A MULTILAYERED GDIF-BASED SETUP FOR STUDYING COARTICULATION IN THE MOVEMENTS OF MUSICIANS

Alexander Refsum Jensenius, Kristian Nymoen, Rolf Inge Godøy
Department of Musicology
University of Oslo
Pb 1017 Blindern
NO-0315 Oslo, Norway
{a.r.jensenius, kristian.nymoen, r.i.godoy}@imv.uio.no

ABSTRACT

The paper presents some challenges faced in developing an experimental setup for studying coarticulation in music-related body movements. This has included solutions for storing and synchronising motion capture, biosensor and MIDI data, and related audio and video files. The implementation is based on a multilayered Gesture Description Interchange Format (GDIF) structure, written to Sound Description Interchange Format (SDIF) files using the graphical programming environment Max/MSP.

1. INTRODUCTION

From our previous studies in the Musical Gestures Project, [1] we believe that several different types of *music-related movements* [2] are focused on what we call *goal-points* [2]. Such goal-points may be salient events in the music such as downbeats, or various accent types, or melodic peaks. In music performance goal-points are often reflected in the positions and shapes of the performers' effectors (fingers, hands, arms, torso, etc.) at certain moments in time, similar to what is called *keyframes* in animation. The movement trajectories between these goal-points, similar to what is known as *inter-frames* in animation, may often demonstrate the phenomenon of *coarticulation*, i.e. that the various smaller movements are subsumed under more superordinate and goal-directed movement trajectories [3].

To test our ideas of coarticulation, we are carrying out observation studies of musician's movements in our laboratory. Here the focus is on relationships between sound-producing actions (e.g. finger movements) and other types of music-related movements seen in the elbows, shoulders and head of the performer. We are particularly interested in looking at relationships between *chunks* of movement and sound, i.e. how separate sound-producing actions are grouped into larger composite movements [2]. We are also interested in studying how these relate to muscle tension in the arms, and whether patterns in the sound-producing actions and ancillary movements can be predicted from such biosignals.

The theoretical basis for our studies is outlined elsewhere [4]. This paper will present the experimental setup developed for the study, some of the challenges encountered in handling and storing data in the system, and a modular system implemented for the setup.

2. THE SETUP

The setup for our observation studies currently consists of the following equipment:

- 9 3D USB accelerometers from Phidgets Inc. [3]
- Polhemus Patriot with 2 6D sensors. [4]
- 2 BioFlex electromyography (EMG) sensors connected to a Wi-MicroDig sensor interface from Infusion Systems. [5]
- Yamaha Disklavier with MIDI I/O.
- Unibrain 520-i high-speed (86 fps) FW-camera.
- 2 channel FW audio device.

The 3D accelerometers are used for obtaining the relative movement of various parts of the body, and constitute a low-cost, yet versatile, motion capture system. Four of the accelerometers are mounted on a custom-built, adjustable "strap-on" system for the upper body: one accelerometer on each shoulder, one on the back of the neck, and one accelerometer on the lower back (see Figure 1). Additionally, one accelerometer is placed on the back of the performer's head, and two accelerometers on each arm; one close to the elbow and one close to the wrist. The USB cables are connected to a regular USB-hub fastened to the belt of the performer.

Absolute position and orientation of the two hands are obtained from the Polhemus Patriot 6D electromagnetic tracking system over a serial connection. The two Polhemus sensors are mounted next to the accelerometers on the wrists (see Figure 1). Each of the sensors output 3D position data (x, y, z) in inches and 3D orientation data

[1] http://musicalgestures.uio.no
[2] For example *sound-producing* and *sound-accompanying* movements. See [5] for a taxonomy of such music-related movements.
[3] http://www.phidgets.com
[4] http://www.polhemus.com
[5] http://www.infusionsystems.com

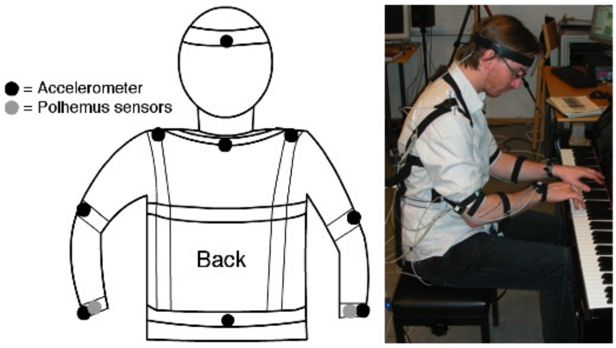

Figure 1. Sketch of the placements of accelerometers and polhemus sensors, and a picture from the pilot study.

(azimuth, elevation, roll) in degrees relative to the reference point placed at the left side of the piano keyboard.

The EMG sensors are placed on the inside of the lower arms to measure muscular activity related to the sound-producing actions (finger movements). These sensors are interfaced through a small Wi-microDig sensor interface communicating through a serial bluetooth connection.

The above mentioned motion capture and biosensing, together with audio, video and MIDI, calls for a solution for handling a wide range of data types. As Table 1 shows, we are dealing with data with different (and varying) sample rates, channels, resolution and bit rates. It has therefore been important to develop a solution for storing and synchronising the different streams, while still being able to efficiently play back and analyse the data later on.

Input	SR (Hz)	Ch.	Bit
Phidgets	60	3	32
Polhemus	60	6	32
BioFlex	100	2	7
Video	86	1	8
Audio	44100	2	16
MIDI	∼1000	1	7

Table 1. List of data used in our setup, columns from left: input device, sampling rate in Hz, number of channels and resolution in bits

3. RECORDING DATA

To allow for easy reconfiguration of our setup for various types of observation studies, we have found the need to develop a software solution that works with all our equipment and which is modular in nature. This has been done within the Jamoma modular framework [6] for Max/MSP/Jitter [8]. The result is a collection of modules, with a graphical interface, that can easily be combined in various ways.

For the current setup we have developed Jamoma modules for the Phidgets accelerometer (jmod.phidgets.accelerometer), the Polhemus Patriot (jmod.polhemus) and the Wi-MicroDig (jmod.wi-microdig). We are also using some of the video input and analysis modules previously developed by one of the authors (e.g. jmod.input% and jmod.-motion%) [5]. This allows for a simple and easy way to connect and work with the different devices.

Communication in and between modules is done using Open Sound Control (OSC), which simplifies creating multi-computer setups based on network communication. We are currently using two computers: a Windows PC is storing data from the accelerometers, the Polhemus system and the Yamaha Disklavier, while a Mac Pro is handling data from the EMG sensors as well as video and audio recording.

3.1. Storing GDIF data in SDIF files

The Gesture Description Interchange Format (GDIF) [7] is currently being developed to tackle problems related to streaming and storage of music-related movement data [6]. GDIF development is focusing on *what* to store, and not *how* to store the data, and has therefore been using different protocols and formats, mainly OSC for streaming and XML for storage [7].

Originally inspired by the Sound Description Interchange Format (SDIF), GDIF has always been thought of as a companion to SDIF. For the current setup we therefore wanted to use SDIF as the basis for the storage of GDIF data. One advantage of this approach is that many challenges related to the handling of multi-stream data files at high sampling rates have already been solved in SDIF [10]. It is also of great importance that movement data can be stored in the same file, or synchronised to, the related audio and MIDI data.

An SDIF file sorts data into separate *streams* along a common timeline, where each stream consists of time-tagged *frames* within which the actual data are stored as *matrices* [11]. The frames and matrices are type-specific, meaning that different frame types consist of a different number of matrices, and different matrix types consist of a different number of rows and columns. This allows for handling data at various resolutions and speeds, something which is ideal for our multidimensional data sets.

3.2. Defining new frame and matrix types

To store movement data into SDIF-files it is necessary to create a set of well-defined frame and matrix types. Taking on a full-body movement description has not been the goal of this study, but rather to develop the types necessary for our setup. The current layout is summarised in Figure 2, and shows that each recorded data file consists of five streams where each stream contains succeeding time-tagged frames. Each of these frames consists of one matrix of data, and they each have a unique ID that can be referred to.

The only standard SDIF frame and matrix type we use

[6] http://www.jamoma.org

[7] http://www.gdif.org

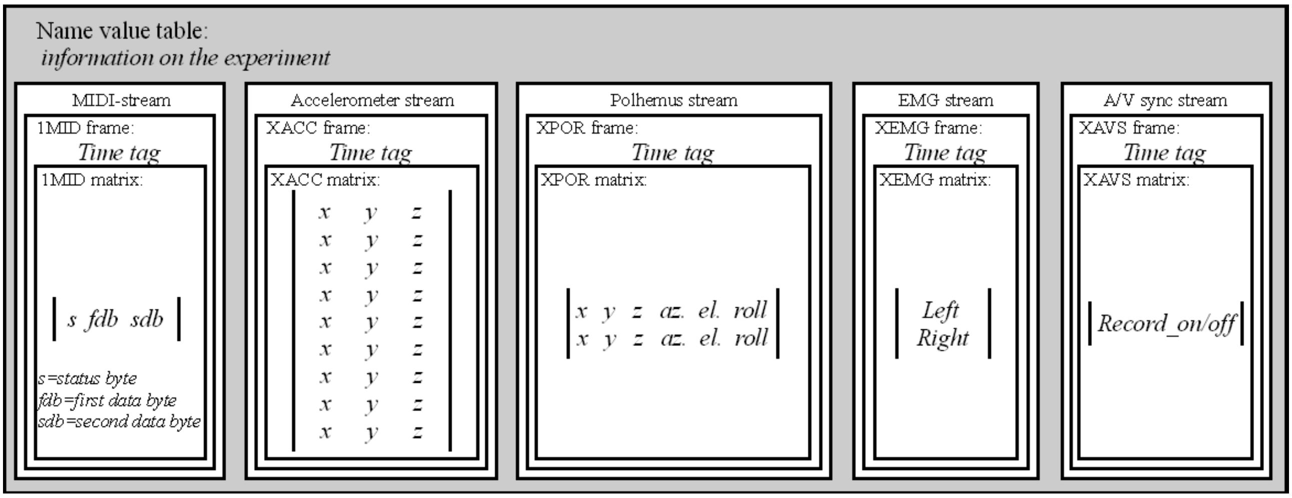

Figure 2. An overview of a cross-section of our recorded files. The various streams consist of different types of frames and matrices, each of which have different resolution and sampling rates.

is the 1MID for MIDI data. For storing position and orientation data, we have defined three different matrix types: [8]

```
1MTD XPOS { X, Y, Z }
1FTD XPOS { XPOS position; }

1MTD XORI { azimuth, elevation, roll }
1FTD XORI { XORI orientation; }

1MTD XPOR { X, Y, Z, azimuth, elevation, roll}
1FTD XPOR { XPOR position_and_orientation; }
```

For the moment we are using cartesian coordinates for the descriptions of points in space, but this should probably be extended to include polar coordinates in the future.

We have also found the need for storing both velocity and acceleration data, and have defined two matrix types for this:

```
1MTD XVEL { X, Y, Z }
1FTD XVEL { XVEL velocity; }

1MTD XACC { X, Y, Z }
1FTD XACC { XACC acceleration; }
```

For EMG data, we only need to store single values from the two sensors, and have therefore defined a very simple EMG type:

```
1MTD XEMG { level }
1FTD XEMG { XEMG electromyopgraphy; }
```

Since audio and video files are recorded on a separate computer, we use a synchronisation stream containing a binary on/off message recorded into a stream called XAVS:

```
1MTD XAVS { record_on }
1FTD XAVS { XAVS audio_video_sync; }
```

[8] The X in the abbreviated names means that these types are not part of the standard SDIF matrix types.

To avoid drift between recorded data and video, we tested storing each frame number of the recorded video into the synchronisation track. However, this increased the network traffic and CPU usage considerably without improving the quality of the recorded data in our fairly short recordings in the pilot study. Drift between recorded sensor data and media may be more of a problem in longer recording sessions and larger setups, and here it may be benificial to record the frame number of the video file at a regular interval (e.g. every 10 seconds).

3.3. Metadata

To secure more efficient data handling and usage within our own research group, but also to promote an increased exchange of recorded data in the community, we believe it is critical to include a generous amount of metadata about the recordings. For now, we include information about the author of the file, date and location of the recording, equipment used, experimental setup and subject name or number. Practice will show whether we need to be more specific in how such metadata should be formatted, and how detailed the descriptions should be. At the moment we believe the most important is that the information is human-readable, but it may also be relevant to devise a set of machine-friendly descriptors that can be used for information retrieval.

Previously, we have typically written metadata to a separate text file, and put it in the same folder as the data files. However, moving data files around, renaming, etc. may sometimes lead to orphaned files, and difficulties in interpreting the data. In our experience this becomes an even bigger problem when sharing files between institutions with different practices of data handling and storage.

One thing we find particularly compelling about storing GDIF-data in SDIF-files, is that the header is written as plain text (data is recorded binary). This makes it possible to quickly open a file in a regular text editor to get detailed information about the content.

3.4. Implementation

In our current setup we record GDIF data to SDIF-files using objects available in the FTM [9] collection for Max/MSP [9]. To simplify the process, we have developed a small collection of Jamoma modules to handle the recording and playback. This includes modules (Figure 3) for generating indexed file names (jmod.new_file_player), writing header information (jmod.sdif.record.nvt), setting up the different streams to be recorded (jmod.sdif.record.control) and writing the files (jmod.sdif.record).

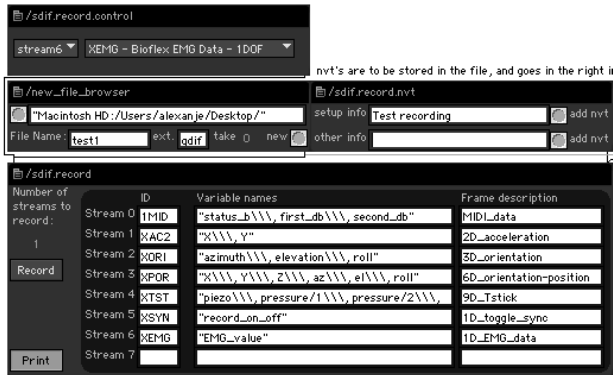

Figure 3. Various modules for setting up and writing GDIF data to SDIF files.

There is also a module for playing back files (jmod.sdif-play), where it is possible to select which stream(s) to output, the playback speed, and the location in the file from where to play (Figure 4).

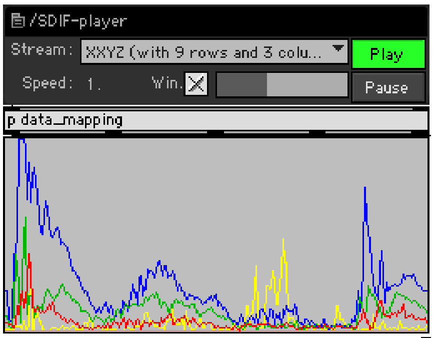

Figure 4. Module for playing back selected streams from a multilayered file.

4. FUTURE WORK

- Develop more frame and matrix types, and define a set of required metadata.
- Continue development of tools for handling GDIF/SDIF-files in Jamoma.
- Synchronisation and sharing of data with other systems, e.g. EyesWeb XMI [1].
- Develop tools for analysing GDIF data in Max/MSP and Matlab.

[9] http://ftm.ircam.fr

5. ACKNOWLEDGMENTS

Thanks to Diemo Schwarz and Norbert Schnell for feedback and comments on FTM and SDIF, and the Jamoma developers for close collaboration.

6. REFERENCES

[1] A. Camurri, G. Castellano, R. Cowie, D. Glowinski, B. Knapp, C. L. Krumhansl, O. Villon, and G. Volpe. The premio paganini project: a multimodal gesture-based approach for explaining emotional processes in music performance. In *Gesture Workshop, Lisbon*, 2006.

[2] R. I. Godøy. Reflections on chunking. In A. Schneider, editor, *Hamburger Jahrbuch für Musikwissenschaft*, volume 24, pages 117–132. Frankfurt, Peter Lang, 2008.

[3] R. I. Godøy, E. Haga, and A. R. Jensenius. Playing "air instruments": Mimicry of sound-producing gestures by novices and experts. In S. Gibet, N. Courty, and J.-F. Kamp, editors, *International Gesture Workshop. Revised Selected Papers*, volume 3881/2006, pages 256–267. Berlin Heidelberg: Springer-Verlag, 2006.

[4] R. I. Godøy, A. R. Jensenius, and K. Nymoen. Production and perception of goal-points and coarticulations in music. In *ASA-EAA Conference*, Paris, France, 2008.

[5] A. R. Jensenius. *Action–Sound : Developing Methods and Tools for Studying Music-Related Bodily Movement*. PhD thesis, University of Oslo, 2007.

[6] A. R. Jensenius, T. Kvifte, and R. I. Godøy. Towards a gesture description interchange format. In *NIME '06: Proceedings of the 2006 International Conference on New Interfaces for Musical Expression*, pages 176–179, Paris, France, 2006. Paris: IRCAM – Centre Pompidou.

[7] E. Maestre, J. Janer, M. Blaauw, A. Pérez, and E. Guaus. Acquisition of violin instrumental gestures using a commercial EMF tracking device. In *Proceedings of the 2007 International Computer Music Conference*, Copenhagen, 2007.

[8] T. Place and T. Lossius. Jamoma: A modular standard for structuring patches in max. In *Proceedings of the 2006 International Computer Music Conference*, pages 143–146, New Orleans, 2006.

[9] N. Schnell, R. Borghesi, D. Schwarz, F. Bevilacqua, and R. Müller. FTM – complex data structures for Max. In *Proceedings of the 2005 International Computer Music Conference*, pages 9–12, Barcelona, 2005.

[10] D. Schwarz and M. Wright. Extensions and applications of the SDIF sound description interchange format. In *Proceedings of the 2000 International Computer Music Conference*, pages 481–484, Berlin, Germany, 2000. San Francisco: ICMA.

[11] M. Wright, A. Chaudhary, A. Freed, D. Wessel, X. Rodet, D. Virolle, R. Woehrmann, and X. Serra. New applications of the sound description interchange format. In *Proceedings of the 1998 International Computer Music Conference*, pages 276–279, Ann Arbor, 1998.