Hochschule für Angewandte
Wissenschaften Hamburg
Hamburg University of Applied Sciences

Development of a Wearable Electronic Sensor Array and Measuring Unit for Spine and Posture Analysis

for Use with Standardized Patients in Medical Simulation and Education

A thesis submitted in partial fulfillment for requirements for the degree of
Master of Science (M.Sc.) in Biomedical Engineering

at the

Faculty of Life Sciences
Hamburg University of Applied Sciences

Björn Krystek
2001748

March 14, 2016

Reviewed by: Prof. Dr.-Ing. Boris Tolg
Prof. Dr.-Ing. Friedrich Ueberle

Abstract

Pre-existing spinal injuries can have severe implications on the patient's medical condition, when subjected to abnormal loads. During transport, these patients are exposed to an increased risk of secondary traumatization. Rescue personnel can be trained for the interaction with patients with pre-existing spinal injuries in specific exercises. A system to measure spinal rotation in medical simulation scenarios can be used to monitor and evaluate the participants. A portable system for spine and posture analysis was developed in this thesis. Based on inertial measurement units, the sensor system can be fitted unobtrusively to the spine for detailed recording. Data analysis and visualization is possible using a dedicated computer program. Evaluation of usability and reliability of the monitoring unit shows the potential of the system in the described use case. The designed prototype provides a solid and functional foundation for future development.

Zusammenfassung

Wirbelsäulenvorschädigungen können bei Über- oder Fehlbelastung massive Auswirkungen auf den Gesundheitszustand von Patienten haben. Besonders bei Krankentransporten sind diese Patienten einem erhöhten Risiko von Folgeverletzungen ausgesetzt. Während gezielter Übungen ist es möglich, das Rettungsdienstpersonal auf entsprechende Einsätze vorzubereiten und zu schulen. Ein System zur Erfassung der Wirbelsäulenrotationen kann bei der Medizinischen Simulation eingesetzt werden um die Leistung der Simulationsteilnehmer zu kontrollieren und zu bewerten. Im Rahmen dieser Thesis wurde ein portables System zur Erfassung der Körperhaltung entwickelt. Das Sensorsystem basiert auf Inertialsensoren und wird versteckt entlang der Wirbelsäule fixiert Sämtliche Bewegungen der Wirbelsäule werden detailliert aufzeichnen. Die Darstellung und Auswertung der Daten erfolgt mittels zugehörigen Computerprogramms. Verschiedene Auswertungen des Sensorsystems bezüglich Nutzbarkeit und Zuverlässigkeit zeigen Potenzial für die zukünftige Verwendung im beschriebenen Anwendungsfall. Der erarbeitete Prototyp bietet eine fundierte und funktionsfähige Grundlage, auf die für eine weitere Entwicklung aufgebaut werden kann.

Affidavit

Ich versichere, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind in allen Fällen unter Angabe der Quelle kenntlich gemacht.

Björn Krystek
Hamburg, den 14. März 2016

Contents

List of Figures

List of Tables

Acronyms

CA Computer Anatometry.

CAD Computer-Aided Design.

CS Chip Select.

CSV Comma Separated Values.

DMP Digital Motion Processor.

DOF Degrees of Freedom.

dps Degree Per Second.

FIFO First In, First Out.

FSU Functional Spinal Unit, *also known as Motion Segment*.

GPIO General Purpose Input/Output.

I^{2}C Inter-Integrated Circuit, *also I2C or IIC*.

IAR Instant Axis of Rotation.

IDE Integrated Development Environment.

IMU Inertial Measurement Unit.

LBD Low Back Disorder.

LC Low Cost.

Li-Ion Lithium-Ion.

LMM Lumbar Motion Monitor.

MCI Mass Casualty Incident.

MCU Micro Controller Unit.

MEMS	Microelectromechanical Systems.
MPU	Motion Processing Unit.
OpenGL	Open Graphics Library.
OSI	Orthopedic Systems Incorporated.
PCB	Printed Circuit Board.
PDE	Processing Development Environment.
QFN	Quat Flat No-Lead.
RTC	Real Time Clock.
SCL	Serial Clock.
SDA	Serial Data.
SEMG	Surface Electromyography.
SMA	Spine Motion Analyzer.
SMT	Surface Mounted Technology.
SP	Standardized Patient, Simulated Patient.
SPI	Serial Peripheral Interface.
THT	Through-Hole Technology.
TWI	Two Wire Interface.
UART	Universal Asynchronous Receiver Transmitter.

1 Introduction

Spinal injuries are dangerous and can have severe implications. Medical simulation is a technique used for the education and training of medical students. Different patient interactions and diagnostic routines can be reenacted with patient-actors, without harming real patients. Careful handling of patients with pre-existing spinal injuries can be practiced in order to prevent secondary spinal traumatization. Especially critical situations like patient transport profit from these exercises.

In order to evaluate the learning curve of the participants of the simulation, a monitoring device to measure effectiveness of the immobilization and residual movement of the spine is proposed in this thesis. An early prototype is built and tested against the scope of applicability for this use case. This spinal monitor may be used in the future for the training of medical personnel in order to reduce the risks of mistreatments and impairment of spinal injuries during the patient transport.

Out of 942 patients with spinal column injuries hospitalized in Ireland between 1999 and 2003 a neurological deficit was seen in 38 % [1]. 35 % of those patients suffered from a complete neurological deficit, leaving 41 patients paraplegic and 84 tetraplegic [1]. At the Essen University Hospital Neuroclinic, injuries of the spinal column and intervertebral discs is one of the most frequent diagnoses; in trauma surgery, fractures of the lumbar vertebrae and pelvis is the top most diagnosed injury and spinal osteosynthesis among the top five surgical interventions in 2013 [2]. Over the last decade, the number of spinal surgeries per year has quintupled in Germany [3]. Due to the serious consequences of spinal injuries, patients need to be treated extra carefully.

Simulation programs in medical education are often used to train young professionals and medical students to increase their basic competence without prior work experience. These programs range from simple task-trainers to complete clinical routines, teaching both practical and soft skills. Careful interaction with patients suffering from spinal injuries is an important behavior to be adapted by medical students. Simulation provides great options to include different lessons around the treatment of the spine.

Recreational injuries, falls, and motor vehicle accidents in particular expose the spine to high loads, causing spinal trauma in many cases. If not treated with caution, spinal trauma can inflict severe and life threatening complications. In emergency situations, patient transport and treatment on scene have to include fast actions in order to assure the best patient care possible. An inaccurate initial diagnosis with inadequate or non-existent spinal immobilization can lead to further traumatization of the spine.

Medical simulation and education offer possibilities to practice different rescue situations with future and current medical personnel without harming real patients. Interaction with patients with or without suspected spinal injuries can be included in these exercises. During the scenarios, medical intervention and spinal immobilization has to be reviewed in order to provide valuable feedback to simulation participants. A system to measure residual movement of the immobilized spine along with the overall posture will help determine, whether interventions were helpful or not and can assist with the evaluation of the rescue workers in the simulation. A sufficient device for these kind of measurements does not exist at the moment.

The goal of this thesis is to develop a system, that records the spinal posture in medical simulation scenarios and visualizes the data for further analyzation. For this specific task, the system will be split into two major components: A wearable sensor-device to acquire measurements from multiple points of reference on the patient's spine and a computer software to display the patients spinal movement on the screen.

The device fitted to the simulated patient has to be portable and unobtrusive, while covering a great part of the spinal length. The patient must not be restricted by the device in any way, keeping his full range of motion and allowing the execution of the scenario without limitations. The participants of the scenario shall not notice the device at all in order to keep the level of fidelity as high as possible. This is especially important for evaluation of the participants, as the additional monitoring device may direct attention to the spine and manipulate the participant's behavior. In order to ensure spinal monitoring for an entire large-scale simulation, the battery life has to last for several hours, before the data can be analyzed using the computer software.

Beside the main purpose of the computer software, the presentation of the recorded data, the program has to be easy to use and understandable for everyone. The data input from the device has to be interpreted and visualized to simplify the means of analyzation. The visual representation of the data output shall be used during the debriefing and feedback session of the medical simulation scenario, therefore participants of the simulation need to be able to understand and interpret the visualization themselves without prior knowledge of the software.

The software shall help operators to find time-frames of extraordinary spinal activity and guide the viewers to the area where the critical movement originates. This helps users to speed-up the feedback preparation and provide a seamless presentation of the recorded postures. In addition to the visualization of pre-recorded measurements, the software shall be able to display a live feed of the simulated patient's current spinal posture if needed.

In this thesis different possibilities to fulfill these requirement are discussed. Ultimately a distinct solution proposal for a custom spinal monitoring system is presented. This Chapter introduces the purpose and context of this thesis together with a roadmap of the following chapters.

Chapter 2 describes the motivation and background behind the problem. Anatomy and biomechanics of the spine are characterized in detail along with different types of spinal injuries. The idea of spinal monitoring in medical simulation programs is briefly introduced, outlining different types of simulation and the debriefing situation.

Chapter 3 follows up on the opportunities for spinal monitoring in medical simulation and discusses different approaches and possibilities when it comes to spinal monitoring systems. Literature and commercially available systems are reviewed and compared with the requirements of spinal monitoring in simulation scenarios.

A spinal monitoring solution explicitly designed for the use with standardized patients in medical simulation and education is presented in Chapter 4. The previous prototype is described along with the insights from the initial prototype tests. The single components of the system are presented in detail along with the effective changes. Collaboration of the wearable sensor array, data acquisition unit, and visualization software is emphasized as the characterization of the whole system.

Individual sensor characteristics are analyzed in Chapter 5. Sensor and system stability is tested along with the validity of measures and overall system usability. Possible improvements of the system are derived from these tests, before a final conclusion is drawn in Chapter 6 including the prospects of further development.

2 Motivation

In emergency situations fast and efficient intervention of rescue workers, emergency physicians and paramedics is a key component of medical treatment at the scene. Spinal injuries are one of the more critical injuries, that may be affected by incautious actions, causing further traumatization. A spinal monitor can help to show the residual movement of an incorrectly stabilized or non-stabilized spine and assist in the training of medical personnel.

2.1 Background

Emergency medical services are split into two preclinical phases. *Time on scene* covers the on site treatment up to the rescue transport, while *en route* covers treatment in the ambulance (or rescue chopper) during the actual transport. In general, vital signs should be stabilized as far as possible on scene, without unnecessarily slowing down the transport. Especially in English-speaking countries (and with an increasing tendency in Germany) keeping the *time on scene* as short as possible and limiting preclinical interventions to unpostponable lifesaving treatments (also known as the *Scoop and Run* or *Load and Go* concept) is seen as a sign of quality [4].

In traumatic accidents the spine is particularly endangered. Even moving the patient to the stretcher can cause secondary traumatisation [5]. Different options for the preclinical immobilization of the patient's spine exist. Most common are spine boards, providing rigid support of the body during transport and cervical collars, immobilizing the head and neck. Because of the high risk of spinal cord injury with cervical fractures, a cervical collar is recommended for every patient with suspected cervical spine injury [5].

If these injuries are not properly identified or misjudged, risks of additional impairment due to unfit patient handling arise. Even when taking care of preventive measures, only 11 % of experienced professional rescue personnel, emergency physicians and assisting personnel apply cervical collars correctly [6]. In these cases, secondary traumatization cannot be excluded.

2.2 Anatomy of the Spine

With a length of 35 % of the total body height, the vertebral column (or spine) is one of the major skeletal structures in the human body. Its main purpose is to provide stability and support to the whole body by absorbing shocks and carrying the loads of the upper body and distributing them to the pelvis and lower extremities. It further permits and restricts movements of the torso and provides a protective barrier to the spinal cord and nerve roots [7].

The spine is a curved composition of multiple vertebrae [8]. Usually consisting of 33 vertebral segments it divides into four regions with seven cervical (C1–C7), 12 thoracic (Th1–Th12), five lumbar (L1–L5), and five fused sacral vertebrae (Os sacrum), as well as the Os coccygis, formed of four rudimentary vertebrae (see Fig. 2.1) [9]. On the sagittal plane each region features a characteristic curvature: Cervical and lumbar spine are convex anteriorly (lordosis) while thoracic spine and the sacrococcygeal region are concave anteriorly (kyphosis) [7].

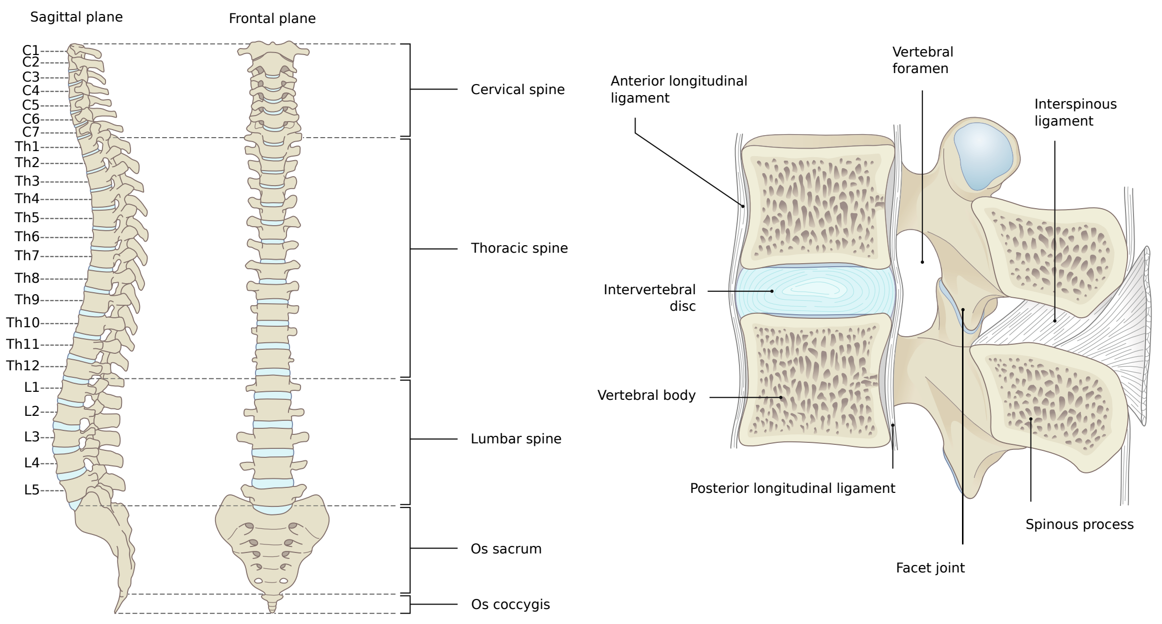


Figure 2.1: Frontal and side view of the vertebral column (left) and the motion segment (right). Modified from [7].

Vertebral body, intervertebral disks, spinous and transversal processes, facet joints as well as ligament and muscle structures make up the most important parts of the spinal elements [7]. Neighboring vertebral segments cranial to the sacrum (except for the junction between the atlas (C1) and axis (C2)) are connected via fibrocartilaginous intervertebral disks [8]. These disks are the main elements for stabilization and shock absorption, causing an even pressure distribution between the vertebral bodies [7].

Along with the adjoining ligaments, two adjacent vertebrae and the intervertebral disk in between form a functional spinal unit (FSU or motion segment, see Fig. 2.1) responsible for the physiological motion of the entire spine. Spinal movements include flection (forward bending), extension (backward bending), lateral bending (left and right) as well as rotation. Particularly during bending movements the intervertebral disks are deformed immensely. Due to the hydrostatic characteristics of the gelatinous core (Nucleus pulposus) and the concentric fibrous outer structures (Anulus fibrosus) pressure is still distributed evenly, even during extreme off-center loading [7].

During dynamic loading, the anterior and posterior longitudinal ligaments are also able to absorb a high amount of energy, while stabilizing the spine and limiting its movement [7]. Further guidance of the spinal movement is managed by the facet joint, which also help stabilizing the spine during rotations; flexion and dorsoventral translation of the vertebral elements are only affected negligibly by them [10]. During extension, the facets joints have to carry an increased percentage of the axial spinal load [11].

Execution of movement is initiated by the musculature of trunk. Flexion and rotation are induced by muscles anterior to the spine (including the abdominal muscles and the iliopsoas), extension by the posterior musculature and lateral bending by lateral muscles. The spinal muscles further ensure the stability of the torso in static and dynamic situations by combining the different muscular movements [7].

2.3 Biomechanics of the Spine

Mechanics of the spine are based on different mechanical concepts that deform or move the spinal components. These can be broken down into two groups: Forces act on a specific point of origin, pressing (compressive force) or pulling (tensile force) an object, while moments (of forces) are the turning effects of forces, acting around a lever and therefore causing rotational movements [7, 12].

For the spine, the instant axis of rotation (IAR) is based on the current position and alignment of the vertebrae and their center of rotation [7, 13]. It is also called neutral or load-bearing axis [7, 13]. The forces acting on the spine are caused by the bodyweight, muscular activity, ligamentous tension and external forces [7].

Different loads are applied to the spine. Static loads are mainly caused by weight loads (of both body parts and external objects), while dynamic loads are associated with complex movement [14].

Static loads on the spine increase from the neck downwards, because of the increasing effective net weight. At the level of the cervical spine, load is primarily determined by the position and orientation of the head. At lumbar spine level the total load is also subject to the position and orientation of the remaining upper body elements [7].

In a neutral, upright position, all motion segments are subject to a forward bending moment, which has to be compensated by the musculature. In the same position, the axial load on the lumbar spine is in the same order as the total body weight. This load is increased during flexion and reaches its maximum when lifting heavy objects distant to the body [7].

Dynamic loads induce higher forces and moments depending on the speed of the movement the resulting load transfer rates. The axial load on the lumbar spine for example increases to 1.5 x of the body weight during normal and 2 x during fast walking. This characteristic also affects the nature of spinal injuries, causing more dislocations and bone fractures with higher speed of movement [7].

2.4 Types of Spinal Injuries

Force effects during trauma, singular overstrain or prolonged overload are the main causes for spinal injuries [7]. Around 30 % of spinal fractures are induced by falls, while 20 % are caused by individual actions such as raising a weight or flexion [15]. The remaining 50 % can be associated as pathologic or fatigue fractures due to single events with poor bone quality or unilateral loading for extended periods of time [7]. Especially fatigue fractures, initiated by microscopic damages to the bone (so called microfractures), are hard to trace to a unique event [7, 16].

In general, traumatic injuries of the spine can be classified into three categories, based on their mechanical foundations: Compressive force, causing impaction, split and burst fractures, tensile force, causing transverse disruption, and axial torque, causing rotational injuries (see Fig. 2.2) [17].

Vertebral body compression fractures (type A) are caused by axial compression of the vertebral body [17]. These fractures primarily occur at the junction of the thoracic and lumbar spine (T11–L1), affecting the anterior and middle column [7, 18]. Type A injuries can be further split into three groups: Impaction fractures (A1) are caused by compression of the spongy bone rather than fragmentation, split fractures (A2) split the vertebral body into multiple parts, filling sizable gaps with disc material and burst fractures (A3), causing partial or complete comminution with a centrifugal displacement of the fragments [17].

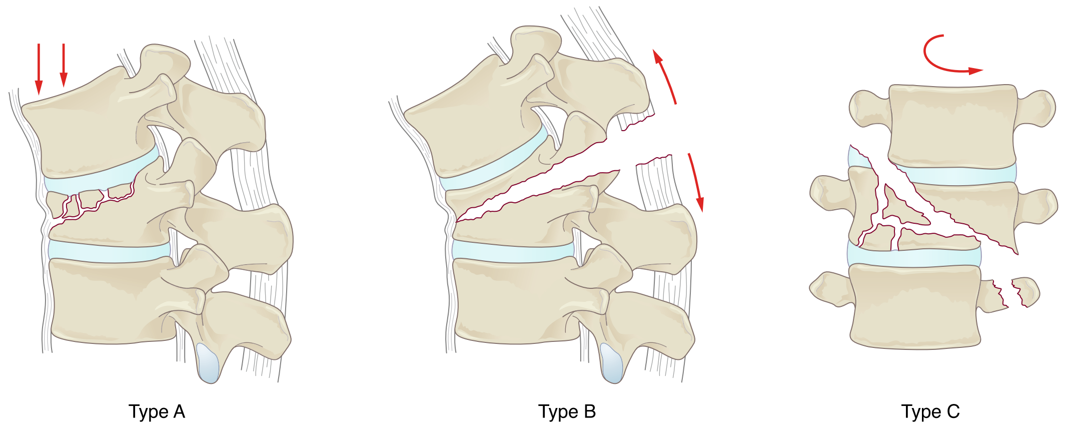


Figure 2.2: Classification of spinal injuries according to Magerl [17]: Compression (type A), distraction (type B), and rotation (type C) injuries [7].

Element injuries with distraction (type B) are caused by transverse disruptions of the spinal columns. Flexion distraction causes ligamentous (B1) or osseous (B2) posterior disruption and elongation, while hyperextension initiates anterior disruption and elongation (B3) [17].

Anterior and posterior element injuries with rotation (type C) are caused by complex injury patterns. They include type A injuries with superimposed rotation (C1), type B injuries with superimposed rotation (C2), and rotation-shear injuries (C3). Rotational fractures rank along the severest injuries of the spine, often causing neurological deficit and a high degree of instability [17].

A complete overview of groups, subgroups, and specifications of type A, B, and C injuries according to Magerl [17] can be seen in Appendix A.

The load-bearing capacity of the spine is highly dependent on the adopted posture, changing the lever radius between the spinal sections [7]. During hyperextension, the spine is able to withstand a 50 % higher axial load before fracture compared to the erect posture and a 100 % increase compared to a flexed posture [19]. On the dorsoventral axis, loads have the tendency of adjusting the spine by resulting in compression and tensile force [7].

These kinds of loads are typical in rear-end collisions, where both hyperextension and hyperflexion are experienced in rapid succession [7]. Whiplash is caused by the resulting compression and abnormally located axes of rotation in the lower cervical segments, leaving the spine vulnerable to injury [20]. The cervical spine features a wide range of motion, but offers limited stability [21]. Even at lower speeds the resulting acceleration of the head relative to the trunk can overload the cervical spine [7].

Clinical stability is defined as a considerable reduction in the ability of the spine to maintain its alignment without causing neurological deficits, deformities, displacements, and incapacitating pain [22]. Injuries, degenerations, and diseases may limit the spinal stabilization system [23]. Neural injuries can occur, when the spinal canal is encroached by dislocated spinal fragments or translational displacement [17]. These unstable neurological injuries are primarily treated with surgery, while other instabilities might be subject to more conservative therapies [21, 7].

Especially fractures and instabilities of the cervical spine require immediate and thorough therapies to prevent severe and life threatening complications. Subsequently spinal immobilization of the patient's cervical spine before the surgical stabilization is of great importance and a cervical collar might be necessary in a short- or mid-term period after the surgery to prevent careless or forced movement [24].

2.5 Spinal Monitoring

During patient transport inadequate or non-existent spinal immobilization can have critical impact on a patient with pre-existing spinal trauma. To prevent secondary traumatization an accurate initial diagnosis and careful patient handling have to be regarded.

Spine monitoring can help demonstrate critical movements and provide feedback to rescue workers or healthcare professionals. This feedback can be used as simulated restrictions lower than the actual threshold for permanent or life-threatening injuries.

An otherwise invisible system can provide additional data for the evaluation of medical personnel in training exercises. In these cases, a small set of sensors can be used to record relative movements of the spine, without revealing the true nature of the test. Simulation programs are common in medical education and already prevalent in many university hospitals.

2.6 Simulation Programs

Simulation is a technique to create fully interactive guided experiences which replicate significant characteristics of the real world [25]. In the clinical and health care context it can be seen as the attempt to recreate one or more aspects of medical practice [26]. It ranges from simple physical tasks (e.g. using oranges for intramuscular injection practice) to complex clinical environments, where participants easily suspend skepticism or disbelief and fully immerse into the simulated setting [25, 26].

In medical education, simulation plays a significant role for practical training. Medical and non-medical skills, such as interview techniques, can be acquired individually and without real patient contact using simulation [27]. With a focus on education, medical simulation is generally split into two phases: The scenario, involving the performance of the participant, and the debriefing, covering a reflection of the performance [28]. In comparison to real-world experience, simulation offers the possibility to provide standardized, repeatable and testable course contents fixed to the curriculum [27].

After initial vocational education, simulation can be used in continual training for clinical personnel and teams. That way simulation can be a key enabling technique for optimization of safety and quality in health care [25].

Facilities for medical simulation can range from simple training rooms of medical departments, clinics or even commercial hosts to more complex interdisciplinary simulation centers run by a hospital or a medical faculty. Generally the facilities consist of at least three rooms per scenario, used for the simulation, debriefing, and as control room. Larger facilities can accommodate multiple courses at once and provide different rooms with medical equipment for distinct settings (e.g. intense care units or surgery simulators) [27].

Many institutions offer a tight integration between skills training and simulation. *Skills Labs* aim to teach medical students practical skills needed for the clinical routine. Compared to simulation centers they offer a wider range of basic competence training without building upon work experience. Standardized and repeatable training in a reproducible setting is the key aspect of skills labs. Apart from practical skills, it is also possible to teach soft skills (e.g. patient communication) by using standardized patients and patient-actors. Today *Skills Labs* are available nationwide at almost every medical faculty in Germany [29].

2.6.1 Types of Simulation

Simulation can take many forms, from the discussion of a case study to the use of high-fidelity simulators. The main purpose of all the different types of simulation remains the same—providing an interactive educational experience with performance-based objectives. The most basic form of simulators are mere task trainers, allowing students to develop the skills necessary to perform certain procedures, such as intubation or suctioning. For more complex training situations full-scale patient simulators or standardized patients can be used [30].

Mannequin-based Patient Simulation

Full-scale mannequin trainers are highly sophisticated and realistic models of real patients [31]. They offer a wide range of possible procedures with interactive physiology and complications [32]. High-fidelity mannequins are either operator-driven, relying on an instructor to provide intervention feedback, or autonomous, utilizing mathematical algorithms for the computation of physiological changes [33].

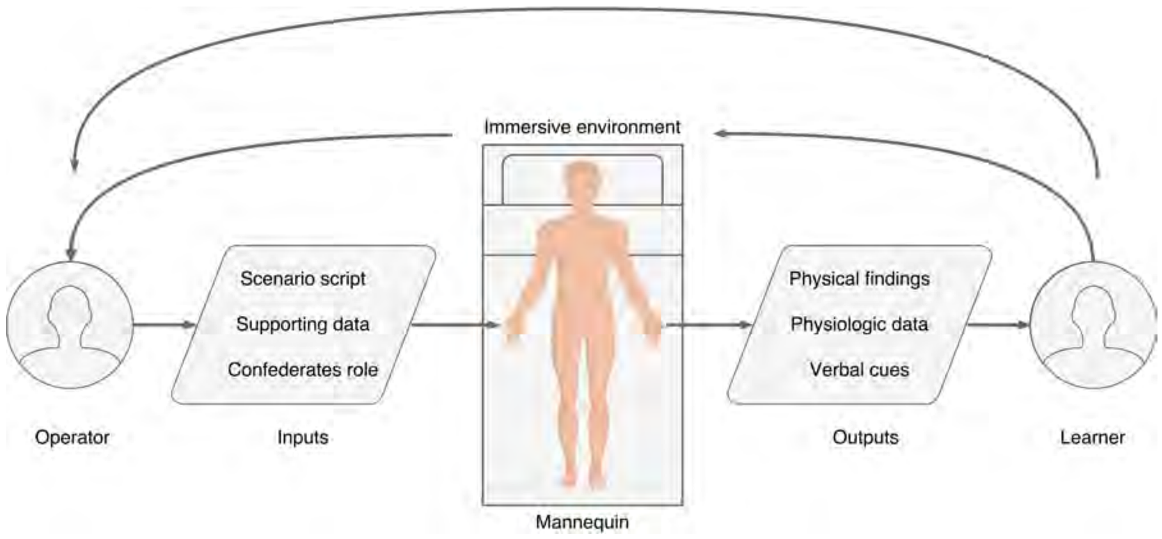


Figure 2.3: In an immersive environment, mannequin-based simulators can be seen as interface between learner and operator. Modified from [33].

A simplified overview of the mannequin-based interaction between operator and learner is shown in Fig. 2.3. The operator provides the scenario script and supporting data. Based on the physiological input, the mannequin reacts to different situations to reflect the pre-set disease. During patient examination the mannequin behavior helps the learner to make the diagnosis and start a treatment. Interactions between learner and mannequin are supervised by the operator, who can draw conclusions from the concurring actions to modify the input. This way, the mannequin can react to the learner to create a fully immersive environment with high fidelity.

Fidelity generally describes the level of reproduction of the real-world environment and highly depend on the intended use of the simulator. Most mannequins feature realistic breathing sounds and movement of the chest, heart sounds (often with palpable pulse), eye opening, blinking, and closing dependent on consciousness, and the ability to speak (or passthrough speech from the operator) adding to the fidelity of the system [33].

Realistic properties are also met for invasive procedures such as endotracheal and nasotracheal intubation, urinary catheter placement, chest tube insertion, and other interventions. Additional sensors are often built into the mannequins to adjust vital signs and parameters according to the received treatment in fully autonomous or operator-assisting patient simulators. Some even include drug recognition systems and can react automatically based on the delivered dose. Various add-ons to facilitate distinct conditions such as amputations or special diagnostic examinations like sonography are available [33].

Standardized Patients

When the term simulation is used in a medical context, it is often prematurely narrowed down to technological simulators, which allow the portrayal of different clinical scenarios with a more or less sophisticated replication of anatomical features [34]. While these simulators offer a wide range of possible applications, they are restricted to very limited patient interaction. A simulated or standardized patient (SP) offers the possibility to add a *human touch* for students into the simulation [30].

Standardized patients are carefully selected and trained actors to portray patients. In simulation they are used to teach and evaluate healthcare professionals and students by acting like a real medical patients with relevant history, physical findings, and affect. Typically, multiple actors are trained to portray the same medical conditions in a standardized way to allow for multiple repetitions and training of numerous trainees in a limited time period [35].

Using standardized patients for educational purposes helps raising confidence in students and ensuring a more professional approach to patient interactions by providing patient experience outside of a real clinical setting [30]. Other notable advantages include the repeatability of encounters with standardized and flexible case presentations and the ability to provide feedback and evaluation for student performances based on different competencies [35]. While physical and mental resilience is limited for real patients, SPs can present the same case over and over again to different groups of students without emotional drawbacks [34].

Disadvantages and challenges for the use of standardized patients include limited fidelity. A patient-actor can never be as authentic as a real patient. Certain physical symptoms and conditions, such as heart sounds and palpatory findings are difficult to simulate using SPs (although hybrid simulation may overcome these problems). Additionally, SPs require payment and, depending on complexity and standardization of their portrayed roles, profound training. Special expertise and infrastructure for recruitment, training, properties, supervision, and administration might also be necessary [34, 35].

Mass Casualty Incidents

A mass casualty incident (MCI) is a situation, where the type, severity, and number of casualties exceed the available logistical and medical capabilities necessary for optimal care [36]. Typical examples are natural disasters (e.g. earthquakes) or large scale accidents (e.g. train collisions) involving many injured people. In situations like these, the main goal is the effective management of the limited resources for the greater good, focusing on the whole population, rather than the individual [36]. Patients have to be prioritized according to their need for immediate care and severity of the injury, while also taking other factors, like the total number of casualties, availability of assets as well as distances and vacancies of the surrounding hospitals into account [36, 37].

Simulation of these high-intensity, low-frequency events provide the opportunity to create interdisciplinary team training exercises, bringing together healthcare providers from different specialities from fire and rescue services to medical personnel [38]. Furthermore, it allows the observation of limitations in existing clinical protocols and improvable bottleneck situations [39]. On the downside, large-scale simulations require advanced planning and management as well as many resources (for both operators and actors as well as participants). Typically, these scenarios can include both standardized patients and human patient simulators and end in an interdisciplinary feedback session.

2.6.2 Debriefing and Technical Equipment

Debriefing is one of the key components—if not the most important part—of realistic simulator training. The aim of these, often video-based, scenario discussions with all the participants right after the end of the simulation session is to reprocess the actions taken and therefore enhance the self-reflection in the participants. Good debriefing should include positive reinforcement while illustrating underlying principles leading to misconceptions and mistakes without concentrating on the error itself. Furthermore, audio-visual equipment can be used to assist with the debriefing sessions [40].

Most simulation centers today offer the possibility to record video with multiple cameras during the simulation and analyze the recording during the feedback session [41]. Besides audio and video recording, much other technical and non-technical equipment can be used to log simulations and other training activities. The control room is often fitted with a one-way mirror allowing the evaluators to overview the entire scene without interruption. In outside scenarios without control rooms, the evaluators might be standing by, taking notes or recording the scene using battery powered equipment.

Computer-based debriefing systems often include multiple options to integrate audiovisual recordings and further data recorded by other event-loggers such as patient monitors or human patient simulators. Instructors can add comments, annotations, and notes for specific periods of time as a guide for the succeeding feedback session. A spine monitoring system could be used as an additional event-logger for critical situations not otherwise noticed during evaluation.

3 Opportunities for Spinal Monitoring in Patient Simulation

Patient simulation offers important advantages and opportunities when used for analyzation of performance of health care professionals compared to genuinely hazardous events. These training exercises offer realistic scenarios, while sparing real patients but also immersing the medical personnel into the events. Unlike real events, these simulations can be pre-planned and offer better means for evaluation.

3.1 Advantages of Simulation

As already established in Chapter 2.5, with relevance to the spine, patient transport is one of the most critical stages in the chain of rescue. For both patient simulation and real medical incidents, spinal monitoring in these situations could be applicable. With regard to evaluation and analysis, the use of simulated patients is preferable to real patients for multiple reasons:

Patient simulation can be arranged to feature specific injuries, without having to wait for a *suitable* patient. Blunt trauma for example can be simulated to evaluate medical personnel based on their response. A scenario like this can be repeated multiple times, without harming a real patient or neglecting the patient's treatment. The monitoring device can be fitted to the simulated patient prior to the start of the simulation. With real medical incidents the health care professionals would have to fit the device to the patients themselves if they suspected a spinal injury. However, injuries that remain undetected are the ones where monitoring would be especially important, since no other precautional measures are taken.

In the ideal case, additional technical equipment of the simulation can be used together with the spinal monitoring device. Most medical simulation facilities already use video recording for later debriefing and feedback. Synchronization of different recordings, be it audio-visual or from any other event-logger, can help to interpret the data and open up new possibilities to trace back and evaluate certain crucial events. A good implementation of a spinal monitoring system made for medical simulation should integrate seamlessly into the pre-existing simulation setup.

3.2 Implementational Possibilities for a Spinal Monitoring System

Medical monitoring systems serve the purpose of observing patient conditions. In a clinical environment bedside monitors are most commonly used to record the patient's vital signs. These monitors usually consist of at least one sensor, a processing unit, and a display. A spinal monitoring system shall serve the purpose of observing the patient's spine with regards to flexion, torsion, and overall posture. Appropriate sensors are needed along with reasonable means for processing and display.

For many years, posture analysis was based on visual observation, using different methodologies including drawings, photographs, and the posturegram, a categorization technique for body posture using multiple points of reference [42, 43]. Continuous development brought computer-aided analyzation with video feeds, but still relied on observational techniques [43]. Supplemented by passive markers these techniques are still widely used in medical research and diagnostics.

Passive optical motion capture systems use highly reflective markers placed on the patient's body. Multiple cameras with light-sources placed around the subject record a video feed of the moving patient. The motion capture data is generated in post-processing by triangulation of the marker positions. Alternatively active markers, using LEDs to emit light may be used for automated marker identification and real time motion capture [44].

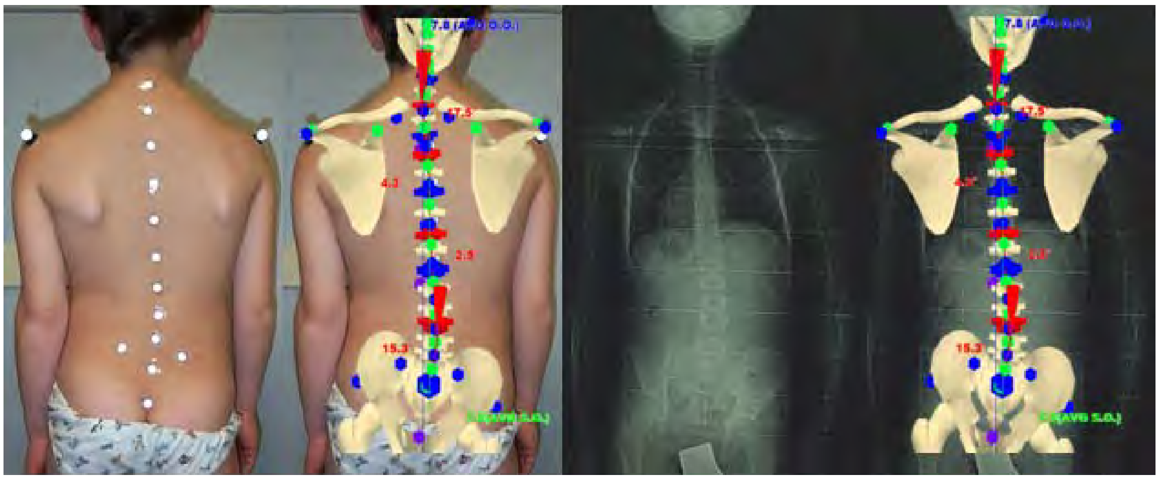


Figure 3.1: Reconstructed skeleton model from passive optical markers in direct comparison with x-ray imaging [45].

With optical techniques no direct measurements are possible. Any output will be calculated from the visual landmarks to comply with the desired output. Different approaches are possible to obtain direct measurements of the spine. Simple strain gauges for example can be used to measure elongation and contraction of the skin on the patient's back and therefore calculate the current flexion of the back. Strain gauges are cheap and simple but mostly meant to measure very small changes. The difference in length of the human back between maximal forward flexion and a regular standing position can exceed the 10 cm mark. Additionally these measurements may be inaccurate due to lateral bending, which might interfere with the forward bending measurements.

Direct angular measurements of body segments can be obtained using mechanical devices, such as goniometers, torsiometers, or inclinometers [46, 43]. In medical context, these devices can be used for the measurement of joint angles and total range of motion (ROM), focussing on a single body segment. Larger devices specifically made for spinal measurements, may feature a combination of multiple goniometers. Electric implementations of these devices allow for continuous real-time posture analysis [43].

Mechanical motion tracking can be split into two categories: Body-based and ground-based systems. Body-based systems utilize a mechanical suit or exoskeleton worn by the user. Electro-goniometers between the moving parts of the exoskeleton take angular measurements used to determine the body posture. Since the measurements are obtained externally, the offset of the center of rotation causes a slight deviation. Additionally, weight and size of the complex framework often compromise the user's full range of motion and restrict a complete immersion into a regular workflow [44].

Ground-based mechanical motion tracking systems are used to measure location and orientation of single body segments. The analyzed body segment is connected to a fixed station on the ground using multiple rigid poles with joints in-between. Position and orientation can be calculated from the joint angles and pole lengths. These systems usually provide very limited range of approximately two meters but feature a very low latency, often used for precise head tracking [44].

Other techniques for motion tracking systems include electromagnetic and acoustic trackers. In electromagnetic systems pulsed magnetic fields are emitted [43]. Small sensors are mounted on body points, containing perpendicular coils to induce a current based on the position within the electromagnetic field [44]. With six degrees of freedom (DOF) several joint movements can be detected simultaneously but the total range is limited to the magnetic field [43].

With acoustic trackers, an ultrasonic emitter is strapped to the subject along with microphones to pick up the sound waves. Distance between emitter and receiver can be determined by the duration of the traveling signal or the phase-shift between emitted and received signals. Additionally velocity of movement may be determined by the received frequency according to the Doppler effect. Ultrasonic systems can be

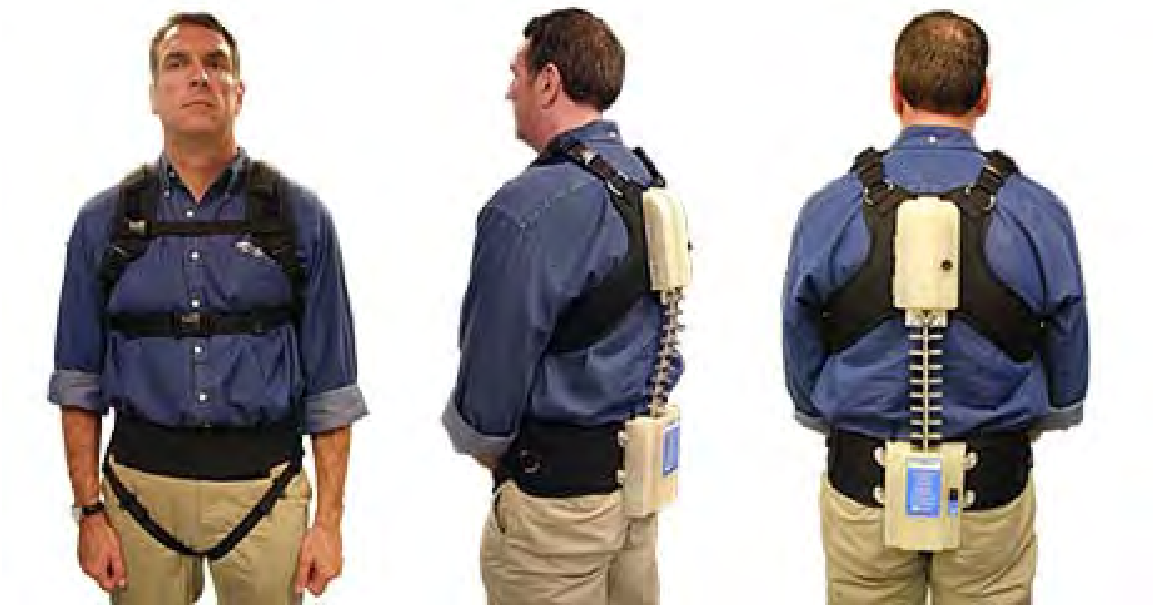


Figure 3.2: Body-based mechanical motion tracking of the spine: The AccuPath Industrial Lumbar Motion Monitor [47].

worn on the body, without the need of external points of reference, such as cameras, but are very sensitive to noise [44, 43].

In recent years inertial sensors have been added to the body posture and motion analysis systems [44]. An inertial measurement unit (IMU) typically consists of multi-axis microelectromechanical (MEMS) gyroscopes and accelerometers. Using microscopic vibrating structures, these IMUs are able to to determine the rate and magnitude of rotation. Acceleration and orientational output can be used to calculate the body posture. These types of sensors are small and effective but tend to drift without external reference [44].

Increasingly, the combination of observational methods along with direct recording methods are used for body motion studies [43]. By adding measurements into the combination, that cannot describe the kinematics on their own but respond to a very specific part of the motion sequence, such as muscular activity, a complex morphological characterization is possible. Integrating data received from force platforms, surface electromyography (SEMG), and foot pressure maps, forces, torques, and electro-muscular activity may be added to the kinematic measurements [45].

A hybrid solution of small body-worn sensor system and an optical analysis without additional markers may be appropriate for medical simulation, since the sessions are recorded anyway. Since a complex morphological characterization is not needed in these cases, the sensors on the body could assist with the task of tracking significant time frames. These timeframes may be analyzed further and in detail using the observational methods. A simple visual representation is needed to show the flexion, torsion, and overall posture. This monitoring could be integrated into a computer-based debriefing system or run as a stand-alone application.

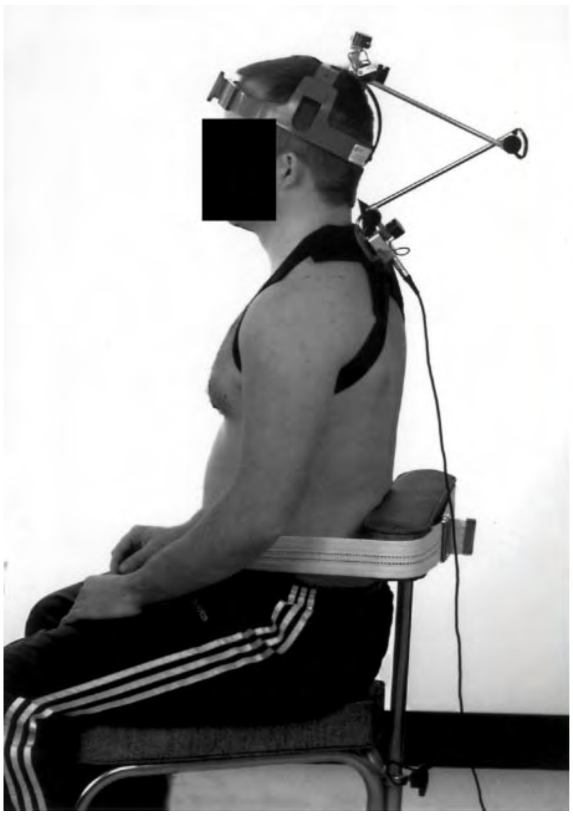
3.3 Currently Available Spine Monitoring Systems

A wide variety of spinal monitoring systems already exist for commercial and medical applications, with even more systems in use for overall motion capture and other biomechanical analyses. The most used techniques for industrial and medical use are based on observational methods. Passive optical motion capture is a very versatile and common type of motion analysis used for life science and medical application but also in the entertainment and engineering industries. In Fig. 3.1, the analyzation of a deformed spine of a scoliotic patient with a leg length discrepancy is shown using optical motion capture in comparison with radiographic imaging [45]. Commercially, theses systems are sold under brand names like Vicon Motion Systems or NaturalPoint OptiTrack.

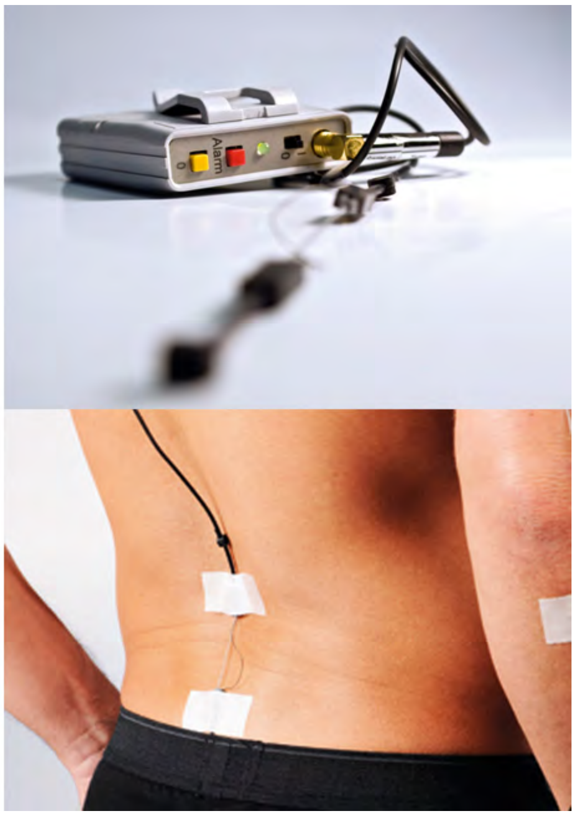
Geared towards the need of spinal monitoring, some mechanical devices are available on the market today. The Lumbar Motion Monitor (LMM), is a device for biomechanical assessment of low back disorder (LBD) risk in the workplace [48]. The LMM is an exoskelton worn by an employee, while doing repetitive manual materials handling tasks [49]. Lumbar motion is detected using a tri-axial electrogoniometer [49]. Newer versions of the LMM are slightly smaller than the model shown in Fig. 3.2 but still feature the characteristic exoskeleton look.

Another well-tested and commercially available mechanical motion tracking device is the Orthopedic Systems Incorporated (OSI) Computerized Anatometry (CA) 6000 Spine Motion Analyser (SMA), which allows natural, unrestricted movement of the spine while obtaining simultaneous measurements of all three planes of motion [50]. The OSI CA 6000 SMA is built similar to a ground-based mechanical motion tracker, strapped on the patients chest instead of the ground. A head strap connects using two linked rods, while goniometers measure their joint angles internally. Fig. 3.3a shows the patient set-up for cervical range of motion measurements.

Smaller devices to monitor posture and log motion patterns are available with the Sels Instruments BodyGuard system. The BodyGuard is based on strain gauge measurement and consists of a small probe and a portable enclosure for the signal processing and data logging unit. Its main advantage is the minimal impact on the user's movement and a very simple setup. Customizable alerts can be created and reported [51]. Fig. 3.3b shows exemplary usage and the data recorder of the BodyGuard system.


[redacted]

(a) Cervical ROM measurements using the CA 6000 SMA [50].

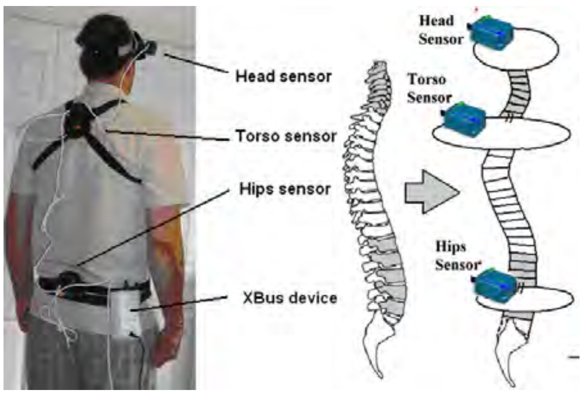


(b) The BodyGuard portable enclosure and sensor [51].

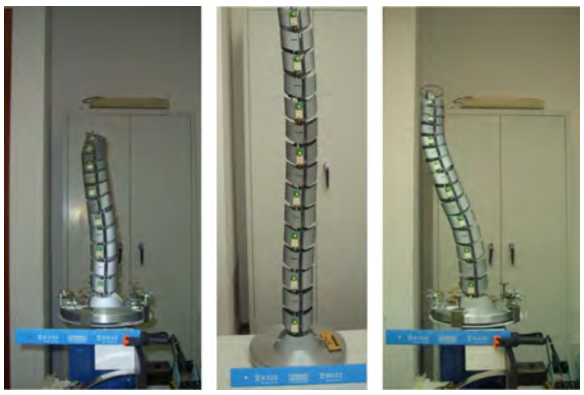
Figure 3.3: Two examples of commercially available spine monitors.

Research on spinal measurements using IMUs are currently done in different approaches. Goodvin et al. preset a method using three IMUs located on head, torso, and hip with a connected processing unit [52]. A demonstration of the device worn by a test subject is shown in Fig. 3.4a. Wong and Wong show, that a similar approach using three sensors is possible to detect postural change in sitting positions [53]. In a further study Lopez-Quintana et al. developed a small, eight sensor setup using triaxial accelerometers [54]. They show a good attempt at a small and reliable sensor system but do not refer to real life usage, wireless battery performance or long-term measurement. Their laboratory test set-up is shown in Fig. 3.4b.

Commercial some motion capture and analysis systems using IMUs are available for different use cases, but mostly for non-medical use. IMU systems by Xsens Technologies, like the MTw Development Kit and MTw Awinda can be used for medical purposes but run using a complex software or need additional development for user-specific tasks [55]. Similar products are available in the InterSense InertiaCube product line by Thales Visionix [56] and the I2M product family from NexGen Er-



(a) Three sensor IMU-based monitoring solution by Goodvin et al. [52]



(b) Laboratory test showing an eight sensor setup by Lopez-Quintana et al. [54].

Figure 3.4: Two examples for the use of IMUs for spinal posture analysis in research.

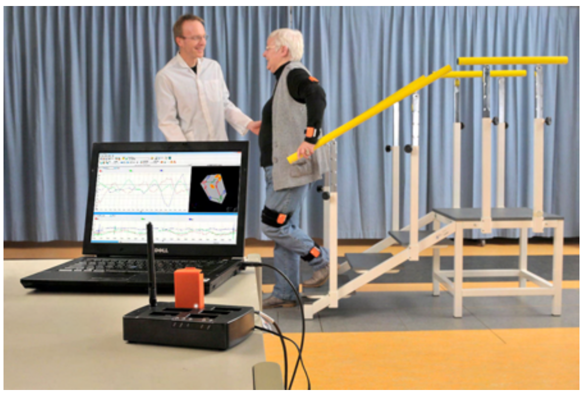
gonomics [57]. These systems are not specific to spinal posture, but can be adapted for most use cases.

Additional devices, which analyze the spinal posture for precautionary or therapeutic training are also available on the market today. The personal posture trainer by UpRight Technologies is aimed at office workers to prevent slouching by providing vibrational feedback [58]. For physical therapy at home, the Hocoma Valedo provides a gaming like experience to perform safe therapeutical exercises [59]. An overview of the described commercially available IMU systems for motion tracking and posture training and is shown in Fig. 3.5.

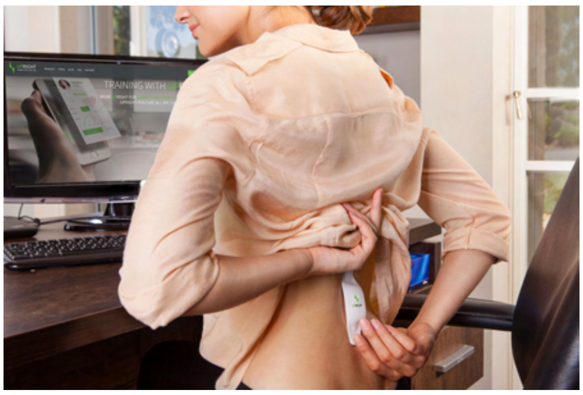
3.4 Device Requirements

A spinal monitoring device for the use in patient simulation and medical evaluation has a slightly extended set of requirements than other spinal monitoring devices, which is not met by any of the available systems. Most importantly, it has to be small and unobtrusive to maintain the fidelity within the simulation. The device has to fit comfortable to the standardized patient's back, without affected the ROM in any means. All aspects of the simulation have to be realized without any impairment from the device, even if the patient lies flat on his back.

When used for educational purposes it is important, that the subjects may not exactly know what will be part of the evaluation. Optical markers or bulky attachments on the simulated patient's back, for instance, will impair the fidelity of the simulation and attract the subjects immediate attention to that area, suggesting a prioritization, which otherwise might have been chosen differently. Keeping the monitoring system


(a) Xsens MTw Development Kit [55].

(b) NexGen Ergonomics I2M [57].


(c) UpRight Posture Trainer [58].

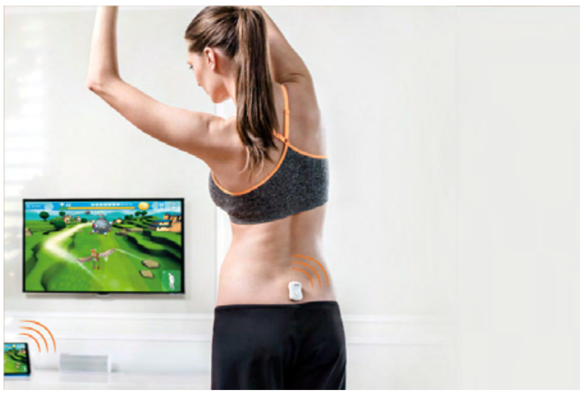

(d) Hocoma Valedo [59].

Figure 3.5: Different IMU systems for motion tracking and posture training.

small will be advantageous in these cases. Additionally the system might be actively hidden beneath kinesiologic tape or bandages to increase the chances of remaining undetected by the subjects of the simulation.

Fixation of the monitoring unit to the spine has to be non-invasive, the easier the better. Taping the unit to the patient's back along the posterior median furrow seems to be the most appropriate solution. When taping, the sensor has to stay close to the body without slipping out of position and the small unit has to be stable enough not to break, when removing it from the aggressive surgical tape.

Out of the available systems, only the BodyGuard system is small enough to be placed on the patients back without any noticeable protrusions. While this could be used in combination with markerless video recording to find the time frames of interest, the system would not suffice for a stand-alone application. A detailed breakdown of vertebrae or vertebral segments and the type of is only possible with many units fitted to the patient.

The small IMU-based systems by Xsens and NexGen Ergonomics feature better possibilities for data analysis and utilization but are noticeably larger than the BodyGuard sensors. For sufficient distinction between vertebrae however, multiple sensors on the spine are needed, increasing the obtrusiveness of the system. The UpRight system is supposed to be worn without being noticed but offers an insufficient amount of reference points on the spine and uncertain medical relevance. A possibility to use any of these IMU-based systems in supine position comfortably seems unlikely but smaller devices with newer MEMS technology could improve on these features. Clear advantages of these systems include the built-in wireless technology for computer analysis of the data.

For data recording and power supply of the unit, the patient cannot be tethered to a computer because that would compromise the fidelity of the simulation. A battery-powered (or batteryless) solution is needed, while data can be transmitted using a wireless technology (such as bluetooth) or, since a live presentation of data is of secondary importance, recorded internally to be read from the device after the simulation. In these cases battery and data storage have to be large enough to allow for a continuous data recording of multiple hours to ensure the complete coverage of a large-scale simulation.

A computer software is needed for data analysis and presentation. During the debriefing a visualization of the spinal posture can be used to show the participants of the simulation any type of movement during the rescue mission. The presentation of the data shall be easy to understand for the participants to interpret the outcome themselves without prior knowledge of the software. A visualization of the occurring spinal posture for example is easier to understand than a series of different plots and graphs. For the operator of the software additional options should be available in order to help with the identification of time-frames of extraordinary spinal activity. Similarly, a visual reference to indicate the origin and severity of the spinal movement can help with the data analysis and selection of time-frames.

4 Experimental Design

The device requirements stated in Chapter 3.4 will be implemented using multiple microelectromechanical gyroscopic sensors to calculate orientation along with dedicated microcontrollers for data processing. The acquired data will be presented using a custom 3D visualization software. This thesis is a continuation of the work of Bandow et al. [60], where an early prototype was built.

4.1 Previous Prototype

The prototype spinal monitoring system built by Bandow et al. [60] consists of three main parts: A data logging unit, sensor array, and a 3D visualization software. At the heart of the system, the sensor array is monitoring orientation at multiple positions.

For monitoring purposes four 6-axis gyroscopic sensors (MPU-6050 by InvenSense) were used, combining accelerometer and gyroscope onto a single chip. These sensors feature a programable gyroscopic full-scale range between ± 250 and ± 2000 dps (degree per second) and an embedded temperature sensor for automatic temperature compensation. The digital motion processor (DMP) on the chip itself is able to convert the raw measurements into different formats (such as quaternions or Euler angles). Data from the sensors can be accessed using the I^2C protocol. The four sensors were placed on individual breakout boards and connected at different lengths of a 20-pin flat ribbon cable to be placed on different locations on the patient's back.

The data logging unit was built with a Teensy LC-based micro controller unit (MCU). A custom circuit board was printed to fit the Teensy, a battery charging controller with buck-boost converter and additional components. A 20-pin ribbon cable connector was used to ensure connection to the sensors, where each sensor would access an individual wire for voltage, ground, data, clock as well as I^2C address selection. A microSD card slot is wired to the Teensy for data recording purposes. Additionally three status LEDs are used to indicate activity. (Before testing a real time clock (RTC) module was also added, which wasn't planned originally.)

A 3D printed case was used to accommodate these components along with a 2600 mA h lithium-ion battery. At 90 mm × 23.6 mm × 45 mm (not much larger than half a pack of cigarettes) it is easy to carry in most pockets. A systematic overview of the data-logger and its fundamental components is shown in Fig. 4.1.

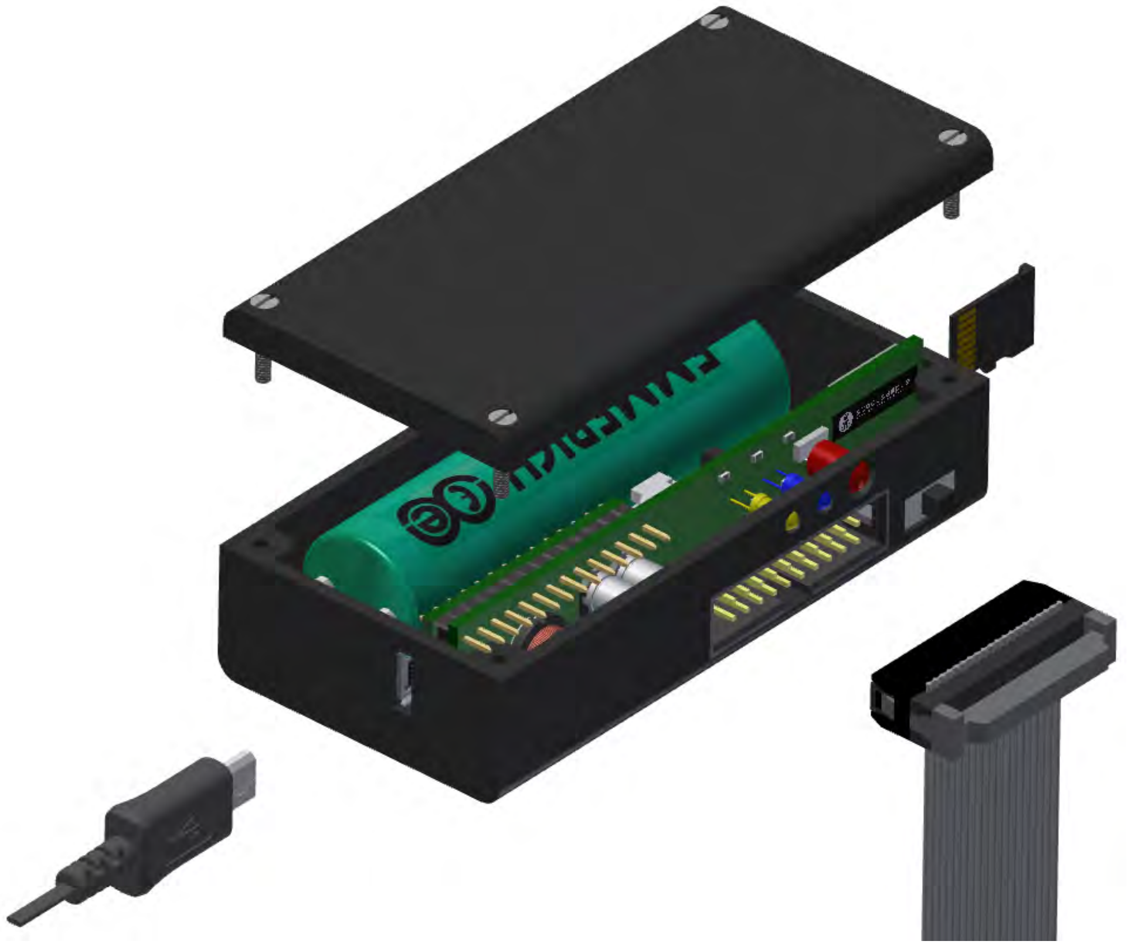


Figure 4.1: Rendering of the data logging unit with Micro-USB, microSD, and 20-pin flat cable sensor connector.

The main data logging unit allowed the electrical connection of up to four sensors, while the sensor's hardware restricted the simultaneous usage of more than two sensors at a time, as the sensor is only accessible using two dedicated I^{2}C addresses. Different software workarounds, like continuous switching and reinitialization of sensors were tried to circumvent these restrictions, without finding a promising solution. It was noted, that faster switching between sensors was possible if unprocessed raw data was received. Further development and testing was done, acquiring raw data of two sensors only.

In addition, a computer software was designed to act as a framework for displaying the acquired data using a 3D model of a human skeleton without extremities as visual reference for the movement. The software was able to display different postures on the screen but not able to recreate these from the sensor output.

4.1.1 Insights from Prototype Testing

The first conceptual prototype was tested during the Emergency Medical Summer Academy (Notfallmedizinische Sommerakademie) at the Essen University Hospital SkillsLab. In two different scenarios, repeated simulated motorcycle crashes and a mass casualty incident, the two sensors were affixed to the upper cervical and thoracic spine of volunteering simulated patients, while the data logging unit remained in the front jeans pockets. The tests were mainly based on functionality and durability of the entire system in close to real life situations and lasted two days with a total runtime of around 8 hours.

Mechanically, the logging unit was able to withstand all critical events during testing, including multiple falls and bearing the weight of the stretcher with patient on its case. The soldered connection between sensor breakout board and cable in combination with the medical tape used to fixate the sensors proved to be a source of trouble when removing the sensors from the simulated patients. On the electrical side, the prototype design showed multiple flaws, including drop of potential in some situations causing reboots and loss of connection to the SD card. These were mainly caused by short circuits due to the limited space inside the housing (after the RTC module was added) and an unsuitable microSD card connector. Sensor overflows limited the effectiveness and resulted in slow downs or had to be eliminated by reinitialization. The software was able to display different error codes on the LEDs and reboot the device if these weren't resolved within a predefined time frame.

In the first scenario, 20 data files were recorded. The longest recordings lasted around half an hour, while the shortest was just a couple of seconds. The second scenario produced no recordable data after breaking the connection of one of the two sensors on the end of the first day. Further testing revolved mainly around mechanical and wear influences. After the scenario the system was still able to run, showing the error code for misconnected sensors.

4.1.2 Necessary Changes

During prototype testing further system issues have emerged, that need to be changed before finalization of the system. Besides unusable sensor output, the first prototype has several teething problems, such as weak sensor wiring and unreliable MicroSD card connections. In some cases error handling is over-prioritized, causing unnecessary delays or termination of data acquisition. These problems have to be overcome.

The main goal of this thesis is to implement a complete prototype, that allows for a correct data acquisition procedure for four or more simultaneously running sensors. The sensors shall output pre-processed data, that can be used directly with the 3D visualization software. The data analyzation software shall be able to work with both pre-recorded and live sensor data in order to recreate the spinal posture on screen.

The second prototype developed in this thesis will feature a complete software overhaul on the data logging unit and the sensor array, while keeping most of the already acquired hardware components. Some hardware changes are necessary for the sensor array, in order to allow data recording of more than two sensors. An additional processing unit per sensor will be used to receive, pre-process, and forward the sensor data to the data-logger.

4.2 Collaboration of the System

As previously mentioned the spine monitoring system consists of three major parts. The data acquisition unit, a sensor array, and a 3D visualization software. The data acquisition unit (or data-logger) and sensor array are the essential parts, worn by the (simulated) patients. A direct wired connection between sensor array and data-logger is used to supply the sensors with power. Communication between the two is based on the I^2C protocol.

The visualization software is used to display a simple 3D model of a human torso skeleton with head. Movement and alignment of the sensors are mirrored to the screen via a USB-based serial communication or retrospective read-outs from the data-logger's internal storage.

From a software point of view, both sensor array and data acquisition unit are programmed in C++ using the Arduino Integrated Development Environment (IDE). The visualization software is written in Processing, a programming language built on Java with simplified syntax, especially created for 2D and 3D visualization and animation.

4.2.1 Sensor Array

For this thesis, a total of five sensors were built. The structure of the individual sensors is described later on. The sensors are created with a modular approach in mind, making it possible to daisy-chain multiple sensors together. The chain of sensors is referred to as *sensor array* and can be connected to the data-logger, which also acts as the power supply of the sensor array.

In the following context, *sensor* will refer to the whole sensing unit, containing the gyroscope (with integrated accelerometer) and the added Arduino, along with additional wiring. The gyroscopic sensor MPU-6050 will be referred to as gyroscope.

System Integration

At the heart of the system, the sensor array is the main unit to detect movements at different locations on the (simulated) patient's back. Each sensor is placed and fixated to a region of interest. With four sensors, for example, these regions could be chosen in accordance to the four regions of the spine, placing the sensors on the pelvis, lower back, upper back, and neck.

The sensors are able to measure orientation on the X-, Y-, and Z-axes as well as acceleration on these axes. The sensor output directly corresponds to the spinal orientation and acceleration at the point of measurement. The output data is fed back to the data-logger, where it is processed. The orientation information of each sensor is then used to align the spinal segments in the 3D visualization software accordingly.

With multiple sensors, the output can be interpreted in direct correlation to each other, where one sensor acts as the base of global rotation. Ideally this sensor would be chosen to be the very first and lowest on the back (i.e. at the pelvis for whole body measurement or at the upper back for neck movement measurement). Output from the higher sensors can then be displayed as relative orientation to the previous sensors, disregarding the general orientation (e.g. upright compared to supine position or different orientations to the compass) of the patient.

Using multiple sensors in the array, distinct complex movements can be detected and measured by providing additional points of reference. This additional information is collected by the data-logger and utilized by the visualization software. Data acquisition is done by the individual sensor, which themselves can be used as a self-contained units, providing measurement of acceleration and orientation in relation to theirs own initial positions.

Sensor Architecture

I^{2}C (also I2C or IIC, for Inter-Integrated Circuit) is a bidirectional two-wire bus developed by Philips Semiconductors. It is used for efficient Inter-IC control and features on-chip interfaces to communicate with other devices directly via the I^{2}C bus using only two bus lines: A serial data line (SDA) and a serial clock line (SCL). The clock signal is generated by the so-called I^{2}C master. Data transfer can be initialized by the master device by sending or requesting information from a slave device. Each device connected to the bus has a unique identifier that is software addressable. This I^{2}C address can either be a 7-, 8- or 10-bit sequence (often abbreviated in hexadecimal digits with the prefix *0x*) [61].

Because of hardware limitations in the MPU-6050 gyroscopic sensors, only two separate (out of the 112 available) 7-bit I^{2}C addresses can be used. On the GY-521 breakout board that is used, the addresses can be switched using the AD0 pin. On default the sensors are set to communicate on the 0x68 address, connecting AD0 to VCC will set the address to 0x69. The same addresses are also used by other components, such as the RTC module. Different methods to use more than two components with the two available addresses, such as using separate I^{2}C ports on the Teensy microcontroller or bit-banging additional connections were tried, although these methods still have the problem of providing only two additional connections per port. Switching ports resulted in very slow sensor read-out rates. Cycling through the sensors one after another with continuous reinitialization in-between, where the active sensor was placed on address 0x68, while inactive sensors remained on 0x69, resulted in slightly faster read-out rates but was only possible using raw data output.

In order to communicate with more than two sensors simultaneously, the sensor array is rebuilt from ground up. Using a dedicated microcontroller for each sensor, the output can be forwarded to another I^{2}C address. The added microcontroller continuously acquires data from the connected sensor. On request from the main data-logger, the latest data set is then forwarded. By separating the microcontroller–sensor connection and the main data-logger–sensor array connection, a modular sensor array can be achieved, allowing for a theoretical connection of up to 128 sensors (the limit of 7-bit I^{2}C addresses, although 16 addresses are reserved for special purposes, leaving 112 addresses available for end users). Physically that limit will be reached earlier, when the maximum supported capacity of 400 pF is exceeded [61].

The new prototype sensor design uses the same GY-521 breakout boards for the integrated gyroscope and accelerometer in the MPU-6050. A low-power, high-performance ATtiny167 was intended as the preprocessing microcontroller but later changed to a ATmega328P. The main advantage of the ATmega328P in this use case is higher available flash memory (32 kB on the ATmega versus a maximum of 16 kB on ATtiny chips) and dedicated I^{2}C pins, while having similar power consumption [62, 63]. In this prototype an Arduino Pro Mini is used as a development board for the ATmega328P. Later versions will feature a custom printed circuit board for both the processing and motion tracking unit.

The Arduino is directly wired to the GY-521 breakout board, connecting pins 9 through 6 with VCC, GND, SCL, and SDA. Power supply of the MPU-6050 is managed by pin 8 and 9, providing a digitally controlled fixed voltage to turn the device on and off using software only. On the downside, these pins only support currents of up to 40 mA, which is still well above the normal operating current of the gyroscope [64]. The I^{2}C connection is established using bit-banging on pins 6 and 7, creating a dedicated bus used for the microcontroller–gyroscope connection. Additionally pin 2 is used to attach an interrupt from the FIFO (First In, First Out) buffer on the MPU-6050 to time the read-outs correctly.

Apart from the microcontroller–gyroscope connection, a simple circuit is designed for the connection between data-logger and the single components of the sensor array. The main 3.3 V power supply as well as the general data-logger I^{2}C data (SDA) and clock (SDL) lines are passed on from one end of the sensor circuit board to the other to allow a modular connection of multiple sensors. When using the modular structure, all sensors are connected in parallel with each other and all I^{2}C connections refer to the same inputs. Additional I^{2}C pull-up resistors are not used in the prototype, because both the Arduino Pro Mini and the Teensy have internal pull-up resistors on their SDA and SCL lines. The sketch in Fig. 4.2 shows a simplified overview of the connections and components of the sensors. On the actual sensor, some wires are placed on the back layer of the circuit board.

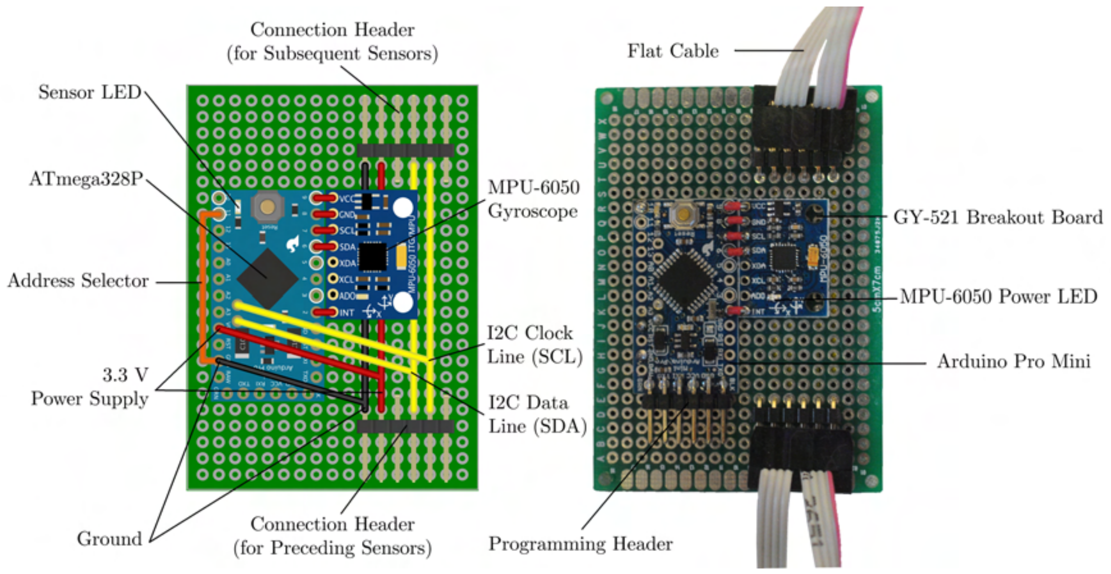


Figure 4.2: Overview of the sensor circuit: Comparison between sketch and actual design.

Further on, a separate ground connection to the pins 10 to 17 is possible. This connection is distinct to every sensor and defines the I^{2}C address. For convenience during development, all sensors are loaded with identical software, that selects the I^{2}C address based on the pin configuration. The preinstalled LED on the Arduino is also used to display status information. Since the LED is internally wired to pin 13, this status indication will not work, when the address selection is configured using this pin.

Software

Each sensor is programmed with a simple software to gather data from the gyroscope and forward it to the data-logger. The ATmega328P is programmed using the Arduino IDE, utilizing open source libraries to simplify the communication between the gyroscope and microcontroller. The software itself is identical for all individual sensors, independent of their intended position on the back and the assigned I^{2}C address. A flow-chart of the program sequence is shown in Fig. 4.3.

During the initialization phase all outputs and inputs of the microcontroller are declared. The digitally controlled fixed voltage is switched to power the MPU6050. After a short wait for power stabilization the serial connection is initialized. This UART (Universal Asynchronous Receiver Transmitter) connection can be used for debugging, when a computer is connected directly to a single sensor. The communication between the individual sensors and the data-logger, as well as the connection between the motion processing unit (MPU) and the sensor microcontroller is based on the I^{2}C protocol.

Atmel chips (like the ATmega328P or other microprocessors used on Arduino boards) have built-in support for the Two Wire Interface (TWI). Both I^{2}C and TWI are essentially the same and compatible with each other. In some cases of other manufacturers TWI indicates an incomplete implementation of the I^{2}C standards. The Arduino *Wire* library for example adds support for TWI and I^{2}C but only allows using 7-bit addressing [65].

Using the Wire library, the sensors are registered as I^{2}C slaves to communicate with the data-logger. The I^{2}C address is set during initialization, depending on the state of the digital pins 10 to 17 of the Arduino Pro Mini, which can be set to match the desired address sequence.

The MPU-6050 gyroscope and accelerometer data can also be accessed via I^{2}C. A dedicated library is available for Arduino (and other microcontroller platforms) to use I^{2}C specifically for the communication between microcontroller and selected sensors. The *I2C Device Library* (or I2Cdevlib) is a collection of classes for different I^{2}C devices [66]. Every supported sensor has access to its specific device class, while making use of the generic I2Cdev code, providing the I^{2}C bit- and byte-level communication separately [66]. With the I2Cdevlib direct readout of the MPUs integrated DMP is possible, allowing to receive different preprocessed data formats.

The DMP can be accessed on I^{2}C address 0x68 or 0x69. In order not to block these addresses on the main I^{2}C lines, and allowing only two MPUs to be used simultaneously, a separate bus on the sensor microcontroller can be used. The ATmega328P features a single hardware TWI bus, while other microcontrollers are available with multiple buses (the Teensy LC used in the data-logger for example has two I^{2}C busses).

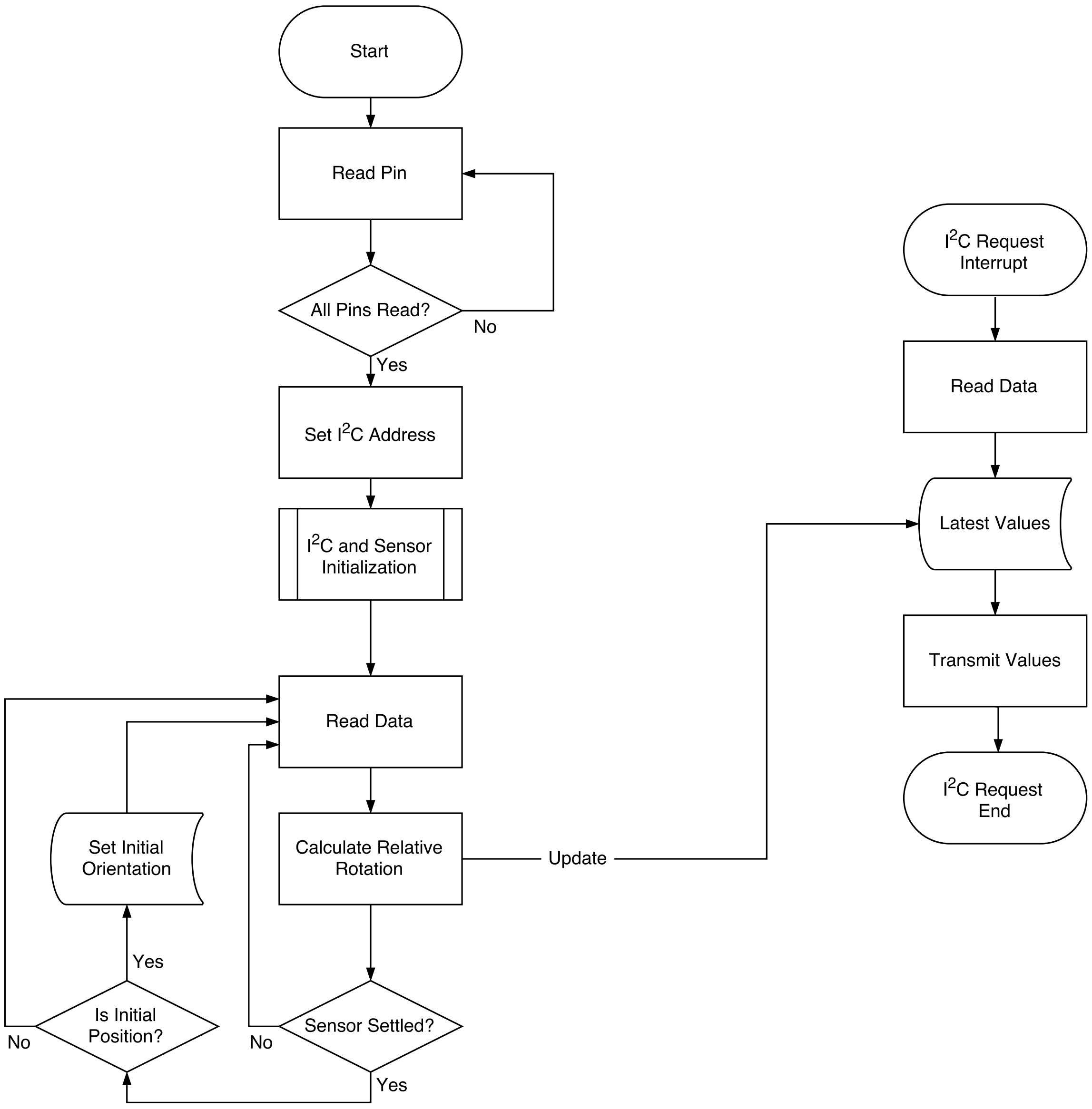


Figure 4.3: Basic overview and flow diagram of the sensor software.

Since the I^{2}C protocol is relatively simple, two spare GPIO (General Purpose Input/Output) lines can be used to simulate the I^{2}C clock and data lines. This way, the I^{2}C bus can be implemented wholly in software without the dedicated on-chip interfaces. This method is also known as bit-banging [67].

To implement a second I^{2}C bus on the ATmega328P, bit-banging is used for the communication between microcontroller and motion processing unit. The third-party Arduino library *SoftI2CMaster* [68] provides a simple solution to use basic I^{2}C functionality with any two digital pins. To access the DMP data via the software bus, a modified version of the generic I2Cdev code [69] that uses the SoftI2CMaster library instead of the Arduino Wire library as I^{2}C implementation is used.

The microcontroller continuously retrieves the orientation from the gyroscopic sensor as quaternion output. Each program cycle takes ~10 ms. Quaternions are four-dimensional complex numbers that can be used as notation for rotation and orientation of rigid bodies in 3D space. The output is expressed by the four factors W, X, Y, and Z from the quaternion $q = W + Xi + Yj + Zk$, where i, j, and k are the three imaginary units. Compared to other representations of rotation such as Euler angles (using yaw, pitch, and roll), quaternions are very simple to compose and avoid discontinuities like the gimbal lock problem [70]. In this case, quaternion output was chosen because of the simple and efficient ways to calculate relative rotation compared to the initial position of each sensor (i.e. orientation) and the relative rotation between the different sensors themselves.

During the first 8–25 seconds (depending on orientation) the DMP has to settle to its initial values. It is important that the sensor is perfectly still and level in that phase, otherwise the DMP will not settle until these conditions are met. Until the DMP is fully initialized a subroutine analyses the output and calculates the standard deviation over the last 100 values. When the standard deviation falls below 0.0001, the green LED on the Arduino is triggered to show that the sensor is fully initialized and settled. The sensor output is then set as initial orientation an base for further rotation. To calculate every subsequent orientation the conjugate of the initial orientation is multiplied with the latest quaternion output, effectively resetting the output to the identity quaternion $q = 1$.

The sensor output during the initialization phase is depicted in Fig. 4.4. The upper plot visualizes the quaternion output over time, where X, Y, and Z are the three imaginary parts and W the real part. The bottom plot interprets the sensor output as orientational directions in three dimensions. The sensor was held at a slight angle for this demonstration to increase the initial sensor offset. Faster data stabilization is possible if the sensor is placed on a flat ground, where the initial output matches the identity quaternion.

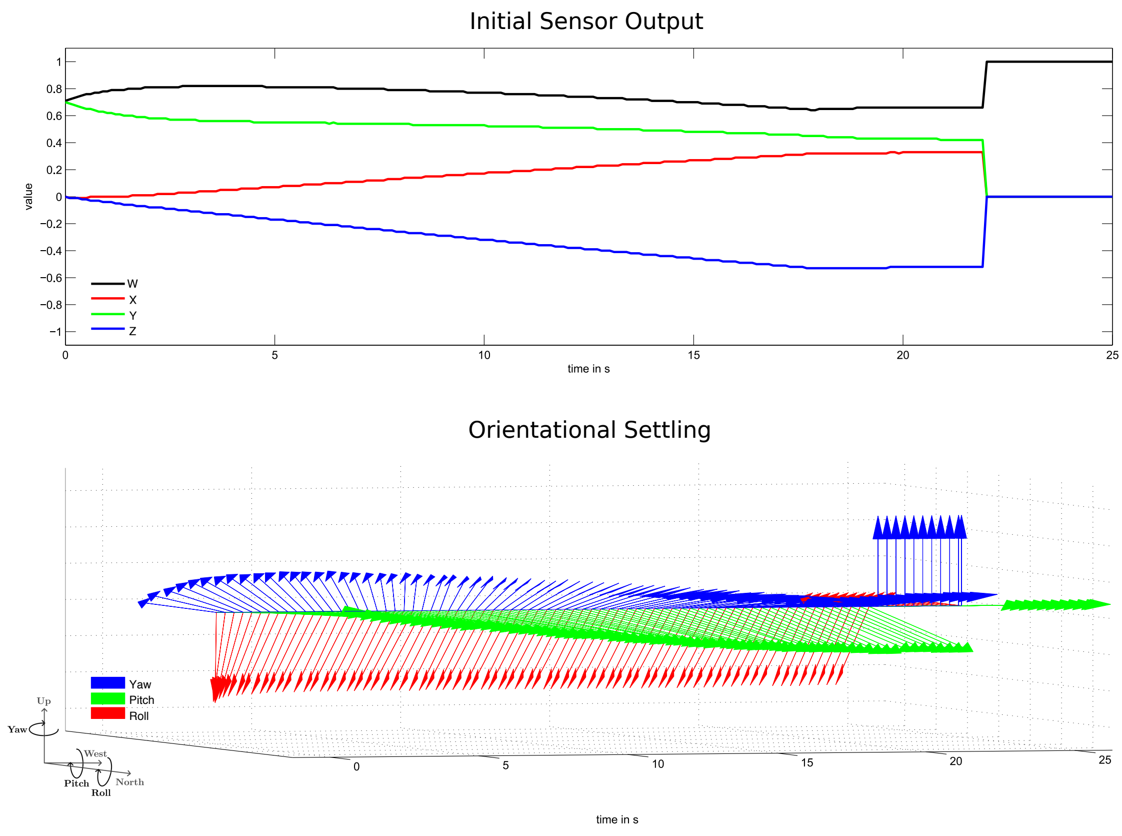


Figure 4.4: Initial drift of sensor values (top) and the consequential orientation in a global coordinate system (bottom). After 22 s the sensor is settled and the output harmonized.

When the sensor is powered on in this demonstration, the initial data output is at $W = 0.71$, $X = 0.00$, $Y = 0.70$, and $Z = 0.00$ but starts drifting immediately. After a couple of seconds, the drift approaches linearity until the final value is reached at the 19 s mark. At this point the sensor is fully settled at $W = 0.66$, $X = 0.33$, $Y = 0.43$, and $Z = -0.52$. After 22 s the sensor output is reset to match the identity quaternion. Every consequential movement will be measured in relation to this orientation.

On the bottom of Fig. 4.4, the sensor data is displayed as orientational output. The three perpendicular arrows are used to indicate rotational yaw, pitch, and roll of the sensor in a world coordinate system. The abscissa is reutilized as time axis, where different points of origin of the arrows indicate the elapsed time. After the sensor is fully settled, the orientation is aligned to match the real world coordinates.

After the initialization, the sensor is ready to take real measurements. While continuously collecting data, the values are only passed on upon request. When a request event is received for the I^{2}C address used by the sensor, the sensor answers using a 32byte sequence containing the latest sensor values. Separation of data collection and transfer eliminates the problems of buffer overflows present if the DMP is not read on time. The I^{2}C request can be triggered by the master device (i.e. the data-logger) independently for all connected sensors in the sensor array.

4.2.2 Data Acquisition Unit

The data acquisition unit (or data-logger) is taken directly from the previous prototype. Unlike the sensor array only limited alterations of the hardware were made. The software, however, was completely rewritten to make use of the changed sensor array.

System Integration

The data-logger forms a bridge between sensor array and the computer program used to evaluate the data. It records the measurements of the sensor array onto a microSD card and allows to view a live data feed from all sensors simultaneously.

While the orientational data is collected by the individual sensors a conjunction of these individual data sets is not made by the sensor array. In the data-logger, each sensor is targeted separately and sampled at a predefined rate. The data-logger is aware of exactly how many sensors are connected in the array and creates a collection of all the gathered data. A time stamp is added to refer the data to a fixed point in time.

The complete data record can either be saved to the internal microSD card or viewed live when the data-logger is tethered to a computer. No alteration of the data is done at this point. The correlation (relative rotation) between the individual sensors is calculated from this output in the visualization software.

Hardware

For the greatest part, the data-logger hardware remains identical to the previous prototype. Only a few changes were made during further development and most of them were reversed in the end.

A Teensy LC (Low Cost) *USB Development Board* is used as the main processing unit. It features a high-performance 32-bit 48 MHz ARM Cortex-M0+ microcontroller with 62 kB of flash memory, hardware serial ports (UART, SPI, and I^{2}C), a micro USB port (for programming and connection of peripherals), and a total of 27 GPIO pins while having a very small footprint. Furthermore, it is fully compatible with the Arduino programming environment and all of the available libraries [71].

In order to use the Teensy with an external battery, the USB power connection has to be separated from the voltage input used by the battery and charging controller. A 3.7 V 2600 mA h Lithium-Ion (Li-Ion) battery (ICR-18650 NQ-SP from Emmerich) is used as the primary power source. It features a built-in protection circuit against depth discharge. To ensure a correct charging procedure, a charging controller (STC4054GR from STMicroelectronics) regulates the charging current up to 800 mA dependent on the battery voltage level. It operates, using the 5 V USB power supplied via the Teensy's micro USB port (regulated to a fixed charging voltage of 4.2 V) and charges the battery in 3.5 h. Two 3 mm LEDs are used to indicate charging status and completion. An additional 5 mm LED can be configured by the microcontroller.

The Li-Ion battery is discharged during operation with varying output voltages between 4.2 V and 2.75 V. Below 2.75 V the voltage is cut-off by the built-in protection circuit of the battery. To supply all active components with the varying output, a synchronous buck–boost converter (LTC3440 from LinearTechnology) is used, allowing to supply a constant voltage of 3.3 V. Above 3.3 V, the converter operates in buck mode, converting voltage downwards until the voltage drops below 3.3 V, when the converter boosts the output voltage. The buck–boost converter is able to operate with an efficiency of 96 %, and is connected directly to the 3.3 V input of the Teensy.

The micro USB port on the Teensy is not only used for charging but also for programming during the development phase. Additionally live sensor feedback can be received via the UART protocol. Simultaneously, the sensor feedback is recorded to a microSD card. The card is connected to the Teensy with a microSD card socket. Both the microSD card and socket were slightly modified after prototype testing, because the metal housing of the socket caused electrical problems. A friction fit holds the microSD card in place but also allows fast removal of the card.

To read the data, the microSD card has to be disconnected from the data-logger and connected to a computer. Alternatively, the data files can be dumped character by character over the UART connection, which is significantly slower. As of yet, the microSD card cannot be accessed as a USB mass storage device. Accessing the microSD card via the micro USB port might be possible in the future, using software modifications.

The sensor array can be connected to a 20-pin flat cable connector. Because of the original implementation of the sensor array, the 20 pins are split into four groups—one for each sensor. Each group has dedicated power lines (3.3 V and ground), as well as two I^{2}C lines (SDA and SCL). The additional, fifth line is independent for each group, acting as chip select (CS) for the individual sensors. The chip select lines

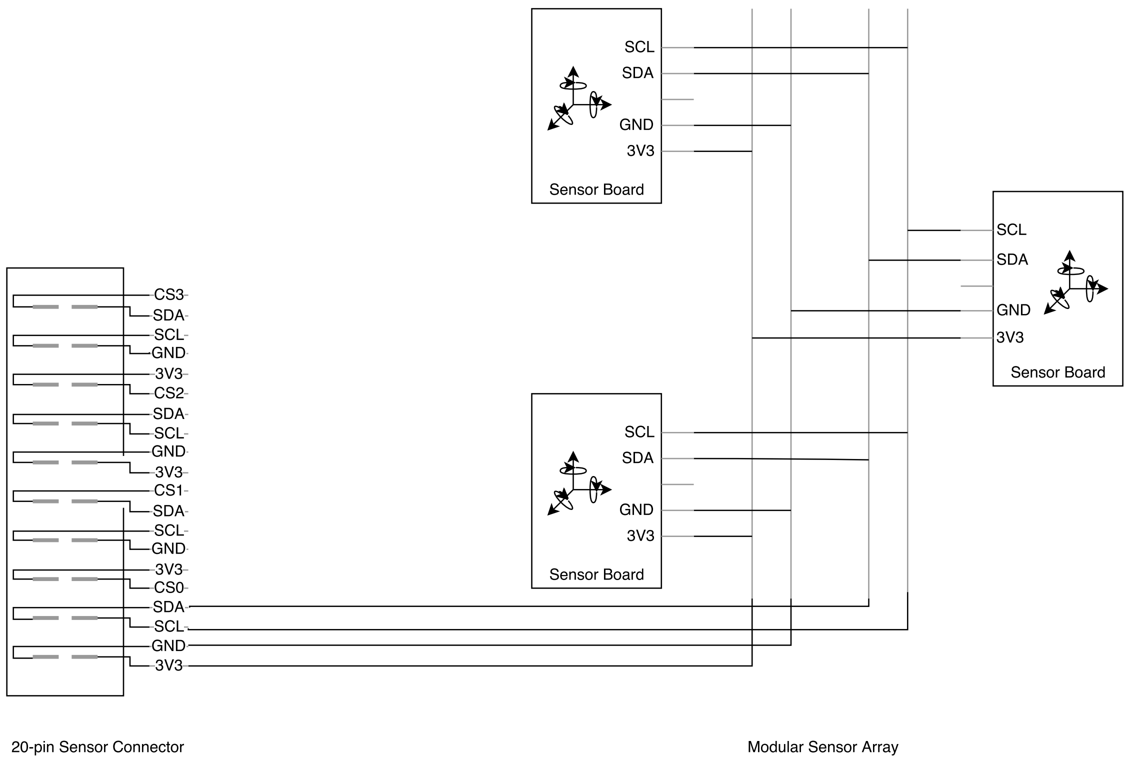


Figure 4.5: Schematic overview of the modular sensor connection along with the old connector layout.

are wired to the Teensy pins 0 through 3. Of the 20 available lines only eight carry different information, as 3V3, GND, SDA, and SCL are wired to dedicated hardware pins or components, while CS0 to CS3 are wired to digital pins, that can be defined in software. In the end, it was possible to reuse the 20-pin connector for the new sensor design by only using the pins dedicated to power supply and I^2C connection. A schematic overview of the connection between the 20-pin connector from the data logger and the new, modular sensor layout is shown in Fig. 4.5. On the sensor array side of the design, the connector is reduced to five pins with four electric lines. If necessary, the remaining four lines (CS0 to CS3) can be added in the future to provide additional features (such as the option to initialize the sensors successively in the right order).

A custom printed circuit board (PCB) was designed using CadSoft EAGLE 7.2 to connect all of the electrical components as well as the Teensy in the data-logger. To keep the data-logger as small as possible, the PCB height matches the Teensy and the diameter of the battery. Components are placed on both sides of the circuit board. 3D renderings of the front and back layer of the PCB are shown in Fig. 4.6. For further size reduction, surface mounted technology (SMT) is used for most of the components on the board. Some of the larger components, like the 20-pin connector and the Teensy itself, however are available as through-hole technology (THT) only. The components were aligned to place the micro USB connector of the Teensy and the microSD card slot on opposite ends of the backside of the circuit board. The battery connector is placed in-between both. On the front side of the circuit board, the charging controller is placed along with the components facing towards the outside, such as the LEDs, sensor connector, and power switch. A full overview of the board schematics can be seen in Appendix B.

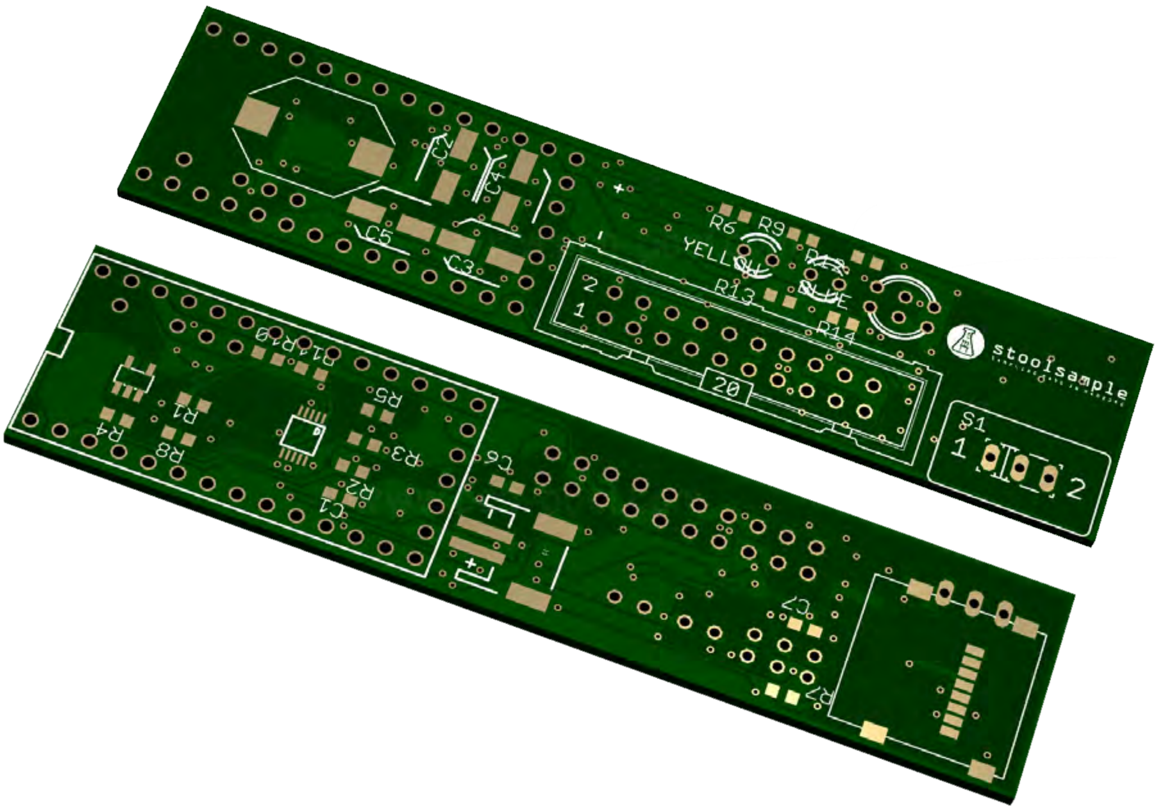


Figure 4.6: 3D Rendering of the data-logger PCB.

The data-logger was cased in a housing to keep all of the components together. As a portable device, the data-logger has to be as small as possible, while also being robust and durable. Within the housing, the PCB and battery have to be fixated, consequently the size of the housing is dependent on these components. The outline of the PCB measures 19.40 mm x 85.80 mm with a maximum depth of 16.96 mm after the components are added. The battery is cylindrical with a diameter of 18.40 mm and a length of 70.00 mm (not counting the connected cables, which also have to be fitted into the housing). The housing was designed using the CAD (computer aided design) software Autodesk Inventor 2015 and manufactured using stereolithographic rapid-prototyping with a layer resolution of 50 µm. With an outline of 90 mm x 23.6 mm x 45 mm, this creates a fairly small and solid case.

The housing is split into two parts and can be opened from the top. The cover is fixated using four M2 screws. Inside the case supporting structures are used to keep the battery and circuit board in place. Cutouts are made for the Micro USB port, 20-pin sensor connector and microSD card slot as well as the three LEDs and the power switch. An overview of the designed components and the final data logging unit is shown in Fig. 4.1. Schematics of the housing can be seen in Appendix B.

Some additional changes were needed for the first prototype testing of the data-logger. Mainly a real time clock was missing to record the data with a correct time stamp. While the Teensy LC features RTC logic on board, it lacks the needed oscillator and battery backup. Unlike the Teensy 3.1, it does not feature dedicated soldering pads to make use of the RTC logic. This fact was overlooked for the first prototype and an additional RTC module was added before prototype testing. A Dallas Semiconductors DS1307 RTC module was used along with an additional backup button cell battery. Because of tight space inside the housing, this module caused some electrical issues and was removed again during further development. Future prototypes will use the Teensy 3.1 model, instead of the Teensy LC to gain RTC capabilities. So long, the data-logger will not be able to get the correct date and time, when the system is started, but the built-in timers of the Teensy will still be able to output the runtime of the system.

During RTC testing the 5 mm LED was also changed from a multi-color LED, to a single-colored red LED. The different colors were very hard to discern and the six pins of the LED were also blocking the second I^{2}C bus of the Teensy, which was then used for the RTC. Instead of just removing the LED, a single color replacement was used. Using different flashing sequences, various response codes are still possible.

Software

From a software perspective, the data-logger was completely overhauled. Like the sensors themselves, the data-logger is programmed using the Arduino IDE. For full use with the Teensy microcontroller board the software add-on *Teensyduino* is necessary. The program on the microcontroller incorporates multiple open source software libraries to use the UART, I^{2}C, and SD card connection. A library for timekeeping is also incorporated, although the RTC hardware component is currently missing. Each iteration of the program will start at the Unix epoch (00:00:00 on January 1st, 1970) and base further timestamps on that starting point. When the RTC is added, the correct time and date will be used instead. A flow chart of the data-logger program sequence is shown in Fig. 4.7.

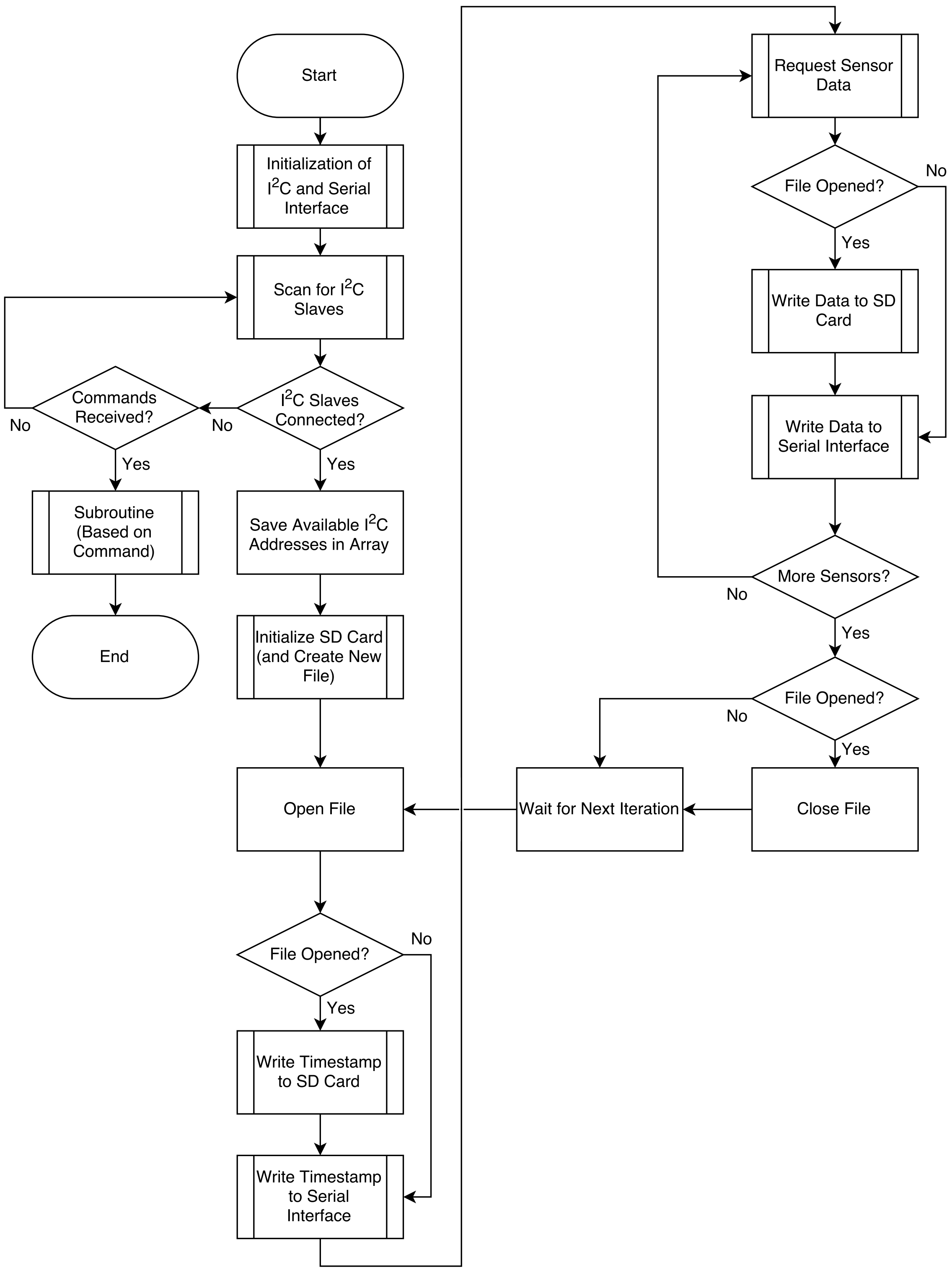

Start
Initialization of I²C and Serial Interface
Scan for I²C Slaves
I²C Slaves Connected?
No
Yes
Commands Received?
No
Yes
Subroutine (Based on Command)
End
Save Available I²C Addresses in Array
Initialize SD Card (and Create New File)
Open File
File Opened?
No
Yes
Write Timestamp to SD Card
Write Timestamp to Serial Interface
Request Sensor Data
File Opened?
No
Yes
Write Data to SD Card
Write Data to Serial Interface
More Sensors?
No
Yes
File Opened?
No
Yes
Close File
Wait for Next Iteration

Figure 4.7: Overview and flow chart of the data-logger program sequence.

In the beginning of the program, the serial connection is initialized and the I^2C bus is joined as the master. The program will then wait for two seconds to allow the connected sensors to be setup as I^2C slave devices. Once that time has passed, the I^2C bus is scanned for connected addresses. The addresses are printed over the serial connection for debugging reasons. If no I^2C slaves are available, the search will be continued until at least one sensor was found. During that time, the serial interface is available for input: Different subroutines, like the SD card readout may be started. After the sensor array was recognized, the devices and addresses are saved in an array for further use.

In the main program, the SD card connection is initialized and verified. A new file with incremented name will be added to the MicroSD on root level. If the SD card is missing or a connection fails, the data-logger will not save the sensor output itself. In that case, an external serial recording device (e.g. a computer) has to be used to save the data. Alternatively the live feedback can be viewed directly using the visualization software.

After the initialization process, the sensor scanning procedure loops until the data-logger is turned off. For each iteration, the created file on the microSD card is opened and the current timestamp written to the file. A 32 byte sequence is requested from every attached sensor one after another. The individual sensors react to that message by calling an interrupt routine to respond with the most recent orientational output. This data sequence is sent via the I^2C interface and received by the data-logger, which then appends these to the data file. The same output (timestamp and the individual sensor data) is printed to the serial interface simultaneously and can be viewed using the visualization software. At the end of the iteration, the data file is closed and the LED is flashed to show that a full line was written to the file. If the SD card was not initialized, the file writing is omitted and the LED is faded slightly instead of flashing, to show that nothing could be written to the microSD card. Before the next iteration of the sensor scanning procedure, the program waits for a predefined duration of currently 100 ms.

4.2.3 Visualization

The spinal measurement data is recorded as a plain text file with comma separated values (CSV) with the *.csv* file extension. Each line represents a combined data record of all connected sensors. A data record line starts with the date and time in “YYYY/MM/DD hh:mm:ss” format followed by milliseconds for absolute ordering. Since the RTC is not yet active, the date, time, and milliseconds are calculated from the program runtime. Once the RTC is added, date and time will be synchronized with it and a different method of absolute ordering might be necessary.

Table 4.1: Sample raw data output from the data-logger, as written to the CSV file.

```
1970/01/01 02:23:37,878,S40,+0.9368,+0.0118,-0.0024,+0.3494,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0061,-0.0030,+0.3236
1970/01/01 02:23:38,040,S40,+0.9368,+0.0118,-0.0024,+0.3494,S41,+0.9410,+0.0083,-0.0005,+0.3382,S42,+0.9461,+0.0061,-0.0031,+0.3236
1970/01/01 02:23:38,201,S40,+0.9368,+0.0118,-0.0024,+0.3494,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9462,+0.0061,-0.0031,+0.3235
1970/01/01 02:23:38,372,S40,+0.9368,+0.0118,-0.0024,+0.3494,S41,+0.9410,+0.0082,-0.0006,+0.3382,S42,+0.9462,+0.0061,-0.0031,+0.3235
1970/01/01 02:23:38,534,S40,+0.9368,+0.0118,-0.0024,+0.3494,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0061,-0.0031,+0.3236
1970/01/01 02:23:38,696,S40,+0.9368,+0.0118,-0.0024,+0.3494,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3236
1970/01/01 02:23:38,858,S40,+0.9368,+0.0118,-0.0024,+0.3495,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3236
1970/01/01 02:23:39,027,S40,+0.9368,+0.0118,-0.0024,+0.3495,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3236
1970/01/01 02:23:39,189,S40,+0.9368,+0.0118,-0.0024,+0.3495,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3236
1970/01/01 02:23:39,353,S40,+0.9368,+0.0118,-0.0024,+0.3495,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3236
1970/01/01 02:23:39,515,S40,+0.9368,+0.0117,-0.0024,+0.3496,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3236
1970/01/01 02:23:39,683,S40,+0.9368,+0.0118,-0.0024,+0.3496,S41,+0.9410,+0.0083,-0.0005,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3236
1970/01/01 02:23:39,846,S40,+0.9368,+0.0118,-0.0024,+0.3496,S41,+0.9410,+0.0083,-0.0005,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3237
1970/01/01 02:23:40,010,S40,+0.9368,+0.0118,-0.0024,+0.3496,S41,+0.9410,+0.0083,-0.0005,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3237
1970/01/01 02:23:40,170,S40,+0.9368,+0.0118,-0.0024,+0.3496,S41,+0.9410,+0.0083,-0.0005,+0.3383,S42,+0.9461,+0.0063,-0.0030,+0.3237
1970/01/01 02:23:40,339,S40,+0.9368,+0.0118,-0.0024,+0.3496,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0062,-0.0031,+0.3237
1970/01/01 02:23:40,503,S40,+0.9368,+0.0118,-0.0024,+0.3496,S41,+0.9410,+0.0083,-0.0005,+0.3383,S42,+0.9461,+0.0063,-0.0031,+0.3238
1970/01/01 02:23:40,665,S40,+0.9368,+0.0117,-0.0024,+0.3496,S41,+0.9410,+0.0082,-0.0006,+0.3383,S42,+0.9461,+0.0063,-0.0031,+0.3238
1970/01/01 02:23:40,880,S40,+0.9368,+0.0117,-0.0024,+0.3496,S41,+0.9410,+0.0083,-0.0005,+0.3384,S42,+0.9461,+0.0063,-0.0031,+0.3238
```

Next, the sensor data is located in the data record, starting with the unique sensor identifier based on its I^2C address. The quaternion output is given with four decimal places for each component and saved with the real part (W) first, followed by the three imaginary parts (X, Y, and Z). For each consecutive sensor, the data will be appended in the same matter until a new line starts a new data record. During data recording, the exact output is sent identically over the UART connection. A sample record which shows the initial drift of three connected sensors simultaneously is shown in Tab. 4.1.

The CSV files can be opened with any regular text editor or spreadsheet program. In addition to the plain text overview of the sensor data, the data can be interpreted with the special purpose computer program, written to visualize and play back the spinal movement during the recorded time frame. Input can be used from the CSV files after the measurement is finalized or as a live feed over the serial connection, when tethered to the computer.

System Integration

Plain text output is not ideal to review the measured data. Although hard transitions can be spotted rather easily in the plain text overview because of the sudden change of the value, this might not be sufficient when looking over the output of multiple sensors. The quaternion data is split into four elusive factors, which make the interpretation even harder. A visual presentation of the data is much more appealing and simpler to understand.

The visualization program uses a three-dimensional representation of the human spine, head, and overall torso to create a direct link between measurement and display. The orientational output of each sensor is taken and applied to the 3D model. Detected movement of the spine is displayed respectively to imitate the spinal posture on screen.

Analyzing the data using the visualization software makes it easier to scrub through the whole recording in order to the find time frames of interest. Simple software modifications make it possible to find these regions even faster. The current software prototype focusses on the depiction of the data output.

Development Environment

The program is written using the Processing Development Environment (PDE) and builds on the Open Graphics Library (OpenGL) interface. This combination offers a lightweight framework and requires a shorter familiarization period compared to alternatives like game engines or 3D animation software suits from a general programming standpoint. The Processing programing language uses simplified Java syntax and can be compiled into stand-alone Java applications.

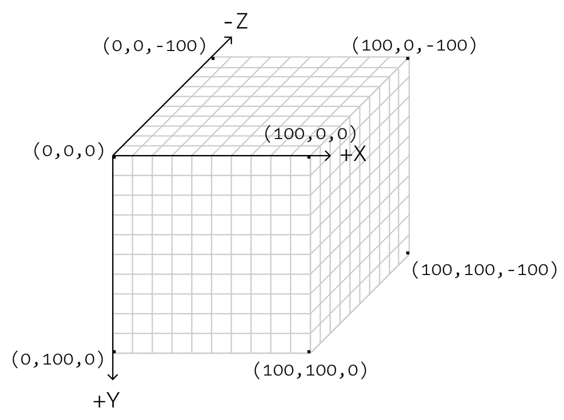
The Processing programing language focusses on visualization and user interaction and was specifically designed for the generation and modification of two- and three-dimensional graphic models [72]. It can be extended easily and offers a vast amount of additional libraries. Useful extensions include options to import (and export) different 3D models into the program or to use data files for input information. This way, the CSV data can be loaded and used as orientational input. It is also possible to expand the program to receive serial input, allowing to receive live sensor input.

The OpenGL interface is available as a pre-included library. It allows the program to use hardware acceleration for high-performance graphics, which makes the rendition of large 3D areas more efficient [72]. All the main components (Java, Processing, and OpenGL) are available on most operating systems, allowing to create platform independent standalone applications running on computers, mobile devices and—to a lesser extent (switching to Javascript and WebGL)—online, through a web browser.

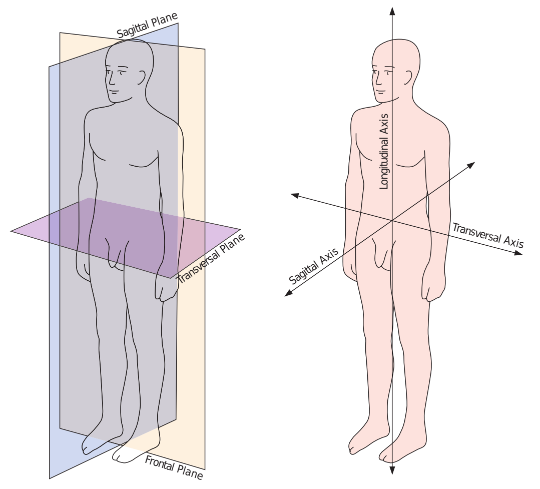
Modeling the Spine

The visualization software is using a low resolution 3D model of a human skeleton without extremities to display the spinal posture measured by the device. The 3D bone model is split into five segments: Pelvis, lumbar spine, thorax, cervical spine, and head. In the program, these segments are stacked above each other, creating a fixed connection point at the junction of two segments. To align the final representation with the sensor output, each segment can be rotated around its lower connection point. The pelvis model is not rotated by the sensor output. It acts as a base for the whole model and can be aligned manually with the mouse.

In Processing, a program runs as a so-called *sketch*. By default, a sketch uses a Cartesian coordinate system for the definition of the window size and position of



(a) Coordinate system used by the Processing Development Environment [72].



(b) Human body planes and axes [73].

Figure 4.8: Comparison between computer coordinates and medical body planes and axes.

objects within that window. Starting at the upper left corner, the X- and Y-axes expand along the width and height of the screen, while the Z-axis for three-dimensional graphics expands into the screen, shown in Fig. 4.8a. The coordinate system can be translated freely on these axes and rotation is possible using either intrinsic Euler angles, where the orientation of the coordinate system rotates with every elementary rotation on the X, Y, and Z axis or axis–angle representation, where the rotation occurs around a vector indicating the direction.

In a medical context, direction and position of structures is often described using three basic planes of reference: The sagittal, transversal, and frontal plane. The planes and corresponding axes are based on the body, independent of its orientation in real word coordinates. The sagittal axis passes from front to back, the transversal axis from side to side, and the longitudinal axis from head to toes. Perpendicular planes are spanned by these axes. Both are shown in Fig. 4.8b.

The 3D models of the spinal segments are imported upright, facing forward and stacked vertically, aligning the longitudinal axis of the skeleton with the Y axis of the sketch. Transversal and sagittal axes align with the X and Z axes respectively. In Processing, objects and structures can only be added to the zero point of the coordinate system. For different implementations, the whole coordinate system has to be modified. To apply the rotation to the individual segments from the sensor output, the coordinate system is rotated before importing the corresponding 3D model. Using these intrinsic rotations, the body axes stay compliant with the current axes of the programming environment.

Without previous rotation, the pelvis is placed onto the sketch to act as a base for the following structures. Whole body rotations will be ignored based on the orientation of the pelvic sensor in order to keep the visualization steady while the patient moves. To calculate the different orientations, the sensor output is taken from the serial connection or file input. A quaternion array holds the current values of all individual sensors. The relative orientation is then calculated from each sensor to the next and is applied to the visualization. After a segment has been placed into the sketch, it is no longer affected by any rotations of the underlying coordinate system. For each consecutive segment, the rotation is applied separately, based on the prior rotation. In the end, the output on the screen matches the spinal posture of the person wearing the sensor array.

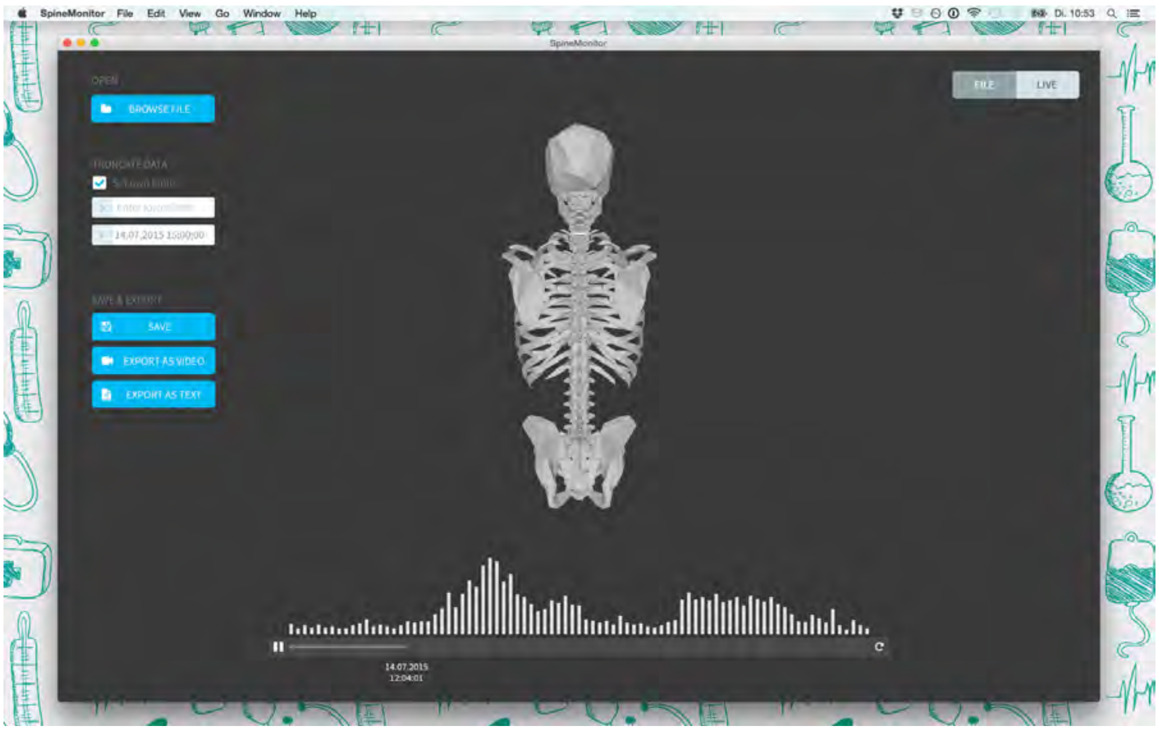


Figure 4.9: Overview of the visualization software showing a perfect posture.

In the final prototype of the software, the 3D model is placed in the middle of the screen, as shown in Fig. 4.9. Beside the spine itself, head, trunk, and pelvis of the body are also shown as points of reference to provide a more distinguishable depiction of torsion and flection in the model. For performance reasons, a low polygon 3D model is used, as every frame of the visualization is built from ground up. The spine is split into its anatomical segments, allowing the model to rotate around the respective distal ends of the individual segments. As a result, the output might cause rough bends and the sensors have to be placed according to these segments. Interpolation of the rotation of intermediate vertebrae is possible but requires precise positioning of the sensors to predefined anatomical landmarks or an option to calibrate the program to the vertebra–sensor-alignment, as well as further calculations of the intermediate values of rotation. This offers further opportunities for upcoming prototypes but is currently not implemented.

Below the 3D model a scrollbar is located along with a bar diagram of sensor activity, to quickly jump to desired time frames of recorded session. This function, as well as the other user interface buttons shown in Fig. 4.9 are not fully implemented yet. Currently the visualization software is divided into two separate programs for serial and file input. Both programs are limited to their base functionality of displaying the skeleton model correctly.

4.3 Final Characterization of the System

The overall functional prototype of the system consists of multiple sensors, combined in the sensor array, a data-logger, and the visualization software for analyzation of the data output. The sensors are placed on the (simulated) patient's back in positions that match the spinal segments. All sensors are connected to each other and form the sensor array. The lowest placed sensor is connected with the data-logger, which also houses the power supply. The data-logger records the orientational output from each sensor to an internal microSD card and can be placed into the patient's jeans pockets as it is not much larger than a mobile phone. After the data was recorded, the visualization software can be used to play back the movement of the patient's spine in the recorded time frame. The source code of the single components is enclosed in Appendix C.

The current prototype is functional and can record orientational data from more than two sensors. Validity of measurements and sensor stability is tested in Chapter 5. The visualization software works and can display the current sensor alignment in real time. Immersion is slightly impaired, as the system is not as stealthy anymore. While the data-logger can be fitted easily into most pockets and stay undetected, the sensor prototype is considerably larger than the previous prototype. In the prototype stadium the large form factor is acceptable, as the sensors are still relatively flat.

In total, five sensors were built to be part of the sensor array. The sensors were built using pre-made sensor breakout boards and microcontroller boards, all adding to the total size. In future prototypes, the sensor can be slimmed down significantly by using a custom PCB, neglecting all unnecessary or duplicated components. Besides

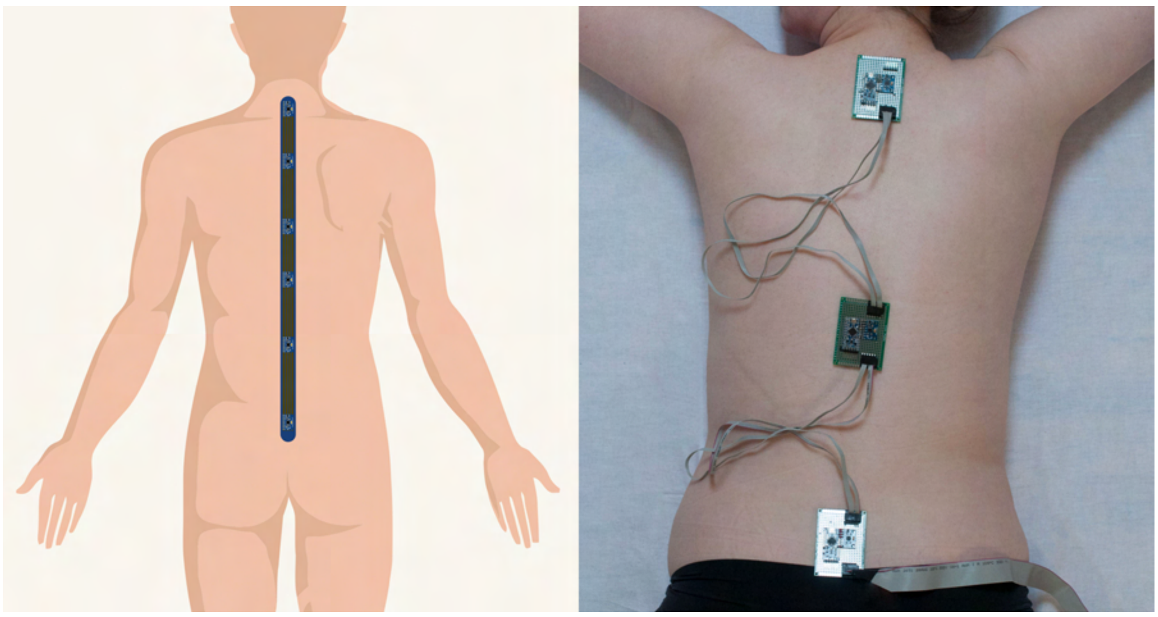


Figure 4.10: Proposed size of the sensor array for future prototypes (left) compared to the current prototype (right).

pull up resistors and decoupling capacitors only the MCU and gyroscope are needed on the sensors. Both components are available as miniature quad flat no-lead (QFN) packages. Fig. 4.10 shows the proposed size of the final sensor array in comparison to the current prototype. Smaller sensors with better cable management can be fitted to the patient in greater numbers to allow more precise distinction between vertebral segments or vertebrae.

5 Analysis of Sensor Characteristics

Tests and measurements of the sensor output and overall system usability are conducted in two stages. First, stability of the sensors and system along with the validity of sensor output is measured in a laboratory environment. The second stage is more user orientated, testing the system's usability with a simulated patient.

Out of the five sensors that were built, two broke during development. On one, the microcontroller is damaged, making it impossible to reprogram the unit, the other one has connection issues with the gyroscope unit. The following tests were all conducted using the remaining three sensors.

5.1 Sensor Stability

The first test is built around sensor stability. In a long-term run, the sensors were tested for the consistency of their measurements. The three sensors of the array were placed on a flat surface and the data-logger was connected to a USB charger to ensure a continuous power supply during the test. The test was conducted for 46 h without moving any of the sensors. In a perfect environment, the sensors should stay perfectly stable after the initialization phase. In the real test however, the sensors show a large amount of sensor drift, as shown in Tab. 5.1.

After 17 s of measurement, the sensors signal that they are settled and reset the values. The actual initialization phase continues for 6 further seconds, when the values reach a steady output at $t = 23\,\mathrm{s}$. The output values remain stable for a couple of minutes but start to show further drifting. After 5 min, the output of sensor 3 already shows a change in the second decimal place of the Z value, while the other imaginary factors (X and Y) remain relatively stable. The real part (W) remains perfectly stable to the fourth decimal place. Sensor 1 and 2 show slightly less drift than sensor 3. The Z values continue to drift further to more than 0.02 after 10 min, where the W value also starts drifting slowly from it initial value. X and Y drift remain in the range of less than 0.0015. After 30 min, the drift of W overtakes the drift of X and Y, while Z continues drifting almost linearly. 2 h into the measurement, the X value of sensor 1 has reached a change in the second decimal place. After 5 h, W and Z now clearly move towards each other, while X and Y drift apart. The values continue

Table 5.1: Condensed data output from the sensor stability test.

TIME STAMP	SENSOR 1				SENSOR 2				SENSOR 3			
t	W	X	Y	Z	W	X	Y	Z	W	X	Y	Z
1970/01/01 00:00:17	+1.0000	-0.0017	-0.0007	+0.0000	+0.9999	-0.0018	-0.0006	+0.0000	+1.0000	-0.0012	-0.0009	-0.0001
1970/01/01 00:00:23	+0.9999	-0.0114	-0.0049	-0.0008	+0.9999	-0.0097	-0.0029	-0.0007	+0.9999	-0.0053	-0.0046	-0.0010
1970/01/01 00:05:00	+0.9999	-0.0107	-0.0051	+0.0079	+0.9999	-0.0088	-0.0030	+0.0093	+0.9999	-0.0046	-0.0049	+0.0108
1970/01/01 00:10:00	+0.9997	-0.0103	-0.0053	+0.0209	+0.9997	-0.0084	-0.0032	+0.0212	+0.9997	-0.0046	-0.0051	+0.0238
1970/01/01 00:15:00	+0.9993	-0.0097	-0.0056	+0.0341	+0.9994	-0.0081	-0.0034	+0.0321	+0.9992	-0.0044	-0.0055	+0.0371
1970/01/01 00:30:00	+0.9973	-0.0085	-0.0062	+0.0720	+0.9982	-0.0074	-0.0038	+0.0589	+0.9973	-0.0040	-0.0062	+0.0727
1970/01/01 01:00:00	+0.9891	-0.0060	-0.0076	+0.1467	+0.9923	-0.0054	-0.0048	+0.1231	+0.9899	-0.0030	-0.0076	+0.1409
1970/01/01 01:30:00	+0.9746	-0.0034	-0.0090	+0.2235	+0.9810	-0.0032	-0.0059	+0.1934	+0.9794	-0.0023	-0.0089	+0.2015
1970/01/01 02:00:00	+0.9570	-0.0011	-0.0101	+0.2898	+0.9645	-0.0010	-0.0070	+0.2639	+0.9660	-0.0016	-0.0103	+0.2582
1970/01/01 03:00:00	+0.9124	+0.0033	-0.0121	+0.4090	+0.9193	+0.0033	-0.0089	+0.3932	+0.9333	-0.0002	-0.0123	+0.3587
1970/01/01 05:00:00	+0.7882	+0.0115	-0.0154	+0.6151	+0.7664	+0.0119	-0.0126	+0.6420	+0.8360	+0.0026	-0.0162	+0.5483
1970/01/01 08:00:00	+0.5020	+0.0228	-0.0189	+0.8643	+0.4220	+0.0229	-0.0159	+0.9061	+0.6133	+0.0066	-0.0206	+0.7895
1970/01/01 12:00:00	+0.0240	+0.0324	-0.0192	+0.9990	-0.1915	+0.0308	-0.0153	+0.9808	+0.1758	+0.0111	-0.0231	+0.9840
1970/01/01 17:00:00	-0.6157	+0.0326	-0.0123	+0.7872	-0.8059	+0.0248	-0.0073	+0.5914	-0.3690	+0.0128	-0.0194	+0.9291
1970/01/01 23:00:00	-0.9931	+0.0149	+0.0025	+0.1161	-0.9761	+0.0024	+0.0061	-0.2173	-0.9053	+0.0094	-0.0057	+0.4245
1970/01/02 00:00:00	-0.9999	+0.0113	+0.0045	-0.0017	-0.9385	-0.0017	+0.0081	-0.3452	-0.9519	+0.0082	-0.0028	+0.3061
1970/01/02 22:25:36	+0.9999	-0.0101	-0.0050	+0.0025	+0.8617	+0.0072	-0.0107	+0.5070	+0.8657	-0.0100	+0.0079	-0.5003

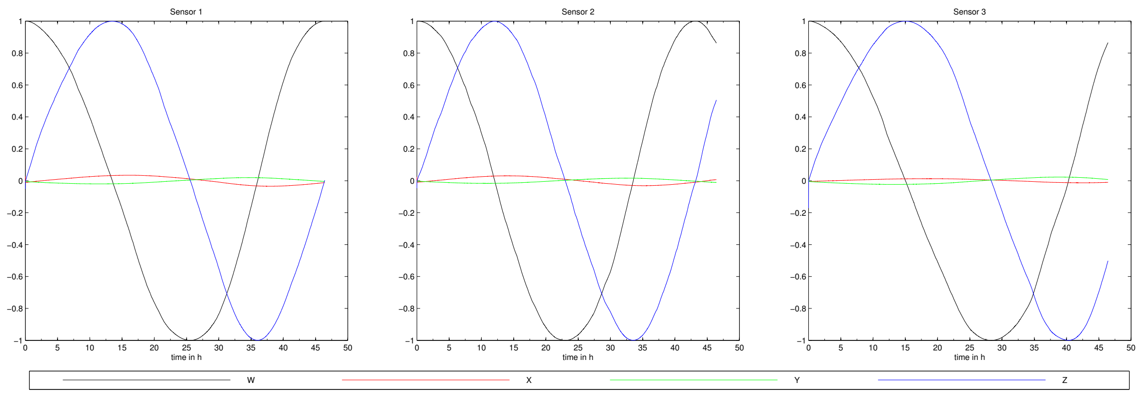


Figure 5.1: Output of the sensor drift during the 46 h test.

to drift until they end up completely reversed after around 24 h and even return to their initial values. At the end of the measurement at $t = 46.43$ h, sensor 1 is almost perfectly at its initial values, while sensor 2 is slightly further and sensor 3 slightly behind. The plot of the complete measurement (Fig. 5.1) shows sinusoidal behavior of all values, where W and Z drift over the complete range with a phase-shift of 90° while X and Y only drift within 3 % of their range and a phase-shift of 180° to each other.

Disregarding the small amount of drift for the X and Y composite of the quaternion, the output is equal to a linear drift on the yaw axis, which is matched to the sagittal axis of the skeleton model. Fig. 5.2 shows the output of the visualization software over the first nine hours of testing. The three sensor values are used as base (sensor 1), lumbar spine rotation (sensor 2), and rotation of the thorax (sensor 3). For the visualization, the points of rotation are set to the junction between pelvis and lumbar spine as well as the junction between lumbar and thoracic spine. The timescale is used in reference to the point of full initialization after 23 s. A plot of the linear drift on the yaw axis as well as the visualization output for the remaining measurements is shown in Appendix D.

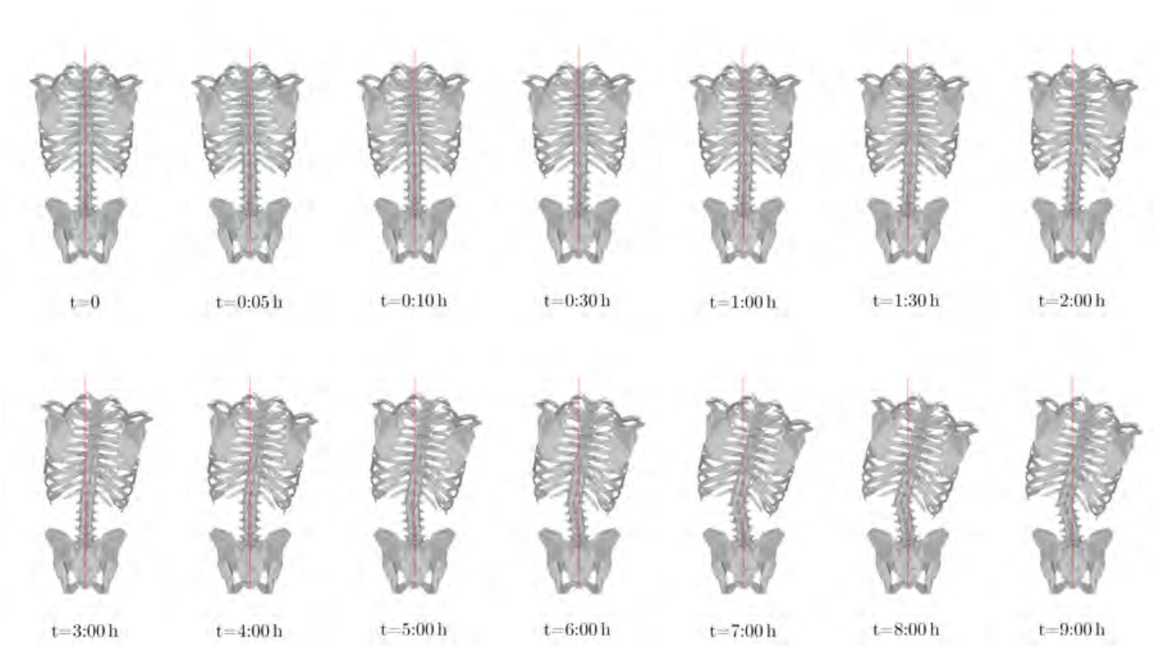


Figure 5.2: Visualization of the sensor output during the first nine hours of testing.

At $t = 0$, the visualization shows no noticeable difference to the straight alignment, even though the sensors indicated settlement 6 s early. After 5 min, very slight left-lateral flexion of both lumbar and thoracic spine is shown. The output seems to stabilize for the first hour but the angle at the thoracolumbar junction indicates inverse rotation for both segments. The thorax then changes the direction of rotation and aligns with the right-lateral flexion of the lumbar spine after 2 h, when the lumbar spine also starts changing its direction to show an opposite lateral flexion of the segments after 5 h. This behavior continues until the thorax is shown almost in a 90° angle to the pelvis, as shown in Fig. D.2.

The sensor stability test shows, that the calculation of the settling phase underestimates the real initialization time. Extending the observed values from 100 to 200 (or even more) or lowering the threshold where the output is harmonized might be simple software solutions to get better output. The sensor's yaw drift itself is a problem that might not be solved entirely using software. Although the drift may be minimized using a special algorithm and further tests show slightly better drift performance when the sensors are aligned equally (as the sensors drift more in pace), additional hardware may result in even better performance. The MPU-9000-series from InvenSense for example integrates the MPU-6050 with an additional 3-axis magnetometer (compass). Internally, the DMP can superimpose the magnetometer output to the gyroscopic output to remove the drift component. If this truly yields a drift-free environment may be tested in the future.

5.2 System Stability

System stability was tested in multiple sessions to ensure a better continuous runtime than the previous prototype, which restarted within a couple of minutes of testing. A total of 25 tests were started, ranging between 21 s and 46 h and most of them longer than one hour. The data-logger was stopped manually for all test runs, except the very first, where the measurement was terminated unexpectedly after 21 s.

During all tests, program loop and sensor response time remained reasonably constant. In the 46 h test from Chapter 5.1 with three connected sensors, values were received every (140 ± 3) ms ($f = (7.15 \pm 0.18)$ Hz). After 10 h, the program loop time reduced with a delay between measurements of (155 ± 14) ms ($f = (6.50 \pm 0.40)$ Hz). At the end of the test, the values were received at an interval of (195 ± 29) ms ($f = (5.19 \pm 0.45)$ Hz). The response times were averaged over a set of 100 measurements. It is noted, that the accuracy of the timing procedure of the program is not known and this might as well factor into the prolonged time between measurements.

In a few cases the data-logger stopped recording the third sensors after it was removed from the holding device used in the tests to ensure initialization in exact identical orientation. When this happened, the sensor was only removed just before finalizing the test to check how the sensor perform differently after some time. Due to the fact, that it only happened after some additional stress was put onto the sensors this may be attributable to a weak data connection, as the sensor itself continued to work.

In one recording, the data-logger wrote random characters into the CSV file after 47 min, instead of writing the values from sensor 2 and 3. This continued for 205 lines in the file. After 36 s of nonsense data, the regular data recording continued flawlessly. All other recordings showed no signs of arbitrary data input.

This result, although containing minor setbacks, is clearly better than the previous prototype. Data can be recorded long enough for multiple simulation session and does continue even after facing problems. In the future, hardware connections may be reviewed to ensure correct sensor output.

5.3 Validity of Measures

Additional sensor tests were conducted to review the accuracy and validity of measures. All three sensors were used simultaneously for this test by aligning them identically and stacking the individual sensors vertically in the holding device. The sensors were then placed on a prepared surface with marked angular measurements ranging from 0° to 180° in 10° steps, along with an intermediate step at 45°.

The yaw, pitch, and roll axes were tested individually by rotating the holder accordingly before initializing the sensors. The sensors were then aligned with the 0° marking on the surface, where the first measurement was taken without any further rotational changes. Between each measurement, the sensors were moved slowly until they were aligned with the next marking. After the full range of 180° was tested, the sensor rotation was slowly reversed by moving the sensors to the 0° position. A last measurement was taken to see whether the initial situation was restored. Fig. 5.3 shows a comparison of the sensor output of yaw, pitch, and roll and the angle of alignment on the surface.

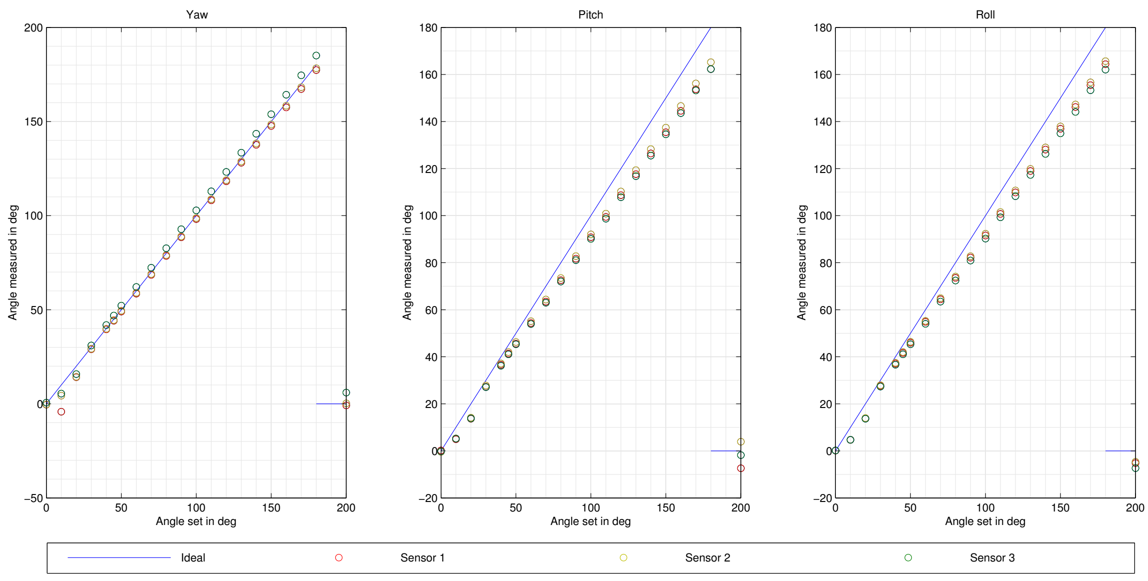


Figure 5.3: Angular measurement of yaw, pitch, and roll.

The yaw measurements show a very close resemblance to the ideal measurements. Sensor 1 and 2 have almost identical output, very close to the ideal line, while the values of sensor 3 are slightly higher. Within the first 20°, a slight drop of the output values below the ideal line is noticeable for all sensors. At 10°, sensor 1 even shows rotation in the wrong direction. This deviation is balanced later on. The deviation of sensor 3 compared to the other sensors is increasing with every measurement. After the sensors are rotated back to their initial position, sensor 3 reads an offset of almost 6°, while sensor 1 and 2 reach 0° with a $\Delta < \pm 1°$.

The slight drop for the 10° and 20° measurement can also be seen in the pitch and roll values. The 30° output is closer to the ideal line but a dropping accuracy is clearly indicated in consecutive measurements. For both pitch and roll, the offset of all three sensors at 180° is $\Delta = -15° \ldots -18°$. When the sensors are returned to their initial orientation the pitch values spread apart to −7.4°, 3.9° and −1.8° for sensor 1, 2, and 3 respectively. The roll angle measurements stay closer together between −5° and −7°.

The test shows very similar behavior between the individual sensors. While only yaw measurements stay close to the ideal values, sensor measurements in all axes stay relatively reliable. Only the last measurement of the pitch angle shows a larger variance between sensors 1, 2, and 3. The reliability of measurements for the pitch and roll axes might indicate to a flawed test-setup, where rotation was not done around the correct axis but with a minimal misalignment. Further investigation has to go into the bad response for the first 20° of every axis.

5.4 System Usability

In the current prototype the workflow of operation has to be followed strictly in order to get usable measurement: First, the sensors have to be connected to the data-logger and aligned equally to allow for identical calibration. They can be placed into a holder for this specific task. After the system was turned on and is fully initialized, the sensors can be fitted carefully to the patient. The system will then record data until it is switch off again or the battery runs out.

With a full battery and three connected sensors the current prototype lasts around 4 hours. The runtime may be increased by removing unused components from the break-out boards like power converters and LEDs, also allowing for a smaller footprint of each sensor. The current prototype is too big to comfortably tape it the the patient's back and a better cable management is needed in further prototypes. The current cables are too long, thick, and inflexible as they do not stretch along with the patient's body.

After the sensors are carefully placed on the patient's body, the visualization software has to be adapted for the patient's spinal curvature, as the sensors might not be in a perfectly vertical orientation. An example of the fitted sensors is shown in Fig. 5.4 along with the live visual feedback. The computer model on the picture shows more forward flexion because the sensor alignment can not be offset manually in the current prototype model. The best way to solve this issue in future prototypes would be an

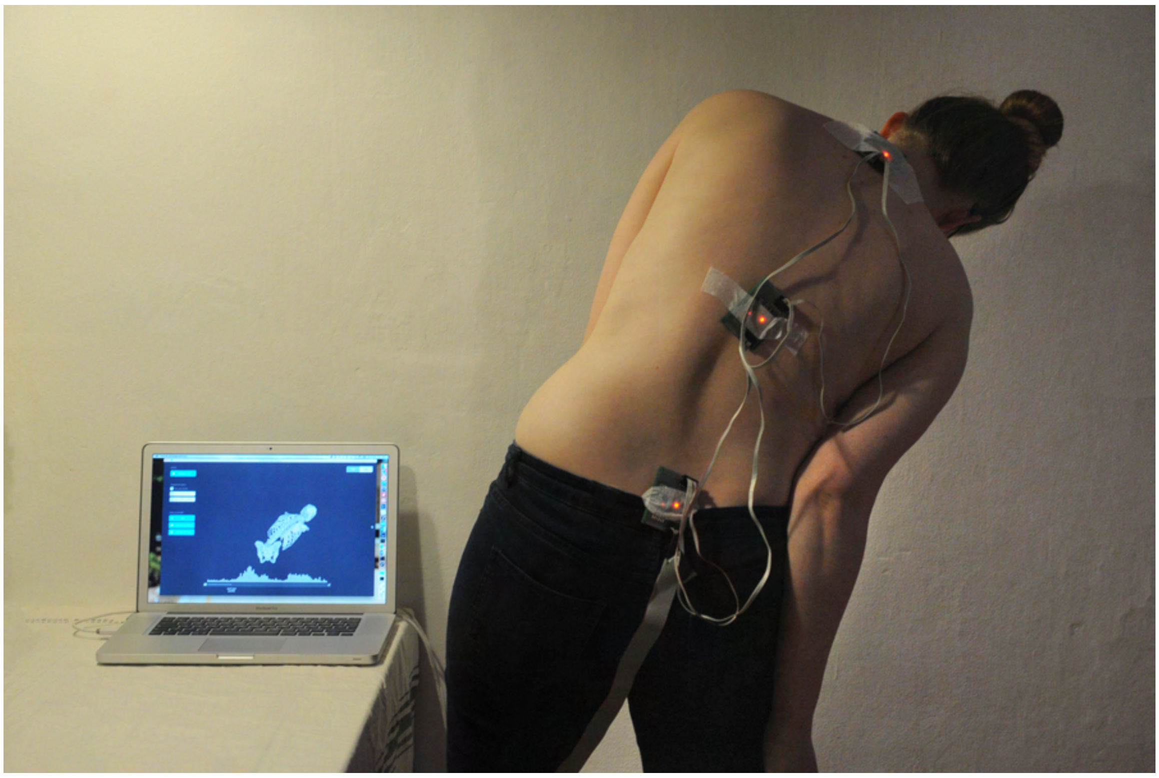


Figure 5.4: Current sensor setup showing a patient with the live feedback on the computer screen.

added on-body calibration feature, allowing the sensors to settle initially and adapt to the individual spinal morphology of the patient.

From a usability aspect, the visualization software is simple to use and understand. While the current prototype is split into two separate programs for live and recorded playback, future prototypes will combine both into a single application. Their use is currently limited to displaying the skeleton model with the identical movement of the patient. Other functions like the bar graph and buttons are currently inactive and display static information only but help to give an idea of the final software.

6 Conclusion

This thesis proposes a method method for spinal monitoring used with standardized patient in medical simulation and education. For this application a small and unobtrusive measuring device is needed in order to allow full immersion into the scenario and hide any indications of the means of evaluation. A prototype was built using gyroscopic sensors and the Arduino platform along with a software to interpret the measurements as visual assistance. The prototype was tested for sensor and system stability, as well as validity of measures and overall usability.

In general, the approach using gyroscopic or inertial measurement to monitor spinal posture proved to be a good option when a small device is needed. By using MEMS components, inertial measurement units provide the best options for hiding the sensors. The two main components of the prototype sensors (IMU and MCU) are available as $4\,\text{mm} \times 4\,\text{mm} \times 0.9\,\text{mm}$ and $5\,\text{mm} \times 5\,\text{mm} \times 1\,\text{mm}$ QFN packages, making it possible to reduce the size of the sensors to a very small overall component of the whole system.

The size of the current prototype sensors is nowhere close to the proposed size of the final product. Compared to the previous prototypes, it is considerably larger, making it unfeasible to use the current prototype hidden (e.g. beneath medical tape or a t-shirt) without being noticed. The prototype sensor is still relatively flat, only adding 1 mm in thickness compared to the prior version. This increase in size was necessary to create a fully working prototype using ready-made breakout and development board. Switching to custom circuit boards and discrete electronic components a smaller size is possible.

The additional microprocessor for each sensor adds the possibility for more preprocessing on the sensor itself. As a result of this, the two sensor limit of the system was increased to a new, theoretical limit of 128 sensors. Measurements can now be received simultaneously from all connected sensors in the desired output format without slow-downs. With the current prototype, the measurements are much more informative and can be used for visualization purposes with both live and pre-recorded playback.

The visualization of the patients spinal posture is simple and easy to understand, as the skeleton model on screen represents the patient directly. Further features, like the histogram indicating overall activity or the colorization of specific vertebrae during movement have yet to be added. Although the developed visualization software is still very limited, these limits can be overcome in future development. With the current prototype, it is possible to represent the simulated patient's spinal posture on screen. Additional calibration features have to be implemented in order to depict everything correctly without offsets, due to the individual spinal morphology of the patient.

Initial performance tests show, that the sensors tend to drift after a short period of time, making long-term measurements inaccurate. Before the sensor drifting begins, overall accuracy and reproducibility of measurements is promising and can be elaborated in further prototypes. The system run-time is only limited by the battery capacity. Unlike the previous prototype there are no severe problems, stopping the recording unexpectedly. Overall usability of the system is acceptable in the current prototype but leaves some room for improvements. On-body sensor calibration, along with better means to place the sensors on the patient's body will be added to the objectives of upcoming development.

The prototype spinal monitoring system developed in this thesis provides the means to measure and display spinal posture in medical simulation scenarios. The system is designed to allow a modular connection of multiple sensors and is currently optimized for the use with three sensors. With a few changes to increase accuracy, reduce size and better algorithms for sensor interpretation as well as more sensors for more detailed measurements, this system might be used in medical simulation in the future. The current prototype provides a solid foundation for this purpose but still needs some refinement before it can be used as a fully operational and stand-alone system.

Bibliography

[1] B. Lenehan, S. Boran, J. Street, T. Higgins, D. McCormack, and A. R. Poynton, "Demographics of acute admissions to a National Spinal Injuries Unit," *European Spine Journal*, vol. 18, no. 7, pp. 938–942, Jul. 2009.

[2] Universitätsklinikum Essen, "Qualitätsbericht der Kliniken 2013/2014," Essen, Tech. Rep., Jul. 2014.

[3] Deutscher Verlag für Gesundheitsinformation GmbH, "Häufige Fragen," http://operation-wirbelsaeule.de/haufige-fragen/, Apr. 2015.

[4] T. Ziegenfuß, "Basis der Notfallmedizin," in *Notfallmedizin*, ser. Springer-Lehrbuch. Berlin, Heidelberg: Springer, 2014, pp. 1–19.

[5] ——, "Grundlegende notfallmedizinische Maßnahmen," in *Notfallmedizin*, ser. Springer-Lehrbuch. Berlin, Heidelberg: Springer, 2014, pp. 49–63.

[6] M. Kreinest, S. Goller, G. Rauch, C. Frank, B. Gliwitzky, C. G. Wölfl, S. Matschke, and M. Münzberg, "Application of Cervical Collars – An Analysis of Practical Skills of Professional Emergency Medical Care Providers," *PLoS ONE*, vol. 10, no. 11, Nov. 2015.

[7] P. Augat, "Biomechanik der Wirbelsäule," in *Chirurgie der verletzten Wirbelsäule*, V. Bühren and C. Josten, Eds. Berlin, Heidelberg: Springer, 2013, pp. 5–18.

[8] S. Standring, *Gray's Anatomy: The Anatomical Basis of Clinical Practice.* Philadelphia, Pa.: Elsevier Health Sciences, Aug. 2015.

[9] A. M. Gilroy, B. R. MacPherson, and L. M. Ross, *Atlas of Anatomy.* Stuttgart: Thieme, Apr. 2012.

[10] D. Resnick, S. Weller, and E. Weller, "Biomechanics of the thoracolumbar spine." *Neurosurgery clinics of North America*, vol. 8, no. 4, pp. 455–469, Oct. 1997.

[11] M. A. Adams and P. Dolan, "Spine biomechanics," *Journal of Biomechanics*, vol. 38, no. 10, pp. 1972–1983, Oct. 2005.

[12] J. Avison, *The World of Physics.* Oxford: Nelson Thornes, Nov. 2014.

[13] R. P. Schlenk, R. J. Kowalski, and E. C. Benzel, "Biomechanics of spinal deformity," *Neurosurgical Focus*, vol. 14, no. 1, pp. 1–15, Jan. 2003.

[14] V. Bühren and C. Josten, Eds., *Chirurgie der verletzten Wirbelsäule: Frakturen, Instabilitäten, Deformitäten*. Berlin: Springer Medizin, 2013.

[15] C. Cooper, E. J. Atkinson, W. MichaelO'Fallon, and J. L. Melton, "Incidence of clinically diagnosed vertebral fractures: A population-based study in Rochester, Minnesota, 1985-1989," *Journal of Bone and Mineral Research*, vol. 7, no. 2, pp. 221–227, Dec. 2009.

[16] E. R. Myers and S. E. Wilson, "Biomechanics of Osteoporosis and Vertebral Fracture:," *Spine*, vol. 22, no. Supplement, pp. 25S–31S, Dec. 1997.

[17] F. Magerl, M. Aebi, S. D. Gertzbein, J. Harms, and S. Nazarian, "A comprehensive classification of thoracic and lumbar injuries," *European Spine Journal*, vol. 3, no. 4, pp. 184–201, Aug. 1994.

[18] F. Holdsworth, "Review Article Fractures, Dislocations, and Fracture-Dislocations of the Spine," *J Bone Joint Surg Am*, vol. 52, no. 8, pp. 1534–1551, Dec. 1970.

[19] C. L. Ewing, A. I. King, and P. Prasad, "Structural Considerations of the Human Vertebral Column under +GZ Impact Acceleration," *Journal of Aircraft*, vol. 9, no. 1, pp. 84–90, Jan. 1972.

[20] N. Bogduk and N. Yoganandan, "Biomechanics of the cervical spine Part 3: minor injuries," *Clinical Biomechanics*, vol. 16, no. 4, pp. 267–275, May 2001.

[21] K. A. Olson and D. Joder, "Diagnosis and treatment of cervical spine clinical instability," *Journal of Orthopaedic & Sports Physical Therapy*, vol. 31, no. 4, pp. 194–206, 2001.

[22] M. M. Panjabi, "The stabilizing system of the spine. Part I. Function, dysfunction, adaptation, and enhancement." *Journal of spinal disorders & techniques*, vol. 5, no. 4, pp. 383–389, 1992.

[23] ——, "The stabilizing system of the spine. Part II. Neutral zone and instability hypothesis." *Journal of spinal disorders & techniques*, vol. 5, no. 4, pp. 390–397, 1992.

[24] C. Josten, "Halswirbelsäule," in *Chirurgie der verletzten Wirbelsäule*, V. Bühren and C. Josten, Eds. Berlin, Heidelberg: Springer, 2013, pp. 135–180.

[25] D. M. Gaba, "The future vision of simulation in health care," *Quality and Safety in Health Care*, vol. 13, no. suppl 1, pp. i2–i10, Jan. 2004.

[26] R. J. Glavin, "When Simulation should and should not be in the Curriculum," in *Clinical Simulation*, R. R. Kyle and W. B. Murray, Eds. Oxford: Academic Press, 2008, pp. 71–75.

[27] G. Breuer and A. Fichtner, "Lernen im Vollzug: Der Erwerb praktischer Fertigkeiten," in *Simulation in der Medizin*, M. St.Pierre and G. Breuer, Eds. Berlin, Heidelberg: Springer, 2013, pp. 71–76.

[28] R. Glavin and N. Maran, "An introduction to simulation in anaestheasia," in *Clinical Teaching – A guide to teaching practical anaesthesia*, J. D. Greaves, Ed. Lisse: Swets and Zeitlinger, 2003.

[29] A. Fichtner, "Lernen für die Praxis: Das Skills-Lab," in *Simulation in der Medizin*, D. M. St.Pierre and G. Breuer, Eds. Berlin, Heidelberg: Springer, 2013, pp. 105–114.

[30] G. A. Shemanko and L. Jones, "To Simulate or not to Simulate: That is the Question," in *Clinical Simulation*, R. R. Kyle and W. B. Murray, Eds. Oxford: Academic Press, 2008, pp. 77–84.

[31] A. Grenvik and J. Schaefer, "From Resusci-Anne to Sim-Man: the evolution of simulators in medicine," *Critical Care Medicine*, vol. 32, no. 2 Suppl, pp. S56–57, Feb. 2004.

[32] K. Kunkler, "The role of medical simulation: an overview," *The international journal of medical robotics + computer assisted surgery: MRCAS*, vol. 2, no. 3, pp. 203–210, Sep. 2006.

[33] C. Epps, M. L. White, and N. Tofil, "Mannequin Based Simulators," in *The Comprehensive Textbook of Healthcare Simulation*, A. I. Levine, S. DeMaria, A. D. Schwartz, and A. J. Sim, Eds. New York, NY: Springer, 2013, pp. 209–232.

[34] K. Schnabel, "Simulation aus Fleisch und Blut: Schauspielpatienten," in *Simulation in der Medizin*, M. St.Pierre and G. Breuer, Eds. Berlin, Heidelberg: Springer, 2013, pp. 115–119.

[35] L. D. Howley, "Standardized Patients," in *The Comprehensive Textbook of Healthcare Simulation*, A. I. Levine, S. DeMaria, A. D. Schwartz, and A. J. Sim, Eds. New York, NY: Springer, 2013, pp. 173–190.

[36] S. F. Chong, "Mass Casualty Incident," in *Disaster Medicine*, D. MacGarty and D. Nott, Eds. London: Springer, 2013, pp. 165–177.

[37] W. Henny, A. H. Buma, R. de Wit, J. M. Ryan, D. G. Burris, A. Brooks, J. Roodenburg, P. Dyer, A. Maas, and C. Bleeker, “Trauma and Surgery,” in *Conflict and Catastrophe Medicine*, A. P. C. C. H. Buma, D. G. Burris, A. Hawley, J. M. Ryan, and P. F. Mahoney, Eds. London: Springer, 2009, pp. 417–492.

[38] S. McLaughlin, S. Clarke, S. Menon, T. P. Noeller, Y. Okuda, M. D. Smith, and C. Strother, “Simulation in Emergency Medicine,” in *The Comprehensive Textbook of Healthcare Simulation*, A. I. Levine, S. DeMaria, A. D. Schwartz, and A. J. Sim, Eds. New York, NY: Springer, 2013, pp. 315–328.

[39] A. Ziv, D. Erez, and H. Berkenstadt, “Clinical Simulation on a National Level: Israel,” in *Clinical Simulation*, R. R. Kyle and W. B. Murray, Eds. Oxford: Academic Press, 2008, pp. 371–375.

[40] M. Rall, T. Manser, and S. K. Howard, “Key elements of debriefing for simulator training,” *European Journal of Anaesthesiology*, vol. 17, no. 8, pp. 516–517, Aug. 2000.

[41] P. Dieckmann, “Gute Nachrede – Debriefing,” in *Simulation in der Medizin*, M. St.Pierre and D. G. Breuer, Eds. Berlin, Heidelberg: Springer, 2013, pp. 153–168.

[42] V. Z. Priel, “A Numerical Definition of Posture,” *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 16, no. 6, pp. 576–584, Dec. 1974.

[43] G. Li and P. Buckle, “Current techniques for assessing physical exposure to work-related musculoskeletal risks, with emphasis on posture-based methods,” *Ergonomics*, vol. 42, no. 5, pp. 674–695, May 1999.

[44] E. R. Bachmann, “Inertial and magnetic tracking of limb segment orientation for inserting humans into synthetic environments,” Ph.D. dissertation, DTIC Document, 2000.

[45] M. D’Amico, R. Bellomo, R. Saggini, and P. Roncoletta, “A 3D spine & full skeleton model for multi-sensor biomechanical analysis in posture & gait,” in *2011 IEEE International Workshop on Medical Measurements and Applications Proceedings (MeMeA)*, May 2011, pp. 605–608.

[46] W. Y. Loebl, “Measurement of spinal posture and range of spinal movement,” *Annals of Physical Medicine*, vol. 9, no. 3, pp. 103–110, Aug. 1967.

[47] B. Glod, “P10010: Motion Tracking Sensor - Establish Target Specifications,” http://edge.rit.edu/edge/P10010/public/Establish%20Target%20Specifications, May 2010.

[48] W. S. Marras, F. A. Fathallah, R. J. Miller, S. W. Davis, and G. A. Mirka, "Accuracy of a three-dimensional lumbar motion monitor for recording dynamic trunk motion characteristics," *International Journal of Industrial Ergonomics*, vol. 9, no. 1, pp. 75–87, Jan. 1992.

[49] W. G. Allread, W. S. Marras, and D. L. Burr, "Measuring trunk motions in industry: variability due to task factors, individual differences, and the amount of data collected," *Ergonomics*, vol. 43, no. 6, pp. 691–701, Jun. 2000.

[50] C. M. Petersen, R. D. Johnson, and D. Schuit, "Reliability of cervical range of motion using the OSI CA 6000 Spine Motion Analyser on asymptomatic and symptomatic subjects," *Manual Therapy*, vol. 5, no. 2, pp. 82–88, May 2000.

[51] Sels Instruments N.V., "The BodyGuard System," http://www.sels-instruments.be/bodyguardsystem.html, Feb. 2016.

[52] C. Goodvin, E. J. Park, K. Huang, and K. Sakaki, "Development of a real-time three-dimensional spinal motion measurement system for clinical practice," *Medical and Biological Engineering and Computing*, vol. 44, no. 12, pp. 1061–1075, Nov. 2006.

[53] W. Y. Wong and M. S. Wong, "Detecting spinal posture change in sitting positions with tri-axial accelerometers," *Gait & Posture*, vol. 27, no. 1, pp. 168–171, Jan. 2008.

[54] A. Lopez-Quintana, J. Tejero-Calado, and F. Vidal-Verdu, "Accelerometer Array and Method to Obtain a 3D Representation of the Spine Posture," in *2011 International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA)*, Oct. 2011, pp. 466–471.

[55] Xsens Technologies B.V., "MTw Development Kit - Products," https://www.xsens.com/products/mtw-development-kit/, Feb. 2016.

[56] Thales Visionix, Inc., "Intersense InertiaCube4," http://www.intersense.com/pages/18/234/, Feb. 2016.

[57] NexGen Ergonomics Inc., "I2M Motion Tracking Inertial Measurement Units (IMUs)," http://www.nexgenergo.com/ergonomics/I2M-IMUs.html, Feb. 2016.

[58] UpRight Technologies Ltd., "Product," http://www.uprightpose.com/product/, Feb. 2016.

[59] Hocoma AG, "Medical Device for Back Therapy at Home - Valedo®," https://www.valedotherapy.com/de_en/, Feb. 2016.

[60] J. Bandow, B. Krystek, and T.-O. Meyer, "Wearable Electronic Sensor Array for Real Time Spine and Posture Analysis Using MEMS Gyroscopes and Advanced Computer Visualization," unpublished.

[61] NXP Semiconductors, "I^{2}C-bus specification and user manual, UM10204, Rev. 6," Apr. 2014.

[62] Atmel Corporation, "ATtiny25/V / ATtiny45/V / ATtiny85/V - Atmel 8-bit AVR Microcontroller with 2/4/8K Bytes In-System Programmable Flash, ATtiny25/45/85 [DATASHEET], Rev. 2586Q–AVR–08/2013," 2013.

[63] ——, "ATmega48A/PA/88A/PA/168A/PA/328/P, Atmel-8271J-AVR-ATmega-Datasheet_11/2015," 2015.

[64] InvenSense Inc., "MPU-6000 and MPU-6050 Product Specification, PS-MPU-6000A-00, Rev. 3.4," 2013.

[65] Arduino LLC, "Reference Wire Library," https://www.arduino.cc/en/Reference/Wire, Jan. 2016.

[66] J. Rowberg, "I2C Device Library," http://www.i2cdevlib.com, Jan. 2016.

[67] S. Mishra, N. K. Singh, and V. Rousseau, *System on Chip Interfaces for Low Power Design.* Burlington, Mass.: Morgan Kaufmann, Nov. 2015.

[68] T. E. Kurt, "SoftI2CMaster," https://github.com/todbot/SoftI2CMaster, Nov. 2015.

[69] A. Dementyev, "SensorTape," https://github.com/vincimrs/SensorTape/blob/master/Slave/MPU6050/I2Cdev.cpp, Jan. 2016.

[70] J. P. Morais, S. Georgiev, and W. Sprößig, *Real Quaternionic Calculus Handbook.* Basel: Springer, 2014.

[71] PJRC.COM, LLC., "Teensy LC - Low Cost," https://www.pjrc.com/teensy/teensyLC.html, Mar. 2016.

[72] C. Reas and B. Fry, *Processing: A Programming Handbook for Visual Designers and Artists.* Cambridge, Mass.: The MIT Press, 2007.

[73] MEDI-LEARN.net GbR, "Medi-Learn Anatomie Skript," http://www.medi-learn.de/ana5-1, May 2015.

A Spinal Injuries According to Magerl

Table A.1: Type A. Vertebral body compression [17].

- A1 Impaction fractures
  - A1.1 Endplate impaction
  - A1.2 Wedge impaction fractures
    - 1 Superior wedge impaction fracture
    - 2 Lateral wedge impaction fracture
    - 3 Inferior wedge impaction fracture
  - A1.3 Vertebral body collapse
- A2 Split fractures
  - A2.1 Sagittal split fracture
  - A2.2 Coronal split fracture
  - A2.3 Pincer fracture
- A3 Burst fractures
  - A3.1 Incomplete burst fracture
    - 1 Superior incomplete burst fracture
    - 2 Lateral incomplete burst fracture
    - 3 Inferior incomplete burst fracture
  - A3.2 Burst-split fracture
    - 1 Superior burst-split fracture
    - 2 Lateral burst-split fracture
    - 3 Inferior burst-split fracture
  - A3.3 Complete burst fracture
    - 1 Pincer burst fracture
    - 2 Complete flexion burst fracture
    - 3 Complete axial burst fracture

Table A.2: Type B. Anterior and posterior element injury with distraction [17].

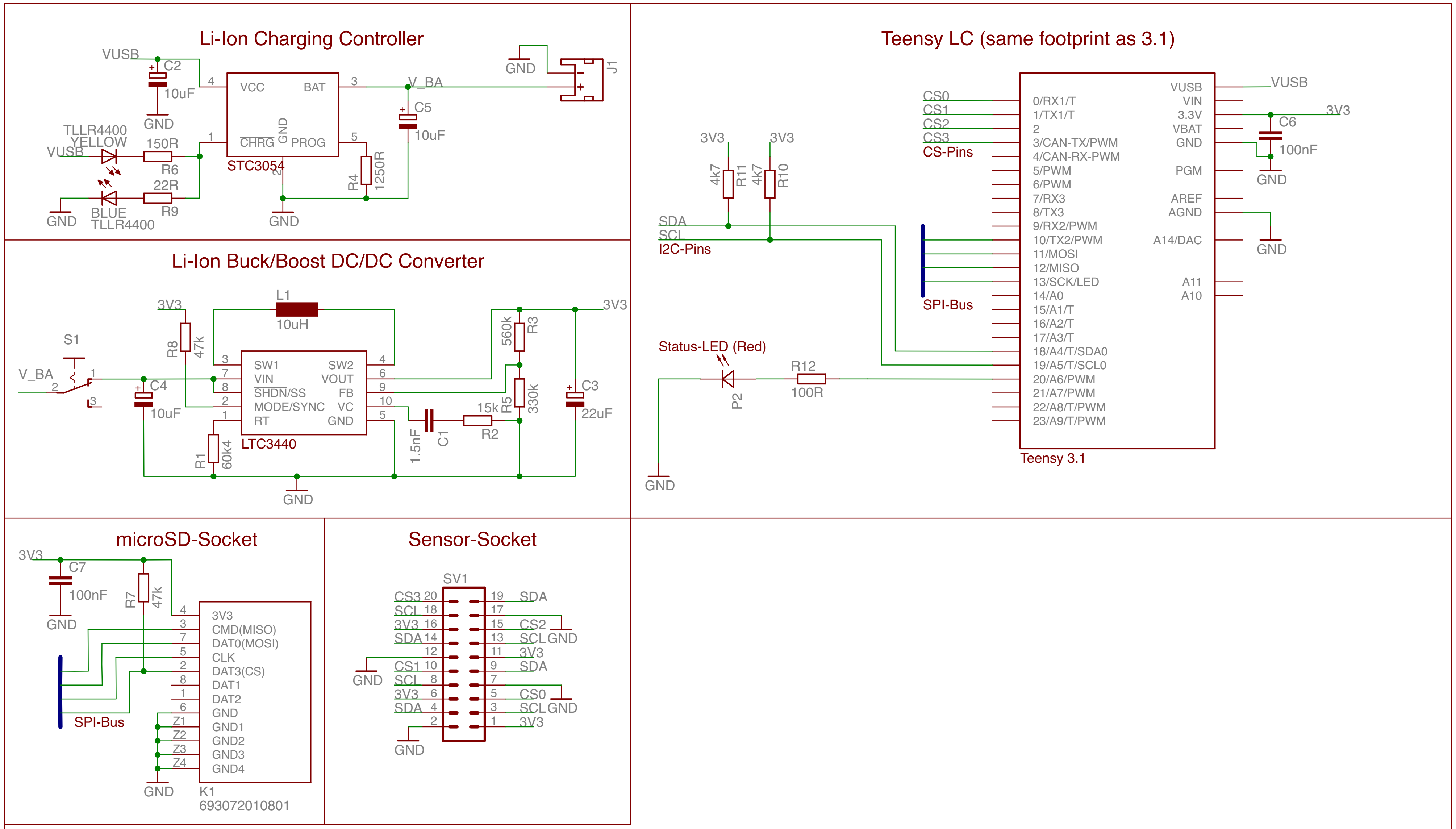
- B1 Posterior disruption predominantly ligamentous (flexion-distraction injury)
  - B1.1 With transverse disruption of the disc
    - 1 Flexion-subluxation
    - 2 Anterior dislocation
    - 3 Flexion-subluxation/anterior dislocation with fracture of the articular processes
  - B1.2 With type A fracture of the vertebral body
    - 1 Flexion-subluxation + type A fracture
    - 2 Anterior dislocation + type A fracture
    - 3 Flexion-subluxation/anterior dislocation with fracture of the articular processes + type A fracture
- B2 Posterior disruption predominantly osseous (flexion-distraction injury)
  - B2.1 Transverse bicolumn fracture
  - B2.2 With transverse disruption of the disc
    - 1 Disruption through the pedicle and disc
    - 2 Disruption through the pars interarticularis and disc (flexion-spondylolysis)
  - B2.3 With type A fracture of the vertebral body
    - 1 Fracture through the pedicle + type A fracture
    - 2 Fracture through the pars interarticularis (flexion-spondylolysis) + type A fracture
- B3 Anterior disruption through the disc (hyperextension-shear injury)
  - B3.1 Hyperextension-subluxations
    - 1 Without injury of the posterior column
    - 2 With injury of the posterior column
  - B3.2 Hyperextension-spondylolysis
  - B3.3 Posterior dislocation

Table A.3: Type C. Anterior and posterior element injury with rotation [17].

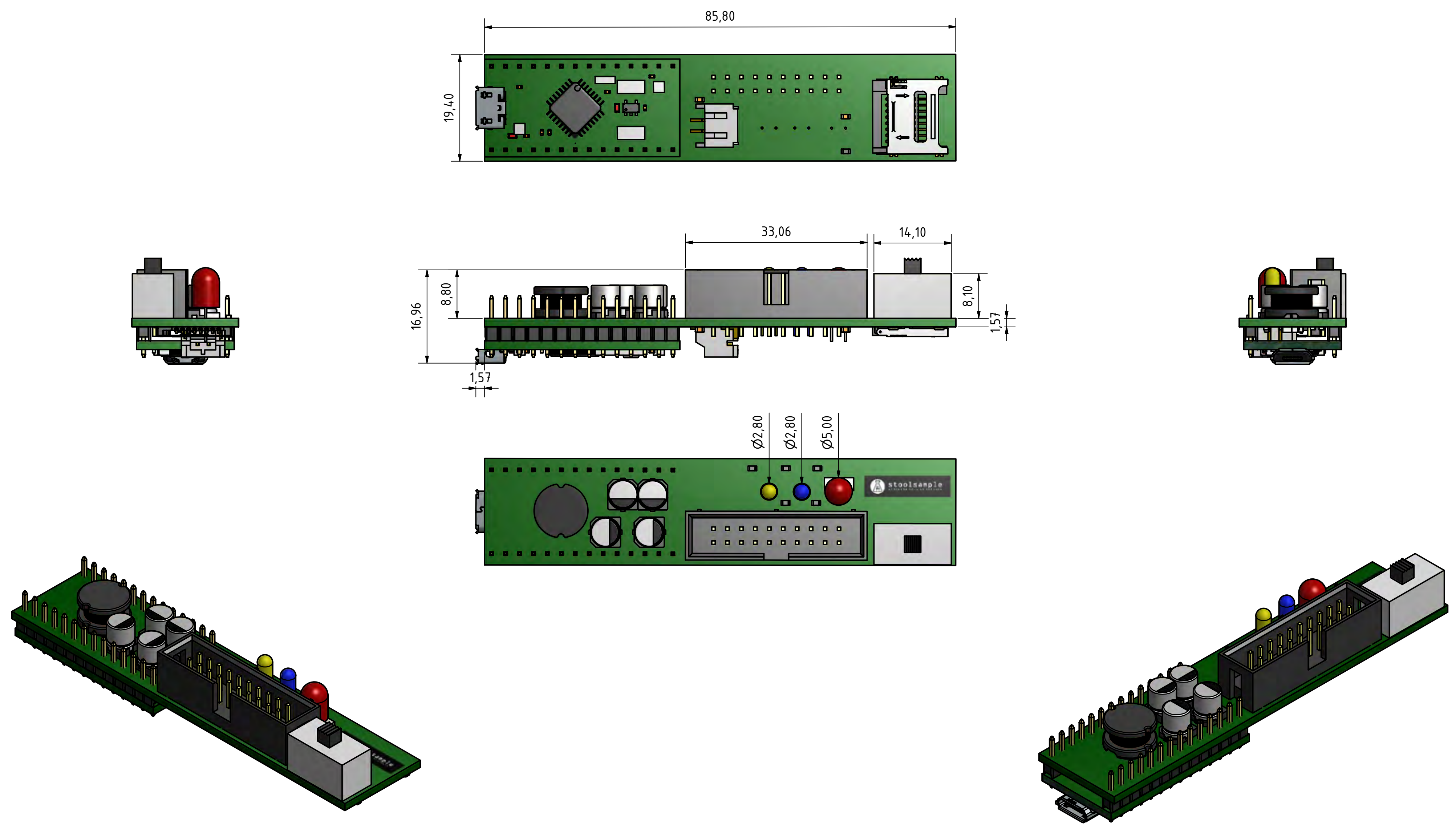
- C1 Type A injuries with rotation (compression injuries with rotation)
  - C1.1 Rotational wedge fracture
  - C1.2 Rotational split fractures
    1. Rotational sagittal split fracture
    2. Rotational coronal split fracture
    3. Rotational pincer fracture
    4. Vertebral body separation
  - C1.3 Rotational burst fractures
    1. Incomplete rotational burst fracture
    2. Rotational burst-split fracture
    3. Complete rotational burst fracture
- C2 Type B injuries with rotation
  - C2.1 B1 injuries with rotation (flexion-distraction injuries with rotation)
    1. Rotational flexion subluxation
    2. Rotational flexion subluxation with unilateral articular process fracture
    3. Unilateral dislocation
    4. Rotational anterior dislocation without/with fracture of articular processes
    5. Rotational flexion subluxation without/with unilateral articular process fracture + type A fracture
    6. Unilateral dislocation + type A fracture
    7. Rotational anterior dislocation without/with fracture of articular processes + type A fracture
  - C2.2 B2 injuries with rotation (flexion distraction injuries with rotation)
    1. Rotational transverse bicolumn fracture
    2. Unilateral flexion spondylolysis with disruption of the disc
    3. Unilateral flexion spondylolysis + type A fracture
  - C2.3 B3 injuries with rotation (hyperextension-shear injuries with rotation)
    1. Rotational hyperextension-subluxation without/with fracture of posterior vertebral elements
    2. Unilateral hyperextension-spondylolysis
    3. Posterior dislocation with rotation
- C3 Rotational-shear injuries
  - C3.1 Slice fracture
  - C3.2 Oblique fracture

B Schematics

This Appendix has the schematics for the data logger board and the housing. Full technical drawings are placed on the following pages.



Sheet Title:	Data-Logger		
Project Title:	Spine Monitor		
Size:	A4	Revision:	3.0
Date:	13.03.16 20:09	Sheet:	1/1
Author:	Tim-Oliwer Meyer		



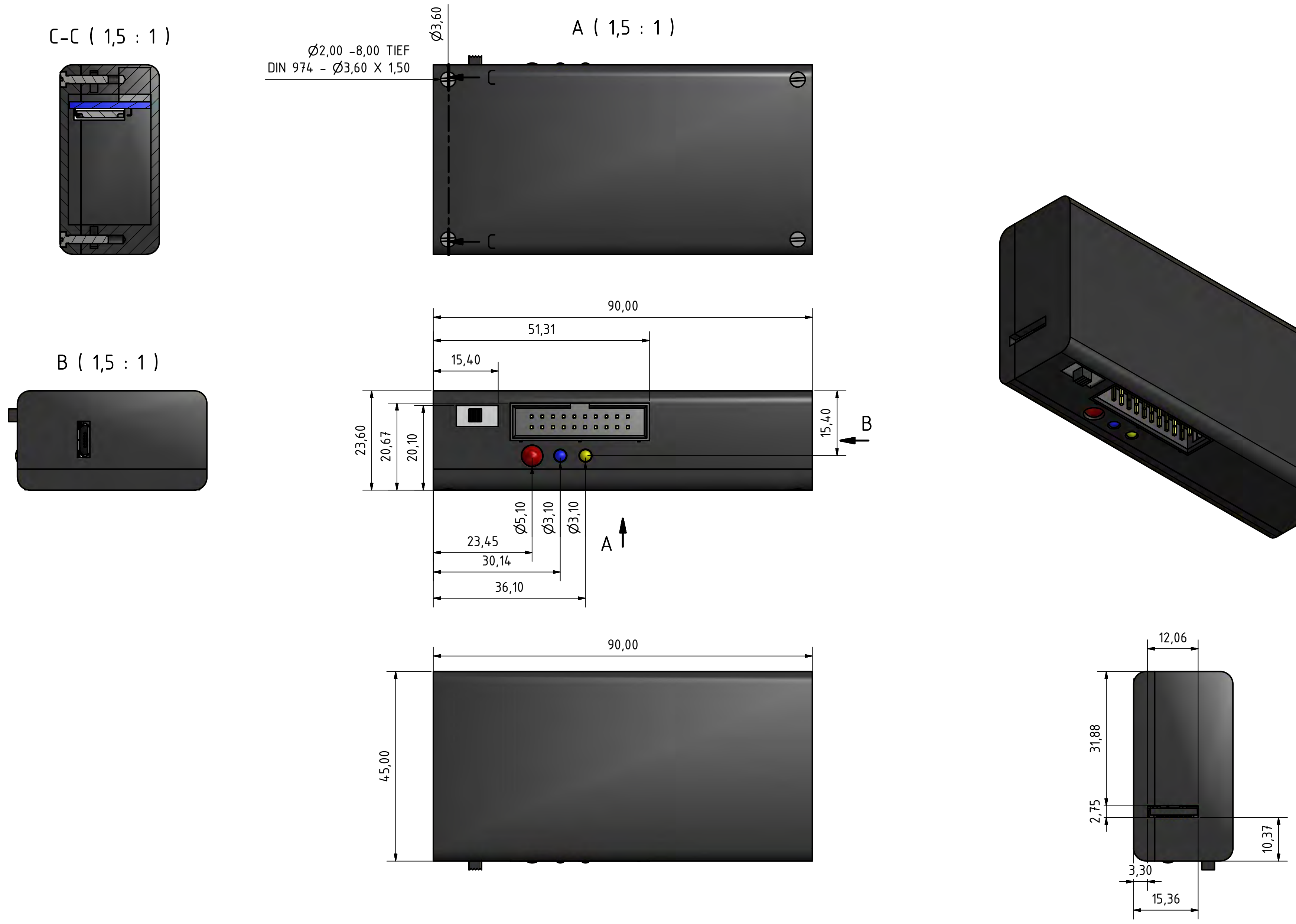
	Datum	Name
Gezeichnet	15.06.2015	Tim-Oliwer Meyer
Kontrolliert		
Norm		

Status	nderungen	Datum	Name

main_pcb

1

A2



	Datum	Name			
Gezeichnet	15.06.2015	Tim-Oliver Meyer			
Kontrolliert					
Norm					
			messbox_v2		1
					A2

Status	nderungen	Datum	Name

C Contents of enclosed CD-ROM

The enclosed CD-ROM contains the source code of the sensor, data-logger, and visualization programs.

D Additional Plots

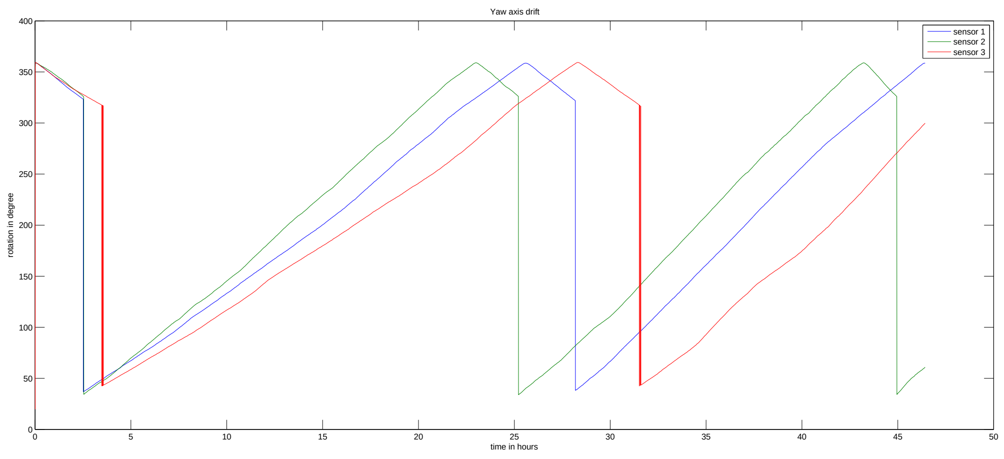


Figure D.1: Yaw drift from the sensor stability test.

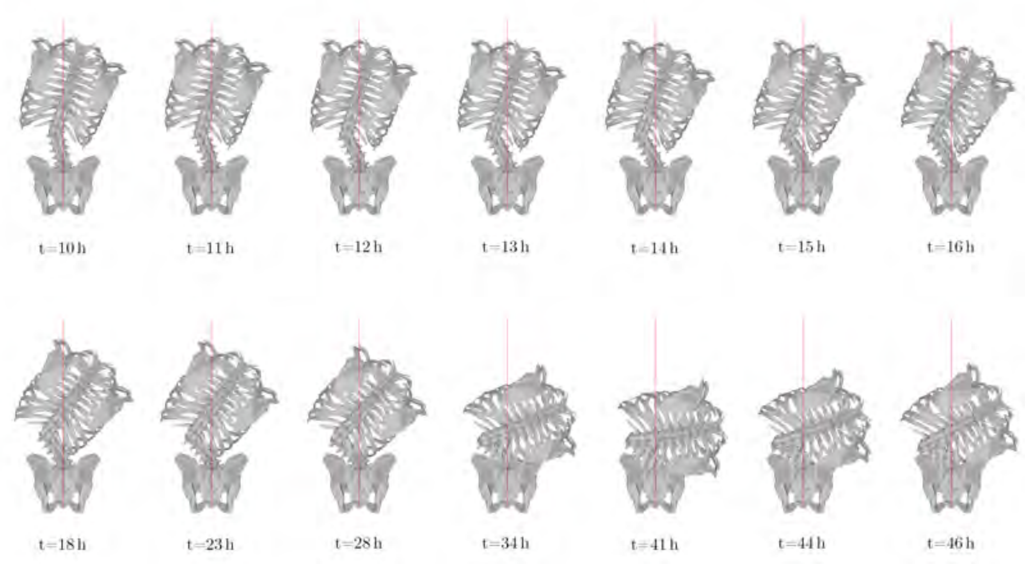


Figure D.2: Visualization of the sensor output of the sensor stability after 10 h of testing.