Tracking limbs motion using a wireless network of inertial measurement units

Pasquale Buonocunto, Mauro Marinoni
Scuola Superiore Sant'Anna, Pisa, Italy
Email: {p.buonocunto, m.marinoni}@sssup.it

Abstract— **Tracking the position and the orientation of human limbs to reconstruct postures and actions is becoming a crucial need in several application domains, including medicine, rehabilitation, sport, and games. However, most available solutions are expensive, imprecise, or require an instrumentation of the environment. This paper presents a low-cost tracking system based on a set of wearable inertial measurement units (IMUs) coordinated as a wireless body area network. After the system description, the characterization of the single node is provided through a set of experiments. Issues related to real-time processing, calibration, data synchronization, and energy consumption are introduced using a preliminary simplified setup with two nodes.**

I. Introduction

Precise tracking of limb motion has always been a crucial requirement for a wide range of applications, like telemedicine, orthopaedic and neurological rehabilitation, sports training, interactive games, and virtual reality. However, for long time application developers have been forced to choose between low-cost solutions with poor performances and very expensive high-precision approaches requiring an heavy environment instrumentation, like vision-based motion capture. Inertial Measurement Units (IMUs) based on MicroElectroMechanical Systems (MEMS) technology changed the approach toward the motion tracking challenge, because they allow acquiring the body dynamic with small integrated circuits and without external hardware to instrument the environment. IMUs also provide the possibility to get rid of some critical constraints, like the requirements of a line of sight to the camera or the impossibility for the user to freely move in the scene. Initially, MEMS performance was poor and often wiring were required between sensors, processing units and transmitters. Recently, the market has been populated by new generations of IMUs allowing higher quality in the measurement together with onboard preprocessing. These devices, together with latest technologies in communication and embedded systems, offer new opportunities previously not possible in wearable systems.

Contributions: This paper presents a very flexible, low-cost tracking system based on inertial measurement units and latest cutting-edge technology for ultra low-power wireless applications. The architecture is described in terms of hardware platform, system software and tracking application. A first set of experiments is reported to show the quality of the acquired data and the preliminary results indicate that the proposed approach can effectively be used to obtain a good tracking of body movements.

Paper structure: The remaining of the paper is organized as follows. Section II discusses some related work regarding various technologies used in body motion tracking and presents advantages and drawbacks of the considered approaches. Section III describes the system architecture and the implemented software solutions for dealing with real-time processing, wireless communication and data synchronization. Section IV illustrates the issues of integrating kinematic constraints for the correct reconstruction of limb postures. Section V reports some experimental results carried out to evaluate the precision of the system. Finally, Section VI states our conclusions and future work.

II. Related Work

Different approaches have been used to track limbs motion with different precision, cost, and complexity. They are briefly summarized below.

- **Optical systems**. They use visual data captured by one or more cameras to triangulate the 3D position of a set of points detected, with the aid of markers attached to the body. Some of them can reach a high precision (i.e, in the range of few millimeters), but also have very high costs, like for example the Vicon [42]. A cheaper solution is represented by Microsoft Kinect [31], which uses only one RGB camera and an infrared depth sensor composed by an infrared laser grid scanner and a related infrared camera. It is not as precise as the more expensive systems: it precision is in the order of centimeters instead of millimeters, and has a limited maximum range (5 m). It also suffers by all typical disadvantages of optical systems, the most important are the need of instrumenting the scene, preparing a setup of cameras and the possibility of operating properly only in the presence of line of sight between the cameras and the object being tracked. There are others kind of systems that exploit the high frequency interference patterns caused by lasers [48] and sense direct or reflected light using a high framerate lensless 2D image sensor. They can be used to track fast (i.e,

This work has been partially supported by Telecom Italia.

up to 1000Hz) motion using minimal hardware. Despite their lower cost and much simpler hardware requirement than camera-based sensor, these systems still need the instrumentation of the scene.

- **Exo-skeleton systems**. They are rigid structures of jointed, straight metal or plastic rods linked together with potentiometers or encoders that articulate at the joints of the body [39]. These systems offer real-time, high precision acquisition and have the advantage of not being influenced by external factors [30], such as visual occlusion, quality and the number of cameras, and they have no limit on maximum capture volume. Some systems also provide haptic limited force feedback. Their main disadvantage is the movement limitation imposed by the mechanical constraints of the exoskeleton structures [15].
- **Magnetic systems**. They calculate the position and orientation of a magnetic sensor probe moving in a magnetic field generated by three orthogonal coils. The main disadvantage of these systems is that they are susceptible to magnetic and electrical interference from metal objects in the environment, which affect the magnetic field, or electromagnetic sources, such as monitors, lights, cables and computers [33].
- **Inertial systems**. Inertial Motion Capture [35] technology is based on miniature inertial sensors, biomechanical models, and sensor fusion algorithms. The motion data of the inertial sensors (inertial guidance system) is often transmitted via wireless to a computer, where the motion is recorded or viewed. Most inertial systems use gyroscopes to measure rotational speed. These rotations are translated to a skeleton by the software. Inertial motion systems capture the full six degrees of freedom body motion of a human in real-time and can also include a magnetic sensor, although these have a much lower resolution and are susceptible to electromagnetic noise. Benefits of using inertial systems include low cost, small dimensions, portability, and large capture areas. Disadvantages include lower positional accuracy and positional drift.

A. Proposed approaches to IMU data integration

The advantages of inertial sensors have helped their rapid adoption in a wide range of applications. To solve the issues related to noise, drift, random walk, etc., a flourishing literature on how to filter and integrate the available parameters has been produced. In the following, we summarize some of the data fusion approaches proposed in the literature.

- **Accelerometer only**. Using just triaxial accelerometer alone, as done by Lee in [23], joint angles can be determined by reconstructing the direction of the gravity vector. This approach, however, can only be used for motions involving just small linear accelerations because gravity affects accelerometer measurement all the time, and this becomes much more noisy when movements become faster.
- **Biaxial accelerometer and one gyroscope**. This approach has been used by Sabatini in [37], who exploited the cyclical feature of human gait to measure some of its spatio-temporal parameters, like stride length and walking speed. Since the accelerometers are unable to detect rotations around the vertical axis, all the motion is assumed to take place in a non-rotating sagittal plane. Being this method highly application specific, it is not suitable for a more general use.
- **Accelerometer, gyroscope and magnetometer**. This is the most general and complete way to produce 3-D orientation estimates relative to an Earth-fixed reference frame, using MEMS technology. Data from a sensor module containing a triad of orthogonally mounted linear accelerometers, a triad of orthogonally mounted angular rate sensors, and a triad of orthogonally mounted magnetometers are processed by suitable data-fusion filters, it is possible to exploit the strength of each type of sensor and overcome their weaknesses.
- **Dual triaxial accelerometers**. This is a quite uncommon approach using only two spatially separated triaxial accelerometers [41]. Error accumulation and drift problems are mitigated with the use of domain specific knowledge, including pause identification and geometric constraints on the two nodes. A major disadvantage of this system is that it is designed to work only when the two accelerometers are located on opposite corners of the bounding cube, so it must be re-tuned when using different geometries. Furthermore, the distance between the two sensors cannot be too small, making this approach not feasible for small wearable sensors.

A variety of filtering techniques have been adopted for data integration, some of which are listed below.

- **Complementary filter.** These filters follow a frequency-based approach, and this is one of the first methodologies used to address these issues [16]. The key idea is to threat one signal through a low-pass filter, the other one through a high-pass filter, and combine them to obtain the final rate. In case of IMUs, it can be better to combine slow moving signals from the accelerometer and magnetometer, and fast moving signals from the gyroscope. The result is to favor accelerometer measurements of orientation at low angular velocities and the integrated gyroscope measurements at high angular velocities. Such an approach is simple but may only be effective under limited operating conditions.
  Bachman *et al.* [3] and Mahony *et al.* [27] presented separate algorithms that both employ a complementary filter process, using adaptive parameters. This algorithm structure has been shown to provide a good trade-off between effective performance and computational expense.
- **Kalman filter.** The Kalman filter [19] [46] has become a basis for the majority of orientation algorithms and commercial inertial orientation sensors, like Xsens [44], Intersense [17], and many others. The widespread use of

Kalman-based solutions is a guarantee of their accuracy and effectiveness [11], [25], [28], [36], [22].
Nevertheless, this method presents some disadvantages. First, the Kalman filter implementation imposes an high computational load due to lot of recursive formulas that need to be calculated to minimize the least mean squared error [43]. Furthermore, different kinds of Kalman filters are needed for different state vectors, dynamic models, and measurement models; and with the increase of accuracy of the result, larger dimension state vectors are needed, that will lead to higher computation demand.
These challenges require a large computational load for implementing Kalman-based solutions, making it an not optimal choice for small low-power microcontrollers.
- **Madwick filter.** The algorithm uses a quaternion representation, allowing accelerometer and magnetometer data to be used in an analytically derived and optimized gradient descent algorithm to compute the direction of the gyroscope measurement error as a quaternion derivative [26]. The algorithm achieves levels of accuracy matching that of Kalman-based methods, but with much lower computational load, and the ability to operate at small sampling rates. These improvements significantly reduce the hardware and computational power necessary to implement the Madwick filter, and this is important when battery duration is a key point in the design of a sensor node.

B. Similar existing platforms

There are several commercial inertial tracking products available today. Typical full body tracking systems, such as the Moven motion capture suite by Xsens [44], or similar product by Intersense [17], and many others. They are very expensive (in the range of 10K euro and beyond), and are not flexible because they cannot be customized and expanded (e.g., adding other kind of sensors), and use proprietary wireless communication protocols that requires special hardware.

There are also several academic works involving motion tracking using wireless nodes with onboard IMU. Nevertheless, most of them are highly application specific [20], [1], [8], [40], [34], [47], [2] focusing on data processing rather than on the sensor node hardware and firmware, because they use already available nodes and the output of their sensors.

Furthermore, they use hardware platforms based on old components that do not exploit the potential of new technologies, both hardware and software (e.g., radio integrated in microcontroller, single chip 9-DoF IMU chip, very low power consumption, real-time processing and communications), that can be very useful for the final application. In particular, the solutions presented in [47], [45], [2] used nodes with old hardware (microcontroller, sensors and wireless) and this translates to not-so-low power consumption compared with the processing power provided.

Concerning wireless communication, most of them uses ZigBee [20], [8], thus not being compatible with pc/tablet without the use of an external dongle. Other works, like [2] and [40] use custom protocols over 2.4Ghz or even 433Mhz [45]. All the works considered uses hardware platforms with different sensors chip for accelerometer, gyroscope and magnetometer (sometimes even with analog outputs). Instead, having a single chip including all of these sensors reduces inevitable errors due to non-optimal axis alignment among various chips, and the chip should have a digital output to reduce additional noise on ADC conversion. Finally, none of them uses a microcontroller with an embedded radio peripheral in its chip, nor real-time policies for processing and wireless communications; these measures positively impact on the overall efficiency, on power consumption and system predictability.

Recently, aided by the trend of integrating multiple sensors in the same package, hardware producers started proposing solutions including filtering and data fusion capabilities. This approach reduces noises and the computation power required by the microcontroller, as well as power consumption. One of these solutions has been integrated by InvenSense in their IMU sensors [18]. They proposed a proprietary onboard filter called Digital Motion Processor (DMP) that will be better explained in Section III.

III. System description

This section describes the elements composing the platform used in the proposed approach. Body movements are tracked by integrating angular information acquired from a set of nodes, each monitoring a single rigid limb segment. The rest of this section first presents the functional and technical requirements that had driven the hardware design; then describes various features of the platform in detail.

The platform development has been driven by the following design objectives:

- **Accuracy**. When monitoring limb positions and movements, a crucial aspect is to guarantee a good level of precision in the measurements. For example, in knee telerehabilitation applications, the flexion/extension angle must be typically monitored with a precision of 1 degree.
- **Flexibility**. The main idea is to design a platform that is flexible enough to be adapted to different applications, in such a way that will not be necessary to redesign it every time. For example, it should be possible to use it in various use cases, like tracking different human limbs or a robotic manipulator. Furthermore, this platform is based on the latest hardware and software technologies, and has an expansion slot to handle several additional sensors.
- **Real-time feedback**. If data are presented to the user as feedback, it is also important to guarantee a bounded delay of a few hundred of milliseconds between movement and visualization, especially in the presence of more than one sensor, each of them with different timing requirements. A similar requirement applies in a wide range of applications, from gaming to remote teleoperated systems.
- **Low-power consumption**. Sensor nodes should provide a good balance between lifetime and dimensions. In

some applications is not possible to stop the activities to recharge the sensor or change batteries. However, large battery packs disturb the measurements (i.e., different behavior while wearing the sensors) and reduce the quality of the user experience.

A. Hardware platform

The platform is composed by a set of wearable sensor nodes sending data to a central unit (master) that performs the integration and provides the posture information to applications. In the current implementation, the central unit is a high-end system that runs the application under the Android Operating System [14]. The number of sensor nodes depends on the number of joints that needs to be monitored. Figure 1 illustrates the block diagram of the main logical components of a node.

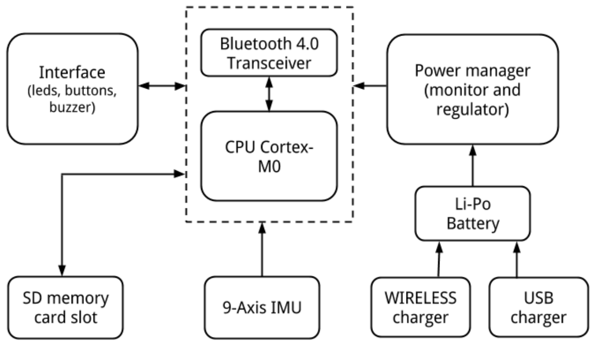

Figure 1: Block diagram of the main node components.

Every node is powered by an ultra low-power Nordic nRF51822 microcontroller with embedded 2.4Ghz transceiver [32]; an InvenSense MPU-9150 9-axis IMU; an integrated onboard chip-antenna with 20m range indoors/80m range outdoors; an USB port; a microSD card for logging and debugging; some I/O devices (3 LEDs, 2 buttons and a buzzer); and six exported GPIO (that can be used both for digital I/O or as analog input with ADC) to have the flexibility to expand the board with other sensors, like ECG, EMG, etc. The node is powered by a single cell LiPo battery that guarantees more than 20 hours of continuous use and can be charged using both USB or wireless recharge. The nRF51822 microcontroller is built around a 32-bit ARM Cortex-M0 CPU with 256kB flash and 16kB RAM, and incorporates a rich selection of analog and digital peripherals. The embedded 2.4GHz transceiver supports Bluetooth 4.0 low energy, as well as 2.4GHz raw operation. The nRF51822 microcontroller is targeted for low-energy applications and supports several low-power operating modes with consumes ranging from 420nA in deep sleep without RAM retention to 13mA in RX mode at 1Mbps.

The Inertial Measurements Unit is an InvenSense MPU-9150, which combines a 3-axis MEMS gyroscope, a 3-axis MEMS accelerometer, a 3-axis MEMS magnetometer, and a DMP hardware accelerator engine. The MPU-9150 features three 16-bit analog-to-digital converters (ADCs) for digitizing the gyroscope outputs, three 16-bit ADCs for digitizing the accelerometer outputs, and three 13-bit ADCs for digitizing the magnetometer outputs. The DMP acquires data from all sensors and computes the quaternion representing the chip orientation with respect to Earth fixed frame with a selectable output frequency up to 200Hz. An on-chip 1024-byte FIFO buffer helps lowering system power consumption by allowing burst reading from the microcontroller. This sensor is connected to the microcontroller through I^2C bus.

B. Wireless protocols

The radio subsystem integrated in the Nordic microcontroller allows selecting between different wireless communication technologies, in particular:

- **Bluetooth 4.0 Low Energy** (BLE) is the new extension of Bluetooth that is intended for strongly energy-constrained applications, such as sensors or disposable devices;
- **ANT** is a low-power proprietary wireless technology developed focusing on sports and fitness sensors that have to communicate with a display unit, for example a watch or cycle computer. Similar to BLE, ANT devices may operate for years on a coin cell.
- **ZigBee** is a low-power wireless specification based on mesh networking to the low-power wireless space and is targeted towards applications such as smart meters, home automation, and remote control units.

While the first two protocol stacks are provided by Nordic in the form of binary-only library (i.e. SoftDevice), the ZigBee one is included in the very latest Erika RTOS kernel distribution [12]. An interesting feature of the BLE SoftDevice is the possibility to reserve some timeslots for accessing the radio peripheral in raw mode, temporarily bypassing the whole BLE stack. This can be used to run different radio protocols concurrently to BLE, and this is useful for several reasons:

- to have a faster, lower latency communication among nodes using custom protocols, including those that can provide temporal guarantees.
- to allow the use of just a single node in BLE mode to transmit collected data to the master (pc/tablet),
- to connect the nodes to other wireless networks that use different protocols.

Other possible solutions are represented by WiFi and Bluetooth. Compared to the ones supported by the Nordic chip, they have a better data throughput but at the cost of high power consumption, which is one of the primary concerns of this platform. Furthermore, the implementation of WiFi and Bluetooth needs an external hardware module in addition to the Nordic microcontroller, and this contributes negatively to energy consumption, device physical dimensions, firmware complexity, and costs.

For this platform, the selected technology has been the BLE for a number of reasons, including its widespread adoption, lower power consumption, better immunity to interferences, and net payload data-rate [21]. Considering the sensors on the platform and typical acquisition rate, the bandwidth provided by the BLE protocol resulted to be sufficient for most of the applications.

Our current implementation exploits raw time slots, but only to achieve time synchronization (as explained in Section III-D). The data transfer to the master (pc/tablet) is done using BLE from every single sensor node.

C. Real-time processing

In complex embedded devices like the one described in this paper, the system must be able to execute several concurrent activities, some of which subject to strict timing constraints, while allowing a modular and flexible development cycle. For instance, IMU data are acquired with a sampling period of 5-10 milliseconds, while data transmission and data storage tasks can be triggered with a lower frequencies and present lower criticality. For example, the data storage function is necessary to save motion data when the connection is not available. An RTOS is needed especially in the presence of more than one sensor with different timing constraints, as well as to maintain a good code clarity and modularity. Furthremore, an RTOS is crucial to predictably handle the radio reservation in order to mix the BLE communication with custom protocols.

To ensure a timely execution of such different activities and perform an off-line guarantee of the timing properties of the application, the software has to be supported by a real-time operating system. The software for the platform presented in this paper has been developed on top of the ERIKA Enterprise real-time kernel [12], which allows achieving high predictable timing behavior with a very small run-time overhead and memory footprint. ERIKA Enterprise is an innovative OSEK/VDX RTOS for small microcontrollers that includes highly predictable real-time kernel mechanisms and uses innovative programming features to support time sensitive applications on a wide range of microcontrollers and multi-core platforms. In addition to the OSEK/VDX standard scheduling algorithm, ERIKA Enterprise implements other scheduling algorithms such as Fixed Priority with preemption thresholds, Stack Resource Policy (SRP) [4], Earliest Deadline First (EDF) [24], resource reservations (FRSH) [29] and hierarchical scheduling (HR) [6], [7] which can be used to schedule tasks with real-time requirements. In particular, ERIKA supports periodic and aperiodic task scheduling according to fixed and dynamic priorities; interrupt handling for urgent peripherals operation (interrupts always preempt task execution); and time bounded resource sharing through the Immediate Priority Ceiling protocol [38], [10].

In the embedded system domain, drivers are often implemented following a polling scheme, i.e., using busy waiting during the interaction with I/O peripherals. Polling-based drivers are typically simple to implement, but increase power consumption and generally allow a lower I/O throughput. On the other hand, interrupt-driven drivers are able to avoid busy waiting, at the price of a more complex implementation. In this case, during I/O operations, the CPU is available to execute unrelated computations or move to a low-power function state, leading to a reduced power consumption. Overall, the asynchronous behavior of interrupt-driven drivers impacts significantly on system predictability, leading to a greater complexity in schedulability analysis with respect to polling-based drivers. In order to squeeze-out the maximum performance from IMU sensors as advertised by InvenSense, without sacrificing low power consumption of the whole system, we had to carefully design and implement the software to handle the sensor. To this purpose, we developed a specific interrupt-driven driver for the InvenSense sensor tailored over the I/O interactions that are strictly required by the application domain. The driver is composed by two nested state-machines showed in Figure 2.

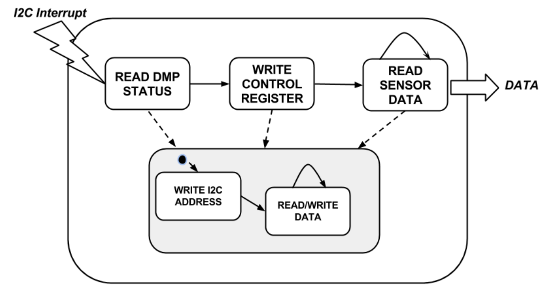

Figure 2: Architecture of the IMU driver.

The inner state-machine is responsible for managing the I^{2}C communication with the sensor. As explained in the previous Section, the InvenSense sensor is connected to the microcontroller through the I^{2}C bus: since the MCU allows only a common interrupt for I^{2}C events, it has been necessary a state-machine implementation of the interrupt handler. Data transfer over the I^{2}C bus requires an I/O interaction composed on different read/write phases. Such a state-machine is in charge of dealing with the communication phases of the I^{2}C bus, tacking track of the current communication phase. Also the InvenSense sensor depend upon I/O interactions based of communication phases. The outer state-machine is therefore in charge to handle the communication with the DMP: in such a way the sensor data are collected in a buffer and made available for the application.

D. Time synchronization

In a wireless network like the one proposed in this paper, the analysis of gathered data is performed by processing samples of measurements coming from different nodes of the network. In order to minimize the error, it is important to elaborate samples acquired at the same instant, especially if the nodes are used to monitor the dynamic of some physical variables, as in limb motion tracking. It is clear that the amplitude of the error increases with the temporal misalignment among the samples.

An example of the problem is illustrated in Figure 3, where two signals $f(t)$ and $f'(t)$ are sampled with a period T_c. Suppose that $f'(t)$ represents the angle of moving knee, and let ϵ be the clock drift rate of the node that samples this signal.

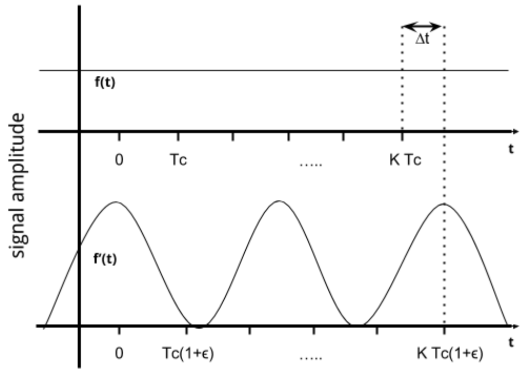

Figure 3: Effect of time synchronization errors.

After K samples, the clock skew Δt between the samples coming from the two nodes and having the same timestamp can be computed as:

$$\Delta t = |KT_c(1+\epsilon) - KT_c| = \epsilon KT_c \ . \tag{1}$$

Under the assumption that the movement in a small interval occurs at a fixed angular velocity ω, the angular error is

$$\Delta\theta = \omega\Delta t \ . \tag{2}$$

Increasing the sampling period T_c can greatly extend battery duration, but cannot be done without a proper time synchronization, otherwise the achievable precision drops. In addition to the advantages given by an accurate time for sampling data, a precise time base can also be used to schedule global tasks, like those used to trigger system reconfiguration or change the behavior of each node in the network.

Clock synchronization allows a precise time alignment of data acquired from nodes. The higher the protocol accuracy, the lower the sampling frequency required to obtain small errors while correlating two different samples. In this actual version, we used the TPSN protocol [13] implemented over the radio reservation not dedicated to BLE.

E. Energy management

Power management is becoming a crucial aspect in several fields of information technology, from mobile devices that need to prolong their lifetime without requiring huge batteries or long recharge time, to server farms where managing thermal issues is crucial to prevent failures and reduce costs. Almost all modern devices are based on the CMOS technology which is characterized by 2 major sources of energy consumption: dynamic power due to switching activities and static power due to the leakage current. For micrometer-scale semiconductor technology, the dynamic power dominates the power consumption of a processor. However, for technology in the deep sub-micron domain, the leakage power consumption is comparable to or even more significant than the dynamic power dissipation. Reducing the voltage decreases the dynamic power consumption, however it also reduces the maximum operating frequency. The overall energy consumption of a computing system also depends on other components.

Power management capabilities provided in modern devices do not take timing constraints into account, thus time guarantees must be performed by proper energy-aware scheduling algorithms implemented at an intermediate software layer. Two major classes of power-aware algorithms can be distinguished based on the kind of power they try to reduce. Dynamic Voltage Scaling (DVS) techniques reduce dynamic power by decreasing the supply voltage (and consequently the clock frequency) of the system, whereas Dynamic Peripheral Management (DPM) approaches achieve energy saving by exploiting operational states with reduced energy consumption (e.g., sleep or idle) whenever possible. When using DVS techniques in the presence of resource constraints (e.g., under mutually exclusive resources or non preemptive regions), the system can experience scheduling anomalies in which a task could even increase its response time when executed at a higher speed [9]. Such problems prevent managing the performance of a real-time application as a function of the processor speed. To cope with the variable requirements coming from different monitoring applications in a transparent way for the application developer, energy-related issues are managed through the abstraction layer proposed by Bambagini *et al.* [5], integrated inside Erika kernel. This library allows selecting a policy customized for a specific device, providing a uniform interface. In this way, it permits to achieve advantages given by energy management while taking timing constraints into account.

IV. Data fusion

Limb postures are reconstructed by integrating the data from different inertial measurement units that are positioned on the body. In order to cope with the noisy IMUs, our approach integrates kinematic constraints that limit the space of possible configuration that the node can assume, thus a better estimation of the final posture could be obtained.

In the following, a formal representation of the kinematic constraints is presented.

A. How to estimate node attitude

The attitude is the orientation of a coordinate system with respect to some reference system. Every node can produce both raw data acquired from all sensors and an attitude information. The IMU mounted in the proposed platform provides a quaternion, which denotes the rotation of the "Earth-fixed frame" with respect to a frame attached to the sensor denoted as "body frame". Compared to other techniques, like rotational matrix and Euler Angles, quaternions improve computational efficiency and avoid singularities problems (e.g., gimbal lock).

We now introduce some notation typically used by the general statement of attitude estimation problem. Given n bodies kinematically constrained as shown in Figure 4, we denote:

- $\{F\}$ as the inertial earth-fixed-frame reference,

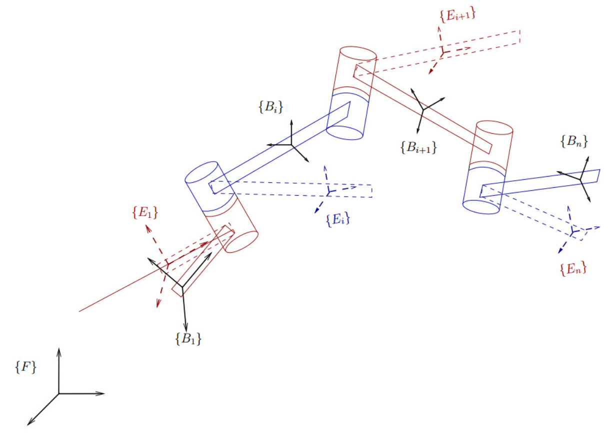

Figure 4: General Case.

- $\{B_i\}$ as the general sensor-fixed-frame reference (also know as body frame),
- $\{E_i\}$ as the estimate sensor-fixed-frame reference,

We assume that the frames $\{B_i\}$ and $\{B_{i+1}\}$ are kinematically constrained. To describe these constraints, we introduce quaternions notation ${}^{F}\mathbf{p}_{B_i}$. which expresses the rotation of the inertial frame F with respect to the body frame B_i of a sensor:

$$ {}^{F}\mathbf{p}_{B_i} = \begin{pmatrix} a & b & c & d \end{pmatrix}^T \tag{3}$$

where T denotes the vector transpose operator. Variables b, c, and d are the "vector part" of the quaternion, and can be thought of as a vector about which rotation should be performed. The element a is the "scalar part" that specifies the amount of rotation that should be performed about the vector part. The inverse of a quaternion $\mathbf{p}$ is computed as

$$\mathbf{p}^{-1} = \frac{1}{a^2+b^2+c^2+d^2}\begin{pmatrix} a & -b & -c & -d \end{pmatrix}^T \ . \tag{4}$$

The attitude quaternion ${}^{F}\mathbf{p}_{B_i}$ can be used to rotate an arbitrary 3-element vector from the inertial frame to the body frame. A vector $\mathbf{v}_I$ can be rotated by considering it as a quaternion with zero real-part and multiplying it by the attitude quaternion ${}^{F}\mathbf{p}_{B_i}$ and its inverse ${}^{F}\mathbf{p}_{B_i}^{-1}$

$$\mathbf{v}_B = {}^{F}\mathbf{p}_{B_i}\begin{pmatrix} 0 \\ \mathbf{v}_I \end{pmatrix}\left({}^{F}\mathbf{p}_{B_i}^{-1}\right) . \tag{5}$$

The frames $\{B_i\}$ and $\{B_{i+1}\}$ are kinematically constrained by the quaternion ${}^{B_{i+1}}\mathbf{p}_{B_i}(q_i)$ where $q_i \in \mathcal{Q}_i$ and $\mathcal{Q}_i \subseteq \mathbb{R}^{n_i}$ is the joint space, with n_i the number of joint variables. For easy readability we omit the dependence of the time from all the variables.

Given SO(3) the space all rotations about the origin of three-dimensional Euclidean space $\mathbb{R}^3$, we define also the following quaternions:

- $\mathbf{p}_i := {}^{F}\mathbf{p}_{B_i} \in \mathrm{SO}(3)$: relative orientation of $\{B_i\}$ respect to $\{F\}$,
- $\hat{\mathbf{p}}_i := {}^{F}\mathbf{p}_{E_i} \in \mathrm{SO}(3)$: (estimated attitude) relative orientation of $\{E_i\}$ with respect to $\{F\}$,
- $\bar{\mathbf{p}}_i := {}^{B_i}\mathbf{p}_{B_{i+1}}(q_i) \in \mathrm{SO}(3)\,|q_i \in \mathcal{Q}_i$: relative orientation of $\{B_{i+1}\}$ respect to $\{B_i\}$. It follows that $\bar{\mathbf{p}}_i = \mathbf{p}_{i+1}^{-1}\mathbf{p}_i$.

We would like to estimate the joint variables of the relative orientation of $\{B_{i+1}\}$ with respect to $\{B_i\}$, i.e. $\{q_i(t) \in \mathcal{Q}_i|\ \bar{\mathbf{p}}_i(q_i(t)) \in \mathrm{SO}(3)\}$. In theory, we could separately estimate the attitudes of the $\{B_i\}$ frames and then find the relative orientation $\bar{p}_i$. Instead, the noisy measurements coming from the IMUs degrades the performance of this straightforward approach.

Lets examine the simple case of a body-fixed frame constrained with inertial fixed-frame. Suppose that the inertial-fixed frame $\{F\}$ and the body-fixed frame $\{A\}$ are constrained each other through a revolute joint. Let $\{E\}$ be the estimate reference frame as depicted in Figure 5. Although this is a really simple case, it is a good starting point to model more complex real-life kinematic scenarios.

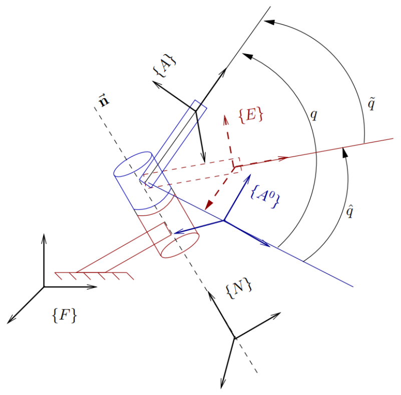

Figure 5: Body-Fixed Frame Constrained with Inertial-Fixed Frame.

We define $\mathbf{p} := {}^{F}\mathbf{p}_A = \mathbf{p}(q) \in \mathrm{SO}(3)$, where $q \in \mathbb{R}$ is the scalar variable to estimate.

The rotation $\hat{\mathbf{p}} := {}^{F}\mathbf{p}_E(\hat{q})$ is the estimated attitude (corresponding to an estimated variable $\hat{q} \in \mathbb{R}$). The orientation error is defined as $\tilde{\mathbf{p}} := {}^{E}\mathbf{p}_A = \hat{\mathbf{p}}^{-1}\mathbf{p} \in \mathrm{SO}(3)$. A perfect attitude estimation is achieved when $\tilde{\mathbf{p}} = I$.

V. Evaluation

In this section some preliminary experiments are presented to evaluate the effectiveness of the proposed approach. The first application test case we identified consists of a single joint. Despite its simplicity, it is already realistic because it models very well the case of knee rehabilitation, and requires a precision of at least 1 degree and up to 0.3 rad/s angular velocity. Two different types of measures were performed: accuracy measures for both orientation and position, and time delay measures.

A. Accuracy measures

The spatial accuracy of a single sensor node was evaluated with respect to a reference given by a Polhemus Patriot system [33]. This device is capable of measuring 6DOF (Degrees-Of-Freedom), thus obtaining the full position and orientation of a mobile probe with respect to a fixed reference. Its working principle is based on emitting a tuned electromagnetic field from the fixed reference (source) and measuring it with the probe (mobile sensor). This procedure avoids the need for hybrid data merging performed via software. The resulting resolution is about 0.03 mm and 0.01 degrees, and a static accuracy of 1.5 mm RMS for the X, Y, and Z position and 0.4 degrees RMS for the orientation. To perform the test, both the Polhemus and the IMU have been mounted on a common board in a way do avoid any measurement error caused by misalignment. Figure 6 shows the experimental setup used to evaluate the IMU accuracy measures.

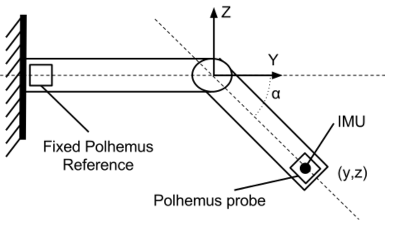

Figure 6: Experimental setup used to evaluate the accuracy.

1) Orientation: The orientation accuracy of a single sensor node was measured by converting quaternions to Euler angles and directly comparing them to the ones given by the Polhemus. Only the angles around X and Y axes were measured, because the measurement around the Z-axis relies on magnetometer which is strongly affected by the magnetic field generated by the Polhemus. Figure 7 shows the IMU measurement error with respect to the Polhemus reference. The majority of errors are around the value of 0.65 degrees; the Mean of the error is 0.8685 degrees and its RMS is 0.9871 degrees. For these reason the measure can be considered a good result because it is obtained with a system that is considerably cheaper than the Polhemus (i.e., few euros compared to several thousand of euros).

Figure 8 shows that the error remains quite small even during fast movements. During sudden changes of direction, the IMU reacts with a little delay with respect to the Polhemus, because it gets fooled by gyroscopes. In fact, the accelerometers are more noisy for faster movements, and the DMP data-fusion algorithm increases the weight to the gyroscopes. Despite such a behavior due to its inner working, the error of the IMU does not substantially vary from the average during these fast circumstances.

2) Position: Position accuracy was measured in a similar setup. While the Polhemus gives out both orientations and positions, the IMU sensor is only capable of angular measurements, thus Y and Z coordinates must be computed using simple trigonometric formulas as a function of the link dimension. Figure 9 shows the distribution of the measured position error on the Y-axis.

B. Processing time and communication delay

Finally, a specific test has been carried out to evaluate the computation time spent by the sensor node to process data

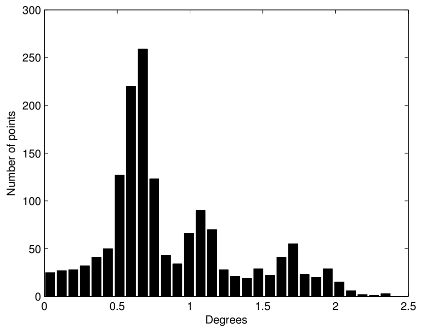

Figure 7: Histogram of α angle error distribution.

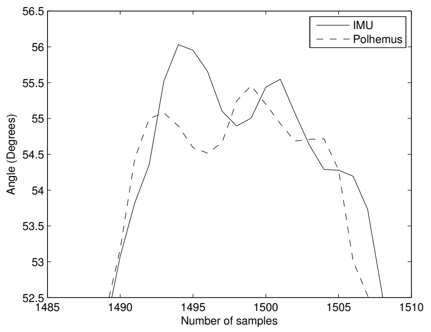

Figure 8: Plot of movements with $\omega = 0.3\ rad/s$

coming from the IMU, and the end-to-end delay of the data acquisition, up to the graphical representation performed on the pc/tablet user interface. Processing timing measurements were performed by switching a GPIO pin at the beginning and at the end of the computation, and capturing them by a multi-channel logic analyzer. We captured the timing of the following processing timing on the sensor node:

- DMP interrupts arrivals;
- duration of packet reading procedure through I^{2}C bus;
- duration of data processing.

To measure the communication delay between nodes and pc/tablet, instead, the simple switching of a GPIO is not practically feasible for at least two reasons. Firstly, on PC's and tablets there is no cheap and simple way to instantaneously change a GPIO pin state as on microcontrollers, considering that we are exclusively using standard programming environments and tools. Secondly, there is no access to the internals of the Bluetooth Low Energy (BLE) stack, neither on the pc with Windows 8.1, nor on the tablet with Android and not even on the nodes running the Nordic BLE library.

The adopted solution was to directly measure the time instant at which the application reacts upon receipt of a data packet. After the elaboration it activates a region of the tablet screen. Thus the measurement was carried out using a photo-resistor attached to the screen to detect the change of a small rectangular image, between black and white. This caused luminance changes, affecting the output of the photo-resistor, thus allowing the logic analyzer to capture it.

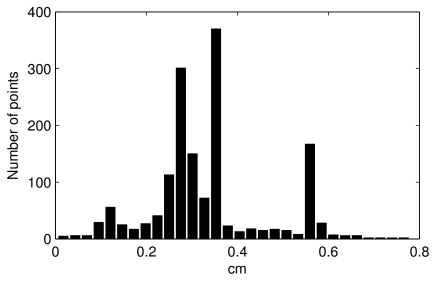

Figure 9: Histogram of position error distribution (Y-axis).

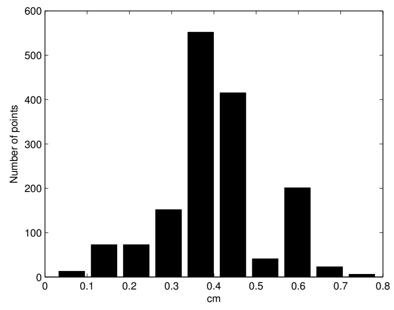

Figure 10: Histogram of norm error vector distribution.

With the above setup, we acquired 60 seconds of data, using a Nexus 7 tablet as a receiver. The sampling frequency configured on the node was 20 Hz, resulting in 1200 samples. Results of this experiment are reported in Table I, that contains execution time statistics for I^{2}Cread routines and the CPU time spent until the sending of the data over radio link, and Table II, that contains delay statistics of radio transmitting times and the final delay measured to shows data on the user interface.

Time(ms)	Mean	RMS	Std Dev.	Variance
$I2Cread$	2.409	2.416	0.1833	0.0335
$computation$	0.218	0.228	0.0660	0.0043

Table I: Execution times (ms).

Delay(ms)	Mean	RMS	Std Dev.	Variance
$BLEsend$	2.627	2.634	0.197	0.039
$display$	23.730	27.575	14.052	197.470

Table II: Delay times (ms).

Variance of display delay is quite high due to several reasons, the most relevant are:

- The delay that can occur during the transmission could suffer due to the BLE protocol (e.g., busy channel, re-transmission, etc.);
- the lack in the Android framework of a structured support for managing time constraints at application and communication layer;
- the presence of a complex video rendering subsystem that is optimized to improve user experience but it is not aimed at a predictable timing behavior in the data representation.

1) Performance differences between Windows and Android: Although BLE is not explicitly designed for transmitting a continuous data stream, but only for a sporadic data exchange, it can be used without problem for this purpose assuming the use of an efficient stack implementation that provides a suitable throughput, both on the master (pc or tablet) and the slaves (sensor nodes). Our first implementation of the master application was developed on Windows 8 and we were unable to achieve more than 24 packets/s. This bottleneck is due to the Windows BLE stack poor implementation, independently of the used BLE devices (i.e., two different USB dongles produced the same throughput). Actually, the only way to achieve better performance on Windows is by using a custom dongle, for example based on the same microcontroller like the one provided by Nordic in their development kit. Instead, the BLE stack included in Android allowed our application to achieve an higher throughput. Using this setup, we were able to reach 200 packets/s coming from two different nodes.

VI. Conclusions and Future work

This paper presented a new set of wearable nodes that integrate a selection of small, low-cost, energy-efficient components and wireless communication to achieve a good balance between cost and performance. The used of real-time methodologies allowed to squeeze-out the maximum performance from ultra-low power hardware without sacrificing application requirements, acquisition rates, and full compatibility with new PC's, smartphones, and tablets. The developed platform allowed to obtain a highly wearable and inexpensive solution characterized by a sufficient precision and good time predictability.

Preliminary experiments with the platform proved the quality of the approach and showed the precision of the measurements. Next steps in the development will be:

- a set of more extensive experiments to evaluate the accuracy of the system and the delay experienced with respect to a more complex case than knee rehabilitation;
- a complete integration of inertial data with kinematic constraints to reduce further measurements errors and simplify the calibration phase;
- a more precise data synchronization among units, achieved through an efficient time synchronization protocol and the use of advanced real-time kernel mechanisms;
- the further reduction of energy consumption, with an integrated power management approach applied at different levels of the architecture. For example, a better synchronization protocol could reduce the sampling frequency and therefore the energy consumption.

VII. Acknowledgements

The authors thank Alessandro Biondi and Daniel Cesarini for the constructive discussions during this work.

References

[1] D. K. Arvind and A. Valtazanos. Speckled tango dancers: Real-time motion capture of two-body interactions using on-body wireless sensor networks. In *Proc. of the 2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*, Washington, DC, USA, 2009. IEEE Computer Society.

[2] R. Aylward, S. Lovell, and J. Paradiso. A compact, wireless, wearable sensor network for interactive dance ensembles. In *Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. International Workshop on*, April 2006.

[3] E. R. Bachman, R. B. McGhee, X. Yun, and M. J. Zyda. Inertial and magnetic posture tracking for inserting humans into networked virtual environments. In *In ACM Symposium on Virtual Reality Software and Technology (VRST*. ACM, 2001.

[4] T. P. Baker. Stack-based scheduling for realtime processes. *Real-Time Systems*, 3(1):67–99, April 1991.

[5] M. Bambagini, M. Marinoni, F. Prosperi, and G. Buttazzo. Energy management for tiny real-time kernels. In *Proceedings of the 2nd Internationa Conference on on Energy Aware Computing (icEAC 2011)*, Istanbul, Turkey, November 2011.

[6] M. Bertogna, N. Fisher, and S. Baruah. Resource-sharing servers for open environments. *IEEE Transactions on Industrial Informatics*, 5(3):202–220, August 1991.

[7] A. Biondi, G. Buttazzo, and M. Bertogna. Schedulability analysis of hierarchical real-time systems under shared resources. Technical report, RETIS Lab, Scuola Superiore SantAnna, 06 2013.

[8] A. Braidot, C. Cifuentes, A. Frizera Neto, M. Frisoli, and A. Santiago. Zigbee wearable sensor development for upper limb robotics rehabilitation. *Latin America Transactions, IEEE*, Feb 2013.

[9] G. Buttazzo. Achieving scalability in real-time systems. *Computer*, 39(5):54–59, May 2006.

[10] G. Buttazzo. *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications, Third Edition*. Springer, New York, 2011.

[11] E. Foxlin. Inertial head-tracker sensor fusion by a complementary separate-bias kalman filter. *Proc. Virtual Reality Annual International Symposium the IEEE*, Mar 1996.

[12] P. Gai, G. Lipari, L. Abeni, M. di Natale, and E. Bini. Architecture for a portable open source real-time kernel environment. In *Proceedings of the Second Real-Time Linux Workshop and Hand's on Real-Time Linux Tutorial*, November 2000.

[13] S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *Proc. of the 1st International Conference on Embedded Networked Sensor Systems*, New York, NY, USA, 2003. ACM.

[14] Google. Android operating system, 2014.

[15] T. Harada and T. M. T. Sato. Human posture probability density estimation based on actual motion measurement and eigenpostures. *IEEE International Conference on Systems, Man and Cybernetics*, Oct, 2004.

[16] R. A. Hyde, L. P. Ketteringham, S. A. Neild, and R. J. S. Jones. Estimation of upper-limb orientation based on accelerometer and gyroscope measurements. *IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING*, Feb 2008.

[17] InterSense Inc. *InertiaCube2+ Manual*, 2008.

[18] InvenSense. *MPU-9150 Product Specification Revision 4.0*, May 2012.

[19] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D), 1960.

[20] J. Kim, S. Yang, and M. Gerla. Stroketrack: Wireless inertial motion tracking of human arms for stroke telerehabilitation. In *Proc. of the First ACM Workshop on Mobile Systems, Applications, and Services for Healthcare*, New York, NY, USA, 2011. ACM.

[21] N. P. Konstantin Mikhaylov and J. Tervonen. Performance analysis and comparison of bluetooth low energy with ieee 802.15.4 and simpliciti. *Journal of Sensor and Actuator Networks*, August 2013.

[22] J. K. Lee and E. J. Park. A fast quaternion-based orientation optimizer via virtual rotation for human motion tracking. *IEEE Trans. Biomed. Eng., vol. 56*, Jul 2009.

[23] J. Lee and I. Ha. Sensor fusion and calibration for motion captures using accelerometers. *Proc. of the 1999 IEEE International Conference on Robotics and Automation, Detroit, Michigan*, Feb 1999.

[24] C. Liu and J. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the Association for Computing Machinery*, 20(1):46–61, January 1973.

[25] H. J. Luinge, P. H. Veltink, and C. T. M. Baten. Estimation of orientation with gyroscopes and accelerometers. *Proc. First Joint [Engineering in Medicine and Biology 21st Annual Conf. and the 1999 Annual Fall Meeting of the Biomedical Engineering Soc.]*, Oct 1999.

[26] S. O. Madgwick, A. J. Harrison, and R. Vaidyanathan. Estimation of imu and marg orientation using a gradient descent algorithm. *IEEE International Conference on Rehabilitation Robotics*, July 2011.

[27] R. Mahony, T. Hamel, and J.-M. Pflimlin. Nonlinear complementary filters on the special orthogonal group. *IEEE TRANSACTIONS ON AUTOMATIC CONTROL*, June 2008.

[28] J. L. Marins, X. Yun, E. R. Bachmann, R. B. McGhee, and M. J. Zyda. An extended kalman filter for quaternion-based orientation estimation using marg sensors. *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2001.

[29] L. Marzario, G. Lipari, P. Balbastre, and A. Crespo. IRIS: A new reclaiming algorithm for server-based real-time systems. In *Proc. of the IEEE Real-Time and Embedded Technology and Applications Symposium*, Toronto, Canada, May 2004.

[30] A. Menache. *Understanding Motion Capture for Computer Animation and Video Games*, 1999.

[31] Microsoft Corporation. *Kinect for Xbox 360*, Nov 2010.

[32] NORDIC Semiconductor. *nRF51822 Product Specification v1.3*, May 2013.

[33] Polhemus. *Polhemus Patriot Product Specification*, Feb 2010.

[34] D. Rodrguez-martn, C. Prez-lpez, A. Sam, J. Cabestany, and A. Catal. A wearable inertial measurement unit for long-term monitoring in the dependency care area, 2013.

[35] Roetenberg, D. L. Henk, and P. Slycke. Xsens mvn: full 6dof human motion tracking using miniature inertial sensors. *Xsens Motion Technologies BV, Tech. Rep*, 2009.

[36] A. M. Sabatini. Quaternion-based extended kalman filter for determining orientation by inertial and magnetic sensing. *IEEE Trans. Biomed. Eng., vol. 53*, Jul 2006.

[37] A. M. Sabatini, C. Martelloni, S. Scapellato, and F. Cavallo. Assessment of walking features from foot inertial sensing. *IEEE Transaction on Biomedical Engineering*, Mar 2005.

[38] L. Sha, R. Rajkumar, and J. Lehoczky. Priority inheritance protocols: An approach to real-time synchronization. *IEEE Transactions on Computers*, 39(9):1175–1185, September 1990.

[39] M. Solazzi, A. Frisoli, and M. Bergamasco. Design of a novel finger haptic interface for contact and orientation display. *IEEE Haptics Symposium*, Mar 2010.

[40] B. Tessendorf, F. Gravenhorst, B. Arnrich, and G. Troster. An imu-based sensor network to continuously monitor rowing technique on the water. In *ISSNIP, 2011 Seventh International Conference on*, Dec 2011.

[41] Y.-L. Tsai, T.-T. Tu, H. Bae, and P. H. Chou. Ecoimu: A dual triaxial-accelerometer inertial measurement unit for wearable applications. *International Conference on Body Sensor Networks (BSN)*, Jun 2010.

[42] Vicon Motion Systems Limited. *Vicon MX Hardware*, Feb 2004.

[43] G. Welch and G. Bishop. *An Introduction to the Kalman Filter*, 1995.

[44] Xsens Technologies B.V. *MTi and MTx User Manual and Technical Documentation.*, May 2009.

[45] A. D. Young, M. J. Ling, and D. K. Arvind. Orient-2: A realtime wireless posture tracking system using local orientation estimation. In *Proc. of the 4th Workshop on Embedded Networked Sensors*, New York, NY, USA, 2007. ACM.

[46] X. Yun and E. R. Bachmann. Design, implementation, and experimental results of a quaternion-based kalman filter for human body motion tracking. *IEEE TRANSACTIONS ON ROBOTICS*, Dec 2006.

[47] T. Zhang, M. Karg, J.-S. Lin, D. Kulic, and G. Venture. Imu based single stride identification of humans. In *RO-MAN, 2013 IEEE*, Aug 2013.

[48] J. Zizka, A. Olwal, and R. Raskar. Specklesense: Fast, precise, low-cost and compact motion sensing using laser speckle. In *Proc. of the 24th Annual ACM Symposium on User Interface Software and Technology*, New York, NY, USA, 2011. ACM.