Towards a Gesture Description Interchange Format

Alexander R. Jensenius
Department of Musicology
University of Oslo
Pb 1017 Blindern
NO-0315 Oslo, Norway
a.r.jensenius@imv.uio.no

Tellef Kvifte
Department of Musicology
University of Oslo
Pb 1017 Blindern
NO-0315 Oslo, Norway
tellef.kvifte@imv.uio.no

Rolf Inge Godøy
Department of Musicology
University of Oslo
Pb 1017 Blindern
NO-0315 Oslo, Norway
r.i.godoy@imv.uio.no

ABSTRACT

This paper presents our need for a Gesture Description Interchange Format (GDIF) for storing, retrieving and sharing information about music-related gestures. Ideally, it should be possible to store all sorts of data from various commercial and custom made controllers, motion capture and computer vision systems, as well as results from different types of gesture analysis, in a coherent and consistent way. This would make it possible to use the information with different software, platforms and devices, and also allow for sharing data between research institutions. We present some of the data types that should be included, and discuss issues which need to be resolved.

Keywords

Gesture description, gesture analysis, standards

1. INTRODUCTION

Our current research evolves around relationships between music-related gestures (i.e. bodily movement) and musical sound. One of the approaches we have taken is to observe how people move in response to musical stimuli, what we call *sound-tracing*, *sound-sketching* or *sound-mimicking* gestures (e.g. air instrument playing [2]). As presented in more detail in [5], we see the need for a consistent way of describing music related gestures and gesture–sound relationships. In this paper we present our needs for a data format to store such information.

We typically use a number of different tools and methods in our observation studies, everything from motion capture systems, video recordings and sensor data to manual annotation. Up until now, all of these have been recorded in various programs on different platforms, and in all sorts of data formats. This is inefficient and also problematic when it comes to analyzing the material. It also leads to some challenges when it comes to synchronization, since the data is recorded with different time coding. The result is that we need to synchronize data manually, which is a very time consuming affair. Not only are these issues problematic when working within our own research group, but it also makes sharing data with other institutions difficult.

NIME 06, June 4-8, 2006, Paris, France

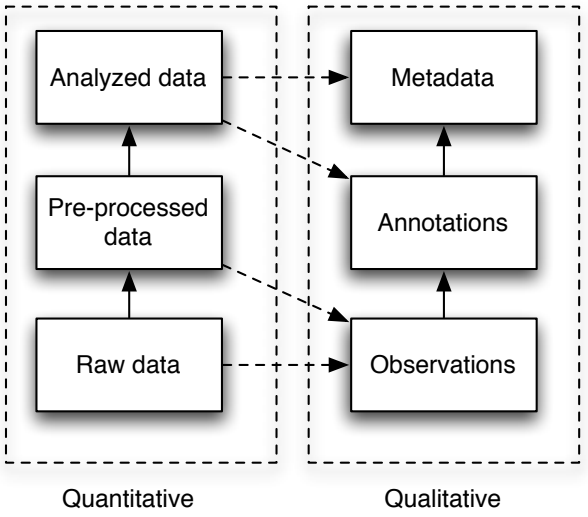

Figure 1: Different levels of data

From our experience, and after talking to colleagues from other institutions, there seems to be a need for an open standard for storing gesture-related data and analysis in a structured way. Inspired by the Sound Description Interchange Format (SDIF), developed by IRCAM and CNMAT in the late 1990s [8], and currently available in a number of audio programs and programming environments [7], we call for development of a Gesture Description Interchange Format (GDIF) to solve some of the above-mentioned issues. We believe it is important to try and use current standards as far as possible, so GDIF could possibly be implemented as an Open Sound Control (OSC) address space, which could allow for both realtime and nonrealtime applications.

This paper presents what we currently find important to include in a GDIF specification, some challenges that need to be addressed, and suggestions for how to proceed developing the standard.

2. STORING OBSERVATION DATA

An example of a typical workflow in one of our observation studies is sketched in Figure 1. The first stage is to record raw data from various sensors and devices, together with our manual observations and descriptions. This information is later pre-processed and analyzed, and forms the basis for annotations and higher level analysis. The following sections will present these different levels, and how it might be possible to store them in a consistent way.

2.1 Controller data

We often use different types of commercially available controllers for our studies. Some of these already have well-defined formats, such as MIDI-instruments, as opposed to for example HI devices[1] and Wacom tablets which typically differ for each device and model. Common for all such devices is that they have moderate sampling rates (around 100 Hz), and resolutions varying from 7-bit for MIDI-devices, 8-bit for most HI controllers and up to 16-bit for Wacom tablets.

As suggested in [9], data from a joystick could be represented with an OSC address space like:

```
/joystick/b xtilt ytilt rotation ...
```

when button 'b' is pressed. A problem with this approach, however, is that the end user will not necessarily know what the different values mean. Even though it is less efficient, we believe it could be wise to split the information into separate components. This would probably make them easier to read and parse for the end user. An example could be:

```
/joystick/button/b [1/0]
/joystick/x/position [0.-1.]
/joystick/x/rotation [0-360]
```

An important issue would be to define the units used. In general, we prefer to normalize values to a 0.-1. range, but for some data it could be more relevant to use other units. This could for example be defined after the value, like:

```
/joystick/x/rotation 270 degrees
```

Similarly, a MIDI device could be described as:

```
/keyboard/midi/[note, velocity, channel] [...]
```

And a Wacom tablet as:

```
/wacom/graphire/[x, y, pressure, xtilt, ...] [...]
```

In the last example, the manufacturer's name and device name is included. For a Wacom tablet this might be relevant information for the end user, while in other cases it might not. Rather than using such names, it could be more relevant to create names describing the quality of the device (e.g. tablet) rather than the brand name.

2.2 Motion capture data

Storing information from commercial motion capture systems should also be quite straight forward. One type of such systems (e.g. Vicon, Eagle 4) use infrared cameras to record the position of reflective markers in a space. The end result is a massive amount of x, y, z data (recorded at up to 4000 Hz) from up to 40 markers. Electromagnetic tracking systems (e.g. Polhemus Liberty/Patriot) typically have lower sampling rates (up to 240 Hz), but can also record orientation (azimuth, elevation, roll) of the markers.

Although most manufacturers of motion capture systems have their own custom built software, there seems to be some agreement on the C3D format[2] as a standard way of coding motion capture data. Since this format is so tied to motion capture systems, and only focuses on absolute positioning, it is not suitable for our needs. It would be better to also code such information into GDIF, for example like:

```
/polhemus/patriot/[x, y, z, azimuth,
                   elevation, roll] [...]
```

2.3 Sensor data

Commercially available controllers and motion capture systems usually have some standardized values that can easily be converted to GDIF messages, but it is more difficult when it comes to data from custom built controllers. They are typically built with all sorts of sensors and the output is also highly dependent on the sensor interface used. Popular interfaces such as iCubeX, Phidgets, Teabox and Kroonde work at all sorts of sampling rates (10 Hz to 4000 Hz), resolutions (7-bit to 16-bit) and ranges (e.g. 0.-1., 0-127, 0-65536), which make it difficult to come up with a general system for how to store the data. For such devices it might be more interesting to store preprocessed information directly, scaled and normalized to for example a 0.-1. range, and grouped according to sensor type, for example:

```
/sensor/accelerometer/[x y z] [...]
```

For custom built controllers it would probably be wise to store information in the header of the GDIF file about how it was built, which sensors and sensor interfaces were used, etc., so it can be possible reproduce and verify the results. The structure of such descriptions, and the level of detail needed, could probably be left open to the end user, but it would be good to provide some examples of good practice.

2.4 Other information

As well as the numerical information mentioned above, we typically also need to store general descriptions about the recording sessions, with information about date, researchers involved in the recording, the subjects, etc. Such information can be written to the header in a file, so that it is easy to get an overview of the data in the file by investigating the header.

Another important issue is to be able to store information about sound and video recordings made in parallel to controller and sensor data. We usually record video and audio in observation sessions, both as raw material for analysis, and as a reference. Working with DV-cameras, this has posed some serious challenges in terms of synchronizing the recordings with sensor data.

To overcome some of these issues, we have developed a set of tools[3] for recording audio, video and gesture data in one program. The tools are based on the Jamoma[4] modular standard for Max/MSP/Jitter, using OSC for all messaging. This allows for easily sending and receiving data from networked computers and devices, and makes it possible to synchronize all the data. In our current setup, all data is stored on one computer, which is a bottleneck when we need to record video from multiple cameras at the same time as recording high density data coming from a motion capture system. We are currently developing a system which can store synchronized information on several computers.

Since our studies often involve analysing movements in relation to prerecorded music, we need a way to store and synchronize original sound files, next to sound recorded during observation sessions. It would also be interesting to have a way of storing higher-level musical descriptors, such

[1] Human interface devices, e.g. game controllers.
[2] www.c3d.org
[3] Available from http://musicalgestures.uio.no
[4] http://www.jamoma.org

that it could be possible to retrieve structural elements in performances.

Finally, during observation sessions we typically also want to store some descriptions about new sections, interesting or unusual things happening, etc. Such invaluable information could come in any form, anything from time markers to text comments, and should also be time-stamped and stored alongside the other data.

3. STORING GESTURE ANALYSIS

Besides storing data, information and descriptions from the recording sessions, we are also interested in storing results from analysis of the data. Here the challenge is not so much on the data processing side, but more on the analytical, as we are still developing methods and tools for analysing musical gestures.

As discussed in more detail in [5], we find it important to differentiate between different *analytical perspectives* (Figure 2). For example, we find that it is a big difference whether the movement is observed with respect to the performer, the listener, or the controller/instrument. Analysing the movements of a pianist, there is a big difference whether the gestures are studied from the pianist's or from the audience's perspective. A performer usually starts to think about, and carry out, a gesture long before it is actually seen from an audience. For the performer, the *goal point* (e.g. key on the piano) of the movement is crucial, as well as the internal feeling of the movement, while for the average listener the larger shape of the gesture is probably more important, and probably also the only thing that is actually seen at a distance from the stage.

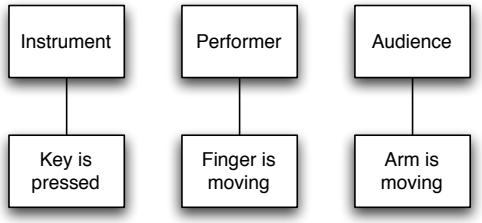

Figure 2: Three different analytical perspectives.

Another important thing to consider in the analysis, is which *attention level* or *analytical resolution* to use. As humans we tend to process information at several different levels simultaneously, and this is also important for how we process information. In terms of a piano performance, we may choose to focus on both small and large gestures at the same time. We also typically operate at several different time scales, looking at rapid attacks of hands and fingers, but also how the upper body is moving over time. The same should be the case in our analysis, so it is important that we have a tool where it is possible to have multiple analytical streams, and different analytical levels next to each other (Figure 3).

In the following sections we will present some of the different analytical methods we employ and how it could be possible to store the results in a GDIF standard.

3.1 Biomechanical analysis

Biomechanical analysis typically focuses on quantitative aspects of human motion, such as velocity and acceleration curves from various joints and limbs. For such analysis, the trick is to get good recordings, and then the analysis is a matter of calculating the relevant information. Since the information is numeric and easily labeled, such information is also easily stored.

We still need to figure out how to better represent the information, for example whether it should focus on body parts:

```
/arm/right/[velocity, acceleration, direction] [...]
```

or be grouped according to kinematic quality:

```
/velocity/arm/[right, left] [...]
```

What to choose often depends on the main focus of the analysis, and what is more practical when the information should be parsed and used in other systems.

3.2 Laban movement analysis

On a more qualitative side, we are interested in storing Laban movement analysis (LMA) focusing on observations of the inner qualities of movement [3]. LMA consists of the descriptors *body*, *effort*, *shape* and *space*, of which all can be broken down to separate elements. For example, the four effort elements are *weight*, *space*, *time* and *flow*, and each of these elements are defined in terms of the following axes:

Weight light — strong

Space direct — indirect

Time sustained — sudden

Flow free — bound

These axes may be a good starting point for creating a numerical way of storing Laban information, since each of the pairs could be defined by a 0.-1. range. Thus it would be possible to store subjective Laban data numerically by for example adjusting a slider between the two extremes when carrying out the analysis.

3.3 Other types of analysis

What we find particularly important in our analyses of gesture–sound relationships, is that of separating and comparing gestures to *musical objects* [6]. Segmentation of movements and sounds can be derived quantitatively, but we typically do segmentation manually, since it is also often several ways of doing this [2]. Again it is important to be able to store multiple layers of information, since segmentation can typically occur at different levels (as sketched in Figure 3).

We are also interested in trying to formalize a way of describing gesture primitives such as *trajectory*, *force* and *pattern*, described in [1], as well as organological parameters related to instrument control and performance, such as presented in [4].

4. TOWARDS GDIF

These are some of the most important criteria when it comes to development of GDIF:

Open so that everyone can use it for any gesture-related data storage.

Human-readable so that it is possible to easily understand what the data means.

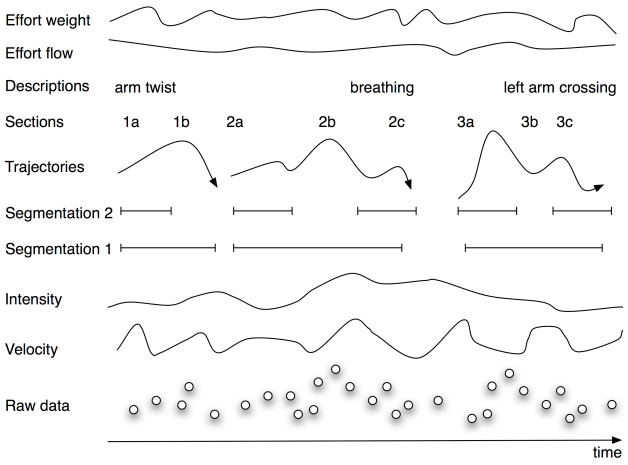

Figure 3: Sketch of gesture content that could be stored in a GDIF file.

Multiplatform so that it will work on any computer platform and software.

Flexible so that it can allow for multiple layers of analysis, dependent on the analytical level or different researcher's opinions.

Extendable so that it is possible to add more descriptors and content when it is necessary.

Simple so that it is easy to get started. This means that there should only be a very limited basic requirement for the data format, probably only a header with some descriptors, while everything else could be decided by the user. This will also make it easier to implement in different programs.

Efficiency is not the most important, as we believe well documented and easy to read codings are more valuable for our research. That said, it is always good trying to be as efficient as possible.

We have currently started to develop GDIF as an OSC address space, and storing the data streams with time tags, but it might be possible that an XML approach might be the way to go.

5. CONCLUSION

The paper has presented some of our current needs when it comes to storing and sharing information about music-related gestures. We suggest the development of a Gesture Description Interchange Format (GDIF), as an open and flexible format for storing gesture-related data and analysis. However, it should not be so open that it serves no purpose, so a number of data types and a standard way of describing certain devices and analytical methods should be required. Ideally, such a standard would allow for sharing information between different software, platforms and research institutions.

In addition to the challenges presented in the paper, we see a number of unresolved issues which will need to be addressed in the future development:

- What type of synchronization is better?
- What type of time-base should be used?
- What precision and resolution is necessary?
- Which audio and video format(s) and compression standard(s) should be used?
- How can chunks of information be represented at various levels?
- Should the information be stored progressively so it would be possible to change the attention level when analysing the material?

Currently, researchers at the University of Oslo and McGill University are involved in development of GDIF, and others are more than welcome to join. The next step is to create a draft specification, along with a number of more detailed example files and software, which can be presented at a relevant conference in the near future.

6. REFERENCES

[1] I. Choi. Gestural primitives and the context for computational processing in an interactive performance system. In M. M. Wanderley and M. Battier, editors, *Trends in Gestural Control of Music*, pages 139–172. Paris: IRCAM - Centre Pompidou, 2000.

[2] 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, *Gesture in Human-Computer Interaction and Simulation: 6th International Gesture Workshop, GW 2005, Berder Island, France, May 18-20, 2005, Revised Selected Papers*, volume 3881/2006, pages 256–267. Springer-Verlag GmbH, 2006.

[3] P. Hackney. *Making Connections: Total Body Integration Through Bartenieff Fundamentals*. New York: Routledge, 2000.

[4] T. Kvifte. *Instruments and the Electronic Age. Towards a Terminology for a Unified Description of Playing Techniques*. Oslo: Solum Forlag, 1989.

[5] T. Kvifte and A. R. Jensenius. Towards a coherent terminology and model of instrument description and design. *NIME06, Paris*, Forthcoming.

[6] P. Schaeffer. *Traité des objets musicaux*. Editions du Seuil, 1966.

[7] D. Schwarz and M. Wright. Extensions and applications of the sdif sound description interchange format. In *Proceedings of the International Computer Music Conference, Berlin, Germany*, pages 481–484, 2000.

[8] 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 International Computer Music Conference, Ann Arbor, Michigan*, pages 276–279, 1998.

[9] M. Wright, A. Freed, A. Lee, T. Madden, and A. Momeni. Managing complexity with explicit mapping of gestures to sound control with osc. In *roceedings of the 2001 International Computer Music Conference, Habana, Cuba*, pages 314–317, 2001.