sensor fusion algorithms and analysis techniques). System functionality is distributed among the base unit and the control software in a way that low level computations having strict timing requirements are performed on the base unit's MCU while higher level tasks that need to be performed only once (like the magnetometer calibration involving eigenvalue / eigenvector calculation) are implemented in the control software and only the relevant results are sent back to the base unit. This scheme allows higher flexibility and efficiency of the total system by keeping the battery powered base unit's power consumption at low level while performing computationally intensive tasks on a higher performance CPU.

Although some of the algorithms described below have been integrated into the base unit's firmware (e.g. the simplified zero motion calibration method or the fusion algorithm), they was implemented and tested first in the control software using the base unit only for streaming raw measurement data. Direct integration of these methods into the device firmware was possible because neither needs visual inspection of the recorded data for reliable operation. However, because of better visualization capabilities of the

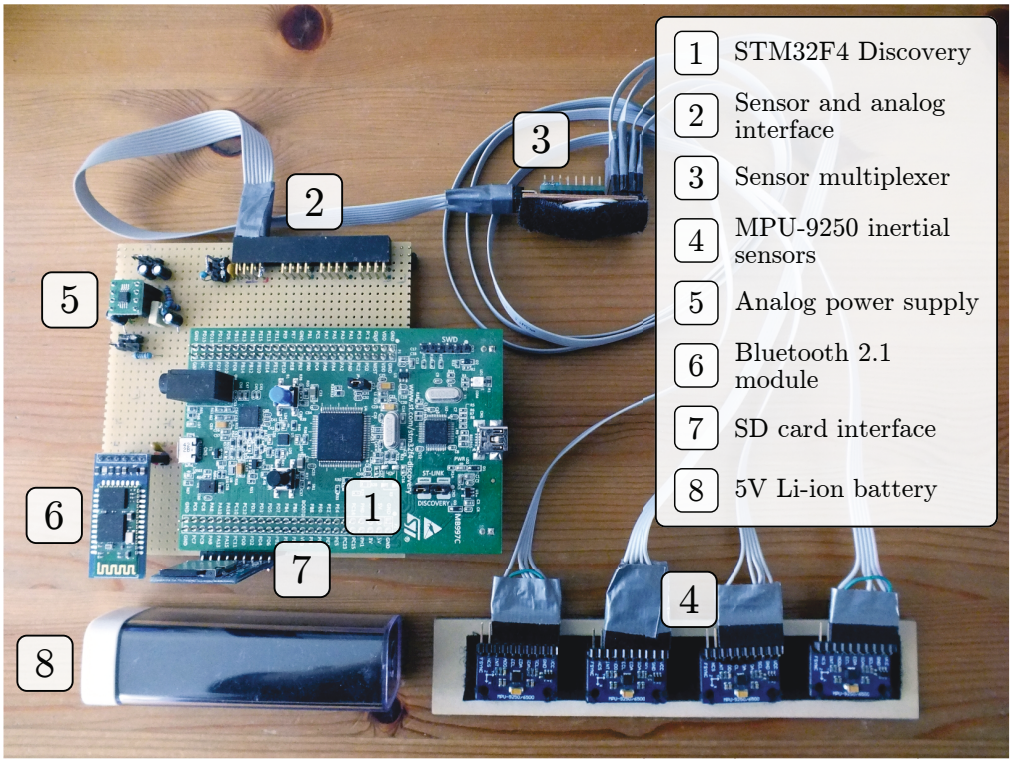


FIGURE 3.5: **The prototype of the measurement device.**

DOI:10.15774/PPKE.ITK.2017.002



control software, these algorithms are also introduced here to give a cleaner overview of the various calibration and analysis methods used in the system.

### 3.3.1 Data visualization

One of the main goals of the control software is to provide device control and data visualization capabilities for different operation scenarios of the measurement device (discussed in section 3.2). In the initial phase of development this was achieved by separate MATLAB scripts, each implementing a specific measurement task and providing the corresponding graphical user interface. The implemented tasks are:

- single sensor data visualization (the actual sensor can be selected),
- magnetometer calibration for all sensors,
- sensor frame alignment calibration for all sensors and
- the actual measurement of arm movements with example applications (e.g. drawing with the arm in 3D or moving virtual objects).

All scripts use the serial port at 230.4 kbaud as the communication interface accessing the measurement device through the serial port profile (SPP) of the Bluetooth classic standard. Recorded data are transmitted in blocks containing 5 samples of each measured feature, resulting in a buffered display of the data at 20 Hz. This value was experimentally determined as the optimal solution to the GUI responsiveness $\leftrightarrow$ MATLAB refresh speed bottleneck. Figure 3.6 shows an example screenshot of a single sensor data visualization task. In this setup, raw sensor data (acceleration, angular velocity and magnetic field) as well as sensor fusion output (quaternions and the resulting 3D orientation) are shown to check device operation in real time.

### 3.3.2 Calibration of raw sensor measurements

Each of the chosen inertial sensors incorporate three individual sub-components for three-axis measurement of linear acceleration (accelerometer), angular velocity (gyroscope) and external magnetic field (magnetometer). When fusion algorithms are used to reconstruct the spatial orientation of the sensor from these physical quantities, proper calibration and inter-component frame alignment are essential to obtain accurate estimations of orientation. While the mutual orthogonality of the individual components' x-, y- and

DOI:10.15774/PPKE.ITK.2017.002



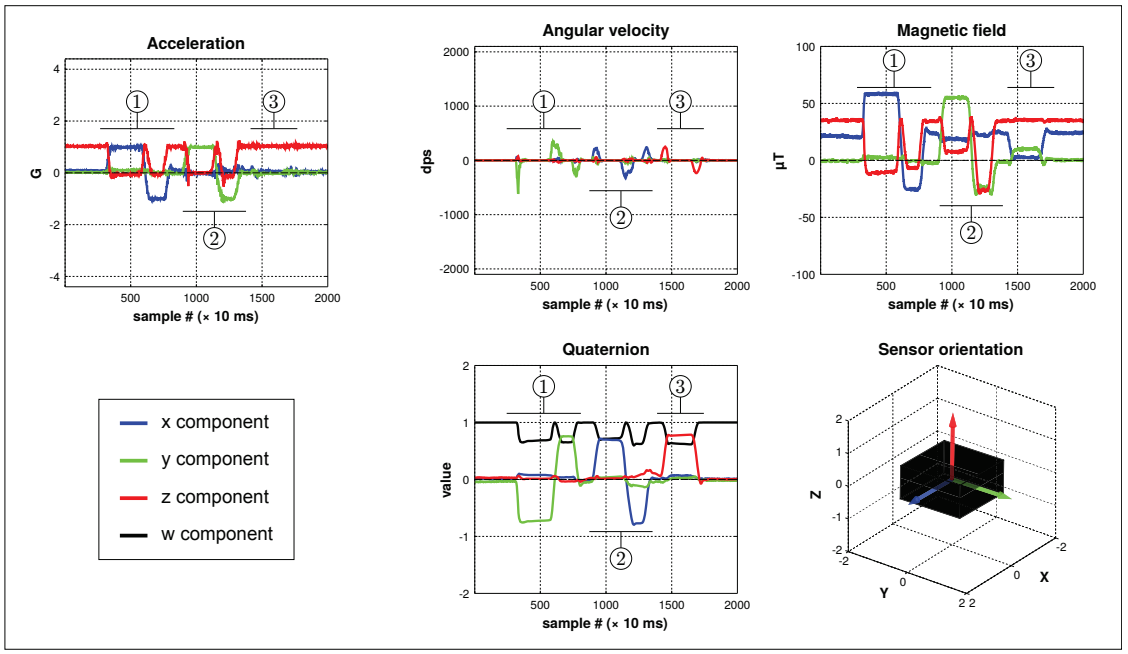


FIGURE 3.6: **Single sensor data visualization.** A representative screenshot with added annotations for the task of showing raw output data and estimated device orientation as the result of the sensor fusion algorithm discussed in section 3.3.4. The recorded movement pattern shown in the figure contained three rotations about each of the main axes of the sensor (x, y and z), each having an estimated peak amplitude of 90°. The starting position was as shown in the actual 3D plot, the z axis being parallel to the direction of gravity. The first movement contained a negative rotation about the sensor's y axis, followed by a positive rotation by about 180° and again a negative to return the sensor to the starting position. The second movement was similar about the x axis with opposite directions. The third movement was a positive and negative rotation about the sensor's z axis.

z-axes along with the alignment of the inter-component local frames is affected by the accuracy of the sensors' manufacturing process, calibration of zero motion offsets and the hard and soft iron errors corrupting magnetometer measurements must be handled in software.

**Zero motion offsets – the naive approach** Independently of the assumption of axis orthogonality and accurate alignment of internal sensor frames, offset errors may occur in the accelerometer and gyroscope readings even in the absence of any motion. To handle this, a simple zero motion calibration protocol was implemented and is performed before a measurement session as follows:

1. The sensors are fixed to a specific holder that keeps them in the same orientation along a straight line. In this reference orientation all sensors' z-axis points against gravity, with the x and y-axis direction being irrelevant.

DOI:10.15774/PPKE.ITK.2017.002



2. While the sensors are in a stationary position, a measurement is performed for 2 seconds. The length of this frame is independent of the actual sampling frequency set for the measurement device.
3. Recorded accelerometer and gyroscope samples are averaged and stored for the correction of further measurements. Before this step however, the z-axis values of the accelerometers are gravity compensated to get the real offset value of that direction.

Table 3.2 shows typical reduction results of zero motion offset errors when the described method is used.

TABLE 3.2: **Zero motion offset values (MPU-9250)**

| **Sensor type** | **Offset compensation** | **x-axis** | **y-axis** | **z-axis** |
|---|---|---|---|---|
| Accelerometer ($G$) | before | 0.0086 ± 0.0022 | 0.0265 ± 0.0021 | 0.0616 ± 0.0034 |
| | after | 0.0003 ± 0.0022 | 0.0006 ± 0.0021 | 0.0275 ± 0.0033 |
| Gyroscope ($^\circ/s$) | before | 2.1148 ± 0.0580 | 1.4948 ± 0.0689 | 3.7185 ± 0.0805 |
| | after | 0.0313 ± 0.0671 | 0.0874 ± 0.0697 | 0.0334 ± 0.0699 |

**Accelerometer calibration** Although Table 3.2 shows satisfactory offset reduction, it should be noted that the naive algorithm may not give accurate results in every case as it assumes zero-g level offsets as the only disturbing factor for sensor outputs. In reality the output of the accelerometer is affected by other factors like axis sensitivity and the sensor's misalignment in the internal frame of the moving body [53]. These factors can be handled by the following extended model of sensor output generation [54].

$$\begin{bmatrix} A_{O_x} \\ A_{O_y} \\ A_{O_z} \end{bmatrix} = \mathbf{M}_{(3\times3)} \begin{bmatrix} \frac{1}{S_x} & 0 & 0 \\ 0 & \frac{1}{S_y} & 0 \\ 0 & 0 & \frac{1}{S_z} \end{bmatrix} \begin{bmatrix} A_{R_x} + O_x \\ A_{R_y} + O_y \\ A_{R_z} + O_z \end{bmatrix} \tag{3.1}$$

where $\mathbf{A_R}$ is the raw accelerometer data, $\mathbf{M}$ is the sensor misalignment matrix, $\mathbf{S}$ is the sensitivity vector, $\mathbf{O}$ contains the offset values and $\mathbf{A_O}$ is the final accelerometer output. Equation 3.1 can be rewritten into the form of

DOI:10.15774/PPKE.ITK.2017.002



$$\begin{aligned} \mathbf{A_O}_{(3\times1)} &= \mathbf{W}_{(3\times3)} \cdot \mathbf{A_R}_{(3\times1)} + \mathbf{v}_{(3\times1)} = \\ &= \begin{bmatrix} \mathbf{W}_{(3\times3)} & \mathbf{v}_{(3\times1)} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{A_R}_{(3\times1)} \\ 1 \end{bmatrix} = \\ &= \mathbf{X}_{(3\times4)} \cdot \mathbf{A_R^{ext}}_{(4\times1)} \end{aligned} \tag{3.2}$$

where $\mathbf{W}$ includes all possible cross-axis interactions and misalignment rotations while $\mathbf{v}$ corresponds to the offset values. The goal of accelerometer calibration is to determine the values of $\mathbf{W}$ and $\mathbf{v}$ that result in $\|\mathbf{A_O}\| = 1$ for any raw measurements in arbitrary positions. To achieve this, accelerometer data in 6 predefined stationary orientations have to be recorded (+1G and -1G for each of the 3 axis) and a general linear model have to be fitted to the recorded data ($\mathbf{X}$ and $\mathbf{A_R^{ext}}$). During this process, recorded and target data corresponding to the stationary orientations ($\mathbf{A_R^{ext}}_{[i]}$ and $\mathbf{A_O}_{[i]}$, $i \in \{1, ..., 6\}$) are pooled together resulting in

$$\mathbf{A_O}_{(3\times n)} = \mathbf{X}_{(3\times4)} \cdot \mathbf{A_R^{ext}}_{(4\times n)} \quad , \tag{3.3}$$

where $n$ is the total number of sample points. From this equation, $\mathbf{X}$ can be determined by using least squares optimization as

$$\mathbf{X} = \mathbf{A_O} \cdot \left( \left[ \left(\mathbf{A_R^{ext}}\right)^T \cdot \mathbf{A_R^{ext}} \right]^{-1} \cdot \left(\mathbf{A_R^{ext}}\right)^T \right). \tag{3.4}$$

Having $\mathbf{X}$ as the optimal least squares fit for the overall recorded data, calibrated accelerometer values can be calculated by using equation 3.2.

While this method takes every aspect of possible accelerometer errors into account, its effectiveness is highly dependent on the accuracy of the used calibration equipment (e.g. a solid calibration box with mutually orthogonal sides and precise sensor mounting). Furthermore, its automation is a bit more problematic than of the naive approach as the stationary orientation segments from the measurement data have to be visually inspected in the current implementation to prevent errors in the calibration data. It has to be mentioned however that this feature could be further developed towards fully automatic operation by detecting zero motion intervals in real-time during the calibration process.

DOI:10.15774/PPKE.ITK.2017.002



**Gyroscope calibration** Similarly to the accelerometer, scale, offset and axis misalignment errors affect MEMS gyroscopes, too. The problem in this case however is that while gravitational acceleration can be used as the reference external signal to calibrate the accelerometer, in a general setup there is no additional equipment available that is able to produce reliable angular velocities required for gyroscope calibration (e.g. a rate table with an optional thermal chamber [55]). Although alternative methods have been proposed that do not require external equipment for gyroscope calibration [56], only the simple zero motion bias compensation method (described earlier) was implemented for the gyroscope components of the used MPU-9250 sensors in the current version of the control software.

**Magnetometer calibration** Just as in the case of the previously shown sensor components, raw output of magnetometers may be affected by error factors of scaling, offset, axis misalignment and nonorthogonality. In addition, surrounding ferromagnetic materials may present further error sources that are either constant or induced with time and orientation dependent properties. The constant term is called *hard iron error* and is generated by ferromagnetic materials in the close vicinity of the sensor component (usually originating from the PCB on which the sensor is mounted) while the induced term is called *soft iron error* and is presented by materials generating their own magnetic field in response to an externally applied field (originating outside of the sensor PCB) [57].

Many algorithms have been proposed to deal with these error factors [57, 58, 59, 60] with the most common approach being based on ellipsoid fitting and its inverse transformation. During this process, raw magnetometer measurements are recorded while the sensor is rotated into various orientations, resulting in a 3D point cloud that represents the actual composition of the surrounding magnetic field around the sensor (the earth's geomagnetic field distorted by hard and soft iron errors). In the error-free case, this point cloud should lie on a sphere centered at the origin of the sensing axes and with a radius of the geomagnetic field's strength at the location of the recording. However, because these magnetic disturbances are inevitable in real world environments, measurements usually show both hard and soft iron errors that cause the recorded point cloud to form a distorted ellipsoid with an offset from the sensing origin. The purpose of the magnetometer calibration algorithm is to find the best fitting ellipsoid to the recorded data and determine the inverse transformation that converts the distorted data points

DOI:10.15774/PPKE.ITK.2017.002



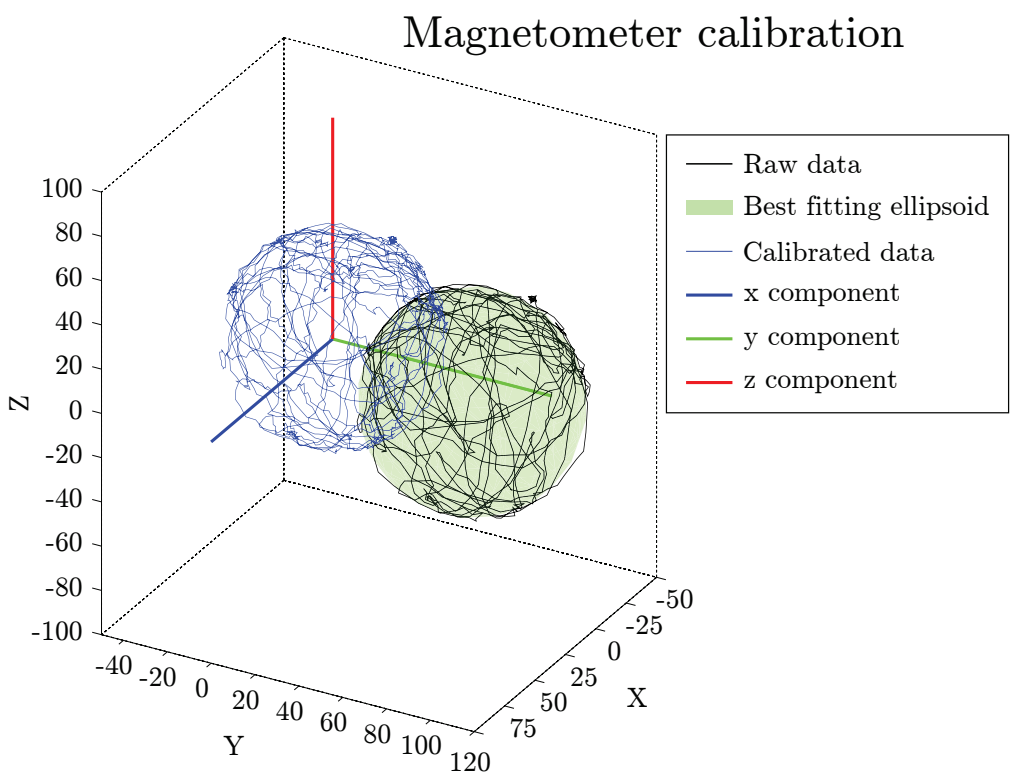


FIGURE 3.7: **Magnetometer calibration.** A representative plot of the magnetometer calibration procedure. Raw measurement data is recorded in various sensor orientations (black line), followed by an ellipsoid fitting step (light green patch). Having the parameters of the best fitting ellipsoid, the hard and soft iron compensation values are calculated and a calibrated dataset is produced (blue line).

into a sphere with a chosen radius (this can be done without loosing information because data fusion algorithms use only the direction (not the magnitude) of the magnetic field for orientation reconstruction).

This method is implemented in the system as an offline process by having a magnetometer calibration mode on the measurement device and a specific part in the control software. During the magnetometer calibration process, the sensors are mounted to a holder that keeps them in the same orientation along a straight line (similarly to the naive approach of zero motion offset compensation). While the holder with the four sensors is rotated into various spatial orientations, the measurement device record raw sensor data at 100 Hz and transmits them to the corresponding script of the control software. As soon as sufficient number of samples are recorded, the measurement is stopped and an ellipsoid is fitted to each set of raw magnetometer data using [61]. Having the parameters of the best fitting ellipsoids, the hard and soft iron compensation values are calculated and sent back to the measurement device to enable real-time magnetometer error correction for later measurements. A representative plot of the procedure is shown in Figure 3.7.

DOI:10.15774/PPKE.ITK.2017.002



It should be noted that using this method to calibrate the magnetometer may not be sufficient in every situation as it assumes that scaling, offset and axis misalignment errors can be handled by the hard and soft iron error compensation algorithm without further corrections. In the currently presented application and with the used sensors this approach seemed to be efficient enough for adequate operation, but care should be taken when applying this calibration method only in other practical setups.

### 3.3.3 Calibration of sensor frame alignments

By having calibrated raw sensor data, the next important step is to determine the misalignment between the sensor frame and the body frame for each sensor package. This is needed because the inertial sensors measure all physical quantities relative to their internal reference frame (with axes directions determined by the edges of the sensor IC package) meaning that if the sensor axes do not coincide with the measured object's own axes (e.g. in the case of an arm segment), the reconstruction of the motion will be incorrect. This phenomenon is inevitable when using inertial sensors for human motion recording because sensor placement on body segments will always be arbitrary and inaccurate as a consequence of soft tissue and skin movement in addition to the fact that internal joint rotation axes of the separate body segments are very hard – if not impossible – to guess accurately in the phase of sensor placement.

To overcome this issue, a calibration algorithm for sensor frame alignment was developed partially based on [59]. Although this algorithm can be used to estimate sensor misalignment in human arm movement recording scenarios, for demonstrative purposes the specific steps will be introduced based on the simple setup shown in Figure 3.8. The calibration steps are the following, where plain text relates to the general case and *italic text* to the example.

1. The inertial sensor is placed arbitrarily on the measured body segment. (*See Figure 3.8.*)
2. Two separate movements are performed in a way that each movement represents a rotation about a principal axis of the measured body segment's internal reference frame (the axes must be different and orthogonal). These calibration movements are performed in an orientation that assures that the actual rotation axis does not coincide with the gravitational force vector. (*Rotations about body $x$ and $y$ axes.*)

DOI:10.15774/PPKE.ITK.2017.002



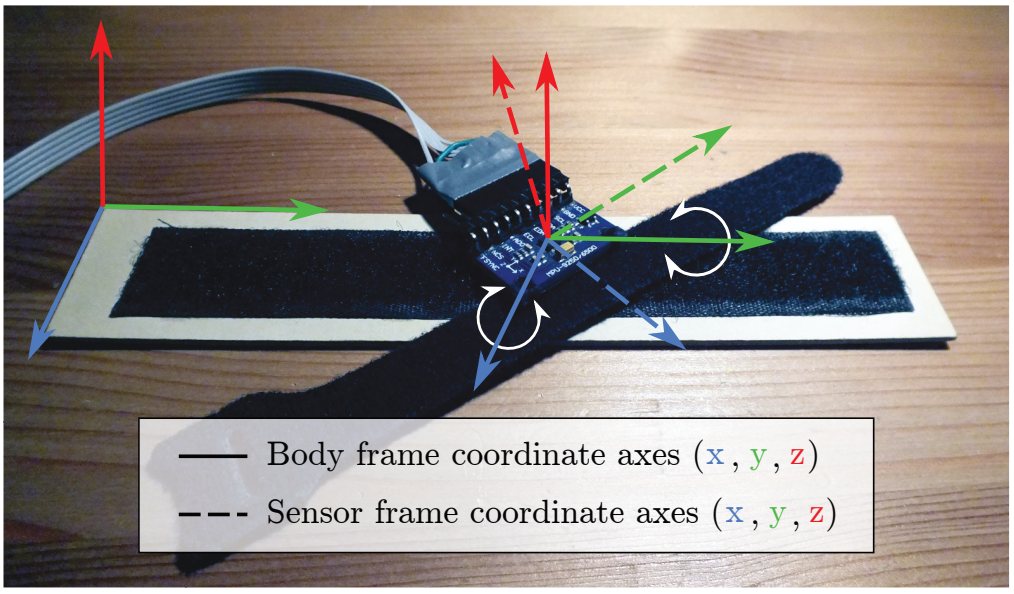


FIGURE 3.8: **Experimental setup for sensor frame alignment calibration of inertial sensors.** In this setup the sensor was intentionally placed with a bad alignment on the measured body to simulate arbitrary sensor placement. The two calibration movements were rotations about the $x$ and $y$ axes of the body's internal reference frame (indicated by white arrows). The task of sensor frame alignment calibration is to find the transformation between the sensor frame and the body frame.

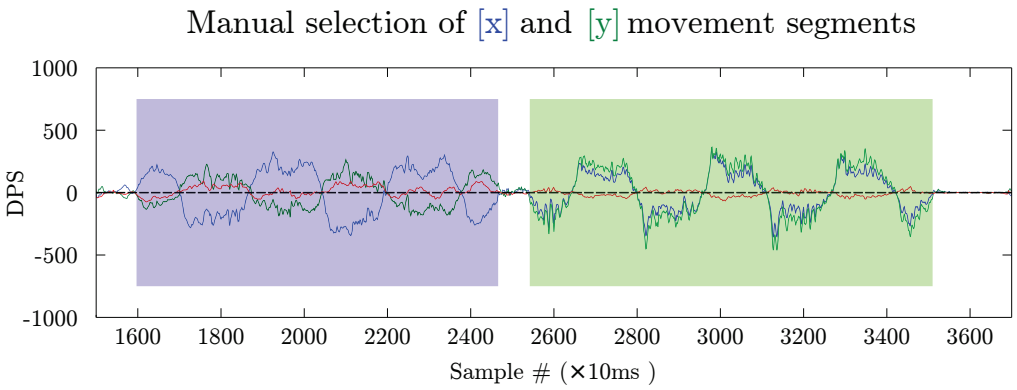


FIGURE 3.9: **Example for manual movement segment selection.**

3. During the rotational movements, raw accelerometer and gyroscope data are recorded and stored on the control PC.
4. After finishing the calibration movements, two raw data sets are manually selected based on visual inspection of angular velocities (gyroscope data) corresponding to the chosen internal reference frame axis (x, y or z). (*See Figure 3.9.*)
5. By having separate data segments representing (mostly) clean rotational movements about specific axes, the recorded accelerometer data can be used to estimate the current axis of rotation. In detail, when rotating the sensor about a specific rotation axis, the gravitational force vector will have a significant impact on the

DOI:10.15774/PPKE.ITK.2017.002



recorded data points. This means that assuming low speed and smooth rotations, the 3D point cloud of the recorded accelerometer data will be spread uniformly around the rotation axis. This phenomenon can be utilized to estimate the direction of the rotation axis as the less variant direction of the point cloud given by Principal Component Analysis (PCA) for example. (*See Figure 3.10.*)

6. After performing the previous step for both measured data segments to estimate the two rotation axes, an orthonormal basis is generated that represents the actual orientation misalignment of the inertial sensor. (*See Figure 3.10.*)
7. The last step of the algorithm is to store the misalignment configuration in quaternion representation for real-time correction of raw sensor data. (*A representative plot of correction results is shown in Figure 3.11.*)

In real human arm movement recording scenarios, the calibration of sensor orientations is performed by specific movements in the trunk and the shoulder, elbow and wrist joints in a way that the corresponding segments move along anatomical axes that are considered to be orthogonal to each other. The movements are 1) trunk flexion and adduction, 2) shoulder abduction and upper arm rotation, 3) elbow flexion and pronation / supination and 4) wrist flexion and deviation, respectively.

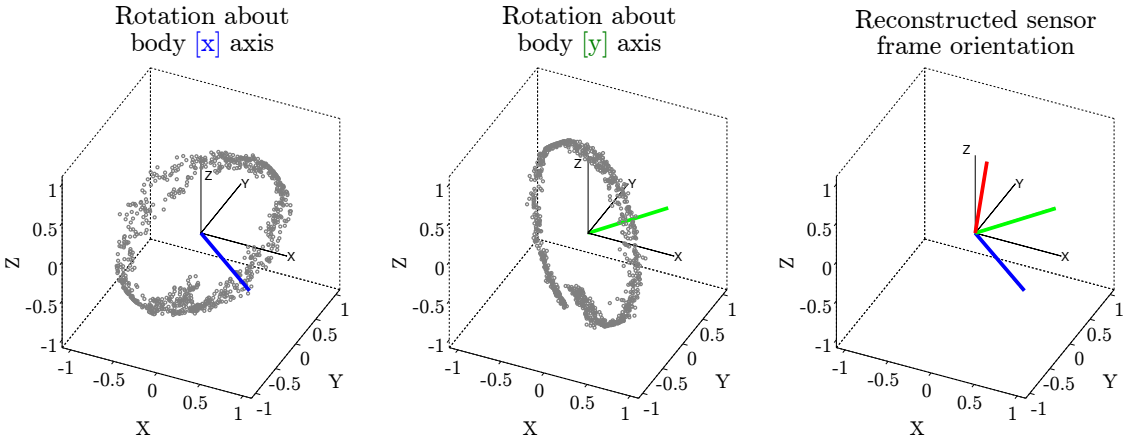


FIGURE 3.10: **Estimation of rotation axes based on accelerometer measurements.**

DOI:10.15774/PPKE.ITK.2017.002



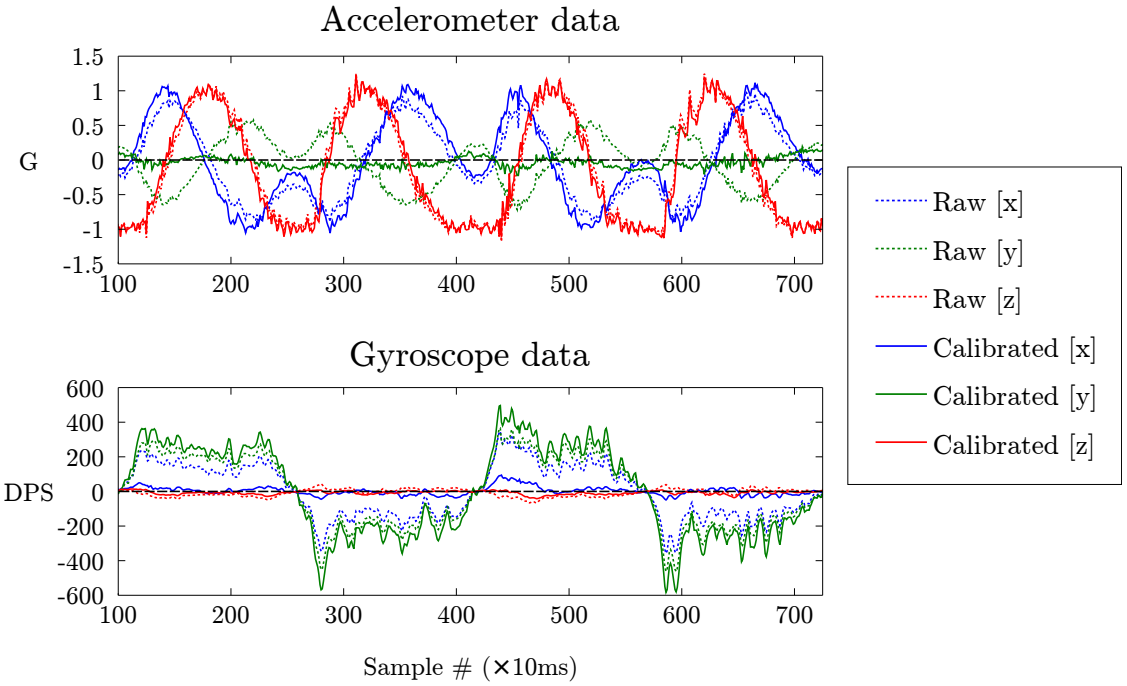


FIGURE 3.11: **Sensor frame alignment results.** The figure shows raw and calibrated recordings from the accelerometer (top) and the gyroscope (bottom). The recorded movement was a periodic rotation of the measured body about the y axis of its internal reference frame (rotation about the green axis in Figure 3.8). In the case of proper sensor placement, y axis data should remain at a low level for the accelerometer while gyroscope readings should show the the biggest amplitude changes with the x and z axes being invariant. As the figure shows, the described calibration method performs acceptably well in the correction of physical sensor misalignment by modifying the recorded raw data according to the estimated orientation.

### 3.3.4 Sensor fusion algorithm

For proper reconstruction of sensor orientations from calibrated raw measurements, a gradient descent based sensor fusion algorithm was used [41] that outputs orientation information in quaternion format, avoiding the problem of gimbal-lock [7] accompanying Euler-angle representation. This approach was selected instead of other methods because its orientation reconstruction accuracy is among the the most widely used algorithms' (e.g. complementary filter or Kalman-filter) [62], it can be implemented in an efficient way (277 scalar operations, 332 bytes of RAM usage [8]) and it is available in open source format. [9] The method estimates the actual orientation in a given time instant by numerically integrating the estimated rate of change of orientation. The algorithm calculates

[7] Gimbal lock is the loss of one degree of freedom in a three-dimensional, three-gimbal mechanism that occurs when the axes of two of the three gimbals are driven into a parallel configuration, "locking" the system into rotation in a degenerate two-dimensional space.

[8] http://x-io.co.uk/res/doc/madgwick_internal_report.pdf

[9] http://x-io.co.uk/open-source-imu-and-ahrs-algorithms/

DOI:10.15774/PPKE.ITK.2017.002



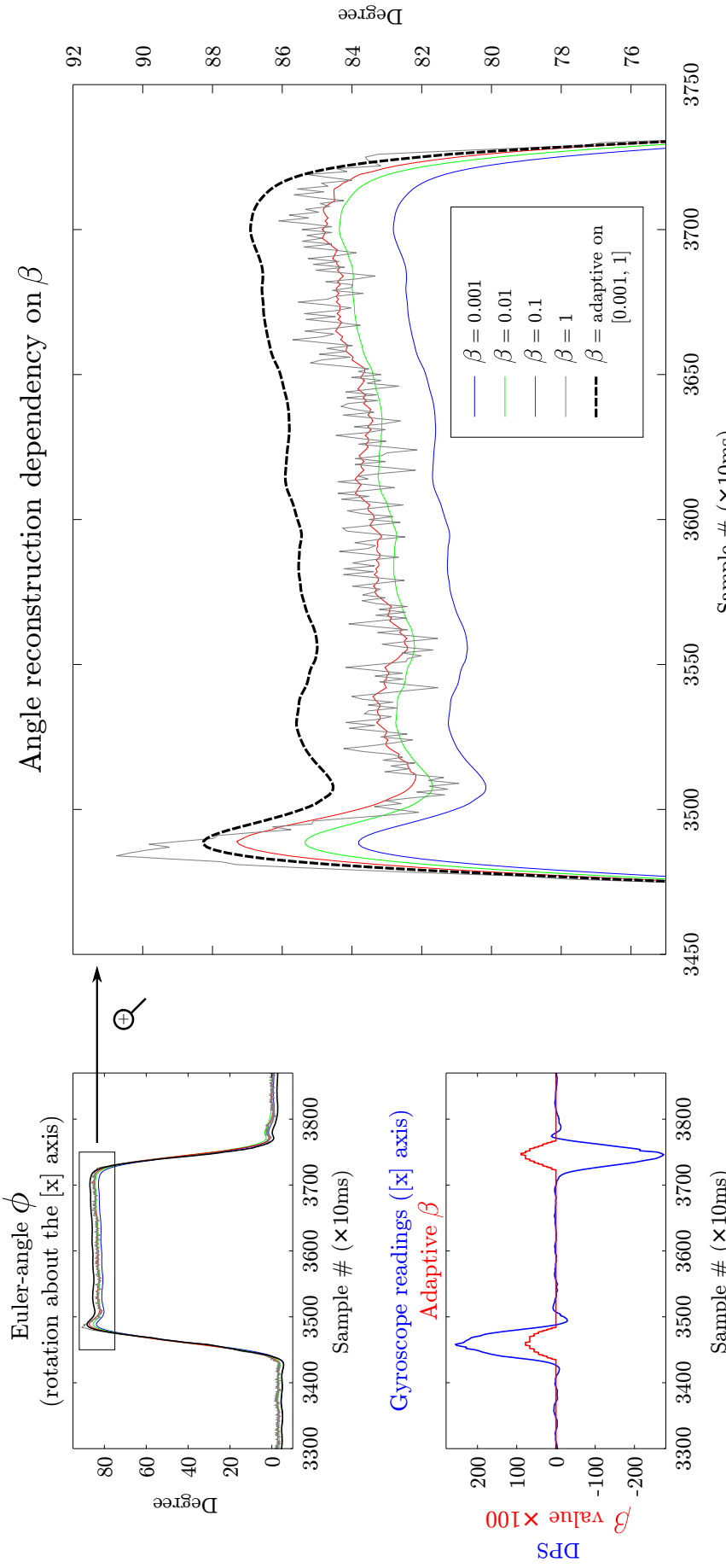


FIGURE 3.12: **The effect of $\beta$ on angle reconstruction.** One of the four inertial sensors was rotated by hand about the x axis of its internal reference frame. The upper subplot on the left shows reconstructed orientation angle values applying different $\beta$-s in Euler-angle representation for better visual interpretation, while the lower subplot shows the corresponding gyroscope readings of the sensor's x axis and the actual $\beta$ values during adaptive operation mode. The subplot on the right shows a magnified view of the reconstructed angles in the interval marked by the rectangle on the upper left plot.

DOI:10.15774/PPKE.ITK.2017.002



this rate of change by using gyroscope data, with the magnitude of the gyroscope error ($\beta$) removed in the direction determined by measurements from the accelerometer and the magnetometer. $\beta$ represents all zero mean gyroscope measurement errors expressed as the magnitude of a quaternion derivative.

In practice, $\beta$ is a tunable filter gain that determines the relative amount of contribution of gyroscope and accelerometer + magnetometer data to the final orientation estimation. As an example, Figure 3.12 shows orientation reconstruction results from a simple representative recording. During the measurement, one of the four inertial sensors was rotated by hand about the x axis of its internal reference frame. Although the intended amount of rotation was 90°, the final angle only approached this value because of performing the task by hand without any reference orientation measurement.

The upper subplot on the left of Figure 3.12 shows the corresponding reconstructed orientation angle with different $\beta$ values in Euler-angle representation for better visual interpretation, while the lower subplot shows the corresponding gyroscope readings of the sensor's x axis and the actual $\beta$ values during adaptive operation mode. The subplot on the right shows a magnified view of the reconstructed angles in the interval marked by the rectangle on the upper left plot. As the figure shows, the value of $\beta$ has a direct impact on reconstruction efficiency in a way that with smaller $\beta$-s the algorithm reacts slower for dynamic movements but keeps static noise at a low level, while larger $\beta$-s allow faster dynamic adaptation at the expense of increased static noise (continuous lines on the right side plot). Based on this (in addition to the statements in the previous paragraphs), $\beta$ can be considered as a filter parameter that controls the passband characteristics of orientation reconstruction (with low and high $\beta$ values corresponding to LPF and HPF operation, respectively).

While the original paper [41] uses a fixed value for $\beta$, it also suggests that applying dynamic values for this parameter may be beneficial in many applications to handle short and long term measurement errors (e.g. accelerations due to motion, distortions in the local magnetic field or gyroscope bias drift). As an example for this, a simple adaptive case was implemented that changes the value of $\beta$ based on the absolute value of the actual angular velocity to deal with both dynamic and static phases of the movement. The lower subplot of Figure 3.12 shows the adaptive changes in the the value of $\beta$ with

DOI:10.15774/PPKE.ITK.2017.002



respect to the corresponding gyroscope measurements, while the dashed line on the right subplot shows the resulted angle values.

While the presented approach seems to solve the dynamic $\leftrightarrow$ static error trade-off for this specific case, it should be noted that without available reference orientation measurements everything shown in the figure should be considered just as demonstration of the concept. Considering real arm movement measurement scenarios, the optimal adaptive control of $\beta$ may be determined by using data from high precision reference measurements (e.g. simultaneous recording with an optical system).

### 3.3.5 Anatomical joint angle reconstruction from sensor orientations

After applying the calibration methods and fine tuning the sensor fusion algorithm presented in the previous sections, one important step still needs to be performed in order to utilize the measurement system in human arm movement analysis. The purpose of all processing steps introduced so far was to measure and reconstruct the spatial orientations of human body segments with wearable inertial sensors as accurately as possible. Although information about body segment orientations in a selected global reference frame allows visualization and kinematic analysis of the movements already (if segment lengths are known, but these can be measured by hand for example), this representation lacks the direct connection with anatomical joint angle definitions, making the findings of the kinematic analysis hard to interpret from the aspects of the movement control system.

The state of the art solution to this problem is to use model-based analysis tools (like SIMM [63] or OpenSim [64]) to "fit" an anatomically reasonable model to the recorded data, resulting in joint angle outputs that generated the actual movement. While this approach has its benefits, because the corresponding workflow has been developed to analyze marker based optical motion capture data, there are some additional processing step necessary to transform segment orientations determined by the presented measurement system to marker information directly usable by these tools. In order to analyze movements with this approach correctly, the generic kinematic model has to be scaled first to match the actual subject's anatomical sizes (making accurate segment length measurements of very high importance). Following this step, the joint angle configuration of the model that gives the best fit for the recorded marker positions is calculated

DOI:10.15774/PPKE.ITK.2017.002



by an optimization algorithm in each time instant. While this approach gives anatomically relevant kinematic results, it has relatively high computational need because of the optimization step, rendering the method unable for real-time operation.

While this is not a problem in most clinical movement analysis scenarios for which the model-based approach was developed, it leaves room for improvement in directions where real-time anatomically relevant kinematic information could be used. As an effort to make contributions to this field, Chapter 4 introduces an algorithm that is capable for real-time reconstruction of anatomical joint angles defined in a widely used upper limb kinematic model without the need of model scaling or computationally intensive optimization methods.

### 3.3.6 Standalone software version

The implementation of the tasks described in the previous sections in separate script files allowed modular testing and validation of the applied methods with targeted data visualization capabilities, however redundancy in the codebase started to develop as new scripts have been implemented along the process. To overcome this issue and to make system usability independent of MATLAB, the re-implementation of the core functionality has been started in Python using the Kivy framework[10] that was created to aid the development of cross-platform multitouch applications. With this approach, one unified codebase may be used to target desktop and mobile (Android and iOS) platforms in the future, allowing the application of the system in various environments.

## 3.4 Conclusion

In this chapter the design and implementation details of a biomedical measurement device were shown. The system is able to record the kinematic state of the human arm by using inertial sensors and to record the surface electrical activities of forearm muscles. Various calibration and data filtering algorithm have been introduced with representative cases presented along the process. Further work includes finalization and integration of EMG frontend design, PCB design and manufacturing of the device and data validity testing with reference measurements using a medical grade optical system.

[10] https://kivy.org/#home

DOI:10.15774/PPKE.ITK.2017.002

# Chapter 4

# Inverse Kinematics for Inertial Sensors

The current chapter is based on the author's articlce entitled *"Real-time inverse kinematics for the upper limb: a model-based algorithm using segment orientations"*. [J2]

## 4.1 Background

Quantitative movement analysis is a key concept in understanding processes of the human movement system. Evolved, high precision measurement devices have advanced research activity in movement rehabilitation [65, 66, 2, 67], performance analysis of athletes [3] and general understanding of the motor system [68, 4, J1, 69] during the last decades by making movement pattern comparison possible. This advancement was further accelerated by model-based analysis approaches that enabled explicit characterization of the studied movement patterns [70, 71, 72, 73, 74, 5].

In addition to accurate measurement methods discussed in the previous chapter, proper evaluation of the recorded motion is an other key building block of human movement analysis. Although various geometric approaches have been developed to describe movement kinematics, the need for standardization of kinematic (and kinetic) analysis of human movements have led to the development of model based tools like SIMM [63] and OpenSim [64] among others. These software packages provide biomechanical models and analysis pipelines to perform various processing steps e.g. model editing, scaling,

DOI:10.15774/PPKE.ITK.2017.002



kinematic and dynamic calculations. Using these tools, obtaining useful movement properties may become a standard process that produces outputs directly comparable across movement tasks and studies.

For the purposes of the current study OpenSim was chosen as the reference model-based movement analysis tool because it uses a mature multibody dynamics engine (Simbody), it can handle SIMM's model format, it provides various APIs (MATLAB, Java, Python) for integration with custom software and it is free and open source with a growing community behind. For kinematic analysis, OpenSim uses the "standard" offline *measurement-scaling-inverse kinematics* pipeline where the actual biomechanical model (single limb to full body) is fitted to measurement data. During this process, positions of virtual markers placed on specific model segments are fitted to experimentally recorded marker positions of the subject with the same arrangement. Scaling is important to generate subject-specific model instances while inverse kinematics (IK) is performed to extract model-defined anatomical joint angles that produced the movement. OpenSim uses a text-based structured XML model format that contains all information needed for the biomechanical description of the human body (bodies, kinematic constraints and forces (i.e. muscles)) that are accessible through API calls, too.

Complex measurement and analysis of upper limb movements including kinematics and muscle activities is an exciting and growing subfield of human movement analysis [75, 76, 77, C2, 78] that promises better understanding of control patterns during specific movements, and as an example benefit may – on the longer term – advance control techniques currently applied to arm and hand prostheses. This process however needs tighter integration of kinematic measurement and reconstruction (from raw data to anatomical joint angles) as the time and computational overhead of the offline *measurement-scaling-inverse kinematics* scheme gives a bottleneck in applications where real-time analysis of the control patterns with respect to the actual kinematics would be beneficial.

The main goal of this chapter is to extend the measurement and analysis workflow of human arm movements with a method that allows accurate and real-time calculation of anatomical joint angles for a widely used SIMM/OpenSim upper limb model in cases when inertial sensors are used for movement recording. For this purpose a custom kinematic algorithm is introduced that utilizes orientation information of arm segments to

DOI:10.15774/PPKE.ITK.2017.002



perform joint angle reconstruction. Accuracy and execution times of the proposed algorithm are validated against OpenSim's IK method on various platforms.

## 4.2 Methods

### 4.2.1 Upper limb model

To analyze arm kinematics with OpenSim, the most complete model available was chosen known as the Stanford VA Upper Limb Model [79]. It is freely available as part of the Simtk project [80] in SIMM model format [81] that can be imported directly into OpenSim. The model is based on experimental data, includes 15 degrees of freedom and 50 muscle compartments and enables the evaluation of kinematics, muscle-tendon lengths, moment arms, muscle forces and joint moments in an anatomically reasonable setup (conforming to the ISB recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion [82]). After importing, the structure of the model follows OpenSim's convention including bodies connected with joints, rotational and translational kinematic constraints and forces defining muscle paths and attributes. The 15 degrees of freedom define the kinematics of the shoulder(3), elbow(2), wrist(2), index finger(4) and thumb(4). As the current work focuses on kinematics of the shoulder, elbow and wrist joints only, any muscles and kinematics of the index finger and the thumb will not be taken into account further in this chapter. The seven degrees of freedom that define the kinematic state of the whole arm excluding the fingers are *elevation plane, thoracohumeral (elevation) angle* and *axial rotation* for the shoulder, *elbow flexion* and *forearm rotation* for the elbow and *deviation* and *flexion* for the wrist.

The model represents the upper limb as a linked kinematic chain of bodies, each having a parent body, a location in the parent's frame and a joint describing the possible relative motion of the child with respect to the parent. The three-dimensional posture of the arm is generated by consecutive rotations of bodies determined by the actual angle values (joint coordinates) in proximal to distal order. As the movement of the shoulder girdle (clavicle, scapula and humerus) is complex and cannot be measured directly in most cases, the model implements regression equations that vary only with the *thoracohumeral angle* to determine the position of the shoulder joint with respect to the thorax.

DOI:10.15774/PPKE.ITK.2017.002



### Orientation from joint coordinates

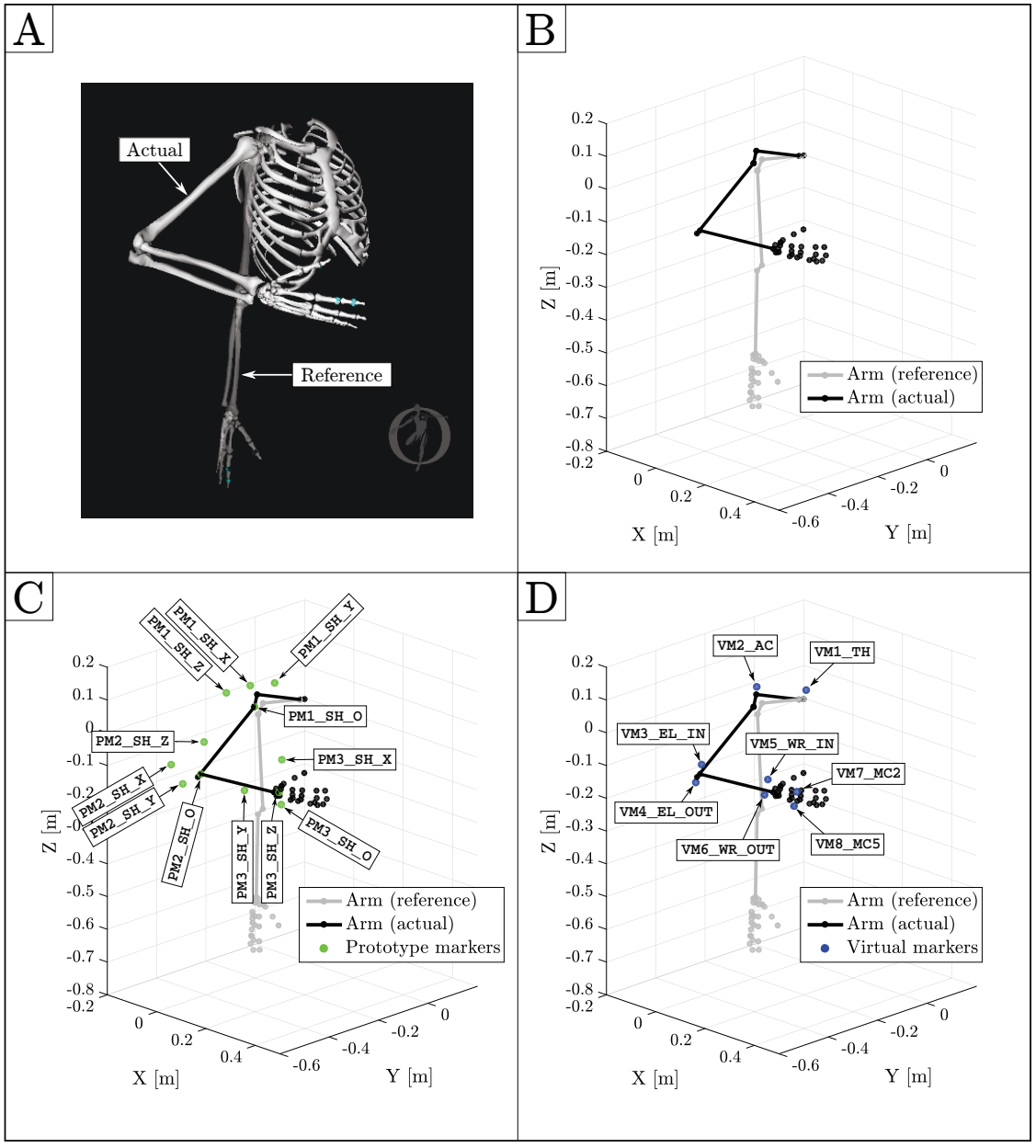


FIGURE 4.1: **Representations of the used upper limb model with reference poses and markers. (A)** Screenshot taken from OpenSim while displaying the used full arm model. The reference configuration is shown as a shaded overlay on an actual example configuration determined by the joint angle vector $[\theta_{\texttt{elv}} = 0°, \theta_{\texttt{sh\_elv}} = 63°, \theta_{\texttt{sh\_rot}} = 15°, \theta_{\texttt{el\_flex}} = 95°, \theta_{\texttt{pro\_sup}} = -60°, \theta_{\texttt{dev\_c}} = 0°, \theta_{\texttt{flex\_c}} = 20°]$. **(B)** Representation of the model's exported structure in MATLAB producing the same actual configuration as in sub-figure (A) using the developed forward kinematics function (functionally equivalent to OpenSim's version). **(C)** Locations of prototype markers that are solely used to the reconstruction of model-defined anatomical joint angles with the proposed algorithm. **(D)** Locations of virtual markers that are used for the algorithm validation process by serving as inputs to OpenSim's inverse kinematics tool directly.

DOI:10.15774/PPKE.ITK.2017.002



The reference orientation of the model (all joint coordinate values equal 0°) occurs when all of the following conditions are true [79] (for a visual reference, see Figure 4.1/A):

1. The shaft of the humerus is parallel to the vertical axis of the thorax.
2. In case of shoulder elevation, the humerus moves in the plane of shoulder abduction.
3. In case of elbow flexion, the forearm moves in the sagittal plane.
4. The hand is in the sagittal plane.
5. The third metacarpal bone in aligned with the long axis of the forearm.

After detailed investigation of the XML file containing all parameters of the selected model, the numerical algorithm with exact rotation axes and order was reproduced that generates arm orientation from actual joint angle values. During this process the position of the shoulder joint is calculated first from the *thoracohumeral angle*. This is followed by four consecutive rotations in the shoulder joint in the order of *elevation plane*, *elevation angle*, *-elevation plane* and *axial rotation*, where the rotation axes of *elevation plane* and *axial rotation* overlap in the reference arm orientation. *Elbow flexion* occurs in the humeroulnar joint while *forearm rotation* takes place in the radioulnar joint. Motion of the wrist is distributed among the proximal and distal rows of carpal bones by having two rotations for each row (four in total) both depending on *flexion* and *deviation* values. For visual reference of the rotations defined in the model, see Figures 4.2 and 4.3.

### Markers, scaling and inverse kinematics

To evaluate subject motion with OpenSim, model parameters have to be adjusted to experimental data. As of OpenSim's latest version at the time of writing (version 3.3), this can be achieved by using marker based motion capture data and virtual markers located on the model at approximately the same places as the experimental markers are located on the subject. This setup allows automatic subject specific scaling of the model [83] and calculation of anatomical joint coordinates (inverse kinematics) during the measured movement using weighted least squares optimization [84]. In the inverse kinematics tool, individual marker weights can be user specified and the least squares problem is solved with a quadratic programming solver (convergence criterion: 0.0001, iteration limit: 1000). As the efficiency of both scaling and inverse kinematics is highly dependent on the accuracy of virtual marker locations, marker placement is usually an iterative process until the best fit to experimental data is found.

DOI:10.15774/PPKE.ITK.2017.002



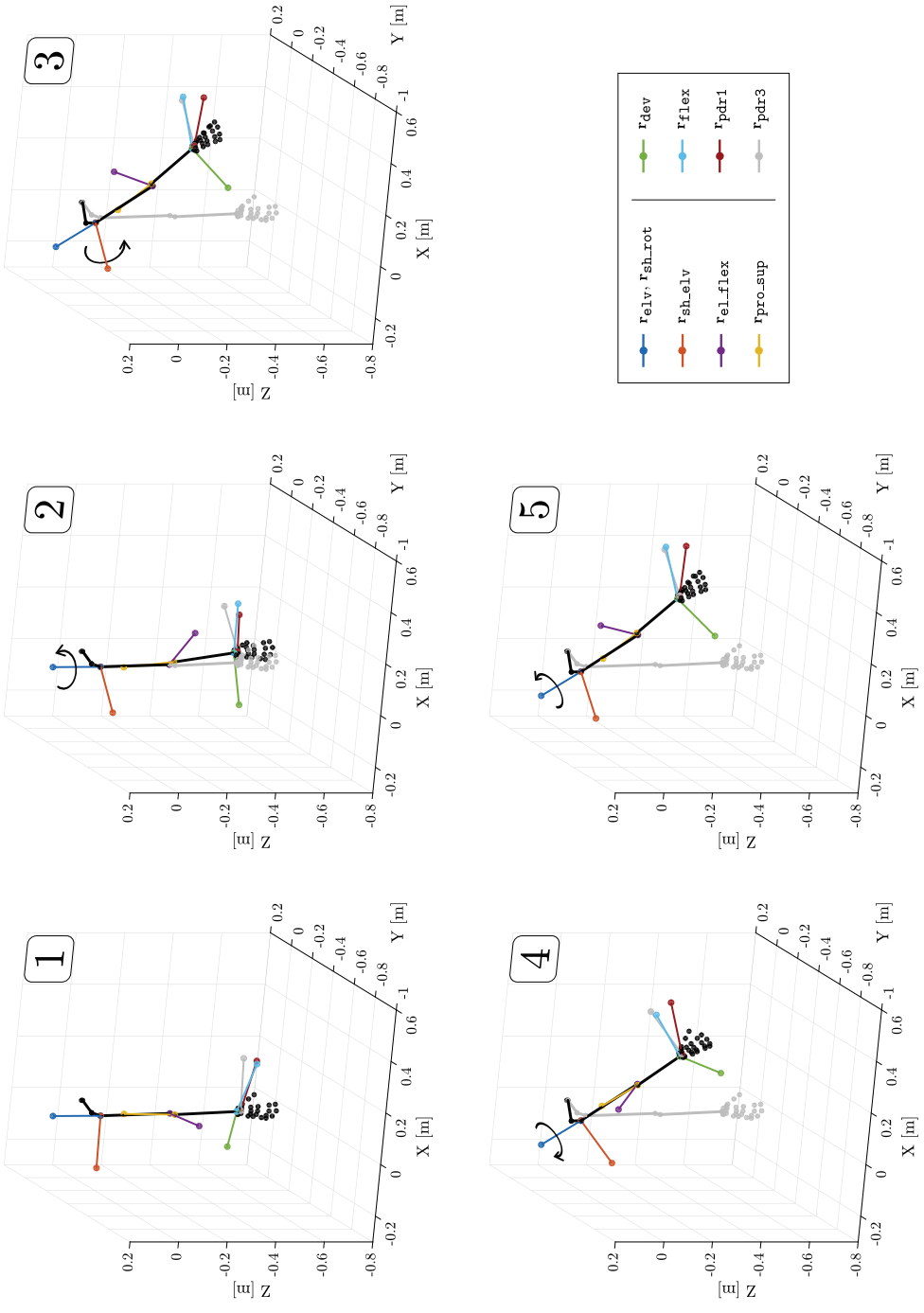


FIGURE 4.2: **Visual representation of model-defined joint angle rotations (part 1).** Colored lines show rotation axes defined in the applied upper limb model. The first subplot shows the resting position. Subplots 2-5 show the four consecutive rotations in the shoulder, based on the XML file defining the model. The actual joint angles shown in the figure are: $\theta_{\texttt{elv}} = 41°$, $\theta_{\texttt{sh\_elv}} = 63°$, $\theta_{\texttt{sh\_rot}} = 36°$, $\theta_{\texttt{el\_flex}} = 67°$, $\theta_{\texttt{pro\_sup}} = -32°$, $\theta_{\texttt{dev}} = 5°$, $\theta_{\texttt{flex}} = -10°$

DOI:10.15774/PPKE.ITK.2017.002



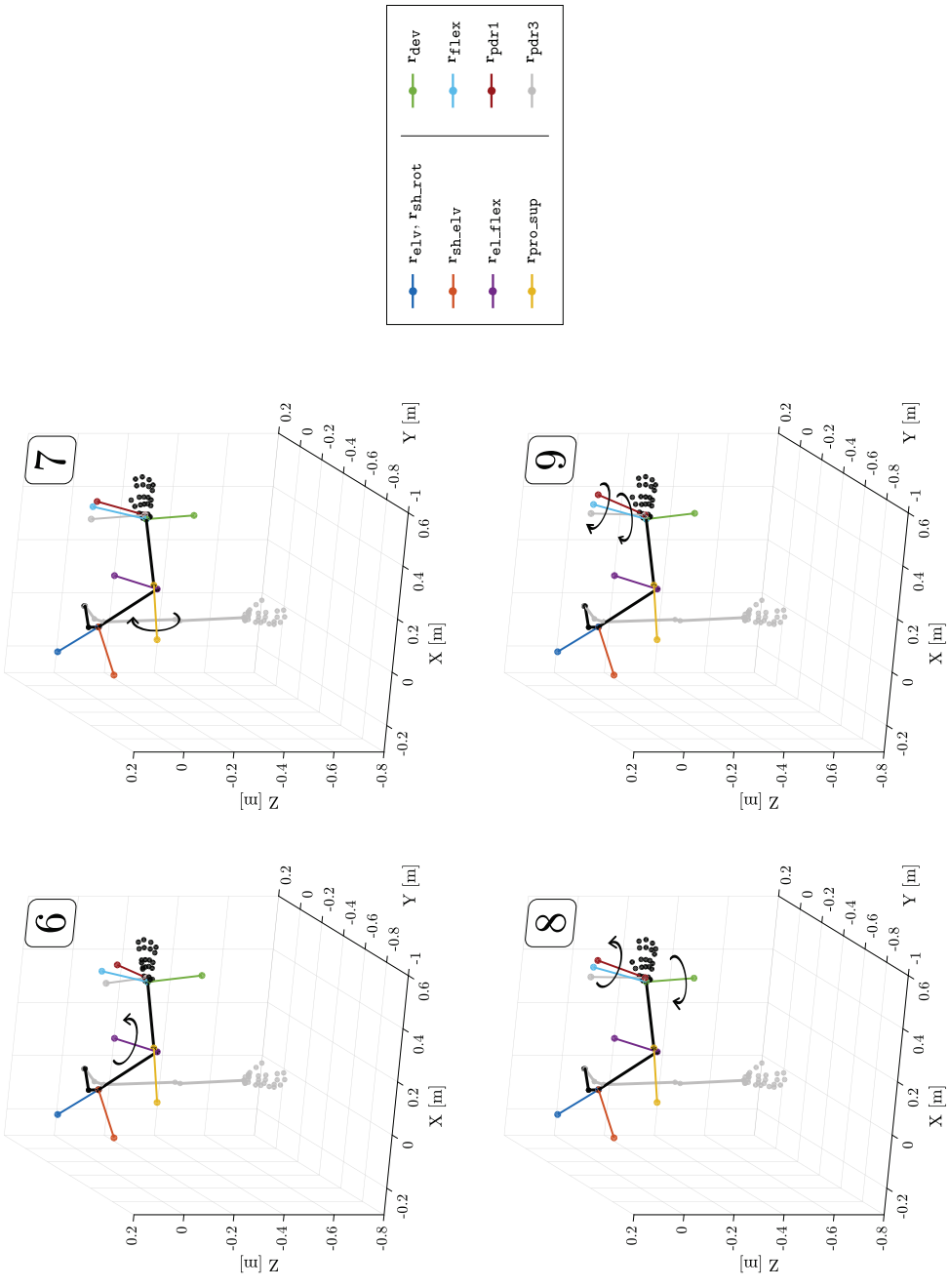


FIGURE 4.3: **Visual representation of model-defined joint angle rotations (part 2).** Colored lines show rotation axes defined in the applied upper limb model. Subplots 6-7 show the rotations of the elbow. Subplots 8-9 show the compound rotations of the wrist that are distributed across two carpal bones to generate the final hand orientation. Rotation order is $\mathbf{r}_{\texttt{dev}}$, $\mathbf{r}_{\texttt{pdr1}}$ $\mathbf{r}_{\texttt{flex}}$ and $\mathbf{r}_{\texttt{pdr3}}$. The actual joint angles shown in the figure are: $\theta_{\texttt{elv}} = 41°$, $\theta_{\texttt{sh\_elv}} = 63°$, $\theta_{\texttt{sh\_rot}} = 36°$, $\theta_{\texttt{el\_flex}} = 67°$, $\theta_{\texttt{pro\_sup}} = -32°$, $\theta_{\texttt{dev}} = 5°$, $\theta_{\texttt{flex}} = -10°$

DOI:10.15774/PPKE.ITK.2017.002



### 4.2.2 Prototype markers

To enable the utilization of the upper limb model with inertial measurements, a prototype marker set was defined. For this purpose, orthonormal bases were formed for each anatomical joint (shoulder, elbow and wrist) and markers were placed at specific locations in these bases to reflect the actual compound rotations among the respective degrees of freedom (for the corresponding mathematical definitions, see appendix B.1).

#### Orthonormal bases

**Shoulder:** The three independent model axes for the shoulder (defined in model bodies *humphant*, *humphant1* and *humerus* that have the same position), collectively denoted as $\mathbf{B}_{sh\_orig}$, were good candidates to form a basis because they are unit length vectors (like all axes in the model) and almost orthogonal to each other (pairwise deviations from right angle are 0.00064°, 0.0029° and 0.0002°). For proper operation of the proposed algorithm however, these axes were orthogonalized using QR decomposition (see appendix B.2) to prevent error accumulation during the calculations. This resulted in the orthonormal basis $\mathbf{B}_{sh\_orth}$.

As a result, rotations in the shoulder can be expressed as elemental rotations of $\mathbf{B}_{sh\_orth}$ with acceptable angle errors due to the pairwise deviations between the original and new basis vectors after orthogonalization (0.000019°, 0.000655° and 0.002925°, respectively).

**Elbow:** As relative orientation of the two rotation axes in the elbow is not close enough to orthogonal and the axes are defined in different parent bodies ($\mathbf{r}_{\texttt{el\_flex}} \rightarrow$ *ulna* and $\mathbf{r}_{\texttt{pro\_sup}} \rightarrow$ *radius*), the orthonormal basis $\mathbf{B}_{\texttt{pro\_sup}}$ and the rotation matrix $\mathbf{R}^{\mathbf{B}_{\texttt{pro\_sup}}}_{\mathbf{r}_{\texttt{el\_flex}}}(\theta_{\texttt{el\_flex}})$ were formed to properly express the compound rotation as the product of an axis-angle and an elementary rotation about the main axis in $\mathbf{B}_{\texttt{pro\_sup}}$. Again, some angle errors are expected while calculating the elbow flexion angle in this basis because $\mathbf{r}_{\texttt{el\_flex}}$ is threated as it would belong to the *radius* body of the model.

**Wrist:** Rotations in the wrist are the most complex among the three anatomical joints. Effects of the two active joint coordinates (*deviation* and *flexion*) are distributed among two model bodies (*lunate* and *capitate)*, each having two nonorthogonal rotations ($\mathbf{r}_{\texttt{dev}}$, $\mathbf{r}_{\texttt{flex}} \rightarrow$ *lunate* and $\mathbf{r}_{\texttt{pdr1}}$, $\mathbf{r}_{\texttt{pdr3}} \rightarrow$ *capitate*) depending on both joint coordinates. To deal with the complexity of this structure, the orthonormal basis $\mathbf{B}_{\texttt{pdr3}}$ and rotation

DOI:10.15774/PPKE.ITK.2017.002



axes $\mathbf{r}_{\mathtt{dev}}^{\mathbf{B}_{\mathtt{pdr3}}}$, $\mathbf{r}_{\mathtt{flex}}^{\mathbf{B}_{\mathtt{pdr3}}}$ and $\mathbf{r}_{\mathtt{pdr1}}^{\mathbf{B}_{\mathtt{pdr3}}}$ were constructed to prepare the calculation of $\theta_{\mathtt{dev}}$ and $\theta_{\mathtt{flex}}$. Using this approach, $\mathbf{r}_{\mathtt{dev}}$ and $\mathbf{r}_{\mathtt{flex}}$ are threated as if they would belong to the *capitate* body of the model.

### Marker placement

In order to add virtual markers to any OpenSim model, the parent body and the location within the parent's frame have to be defined for each marker. Having the orthonormal bases from the previous section ($\mathbf{B}_{sh\_orth}$, $\mathbf{B}_{\mathtt{pro\_sup}}$ and $\mathbf{B}_{\mathtt{pdr3}}$), 12 prototype markers were placed on the model as follows (for reference, see Figure 4.1/C):

- Four markers were placed into each orthonormal basis having one at the origin of the actual basis ([0 0 0] in its parent body) and one in each axis of the basis.
- The markers were named using the convention `PMx_[SH|EL|WR]_[O|X|Y|Z]` where `PM` refers to *prototype marker*, `x` is the serial number of the basis in which the marker is located (1-3), `[SH|EL|WR]` refers to the anatomical joint in which the marker is located and `[O|X|Y|Z]` refers to the marker's location within the actual basis (origin or any of the axes). For example the name of the wrist's origin marker is `PM3_WR_O`.

Because all markers follow their parent bodies' orientation during analyzed movements, coordinates of the difference vectors between the origin markers and the same basis' axis markers reflect the compound rotation matrix in each anatomical joint (in the corresponding basis) at any time instant. To utilize this feature it is crucial that the structure of each joint's marker subset remains consistent during measurements (by keeping the formation of an orthonormal basis), because any deviation in relative marker positions renders the derived compound rotation matrix inaccurate. As a consequence, it is recommended to use arm segment *orientations* to calculate the actual positions of prototype markers instead of measuring them directly. This can be achieved when using optical motion capture devices as segment orientations can be reconstructed with most systems by having at least three markers on each segment, however this is still an offline process. More importantly, utilizing orientation information makes the application of inertial sensors possible and beneficial in this setup as they are used to determine orientation directly. As an additional benefit, the offset-independent nature of orientation information enables subject-independent joint angle reconstruction, rendering the scaling step of

DOI:10.15774/PPKE.ITK.2017.002



the standard inverse kinematics approach unnecessary in the process. Using this feature a real-time inverse kinematics algorithm is introduced in the next section that provides joint coordinate outputs coherent with OpenSim's inverse kinematics tool.

### 4.2.3 Algorithm description

The key point in accelerating the selected upper limb model's inverse kinematics calculation is the model specific determination of prototype marker locations. By constructing representative orthonormal bases in each anatomical joint of interest ($\mathbf{B}_{sh\_orth}$ in the shoulder, $\mathbf{B}_{\texttt{pro\_sup}}$ in the elbow and $\mathbf{B}_{\texttt{pdr3}}$ in the wrist) joint specific rotations can be addressed as elementary or axis-angle rotations in the corresponding bases. Having prototype markers in locations that reflect the actual orientations of these bases gives the possibility to express joint coordinates (rotation angles) in an efficient way, even in closed algebraic form in the shoulder and the elbow. As there was no closed form solution found to calculate angles in the wrist, a numerical algorithm is given for this part of the problem. MATLAB R2015b (Mathworks, Natick, MA, USA) was used for algorithm prototyping and development.

#### Shoulder

Because shoulder prototype markers are placed on the model in a way that they show the actual orientations of the main axes of $\mathbf{B}_{sh\_orth}$, an experimental (numerical) compound rotation matrix can be constructed from their spatial coordinates as shown in (4.1), where each marker position should be considered as a row vector.

$$\widetilde{\mathbf{R}}^{shoulder} = \left( \begin{bmatrix} \texttt{PM1\_SH\_X} - \texttt{PM1\_SH\_O} \\ \texttt{PM1\_SH\_Y} - \texttt{PM1\_SH\_O} \\ \texttt{PM1\_SH\_Z} - \texttt{PM1\_SH\_O} \end{bmatrix} \mathbf{B}_{sh\_orth} \right)^T \tag{4.1}$$

By utilizing the kinematic structure of the shoulder joint (and keeping the assumption that $\widetilde{\mathbf{R}}^{shoulder} = \mathbf{R}^{shoulder}$ as detailed in appendix B.3), estimations of rotation angle values can be calculated as follows:

DOI:10.15774/PPKE.ITK.2017.002



$$\widetilde{\theta}_{\texttt{sh\_elv}} = \arccos\left(\widetilde{\mathbf{R}}^{shoulder}_{(2,2)}\right) \tag{4.2a}$$

$$\widetilde{\theta}_{\texttt{elv}} = \operatorname{atan2}\left(\widetilde{\mathbf{R}}^{shoulder}_{(3,2)}, -\widetilde{\mathbf{R}}^{shoulder}_{(1,2)}\right) \tag{4.2b}$$

$$\widetilde{\theta}_{\texttt{sh\_rot}} = \arcsin\left(\frac{A\widetilde{\mathbf{R}}^{shoulder}_{(2,1)} + B\widetilde{\mathbf{R}}^{shoulder}_{(2,3)}}{A^2 + B^2}\right) \tag{4.2c}$$

$$\begin{aligned} \text{where} \quad A &= \sin(\widetilde{\theta}_{\texttt{elv}})\sin(\widetilde{\theta}_{\texttt{sh\_elv}}) \\ B &= \cos(\widetilde{\theta}_{\texttt{elv}})\sin(\widetilde{\theta}_{\texttt{sh\_elv}}) \end{aligned}$$

Although the formulations in (4.2a) and (4.2c) could be susceptible to modulo $\pi$ and sign errors in general, the allowed angle ranges defined in the model ($\theta_{\texttt{sh\_elv}}$: $0° \rightarrow 180°$, $\theta_{\texttt{sh\_rot}}$: $-90° \rightarrow 20°$) keep these equations safe to use until the experimental data does not force the model outside of these ranges.

### Elbow

Similarly to the shoulder, the experimental compound rotation matrix can be constructed from the actual spatial positions of the elbow's prototype markers. Because the model implements rotations in an incremental way, a reverse rotation of the extracted frame have to be performed in the shoulder's basis to get the correct experimental rotation matrix for the elbow as shown in (4.3).

$$\widetilde{\mathbf{R}}^{elbow} = \left( \begin{bmatrix} \texttt{PM2\_EL\_X} - \texttt{PM2\_EL\_0} \\ \texttt{PM2\_EL\_Y} - \texttt{PM2\_EL\_0} \\ \texttt{PM2\_EL\_Z} - \texttt{PM2\_EL\_0} \end{bmatrix} \left(\mathbf{B}_{sh\_orth}\ \widetilde{\mathbf{R}}^{shoulder}\ \mathbf{B}^T_{sh\_orth}\right)\ \mathbf{B}_{\texttt{pro\_sup}} \right)^T \tag{4.3}$$

DOI:10.15774/PPKE.ITK.2017.002



Having $\widetilde{\mathbf{R}}^{elbow} = \mathbf{R}^{elbow}$ estimations of joint angle values in the elbow can be calculated as (for further details, see appendix B.5):

$$\widetilde{\theta}_{\texttt{el\_flex}} = \arccos\left(\frac{\widetilde{\mathbf{R}}^{elbow}_{(1,1)} - x^2}{1 - x^2}\right) \tag{4.4a}$$

$$\widetilde{\theta}_{\texttt{pro\_sup}} = \arcsin\left(\frac{A\widetilde{\mathbf{R}}^{elbow}_{(1,2)} + B\widetilde{\mathbf{R}}^{elbow}_{(1,3)}}{A^2 + B^2}\right) \tag{4.4b}$$

$$\begin{aligned} \text{where} \quad \mathbf{r}^{\mathbf{B}_{\texttt{pro\_sup}}}_{\texttt{el\_flex}} &= [x\ y\ z] \\ A &= y\sin\left(\widetilde{\theta}_{\texttt{el\_flex}}\right) - xz\left(\cos(\widetilde{\theta}_{\texttt{el\_flex}}) - 1\right) \\ B &= z\sin\left(\widetilde{\theta}_{\texttt{el\_flex}}\right) + xy\left(\cos(\widetilde{\theta}_{\texttt{el\_flex}}) - 1\right) \end{aligned}$$

As in the case of the shoulder, (4.4a) and (4.4b) should be used with care because of modulo $\pi$ and sign errors, but again having sufficient joint angle limits in the model ($\theta_{\texttt{el\_flex}}$: $0° \rightarrow 130°$, $\theta_{\texttt{pro\_sup}}$: $-90° \rightarrow 90°$) application of these formulas is safe until experimental data does not force the model outside of these ranges.

### Wrist

The experimental compound rotation matrix for the wrist can be constructed from the actual spatial positions of its prototype markers. Because of incremental rotations in the model, reverse rotations of the extracted frame have to be performed in the shoulder's and elbow's bases to get the correct experimental rotation matrix as shown in (4.5).

$$\begin{aligned} \widetilde{\mathbf{R}}^{wrist} = \Bigg( \begin{bmatrix} \texttt{PM3\_WR\_X} - \texttt{PM3\_WR\_0} \\ \texttt{PM3\_WR\_Y} - \texttt{PM3\_WR\_0} \\ \texttt{PM3\_WR\_Z} - \texttt{PM3\_WR\_0} \end{bmatrix} \left(\mathbf{B}_{sh\_orth}\ \widetilde{\mathbf{R}}^{shoulder}\ \mathbf{B}^T_{sh\_orth}\right) \cdot \\ \cdot \left(\mathbf{B}_{\texttt{pro\_sup}}\ \widetilde{\mathbf{R}}^{elbow}\ {\mathbf{B}_{\texttt{pro\_sup}}}^T\right) \mathbf{B}_{\texttt{pdr3}} \Bigg)^T \end{aligned} \tag{4.5}$$

DOI:10.15774/PPKE.ITK.2017.002



Although there is no closed form solution to calculate joint angle rotations in the wrist, the flexion angle can be determined as the solution of the following root finding problem (further details and definitions of $a$, $b$, $c$, $x$, $y$ and $z$ can be found in appendix B.7):

$$\begin{aligned}
&\text{Given } F\left(\theta_{\texttt{flex}}, \sigma\right) = -\theta_{\texttt{flex}} + \eta + \sigma \operatorname{atan2}\left(\operatorname{Re}\left(\sqrt{\xi - c^2}\right),\ c\right), \\
&\text{where} \\
&\qquad \begin{aligned} \eta &= \operatorname{atan2}(b, a) \\ \sigma &\in \{-1, 1\} \\ \xi &= a^2 + b^2 \end{aligned} \\
&\text{find } \theta_{\texttt{flex}} = \mu \text{ such that } F\left(\mu, \sigma\right) = 0.
\end{aligned} \tag{4.6}$$

Based on this definition, the following properties hold for $F\left(\theta_{\texttt{flex}}, \sigma\right)$:

1. $(-\theta_{\texttt{flex}} + \eta)$ defines a baseline with constant negative slope for the two possible solutions $F\left(\theta_{\texttt{flex}}, -1\right)$ and $F\left(\theta_{\texttt{flex}}, 1\right)$.
2. Because of the definition of the atan2(y,x) function, the value of $\operatorname{atan2}\left(\sqrt{\xi - c^2},\ c\right)$ will always be positive if $\sqrt{\xi - c^2}$ is real (i.e. $c^2 \leq \xi$). This implies that the two solutions to $F\left(\theta_{\texttt{flex}}, \sigma\right)$ do not cross the baseline but remain "below" $(\sigma = -1)$ and "above" $(\sigma = 1)$ of it for all values of $\theta_{\texttt{flex}}$.

As $c$ depends on the actual compound rotation matrix $\widetilde{\mathbf{R}}^{wrist}$, its value is influenced by both $\theta_{\texttt{dev\_c}}$ and $\theta_{\texttt{flex\_c}}$. As a consequence, there may be wrist configurations where $c^2 > \xi$ for some regions of $\theta_{\texttt{flex}}$, driving $F\left(\theta_{\texttt{flex}}, \sigma\right)$ into a singular state within these regions. To prevent problems arising from this situation during the root finding process, singularity border points for $\theta_{\texttt{flex}}$ can be determined as follows. Let (4.7) as defined in (B.17), only $\theta_{\texttt{flex}}$ changed to $\vartheta$ to denote specific singularity border points.

$$c = x\cos(\vartheta) + y\sin(\vartheta) + z \tag{4.7}$$

Considering (4.6), singularity borders occur at locations where $c^2 = \xi$, resulting in $c_{1,2} = \pm\sqrt{\xi}$. Using these equalities and Euler's formula, $c$ can be rewritten into an exponential

DOI:10.15774/PPKE.ITK.2017.002



form that can be solved for $\vartheta$ resulting in the formulas shown below.

$$c_1 = \sqrt{\xi}: \qquad \vartheta_{1,2}^{(1)} = -\ln\left(\frac{\sqrt{\xi} - z \pm \sqrt{\xi - 2z\sqrt{\xi} - x^2 - y^2 + z^2}}{x - \mathrm{i}y}\right)\mathrm{i} \tag{4.8}$$

$$c_2 = -\sqrt{\xi}: \qquad \vartheta_{1,2}^{(2)} = -\ln\left(-\frac{\sqrt{\xi} + z \pm \sqrt{\xi + 2z\sqrt{\xi} - x^2 - y^2 + z^2}}{x - \mathrm{i}y}\right)\mathrm{i} \tag{4.9}$$

As a result, four separate complex-valued singularity border points can be determined for all wrist configurations. To get a better understanding of the structure of $F\left(\theta_{\texttt{flex}}, \sigma\right)$, the function was visually inspected with an interactive MATLAB script developed for this purpose. The tool allows the simulation of different user-defined wrist configurations through separate sliders for $\theta_{\texttt{dev\_c}}$ and $\theta_{\texttt{flex\_c}}$ while plotting all relevant information about the problem (a representative screenshot is shown in Figure 4.4). Based on manual testing throughout the model-defined ranges for $\theta_{\texttt{dev\_c}}$ and $\theta_{\texttt{flex\_c}}$, the following observations were made:

1. The condition in (B.12) is always met.
2. $\vartheta_{1,2}^{(k)}$ $((k=1) \vee (k=2))$ are separate real numbers if there is a singularity region in the actual wrist configuration for $c_k$. In this case $\theta_{\texttt{flex}} = \vartheta_1^{(k)}$ and $\theta_{\texttt{flex}} = \vartheta_2^{(k)}$ indicate singularity border locations directly.
3. $\vartheta_{1,2}^{(k)}$ $((k=1) \vee (k=2))$ are complex conjugates if there is no singularity region in the actual wrist configuration for $c_k$. In this case $\theta_{\texttt{flex}} = \mathrm{Re}\left(\vartheta_1^{(k)}\right) = \mathrm{Re}\left(\vartheta_2^{(k)}\right)$ indicates the location where the values of $F\left(\theta_{\texttt{flex}}, -1\right)$ and $F\left(\theta_{\texttt{flex}}, 1\right)$ are closest to $(k=1)$ and furthest from $(k=2)$ each other.
4. $\mathrm{Re}\left(\vartheta_{1,2}^{(2)}\right)$ always remain outside the model defined range of $\theta_{\texttt{flex}}$.
5. $\theta_{\texttt{flex}} = \eta$ is the "gluing point" of $F\left(\theta_{\texttt{flex}}, -1\right)$ and $F\left(\theta_{\texttt{flex}}, 1\right)$, meaning that the singularity region for $c_1$ starts to develop from this location, driving $F\left(\theta_{\texttt{flex}}, \sigma\right)$ to "stick" to the baseline.
6. If there is a singularity region for $c_1$, $\mathrm{Re}\left(\vartheta_1^{(1)}\right)$ remains always smaller than $\mu$ where $F\left(\mu, \sigma\right) = 0$.
7. In cases when the singularity region starts to develop (i.e. $\left|\mathrm{Im}\left(\vartheta_1^{(k)}\right)\right|$ is sufficiently small but not zero), two separate roots may appear, but only one being valid.

DOI:10.15774/PPKE.ITK.2017.002



8. $F\left(\theta_{\texttt{flex}}, \sigma\right)$ will have a valid root at $\theta_{\texttt{flex}} = \mu$ if and only if $\sigma = \text{sign}\left(\mu - \eta\right)$.

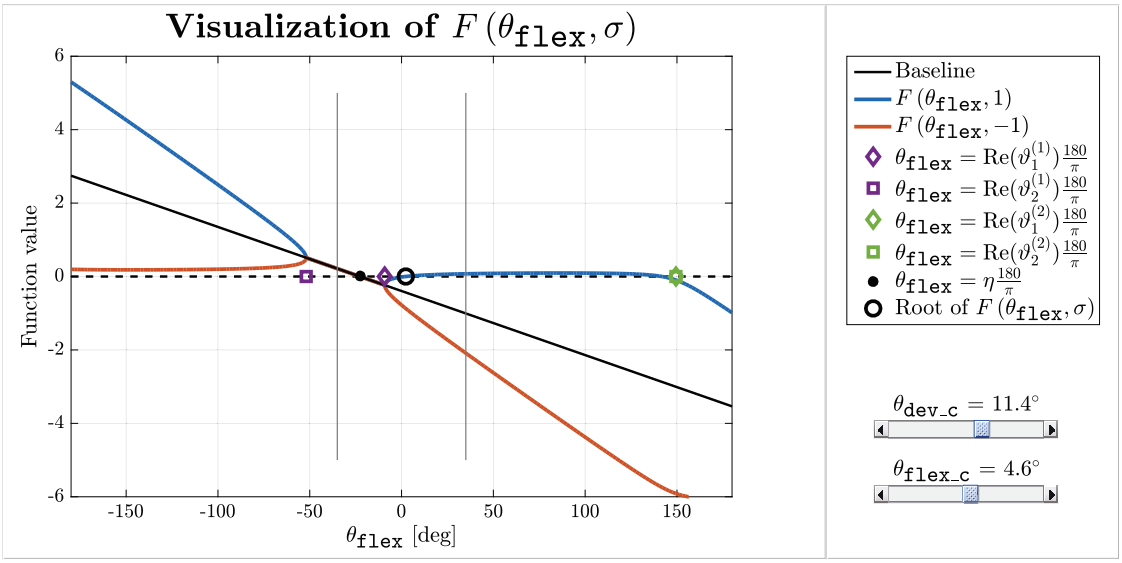


FIGURE 4.4: **Representative screenshot of the tool developed for visual inspection of $F\left(\theta_{\texttt{flex}}, \sigma\right)$ (defined in Equation (4.6)).** The interactive MATLAB script allows simulation of different user-defined wrist configurations through separate sliders for $\theta_{\texttt{dev\_c}}$ and $\theta_{\texttt{flex\_c}}$ while plotting all relevant information about the optimization problem. The two thinner vertical black lines located at $\pm 35°$ indicate model-defined limits for $\theta_{\texttt{flex}}$.

Based on these observations, (4.6) can be solved with Algorithm 1. Having the value of $\theta_{\texttt{flex}}$, $\theta_{\texttt{flex\_c}}$ and $\theta_{\texttt{dev\_c}}$ can be calculated as follows:

$$\widetilde{\theta}_{\texttt{flex\_c}} = 2 * \theta_{\texttt{flex}} \tag{4.10a}$$

$$\widetilde{\theta}_{\texttt{dev\_c}} = \text{atan2}\left(\mathbf{w}_1^T \mathbf{r}_1 \times \mathbf{v}_1, \mathbf{v}_1^T \mathbf{w}_1 - \left(\mathbf{v}_1^T \mathbf{r}_1\right)\left(\mathbf{w}_1^T \mathbf{r}_1\right)\right) \tag{4.10b}$$

where $\mathbf{v}_1 = \exp\left(\theta_{\texttt{flex}} \widehat{\mathbf{r}}_{\texttt{flex}}^{\mathbf{B}_{\texttt{pdr3}}}\right) \mathbf{r}_{\texttt{pdr1}}^{\mathbf{B}_{\texttt{pdr3}}}$ and $\mathbf{w}_1 = \left(\widetilde{\mathbf{R}}^{wrist} \exp\left(-\theta_{\texttt{flex}} \widehat{[1\ 0\ 0]}\right)\right) \mathbf{r}_{\texttt{flex}}^{\mathbf{B}_{\texttt{pdr3}}}$.

Although the computational demand of wrist angle calculations is higher than of the shoulder and the elbow, the algorithm has still higher overall time efficiency than the optimization approach used by OpenSim's Inverse Kinematics tool, as it is shown in the Results section.

DOI:10.15774/PPKE.ITK.2017.002



**Algorithm 1:** Numerical algorithm to calculate $\theta_{\texttt{flex}}$

**Data:** $x, y, z, \eta$ and $\xi$ from (B.17) and (4.6)

**Result:** $\theta_{\texttt{flex}} = \mu$ such that $F(\mu, \sigma) = 0$

calculate $\vartheta_1^{(1)}$ and $\vartheta_2^{(1)}$ from (4.8)

determine the interval $[\zeta_1, \zeta_2]$ in which $F(\mu, \sigma)$ changes sign

**if** $\left|\vartheta_1^{(1)} - \overline{\vartheta}_2^{(1)}\right| < 10^{-10}$ **then**

  $\theta_{\texttt{flex}} \longleftarrow \underset{\mu \in [\zeta_1, \zeta_2]}{\arg\,\text{zero}}\ F(\mu, 1)$

**else if** $Re\left(\vartheta_1^{(1)}\right) > \eta$ **then**

  $\theta_{\texttt{flex}} \longleftarrow \underset{\mu \in [\zeta_1, \zeta_2]}{\arg\,\text{zero}}\ F(\mu, 1)$

**else if** $Re\left(\vartheta_1^{(1)}\right) < \eta$ **then**

  $\theta_{\texttt{flex}} \longleftarrow \underset{\mu \in [\zeta_1, \zeta_2]}{\arg\,\text{zero}}\ F(\mu, -1)$

**else**

  $\theta_{\texttt{flex}} \longleftarrow \eta$

**end**

### 4.2.4 Algorithm validation

Testing and validation of the described algorithm was automated using OpenSim with its Python API and MATLAB. To make direct comparison possible between OpenSim's optimization method and the proposed algorithm, 8 additional virtual markers were placed on the model at locations that are suitable for optical motion capture (e.g. using Vicon) simulating an environment where OpenSim is generally applied. The virtual marker locations are the following (for visual reference, see Figure 4.1/D):

- `VM1_TH` : Thorax marker at the upper end of the sternum.
- `VM2_AC` : Acromio-clavicular joint of the shoulder girdle.
- `VM3_EL_IN` : Medial epicondyle of the humerus.
- `VM4_EL_OUT` : Lateral epicondyle of the humerus.
- `VM5_WR_IN` : Distal head of the radius.
- `VM6_WR_OUT` : Distal head of the ulna.
- `VM7_MC2` : Distal head of the second metacarpal bone.
- `VM8_MC5` : Distal head of the fifth metacarpal bone.

DOI:10.15774/PPKE.ITK.2017.002



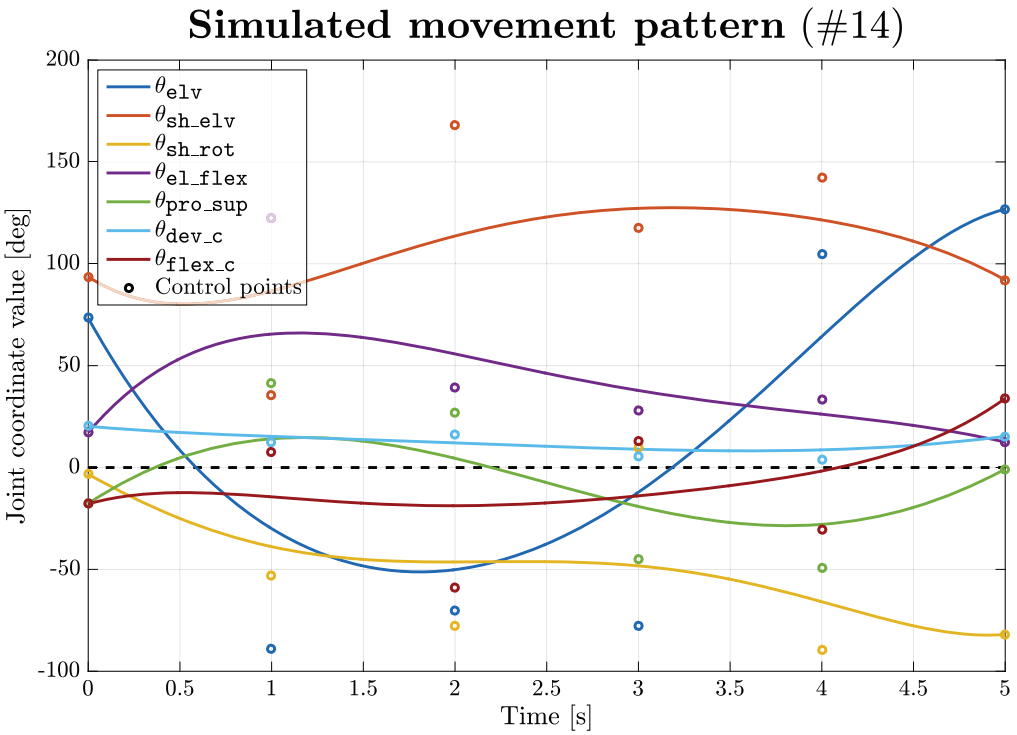


FIGURE 4.5: **Representative simulated movement pattern used for algorithm validation.** Simulated movement patterns were generated to validate the proposed kinematic algorithm. 100 separate pseudo-random joint coordinate trajectories were constructed as $5^{th}$ order Bézier-curves having 5 seconds duration and 100 Hz sampling frequency. `PMx` and `VMx` marker trajectories were calculated with our forward kinematics MATLAB function to generate simualted "measurement" data for the proposed algorithm and OpenSim.

The structure of the upper limb model (including marker positions) was extracted using OpenSim's Python API and saved into a *.mat* file for further processing with MATLAB. A forward kinematics function (functionally equivalent to OpenSim's implementation) was developed in MATLAB to calculate body and marker positions for specific joint coordinate vectors of [$\theta_{\texttt{elv}}$, $\theta_{\texttt{sh\_elv}}$, $\theta_{\texttt{sh\_rot}}$, $\theta_{\texttt{el\_flex}}$, $\theta_{\texttt{pro\_sup}}$, $\theta_{\texttt{dev\_c}}$, $\theta_{\texttt{flex\_c}}$] in the model, enabling the analysis of trajectories for both `PMx` and `VMx` markers from artificially generated movement patterns (Figure 4.1/B-D).

### Simulated movement patterns

To avoid possible problems accompanying experimental measurements, simulated movement patterns were generated to test the performance and validity of the proposed algorithm. 100 separate pseudo-random (random seed = 10) joint coordinate trajectories were constructed in MATLAB having a duration of 5 seconds and a sampling frequency of 100 Hz. The trajectories were generated as $5^{th}$ order Bézier-curves as shown in (4.11)

DOI:10.15774/PPKE.ITK.2017.002



using 6 uniformly distributed control points (0%, 20%, ..., 100%) with randomly chosen values for each joint coordinate from their valid intervals defined in the model. A representative movement pattern is shown in Figure 4.5.

$$\begin{aligned} \mathbf{B}_5(t) &= \sum_{i=0}^{5} \binom{5}{i} t^i \, (1-t)^{5-i} \, \mathbf{P}_i = \\ &= (1-t)^5\mathbf{P}_0 + 5t(1-t)^4\mathbf{P}_1 + 10t^2(1-t)^3\mathbf{P}_2 + \\ &\quad +10t^3(1-t)^2\mathbf{P}_3 + 5t^4(1-t)\mathbf{P}_4 + t^5\mathbf{P}_5 \\ \text{where} \quad & t \in [0,1] \text{ and} \\ & \mathbf{P}_i, i \in \{0,\ldots,5\} \text{ are the control points.} \end{aligned} \tag{4.11}$$

Following this step, forward kinematics was performed for each of the simulated patterns to calculate `PMx` and `VMx` marker trajectories yielding simulated "measurement" data as it would have been recorded during a real movement. The resulted trajectories were then used as inputs to inverse kinematics calculations with OpenSim (`VMx`) and our algorithm (`PMx`) while the corresponding movement patterns served as reference for the outputs of each of the tested methods.

### Inverse kinematics with OpenSim

To speed up the validation process, OpenSim (v3.3) was compiled from source on a Supermicro server having two Intel® Xeon® E5-2695 v3 CPUs (with a total of 56 execution threads) and 64 GB RAM, running Ubuntu Server 14.04.2 LTS operating system. Although the inverse kinematics (IK) algorithm in the used OpenSim version do not utilize multi-core architectures natively, each IK task can be divided into separate subtasks that can run in parallel thanks to the applied optimization method (there is no data dependency between time frames). To utilize this property, a pipeline was developed using MATLAB and Bash to prepare `VMx` marker data and the required files for OpenSim and manage file transfers, multi-threaded IK execution, results collection and evaluation. One important step before performing the IK calculation in OpenSim is subject-specific scaling of the used model and relative weighting of the markers. As only simulated data were used in the current study on an unmodified upper limb model, the scaled model

DOI:10.15774/PPKE.ITK.2017.002



file was identical to the original file during IK execution, while all marker weights were equal.

### Algorithm implementation

The prototype of the proposed algorithm was implemented in MATLAB and tested with the simulated `PMx` marker trajectories. Calculation of (4.6) was performed using MATLAB's built-in `fzero()` function. Based on the MATLAB version, the algorithm was implemented in ANSI C to target practical applications. In this case (4.6) was solved with Brent's root finding algorithm from [85]. Furthermore, compilation options were included to assess the effects of different data precisions (`float` or `double`) on the accuracy and execution time of the algorithm. This was not an option with MATLAB because `fzero()` cannot be used with `float` input.

To address possible accuracy problems arising from the lower precision of `float` data, an additional test case with a simple output continuity check for wrist angles was included, namely when the absolute difference between two successive $\theta_{\texttt{flex\_c}}$ values is larger than 5°, the actual $\theta_{\texttt{flex\_c}}$ will be the previous $\theta_{\texttt{flex\_c}}$+0.5°. This modified version of the algorithm is denoted with *mod.* suffix among the results. Estimated memory footprints of the implemented algorithm variants (for the ARM builds) are listed in Table 4.1.

TABLE 4.1: **Memory footprint estimations of the implemented algorithm for the ARM builds.** Each row represents a separate implementation variant. Table values show the estimated memory footprint (Flash and Static RAM) of the implementation calculated by running the `arm-none-eabi-size` command on the generated executable files.

| Implementation variant | Flash | SRAM |
|---:|---|---|
| **Float** | 23.9 kB | 2.4 kB |
| **Float mod.** | 24.1 kB | 2.4 kB |
| **Double** | 33.4 kB | 2.6 kB |

### Evaluation platforms

MATLAB and C implementations of the proposed algorithm were tested on a system with an Intel® Core® i5-540M processor running Ubuntu Desktop 14.04.4 LTS. In addition, the C implementation was evaluated on the following microcontroller units (MCUs) that are capable of targeting resource constrained environments (e.g. wearable measurement devices) with high performance:

DOI:10.15774/PPKE.ITK.2017.002



1. STM32F407VG – ARM Cortex-M4 core with single precision floating point unit (FPU), up to 168 MHz core clock, 1 MB Flash memory, 192 KB SRAM
2. STM32F746NG – ARM Cortex-M7 core with single precision FPU and L1-cache, up to 216 MHz core clock, 1 MB Flash memory, 320 KB SRAM

For proper comparison, both MCUs were clocked at 168 MHz and the source codes differed only in device specific details (e.g. hardware initialization). Algorithm evaluation on the MCUs was controlled with MATLAB via a UART link including data preparation, transmission and storage.

#### Performance metrics

To evaluate the overall performance of the algorithm compared to OpenSim's IK method, accuracy and execution times were analyzed in all cases (OpenSim, MATLAB and C implementations). To assess accuracy, RMS values were computed for the differences between the calculated and simulated joint coordinate trajectories for each trial. Means and standard deviations of these RMS values were then calculated across trials for each platform and precision (where this was applicable).

Running times of OpenSim's IK evaluation were calculated as a sum of subtask execution times from the IK log output directly. Algorithm execution times were measured by the `tic` and `toc` methods in MATLAB, the `clock()` function from the `<time.h>` library for the C implementation on PC and on-chip hardware timers clocked at 1 MHz for both MCUs.

#### Data exclusion from OpenSim trials

Although inverse kinematics in OpenSim was calculated using an unmodified and unscaled model in each trial, there were cases when large step errors occurred at seemingly random locations in the IK output (independently of subtask borders mentioned in section *Inverse kinematics with OpenSim*). This phenomenon may be handled by marker placement adjustment or error checking in measurement data in general. As IK input was strictly controlled by using simulated trajectories and the markers remained intact in the model between trials, further troubleshooting would have been needed to find a solution to this issue. Because the main emphasis of the study is the proposed algorithm

DOI:10.15774/PPKE.ITK.2017.002



and not OpenSim's internal workings and IK troubleshooting, all OpenSim trials were excluded from final accuracy assessment where any of the resulted joint coordinate RMS errors exceeded 5° to not bias the results with incorrect data. As a result, only 59 trials out of 100 were used to calculate the accuracy of OpenSim's IK algorithm. This however did not have any effect on the other measurements, so MATLAB and all C results were calculated across 100 trials.

## 4.3 Results

### 4.3.1 Accuracy

RMS errors from algorithm evaluation are shown in Table 4.2. The results show that considering the mean of all valid trials (59 for OpenSim, 100 for all others), all platforms performed reasonably well producing errors below 3° for all joint coordinates.

Regarding OpenSim it can be seen that errors for each joint coordinate are larger than those provided by our algorithm. The reason for this can lie in the optimization approach of OpenSim that in fact contains hard-coded convergence (0.0001) and iteration (1000) limits. However these limits prevent OpenSim's IK algorithm to match the simulated movement patterns perfectly, they provide a practical solution to the *accuracy* $\leftrightarrow$ *running time* trade-off for the software's general usage.

MATLAB and C implementations of the proposed algorithm performed equally well for shoulder and elbow angles independent of the used data precision (`double` / `float`). This could occur because of the relatively low number of operations needed by these joint coordinates shown in equation groups (4.2) and (4.4) that prevented considerable error accumulation due to the lower precision of `float`. Regarding wrist angles a clear distinction can be made between `double` and `float` (MATLAB uses `double` as default). The two main reasons for this phenomenon are 1: the significantly larger computational demand of $\theta_{\texttt{dev\_c}}$ and $\theta_{\texttt{flex\_c}}$ involving iterative processes that can lead to precision error accumulations and 2: rounding error based mismatch in the root finding process involved in the calculation of $\theta_{\texttt{flex}}$ in rare cases when two roots are present in (4.6). A deeper analysis among the trial-wise results revealed that the second reason was more significant as roughly 70% of the trials ended up in no more than 0.1° maximum error with

DOI:10.15774/PPKE.ITK.2017.002



`float` precision. The rest of the trials contained 1-5 "wrong" samples showing 15-20° impulse-like errors while the remaining samples within the trial did not have this problem. Investigation of the erroneous samples revealed that indeed a wrong root for (4.6) was found in these cases. To deal with this issue, an output continuity checking step was implemented for `float` precision in cases denoted with the *mod.* suffix. This turned out to be a simple yet effective solution to the problem as the corresponding results show the disappearance of the impulse-like errors.

### 4.3.2 Execution time

To assess overall performance, execution times were compared between OpenSim's IK method and our proposed algorithm on different platforms and are shown in Table 4.3.

Measurement results show that the optimization approach of OpenSim performed the calculation of a single iteration in 145 ms on average. Because of the application specific nature of the proposed algorithm, its running times considering different implementations (MATLAB / C), data precisions (`double` / `float`) and platforms (PC / ARM Cortex-M) all showed a significant increase in execution performance compared to OpenSim, the worst result being about 5 ms on average for a single iteration.

As expected, the C implementation is more than two orders of magnitude faster than the MATLAB version on the PC, yielding execution times per iteration about 10 $\mu$s with all precision variants (`double`, `float` and `float mod.`). Opposed to this, running times on embedded platforms showed more scattered results. The difference between `double` and `float` is more expressed in these cases while application of the FPU accelerates `float` computations even further (*hard float* entries in Table 4.3). Regarding the modified algorithm variant it can be seen that even the extra continuity check adds some amount to the execution time per iteration, the possibility to use `float` precision brings more speed advantage, especially with the FPU enabled. These findings are true for both tested MCUs with the observation that ARM's M7 architecture is about twice as fast as M4 when running the presented algorithm with the same core clock.

DOI:10.15774/PPKE.ITK.2017.002



TABLE 4.2: **RMS errors.** Each row represents a separate test environment for the reference (OpenSim) and proposed inverse kinematics algorithm. The columns show *mean* ± *standard deviation* joint angle RMS errors across all valid trials (59 for OpenSim, 100 otherwise) for each test environment.

| **Test environment** | $\theta_{\texttt{elv}}$ | $\theta_{\texttt{sh\_elv}}$ | $\theta_{\texttt{sh\_rot}}$ | $\theta_{\texttt{el\_flex}}$ | $\theta_{\texttt{pro\_sup}}$ | $\theta_{\texttt{dev\_c}}$ | $\theta_{\texttt{flex\_c}}$ |
|---|---|---|---|---|---|---|---|
| **OpenSim** | 0.0429 ± 0.0339 | 0.0192 ± 0.0053 | 0.1472 ± 0.0760 | 0.0764 ± 0.0288 | 0.6365 ± 0.1701 | 0.9198 ± 0.2477 | 2.2916 ± 1.1142 |
| **MATLAB** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0014 ± 0.0007 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.0023 ± 0.0041 | 0.0049 ± 0.0087 |
| **PC double** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0014 ± 0.0007 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.0025 ± 0.0051 | 0.0053 ± 0.0107 |
| **PC float** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.4193 ± 0.8995 | 1.1148 ± 2.4730 |
| **PC float mod.** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.0045 ± 0.0092 | 0.0097 ± 0.0195 |
| **ARM M4 double** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0014 ± 0.0007 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.0025 ± 0.0051 | 0.0053 ± 0.0107 |
| **ARM M4 soft float** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.4193 ± 0.8995 | 1.1147 ± 2.4730 |
| **ARM M4 hard float** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.4095 ± 0.9051 | 1.0944 ± 2.4840 |
| **ARM M4 soft float mod.** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.0078 ± 0.0128 | 0.0170 ± 0.0276 |
| **ARM M4 hard float mod.** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.0077 ± 0.0128 | 0.0167 ± 0.0275 |
| **ARM M7 double** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0014 ± 0.0007 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.0025 ± 0.0051 | 0.0053 ± 0.0107 |
| **ARM M7 soft float** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.4193 ± 0.8995 | 1.1147 ± 2.4730 |
| **ARM M7 hard float** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.4095 ± 0.9051 | 1.0944 ± 2.4840 |
| **ARM M7 soft float mod.** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.0078 ± 0.0128 | 0.0170 ± 0.0276 |
| **ARM M7 hard float mod.** | 0.0028 ± 0.0003 | 0.0006 ± 0.0002 | 0.0016 ± 0.0013 | 0.0005 ± 0.0002 | 0.0008 ± 0.0006 | 0.0077 ± 0.0128 | 0.0167 ± 0.0275 |

DOI:10.15774/PPKE.ITK.2017.002



TABLE 4.3: **Execution times.** Each row represents a separate test environment for the reference (OpenSim) and proposed inverse kinematics algorithm. Table values show *mean* ± *standard deviation* for a single iteration across all valid trials (59 for OpenSim, 100 otherwise) and the speed increase of each tested setup with respect to OpenSim.

| Test environment | Execution Time per iteration (ms) | Speedup wrt. OpenSim |
|---|---|---|
| **OpenSim** | 145.0532 ± 10.0669 | 1x |
| **MATLAB** | 2.3656 ± 0.6689 | 61x |
| **PC double** | 0.0111 ± 0.0013 | 13011x |
| **PC float** | 0.0088 ± 0.0008 | 16416x |
| **PC float mod.** | 0.0097 ± 0.0013 | 14982x |
| **ARM M4 double** | 4.8777 ± 0.3554 | 30x |
| **ARM M4 soft float** | 2.7327 ± 0.0928 | 53x |
| **ARM M4 hard float** | 0.9713 ± 0.0214 | 149x |
| **ARM M4 soft float mod.** | 2.7394 ± 0.0930 | 53x |
| **ARM M4 hard float mod.** | 0.9740 ± 0.0216 | 149x |
| **ARM M7 double** | 2.3124 ± 0.1704 | 63x |
| **ARM M7 soft float** | 1.4293 ± 0.0504 | 101x |
| **ARM M7 hard float** | 0.4462 ± 0.0117 | 325x |
| **ARM M7 soft float mod.** | 1.4296 ± 0.0505 | 101x |
| **ARM M7 hard float mod.** | 0.4478 ± 0.0115 | 324x |

## 4.4 Discussion

Evaluation results of the tested algorithms show that each approach provides proper accuracy for most common arm movement analysis scenarios. One important aspect however is that while OpenSim provides a useful general tool for biomechanical analysis including fields beyond inverse kinematics (e.g. inverse and forward dynamics), the calculation of joint angles from the actual experimental data is rather demanding computationally. As the output of this step gives the basis for all other analysis methods in the software, the amount of time needed for the overall processing pipeline highly depends on the efficiency of this algorithm. As Table 4.3 shows, the average amount of time needed for OpenSim's IK algorithm to perform a single iteration would allow about 7 Hz operation that falls behind the generally accepted practice in human movement recording of at least 50 Hz. This property excludes OpenSim from tight integration with systems requiring real-time movement kinematics, however that is not the software's original target application anyway (up to version 3.3 at least).

Considering the algorithm proposed in the study Tables 4.2 and 4.3 show a significant improvement in performance in both accuracy and execution time when compared to OpenSim's IK method. The main reason for this difference is the algorithm's application specific nature with the utilization of both the internal structure of the used upper limb

DOI:10.15774/PPKE.ITK.2017.002



model and inertial sensing of movement to determine limb segment orientations directly. As the MATLAB version showed proper accuracy and sufficiently short execution time on the PC, implementation of the algorithm in ANSI C was reasonable to assess its "real" performance without the overhead of a general prototyping tool that MATLAB essentially has. Because accuracy results are the same or very similar across specific variants of the C implementation (i.e. using `double` / `float` precision), only execution time differences are discussed later in the text.

Running times of the algorithm's C implementation showed more than four orders of magnitude speedup on the tested Intel® Core® i5-540M processor compared to OpenSim's IK algorithm on a more recent and higher performance server CPU with Xeon® architecture, yielding about 10 $\mu$s execution time per iteration for all variants. However this is an impressive improvement, running the algorithm on PC would still pose problems from practical aspects of possible applications (e.g. total size and mobility of the measurement system or communication overhead between the measuring and processing device), so the real benefit of this speed increase lie in the "spare" performance that opened the way to testing the algorithm in resource constrained embedded environments. Evaluation of the proposed method on high performance MCUs showed that all implementation variants that provided good accuracy (`double` and `[soft/hard] float mod.`) had acceptable execution times on both architectures (M4 and M7) for real-time operation, considering 100 Hz as sufficient sampling frequency for human movement analysis. Based on these results, the specific implementation variant should be chosen taking the overall design requirements of the actual practical application into account (i.e. wearable measurement devices like the one presented in [C1]) as in most cases the algorithm should fit into a system containing other computationally demanding processes (like sensor fusion algorithms) with power consumption being a critical part of the design for example.

An other practical advantage of the described algorithm is that it enables subject-independent joint angle reconstruction during the measurements. This means that by taking advantage of the offset-independent nature of orientation sensing, no scaling is required for the proper calculation of inverse kinematics (opposed to OpenSim) as long as the IMUs are able to produce good approximations of limb segment orientations.

It needs to be emphasized however that the application specific nature of the algorithm and its dependency on the used upper limb model induce some practical considerations,

DOI:10.15774/PPKE.ITK.2017.002



too. Even in cases when the sensors provide accurate orientation information of the measured limb, care must be taken when determining the limb's reference orientation based on the measurements. The reason for this is mainly inter-subject variability in the sense that even the model defines the reference posture clearly, it cannot be assumed that any actual subject will reproduce the same posture very accurately that can lead to offset errors during the measurement. Furthermore, the assumption was made during algorithm development that the measured movement always remains within the valid joint angle ranges defined in the model. As long as this assumption holds (as in the case of simulated movement patterns presented in this study), the algorithm should not have problems with proper joint angle reconstruction. However, if outliers are present in the experimental data (e.g. reference posture errors, inaccuracies in the measurement or the sensor fusion algorithm or extreme anatomical ranges of a subject) undefined output states can occur. This may be handled with a simple saturation technique on the algorithm level but rather should be prevented by applying proper experimental design and calibration methods. In a practical setup this involves proper sensor placement and various steps before the measurements including zero motion offset compensation, hard and soft iron error compensation in the magnetometer and determining relative sensor orientations with respect to the measured segments [59, 86] for example.

## 4.5 Conclusion

With keeping the upper mentioned considerations in mind it can be stated that the proposed algorithm is capable for real-time reconstruction of standardized anatomical joint angles even in embedded environments, opening the way to complex applications requiring accurate and fast calculation of model-based movement kinematics. Having this property the proposed method brings the possibility to widen the application areas of OpenSim and accelerate its overall analysis pipeline by transferring the calculation of inverse kinematics into the measurement device in cases when inertial movement sensing is applicable.

As an example, in cases when kinematics and muscle activities are both recorded during arm movements, in addition to the reconstruction of joint angles, on-line labeling of muscle activity data can be achieved based on the actual kinematic state of the arm within the measurement device. This may result in reduced overall analysis time with

DOI:10.15774/PPKE.ITK.2017.002



OpenSim and produce a training set for analysis of complex movement activation patterns that can be used to improve the control methods of upper limb prostheses in the future.

DOI:10.15774/PPKE.ITK.2017.002

# Chapter 5

# Conclusion

Each chapter describing the particular field of measurement and analysis of human arm movements is concluded with its own conclusion section. In this chapter the new scientific results are highlighted.

## 5.1 New scientific results

**Thesis I.** *I have shown experimentally that during target tracking arm movements the human movement system optimizes different cost functions based on knowledge about the target trajectory in a way that for visually driven tracking of unfamiliar trajectories the task error to be minimized is defined in target coordinates, whereas for familiar trajectories it is defined in motor coordinates.*

Corresponding publication: [J1]

I have designed an experimental study and the corresponding measurement setup to investigate the differences in motor synergies between predictive and unpredictive tracking arm movements for cases when subjects tracked a target moving in 2D on a graphics tablet with a hand-held pen, while their arm movements were not restricted. The measurement setup assured time accurate presentation of the visual stimulus to trigger subject movement while synchronized recording of the pen's planar position and 3D kinematics of the subject's arm were also realized. By applying the Uncontrolled Manifold Method and techniques from optimal feedback control theory of human arm movements,

DOI:10.15774/PPKE.ITK.2017.002



I have shown that the movement goal differs between tracking of familiar and unfamiliar trajectories. The difference can be characterized by a modification of the task error being minimized for different movement execution modes.

**Thesis II.** *I have developed a wearable measurement device and a corresponding model-based kinematic reconstruction algorithm that is able to determine the arm's anatomical joint angles in real-time, based on the spatial orientations of arm segments. The overall performance gain of the method compared to the approach of a widely used biomechanics simulation software is 1) up to* x14982 *on CPU, 2) up to* x149 *on an ARM Cortex-M4 MCU and 3) up to* x324 *on an ARM Cortex-M7 MCU while it maintains numerical accuracy with the reference solution.*

Corresponding publications: [C1], [J2]

Model based analysis of human upper limb movements has key importance in understanding the motor control processes of our nervous system. Various simulation software packages have been developed over the years to perform model based analysis. These packages provide computationally intensive – and therefore off-line – solutions to calculate the anatomical joint angles from motion captured raw measurement data (also referred as inverse kinematics). In addition, recent developments in inertial motion sensing technology show that it may replace large, immobile and expensive optical systems with small, mobile and cheaper solutions in cases when a laboratory-free measurement setup is needed. The thesis contributes to the workflow of measurement and analysis of human arm movements with an engineering prototype of a wearable measurement system and an algorithm that allows accurate and real-time estimation of anatomical joint angles for a widely used OpenSim upper limb kinematic model when inertial sensors are used for movement recording.

By utilizing the inherent kinematic structure of the selected OpenSim upper limb model (Stanford VA Upper Limb Model [79]), I have created a numerical algorithm that is able to reconstruct model-defined custom rotation angles based on marker positions within a virtual marker set specifically defined for this task. The virtual markers are placed in specific locations within the local coordinate frames of selected model bodies in a way that they form separate orthonormal bases in each anatomical joint of interest

DOI:10.15774/PPKE.ITK.2017.002



(shoulder, elbow and wrist) and represent the corresponding compound rotation matrices of model-defined joint angles in the global reference frame. Having the markers bound to their parent bodies, their positions in the global reference frame are determined by the actual orientation of their corresponding arm segments during any movement within the valid joint limits defined by the model. As the orientation of inertial sensors can be reconstructed from their measured physical quantities with efficient algorithms, by proper placement and calibration they can be used to update virtual marker positions – and as a result, the compound rotation matrices of model-defined joint angles – during measured arm movements. The developed numerical algorithm utilizes this feature to reconstruct the anatomical joint angles of the model in real-time by extracting angle values from the corresponding compound rotation matrices.

DOI:10.15774/PPKE.ITK.2017.002



## 5.2 Application of the results

Given that the thesis covers both theoretical and technical topics of human movement science, several fields of application are possible. In the first part, the behavioral aspects of specific target tracking arm movements were investigated. It was found that available knowledge about the target trajectory has an impact on the actual execution mode of the movement. Experimental data showed that subjects tried to minimize the pen position error when the trajectory of the target was unknown while this goal was shifted towards the minimization of joint angle variability in the case when target trajectory was known as a result of preliminary training. While these findings contribute to the understanding of the human movement system in general, they may be utilized in practical rehabilitation applications as well. Considering post-stroke assessment, the developed experimental setup and procedure may be used to give deeper insight into the actual state of the patient's movement system and reveal higher level effects of the injury (e.g. reduced effectiveness of motor learning and visuo-motor coordination).

In the second part of the thesis, by developing the prototype of a wireless and wearable measurement device based on inertial sensors, the evaluation of laboratory-free measurement of human arm movements was started. As the prototype enables evaluation and analysis of various sensor calibration, filtering and sensor fusion algorithms in a fully customizable setup, it may be used in various applications where measuring the actual kinematic state of the arm can be utilized (e.g. state assessment for rehabilitation, human-machine interfaces or better presence integration in virtual reality environments).

Another contribution to the field of human movement recording was the development of a real-time reconstruction algorithm that is ~~cap~~able to determine model based anatomical joint angles from inertial sensor data directly. As a result, tighter integration of kinematic measurement and reconstruction can be achieved to resolve the time and computational overhead of the offline *measurement-scaling-inverse kinematics* scheme applied in human movement science that has been giving a bottleneck in applications where real-time analysis of the control patterns with respect to the actual kinematics would have been beneficial. As an example, the algorithmic concept of a system for the classification of forearm muscle activity signals based on the arm's kinematic state is presented in [C2], while the design of a practical implementation using real-time data labeling with the developed prototype is shown in [C3].

DOI:10.15774/PPKE.ITK.2017.002

# Appendix A

# Chapter 2: Mathematical Background

## A.1 Partial compensation of planning noise

The planned trajectory $\bar{\underline{x}}_{pl}^{(k)}$ in trial k is described [34] as a random walk process with partial compensation of the error $\bar{\underline{e}}^{(k)} = \left(\bar{\underline{x}}_{pl}^{(k)} - \underline{m}_{pl}\right)$. Starting from $\bar{\underline{x}}_{pl}^{(0)} = \underline{m}_{pl} + \underline{r}_{pl}^{0}$ the random walk is defined by the recursion

$$\bar{\underline{x}}_{pl}^{(k+1)} = \bar{\underline{x}}_{pl}^{(k)} - \mathbf{D}\bar{\underline{e}}^{(k)} + \underline{r}_{pl}^{(k)} \quad , \tag{A.1}$$

with spherical Gaussian noise $\underline{r}_{pl}^{(k)} \sim N\left(\underline{0}, \rho^2\mathbf{I}\right)$. The compensation is proportional to the gain matrix $\mathbf{D} = \mathbf{Q}\left[\delta(i-j)c_i\right]_{ij}\mathbf{Q}^T$, which corrects task-relevant and task-irrelevant errors by a fixed proportion $c_{\mathrm{ORT}}$ and $c_{\mathrm{UCM}}$ respectively:

$$c_i = \begin{cases} c_{\mathrm{ORT}} \text{ for } 1 \leq i \leq DoF_{\mathrm{ORT}} \\ c_{\mathrm{UCM}} \text{ for } 1 + DoF_{\mathrm{ORT}} \leq i \leq DoF_{\mathrm{ORT}} + DoF_{\mathrm{UCM}} \end{cases} \tag{A.2}$$

with $0 < c_{\mathrm{UCM}} < c_{\mathrm{ORT}} < 1$, because correction in the task-relevant dimensions is larger than in the task-irrelevant dimensions. The matrix $\mathbf{Q}$ contains a complete orthonormal basis composed of the two bases of the two subspaces $\mathbf{Q} = [\mathbf{B}_{\mathrm{ORT}}, \mathbf{B}_{\mathrm{UCM}}]$, $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$.

DOI:10.15774/PPKE.ITK.2017.002



Subtracting $\underline{m}_{pl}$ and left multiplying both sides of the recursion in $\bar{\underline{x}}_{pl}^{(k)}$ with $\mathbf{Q}^T$ shows that in centered and normalized coordinates the random walk can be written as

$$\bar{\underline{y}}_{pl}^{(k+1)} = \bar{\underline{y}}_{pl}^{(k)} - [\delta(i-j)c_i]\,\bar{\underline{y}}_{pl}^{(k)} + \mathbf{Q}^T\underline{r}_{pl}^{(k)} \tag{A.3}$$

with $\bar{\underline{y}}_{pl}^{(0)} = \mathbf{Q}^T\underline{r}_{pl}^{0}$ and $\mathbf{Q}^T\underline{r}_{pl}^{(k)} \sim N\left(\underline{0}, \rho^2\mathbf{I}\right)$. The normalized variance in the two subspaces is computed as

$$V_{\mathrm{ORT}}^{(k)} = \rho^2 \sum_{i=0}^{k} (1-c_{\mathrm{ORT}})^{2i} = \rho^2 \frac{1-(1-c_{\mathrm{ORT}})^{2k+2}}{2c_{\mathrm{ORT}} - c_{\mathrm{ORT}}^2} \tag{A.4}$$

and

$$V_{\mathrm{UCM}}^{(k)} = \rho^2 \sum_{i=0}^{k} (1-c_{\mathrm{UCM}})^{2i} = \rho^2 \frac{1-(1-c_{\mathrm{UCM}})^{2k+2}}{2c_{\mathrm{UCM}} - c_{\mathrm{UCM}}^2} \tag{A.5}$$

Thus, the time course of the synergy index is

$$si^{(k)} = \frac{V_{\mathrm{UCM}}^{(k)}}{V_{\mathrm{ORT}}^{(k)}} = \frac{2c_{\mathrm{ORT}} - c_{\mathrm{ORT}}^2}{2c_{\mathrm{UCM}} - c_{\mathrm{UCM}}^2} \cdot \frac{1-(1-c_{\mathrm{UCM}})^{2k+2}}{1-(1-c_{\mathrm{ORT}})^{2k+2}} \quad . \tag{A.6}$$

This shows that the synergy index starts at the value $si^{(0)} = 1$, increases during its time course, and converges to

$$\lim_{k\to\infty} si^{(k)} = \frac{2c_{\mathrm{ORT}} - c_{\mathrm{ORT}}^2}{2c_{\mathrm{UCM}} - c_{\mathrm{UCM}}^2} > 1 \quad . \tag{A.7}$$

## A.2 Separation of feedforward and feedback components with complete knowledge of the target trajectory

An optimal controller was designed for a plant described by

$$\underline{x}_{t+1} = \mathbf{A}\underline{x}_t + \mathbf{B}\underline{u} + \underline{w} \tag{A.8}$$

DOI:10.15774/PPKE.ITK.2017.002



with $\mathbf{A} = 0.8 \cdot \mathbf{I}_{2\times 2}$, $\mathbf{B} = \mathbf{I}_{2\times 2}$ and Gaussian motor noise $\underline{w}_{\mathrm{MOT}} \sim N\left(\underline{0}, \sigma^2_{\mathrm{MOT}}\mathbf{I}\right)$. The control law was designed to minimize the sum of task error and control effort across all time points of the movement:

$$\varepsilon^2 = \sum_{t=0}^{T} \left\| \mathbf{C}\underline{x}_t - y_t \right\|^2 + \left\| \underline{u}_t \right\|^2 \quad . \tag{A.9}$$

The task was to keep the 2D position $\underline{x}$ on a diagonal line (running from left-up to right-down) that intersects the $y$-axis at the moving point $y_t$ (i.e. $\mathbf{C} = [1, 1]$). Yüksel et al. [35] have shown that the optimal control law is given by $\underline{u}_t = \underline{u}^*_t - \mathbf{M}\left(\underline{\hat{x}}_t - \underline{x}^*_t\right)$, where $\underline{x}^*_t$ is the feedforward component that minimizes

$$\varepsilon^{2^*} = \sum_{t=0}^{T} \left\| \mathbf{C}\underline{x}^*_t - y_t \right\|^2 + \left\| \underline{u}^*_t \right\|^2 \tag{A.10}$$

in the absence of any motor noise: $\underline{x}^*_{t+1} = \mathbf{A}\underline{x}^*_t + \mathbf{B}\underline{u}^*$. Feedback of $\underline{x}_t$ was provided by a measurement which was contaminated by Gaussian sensory (proprioceptive) noise $\underline{n}_{\mathrm{PROP}} \sim N\left(\underline{0}, \xi^2_{\mathrm{PROP}} \cdot \mathbf{I}\right)$. A Wiener filter was used to obtain the state estimate $\underline{\hat{x}}_t$ from the measurement:

$$\underline{\hat{x}}_t = \underline{x}^*_t + \mathbf{S}\left(\underline{x}_t + \underline{n}_{\mathrm{PROP}} - \underline{x}^*_t\right) \tag{A.11}$$

with $\mathbf{S} = \dfrac{\sigma^2_{\mathrm{MOT}}}{\sigma^2_{\mathrm{MOT}} + \xi^2_{\mathrm{PROP}}} \cdot \mathbf{I}$.

## A.3 Extension to incomplete knowledge about the target trajectory

Extending the optimal solution given in [35] to situations with incomplete knowledge about the trajectory, it is only necessary to extend the system states $\underline{x}$ by an additional state (trajectory error $\Delta y_t$) expressing the unexpected deviation between the actual and the expected target location:

DOI:10.15774/PPKE.ITK.2017.002



$$\widetilde{\underline{x}}_t = \begin{bmatrix} \underline{x}_t \\ \Delta y_t \end{bmatrix} \tag{A.12}$$

$$\widetilde{\underline{x}}_{t+1} = \begin{bmatrix} \mathbf{A} & \underline{0} \\ \underline{0} & 1 \end{bmatrix} \widetilde{\underline{x}}_t + \begin{bmatrix} \mathbf{B} \\ \underline{0}^T \end{bmatrix} \underline{u} + \begin{bmatrix} \underline{w}_{\mathrm{MOT}} \\ w_T \end{bmatrix} \tag{A.13}$$

The task error is now defined by $\widetilde{\mathbf{C}}\widetilde{\underline{x}} - y_t$ with $\widetilde{\mathbf{C}} = [\mathbf{C}, -1]$. The process noise is extended to the vector

$$\begin{bmatrix} \underline{w}_{\mathrm{MOT}} \\ w_T \end{bmatrix} \sim N\left(\underline{0}, \widetilde{\mathbf{W}}\right) \tag{A.14}$$

with

$$\widetilde{\mathbf{W}} = \begin{bmatrix} \sigma^2_{\mathrm{MOT}} \cdot \mathbf{I} & 0 \\ 0 & \sigma^2_T \end{bmatrix} \tag{A.15}$$

where the covariance $\sigma^2_T$ of the process noise of the trajectory error $\Delta y_t$ quantifies the size of the uncertainty about the trajectory $y_t$. Visual feedback of the actual target position provides, after subtraction of the expected trajectory, a measurement for the actual trajectory error. In this way, the measurement

$$\underline{x}_t + \underline{n}_{\mathrm{PROP}} - \underline{x}^*_t \tag{A.16}$$

is extended to

$$\widetilde{\underline{x}}_t + \begin{bmatrix} \underline{n}_{\mathrm{PROP}} \\ n_T \end{bmatrix} - \widetilde{\underline{x}}^*_t \ , \tag{A.17}$$

where the corresponding measurement noise is extended to the vector

$$\begin{bmatrix} \underline{n}_{\mathrm{PROP}} \\ n_T \end{bmatrix} \sim N\left(\underline{0}, \widetilde{\mathbf{N}}\right) \tag{A.18}$$

DOI:10.15774/PPKE.ITK.2017.002



with

$$\widetilde{\mathbf{N}} = \begin{bmatrix} \xi^2_{\mathrm{PROP}} \cdot \mathbf{I}_{2\times 2} & 0 \\ 0 & \xi^2_T \end{bmatrix} . \tag{A.19}$$

With these modifications the uncertainty about the trajectory $y_t$ was equivalently expressed by an uncertainty in the extended system state $\underline{\widetilde{x}}_t$ . For this modified system, the separation of the optimal control signal $\underline{u}$ into feedforward and feedback components can be performed in exactly the same way as shown by Yüksel et al. [35]. This consideration shows that the feedback control law applied to the estimate of the state error $\underline{\widetilde{x}}$ itself is not expected to depend on the availability of prior knowledge about the trajectory, since both the cost function and system dynamics are not expected to depend on target predictability. In contrast, the optimal filter for estimating the state error depends on the precision of the prior knowledge because the gain matrix $\widetilde{\mathbf{S}}$ of the Wiener filter for the extended system is

$$\widetilde{\mathbf{S}} = \begin{bmatrix} \dfrac{\sigma^2_{\mathrm{MOT}}}{\sigma^2_{\mathrm{MOT}} + \xi^2_{\mathrm{PROP}}} \cdot \mathbf{I}_{2\times 2} & \underline{0} \\ \underline{0}_T & \dfrac{\sigma^2_T}{\sigma^2_T + \xi^2_T} \end{bmatrix} . \tag{A.20}$$

Thus, optimal feedback control predicts that visual feedback should be ignored with very precise prediction of the target trajectory ($\sigma^2_T \to 0$). Similarly, also the estimation gain for the motor states decrease with decreasing motor noise $\sigma^2_{\mathrm{MOT}}$.

## A.4 Task error computed as weighted average of tracking errors in the target space and in the effector space

The optimal feedback controller for the cost function with the variable task error

$$\varepsilon(\theta)^2 = \sum_{t=0}^{T} \theta \cdot \left\| \mathbf{C}\underline{x}_t - \underline{y}_t \right\|^2 + (1-\theta) \cdot \frac{\dim(\underline{y}_t)}{\dim(\underline{x}_t)} \cdot \left\| \mathbf{F}\left(\underline{x}_t - \underline{x}^*_t\right) \right\|^2 + \left\| \underline{u}^T_t \mathbf{R} \underline{u}_t \right\|^2 \tag{A.21}$$

DOI:10.15774/PPKE.ITK.2017.002



was computed for the extended system described in the last paragraph (the $\sim$ is omitted here for simplicity). The projection matrix was $\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$. The feedforward and the feedback components were again computed using the algorithm of Yüksel et al. [35]. In this section it is shown that this is possible because the cost function $\varepsilon(\theta)^2$ can be transformed as required by this algorithm. For each $\theta$ in the range $0 < \theta < 1$, the weighted average of the two tracking errors can be expressed as the square norm of a tracking error expressed in a third coordinate system as

$$\theta \cdot \left\| \mathbf{C}\underline{x}_t - \underline{y}_t \right\|^2 + (1-\theta) \cdot \frac{\dim(\underline{y}_t)}{\dim(\underline{x}_t)} \cdot \left\| \mathbf{F}\left(\underline{x}_t - \underline{x}_t^*\right) \right\|^2 = \left\| \mathbf{D}(\theta) \cdot \underline{x}_t - \underline{d}_t(\theta) \right\|^2 + c_t(\theta) \quad \text{(A.22)}$$

with

$$\mathbf{D}(\theta) = \mathbf{L}^{0.5}(\theta) \cdot \mathbf{V}^T(\theta) \ , \quad \text{(A.23)}$$

$$\underline{d}_t(\theta) = \mathbf{D}^{-1^T}(\theta) \left[ \theta \mathbf{C}^T \underline{y}_t + (1-\theta) \frac{\dim(\underline{y}_t)}{\dim(\underline{x}_t)} \mathbf{F}^T \mathbf{F} \underline{x}_t^* \right] \quad \text{(A.24)}$$

and

$$c_t(\theta) = \theta \underline{y}_t^T \underline{y}_t + (1-\theta) \frac{\dim(\underline{y}_t)}{\dim(\underline{x}_t)} \underline{x}_t^{*T} \mathbf{F}^T \mathbf{F} \underline{x}_t^* - \underline{d}_t^T(\theta) \underline{d}_t(\theta) \ . \quad \text{(A.25)}$$

$\mathbf{V}(\theta)$ and $\mathbf{L}(\theta)$ denote the eigenvectors and eigenvalues of

$$\left[ \theta \mathbf{C}^T \mathbf{C} + (1-\theta) \frac{\dim(\underline{y}_t)}{\dim(\underline{x}_t)} \mathbf{F}^T \mathbf{F} \right] = \mathbf{V}(\theta) \cdot \mathbf{L}(\theta) \cdot \mathbf{V}^T(\theta) \ . \quad \text{(A.26)}$$

Since $c_t(\theta)$ does not depend on the control signal, the minimization of $\varepsilon(\theta)^2$ in $\underline{u}_t$ gives the same result as minimizing

$$\delta(\theta)^2 = \sum_{t=0}^{T} \left\| \mathbf{D}(\theta) \underline{x}_t - \underline{d}_t(\theta) \right\|^2 + \left\| \underline{u}_t^T \mathbf{R} \underline{u}_t \right\|^2 \ . \quad \text{(A.27)}$$

This cost function has the form required by the algorithm of [35].

DOI:10.15774/PPKE.ITK.2017.002



## A.5 Parameter settings for the simulation

Optimal control (feedforward and feedback) of the specified extended system was simulated using the following settings:

- $\xi_T^2 = 0.1$
- $\xi_{\text{PROP}}^2 = 0.05$
- $\sigma_{\text{MOT}}^2 = [0.05, 0.15, 0.25, 0.35, 0.45]$
- $\sigma_T^2 = [0.1, 0.3, 0.5, 0.7, 0.9]$

For each combination of motor noise and trajectory noise 100 trajectories were simulated. Each trajectory was composed of 101 sampling points. The expected trajectory was a sinusoidal movement: $y_t = \sin(\frac{2\pi t}{N+1})$ for $0 \leq t \leq 100$. The resulting tracking errors in the effector space $\mathbf{F}(\underline{x}_t - \underline{x}_t^*)$ were concatenated into a $10100 \times 2$ matrix. The synergy indices shown in Fig. 5A/C were computed as the ratio between the variances of these error distributions projected on the two diagonals. The total variance shown in Fig. 5B/D is the sum of these two variances.

DOI:10.15774/PPKE.ITK.2017.002

# Appendix B

# Chapter 4: Mathematical Background

## B.1 Definitions

To facilitate algorithm description in the text, notations in Table B.1 will be used to identify each degree of freedom and the rotation axes and angle values for them, where each rotation axis ($\mathbf{r}_{\texttt{axis\_id}}$) should be considered as a row vector.

TABLE B.1: **Axis and angle notations.** The table lists identifiers derived from the model file and notations of rotation axes and angles for all relevant degrees of freedom used for algorithm formulation.

| Anatomical joint | Degree of Freedom | Identifier | Rotation axis | Rotation angle |
|---|---|---|---|---|
| Shoulder | *elevation plane* | elv | $\mathbf{r}_{\texttt{elv}}$ | $\theta_{\texttt{elv}}$ |
| | *thoracohumeral (elevation) angle* | sh_elv | $\mathbf{r}_{\texttt{sh\_elv}}$ | $\theta_{\texttt{sh\_elv}}$ |
| | *axial rotation* | sh_rot | $\mathbf{r}_{\texttt{sh\_rot}}$ | $\theta_{\texttt{sh\_rot}}$ |
| Elbow | *elbow flexion* | el_flex | $\mathbf{r}_{\texttt{el\_flex}}$ | $\theta_{\texttt{el\_flex}}$ |
| | *forearm rotation* | pro_sup | $\mathbf{r}_{\texttt{pro\_sup}}$ | $\theta_{\texttt{pro\_sup}}$ |
| Wrist | *deviation* | dev | $\mathbf{r}_{\texttt{dev}}$ | $\theta_{\texttt{dev}}$ |
| | *flexion* | flex | $\mathbf{r}_{\texttt{flex}}$ | $\theta_{\texttt{flex}}$ |
| | *proximal-distal r1* | pdr1 | $\mathbf{r}_{\texttt{pdr1}}$ | $\theta_{\texttt{pdr1}}$ |
| | *proximal-distal r3* | pdr3 | $\mathbf{r}_{\texttt{pdr3}}$ | $\theta_{\texttt{pdr3}}$ |

Furthermore, the following definitions are introduced (for a visual reference, see Figure B.1):

1. An orthonormal basis with a selected rotation axis in its main axis will be noted as $\mathbf{B}_{\texttt{axis\_id\_0}}$, where `axis_id_0` is the identifier of the axis (e.g. for a basis with $\mathbf{r}_{\texttt{elv}}$ in its main axis the basis will be $\mathbf{B}_{\texttt{elv}}$). If the basis is not formed by orthogonalization

DOI:10.15774/PPKE.ITK.2017.002



of rotation vectors in the model (as in the case of the shoulder), $\mathbf{B}_{\texttt{axis\_id\_0}}$ is defined as

$$\mathbf{B}_{\texttt{axis\_id\_0}} = \begin{bmatrix} \mathbf{r}_{\texttt{axis\_id\_0}} \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix}^T \quad \text{where} \tag{B.1a}$$

$$\mathbf{r}_2 = \frac{\mathbf{r}_{\texttt{axis\_id\_0}} \times (\mathbf{r}_{\texttt{axis\_id\_0}} - [1\ 0\ 0])}{\|\mathbf{r}_{\texttt{axis\_id\_0}} \times (\mathbf{r}_{\texttt{axis\_id\_0}} - [1\ 0\ 0])\|} \quad \text{and} \tag{B.1b}$$

$$\mathbf{r}_3 = \frac{\mathbf{r}_{\texttt{axis\_id\_0}} \times \mathbf{r}_2}{\|\mathbf{r}_{\texttt{axis\_id\_0}} \times \mathbf{r}_2\|} \tag{B.1c}$$

2. The rotation axis $\mathbf{r}_{\texttt{axis\_id}}$ expressed in the basis $\mathbf{B}_{\texttt{axis\_id\_0}}$ is defined as

$$\mathbf{r}_{\texttt{axis\_id}}^{\mathbf{B}_{\texttt{axis\_id\_0}}} = \mathbf{r}_{\texttt{axis\_id}} \mathbf{B}_{\texttt{axis\_id\_0}} \tag{B.2}$$

3. Given a unit length vector $\mathbf{r} \in \mathbf{R}^3$, Rodrigues' formula gives the rotation matrix about $\mathbf{r}$ of an arbitrary angle $\theta \in [-\pi, \pi[$ as

$$\begin{gathered} \exp\left(\theta \widehat{\mathbf{r}}\right) = I_3 + \sin\left(\theta\right) \widehat{\mathbf{r}} + \left(1 - \cos\left(\theta\right)\right) \widehat{\mathbf{r}}^2 \\ \text{where} \quad \widehat{\mathbf{r}} : \mathbf{R}^3 \to \mathbf{R}^3 \quad \stackrel{def}{=} \quad \widehat{\mathbf{r}}\ \mathbf{v} = \mathbf{r} \times \mathbf{v} \end{gathered} \tag{B.3}$$

4. The matrix representation of an axis-angle rotation formed from a unit axis $\mathbf{r}_{\texttt{axis\_id}}$ and angle $\theta_{\texttt{axis\_id}}$ expressed in the basis $\mathbf{B}_{\texttt{axis\_id\_0}}$ is given as follows:

$$\begin{gathered} \mathbf{R}_{\mathbf{r}_{\texttt{axis\_id}}}^{\mathbf{B}_{\texttt{axis\_id\_0}}}(\theta_{\texttt{axis\_id}}) = \exp\left(\theta_{\texttt{axis\_id}}\ \widehat{\mathbf{r}}_{\texttt{axis\_id}}^{\mathbf{B}_{\texttt{axis\_id\_0}}}\right) \equiv \\ \begin{bmatrix} xxC + c & xyC - zs & xzC + ys \\ yxC + zs & yyC + c & yzC - xs \\ zxC - ys & zyC + xs & zzC + c \end{bmatrix} \\ \begin{aligned} \text{where} \quad \mathbf{r}_{\texttt{axis\_id}}^{\mathbf{B}_{\texttt{axis\_id\_0}}} &= [x\ y\ z] \\ s &= \sin\left(\theta_{\texttt{axis\_id}}\right) \\ c &= \cos\left(\theta_{\texttt{axis\_id}}\right) \\ C &= 1 - c \end{aligned} \end{gathered} \tag{B.4}$$

5. The $i$-th element of vector $\mathbf{v}$ is denoted as $\mathbf{v}_{(i)}$. Similarly, the $(i,j)$-th element of matrix $\mathbf{R}$ is denoted as $\mathbf{R}_{(i,j)}$, where $i$ is the row index and $j$ is the column index.

DOI:10.15774/PPKE.ITK.2017.002



Indexing complete rows and columns is denoted as $\mathbf{R}_{(i,:)}$ and $\mathbf{R}_{(:,j)}$, respectively.

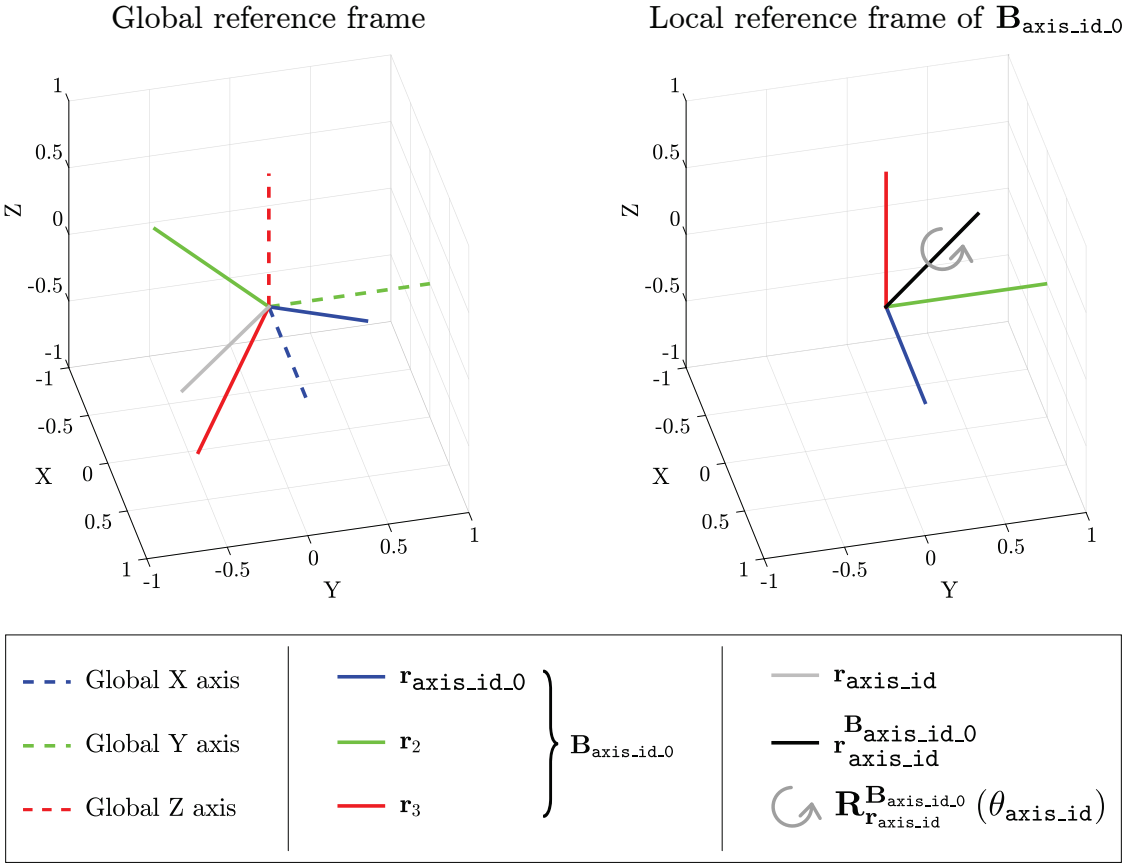


FIGURE B.1: **Visual representation of the definitions.** The subplot on the left side shows elements related to definitions 1 and 2 in the global reference frame ($\mathbf{r}_{\texttt{axis\_id\_0}} = [1, 0.5, 0.5]$, $\mathbf{r}_{\texttt{axis\_id}} = [1, -1, 0]$). The subplot on the right side shows elements related to definitions 2, 3 and 4 in the local reference frame of $\mathbf{B}_{\texttt{axis\_id\_0}}$. The gray arrow shows the spatial rotation that $\mathbf{R}_{\mathbf{r}_{\texttt{axis\_id}}}^{\mathbf{B}_{\texttt{axis\_id\_0}}}(\theta_{\texttt{axis\_id}})$ represents in matrix form.

## B.2 QR orthogonalization

The general formula of QR orthogonalization is shown below, where $\mathbf{A}$ is a regular matrix, $\mathbf{Q}$ is an orthogonal matrix, $\mathbf{R}$ is an upper triangular matrix, $\mathbf{S}$ is the sign diagonal matrix of $\mathbf{R}$ and $\mathbf{B}$ is the resulting orthogonal matrix.

$$\mathbf{A} = \mathbf{QR} \tag{B.5}$$

$$\mathbf{S} = \begin{cases} s_{ii} = 1 & \text{if } r_{ii} > 0 \\ s_{ii} = -1 & \text{if } r_{ii} < 0 \\ s_{ij} = 0 & \text{if } i \neq j \end{cases} \tag{B.6}$$

DOI:10.15774/PPKE.ITK.2017.002



$$\mathbf{B} = \mathbf{QS} \tag{B.7}$$

## B.3 Auxiliary calculations for the shoulder

The orientation of the humerus is determined by four consecutive rotations in the shoulder in the order of *elevation plane*, *elevation angle*, *-elevation plane* and *axial rotation* degrees of freedom (for axis and angle notations, see Table B.1). Based on axis definitions in the model, rotations about $\mathbf{r}_{\texttt{elv}}$ and $\mathbf{r}_{\texttt{sh\_rot}}$ can be estimated with an elementary rotation about the second axis of $\mathbf{B}_{sh\_orth}$ while the estimation of the rotation about $\mathbf{r}_{\texttt{sh\_elv}}$ can be done with an elementary rotation about $\mathbf{B}_{sh\_orth}$'s third axis as shown in (B.8).

$$\mathbf{R}_{\mathbf{r}_{\texttt{elv}}}^{\mathbf{B}_{sh\_orth}}(\theta_{\texttt{elv}}) = \begin{bmatrix} \cos(\theta_{\texttt{elv}}) & 0 & \sin(\theta_{\texttt{elv}}) \\ 0 & 1 & 0 \\ -\sin(\theta_{\texttt{elv}}) & 0 & \cos(\theta_{\texttt{elv}}) \end{bmatrix} \tag{B.8a}$$

$$\mathbf{R}_{\mathbf{r}_{\texttt{sh\_rot}}}^{\mathbf{B}_{sh\_orth}}(\theta_{\texttt{sh\_rot}}) = \begin{bmatrix} \cos(\theta_{\texttt{sh\_rot}}) & 0 & \sin(\theta_{\texttt{sh\_rot}}) \\ 0 & 1 & 0 \\ -\sin(\theta_{\texttt{sh\_rot}}) & 0 & \cos(\theta_{\texttt{sh\_rot}}) \end{bmatrix} \tag{B.8b}$$

$$\mathbf{R}_{\mathbf{r}_{\texttt{sh\_elv}}}^{\mathbf{B}_{sh\_orth}}(\theta_{\texttt{sh\_elv}}) = \begin{bmatrix} \cos(\theta_{\texttt{sh\_rot}}) & -\sin(\theta_{\texttt{sh\_elv}}) & 0 \\ \sin(\theta_{\texttt{sh\_elv}}) & \cos(\theta_{\texttt{sh\_rot}}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{B.8c}$$

Using these definitions, the symbolic expression of the compound rotation matrix that represents the actual orientation in the shoulder can be calculated as shown in (B.9). For the convenience of calculation, this step was performed with MATLAB's Symbolic Math Toolbox using the script shown in appendix B.4.

DOI:10.15774/PPKE.ITK.2017.002



$$\mathbf{R}^{shoulder} = \mathbf{R}^{\mathbf{B}_{sh\_orth}}_{\mathbf{r}_{\mathtt{elv}}}(\theta_{\mathtt{elv}})\ \mathbf{R}^{\mathbf{B}_{sh\_orth}}_{\mathbf{r}_{\mathtt{sh\_elv}}}(\theta_{\mathtt{sh\_elv}})\ \mathbf{R}^{\mathbf{B}_{sh\_orth}}_{\mathbf{r}_{\mathtt{elv}}}(-\theta_{\mathtt{elv}})\ \mathbf{R}^{\mathbf{B}_{sh\_orth}}_{\mathbf{r}_{\mathtt{sh\_rot}}}(\theta_{\mathtt{sh\_rot}}) =$$

$$\begin{bmatrix} A\cos(\theta_{\mathtt{sh\_rot}}) - B\sin(\theta_{\mathtt{sh\_rot}}) & -\cos(\theta_{\mathtt{elv}})\sin(\theta_{\mathtt{sh\_elv}}) & A\sin(\theta_{\mathtt{sh\_rot}}) + B\cos(\theta_{\mathtt{sh\_rot}}) \\ \sin(\theta_{\mathtt{elv}})\sin(\theta_{\mathtt{sh\_elv}})\sin(\theta_{\mathtt{sh\_rot}}) + \cos(\theta_{\mathtt{elv}})\sin(\theta_{\mathtt{sh\_elv}})\cos(\theta_{\mathtt{sh\_rot}}) & \cos(\theta_{\mathtt{sh\_elv}}) & \cos(\theta_{\mathtt{elv}})\sin(\theta_{\mathtt{sh\_elv}})\sin(\theta_{\mathtt{sh\_rot}}) - \sin(\theta_{\mathtt{elv}})\sin(\theta_{\mathtt{sh\_elv}})\cos(\theta_{\mathtt{sh\_rot}}) \\ B\cos(\theta_{\mathtt{sh\_rot}}) - C\sin(\theta_{\mathtt{sh\_rot}}) & \sin(\theta_{\mathtt{elv}})\sin(\theta_{\mathtt{sh\_elv}}) & B\sin(\theta_{\mathtt{sh\_rot}}) + C\cos(\theta_{\mathtt{sh\_rot}}) \end{bmatrix}$$

$$\begin{aligned} \text{where} \quad A &= \cos(\theta_{\mathtt{sh\_elv}})\cos^2(\theta_{\mathtt{elv}}) + \sin^2(\theta_{\mathtt{elv}}) \\ B &= (1 - \cos(\theta_{\mathtt{sh\_elv}}))\cos(\theta_{\mathtt{elv}})\sin(\theta_{\mathtt{elv}}) \\ C &= \cos(\theta_{\mathtt{sh\_elv}})\sin^2(\theta_{\mathtt{elv}}) + \cos^2(\theta_{\mathtt{elv}}) \end{aligned} \tag{B.9}$$

## B.4 MATLAB code for the compound rotation matrix of the shoulder

```
% anonymous function to generate rotation matrix about the second axis of
% the actual basis
R_2nd =@(theta) [ cos(theta)   0   sin(theta) ;
                      0        1       0      ;
                 -sin(theta)   0   cos(theta)];

% anonymous function to generate rotation matrix about the third axis of
% the actual basis
R_3rd =@(theta) [cos(theta)   -sin(theta)   0 ;
                 sin(theta)    cos(theta)   0 ;
                     0             0        1];



% symbolic variables for the joint coordinates (rotation angles)
syms elv_angle shoulder_elv shoulder_rot;
```

DOI:10.15774/PPKE.ITK.2017.002



```
% generate the individual rotation matrices
R_elv_angle = R_2nd(elv_angle);
R_shoulder_elv = R_3rd(shoulder_elv);
R_elv_angle_minus = R_2nd(-elv_angle);
R_shoulder_rot = R_2nd(shoulder_rot);

% calculate the compound rotation matrix
R_shoulder = R_elv_angle * R_shoulder_elv * R_elv_angle_minus * R_shoulder_rot;
```

## B.5 Auxiliary calculations for the elbow

The orientation of the forearm is defined by two consecutive rotations in the order of *elbow flexion* and *forearm rotation*. If expressed in $\mathbf{B}_{\mathtt{pro\_sup}}$, the rotation matrix about $\mathbf{r}_{\mathtt{el\_flex}}$ can be estimated as $\mathbf{R}^{\mathbf{B}_{\mathtt{pro\_sup}}}_{\mathbf{r}_{\mathtt{el\_flex}}}(\theta_{\mathtt{el\_flex}})$ while rotation about $\mathbf{r}_{\mathtt{pro\_sup}}$ corresponds to the elementary rotation about the first axis of $\mathbf{B}_{\mathtt{pro\_sup}}$ (denoted as $\mathbf{R}^{\mathbf{B}_{\mathtt{pro\_sup}}}_{\mathbf{r}_{\mathtt{pro\_sup}}}(\theta_{\mathtt{pro\_sup}})$). Multiplication of these matrices yields the compound rotation matrix in the elbow as shown in (B.10) (the corresponding MATLAB script can be found in appendix B.6).

DOI:10.15774/PPKE.ITK.2017.002



$$\mathbf{R}^{elbow} = \mathbf{R}^{\mathbf{B}_{\text{pro\_sup}}}_{\mathbf{r}_{\text{el\_flex}}}(\theta_{\text{el\_flex}})\ \mathbf{R}^{\mathbf{B}_{\text{pro\_sup}}}_{\mathbf{r}_{\text{pro\_sup}}}(\theta_{\text{pro\_sup}}) =$$

$$= \begin{bmatrix} (1-\cos(\theta_{\text{el\_flex}}))\,x^2 + \cos(\theta_{\text{el\_flex}}) & A\sin(\theta_{\text{pro\_sup}}) - B\cos(\theta_{\text{pro\_sup}}) & A\cos(\theta_{\text{pro\_sup}}) + B\sin(\theta_{\text{pro\_sup}}) \\ z\sin(\theta_{\text{el\_flex}}) - C & D\cos(\theta_{\text{pro\_sup}}) - E\sin(\theta_{\text{pro\_sup}}) & -D\sin(\theta_{\text{pro\_sup}}) - E\cos(\theta_{\text{pro\_sup}}) \\ -y\sin(\theta_{\text{el\_flex}}) - F & G\sin(\theta_{\text{pro\_sup}}) + H\cos(\theta_{\text{pro\_sup}}) & G\cos(\theta_{\text{pro\_sup}}) - H\sin(\theta_{\text{pro\_sup}}) \end{bmatrix}$$

where

$$\begin{aligned} \mathbf{r}^{\mathbf{B}_{\text{pro\_sup}}}_{\text{el\_flex}} &= [x\ y\ z] & D &= (1-\cos(\theta_{\text{el\_flex}}))\,y^2 + \cos(\theta_{\text{el\_flex}}) \\ A &= y\sin(\theta_{\text{el\_flex}}) - F & E &= x\sin(\theta_{\text{el\_flex}}) + yz\,(\cos(\theta_{\text{el\_flex}}) - 1) \\ B &= z\sin(\theta_{\text{el\_flex}}) + C & F &= xz\,(\cos(\theta_{\text{el\_flex}}) - 1) \\ C &= xy\,(\cos(\theta_{\text{el\_flex}}) - 1) & G &= (1-\cos(\theta_{\text{el\_flex}}))\,z^2 + \cos(\theta_{\text{el\_flex}}) \\ & & H &= x\sin(\theta_{\text{el\_flex}}) - yz\,(\cos(\theta_{\text{el\_flex}}) - 1) \end{aligned} \tag{B.10}$$

## B.6 MATLAB code for the compound rotation matrix of the elbow

```
% anonymous function to generate rotation matrix about the first axis of
% the actual basis
R_1st =@(theta) [1      0           0      ;
                 0  cos(theta) -sin(theta) ;
                 0  sin(theta)  cos(theta)];

% symbolic variables for axis coordinates (spatial) and
% joint coordinates (rotation angles)
syms x y z elbow_flexion pro_sup;

% auxiliary variables for the axis-angle rotation matrix
```

DOI:10.15774/PPKE.ITK.2017.002



```
c = cos(elbow_flexion);
s = sin(elbow_flexion);
C = 1 - c;

% generate the individual rotation matrices
R_pro_sup = R_1st(pro_sup);
R_elbow_flexion = [x*x*C+c   , x*y*C-z*s , x*z*C+y*s ;
                   y*x*C+z*s , y*y*C+c   , y*z*C-x*s ;
                   z*x*C-y*s , z*y*C+x*s , z*z*C+c  ];

% calculate the compound rotation matrix
R_total = R_elbow_flexion * R_pro_sup;
```

## B.7 Auxiliary calculations for the wrist

The orientation of the hand is defined in the model by four consecutive rotations in the order of *deviation*, *flexion*, *proximal-distal r1* and *proximal-distal r3*. Although *deviation* and *flexion* are used here as intermediate rotations, naming convention of actively controlled joint coordinates and intermediate rotations is inconsistent in the model file at this point because active joint coordinates of the wrist are called *deviation* and *flexion*, too. To prevent ambiguity, angle values of controlled joint coordinates of the wrist will be denoted as $\theta_{\texttt{dev\_c}}$ and $\theta_{\texttt{flex\_c}}$ further in the text. Intermediate wrist rotations are distributed among two rows of carpal bones with different rotation angles depending on the actual values of $\theta_{\texttt{dev\_c}}$ and $\theta_{\texttt{flex\_c}}$ as follows:

1. *deviation* and *flexion* are defined in the *lunate* body with rotation angle values of $\theta_{\texttt{dev}} = \theta_{\texttt{dev\_c}}$ and $\theta_{\texttt{flex}} = 0.5 * \theta_{\texttt{flex\_c}}$.
2. *proximal-distal r1* and *proximal-distal r3* are defined in the *capitate* body with rotation angle values of $\theta_{\texttt{pdr1}} = 1.5 * \theta_{\texttt{dev\_c}}$ if $\theta_{\texttt{dev\_c}}$ is negative and $\theta_{\texttt{pdr1}} = \theta_{\texttt{dev\_c}}$ otherwise, and $\theta_{\texttt{pdr3}} = 0.5 * \theta_{\texttt{flex\_c}}$.
3. The angle limits for the controlled coordinates are: $\theta_{\texttt{dev\_c}} \in [-10°, 25°]$ and $\theta_{\texttt{flex\_c}} \in [-70°, 70°]$.

DOI:10.15774/PPKE.ITK.2017.002



If expressed in $\mathbf{B}_{\mathtt{pdr3}}$, the rotation matrices about $\mathbf{r}_{\mathtt{dev}}$ and $\mathbf{r}_{\mathtt{flex}}$ can be estimated as $\mathbf{R}_{\mathbf{r}_{\mathtt{dev}}}^{\mathbf{B}_{\mathtt{pdr3}}}(\theta_{\mathtt{dev}})$ and $\mathbf{R}_{\mathbf{r}_{\mathtt{flex}}}^{\mathbf{B}_{\mathtt{pdr3}}}(\theta_{\mathtt{flex}})$, while rotation matrices for $\mathbf{r}_{\mathtt{pdr1}}$ and $\mathbf{r}_{\mathtt{pdr3}}$ can be expressed as $\mathbf{R}_{\mathbf{r}_{\mathtt{pdr1}}}^{\mathbf{B}_{\mathtt{pdr3}}}(\theta_{\mathtt{pdr1}})$ and an elementary rotation about the first axis of $\mathbf{B}_{\mathtt{pdr3}}$, denoted by $\mathbf{R}_{\mathbf{r}_{\mathtt{pdr3}}}^{\mathbf{B}_{\mathtt{pdr3}}}(\theta_{\mathtt{pdr3}})$. Similarly to the shoulder and elbow joints, the compound rotation matrix in the wrist can be written as follows:

$$\mathbf{R}^{wrist} = \mathbf{R}_{\mathbf{r}_{\mathtt{dev}}}^{\mathbf{B}_{\mathtt{pdr3}}}(\theta_{\mathtt{dev}})\ \mathbf{R}_{\mathbf{r}_{\mathtt{flex}}}^{\mathbf{B}_{\mathtt{pdr3}}}(\theta_{\mathtt{flex}})\ \mathbf{R}_{\mathbf{r}_{\mathtt{pdr1}}}^{\mathbf{B}_{\mathtt{pdr3}}}(\theta_{\mathtt{pdr1}})\ \mathbf{R}_{\mathbf{r}_{\mathtt{pdr3}}}^{\mathbf{B}_{\mathtt{pdr3}}}(\theta_{\mathtt{pdr3}}) \tag{B.11}$$

One difficulty however is that this formulation contains three axis-angle rotations out of four that makes $\mathbf{R}^{wrist}$ a very complex symbolic expression with no closed form algebraic solution for the individual rotation angle values.

This problem was handled using a decomposition approach from the literature. As it is shown by Piovan and Bullo in [87], three Euler angles about arbitrary axes can be determined from a rotation matrix if the rotation axes are known and the following condition is fulfilled:

$$\left|\mathbf{r}_1^T\left(\mathbf{R}-\mathbf{r}_2\,\mathbf{r}_2^T\right)\mathbf{r}_3\right| \leq \sqrt{1-\left(\mathbf{r}_1^T\mathbf{r}_2\right)^2}\ \sqrt{1-\left(\mathbf{r}_3^T\mathbf{r}_2\right)^2} \tag{B.12}$$

where $\mathbf{R}$ is the rotation matrix and $\{\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3\}$ are the column vector rotation axes. If we assume that $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$ and $\mathbf{R}$ satisfy (B.12), Euler angle values $\{\theta_1, \theta_2, \theta_3\}$ can be calculated as follows:

- $\theta_2$ is one of the two solutions to

$$(\theta_2)_{1,2} = \operatorname{atan2}(b, a) \pm \operatorname{atan2}\left(\sqrt{a^2+b^2-c^2},\ c\right) \tag{B.13}$$

  where $a = -\mathbf{r}_1^T\widehat{\mathbf{r}}_2^2\mathbf{r}_3$, $b = \mathbf{r}_1^T\widehat{\mathbf{r}}_2\mathbf{r}_3$ and $c = \mathbf{r}_1^T\left(\mathbf{R} - I_3 - \widehat{\mathbf{r}}_2^2\right)\mathbf{r}_3$.
- if $\mathbf{R}^T\mathbf{r}_1 \neq \pm\mathbf{r}_3$, then the angles $\theta_1$ and $\theta_3$ are uniquely determined by

$$\theta_1 = \quad \operatorname{atan2}\left(\mathbf{w}_1^T\mathbf{r}_1 \times \mathbf{v}_1, \mathbf{v}_1^T\mathbf{w}_1 - \left(\mathbf{v}_1^T\mathbf{r}_1\right)\left(\mathbf{w}_1^T\mathbf{r}_1\right)\right) \tag{B.14}$$

$$\theta_3 = -\operatorname{atan2}\left(\mathbf{w}_3^T\mathbf{r}_3 \times \mathbf{v}_3, \mathbf{v}_3^T\mathbf{w}_3 - \left(\mathbf{v}_3^T\mathbf{r}_3\right)\left(\mathbf{w}_3^T\mathbf{r}_3\right)\right) \tag{B.15}$$

  where $\mathbf{v}_1 = \exp\left(\theta_2\widehat{\mathbf{r}}_2\right)\mathbf{r}_3$, $\mathbf{w}_1 = \mathbf{R}\mathbf{r}_3$, $\mathbf{v}_3 = \exp\left(-\theta_2\widehat{\mathbf{r}}_2\right)\mathbf{r}_1$, $\mathbf{w}_3 = \mathbf{R}^T\mathbf{r}_1$.

DOI:10.15774/PPKE.ITK.2017.002



As the inverse of any rotation matrix equals its transpose and the model defines the equality of $\theta_{\texttt{flex}} = \theta_{\texttt{pdr3}}$, (B.11) can be rewritten into

$$\mathbf{R}^{wrist} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_{\texttt{flex}}) & \sin(\theta_{\texttt{flex}}) \\ 0 & -\sin(\theta_{\texttt{flex}}) & \cos(\theta_{\texttt{flex}}) \end{bmatrix} = \mathbf{R}^{\mathbf{B}_{\texttt{pdr3}}}_{\mathbf{r}_{\texttt{dev}}}(\theta_{\texttt{dev}})\, \mathbf{R}^{\mathbf{B}_{\texttt{pdr3}}}_{\mathbf{r}_{\texttt{flex}}}(\theta_{\texttt{flex}})\, \mathbf{R}^{\mathbf{B}_{\texttt{pdr3}}}_{\mathbf{r}_{\texttt{pdr1}}}(\theta_{\texttt{pdr1}}). \quad \text{(B.16)}$$

Having $\widetilde{\mathbf{R}}^{wrist} = \mathbf{R}^{wrist}$, (B.13) can be used to express $\theta_{\texttt{flex}}$ from (B.16) as follows (all $\mathbf{r}^{\mathbf{B}_{\texttt{axis\_id\_0}}}_{\texttt{axis\_id}}$ here are considered as column vectors):

$$(\theta_{\texttt{flex}})_{1,2} = \operatorname{atan2}(b,a) \pm \operatorname{atan2}\left(\sqrt{a^2+b^2-c^2},\ c\right),$$

where

$$\begin{aligned} a &= -\left(\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{dev}}\right)^T \left(\widehat{\mathbf{r}}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{flex}}\right)^2 \mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{pdr1}} \\ b &= \left(\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{dev}}\right)^T \widehat{\mathbf{r}}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{flex}}\ \mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{pdr1}} \\ c &= \left(\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{dev}}\right)^T \left(\widetilde{\mathbf{R}}^{wrist} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_{\texttt{flex}}) & \sin(\theta_{\texttt{flex}}) \\ 0 & -\sin(\theta_{\texttt{flex}}) & \cos(\theta_{\texttt{flex}}) \end{bmatrix} - I_3 - \left(\widehat{\mathbf{r}}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{flex}}\right)^2\right) \mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{pdr1}} = \\ &= x\cos(\theta_{\texttt{flex}}) + y\sin(\theta_{\texttt{flex}}) + z \end{aligned} \quad \text{(B.17)}$$

where $x$, $y$ and $z$ are defined as

$$\begin{aligned} x &= \left(\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{pdr1}}\right)^T \left[0, \left(\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{dev}}\right)^T \widetilde{\mathbf{R}}^{wrist}_{(:,(2,3))}\right]^T \\ y &= \left[\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{pdr1}\ (3)}, -\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{pdr1}\ (2)}\right] \left[\left(\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{dev}}\right)^T \widetilde{\mathbf{R}}^{wrist}_{(:,(2,3))}\right]^T \\ z &= \left[\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{pdr1}\ (1)}, -\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{pdr1}\ (2)}, -\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{pdr1}\ (3)}\right] \left( \begin{bmatrix} \widetilde{\mathbf{R}}^{wrist}_{(1,1)} & \widetilde{\mathbf{R}}^{wrist}_{(2,1)} & \widetilde{\mathbf{R}}^{wrist}_{(3,1)} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \right. \\ &\quad \left. + \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{flex}} \left(\mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{flex}}\right)^T \right) \mathbf{r}^{\mathbf{B}_{\texttt{pdr3}}}_{\texttt{dev}} \end{aligned}$$

DOI:10.15774/PPKE.ITK.2017.002

# References

## Journal publications of the Author

[J1] **Bence J. Borbély**, Andreas Straube, and Thomas Eggert. "Motor synergies during manual tracking differ between familiar and unfamiliar trajectories". In: *Experimental Brain Research* 232.3 (2013), pp. 1–13. ISSN: 00144819. DOI: 10.1007/s00221-013-3801-0.

[J2] **Bence J. Borbély** and Péter Szolgay. "Real-time inverse kinematics for the upper limb: a model-based algorithm using segment orientations". In: *BioMedical Engineering OnLine* 16.1 (2017), p. 21. ISSN: 1475-925X. DOI: 10.1186/s12938-016-0291-x.

[J3] Melanie Krüger, **Bence J. Borbély**, Thomas Eggert, and Andreas Straube. "Synergistic control of joint angle variability: Influence of target shape." In: *Human movement science* 31.5 (Oct. 2012), pp. 1071–89. ISSN: 1872-7646. DOI: 10.1016/j.humov.2011.12.002.

## Conference publications of the Author

[C1] **Bence J. Borbély**, Attila Tihanyi, and Péter Szolgay. "A measurement system for wrist movements in biomedical applications". In: *European Conference on Circuit Theory and Design (ECCTD)* (Aug. 2015). DOI: 10.1109/ECCTD.2015.7300047.

[C2] **Bence J. Borbély** and Péter Szolgay. "Estimating the instantaneous wrist flexion angle from multi-channel surface EMG of forearm muscles". In: *2013 IEEE Biomedical Circuits and Systems Conference, BioCAS 2013* (2013), pp. 77–80. DOI: 10.1109/BioCAS.2013.6679643.

[C3] **Bence J. Borbély** and Péter Szolgay. "A system concept for EMG classification from measurement to deployment". In: *2016 15th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA)*. 2016, pp. 121–122.

DOI:10.15774/PPKE.ITK.2017.002



[C4] **Bence J. Borbély**, Zoltán Kincses, Zsolt Vörösházi, Zoltán Nagy, and Péter Szolgay. "Analysis of myoelectric signals using a Field Programmable SoC". In: *Circuit Theory and Design (ECCTD), 2013 European Conference on*. Sept. 2013, pp. 1–4. DOI: 10.1109/ECCTD.2013.6662255.

[C5] **Bence J. Borbély**, Zoltán Kincses, Zsolt Vörösházi, Zoltán Nagy, and Péter Szolgay. "A modular test platform for real-time measurement and analysis of EMG signals for improved prosthesis control". In: *2014 14th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA)*. July 2014, pp. 1–2. DOI: 10.1109/CNNA.2014.6888643.

## References cited in the thesis

[1] Mark F. Bear, Barry W. Connors, and Michael A. Paradiso. *Neuroscience : exploring the brain*. Third Edition. Lippincott Williams & Wilkins, 2007, p. 857. ISBN: 0781760038.

[2] Huiyu Zhou and Huosheng Hu. "Human motion tracking for rehabilitation-A survey". In: *Biomedical Signal Processing and Control* 3.1 (2008), pp. 1–18. ISSN: 17468094. DOI: 10.1016/j.bspc.2007.09.001.

[3] Peter O'Donoghue. *Research Methods for Sports Performance Analysis*. Abingdon, Oxon: Routledge, 2010. ISBN: 978-0415496230.

[4] Annelies Vandenberghe, Oron Levin, Joris De Schutter, Stephan Swinnen, and Ilse Jonkers. "Three-dimensional reaching tasks: Effect of reaching height and width on upper limb kinematics and muscle activity". In: *Gait and Posture* 32.4 (2010), pp. 500–507. ISSN: 09666362. DOI: 10.1016/j.gaitpost.2010.07.009.

[5] Bart Bolsterlee, Dirkjan H E J Veeger, and Edward K. Chadwick. "Clinical applications of musculoskeletal modelling for the shoulder and upper limb". In: *Medical and Biological Engineering and Computing* 51.9 (2013), pp. 953–963. ISSN: 01400118. DOI: 10.1007/s11517-013-1099-5.

[6] G R Barnes and P T Asselman. "The mechanism of prediction in human smooth pursuit eye movements." In: *The Journal of physiology* 439 (Aug. 1991), pp. 439–61. ISSN: 0022-3751.

[7] W Becker and AF Fuchs. "Prediction in the oculomotor system: smooth pursuit during transient disappearance of a visual target". In: *Experimental Brain Research* (1985), pp. 562–575.

DOI:10.15774/PPKE.ITK.2017.002



[8] GM Gauthier, JL Vercher, F Mussa Ivaldi, and E Marchetti. "Oculo-manual tracking of visual targets: control learning, coordination control and coordination model". In: *Experimental brain research* (1988), pp. 127–137.

[9] R C Miall and G Z Reckess. "The cerebellum and the timing of coordinated eye and hand tracking." In: *Brain and cognition* 48.1 (Feb. 2002), pp. 212–26. ISSN: 0278-2626. DOI: 10.1006/brcg.2001.1314.

[10] GR Barnes and JF Marsden. "Anticipatory control of hand and eye movements in humans during oculo-manual tracking". In: *The Journal of physiology* (2002), pp. 317–330. DOI: 10.1013/jphysiol.2001.012979.

[11] K.C. Engel and J.F. Soechting. "Manual tracking in two dimensions". In: *Journal of Neurophysiology* 83.6 (2000), p. 3483.

[12] Leigh A Mrotek, C C A M Gielen, and Martha Flanders. "Manual tracking in three dimensions." In: *Experimental brain research.* 171.1 (May 2006), pp. 99–115. ISSN: 0014-4819.

[13] J P Scholz and G Schöner. "The uncontrolled manifold concept: identifying control variables for a functional task." In: *Experimental brain research* 126.3 (June 1999), pp. 289–306. ISSN: 0014-4819.

[14] Mark L Latash, John P Scholz, and Gregor Schöner. "Toward a new theory of motor synergies." In: *Motor control* 11.3 (July 2007), pp. 276–308. ISSN: 1087-1640.

[15] SMSF de Freitas, JP Scholz, and AJ Stehman. "Effect of motor planning on use of motor abundance". In: *Neuroscience letters* 417.1 (2007), pp. 66–71.

[16] John P Scholz, Frederic Danion, Mark L Latash, and Gregor Schöner. "Understanding finger coordination through analysis of the structure of force variability." In: *Biological cybernetics* 86.1 (Jan. 2002), pp. 29–39. ISSN: 0340-1200.

[17] Ning Kang, Minoru Shinohara, Vladimir M Zatsiorsky, and Mark L Latash. "Learning multi-finger synergies: an uncontrolled manifold analysis." In: *Experimental brain research* 157.3 (Aug. 2004), pp. 336–50. ISSN: 0014-4819. DOI: 10.1007/s00221-004-1850-0.

[18] JR Martin and MK Budgeon. "Stabilization of the total force in multi-finger pressing tasks studied with the 'inverse piano'technique". In: *Human movement science* 30.3 (2011), pp. 446–458. DOI: 10.1016/j.humov.2010.08.021.Stabilization.

[19] Dmitry Domkin, Jozsef Laczko, Slobodan Jaric, Hakan Johansson, and Mark L Latash. "Structure of joint variability in bimanual pointing tasks." In: *Experimental brain research* 143.1 (Mar. 2002), pp. 11–23. ISSN: 0014-4819. DOI: 10.1007/s00221-001-0944-1.

DOI:10.15774/PPKE.ITK.2017.002



[20] Dmitry Domkin, Jozsef Laczko, Mats Djupsjöbacka, Slobodan Jaric, and Mark L Latash. "Joint angle variability in 3D bimanual pointing: uncontrolled manifold analysis." In: *Experimental brain research.* 163.1 (May 2005), pp. 44–57. ISSN: 0014-4819. DOI: 10.1007/s00221-004-2137-1.

[21] Emanuel Todorov and Michael I Jordan. "Optimal feedback control as a theory of motor coordination." In: *Nature Neuroscience* 5.11 (2002), pp. 1226–1235. ISSN: 10976256. DOI: 10.1038/nn963.

[22] A.V. Jr. Hays, B.J. Richmond, and L.M. Optican. "Unix-based multiple-process system, for real-time data acquisition and control". English. In: (Jan. 1982).

[23] C De'Sperati and P Viviani. "The relationship between curvature and velocity in two-dimensional smooth pursuit eye movements." In: *The Journal of neuroscience : the official journal of the Society for Neuroscience* 17.10 (May 1997), pp. 3932–45. ISSN: 0270-6474.

[24] P Viviani, P Campadelli, and P Mounoud. "Visuo-manual pursuit tracking of human two-dimensional movements." In: *Journal of Experimental Psychology: Human Perception and Performance* 13.1 (1987), pp. 62–78. ISSN: 0096-1523. DOI: 10.1037/0096-1523.13.1.62.

[25] P Viviani and P Mounoud. "Perceptuomotor compatibility in pursuit tracking of two-dimensional movements." In: *Journal of motor behavior* 22.3 (1990), pp. 407–443. ISSN: 0022-2895. DOI: 10.1080/00222895.1990.10735521.

[26] Eric D Vidoni, Jason S McCarley, Jodi D Edwards, and Lara a Boyd. "Manual and oculomotor performance develop contemporaneously but independently during continuous tracking." In: *Experimental brain research. Experimentelle Hirnforschung. Expérimentation cérébrale* 195.4 (June 2009), pp. 611–20. ISSN: 1432-1106. DOI: 10.1007/s00221-009-1833-2.

[27] G. R. Barnes and S. G. Wells. "Elling Prediction in Ocular Pursuit". In: *Current Oculomotor Research: Physiological and Psychological Aspects*. Ed. by Wolfgang Becker, Heiner Deubel, and Thomas Mergner. Boston, MA: Springer US, 1999, pp. 97–107. ISBN: 978-1-4757-3054-8. DOI: 10.1007/978-1-4757-3054-8_14.

[28] John F Soechting, Hrishikesh M Rao, and John Z Juveli. "Incorporating prediction in models for two-dimensional smooth pursuit." In: *PloS one* 5.9 (Jan. 2010), e12574. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0012574.

[29] Halla Olafsdottir, Naoki Yoshida, Vladimir M. Zatsiorsky, and Mark L. Latash. "Anticipatory covariation of finger forces during self-paced and reaction time force production". In: *Neuroscience Letters* 381.1-2 (2005), pp. 92–96. ISSN: 03043940. DOI: 10.1016/j.neulet.2005.02.003. arXiv: NIHMS150003.

DOI:10.15774/PPKE.ITK.2017.002



[30] Kun Shim Jae, Halla Olafsdottir, Vladimir M. Zatsiorsky, and Mark L. Latash. "The emergence and disappearance of multi-digit synergies during force-production tasks". In: *Experimental Brain Research* 164.2 (2005), pp. 260–270. ISSN: 00144819. DOI: 10.1007/s00221-005-2248-3.

[31] Sun Wook Kim, Jae Kun Shim, Vladimir M Zatsiorsky, and Mark L Latash. "Anticipatory adjustments of multi-finger synergies in preparation for self-triggered perturbations." In: *Experimental brain research. Experimentelle Hirnforschung. Expérimentation cérébrale* 174.4 (Oct. 2006), pp. 604–12. ISSN: 0014-4819. DOI: 10.1007/s00221-006-0505-8.

[32] Miriam Klous, Pavle Mikulic, and Mark L Latash. "Two aspects of feedforward postural control: anticipatory postural adjustments and anticipatory synergy adjustments." In: *Journal of neurophysiology* 105.5 (2011), pp. 2275–2288. ISSN: 0022-3077. DOI: 10.1152/jn.00665.2010.

[33] Emanuel Todorov. "Optimality principles in sensorimotor control." In: *Nature neuroscience* 7.9 (Sept. 2004), pp. 907–15. ISSN: 1097-6256. DOI: 10.1038/nn1309. arXiv: NIHMS150003.

[34] Robert J van Beers, Eli Brenner, and Jeroen B J Smeets. "Random walk of motor planning in task-irrelevant dimensions." In: *Journal of neurophysiology* 109.4 (Feb. 2013), pp. 969–77. ISSN: 1522-1598. DOI: 10.1152/jn.00706.2012.

[35] S. Yuksel, H. Hindi, and L. Crawford. "Optimal tracking with feedback-feedforward control separation over a network". In: *Proceedings of American Control Conference* (2006), pp. 3500–3506. ISSN: 07431619. DOI: 10.1109/ACC.2006.1657260.

[36] Michael Athans. *The Role and Use of the Stochastic Linear-Quadratic-Gaussian Problem in Control System Design*. Dec. 1971. DOI: 10.1109/TAC.1971.1099818.

[37] Emanuel Todorov. "Stochastic optimal control and estimation methods adapted to the noise characteristics of the sensorimotor system." In: *Neural computation* 17.5 (May 2005), pp. 1084–1108. ISSN: 0899-7667. DOI: 10.1162/0899766053491887. arXiv: NIHMS150003.

[38] Okihide Hikosaka, Katsuyuki Sakai, Xiaofeng Lu, Hiroyuki Nakahara, Miya K Rand, Kae Nakamura, Shigehiro Miyachi, and Kenji Doya. *Parallel neural networks for learning sequential procedures*. Oct. 1999. DOI: 10.1016/S0166-2236(99)01439-3.

[39] H. J. Luinge and Peter H. Veltink. "Measuring orientation of human body segments using miniature gyroscopes and accelerometers". In: *Medical and Biological Engineering and Computing* 43.2 (2005), pp. 273–282. ISSN: 01400118. DOI: 10.1007/BF02345966.

DOI:10.15774/PPKE.ITK.2017.002



[40] S Sabatelli, M Galgani, L Fanucci, and A Rocchi. "A double stage Kalman filter for sensor fusion and orientation tracking in 9D IMU". In: *Sensors Applications Symposium (SAS), 2012 IEEE*. Feb. 2012, pp. 1–5. DOI: 10.1109/SAS.2012.6166315.

[41] Sebastian O H Madgwick, Andrew J L Harrison, and Andrew Vaidyanathan. "Estimation of IMU and MARG orientation using a gradient descent algorithm." English. In: *IEEE International Conference on Rehabilitation Robotics* 2011 (Jan. 2011), p. 5975346. ISSN: 1945-7901. DOI: 10.1109/ICORR.2011.5975346.

[42] Robert Mahony, Tarek Hamel, and Jean Michel Pflimlin. "Nonlinear complementary filters on the special orthogonal group". In: *IEEE Transactions on Automatic Control* 53.5 (2008), pp. 1203–1218. ISSN: 00189286. DOI: 10.1109/TAC.2008.923738.

[43] Ya Tian, Hongxing Wei, and Jindong Tan. "An adaptive-gain complementary filter for real-time human motion tracking with MARG sensors in free-living environments". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 21.2 (2013), pp. 254–264. ISSN: 15344320. DOI: 10.1109/TNSRE.2012.2205706.

[44] A Olivares, J M Górriz, J Ramírez, and G Olivares. "Accurate human limb angle measurement: sensor fusion through Kalman, least mean squares and recursive least-squares adaptive filtering". In: *Measurement Science and Technology* 22.2 (2011), p. 25801.

[45] Wonkeun Youn and Jung Kim. "Development of a compact-size and wireless surface EMG measurement system". In: *ICCAS-SICE, 2009*. Aug. 2009, pp. 1625–1628.

[46] Mohammadreza Balouchestani and Sridhar Krishnan. "Effective low-power wearable wireless surface EMG sensor design based on analog-compressed sensing." In: *Sensors (Basel, Switzerland)* 14.12 (Jan. 2014), pp. 24305–28. ISSN: 1424-8220. DOI: 10.3390/s141224305.

[47] Kang-Ming Chang, Shin-Hong Liu, and Xuan-Han Wu. "A wireless sEMG recording system and its application to muscle fatigue detection." In: *Sensors (Basel, Switzerland)* 12.1 (Jan. 2012), pp. 489–99. ISSN: 1424-8220. DOI: 10.3390/s120100489.

[48] Xun Chen and Z. Jane Wang. "Pattern recognition of number gestures based on a wireless surface EMG system". In: *Biomedical Signal Processing and Control* 8.2 (Mar. 2013), pp. 184–192. ISSN: 17468094. DOI: 10.1016/j.bspc.2012.08.005.

[49] C. A. Avizzano, E. Ruffaldi, and M. Bergamasco. "A novel wearable biometric capture system". In: *22nd Mediterranean Conference on Control and Automation*. June 2014, pp. 351–355. DOI: 10.1109/MED.2014.6961396.

DOI:10.15774/PPKE.ITK.2017.002



[50] L. Peppoloni, A. Filippeschi, E. Ruffaldi, and C. A. Avizzano. "(WMSDs issue) A novel wearable system for the online assessment of risk for biomechanical load in repetitive efforts". In: *International Journal of Industrial Ergonomics* 52 (2014), pp. 1–11. ISSN: 18728219. DOI: 10.1016/j.ergon.2015.07.002.

[51] InvenSense Inc. *MPU-9250 Product Specification*. 2014.

[52] Peter Konrad. *The ABC of EMG*. Tech. rep. March. 2006, pp. 1–61. URL: http://www.noraxon.com/docs/education/abc-of-emg.pdf.

[53] Mark Pedley. *High-precision calibration of a three axis accelerometer (AN4399)*. Tech. rep. Oct. 2015, pp. 1–35.

[54] STMicroelectronics. *Parameters and calibration of a low-g 3-axis accelerometer (AN4508)*. Tech. rep. June. 2014, pp. 1–13.

[55] Bin Fang, Wusheng Chou, and Li Ding. "An optimal calibration method for a MEMS inertial measurement unit". In: *International Journal of Advanced Robotic Systems* 11.1 (2014), pp. 1–14. ISSN: 17298806. DOI: 10.5772/57516.

[56] Sara Stancin and Saso Tomazic. "Time- and computation-efficient calibration of MEMS 3D accelerometers and gyroscopes". In: *Sensors (Basel, Switzerland)* 14.8 (2014), pp. 14885–14915. ISSN: 14248220. DOI: 10.3390/s140814885.

[57] Demoz Gebre-Egziabher, Gabriel H. Elkaim, J. David Powell, and Bradford W. Parkinson. "Calibration of Strapdown Magnetometers in Magnetic Field Domain". In: *Journal of Aerospace Engineering* 19.April (2006), pp. 87–102. ISSN: 0893-1321. DOI: 10.1061/(ASCE)0893-1321(2006)19:2(87).

[58] Valérie Renaudin, Muhammad Haris Afzal, and Gérard Lachapelle. "Complete triaxis magnetometer calibration in the magnetic domain". In: *Journal of Sensors* 2010 (2010). ISSN: 1687725X. DOI: 10.1155/2010/967245.

[59] S. Bonnet, C. Bassompierre, C. Godin, S. Lesecq, and a. Barraud. "Calibration methods for inertial and magnetic sensors". In: *Sensors and Actuators A: Physical* 156.2 (Dec. 2009), pp. 302–311. ISSN: 09244247. DOI: 10.1016/j.sna.2009.10.008.

[60] Manon Kok and Thomas B. Schön. "Magnetometer calibration using inertial sensors". In: (2016), pp. 1–31. ISSN: 1530437X. DOI: 10.1109/JSEN.2016.2569160. arXiv: 1601.05257. URL: http://arxiv.org/abs/1601.05257.

[61] Yuri Petrov. *Ellipsoid fitting algorithm for MATLAB*. 2009. URL: https://www.mathworks.com/matlabcentral/fileexchange/24693-ellipsoid-fit.

DOI:10.15774/PPKE.ITK.2017.002



[62] A. Cavallo, A. Cirillo, P. Cirillo, G. De Maria, P. Falco, C. Natale, and S. Pirozzi. "Experimental comparison of sensor fusion algorithms for attitude estimation". In: *IFAC Proceedings Volumes (IFAC-PapersOnline)* 19 (2014), pp. 7585–7591. ISSN: 14746670. DOI: 10.3182/20140824-6-ZA-1003.01173.

[63] Scott L. Delp, J. Peter Loan, Melissa G. Hoy, Felix E. Zajac, Eric L. Topp, and Joseph M. Rosen. "An Interactive Graphics-Based Model of the Lower Extremity to Study Orthopaedic Surgical Procedures". In: *IEEE Transactions on Biomedical Engineering* 37.8 (1990), pp. 757–767. ISSN: 15582531. DOI: 10.1109/10.102791.

[64] Scott L Delp, Frank C Anderson, Allison S Arnold, Peter Loan, Ayman Habib, Chand T John, Eran Guendelman, and Darryl G Thelen. "OpenSim: open-source software to create and analyze dynamic simulations of movement." In: *IEEE transactions on bio-medical engineering* 54.11 (Nov. 2007), pp. 1940–1950. ISSN: 0018-9294. DOI: 10.1109/TBME.2007.901024.

[65] H. Zheng, N. D. Black, and Nigel D. Harris. "Position-sensing technologies for movement analysis in stroke rehabilitation". In: *Medical and Biological Engineering and Computing* 43.4 (2005), pp. 413–420. ISSN: 01400118. DOI: 10.1007/BF02344720.

[66] Ching Yi Wu, Keh Chung Lin, Hsieh Ching Chen, I Hsuen Chen, and Wei Hsien Hong. "Effects of modified constraint-induced movement therapy on movement kinematics and daily function in patients with stroke: a kinematic study of motor control mechanisms." In: *Neurorehabilitation and neural repair* 21.5 (2007), pp. 460–6. ISSN: 1545-9683. DOI: 10.1177/1545968307303411.

[67] Jennifer L. Stephenson, Anouk Lamontagne, and Sophie J. De Serres. "The coordination of upper and lower limb movements during gait in healthy and stroke individuals". In: *Gait and Posture* 29.1 (2009), pp. 11–16. ISSN: 09666362. DOI: 10.1016/j.gaitpost.2008.05.013.

[68] Nianfeng Yang, Ming Zhang, Changhua Huang, and Dewen Jin. "Motion quality evaluation of upper limb target-reaching movements". In: *Medical Engineering and Physics* 24.2 (2002), pp. 115–120. ISSN: 13504533. DOI: 10.1016/S1350-4533(01)00121-7.

[69] Nicolas Vignais and Frédéric Marin. "Analysis of the musculoskeletal system of the hand and forearm during a cylinder grasping task". In: *International Journal of Industrial Ergonomics* 44.4 (2014), pp. 535–543. ISSN: 18728219. DOI: 10.1016/j.ergon.2014.03.006.

DOI:10.15774/PPKE.ITK.2017.002



[70] D. Song, N. Lan, G. E. Loeb, and J. Gordon. "Model-based sensorimotor integration for multi-joint control: Development of a virtual arm model". In: *Annals of Biomedical Engineering* 36.6 (2008), pp. 1033–1048. ISSN: 00906964. DOI: 10.1007/s10439-008-9461-8.

[71] Moon Seok Park, Chin Youb Chung, Sang Hyeong Lee, In Ho Choi, Tae Joon Cho, Won Joon Yoo, B. S Myoung Yl Park, and Kyoung Min Lee. "Effects of distal hamstring lengthening on sagittal motion in patients with diplegia. Hamstring length and its clinical use". In: *Gait and Posture* 30.4 (2009), pp. 487–491. ISSN: 09666362. DOI: 10.1016/j.gaitpost.2009.07.115.

[72] Edith M. Arnold, Samuel R. Ward, Richard L. Lieber, and Scott L. Delp. "A model of the lower limb for analysis of human movement". In: *Annals of Biomedical Engineering* 38.2 (2010), pp. 269–279. ISSN: 00906964. DOI: 10.1007/s10439-009-9852-5.

[73] DirkJan H E J Veeger. "" What if": The use of biomechanical models for understanding and treating upper extremity musculoskeletal disorders". In: *Manual Therapy* 16.1 (2011), pp. 48–50. ISSN: 1356689X. DOI: 10.1016/j.math.2010.09.004.

[74] Agneta Gustus, Georg Stillfried, and Judith Visser. "Human hand modelling: kinematics, dynamics, applications". In: *Biological Cybernetics* (2012), pp. 741–755. DOI: 10.1007/s00422-012-0532-4.

[75] Silvia Muceli and Dario Farina. "Simultaneous and proportional estimation of hand kinematics from EMG during mirrored movements at multiple degrees-of-freedom." In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 20.3 (May 2012), pp. 371–8. ISSN: 1558-0210. DOI: 10.1109/TNSRE.2011.2178039.

[76] Ning Jiang, Johnny Lg Vest-Nielsen, Silvia Muceli, and Dario Farina. "EMG-based simultaneous and proportional estimation of wrist/hand dynamics in uni-Lateral trans-radial amputees". In: *Journal of NeuroEngineering and Rehabilitation* 9.1 (2012), p. 42. ISSN: 1743-0003. DOI: 10.1186/1743-0003-9-42.

[77] Ning Jiang, Silvia Muceli, Bernhard Graimann, and Dario Farina. "Effect of arm position on the prediction of kinematics from EMG in amputees". In: *Medical and Biological Engineering and Computing* 51.1-2 (2013), pp. 143–151. ISSN: 01400118. DOI: 10.1007/s11517-012-0979-4.

[78] Dimitra Blana, Theocharis Kyriacou, Joris M Lambrecht, and Edward K Chadwick. "Feasibility of using combined EMG and kinematic signals for prosthesis control: A simulation study using a virtual reality environment". In: *Journal of*

DOI:10.15774/PPKE.ITK.2017.002



*Electromyography and Kinesiology* 29 (2015), pp. 21–27. ISSN: 1873-5711. DOI: 10.1016/j.jelekin.2015.06.010.

[79] Katherine R S Holzbaur, Wendy M Murray, and Scott L Delp. "A Model of the Upper Extremity for Simulating Musculoskeletal Surgery and Analyzing Neuromuscular Control". In: *Annals of Biomedical Engineering* 33.6 (June 2005), pp. 829–840. ISSN: 0090-6964. DOI: 10.1007/s10439-005-3320-7.

[80] Katherine R S Holzbaur, Wendy M Murray, and Scott L Delp. *Upper Extremity Kinematic Model, Simtk resource*. URL: https://simtk.org/home/up-ext-model (visited on 07/06/2016).

[81] Scott L Delp, Peter Loan, and Blaikie Krystyne. *SIMM 7.0 for Windows User's Manual*. 2013. URL: http://www.musculographics.com/download/SIMM7.0UserGuide.pdf (visited on 07/06/2016).

[82] Ge Wu et al. "ISB recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion - Part II: Shoulder, elbow, wrist and hand". In: *Journal of Biomechanics* 38.5 (2005), pp. 981–992. ISSN: 00219290. DOI: 10.1016/j.jbiomech.2004.05.042. eprint: 111.

[83] J Hicks. *OpenSim Documentations: How Scaling Works*. URL: http://simtk-confluence.stanford.edu:8080/display/OpenSim/How+Scaling+Works (visited on 07/06/2016).

[84] J Hicks. *OpenSim Documentations: How Inverse Kinematics Works*. URL: http://simtk-confluence.stanford.edu:8080/display/OpenSim/How+Inverse+Kinematics+Works (visited on 07/06/2016).

[85] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C (2Nd Ed.): The Art of Scientific Computing*. New York, NY, USA: Cambridge University Press, 1992, pp. 359–362. ISBN: 0-521-43108-5.

[86] M Vanegas and L Stirling. "Characterization of inertial measurement unit placement on the human body upon repeated donnings". In: *2015 IEEE 12th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*. June 2015, pp. 1–6. DOI: 10.1109/BSN.2015.7299398.

[87] Giulia Piovan and Francesco Bullo. "On coordinate-free rotation decomposition: Euler angles about arbitrary axes". In: *IEEE Transactions on Robotics* 28.3 (2012), pp. 728–733. ISSN: 15523098. DOI: 10.1109/TRO.2012.2184951.