OSC Implementation and Evaluation of the Xsens MVN suit

Ståle A. Skogstad and Kristian Nymoen
fourMs group - Music, Mind, Motion, Machines
University of Oslo,
Department of Informatics
{savskogs,krisny}@ifi.uio.no

Yago de Quay
University of Porto, Faculty of Engineering
Rua Dr. Roberto Frias, s/n
4200-465 Portugal
yagodequay@gmail.com

Alexander Refsum Jensenius
fourMs group - Music, Mind, Motion, Machines
University of Oslo,
Department of Musicology
a.r.jensenius@imv.uio.no

ABSTRACT

The paper presents research about implementing a full body inertial motion capture system, the Xsens MVN suit, for musical interaction. Three different approaches for streaming real time and prerecorded motion capture data with Open Sound Control have been implemented. Furthermore, we present technical performance details and our experience with the motion capture system in realistic practice.

1. INTRODUCTION

Motion Capture, or MoCap, is a term used to describe the process of recording movement and translating it to the digital domain. It is used in several disciplines, especially for bio-mechanical studies in sports and health and for making lifelike natural animations in movies and computer games. There exist several technologies for motion capture [1]. The most accurate and fastest technology is probably the so-called infra-red optical marker based motion capture systems (IrMoCap)[11].

Inertial MoCap systems are based on sensors like accelerometers, gyroscopes and magnetometers, and perform *sensor fusion* to combine their output data to produce a more drift free position and orientation estimation. In our latest research we have used a commercially available full body inertial MoCap system, the Xsens MVN[1] suit [9]. This system is characterized by having a quick setup time and being portable, wireless, moderately unobtrusive, and, in our experience, a relatively robust system for on-stage performances. IrMoCap systems on the other hand have a higher resolution in both time an space, but lack these stage-friendly properties. See [2] for a comparison of Xsens MVN and an IrMoCap system for clinical gait analysis.

Our main research goal is to explore the control potential of human body movement in musical applications. New MoCap technologies and advanced computer systems bring new possibilities of how to connect human actions with musical expressions. We want to explore these possibilities and see how we can increase the connection between the human body's motion and musical expression; not only focusing on the performer, but also on how the audience perceives the performance.

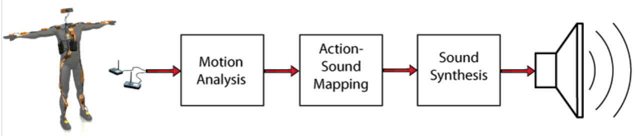

Figure 1: The Xsens suit and possible data flow when using it for musical interaction.

To our knowledge, we are among the first to use a *full body* inertial sensor based motion capture suit in a musical setting, and hence little related work exists. Lympouridis et. al. has used the inertial system Orient-2/-3 for sonification of *gestures* and created a framework for "bringing together dancers, composers and musicians" [6][5]. Meas et. al have used 5 inertial (Xsens) sensors to quantify the relation between sound stimuli and bodily response of subjects [7]. An upper body mechanical system has briefly been examined by [3]. See [11] for a review of related work in the area of IrMoCap for musical interaction.

In the next section, we will give a brief overview of the Xsens MVN technology. Then in section 3 we will report on three Open Sound Control implementations for the Xsens system and discuss some of our reflections. In section 4 we will give our evaluation and experience with the Xsens MVN system, before we propose a technology independent real time MoCap toolbox in section 5.

2. THE XSENS MVN TECHNOLOGY

The Xsens MVN technology can be divided into two parts. First, the sensor and communication hardware are responsible for collecting and transmitting the raw sensor data. Second, these data are treated by the Xsens MVN software engine, which interprets and reconstructs the data to full body motion while trying to minimize drift.

2.1 The Xsens MVN Suit (Hardware)

The Xsens MVN suit consists of 17 inertial MTx sensors, which are attached to key areas of the human body [9]. Each sensor consists of a 3D gyroscope, 3D accelerometer and magnetometer. The raw signals from the sensors are connected to a pair of Bluetooth 2.0 based wireless transmitters, which transmit the raw motion capture data to a pair of wireless receivers. The total weight of the suit is approximately 1.9 kg and the whole system comes in a suitcase with the total weight of 11 kg.

2.2 The Xsens MVN engine (Software)

The data from the Xsens MVN suit is fed to the MVN software engine that uses sensor fusion algorithms to produce

[1] *Xsens MVN* (MVN is a name not an abbreviation) is a motion capture system designed for the human body and is not a generic motion capture device.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
NIME'11, 30 May–1 June 2011, Oslo, Norway.
Copyright remains with the author(s).

absolute orientation values, which are used to transform the 3D linear accelerations to global coordinates. These in turn are translated to a human body model which implements joint constraints to minimize integration drift [9].

The Xsens MVN system outputs information about body motion by expressing body postures sampled at a rate up to 120Hz. The postures are modelled by 23 body segments interconnected with 22 joints [9]. The Xsens company offers two possibilities of using the MVN fusion engine: the Windows based *Xsens MVN Studio* and a software development kit called *Xsens MVN SDK*.

2.3 How to use the System

There are three main suit configurations; full body, upper body or lower body. When the suit is properly configured, calibration is needed to initialize the position and orientation of the different body segments. When we are satisfied with the calibration the system can be used to stream the motion data to other applications in real-time or perform recordings for later playback and analysis.

How precise one needs to perform the calibration may vary. We have found that so-called *N-pose* and *T-pose* calibrations are the most important. A *hand touch* calibration is recommended if a good relative position performance between the left and right hand is wanted. Recalibration can be necessary when the system is used over a longer period of time. It is also possible to input body measurements of the tracked subject to the MVN engine, but we have not investigated if this extra calibration step improves the quality of data for our use.

In our experience, setting up the system can easily be done in less than 15 minutes compared to several hours for IrMoCap systems [2].

2.4 Xsens MVN for Musical Interaction

A typical model for using the Xsens suit for musical application is shown in Figure 1. In most cases, motion data from the Xsens system must be processed before it can be used as control data for the sound engine. The complexity of this stage can vary from simple scaling of position data to more complex pattern recognition algorithms that look for mid/higher-level cues in the data. We will refer to this stage as *cooking* the motion capture data.

The main challenges of using the Xsens suit for musical interaction fall into two interconnected groups. Firstly, the purely technical challenges, such as minimizing latency, managing network protocols and handling data. Secondly, the more artistic challenges involving questions like how to make an aesthetically pleasing connection between action and sound. This paper will mainly cover the technical challenges.

3. IMPLEMENTATION

To be able to use the Xsens MVN system for musical interaction, we need a way to communicate the data that the system senses to our musical applications. It was natural to implement the OSC standard since the Xsens MVN system offers motion data which is not easily related to MIDI signals. OSC messages are also potentially easier to interpret since these can be written in a human readable form.

3.1 Latency and Architecture Consideration

Low and stable latency is an important concern for *real-time* musical control [12]. This is therefore an important issue to consider when designing our system. Unfortunately, running software and sending OSC messages over normal computer networks offers inadequate support for synchronization mechanisms, since standard operating systems do not support this without dedicated hardware [10]. In our experience, to get low latency from the Xsens system, the software needs to run on a fast computer that is not overloaded with other demanding tasks. But how can we further minimize the latency?

3.1.1 Distribution of the Computational Load

From Figure 1 we can identify three main computationally demanding tasks that the data need to traverse before ending up as sound. If these tasks are especially demanding, it may be beneficial to distribute these computational loads to different computers. In this way we can prevent a computer from suffering too much from computational load, which can lead to a dramatic increase of latency and jitter. This is possible with fast network links and a software architecture that supports the distribution of computational loads. However, it comes at the cost of extra network overhead, so one needs to check if the extra cost does not exceed the benefits.

3.1.2 The Needed Communication Bandwidth

The amount of data sent through a network will partly be related to the experienced network latency. For instance, we should try to keep the size of the OSC bundles lower than the maximum network buffer size,[2] if the lowest possible network latency is wanted. If not, the bundle will be divided into several packages [10]. To achieve this, it is necessary to restrict the amount of data sent. If a large variety of data is needed, we can create a dynamic system that turns different data streams on when needed.

3.2 OSC Implementations

There are two options for using the Xsens MVN motion data in real time, either we can use the Xsens Studio's UDP network stream, or make a dedicated application with the SDK. The implementation must also support a way to effectively cook the data. We begun using the UDP network stream since this approach was the easiest way to start using the system.

3.2.1 MVN Network Stream Unpacker in Max/MSP

A MXJ Java datagram unpacker was made for Max/MSP, but the implementation was shown to be too slow for real time applications. Though a dedicated Max external (in C++) would probably be faster, this architecture was not chosen for further development since Max/MSP does not, in our opinion, offer an effective data cooking environment.

3.2.2 Standalone Datagram Unpacker and Cooker

We wanted to continue using the Xsens Studio's UDP network stream, but with a more powerful data cooking environment. This was accomplished by implementing a standalone UDP datagram unpacking application. The programming language C++ was chosen since this is a fast and powerful computational environment. With this implementation we can either cook the data with self produced code or available libraries. Both raw and cooked data can then be sent as OSC messages for further cooking elsewhere or to the final sound engine.

3.2.3 Xsens MVN SDK Implementation

The Xsens MVN software development kit offers more data directly from the MVN engine compared to the UDP network stream. In addition to position, we get: positional and angular acceleration, positional and angular velocity and information about the sensor's magnetic disturbance. Every

[2] Most Ethernet network cards support 1500 bytes. Those supporting Jumbo frames can support up to 9000 bytes.

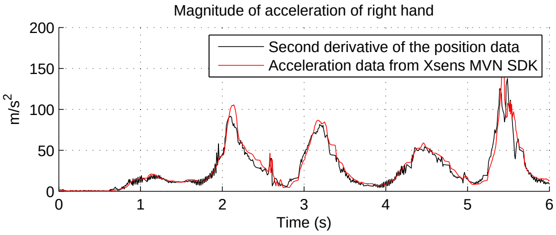

Figure 2: Difference between the second derivative of the position data versus the acceleration data obtained directly from MVN engine (SDK).

time frame is also marked with a time stamp that can be useful for analysis and synchronizing. Another benefit is that we have more control since we are directly communicating with the MVN engine and not listening for UDP packages. The drawback with the SDK is that we lose the benefit of using the user friendly MVN Studio and its GUI.

We implemented a terminal application with the SDK, that supports the basic Xsens features (calibration, playback, etc.). Since the application is getting data directly from the MVN engine we can save network overhead by cooking them in the same application before sending them as OSC messages. We also implemented a function that can send the motion data in the same data format as the Network UDP Datagram stream. This stream can then be opened by MVN Studio to get real-time visual feedback of the MoCap data.

3.2.4 Discussion

Since the solution presented in 3.2.2 offered a fast environment for data cooking, and let us use the user friendly MVN Studio, we have mainly used this approach in our work. We later discovered that the network stream offered by MVN Studio suffers from frame loss when driven in live mode, which affects both solutions presented in 3.2.1 and 3.2.2. Because of this we plan to focus on our SDK implementation in the future. An added advantage is that we no longer need to differentiate the segments positional data to be able to get properties like velocity and acceleration, since the SDK offers this directly from the MVN Engine. These data, especially the acceleration, seems to be of a higher quality since they are computed directly on the basis of the Xsens sensors and not differentiated from estimated position data as shown in Figure 2.[3]

3.3 Cooking Full Body MoCap Data

The Xsens MVN offers a wide range of different data to our system. If we use the network stream from the MVN Studio, each frame contains information about the position and orientation of 23 body segments. This yields in total 138 floating points numbers at a rate of 120Hz. Even more data will be available if one instead uses the MVN SDK as the source. Also different transformations and combinations of the data can be of interest, such as calculating distances or angles between body limbs.

Furthermore, we can differentiate all the above mentioned data to get properties like velocity, acceleration and jerk. Also, filters can be implemented to get smoother data or to emphasize certain properties. In addition, features like quantity of motion or "energy" can be computed. And with pattern recognition techniques we have the potential to recognize even higher level features [8].

We are currently investigating the possibilities that the Xsens MVN suit provides for musical interaction, but the mapping discussion is out of scope for this paper. Nevertheless, we believe it is important to be aware of the characteristics of the data we are basing our action-sound mappings on. We will therefore present technical performance details of the Xsens MVN system in the following section.

4. PERFORMANCE

4.1 Latency in a Sound Producing Setup

To be able to measure the typical expected latency in a setup like that of Figure 1 we performed a simple experiment with an audio recorder. One laptop was running our SDK implementation and sent OSC messages containing the acceleration of the hands. A patch in Max/MSP was made that would trigger a simple impulse response if the hands' acceleration had a high peak, which is a typical sign of two hands colliding to a sudden stop. The time difference between the acoustic hand clap and the triggered sound should then indicate the typical expected latency for the setup.

The Max/MSP patch was in experiment 1 running on the same laptop[4] as the SDK. In experiment 2 the patch was run on a separate Mac laptop[5] and received OSC messages through a direct Gbit Ethernet link. Experiment 3 was identical to 2 except that the Mac was replaced with a similar Windows based laptop. All experiments used the same firewire soundcard, *Edirol FA-101*. The results are given in Table 1 and are based on 30 measurements each which was manually examined in audio software. The standard deviation is included as an indication of the jitter performance. We can conclude that experiment 2 has the fastest sound output response while experiments 1 and 3 indicate that the Ethernet link did not contribute to a large amount of latency.

The Xsens MVN system offers a direct USB connection as an option for the Bluetooth wireless link. We used this option in experiment 4, which was in other ways identical to experiment 2. The results indicate that the direct USB connection is around 10-15 milliseconds faster and has a lower jitter performance than the Bluetooth link.

The upper boundary for "intimate control" has been suggested to be 10ms for latency and 1ms for its variations (jitter) [12]. If we compare the boundary with our results, we see that overall latencies are too large and that the jitter performance is even worse. However, in our experience, the system is still usable in many cases dependent on the designed action-sound mappings.

Table 1: Statistical results of the measured action to sound latency, in milliseconds.

Experiment	min	mean	max	std. dev.
1 Same Win laptop	54	66.7	107	12.8
2 OSC to Mac	41	52.2	83	8.4
3 OSC to Win	56	68	105	9.8
4 OSC to Mac - USB	28	37.2	56	6.9

4.2 Frame Loss in the Network Stream

We discovered that the Xsens MVN Studio's (version 2.6 and 3.0) network stream is not able to send all frames when running at 120Hz in real time mode on our computer.[3] At this rate it is skipping 10 to 40 percent of the frames. This does not need to be a significant problem if one use "time independent" analysis, that is analysis that does not look at the history of the data. But if we perform differential calculations on the Xsens data streams, there will be large jumps

[3] The systems that tries to minimize positional drift probably contributes to a mismatch between differentiated positional data and the velocity and acceleration data from the MVN engine.

[4] Dell Windows 7.0 Intel i5 based laptop with 4GB RAM

[5] MacBook Pro 10.6.6, 2.66 GHz Duo with 4GB RAM

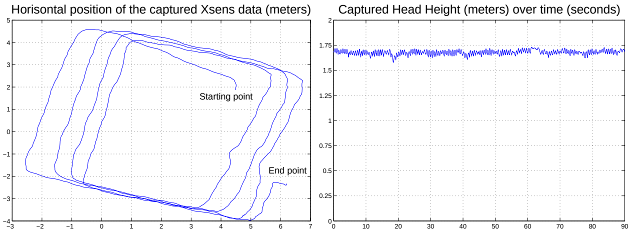

Figure 3: Plots of the captured horizontal (left) and vertical (right) position of the head.

in differentiated values during lost frames, hence noise. This was partly dealt with in the implementation described in 3.2.2. Whenever frames are detected as missing, the software will perform an interpolation. However, frame loss is still a major problem since we are not getting all the motion capture data and can lose important details in the data stream. For instance, if a trigger algorithm is listening for some sudden action, a couple of lost frames can make the event unrecognisable.

4.3 Positional Drift

The sensors in the Xsens MVN suit can only observe relative motion and calculate position through integration. This introduces drift. To be able to observe this drift we conducted a simple test by letting a subject walk along a rectangular path (around 6x7 meters) four times. Figure 3 shows a horizontal positional drift of about 2 meters during the 90 second long capture session. We can therefore conclude that Xsens MVN is not an ideal MoCap system if absolute horizontal position is needed.[6] The lack of drift in the vertical direction however, as can be seen in the right plot in Figure 3, is expected since the MVN engine maps the data to a human body model and assumes a fixed floor level.

4.4 Floor Level

If the motion capture area consists of different floor levels, like small elevated areas, the MVN engine will match the sensed raw data from the suit against the floor height where the suit was calibrated. This can be adjusted for in the post processing, but the real-time data will suffer from artifacts during floor level changes.

4.5 Magnetic Disturbance

The magnetic disturbance is critical during the calibration process but does not, to our experience, alter the motion tracking quality dramatically. During a concert we experienced significant magnetic disturbance, probably because of the large amount of electrical equipment on stage. But this did not influence the quality of MoCap data in such a way that it altered our performance.

4.6 Wireless Link Performance

Xsens specifies a maximum range up to 150 meters in an open field [13]. In our experience the wireless connection can easily cover an area with a radius of more than 50 meters in open air. Such a large area cannot be practically covered using IrMoCap systems.

We have performed concerts in three different venues.[7] During the two first concerts we experienced no problems with the wireless connection. During the third performance we wanted to test the wireless connection by increasing the distance between the Xsens suit and the receivers to about 20 meters. The wireless link also had an added challenge since the concert was held in a conference venue where we expected constant WIFI traffic. This setup resulted in problems with the connection and added latency. The distance should therefore probably be minimized when performing in venues with considerable wireless radio traffic.

4.7 Final Performance Discussion

We believe that the Xsens MVN suit, in spite of its shortcomings in latency, jitter and positional drift, offers useful data quality for musical settings. However, the reported performance issues should be taken into account when designing action-sound couplings. We have not been able to determine whether the Xsens MVN system preserves the motion qualities we are most interested in compared to other MoCap systems, nor how their performance compares in real life settings. To be able to answer more of these questions we are planning systematic experiments comparing Xsens MVN with other MoCap technologies.

5. FUTURE WORK

In Section 3.3 we briefly mentioned the vast amount of data that is available for action-sound mappings. Not only are there many possibilities to investigate, it also involves many mathematical and computational details. However, the challenges associated with the cooking of full body MoCap data are not specific to the Xsens MVN system. Other motion capture systems like IrMoCap systems offer similar data. It should therefore be profitable to make one cooking system that can be used for several MoCap technologies.

The main idea is to gather effective and fast code for real time analysis of motion capture data; not only algorithms but also knowledge and experience about how to use them. Our implementation is currently specialized for the the Xsens MVN suit. Future research includes incorporating this implementation with other motion capture technologies and develop a real time motion capture toolbox.

6. REFERENCES

[1] http://en.wikipedia.org/wiki/motion_capture.

[2] T. Cloete and C. Scheffer. Benchmarking of a full-body inertial motion capture system for clinical gait analysis. In *EMBS*, pages 4579 –4582, 2008.

[3] N. Collins, C. Kiefer, Z. Patoli, and M. White. Musical exoskeletons: Experiments with a motion capture suit. In *NIME*, 2010.

[4] R. Dannenberg. *Real-time scheduling and computer accompaniment.* MIT Press, 1989.

[5] V. Lympourides, D. K. Arvind, and M. Parker. Fully wireless, full body 3-d motion capture for improvisational performances. In *CHI*, 2009.

[6] V. Lympouridi, M. Parker, A. Young, and D. Arvind. Sonification of gestures using specknets. In *SMC*, 2007.

[7] P.-J. Maes, M. Leman, M. Lesaffre, M. Demey, and D. Moelants. From expressive gesture to sound. *Journal on Multimodal User Interfaces*, 3:67–78, 2010.

[8] G. Qian, F. Guo, T. Ingalls, L. Olson, J. James, and T. Rikakis. A gesture-driven multimodal interactive dance system. In *ICME*, 2004.

[9] D. Rosenberg, H. Luinge, and P. Slycke. Xsens mvn: Full 6dof human motion tracking using miniature inertial sensors. *Xsens Technologies*, 2009.

[10] A. Schmeder, A. Freed, and D. Wessel. Best practices for open sound control. In *LAC*, 2010.

[11] S. Skogstad, A. R. Jensenius, and K. Nymoen. Using ir optical marker based motion capture for exploring musical interaction. In *NIME*, 2010.

[12] D. Wessel and M. Wright. Problems and prospects for intimate musical control of computers. In *NIME*, 2001.

[13] Xsens Technologies B.V. *Xsens MVN User Manual.*

[6] The product *MVN MotionGrid* will improve this drift.

[7] First concert: `www.youtube.com/watch?v=m1OffxIArrAi`