digitalemedien

Bachelor Report

Tweaking Keyframe Animation with an Accelerometer Interface

A Mobile Low-cost Motion Capture System based on Accelerometers

Zeitraum:
14.03.2006 - 14.07.2006

Betreuer:
Prof. Dr. Jörn Loviscach (Hochschule Bremen)

Gutachter:
Prof. Dr. Jörn Loviscach (Hochschule Bremen)
Prof. Dr. Frieder Nake (Universität Bremen)

Jan-Phillip Tiesel
Matrikel-Nr. 1674708
Universität Bremen - Fachbereich 3
Studiengang Digitale Medien
Sommersemester 2006

Table of Contents

Tweaking Keyframe Animation with an Accelerometer Interface

A Mobile Low-cost Motion Capture System Based On Accelerometers

Jan-Phillip Tiesel

University of Bremen

Abstract. Diese Arbeit zeigt, wie kostengünstige Beschleunigungsmesser zum Aufbau eines Motion Capture Systems verwendet werden können, welches vollständig in Anwendungssoftware zur 3D-Animation integriert ist. Die Sensoren, auch Accelerometer genannt, werden vom Schauspieler an den Körpergliedmaßen getragen. Eine stabile Methode zum Extrahieren von kinematischen Daten und Zeitinformationen über die aufgenommene Bewegung aus den Sensordaten wird vorgestellt. Da die reinen Beschleunigungsdaten keine vollständige und fehlerfreie Rekonstruktion der ursprünglichen Bewegung zulassen, werden Wege aufgezeigt, um diese Beschränkungen zu kompensieren. Anwendungsbeispiele, Ergebnisse und einige mögliche Ansätze für zukünftige Forschungsarbeit werden präsentiert.

Abstract. This paper presents an approach to use low-cost accelerometers as a means of a motion capture system which is fully integrated into standard animation software. A robust method to extract kinematic and motion timing information from the sensors, which are attached to the actor's limbs, is introduced. As acceleration data only cannot provide complete and error-free information in order to satisfactorily reconstruct the captured motion, ways are shown to compensate for those restrictions. Application scenarios, results, and some possible directions for future work are displayed.

1 Introduction

1.1 Motivation

The need for synthesized realistic human motion is steadily increasing. With research and application areas extending from character animation for the entertainment industry to kinesiology or medical monitoring, this demand is unlikely to recede in the near future. During the past two decades, motion capture technology for computer articulated characters evolved into a serious alternative to traditional keyframe animation. While creating believable and lifelike motion by keyframing poses and breakdowns requires highly skilled animators and generally takes a long time, motion capture technology is theoretically able to provide

the wanted data and reproduce the motion immediately after the real-time capturing process. An overview of the research in vision based motion capturing is given by Moeslund and Granum [1].

Most modern motion capture systems used for professional applications work with multiple stationary cameras and reflective markers attached to the actor (e.g. [2]). Although improvements in technology and algorithms have reduced the amount of time and work needed to transfer the captured motion into a digital representation that resembles the desired motion well enough, there are still some disadvantages to this way of data collection. Occlusion of the markers can occur and may create the need to manually fix the motion data after recording. As the cameras can maintain a certain degree of detail only within a limited range and can usually not be moved while capturing, their measurement volume might be too narrow for capturing the desired motion (e.g. ice skating). Above all, the accuracy of the recorded motion is limited by the pixel resolution of the used input device.

Most systems are still very expensive, must be operated by trained professionals, and usually take a comparatively long time to set up and calibrate. The typical price for a commercially available, vision-based, motion capture system does not fall below tens of thousands of USD. Generally speaking, sophisticated vision-based motion capture is not available for application scenarios with a low budget or with the demand for low complexity of setup and operation.

Several approaches were presented to reduce the complexity of the capture system by decreasing the number of cameras or the setup effort needed. Widgets can be employed instead of reflective markers in order to abbreviate the setup time [3]. Fewer cameras or alternative low-dimensional controllers have been successfully used to synthesize new animations by the concatenating and blending of existing motion capture data [4, 5]. Ren et al. [6] and de Aguiar et al. [7] showed how human silhouette features acquired by multiple cameras can be used to extract motion patterns. While most of the stated methods succeed in creating very realistic motion representations, they lack the ability to operate low-cost solutions with unlimited measurement volume. The decreased complexity in the hardware setup often leads to an increase in the complexity of system operation (e.g. the need for building an extensive motion capture database before the capturing process). The overall price of the used hardware could be lowered but is still in the range of several hundred to several thousand USD.

Mechanical motion capture offers a promising alternative to the approaches stated above as it employs sensors attached to the actor's body to estimate joint angles. Those systems are free-of-occlusion and often utilize wireless data transfer to permit mobile applications. While Measurand Inc. [8] uses a flexible tape that bends in accordance with the actor's limb movements, inertial sensors are used by Animazoo [9] to capture real-time motion. Both manufacturers state that the setup and calibration time as well as the essential user training are minimal compared to vision based solutions. Up to date those systems are not available for less than USD 20,000 and USD 60,000 respectively.

1.2 Approach

Inertial sensors in the form of micromachined accelerometers and miniaturized gyroscopes are widely available today. While gyroscopes measure angular velocity, acceleration sensors detect the linear acceleration of an object along a fixed axis. Modern accelerometers are light, inexpensive, and provide reasonable sensing accuracy. Although tri-axes gyroscopes can be easily employed to estimate the position change of the sensor by integration of the output signal which represents the angular velocity about three axes, units providing fairly accurate and drift-free measurements are not available for less than USD 100. In order to fathom the capabilities of an ultra low-cost approach, it was decided to exclusively use accelerometers for measuring purposes.

Acceleration sensors traditionally consist of a mass which is connected to the housing of the sensor by a damped spring. The mass can only move in one fixed direction. This is the sensitive axis of the sensing unit as acceleration along this axis will cause a change in the extension of the spring and therefore the electric output signal of the transducer. Contemporary accelerometers employ more sophisticated methods of determining the linear acceleration. The sensing units used for this research utilize suspended silicon structures which are free to move on a plane. They are attached to a substrate parallel to this plane in a few points called anchors. The mass displacement causes an imbalance in the sense capacitor which is then used to calculate the output signal [10]. The sensed linear acceleration is the sum of dynamic manual acceleration and the constant gravitational acceleration. Tri-axis accelerometer can be assembled by using three singular axis sensing units and by arranging them perpendicular to each other.

Accelerometers are also used for a broad variety of commercial applications. A few of these include: airplane navigation, automatic car headlight alignment, video projector calibration, hard drive protection in Apple^{TM}computers against damage due to shock, video game controllers and man-down alarm systems. Several interactive applications like the gesture-based control of DVD players [11] or PDA computers [12] have been presented in the past.

This paper examines different approaches to how sensor data of autonomous accelerometers can be interpreted for character animation and then utilized and edited in a standard animation software package. Instead of tracking visual features of the human body, the orientation of limbs is estimated using acceleration sensors that can be attached to an actor's clothes. By using the fixed direction of gravitational acceleration, profound assumptions on the posture of body segments can be made.

In addition, the magnitude of an object's dynamic acceleration is used to retime existing keyframe animation. This method can be employed as an extension to the orientation estimation described in Section 4. As it is assumed that the manual acceleration during the whole duration of capturing is negligibly low, unpredictable errors will occur during motions that imply high dynamic acceleration (e.g. running). The acting of the motion has to be executed twice if fast movements are desired. First, the kinematic information is established by limb

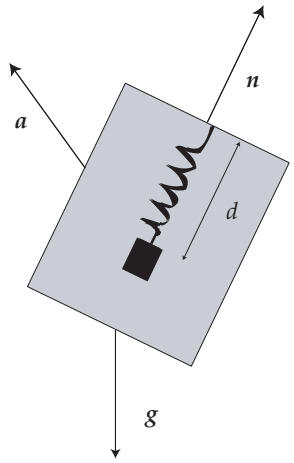

Fig. 1. Schematic view of a single axis accelerometer providing a simple way of measuring linear acceleration. The measured distance d is the amplitude of the spring depending on the manual acceleration $\boldsymbol{a}$ and the acceleration due to gravity $\boldsymbol{g}$. The sensitive axis is depicted by $\boldsymbol{n}$

orientation estimation without using the proper timing. Therefore, all motions are acted out in a slow-motion manner. The second step includes the recording of the acceleration magnitude information during a second acting pass where the motion is performed in real-time and without paying attention to low dynamic acceleration. The animation retiming procedure explained in Section 5.3 is then applied to the previously recorded kinematic data in order to enforce the proper timing.

The interpretation of the accelerometer data will most likely contain errors and will not yield an accurate and complete description of the initial motion. Several ways are therefore investigated to overcome these restrictions by giving the user tools to refine the recorded data. Motion warping is employed as an easy and powerful means to modify the captured motion sequence by adding displacement curves to the original signal.

The focus of this paper is on practicable solutions that apply low-cost hardware and that require minimum preparation and user training. The proposed algorithms were designed in order to use several inexpensive accelerometer devices for rapid prototyping and editing of realistic character animation. The system presented is fully integrated into Autodesk Maya$^{\mathrm{TM}}$and could be easily adapted to other 3D animation software.

1.3 Contribution of the thesis

The contribution of this paper is to present a layout for a mobile ultra low-cost motion capturing system which is entirely integrated into standard animation software. A method for signal range and orientation calibration of the utilized

acceleration sensors is given. A robust interpretation of long-term input signals free of drifting errors is introduced to estimate the 3D orientation of the sensing unit. Ways to synthesize the captured motion in a standard animation package are shown. Lastly, methods for intuitive editing of the resulting motion data are presented to compensate for the information that could not be retrieved by the sensors. The overall capturing process can be broken down into six steps:

1. Accelerometer signal range calibration *(only once per sensor)*
2. Global rotation matrix calibration *(once per capturing session)*
3. Real-time capturing of estimated limb orientations *(only slow motion movements to minimize errors due to manual acceleration)*
4. Smoothing of capture data and establishing the kinematic chain representation in the animation software
5. Re-enact the previous motion with proper timing to retime the kinematic data *(only if fast movements occurred during step 3)*
6. Refinement of the captured data using motion warping

This thesis is divided into three main sections: first an overview of related work is given to compare the approach to past research and point out the contribution of this work. In the second section, a detailed summary of the implemented system is given; finally, some results are presented and ideas for future work are highlighted.

2 Related work

2.1 Inertial sensing with accelerometers

Inertial sensing has been used to estimate the attitude and position of rigid bodies relative to an absolute reference frame. Aminian and Najafi [13] give an overview of available body-fixed sensors and discuss their potential for motion capture applications. Bachmann et al. [14], Zhu and Zhou [15] used MARG sensors (measuring **m**agnetic field, **a**ngular **r**ate and **g**ravity) to determine rigid body orientation in order to facilitate real-time insertion of human characters into virtual environments.

An extensive amount of research was done using inertial sensors for medical applications like gait analysis [16], observation of physical activity [17, 18], or stroke rehabilitation monitoring [19]. Mayagoitia et al. [20] presented a combination of accelerometer and gyroscope measurements to obtain kinematic parameters such as knee angle or thigh linear acceleration. Only the sagittal plane of the examined limbs was taken into account for calculation. Luinge and Veltink [21] investigated the maximum amount of information that can be extracted from a tri-axis accelerometer without intending to reconstruct the full 3D orientation of the sensor. Instead, it was shown how a long-term estimation of the sensor's *inclination* can be extracted from its output data using the angle between the sensor's z-axis and the negative gravity vector.

Hansson et al. [22] pointed out two of the major limitations of orientation estimation by the exclusive use of accelerometers. First, rotation about the line

of gravity cannot be detected; second, the ability to measure rotations about the object's axes depends on its overall orientation. Kemp et al. [23] deployed a combination of accelerometers and earth-magnetic field sensors enabling them to estimate horizontal orientation as well. The work of Giansanti et al. [24] indicated that the reconstruction of both orientation *and* position from only accelerometer data does not yield feasible results.

Complex state estimation models such as Kalman filtering [25] were employed to minimize measurement errors due to noise, drifting or offset displacement. Rehbinder and Hu [26] proposed fusing data from rate gyros and accelerometers to give long-term, drift-free attitude estimates. A combination of computer vision motion tracking and inertial tracking using the Kalman filter was presented by Welch [27].

2.2 Animation timing

While state-of-the-art 3D animation software provides the user with instant feedback and intuitive editing of the character's appearance and poses, one of the main challenges that animators face is getting the timing of the overall motion to be as lifelike and believable as possible. According to Lasseter [28], the traditional animation principle of timing is very important to modern day 3D computer animation as well:

> *"Timing, or the speed of an action, is an important principle because it gives meaning to movement – the speed of an action defines how well the idea behind the action will read to an audience. It reflects the weight and size of an object, and can even carry emotional meaning."*

Most people even without a background in character animation of any kind would be able to act out the specific timing of a human motion. Due to the lack of interfaces, it is still a very tedious work to convey those motions to a proper digital representation without the use of costly motion capture systems. Terra and Metoyer [29] presented a promising approach by allowing the user to draw a sketch of the desired motion using the proper timing. They then used the timing information of the user's sketch to retime an existing keyframed animation. Thorne et al. [30] implemented an animation system based entirely on sketching out the favored motion and using the kinematic and temporal information to reproduce the completed motion sequence.

2.3 Motion curve editing

Several techniques for editing of complex motion data have been discussed. Bruderlin and Williams [31] applied techniques from the signal processing domain to motion data editing which resulted in specific algorithms like *dynamic timewarping* or *motion displacement mapping*. The same simple but powerful technique of refining motion curves by functional composition was introduced as *motion warping* by Witkin and Popovic [32].

3 Hardware setup

The sensor hardware (see Figure 2) consists of a battery-powered central circuit board wired to five boxes, which can be fixed on the body through hook-and-loop tape. Each box essentially contains only an accelerometer chip LIS3L02AQ from STMicroelectronics^{TM}. This micromechanical device measures acceleration along three perpendicular axes and outputs three corresponding voltages. The sensors were affixed with glue inside the case with their sensitive axes aligned to the cases' edges. The central circuit board contains a clock generator (NE 555), a

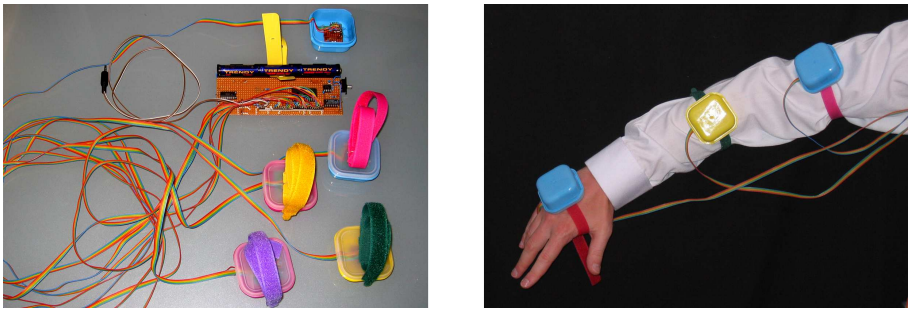

Fig. 2. The hardware interface consisting of five tri-axis accelerometers and the main board with line-out connector

counter (MOS 4024), and an analog 16:1 multiplexer (MOS 4067). The 15 output voltages of the accelerometers are connected to the inputs of the multiplexer. A reference voltage, stabilized by an operational amplifier (1/4 of RC 4136), is connected to the 16^{th} input. The timer and the counter are used to continuously cycle the multiplexer's output through the 16 inputs, so that each appears 60 times per second.

The output is decoupled through another operational amplifier (1/4 of RC 4136) and fed into the left audio input channel of the PC's sound card. The most significant (i.e., most slowly varying) bit of the counter is fed into the right channel. On the PC, our software synchronizes with this 60 Hz signal and records the left channel. Per cycle, it forms 16 time slots over which the signal is averaged to suppress the transients (see Figure 3). Standard PC sound cards do not reproduce DC voltages. To recover them, we form differences with the measurement for the reference voltage, which forms the 16^{th} input.

(indented text provided by Prof. Dr. Jörn Loviscach)

As an alternative, the 2-channel output signal could be digitized and recorded by a mobile MP3 player which has recording capabilities for uncompressed wave

signals. The gathered data can be transferred to a computer after the capturing process is completed. The overall cost of the setup was substantially less than USD 100. The assembled hardware parts and the code to retrieve the raw sensor data from the generated output signal were provided by Prof. Dr. Jörn Loviscach (thesis supervisor).

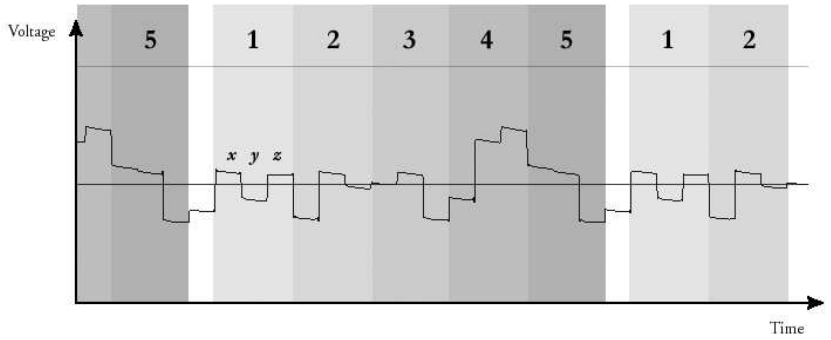

Fig. 3. Data retrieval from the digitized output signal: 15 values hold the accelerometer output of the five sensors, one value is used for offset correction

The software for analyzing and initial processing of the input data was written in Microsoft Visual C# 2005 Express Edition$^{\mathrm{TM}}$. Only standard math and multimedia libraries (like Microsoft DirectX$^{\mathrm{TM}}$) were utilized. The processed data was saved to a file that also held general information about the capture process (sample rate, number of active sensors, sensor naming). A library of procedures was written in MEL, the scripting language of Autodesk Maya$^{\mathrm{TM}}$, to provide the user with a graphical interface that is fully integrated into the animation package. It allows the user to import, edit and utilize the captured data. Autodesk Maya$^{\mathrm{TM}}$was chosen because of its current prevalence in the character animation industry and its expandability.

4 Processing the captured input data

4.1 Accelerometer signal range calibration

The output signal $\boldsymbol{B}_{\mathrm{sensor}}$ of a tri-axis accelerometer consisting of the value triple v_x, v_y and v_z can be expressed as

$$\boldsymbol{B}_{\mathrm{sensor}} = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} = \boldsymbol{a} + \boldsymbol{g}_{proj} + \boldsymbol{\sigma} \ , \tag{1}$$

where $\boldsymbol{a}$ denotes the dynamic manual acceleration of the sensing unit, $\boldsymbol{g}_{proj}$ is the acceleration due to gravity projected on the sensor's axes and $\boldsymbol{\sigma}$ is an

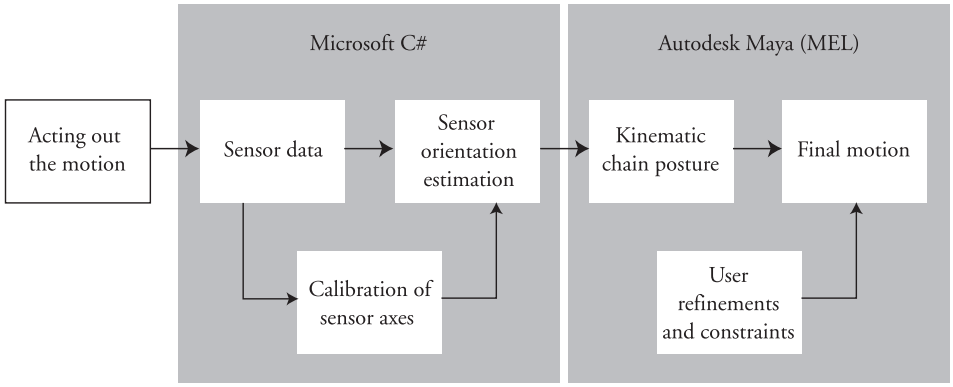

Fig. 4. Simplified chart depicting the flow of data and information from initial physical motion to its digital representation

unidentified amount of signal noise. The magnitude and absolute direction of the gravity vector in the world reference frame is assumed to be the constant vector $[0 \;\; 0 \;\; -g]^T$ with $g \approx 9.81 m/s^2$. For better legibility, a normalized version of $\boldsymbol{B}_{\text{sensor}}$ is calculated:

$$\boldsymbol{B}_{\text{sensorNorm}} = \frac{1}{g} \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} \; . \tag{2}$$

It is assumed that during periods of little motion, the magnitude of $\boldsymbol{a}$ is so small that $\boldsymbol{a}$ can be neglected. In that case, optimal conditions should result in $|\boldsymbol{B}_{\text{sensorNorm}}| = 1$. During arbitrary rotations of the sensor about any of its three axes it is therefore necessary that

$$-1 \leq v_x, v_y, v_z \leq 1 \;\; \forall \;\; v_x, v_y, v_z \; . \tag{3}$$

Experiments showed that the minimum and maximum output voltage of many sensors either exceed or undershoot those limits. This can be caused by misalignment of the axes, manufacturing tolerances, or signal errors. In order to compensate for this problem, a calibration procedure is introduced. During a user-defined calibration period, the lowest and highest emerging values of v_x, v_y and v_z are established while the user rotates the sensor about all of its axes. In order to compensate for unwanted peaks in the sampled data due to increased manual acceleration, a second calibration loop with length t_{max} (about 0.5 seconds in our examples) is introduced and evaluated repetitively. Let t be the current time and initial variable values $\boldsymbol{B}_{\text{minTemp}} = [\infty, \infty, \infty]^T$ and $\boldsymbol{B}_{\text{maxTemp}} = [-\infty, -\infty, -\infty]^T$.

$$\begin{aligned} &\textit{while } (t < t_{max}) \; \{ \\ &\quad \boldsymbol{B}_{\text{maxTemp}} = \max\left(\boldsymbol{B}_{\text{maxTemp}}, \boldsymbol{B}_{\text{Norm}}\right) \\ &\quad \boldsymbol{B}_{\text{minTemp}} = \min\left(\boldsymbol{B}_{\text{minTemp}}, \boldsymbol{B}_{\text{Norm}}\right) \end{aligned}$$

$$
\begin{aligned}
&\}\\
&\boldsymbol{B}_{\max} = \max\left(\boldsymbol{B}_{\min\mathrm{Temp}}, \boldsymbol{B}_{\max}\right)\\
&\boldsymbol{B}_{\min} = \min\left(\boldsymbol{B}_{\max\mathrm{Temp}}, \boldsymbol{B}_{\min}\right)
\end{aligned}
$$

After each loop iteration, $\boldsymbol{B}_{\min}$ and $\boldsymbol{B}_{\max}$ hold the new minimum and maximum values that are used to scale the input vector $\boldsymbol{B}_{\mathrm{sensorNorm}}$ to the correct range after calibration:

$$
\boldsymbol{B}_{\mathrm{sensorFinal}} = 2 \cdot \begin{pmatrix} \frac{v_{\mathrm{Norm}_x} - v_{\min_x}}{v_{\max_x} - v_{\min_x}} \\ \frac{v_{\mathrm{Norm}_y} - v_{\min_y}}{v_{\max_y} - v_{\min_y}} \\ \frac{v_{\mathrm{Norm}_z} - v_{\min_z}}{v_{\max_z} - v_{\min_z}} \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} . \tag{4}
$$

This calibration method has to be executed only once per accelerometer unless altered conditions cause drastic changes in the sensor's measuring behavior (e.g. temperature change or low battery voltage).

4.2 Calibration of the global rotation matrices

The next step requires attaching the sensors to the body segments whose orientations need to be captured (a typical configuration would be upper arm, forearm, and palm). It cannot be assumed that the sensitive axes of all utilized sensors are aligned to each other. Therefore the different initial orientations are estimated by a brief calibration procedure that establishes a rotation matrix which transforms the sensors' axes to the limb axes (*sensor* to *limb* reference frame, see Figure 5). Let the axis pointing along the joint the sensor is attached to be the local x-axis in a right-handed Cartesian coordinate system. In the case stated above, the user extends his arm horizontally and vertically one after the other. $\boldsymbol{a}$ and $\boldsymbol{b}$ are the two input vectors (measured in the *sensor* reference frame) at the horizontal and vertical calibration states. In most cases, the two measured vectors will not be exactly perpendicular to each other. As we need to have three perpendicular vectors to obtain the rotation matrix R, $\boldsymbol{a}$ and $\boldsymbol{b}$ are rotated about the rotation axis $\boldsymbol{a} \times \boldsymbol{b}$ in a way that the resulting normalized vectors $\boldsymbol{a}_{\mathrm{Norm}}$ and $\boldsymbol{b}_{\mathrm{Norm}}$ enclose an angle of 90 degrees (see Figure 6). If the demanded rotation matrix is denoted as R, it has to transform the negative direction of the local z-axis to the first and the local x-axis to the second input vector:

$$
R \cdot \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} = \boldsymbol{a}_{\mathrm{Norm}} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} , \tag{5}
$$

$$
R \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \boldsymbol{b}_{\mathrm{Norm}} = \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} . \tag{6}
$$

If the third unit vector $\boldsymbol{c_{Norm}}$ is found using the cross product

$$
\boldsymbol{c}_{\mathrm{Norm}} = \boldsymbol{a}_{\mathrm{Norm}} \times \boldsymbol{b}_{\mathrm{Norm}} , \tag{7}
$$

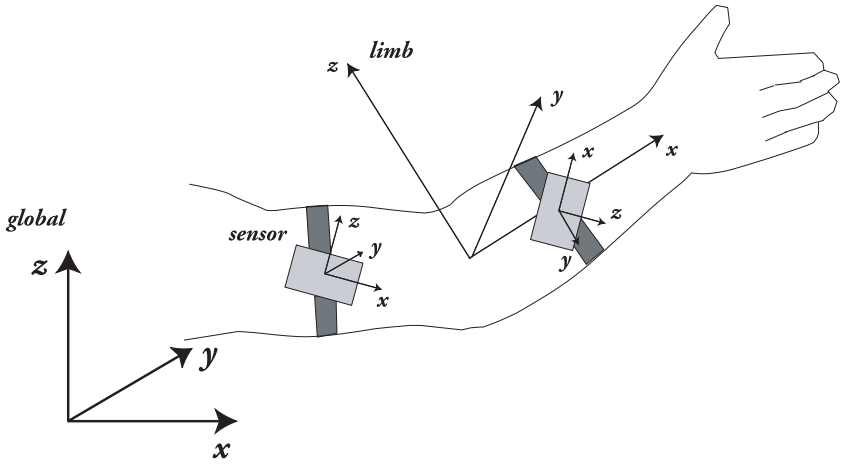

Fig. 5. Schematic view of the three reference frames used in the calculations: *sensor*, *limb*, and *global*

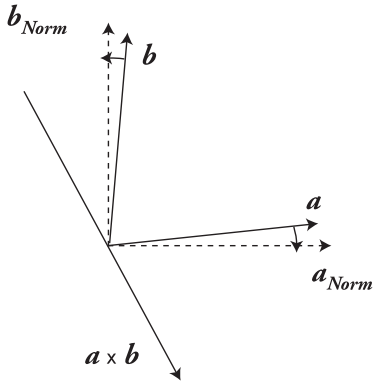

Fig. 6. The measured vectors are rotated to establish perpendicular axes for the sensor's global rotation matrix

it must hold true that $\boldsymbol{c}_{\text{Norm}}$ is the transformed sensor output expected to be measured while the sensor's y-axis points upwards:

$$R \cdot \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} = \boldsymbol{c}_{\text{Norm}} = \begin{pmatrix} c_x \\ c_y \\ c_z \end{pmatrix} . \tag{8}$$

Combining (5), (6), and (8) yields

$$R = \begin{pmatrix} b_x & -c_x & -a_x \\ b_y & -c_y & -a_y \\ b_z & -c_z & -a_z \end{pmatrix} . \tag{9}$$

As we want to transfer the input vector $\boldsymbol{B}_{\text{sensorFinal}}$ measured in the *sensor* frame to the *limb* coordinate system, we calculate R for all sensors independently and apply the inverse matrix of R to all input data:

$$\boldsymbol{B}_{\text{limb}} = R^{-1} \cdot \boldsymbol{B}_{\text{sensorFinal}} . \tag{10}$$

4.3 Estimating the limb's orientation

As every transducer is expected to be translated and rotated independently once the calibration is finished, a robust method is needed to calculate the local rotation matrix of the individual limbs. It is still assumed that the manual acceleration at every sensor is so low that it can be neglected. Equation (10) is used to calculate $\boldsymbol{B}_{\text{limb}}$, which is the global acceleration vector projected into the calibrated coordinate system of the limb that the individual sensor is attached to. If the limb's axes were aligned with the axes of the *global* reference frame, $\boldsymbol{B}_{\text{limb}}$ should be $[0 \;\; 0 \;\; -1]^T$ as shown in Figure 7.

Any change in orientation due to rotation of the sensing unit about the x or y-axis will cause a change of this vector. As there is no definite solution to decide what kind of rotation was applied to the object, the rule of *Occam's razor* was applied instead. This heuristic maxim advises simplicity in scientific theories of all kind [33] and is generally stated as

> *"Entities should not be multiplied beyond necessity."*

In terms of this calculation a *single* rotation about an arbitrary axis is chosen. As the rotation axis can be easily found using the cross product, the rotation with the smallest angle is considered the most simple solution that gets picked. The rotation matrix S containing this single rotation will be the estimated orientation matrix of the sensor and will have to transform $[0 \;\; 0 \;\; -1]^T$ into $\boldsymbol{B}_{\text{limb}}$:

$$S \cdot \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} = \boldsymbol{B}_{\text{limb}} = \begin{pmatrix} w_x \\ w_y \\ w_z \end{pmatrix} . \tag{11}$$

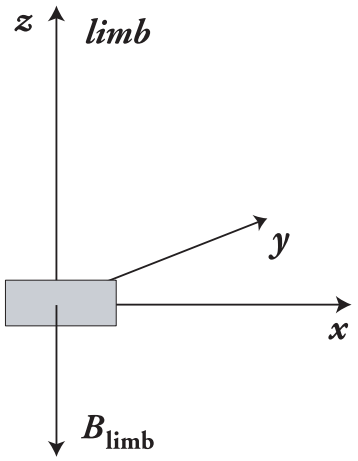

Fig. 7. If the accelerometer is not tilted but aligned to the global coordinate axes, $\boldsymbol{B}_{\text{limb}}$ matches the negative z-axis

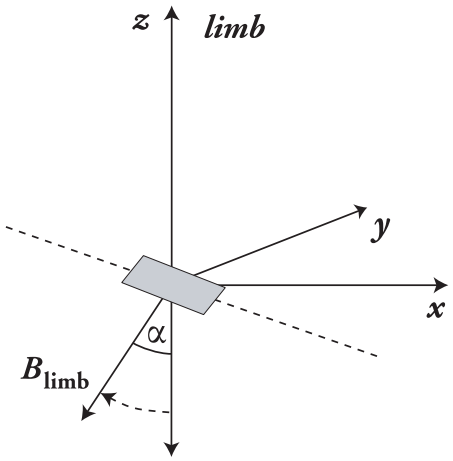

Fig. 8. The angle α and the rotation axis found by applying the cross product of the negative z-axis and the measured acceleration vector are used to establish the rotation matrix

The simplest way of achieving this transformation is a rotation of angle α about the axis $\boldsymbol{r}$ obtained by the cross product:

$$\boldsymbol{r} = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} \times \boldsymbol{B}_{\text{limb}} \ , \tag{12}$$

$$\alpha = \text{atan2}(\sqrt{w_x^2 + w_y^2} \ , \ w_z) \ . \tag{13}$$

The rotation matrix S is established using standard trigonometry. Finally, the local x-axis in the *limb* reference frame is transformed into the *global* reference frame using S. As it was assumed that the x-axis of the *limb* coordinate system points along the limbs they are attached to, this is the estimated orientation of the limb in the *global* reference frame.

$$\boldsymbol{B}_{\text{global}} = S \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \ . \tag{14}$$

$\boldsymbol{B}_{\text{global}}$ is calculated in real-time during the capturing process and written to the output file for all sensors that were properly calibrated earlier.

5 Recreating the captured motion

5.1 Preparing the input data

After importing the processed data into Maya, a polynomial Savitzky-Golay smoothing filter with variable filter width and polynom order is applied to remove high frequency noise [34, 35]. This filter type is the preferred choice in this context because it is much more reliable in keeping the magnitude of local extreme values and preserving detail (see Figure 9).

A transform node is created to hold the imported data which can be edited with Maya's keyframe curve editing tools.

5.2 Establishing the kinematic chain

If the sensors used during recording were part of a kinematic chain, the user would have to define the way the limbs were connected. A skeleton rig using Maya's internal joint system is created dynamically according to the user's specifications. By definition, the x-axis of the created joints points along the bone to secure conformity with the alignment described in Section 4.2. The individual position of the root joint(s) and the length of every bone can be easily edited with the provided user interface. To connect the kinematic chain with the sensor data, Maya's aim constraint functionality is used to transfer the orientation information to the individual limbs. Let $\boldsymbol{j_n}(t)$ be the position of the nth joint

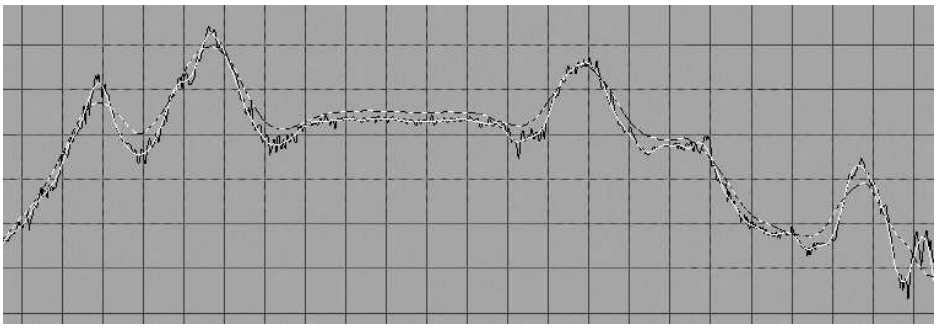

Fig. 9. Comparison of smoothing with a moving average filter and the Savitzky-Golay filter with typical filter parameters (dotted: moving average, white: Savitzky-Golay, black: original data)

at time t, and $\boldsymbol{B}^{\mathrm{n}}_{\mathrm{global}}(t)$ the captured orientation of limb n at time t. The aim vector $\boldsymbol{aim_n}(t)$ for joint n can be calculated by

$$\boldsymbol{aim_n}(t) = \boldsymbol{j_n}(t) + \boldsymbol{B}^{\mathrm{n}}_{\mathrm{global}}(t) \; . \tag{15}$$

If the length of the nth joint is denoted by l_n, the position of joint n can be found using

$$\boldsymbol{j_n}(t) = \boldsymbol{j_{n-1}}(t) + \left(l_n \boldsymbol{B}^{\mathrm{n}}_{\mathrm{global}}(t)\right) \; , \tag{16}$$

where $\boldsymbol{j_0}$ has to be specified by the user.

As mentioned earlier, the captured data is lacking any rotation about the world-up-axis (the z-axis in all examples given). This is due to the fact that the measured gravitational acceleration will not change during a rotation about the line of gravity. The effect this has on the recreated motion depends largely on the type of motion performed by the actor. While a complex arm movement sequence like picking up an item from the ground and primarily moving it along a vertical path will be captured with good accuracy, the motion of an arm while cleaning a horizontal surface will hardly be captured at all. To compensate for this inability, the user can animate the joint rotation about the global z-axis manually. Therefore the desired angle θ is used to transform the original vector:

$$\boldsymbol{B}_{\mathrm{globalRotated}} = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \boldsymbol{B}_{\mathrm{global}} \; . \tag{17}$$

5.3 Animation retiming with acceleration data

In a second approach that can be used independently or in combination with the presented method of orientation estimation we utilize the acceleration magnitude of a sensing unit to retime an existing motion. This can be either a hand-keyframed motion or a motion sequence previously captured by the accelerometers.

In contrast to the first approach, the sampled acceleration data of the sensing units is not used to derive kinematic or angular information but to extract information about the timing of the performed motion. This is done by trying to fit the acceleration curve of different objects over time to the measured acceleration information of the sensor(s). First the user chooses the 3D object for which he wants to act out the previously keyframed motion. This object needs to provide information about its position in space at a specific point in time.

A numerical approximation is used to calculate the second derivative of the object's translation:

$$f''(t) \approx \frac{f(t - \Delta t) - 2f(t) + f(t + \Delta t)}{\Delta t^2} \ , \tag{18}$$

where t is the current time and $f(t)$ is one of the xyz translation curves of the selected object. As animation curves are modeled as cubic splines in the animation system that is used, the approximated acceleration curves are linear functions. The acceleration magnitude of the object is then calculated by

$$g_{Acc}(t) = \sqrt{f''_x(t)^2 + f''_y(t)^2 + f''_z(t)^2} \ . \tag{19}$$

The next step involves finding peaks of high magnitude in g_{Acc}. Therefore all local maxima with magnitudes higher than a predefined threshold value are examined; a peak marker is inserted if there is a sign change in the slope of g_{Acc} at t_n and if the slope change $\Delta\, m$ is larger than a certain threshold value ϵ.

$$\Delta\, m = |g_{Acc}(t_n - \Delta t) - g_{Acc}(t_n + \Delta t)| \ . \tag{20}$$

The acceleration magnitude of the measured sensor data is then calculated using the following formula:

$$h_{Acc}(t) = |(|\boldsymbol{B}_{\mathrm{sensorNorm}}(t)| - g)| \ , \tag{21}$$

where $\boldsymbol{B}_{\mathrm{sensorNorm}}(t)$ is the non-calibrated sensor output at time t as outlined in Section 4.1. Magnitude peaks are identified using the method described above. The user is able to manipulate the established peaks of both curves independently using Maya's integrated keyframe editing tools. Once the peaks of the two curves are determined accordingly, the program redistributes all of the object's keyframes. Keyframes situated at acceleration peaks of the original motion curve are placed at the time of the corresponding peak of the sensor data curve. The keyframes between the acquired peaks are distributed over time using linear interpolation.

5.4 Motion warping

The user can then edit the captured motion by adding kinematic constraints to the existing motion curve. A typical example would be the captured movement of an arm reaching for an object. In order to make the arm reach out just as

far as wanted, it is desirable to be able to redefine the orientation of the limbs at the moment of grasping the object. Therefore, two additional cubic spline curves per coordinate channel are introduced to displace the original motion curve according to the user's specifications.

While $s(t)$ is used to scale the original motion curve, $o(t)$ is added to it as an offset. The functions are initialized with

$$s(t) = 1 \ \ \forall\, t\ , \tag{22}$$

$$o(t) = 0 \ \ \forall\, t\ . \tag{23}$$

The warped motion curve $\boldsymbol{B}_{\mathrm{globalWarped}}(t)$ is calculated by

$$\boldsymbol{B}_{\mathrm{globalWarped}}(t) = \boldsymbol{B}_{\mathrm{globalRotated}}(t)\boldsymbol{s}(t) + \boldsymbol{o}(t)\ . \tag{24}$$

Every time the user wants to add a kinematic constraint, he gets to decide whether the displacement should be accomplished using scale or offset warping. The graphical interface provides a handle at the selected joint which can be translated by the user to specify the desired orientation $\boldsymbol{B}_{\mathrm{globalWarped}}(t_c)$ of the joint at the current time t_c. Once the user is satisfied with the new joint orientation, Equation 24 is solved for either $s(t_c)$ or $o(t_c)$ (depending on the user's choice) and a keyframe on the scale or offset curve at time t_c is added. In order to use the refined limb orientation data, Equation 15 has to be rewritten to

$$\boldsymbol{aim_n}(t) = \boldsymbol{j_n}(t) + \boldsymbol{B}^{\mathrm{n}}_{\mathrm{globalWarped}}(t)\ . \tag{25}$$

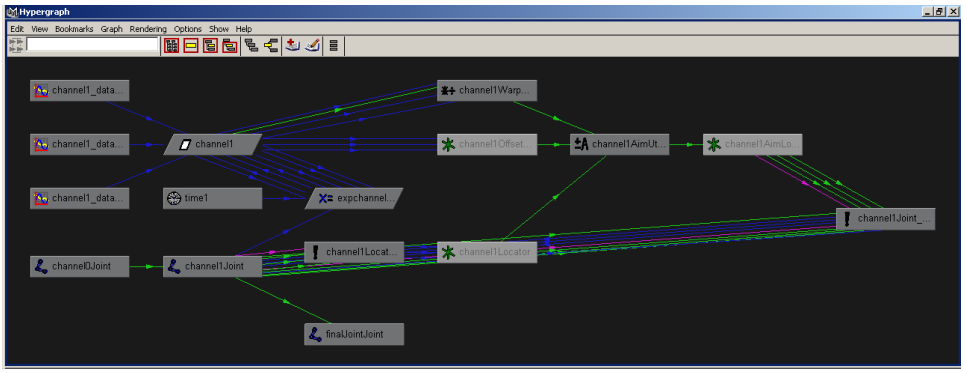

Fig. 10. Screenshot of the Maya node network establishing the motion representation and employing motion warping functionality

6 Results

The calibration procedure for establishing the rotation matrices for the individual sensors is easy and fast. It typically takes just a few seconds per limb or

even less if several sensors are part of the same kinematic chain and can be calibrated simultaneously (e.g. arm or leg). The obtained matrices yield accurate results even if the angle between the two measured calibration vectors differed significantly from 90 degrees.

Several typical motion sequences of the upper torso with sensors attached to the actor's arm, forearm, and hand were executed and captured. The resemblance of the physical motion to the synthesized representation depends largely on the amount of rotation about the axis of gravity. No movements can be captured while limbs are moved on a plane perpendicular to the line of gravity (e.g. on a table). In contrast, movements along a vertical plane (e.g. swing of arms during walking, arm waving, arm reaching out for a glass of water, legs walking) are reproduced with good accuracy.

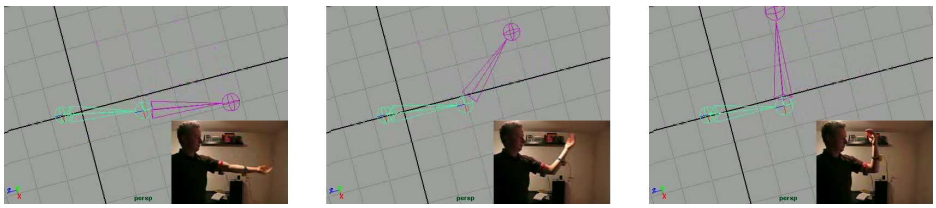

Fig. 11. Comparison of actual limb orientation and recreated kinematic chain posture in Maya. Two accelerometer units were used in this example

A lot of movements can be captured without the need of acting them out in slow motion. Movements that imply high acceleration peaks (e.g. ground contact during walking, fast or sudden movements in sports or action sequences) have to be acted out in slow motion. As no drift or offset error is introduced during periods of high acceleration, the user can either retime the captured slow motion sequence or act out the fast motion and fix kinematic errors due to acceleration peaks in Maya.

Although the user is provided with tools to overcome the known restrictions, it turned out that often there is no intuitive way to identify and add the "missing" amount of rotation to the captured motion. This can lead to a marked increase in the time needed to establish the desired quality of the motion.

The animation retiming method presented in Section 5.3 works reliably if the captured motion implies the occurrence of well-defined acceleration peaks (e.g. strong waving). If the algorithm fails to determine all peaks in the captured motion, Maya's keyframe editing tools have to be used to add or delete peak markers. Often, the original motion has to be exaggerated in order to produce identifiable acceleration peaks. In addition, the approach implies that the original keyframes need to have proper relative timing between acceleration peaks as the keyframes are distributed using linear interpolation. Using the arclength of the acceleration magnitude curve between t_n and t_{n+1} for distribution of the keyframes between peaks did not produce any satisfying results.

7 Future work

Further research has to be done in order to find ways to improve the reliability of the sensor data interpretation. Luinge and Veltink [36] showed how Kalman filtering can be employed to improve orientation estimation using erroneous sensor data.

Instead of estimating the orientation of the captured limb's x-axis, the rotation of the limb about the x- and y-axis could be extracted from the sensor data. So far, Maya establishes all three angular degrees of freedom for each joint using the given aim constraint. If a way can be found to obtain reliable rotation values, they could be assigned directly to the joints' x and y rotation channel. The user would get to animate the third rotation channel.

In order to improve the presented animation retiming method, a curve matching algorithm such as Bruderlin's *dynamic timewarping* [31] could be employed. So far, only peaks in the sensor data are used for redistribution of keyframes. Acceleration peaks will also occur while accidentally hitting the sensor against an external object. *Dynamic timewarping* in combination with a global optimization algorithm like *simulated annealing* could drastically improve the quality of the resulting motion.

8 Bibliography

References

1. Moeslund, T.B., Granum, E.: A survey of computer vision-based human motion capture. Computer Vision and Image Understanding: CVIU **81** (2001) 231–268
2. Vicon Peak: Motion Capture Systems from Vicon Peak (2006) http://-www.vicon.com [Online; accessed 13-June-2006].
3. Dontcheva, M., Yngve, G., Popovic, Z.: Layered acting for character animation. ACM Transactions on Graphics **22** (2003) 409–416
4. Chai, J., Hodgins, J.K.: Performance animation from low-dimensional control signals. ACM Transactions on Graphics **24** (2005) 686–696
5. Lee, J., Chai, J., Reitsma, P.S.A., Hodgins, J.K., Pollard, N.S.: Interactive control of avatars animated with human motion data. In: SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques, New York, ACM Press (2002) 491–500
6. Ren, L., Shakhnarovich, G., Hodgins, J.K., Pfister, H., Viola, P.: Learning silhouette features for control of human motion. ACM Transactions on Graphics **24** (2005) 1303–1331
7. de Aguiar, E., Theobalt, C., Magnor, M., Theisel, H., Seidel, H.P.: M^3: Marker-free model reconstruction and motion tracking from 3D voxel data. In: Computer Graphics and Applications, 12th Pacific Conference on (PG'04), Los Alamitos, IEEE Computer Society (2004) 101–110
8. Measurand Inc.: Motion Capture - ShapeWrap II (2006) http://www.-measurand.com/products/ShapeWrap.html [Online; accessed 13-June-2006].
9. Animazoo: GypsyGyro-18 inertial motion capture (2006) http://www.-animazoo.com/products/gypsyGyro.htm [Online; accessed 13-June-2006].

10. STMicroelectronics: LIS3L02AQ (2006) http://www.st.com/stonline/products/-literature/ds/9321/LIS3L02AQ.pdf [Online; accessed 13-June-2006].
11. Mäntyjärvi, J., Kela, J., Korpipää, P., Kallio, S.: Enabling fast and effortless customisation in accelerometer based gesture interaction. In: MUM '04: Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia, New York, ACM Press (2004) 25–31
12. Benbasat, A.Y., Paradiso, J.A.: An inertial measurement framework for gesture recognition and applications. In: GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction, London, Springer-Verlag (2002) 9–20
13. Aminian, K., Najafi, B.: Capturing human motion using body-fixed sensors: outdoor measurement and clinical applications. Computer Animation and Virtual Worlds **15** (2004) 79–94
14. Bachmann, E.R., McGhee, R.B., Yun, X., Zyda, M.J.: Inertial and magnetic posture tracking for inserting humans into networked virtual environments. In: VRST '01: Proceedings of the ACM symposium on Virtual reality software and technology, New York, ACM Press (2001) 9–16
15. Zhu, R., Zhou, Z.: A real-time articulated human motion tracking using tri-axis inertial/magnetic sensors package. IEEE Transactions on Neural Systems and Rehabilitation Engineering **12** (2004) 295–302
16. Sabatini, A.M.: Quaternion-based strap-down integration method for applications of inertial sensing to gait analysis. Medical and Biological Engineering and Computing **43** (2005) 94–101
17. Chen, K.Y., David R. Bassett, J.: The technology of accelerometry-based activity monitors: Current and future. Medicine & Science in Sports & Exercise **37** (2005) S490–S500
18. Mathie, M.J., Celler, B., Lovell, N., Coster, A.: Classification of basic daily movements using a triaxial accelerometer. Medical and Biological Engineering and Computing **42** (2004) 679–687
19. Zhou, H., Hu, H.: Inertial motion tracking of human arm movements in stroke rehabilitation. In: Proceedings of the IEEE International Conference on Mechatronics & Automation, Niagara Falls, IEEE (2005) 1306–1311
20. Mayagoitia, R.E., Nene, A.V., Veltink, P.H.: Accelerometer and rate gyroscope measurement of kinematics: an inexpensive alternative to optical motion analysis systems. Journal of Biomechanics **35** (2002) 537–542
21. Luinge, H.J., Veltink, P.H.: Inclination Measurement of Human Movement Using a 3-D Accelerometer With Autocalibration. IEEE Transactions on Neural Systems and Rehabilitation Engineering **12** (2004) 112–121
22. Hansson, G., Asterland, P., Holmer, N., Skerfving, S.: Validity and reliability of triaxial accelerometers for inclinometry in posture analysis. Medical and Biological Engineering and Computing **39** (2001) 405–413
23. Kemp, B., Janssen, A.J., van der Kamp, B.: Body position can be monitored in 3D using miniature accelerometers and earth-magnetic field sensors. Electroencephalography and clinical Neurophysiology **109** (1998) 484–488
24. Giansanti, D., Macellari, V., Maccioni, G., Cappozzo, A.: Is it feasible to reconstruct body segment 3-D position and orientation using accelerometric data? IEEE Transactions on Biomedical Engineering **50** (2003) 476– 483
25. Welch, G.F., Bishop, G.: An introduction to the kalman filter. Technical report, University of North Carolina at Chapel Hill, NC (1995)
26. Rehbinder, H., Hu, X.: Drift-free attitude estimation for accelerated rigid bodies. Automatica **40** (2004) 653–659

27. Welch, G.F.: Hybrid self-tracker: An inertial/optical hybrid three-dimensional tracking system (1995)
28. Lasseter, J.: Principles of traditional animation applied to 3D computer animation. In: SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques, New York, ACM Press (1987) 35–44
29. Terra, S.C.L., Metoyer, R.A.: Performance timing for keyframe animation. In: SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation, New York, ACM Press (2004) 253–258
30. Thorne, M., Burke, D., van de Panne, M.: Motion doodles: an interface for sketching character motion. ACM Transactions on Graphics **23** (2004) 424–431
31. Bruderlin, A., Williams, L.: Motion signal processing. In: SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, New York, ACM Press (1995) 97–104
32. Witkin, A., Popovic, Z.: Motion warping. In: SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, New York, ACM Press (1995) 105–108
33. Wikipedia: Occam's razor – Wikipedia, The Free Encyclopedia (2006) http://en.wikipedia.org/wiki/Occam's_Razor [Online; accessed 13-June-2006].
34. Taygeta Scientific Inc.: Digital Filtering, Savitzky-Golay Digital Filter Software in C++ (2006) http://www.taygeta.com/dfilter.xml [Online; accessed 13-June-2006].
35. Press, W., Teukolsky, S., Vetterling, W., Flannery, B.: Numerical recipes in C: the art of scientific computing. Cambridge University Press (1992) 650-655.
36. Luinge, H.J., Veltink, P.H.: Measuring orientation of human body segments using miniature gyroscopes and accelerometers. Medical and Biological Engineering and Computing **43** (2005) 273–282

Books used for reference:

– Adams, M., Miller, E., and Sims, M.: Inside Maya 5. New Riders Publishing (2003).
– Gould, D.: Complete Maya Programming. An Extensive Guide to MEL and the C++ API. Morgan Kaufmann Publishers (2003).
– Ritchie, K., Callery, J., and Biri, K.: The Art of Rigging, Volume 1. CG Toolkit (2005).
– Shirley, P.: Fundamentals of Computer Graphics. A. K. Peters (2002).

9 Appendices

CD-ROM containing

- Maya MEL Scripts
- C# .NET Source Code + Executable
- Video showing calibration and capture example

10 Acknowledgements

I gratefully acknowledge the supporting work, ideas, and efforts of my thesis supervisor Prof. Dr. Jörn Loviscach. I would like to thank my family and Melissa K. Berz for love and support. Finally, I want to thank God for his unconditional love and the blessing of life.

11 Ehrenwörtliche Erklärung

Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Arbeit selbständig, ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel und Literatur angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle verzeichnet.

Die nachstehend aufgeführten Personen haben in der jeweils beschriebenen Weise unentgeltlich geholfen:

- Prof. Dr. Jörn Loviscach: Ideenfindung und Betreuung der vorliegenden Arbeit, Bereitstellung der verwendeten Hardwarekompenenten, Bereitstellung einer Textpassage mit Erläuterungen zur verwendeten Hardware (im Text gekennzeichnet)
- Melissa K. Berz: Korrektur von Grammatik- und Rechtschreibfehlern im Text

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten in Anspruch genommen. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

Ich versichere ehrenwörtlich, dass ich nach bestem Wissen die reine Wahrheit gesagt und nichts verschwiegen habe.

Bünde, den 4. Juli 2006

Jan-Phillip Tiesel