Tracking conductor of an orchestra using artificial neural networks

Tommi Ilmonen

13.4.1999

Contents

Chapter 1

Introduction

Live performance using a combination of musicians and electronics has been a problematic art. One big problem is to make a performance where musicians and electronics create coherent sound. One simple solution is to prepare the electronics in advance. In this case, musicians will adjust to the sound coming from tape or a sequencer. As a result, the musicians are left with little room for new interpretations. Another possibility is to prepare a sequence of music (MIDI file) and a have real person to take care of synchronizing the sequence with the musicians. This is problematic when one should make the sequence react fast to the music. Controlling the expression parameters other than just tempo and feeding them to computer quickly is a difficult problem. One can of course play the electronics in real-time, but this may be impractical with many sound generation/processing techniques. Mastering the wide range of electronics (synthesizers, sound/visual effects) is difficult. Some of the user interfaces are unsuited for real-time control.

Automatic conductor following is one approach to solve the problem. As condictor is the central source of control, it is a natural idea to follow him/her in order to extract musical parameters (figure 1.2). Computers can be made to analyze the movements of the conductor and react to them in real-time. With such a system one can also conduct a virtual band which is useful for conductor training.

Figure 1.2 displays the communication between musicians and conductor. Mimicking this interaction is the goal of the conductor follower. piece.

Conductor following poses many problems inherent in modeling human-human interaction using a computer. In this case, one must recognize gestures of the conductor. These will always be slightly different and sometimes completely new. Still a human will adapt to them quickly. One needs to react to these gestures musically. How to express musical parameters (such as "violently") with synthesizers is a problem. The designer of the conductor follower cannot dictate the gestures to be used, but must work with gestures that are already in use. These gestures can be badly suited for

Figure 1.1: Conductor in data dress suit.

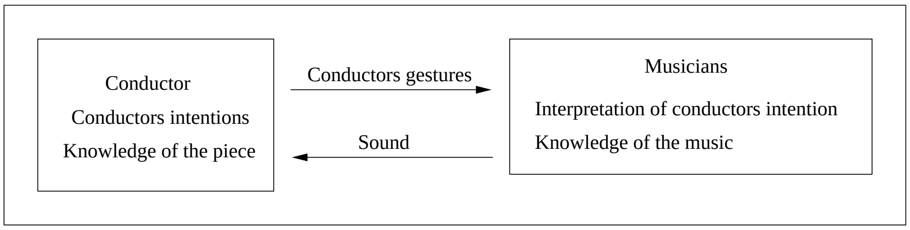

Figure 1.2: Primary communication methods between conductor and musicians in concert situation. Conductor and musicians form a feedback system.

computational analysis.

Little work has been reported on this field. Played music has been followed ((Coda, 1998) and (Toiviainen, 1997)) as well as conductor's movements ((Brecht & Garnett, 1995), (Carosi & Bertini, 1992) (Mathews, 1991), (Tobey & Fujinaga, 1996) and (Keane & Gross, 1989)). Toiviainen has studied competing oscillators to model the perception of rhythm ((1997) and (1998)). His application is a real-time accompaniment that responds to MIDI note-on commands coming from keyboard. A commercial application Smart Music tracks tempo by analyzing the sounds a player plays and comparing them to a score (Coda, 1998). In these cases the aim is to synchronize playback of a sequence with a performer that is playing part of the music.

A similar target is shared by all conductor following projects: follow the conductor in real-time and react musically. As input data they use conductor motion. Mathews' (1991) approach has been to track rhythm by simply observing the up- and down-beats of a conductor. More sophisticated work has been presented by Brecht (1995). Artificial neural networks (ANNs) were used on an experimental basis to track the rhythm continuously. Tobey (1996) has reportedly built a comprehensive system, but little information on it is available.

Usa has made the most elaborate system so far (1998). As a novelty the system includes eye-tracking mechanism.

Baton has also been used in less traditional way. Marrin has presented a "Digital Baton" which is used to conduct a computer-generated musical score (1996). What makes this work different from others is that the tempo of the piece is not affected by the conductor and the control gestures differ from traditional music conducting.

In this paper, methods for conductor following are presented and discussed. The implemented following system is described in detail. Our system uses multiple methods for rhythm tracking. 3D motion tracking is used to obtain sufficient data of the conductors movements. Neural networks are used when accurate, continuous following is wanted. An oscillator method resembling Toiviainen (1997) is applied when less accurate following appropriate. A heuristic algorithm is used to construct the final music output. Music is played with standard synthesizer.

Our long-term research interest is to study the use of artificial intelligence in environments where temporal information and implicit knowledge are important. Music is a fine example of such.

Applications of these techniques are live performance, conductor training and entertainment.

1.1 The idealized follower

In the perfect world the follower will behave, sound and look like a real player in a concert hall. Hopefully even better. It should naturally adapt to the conductor like humans do.

There are three tasks that a conductor follower must perform.

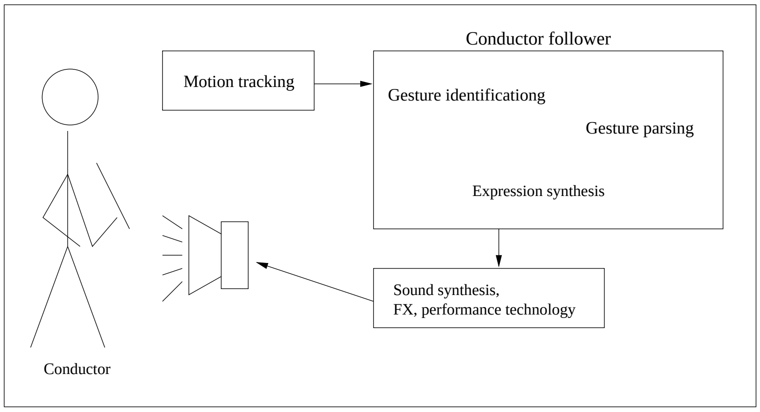

Figure 1.3: Components of the ideal follower.

To begin with - the gestures of the conductor must be identified. Then one must interpret what these gestures mean in the context of the piece and what aspects of the performance should be affected. Finally all available information must be used in real-time in a musical way.

These are not isolated tasks. The knowledge of the piece can be useful in gesture identification and in the playing of music. On the other hand, the nuancing of the piece may be changed based on the analysis of the gestures.

The three parts are presented as separate entities only for conceptual clarity. An implementation of a conductor follower does not have to consists of three separate parts.

In current conductor follower projects an explicit player model is usually omitted. In its place is a heuristic system for playing music.

Performing these tasks requires utilization of many information technology methods. Handling the conductor motion data requires signal processing tools. Pattern recognition techniques are needed to identify gestures. One needs artificial intelligence tools that can imitate a musician playing a piece.

1.1.1 Gesture identification

The gestures of the conductor vary a lot. Tempo is shown with one hand that is constantly moving, the other hand gives signs and the stance may imply emotion.

Theoretically one could construct an rule-based system that detects every single gesture. It could be an expert system that imitates a methodology of a known expert. Partially the needed information is available in textbooks (eg. (McElheran, 1989)) and could serve as the basis of an expert system. Experts systems can be analyzed and understood better than ANNs or statistical methods. This can be valuable eg. for musicological studies.

Alternatives to an expert system are ANNs and statistical methods. These methods use raw numeric data to find mappings from motion data to gestures. They need preprocessing of the conductor motion data. The preprocessing makes detecting certain types of gestures easier. At the same time the preprocessing limits the kind of gestures that can be detected. Besides one must know what are the gestures to be recognized.

One can argue whether explicit gesture identification is a valid approach. The large number of different emotional states can be difficult to handle. How does one characterize emotional features embedded in gestures ? Is it a natural approach to seek for separate gestures of happiness, jolliness, sleepiness, weariness, anger, laziness and all other feelings ? Even describing such feelings is difficult and results in vague definitions. An alternative would be an algorithm that creates an implicit holistic description of the emotions. We do not need an explicit representation of emotions. Appropriate reaction and behaviour are sufficient.

The follower cannot be expected to have full representation of human movements. Figure 1.4 shows the way a conductor follower could see the conductor. Some parts of the conductor may not be visible to the follower at all. As a result, details of motion are easily lost (finger positions, facial expression).

1.1.2 Meaning of the gestures

The gestures alone do not tell how to play. Musical knowledge of the piece is necessary eg. tempo changes, fermatas and accents. There is also other, general musical knowledge that tells how music is actually played. The identified gestures must be matched to the knowledge and possibly the knowledge should be adjusted. The end result is an assumption of what conductor wants to express.

How does one extract information from gestures using prior knowledge ? How should the knowledge and the information be presented ?

Where to get knowledge of the piece ? To begin with, one should study the score as much as possible. The score contains the notes and the performance instructions (markers, dynamics and tempo values). It tells a musician a great deal about how conductors movements relate to the music. Translating the information from score to computationally useful temporal and hierarchical data structures is an extra task. In training situations musicians learn how the piece should be played. This learning process should be simulated in ideal follower.

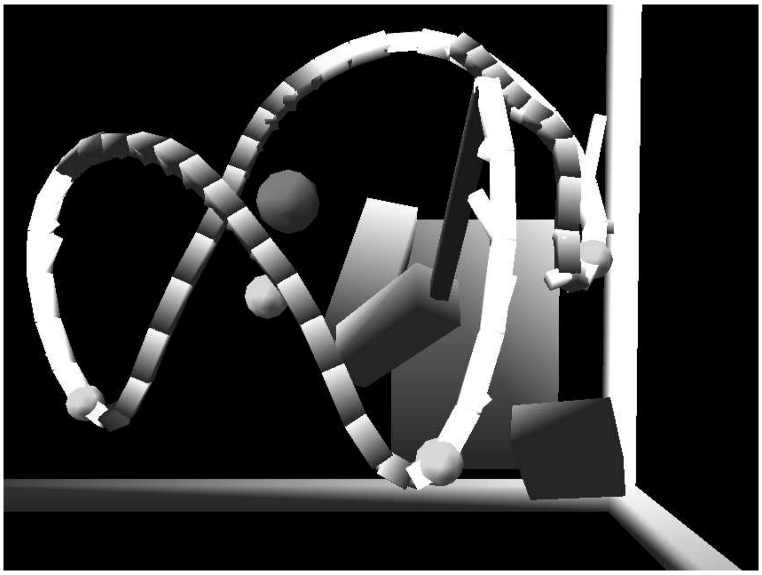

Figure 1.4: 3D visualization snapshot. The continuous curve represents baton tip trajectory. Other objects represent the conductor and indicate performance parameters.

1.1.3 Expression synthesis

If the conductor wants violent playing, this is what a musician should do. Mimicking a musician playing a piece is a little-studied problem: How does a computer program know how to play in different styles ? How to fine tune the volume of individual notes to get musical results ? How about the other nuances ? What exactly is the timing in accelerando ?

Simply scaling playback tempo and volume up and down is hardly a musical solution. A player model would be the proper tool. A model of a musician - like a real layer - must take many things into account: The things conductor is expressing, the musical knowledge, the actions of other musicians, the way he is self playing and the physical movements he is performing. Multiple player models are needed to represent the different musicians in an orchestra.

Parncutt (1997) has hypothesized a player model for the piano and Chafe (1992) has studied the differences between two performances of the same piece. Friberg and Bresin have made rule-based system to automatically make playback of MIDI-sequences more expressive ((Friberg, 1995), (Friberg & Bresin, 1997) and (Bresin & Friberg, 1998)). Similar work has been carried out by Canazza (1998).

Expression synthesis is the reverse operation of gesture identification. From parameters describing the style and emotion of performance one should be able to produce audible results.

1.1.4 More than reaction - the environment

Performance is more than sound. There are also musicians, an coustic space and the audience. Conducting a pair of loudspeakers hardly gives a feeling of being in front of an orchestra. An animated orchestra would make the situation feel more realistic.

More realistic sound also makes the environment more life-like. Physically modeled musical instruments and simulated concert hall acoustics are the tools for this ((Välimäki & Takala, 1996) and (Takala T. & M., 1996)).

Whether realism and immersiveness are good goals depends on the application. In training situation realism is desirable. In cyber-performance things are very different. One can imagine virtual musicians mutated to abstract shapes playing imaginary instruments.

Our conductor follower is combined with band animation and sound auralization to provide immersive environment (Appendix A).

1.2 Related work

1.2.1 Conductor following

The first conductor follower was published in the eighties by Max Mathews - the Radio Baton (Mathews, 1991). The system relies solely on down-beats for determining the tempo. It was very simple and posed strict limitations on how the conductor could move.

Brecht used ANN (multi-layer perceptron) to detect the motion phase and to predict next down-beat (Brecht & Garnett, 1995). This allows the system to react more swiftly to changes in the tempo.

Tobey has reportedly built a comprehensive system, but no information of the methods used is available (1996). Usa has reported a big system that can even detect and use eye direction information (Usa & Yasunori, 1998). That system resembles the follower described here as far as functionality is concerned. However, details are different - hidden Markov models instead of ANNs.

General discussion of how a conductor follower should work and the problems one may encounter can be found in Morita's work (1990).

1.2.2 Other musical performance followers

Following a musician playing a piece is a problem similar to conductor following. This has been studied by Toiviainen whose system can be used as an auto-accompaniment for a keyboardist (1998). As the system depends on MIDI it cannot be used with acoustic instruments (at least not directly). The commercial system SmartMusic can track a musician playing a piece and even reportedly switch the key automatically (Coda, 1998).

Marrin's masters thesis reports a very different performance interface (Marrin, 1996). The conductor/musician/controller uses a baton to control volume parameters of multi-channel sound-files. It lacks a way control the tempo of the piece. In its place, the volume of the sound can be controlled.

1.2.3 Research fields with similar problems

Many problems encountered in conductor following can be generalized. Pattern recognition in temporal environment is the problem of speech recognition, hand-written text recognition and general gesture recognition.

There are several annual pattern recognition conferences, for example *IEEE Conf. on Computer Vision and Pattern Recognition* and *IEEE International Conference on Pattern Recognition*. Browsing thru the proceedings of conferences one can find a number of articles on gesture recognition (for example (Watanabe T., 1996) and (Bregler, 1997)).

1.3 Project history

The conductor follower project was started in June 1996. I began my work at that time. Another student was also employed to work on conductor following, Jarmo Hiipakka. This work has been done in the Tacit knowledge research project.

The first version of the follower became functional around Christmas 1996. This formed the base of the system that is used today. The follower is a part of the DIVA system (section A) which has had contributions from many research projects (Tacit knowledge, NETVE PAULA).

The system had been found useful, but it had a number of limitations. Realistic conducting may lead to highly unrealistic output. Nevertheless, one could learn to conduct the system easily.

The old system is partially described in the author's paper (1998). More thorough description has been submitted as a report for Information Science, Special Assignment I (Tik-61.195) in HUT.

This document describes some work that was done earlier. Magnetic motion tracking mechanism (chapter 3) had been coded in 1996. The tempo tracking methods in sections 4.1.2 - 4.1.5 were also available in the beginning. Rest of the work is original and done by me during 6.1998-1.1999.

1.4 The original plan

1.4.1 Planned improvements

New features were needed and old features were to be improved. The old follower relied on getting information from the vertical z-axis. This needed improvement. The methods were redesigned to take advantage two dimensions (horizontal and ertical). A method to track beat patterns was also needed. Tracking beat pattern makes the system more reliable. New control gestures were needed. Besides tempo and dynamics one would also like to control nuancing (staccato, legato etc.). Additionally the follower should be able to produce more expressive music, taking advantage of the gesture analysis results.

Of these goals 2D-following and beat pattern tracking goals were well met. I wish the system had advanced more in other fields. The system does not detect emotion as well as intended and the musical expression mechanism is not as good as desired.

1.4.2 Schedule

Goal was to get text ready in january 1999 with all research goals studied at that time. The paper was nearly finished in february 1999 and finally finished in march after being commented by Tapio Takala. The submission date slipped to March event tough the system was built quite close to the intended timetable.

1.5 System architecture

In this sections the system architecture is quickly overviewed. A block diagram of the system is in figure 1.5. Structure resembles the hypothetical ideal follower found in figure 1.3. A Programming-oriented diagram can be found in figure 9.1.

1.5.1 Components

Software-wise the structure of the follower implements the components outlined in section 1.1.

Motion capture and filtering is performed with specific motion tracking hardware. The motion data is filtered with custom algorithms (chapter 3).

The interesting part is the system that analyzes the motion for gestures, forms an estimation of intention based on the gestures and synthesizes musical output (relates to sections 1.1.1 - 1.1.3). While division to three segments (analysis, parsing, synthesis) is a conceptual division it helps keep the final C++ sources manageable.

1.5.2 Gesture analysis

There are multiple gesture analyzers. Sometimes a particular movement may be assigned its own analyzer (chapter 5). Some gestures are embedded in

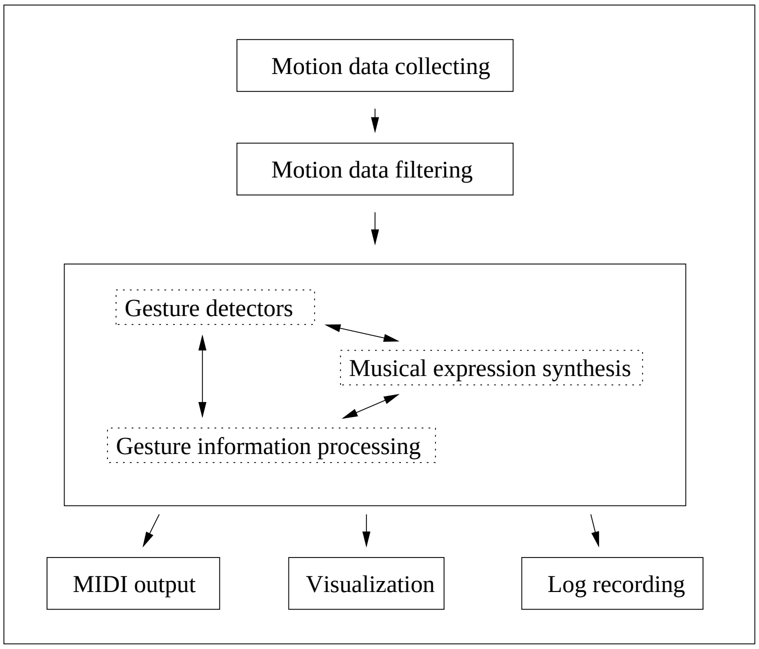

Figure 1.5: Logical structure of the conductor follower.

the motion and generic motion parameters are calculated to gain knowledge of them (chapter 6).

The gesture detection and analysis methods are described in chapters 4 - 6.

Beat pattern tracking (section 4.4) is the most prominent example of matching gestures with knowledge of the piece. Other usages also exist, as in the case of fermata processing (section 5.3).

1.5.3 Player modeling

Different approaches to player modeling have been used. So far all of these have been hard-coded algorithms with few parameters for fine-tuning the musical response.

Chapter 4 describes the various methods that can be used to track the rhythm of the piece. Additional tricks that can be done are in chapter 7.

Chapter 2

Gestures of the conductor

In concert the conductor communicates to orchestra with his movements. These movements contain the information that is needed by musicians playing the piece.

The rules that govern how conductor should conduct are flexible and can be broken. It is notable that musicians are not always aware of how the conductor should move or what are the gestures they use. This does not prevent successful communication.

There is also variance between conductors. A great part of the communication can be done in very individual way that is difficult to describe with a set of fixed rules. The bottom line is after all not that the movement should follow some standard. The point is to deliver the artistic message.

This chapter describes the basics of conducting gestures.

2.1 Showing tempo

As a rule of a thumb the right hand of the conductor (given right-handed conductor) shows the tempo of the piece. The movement is essentially 2 dimensional. The vertical z-axis shows almost alone the tempo and movement along the horizontal x-axis gives musicians information related to the time signature. This is an idealized case. In practice conductors may conduct in a way where z-axis alone does not give information of the tempo. In these cases the tempo can be indicated by x-axis movement.

The musically "strong" bar parts are expected to be in the bottom of the beat (down beat, vertical minimum) while the top of beat implies weak bar part. What are the strong bar parts is something that may change, but musicians should know this before playing can start. Without this information players may get badly lost - conductor following can be difficult even for a real person.

In figure 2.1 are the standard patterns for time signatures 3/4 and 4/4 (McElheran, 1989). By observing the conductor's hand position and direction of movement musician should be able to determine which part of the bar is meant. In comparison - a measured baton tip trajectory for 4/4 can be found in figure 2.2.

Figure 2.3 shows the vertical movement of the baton tip. The data is the same as in figure 2.2.

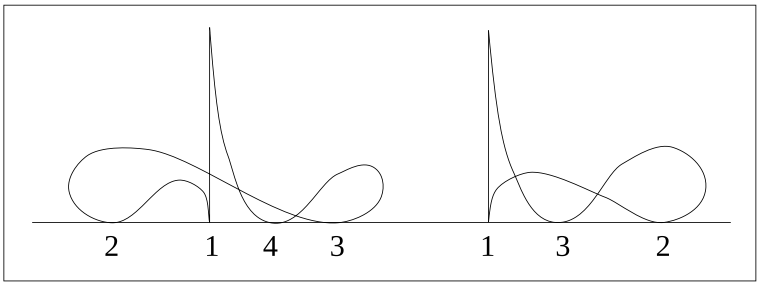

Figure 2.1: Conductor's view of beat patterns for time signatures 4/4 and 3/4.

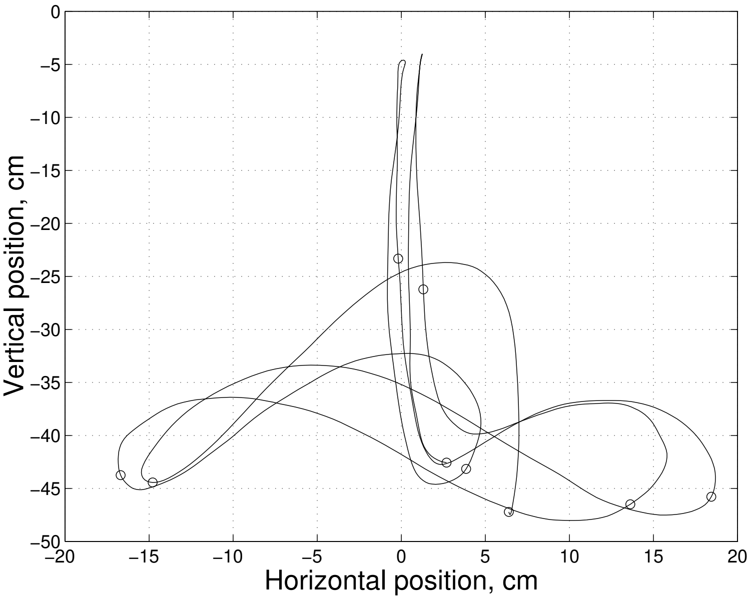

Figure 2.2: Measured baton tip movement for two bars of 4/4. Conducted very carefully by conductor Hannu Norjanen. The little circles indicate the point of curve where musical beats are supposed to be, see figure 2.3

Close analysis of conductor motion shows that the minima of z-axis do not in fact always match exactly with the beats of music. Especially in faster tempos the musical beat corresponding to the last stroke often appears slightly after baton reaches the bottom. This is illustrated in figure 2.3.

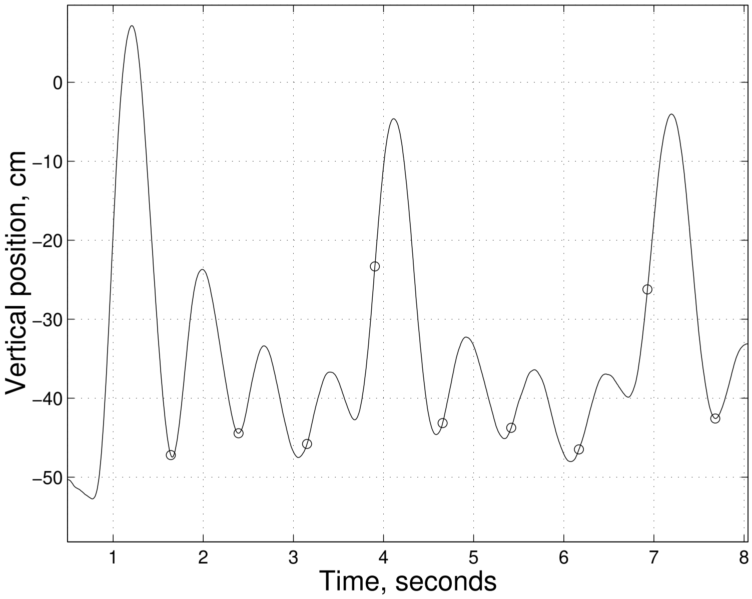

Figure 2.3: Measured baton tip movement. Little circles represent even division of beats for two bars. The first and last point have been selected manually and the others are interpolated directly.

Musicians do not usually even realize details like this as they play. Nor should they, since the conductor does not try to express tempo changes in the last parts of the bar.

Another common feature is that conductors don't do vertical movement all the time, but leave the horizontal motion to do the work when possible. In figure 2.1 for rhythm 4/4 this could be done for all beats except the first.

2.2 Time signature and rhythmic patterns

In figure 2.3 all strokes were (nearly) equally spaced in time. Uneven division is also possible. For example 5/8 can be divided as 2/8 + 3/8 or 3/8 + 2/8. The first means that down-beat match times 0/8 and 2/8. In the second one the the times are 0/8 and 3/8. In figure 2.4 there is an example of how baton tip has moved (z-axis) when conducting 5/8 divided to 2/8 + 3/8.

The conductor may also divide a bar to more beats than anticipated or leave beats out without a warning. This causes unpredictable changes in the beat pattern. For example if the time signature is specified as 4/4 with four beats per bar one could suddenly divide the second and thrid beat into 1/8 + 1/8 while otherwise conducting simple 1/4's (figure 2.5). This raises the problem of whether the conductor wants to show accelerating tempo or a subdivision of a beat. A musician should know the right answer based on

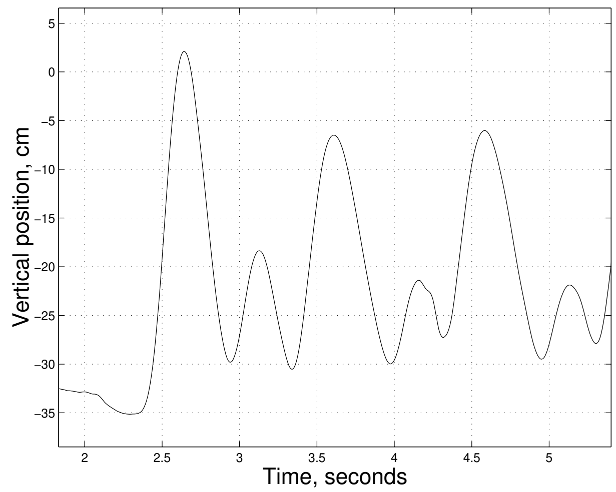

Figure 2.4: Time signature 5/8 divided to 2/8 + 3/8.

prior musical knowledge and the form of motion.

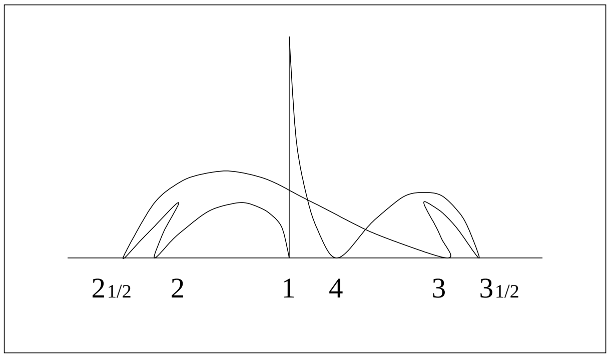

Figure 2.5: Subdividing the second and third beat in time signature 4/4.

Morphed patterns are another consideration. Conductor may shift from one beat pattern to another using intermediate forms. The time signature may stay the same, but the beat pattern slowly changes as in figure 2.6.

2.3 Start and end of piece

A preliminary beat is shown to indicate beginning of piece. The conductor may stand and move for a while before he wants to start the music. An example of the beginning of a piece is in figure 2.7.

At the end of piece there can be a counterpart to the beginning stroke, the ending gesture. Often a particular ending stroke is used, but there are other ways to finish the piece. Squeezing hand to a fist as well as a violent

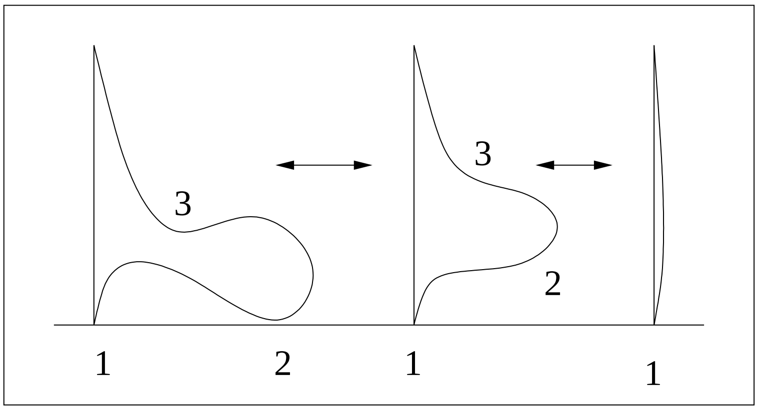

Figure 2.6: Morphing patterns. In this example time signature can be 3/4 in all cases, but the number of strokes per bar changes.

stroke can be the sign to end the final note. In fact anything that looks final can be used to end a piece. When the piece should end prematurely many ending strokes as well as some yelling may be necessary to stop the orchestra.

2.4 Dynamics

Conductors movements may also portray the volume of the music to the players. The sheer size of the movements may tell how to play (altough it does not necessarily correlate with loudness) and the left hand has specific gestures that can be used. By holding hand with open palm up the conductor orders the band to play louder. As the opposite he may also make the band play softer by keeping left hand palm downwards. Confusingly, the conductor may do a number of other things with the left hand as well.

2.5 Other gestures

Host of other things can also be implied by hand and body movements, facial expression, eye movement etc. These include - for example - staccato (shortened notes), legato (notes glued together) and fermata (extended single note duration) and emotional nuances ("lets play in violent fashion..."). Typically there is no particular gesture that would indicate these nuances - the information is embedded in the motion.

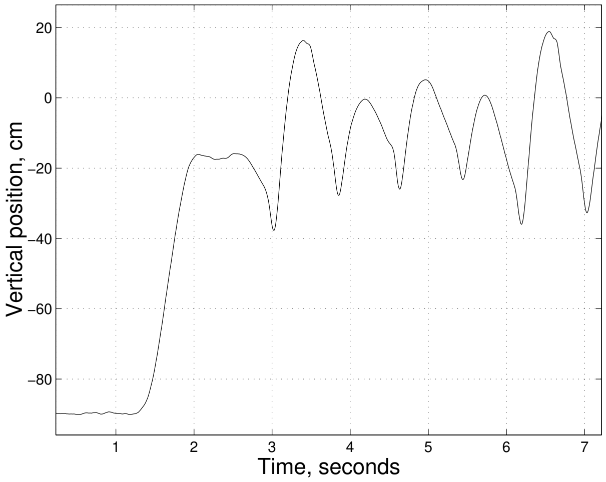

Figure 2.7: Movements before the music starts. In the beginning the hand is lowered. Then baton is raised to "ready" position and a preliminary beat is shown at 3.0 seconds. The piece is began with the first beat in time 3.8 seconds. The follower should start operation before the first beat to be able to play the first beat in time in time.

2.6 Who should react to gestures ?

The conductor may wish to communicate with the whole orchestra or just one part of it, at times even with individual musicians. The scope of communication may be determined by directly pointing or looking a person. More implicit methods can also be used like in any human communication.

2.7 Limitations

Sometimes using all gestures can be risky. It is expected that the musicians know their notes and do not really need the extra information supplied. Besides, the information can be misleading. This is the case when conductor chooses to conduct non-staccato passages in staccato fashion to get more precision. Wide movements are typically asociated with strong playing, but quiet passages can also be conducted with wide movements. In such situations the musicians would still understand what to play, although the conductors gestures do not portray the intention one to one. On the contrary, often musicians claim they tend to get lazy about playing nuances if the conductor does not show them with his movements.

Many conductors make dramatic gestures in concert situation not to get the artistic message to the musicians, but to the watching audience.

Same motion may indicate different purpose depending on the context Example in figure 2.8 shows a situation with three choices 1) slowing tempo

2) fermata 3) changing beat pattern. The musician knows the piece and the conductor. Conductor has communicated (hopefully!) his intentions to the band in advance and the musician will play correctly.

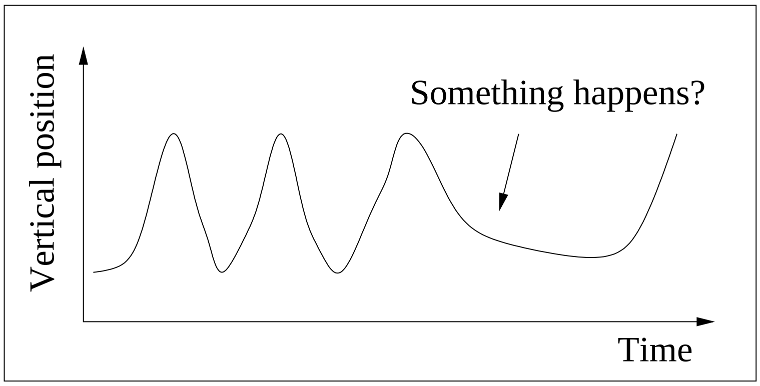

Figure 2.8: With no contextual information this gesture is ill-defined.

Chapter 3

Motion capture

Before conductors movements can be analyzed they must be captured. In ideal case one would get full 3D representation of the conductor, including limb movements, finger movements and facial expression. Such tracking system would be difficult to build and expensive with current technology. Nevertheless, Usa has used eye tracking to distinguish between areas of greater and lesser importance (1998). An interesting extension to motion tracking is biotracking - tracking the biological parameters of the conductor. Tool for EMG-tracking has been designed and implemented in Marrin's later work (Marrin Teresa, 1998). The system she built was effectively a jacket that gathered EMG data from hands and body of the conductor.

Hand movements are supposed to give most of the information to musicians, but facial expression and finger movement are also important. Not all musicians will be able to see the face of the conductor well. For example a person in the rear of the orchestra with poor sight won't be able to see the conductors face. Thus body and hand movements must give great part of the expressive information. Tracking them should be sufficient for good conductor following.

Numerous motion tracking tools have been used in conductor following. Used methods include accelerometers, ultrasonics, cameras, magnetic trackers and radio frequency capacitance measurement tools.

For practical conductor motion tracking cameras that track the baton have been used ((Carosi & Bertini, 1992) (Tobey & Fujinaga, 1996)). Another approach has been to use IR cameras with small IR emitter attached to the baton (Brecht & Garnett, 1995). Usa uses accelerometers (1998). These provide information (1 to 3 dimensional depending on the implementation) on how the baton is moving, but don't give any information of other movements of the conductor. Besides, the baton must be visible to the camera at all times.

Tracking the conductors limbs and not just the baton gives more information. One can use the body and left hand movements in the analysis.

3.1 Magnetic tracker

We have used magnetic tracker as it is a straightforward tool and does not limit the way in which the conductor can move.

The motion tracking system (Ascension Motionstar (Ascension, 1999)) consists of a single magnetic pulse generator and multiple small sensors. The sensors provide six dimensional data, 3 position coordinates + 3 rotation angles. Wires connect the sensors to the computer which sends data to clients by local Ethernet using TCP/IP or UDP/IP.

Limitations are that sensors should one to three metres fromthe magnetic pulse generator. Any magnetically active materials or devices can cause noise and distortion in the measurement.

3.2 Sensors' positions

The sensors are placed to the points a musician would look at when playing in a band. The most important sensor is the one in conductors baton hand. That sensor is either mounted directly to the baton (so one can calculate the baton tips position) or to the hand or glove. Thus conductor can conduct with or without a baton. The baton with sensor is not as comfortable as normal baton, but so far conductors have not found it a problem.

Other places where the sensors are installed are left hand (using a glove) and finally just behind conductors neck. That sensor is used as a reference point, since we need to know where the conductor is facing and the relative positions of the limbs.

As can be noticed there are only three active sensors. So far there has been no real need for more thorough measurement. Using the data from these three sensors is difficult enough.

3.3 Sensors' mount mechanism

The sensors must be mounted to the conductor. One has to be careful that the sensors won't disturb the conductor or limit his movements. The goal is to be as non-intrusive as possible.

Our approach was to use a data jacket that can be worn like a normal garment. The wires of the sensors come to the jacket in one group. The group is attached to the conductors back since that part of body is not likely to move fast during conducting.

Wires have support in the jacket so they don't have to hang in the air for long distances (like 50 cm). Support is needed since the conducting movements can be aggressive and any slack wires are disturbing.

Sensors are attached with adhesive strips to baton and jacket to make system maintenance quick. The mount mechanism is illustrated in figure

3.1.

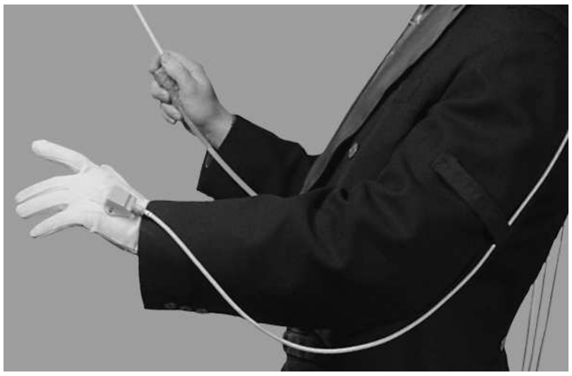

Figure 3.1: Hands of the conductor.

3.4 Sampling rate

Motion sampling rate must be high enough to prevent motion (signal) aliasing. We use sampling rates in range of 25 - 35 Hz. To test the difference between sampling rates we first down-sampled data (violent staccato movements) from 50 Hz to 25 Hz. Then the data was up-sampled back to 50 Hz. The result matched the original movement with accuracy in range of 1-2 %. The conclusion is that little extra information is captured with higher sampling rates. This is understandable as human sight isn't very fast and conductors quick movements would only confuse musicians. This is not to say musicians cannot synchronize with accuracy better than 40 ms (matching 25 Hz) using only eyes. They certainly can, but the accuracy comes from understanding the flow of the movement and experience, not extremely accurate and fast eyes.

Initially noisy measurements were a problem. It has been found that the measurement noise depends heavily on the measurement frequency. Good measurement rates can be found in ranges 30-40, 60-70 and 80-105 Hz in our operation environment. In these ranges the measurement noise is around 1 mm (distance) and 0.01 radians (angles). The noisy measurement ranges imply that 50Hz AC power lines disturb the tracker.

3.5 Accelerometers

The magnetic tracker is a rather costly tool. As an alternative we have used accelerometers to track the baton motion. The output voltage from an acceleration sensor indicates directly the acceleration the sensor is experiencing. By integrating the acceleration values one can calculate velocity and further integration yields position. Unfortunately such methods always suffer from drifting position and velocity estimations.

Heuristics can be used to counter the effect of drifting. In conductor following the sensor (in baton) is confined to certain space and it is held in human fashion. The knowledge of the setup is used to DSP filters to counter the effects of drifting. Leaky integrators can be used to guarantee that the position does not drift away from zero position.

The baton can move freely, it has six degrees of freedom. One could have multiple accelerometers to detect each dimension and keep track of position and rotation. The implementation we built used only two sensors. One sensor is expected to measure vertical acceleration while the other tracks horizontal acceleration (see figure 3.2).

Since the baton can be held in any way the system must automatically detect the orientation. To accomplish this the system integrates each sensor separately and tries to estimate the direction of gravity. Gravity presents itself as persistent acceleration component to one direction. By observing the output from leaky integrators one can deduce the direction of gravity. The coordinates of the baton sensors can then be transformed to relative world coordinates by matrix multiplication. In the relative world coordinates the absolute potition is not known, but the vertical and horizontal axis are presented correctly. When the baton rotation changes the system uses wrong orientation estimation for a while. During this time the tracking is very inaccurate. It has been found that most users do not rotate the baton (twist their arm) when conducting.

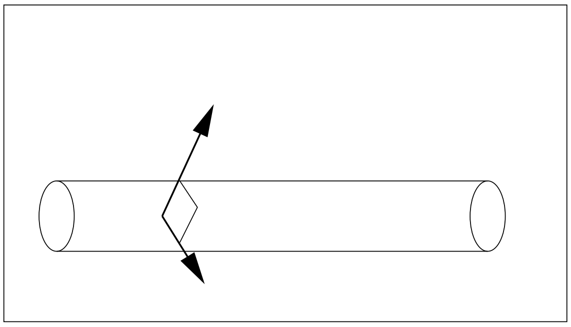

Figure 3.2: The measurement directions of baton accelerometers.

If the baton is turned while conducting the motion tracking mechanism gets badly lost for a few seconds. The loss of measurement accuracy is caused by faulty estimation of gravity direction. Since there is no direct way to detect baton rotation with two sensors this has to be tolerated.

Even with all filtering the accuracy of accelerometers is significantly weaker than the accuracy of magnetic tracker. One cannot deduce the global position by using the accelerometers. Luckily the absolute position is not needed. We know that the baton is always in front of the conductor. Slow motions are filtered out by the leaky integrators.

Unfortunately we chose sensors with too narrow measurement range - three gravities. Since human arm can easily produce more than 10 g acceleration the sensor output clips during forceful movement. This in turn led to distortion in velocity and position measurement. In spite of all limitations the measurement system works and can be used for motion tracking in this application.

The accelerometers are light and small, smaller than a sugar cube. The sensors were received as sponsoring from VTI Hamlin (Hamlin., n.d.).

Chapter 4

Tempo

Recognizing tempo and controlling the playing of the music with this tempo has been the main research goal so far.

The tempo should change swiftly when conductor wants to change it, but it should stay stable in other cases, even if the movements of the conductor don't match an exactly even tempo. Fullfilling these contradicting aims at the same time is one of the main problems of tempo tracking (figure 4.1). Another way to state the problem is: "How to perform a musical tempo change when needed ?".

4.1 Methods for estimating tempo

There are many methods for tracking the rhythm. The most direct methods attempt to take the rhythm directly from the baton movement. The more elaborate methods do not try to follow the rhythm directly, but try to localize the phase within conductor's beat. The phase and phase velocity (fig 4.2) can be used to calculate the tempo estimation.

In this case, one can follow the rhythm even if the division is more complex (like 5/8 in figure 2.4) and when conductor leaves single beat out or adds them. The more simple methods of rhythm tracking assume (often implicitly) certain beat pattern that can't be varied.

Most methods used for conductor following rely on using only the movement along the vertical z-axis since it is the most simple to interpret.

The most direct (and crude) approach is to measure the time between down-beats and use that for calculating the tempo. Sophistication can be added by analyzing also the movement between down-beats. Phase-locked oscillators can be used to synchronize an oscillator to the motion curves. The motion curves can also be approximated with function fitting. The best-matching function fitting parameters contain information of the beat and style. Neural networks can classify the motion parameters or calculate musical parameters directly from the motion data.

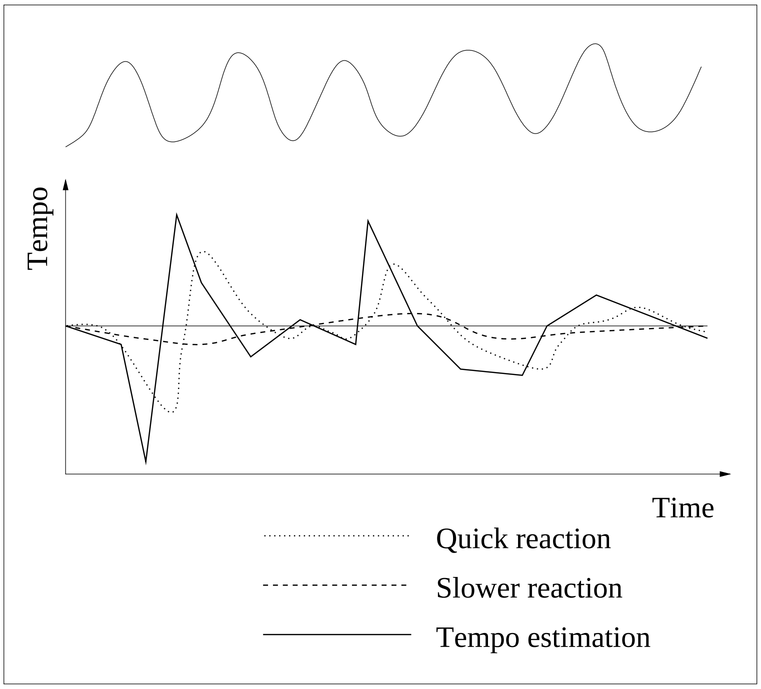

Figure 4.1: Two alternate ways to react to estimated tempo change. Note that tempo is intended to be completely stable.

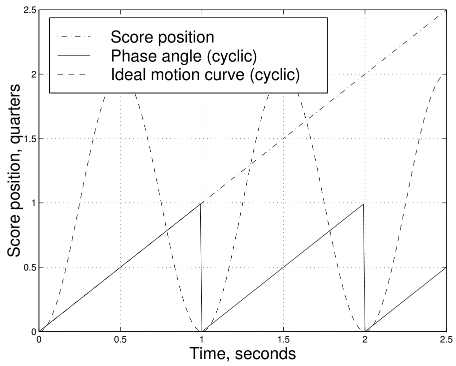

Figure 4.2: How the score position is wrapped to motion phase angle.

If one thinks about the three parts of the conductor follower these methods look very different. The more simple methods mix the parts together. The more complex methods have clearer distinction between different processing tasks.

In our solution the problem is solved in three parts. First one must estimate motion phase (section 4.2) and detect beats (section 4.3.1). In the second stage the beats are classified and matched to expected beat patterns (sections 4.3.3 - 4.4). To generate a musical output that sounds good, ie. the conductor follower uses heuristic methods (section 4.5.1).

4.1.1 From phase to score position

The motion that controls rhythm is cyclical - the end of one bar/beat is the beginning of the next. It is a natural approach to analyze what is the phase of the cycle in given moment. The cycle length is one physical beat. The motion phase can be anything between zero and one.

The cyclic phase angle has to mapped to musical output. Usually this is done by unwrapping the cyclic input as in figure 4.2. Finding the phase within one beat has usually been the key method in rhythm tracking. Mapping this phase to proper score position is the next interesting stage. The discontinuity at cycle adges can be avoided by using sin/cos pair to indicate phase.

4.1.2 Vertical extrema

The most simple method is to follow the extrema of the movement, basically the top and bottom points of the beat (Mathews, 1991). This method lacks the capability to predict the movements of a conductor. Problem is due to the fact that there are only two extrema (up and down) during one beat. While there are no up- or down beats the following program cannot draw any conclusions of what will happen with the rhythm. When there are no rapid tempo changes this method can be used. This method does not assume anything about motion between the beats and therefore limits the movements that can be used little.

Once the beat is detected tempo needs to be adjusted. The brute force method is to update the tempo as a new beat is observed and - at the same time - move to a proper score position. Whenever a beat comes earlier or later than expected there will be a clear discontinuity in the rhythm. Tempo and position can be corrected over a period of time. The rhythm will react slower, but the tempo will be more stable.

This method has been used by Large (1994) for tracking the tempo of an impulse train. The algorithm by Large also handles missing or extra beats by ignoring them. Using multiple oscillators that track also the subdivision beats has been done by Toiviainen (1997). An algorithm that does not create discontinuities in the tempo has also been formulated by Toiviainen (1998). In this case, one changes the tempo quickly, but not instantly after a new beat. With deliberate temporary overshooting of the tempo change one can correct the position in score while updating the tempo. This is demonstrated in figure 4.3. It would be useful to test what would be the optimal shape for the correction curve.

I have implemented the algorithm by Toiviainen and tested them in

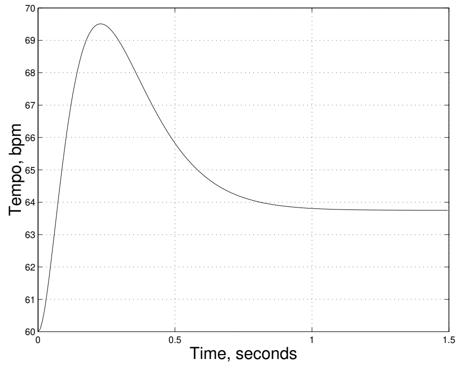

Figure 4.3: Possible tempo change curve as conductor is found to be ahead of the follower.

the context of conductor following. In the current system the method by Toiviainen can be used when the tempo is expected to be stable.

Old and well understood method used in electrical engineering has been the phase locked loop (PLL). PLLs have been designed to work well when the phase velocity (tempo) of the input does not change very fast. With proper parameter settings this method can be useful. In fact it produces tempo change curves fairly similar to the algorithms of Toiviainen.

4.1.3 Oscillator

We also tried to use oscillator for tempo tracking. Oscillator had a waveform resembling the conductors movements and its output was multiplied with conductors movements coordinates. Thus correlation between oscillator output and measured motion could be used to determine if there is need to change tempo.

This method was too sensitive to small variations in conducting style and large set of heuristics were needed to follow even a simplistic stable rhythm. To work well the oscillator must be conducted with exactly the same movement trajectory as the oscillator waveform.

The correlation does not indicate quickly if the oscillator is tracking the conductor badly, preventing fast tempo changes.

This method relates closely to phase-locked loop oscillators from which the inspiration came.

4.1.4 Function fitting

Function fitting can be used to estimate motion phase. One can take a cyclic function that represents the waveform of conductors movements and try to match the movement to that. In this case, the function would resemble a sine. Generating the optimal mathematical function that fits the data is a problem. One can construct it by hand, but this requires new hand work whenever new style is to be learned. One could also analyze the data to find an average waveform.

This is probably a workable solution. I suspect this approach would have problems since the conductors movements don't always look the same, while the function does. Clever parametrization of the function might overcome this problem.

4.1.5 Nearest neighbour classifier

We also tried using a nearest neighbour (NN) classifier ((Schalkoff, 1992), chapter 3). This meant creating a group of feature vectors that hold information of baton tip movement. Each data vector has corresponding phase angle. The input vector is taken from real movement and the best matching unit is searched from the map.

The feature vector contained baton tip position and multiple delayed velocities. The delays of the movement velocity were matched to be musically relevant, so rather than using delay of 5 samples one would use a delay of one sixteenth of a note. This has the drawback that before we know the tempo we should already know the tempo to get the delays correct. One can make a scientific guess about the tempo by calculating the time difference between two down-beats. method to continuously track the tempo. When the tempo is changing fast the delay times will adjust slightly late. This results in tracking errors. In practice this method cannot be used to get accurate non-parametric phase estimation.

Tests with constant delays showed major differences in tempo tended to make the networks output very unstable as the preprocessing did not match the training data delays. By using more samples conducted in different tempos this problem might be overcome.

Another problem with NN-classifier is that the parameters that form the input vector need to be scaled to give each parameters proper relative weight. Finding the proper weights is a non-trivial problem.

The positive thing with NN-classifier is that the distance between measured feature vector and best matching unit indicates how well the network is following the conductor. The distance can be used as an error indicator.

4.1.6 Self organizing map ANN

The results of nearest neighbour classifier can be duplicated with self organizing map (SOM - (Kohonen, 1995)). All map vectors need to be labeled with corresponding phase angle. This can be done by assigning each map unit a phase angle (section 8.1) and updating it during training.

Modified SOM architecture yeilded better results. All units of SOM had an output vector that was trained with the input vector using sine and cosine of the phase angle. The resuls of using this approach were nearly a good as the results obtained with chosen MLP method. The SOM acrhitecture is described in section 8.1.

We have found that the SOM can be used for rhythm tracking. The only problem it is the need to combine very different parameters. Completely different parameters like position (from -1 to 1 meters) and staccato (on/off) may be uncomparable with the normal distance measures.

4.2 Detecting beat phase

A multi-MLP network (section 8.2) has been adopted as the beat phase estimation tool. The network input vector contains motion parameters. The training target was the phase angle of the beat. One can use the physical down-beats or played music as reference when creating training data.

4.2.1 Pre-processing

The feature vector contains the current baton tip position and past velocities. Typically one would have 5-10 delayed velocity values (horizontal and vertical), with maximum delay of about one second. This yields non-parametric pre-processing which includes short memory.

Once I accidentally used the beat-type prediction data as MLP training data (described in section 4.3.3). The results were roughly as good as the results obtained with the normal method.

4.2.2 Multi-layer perceptron

Minor extensions to standard MLP structure have been made for this particular situation. These are multiple MLPs and MLP with cyclic output (sections 8.2 and 8.3).

Figure 4.4 shows an example of how the network can handle the motion. At first gradient descend training algorithms were used. The optimization process often led to local minimums. Simulated annealing was then tried. This yielded good resistance against local minimums during training.

We expected that over-learning would not be a problem. Figure 4.5 shows results of using a separate test set to validate this hopythesis. In the simple test (right plots) the original training set was split in two by assigning every other training sample to training set and the rest to test set. In this comparison the training set and test set errors correlate closely.

In the more demanding test the test set was completely separate from the training set (same conductor tough). Thus the conducting style present in test set was not contained in the training set. As expected, the net optimizes

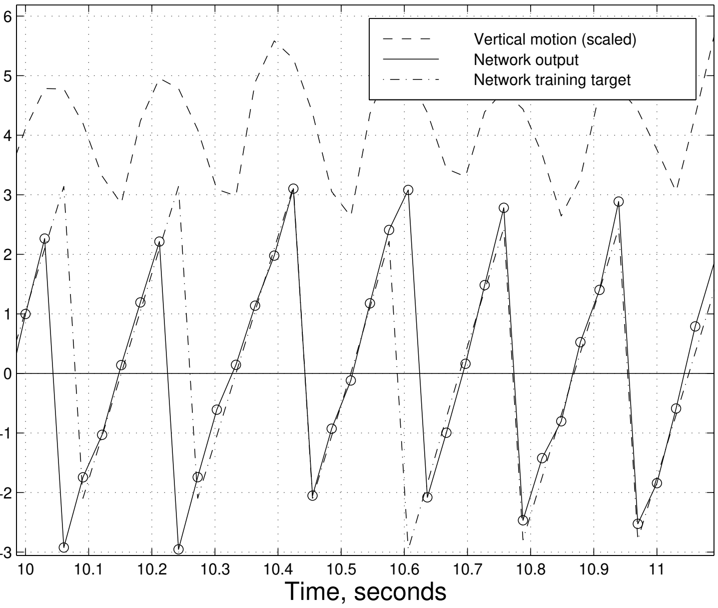

Figure 4.4: ANN-estimated beat phase compared to target beat phase.

its performance for the styles present in the training data. Fortunately the test set error does not increase as the MLP is fine-tuned for the training set. In these tests the size of MLP was less than 20 neurons and the size of training and test sets greater that 1000 examples.

4.2.3 Gathering training data

The training data should be recorded as unobtrusively as possible. Limitations are the time limits for participants and the amount of effort one is willing to spend on the measurement. In ideal case one could take the measurement setup to concert or training session and measure motion in natural environment.

Our measurement setup is illustrated in figure 4.6. Metronome gives tempo to the conductor. The timing of metronome clicks is measured with the motion. Both are time-stamped with millisecond accuracy and saved to a file. The metronome has sometimes been replaced with hand-clapping from a musician. Thus passages with varying tempo can be conducted with reference time (the claps). This way to gather training data for MLP resembles the method used by Brecht and Garnett (1995).

The person controlling the setup tells the conductor how to conduct (style, time signature). This method gives highly controllable environment where results are easy to interpret.

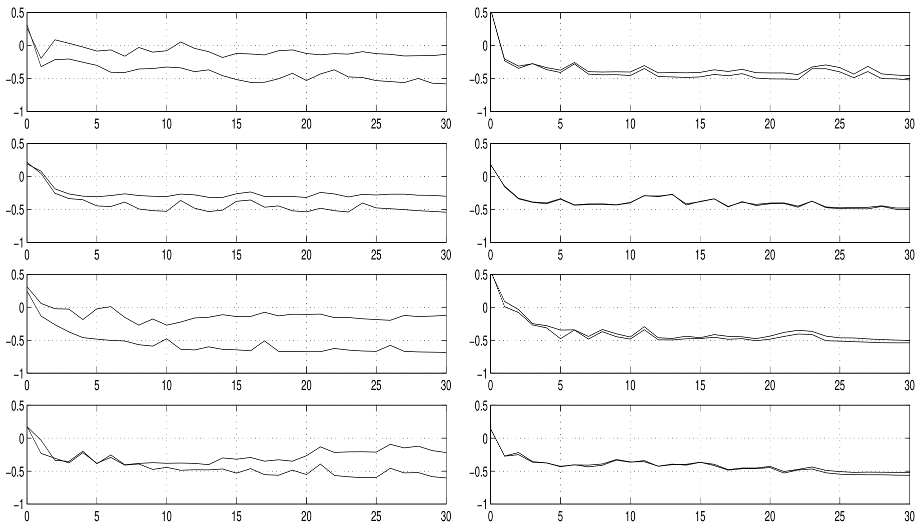

Figure 4.5: Over-learning with multiple MLPs (four MLPs, 2x2 SOM distribution - section 8.2). Vertical axis indicates logarithm of error for both training and test sets. In all cases the lower line represents training set error.

The method can be criticized for various shortcomings. The person who is measured knows very well the situation. This affects his behaviour. Having a metronome conduct the conductor is exactly the wrong way around. Preferably one should have a musician playing an instrument.

4.3 Beat patterns

Beat pattern presents useful information to musicians. It indicates which beat should be played. This is valuable information for error correction should the musician get lost. The knowledge can be used to help predict the down-beats. Being able to guess the form (type) of the next beat makes predicting the beats easier.

Different types of information that can be obtained are the time of down/up-beats, the type of beats and the beat pattern.

4.3.1 Finding down-beats

One needs to know when the beats are taking place. The most obvious way to find up- and down beats is by looking for minima and maxima in the vertical movement as in figure 4.7. Since any extrema is considered a beat this method is very sensitive to false beats. Additional heuristic rules reduce the number of misdetected beats. The rules discard beats that are too close to each other (distance is measured in time or spatial distance).

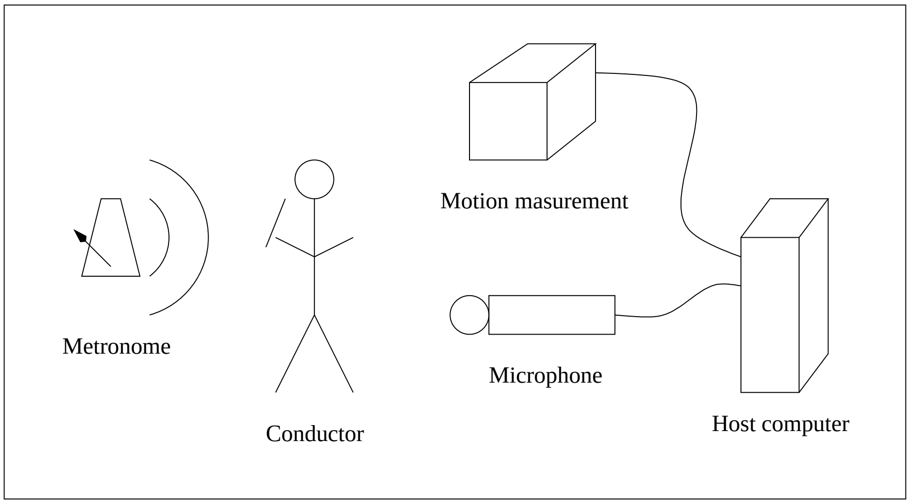

Figure 4.6: Measurement setup

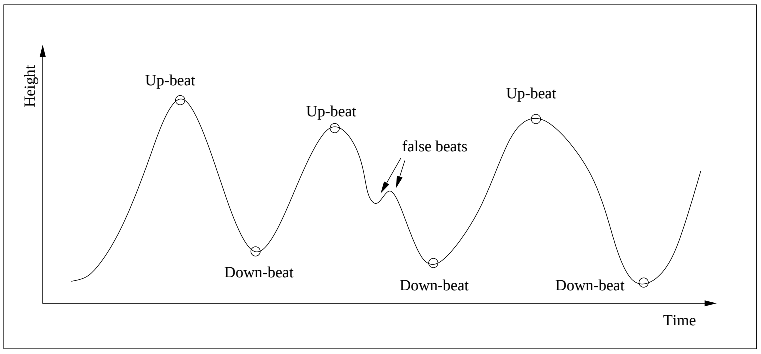

Figure 4.7: Using extremums to find beat times.

Curvature can be used to find up and down-beats. This method can be used in conjunction with the extremum method for higher reliability. Figure 4.8 shows how curvature is related to beats.

4.3.2 Beat classes

With standard conducting style, there are clearly different beat types for each bar part (McElheran, 1989).

One way to classify beats is using their position in beat pattern as an indicator. Different beat patterns consist of similar beats that are arranged in different order. I have selected four primary beat types.

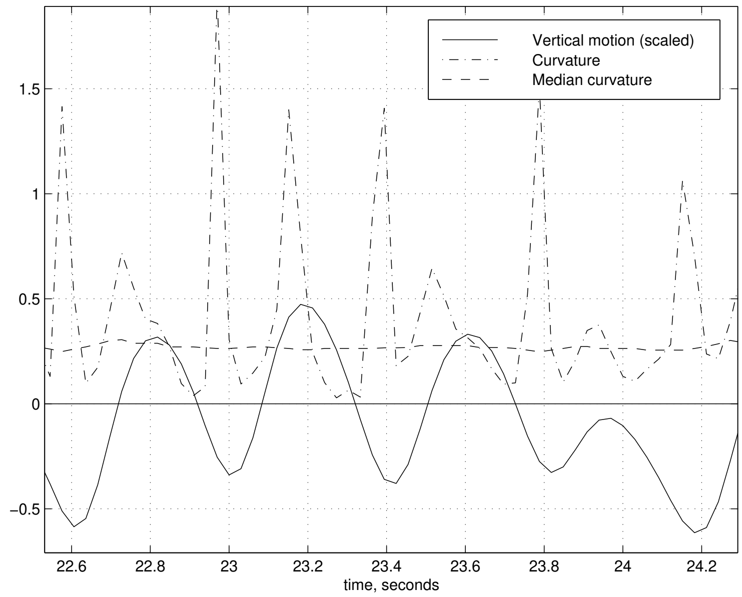

Figure 4.8: Curvature. Beat pattern is standard 4/4.

All of these are found in time signature 4/4. These relate to figure 2.2.

Down-beat	the first beat
To Left	the second beat
To Right	the third beat
To the center	the final beat

Another way to classify beats is considering them either sub-division beats or normal beats. In figure 2.5 the beats 2 1/2 and 3 1/2 are subdivisions.

The two classifications are used in parallel, resulting in total of eight permutations. Using this classification all beats in figure 2.5 can have unique class. If there are more sub-divisions the classes are no longer unique but classification still provides useful data. Such difficult case could be made by dividing a beat to three parts (eg. 2, 2 1/3, 2 2/3). Beat parts 2 2/3 and 2 2/3 are both of the same class (to left, subdivision).

4.3.3 Beat classification method

Beats are classified using self organizing map (SOM (Kohonen, 1995)). Input vectors for the map are formed from baton tip motion. Map units contain output vector containing classification information.

The features used as input present the form of baton tip motion during the last two beats. As new down-beat is detected (section 4.3.1) the motion between two latest beats is sampled to give a break-point graph of the motion. The coordinate values are the used as SOM inputs. A group of break-point graphs is illustrated in figure 4.9.

Different motion amplitudes produce graphs similar in form, but different in amplitude. To counter this the curve is normalized in relation to current movement size. The current movement size is estimated by calculating the bounding box of motion for the last few bars. The maximum dimension of the box is used to scale the input feature vector. The data contains information of the form of the motion, with implicit timing information.

An alternative scaling would scale both axis independently. Such preprocessing could distort the movement badly . A thin ellipse and a circle would have same final representation which is clearly undesirable.

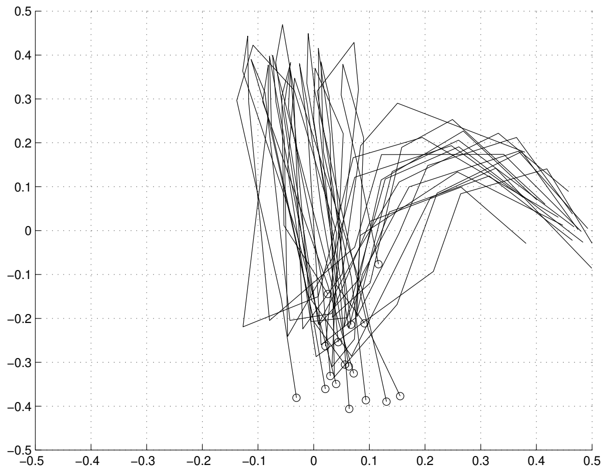

Figure 4.9: Beat pattern feature graphs for the first beat of the bar. Circles represent the end point of the beat.

The timing information can be problematic, figure 4.10 shows a case where two curves are similar in shape, but result in different feature vector. This problem could be overcome by choosing the re-sampling times with greater care. Rather than dividing a beat to equal-time intervals the beat could be divided to equal-distance intervals. This idea has not been tested yet.

In training phase input vectors have an associated output vector. The output vectors contain estimated probabilities of different classes for a given map unit. Thus the output vector elements have values from zero to one. The first four elements of the vector indicate which of the four primary classes the beat belongs to. The correct beat class has an output weight of 1 while the other three are zeroes. The fifth element indicates whether this is a sub-division beat or not. Figure 4.13 displays classification vectors.

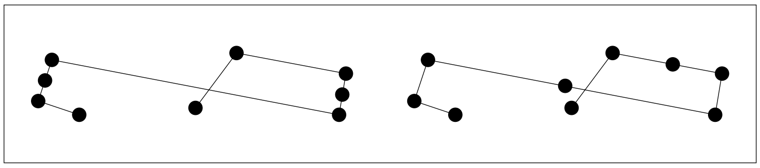

Figure 4.10: Two very similar motion curves with different velocity distribution. The dots represent sampling points (four samples per beat).

The SOM training algorithm is slightly modified to allow output vectors to be used (more details in section 8.1). In training phase the best matching unit is searched as usual (based on the input vector). The input and output vectors of map units are trained together.

As long as down-beat recognition works well this method is fairly reliable in recognizing the correct beat type. In good circumstances the misclassification rate can be as low as 5 %.

Tests with further preprocessing yielded slightly better results. Rather than using breakpoint description of position one can calculate differences between the points. This method has proved marginally better at distinguishing between beat types (improvement less than 10 %). The problem of different speed distributions (figure 4.10) is not solved by this approach.

4.3.4 Predicting the class of the next beat

Knowing the class of beat before the beat takes place can be very useful. It helps in determining which score position should match the next down beat.

The ANN used for this purpose is exactly the same as the one used for beat classification. The difference is in preprocessing of motion data.

The data for beat type classification uses the knowledge of the timing of beats. The predictor has little knowledge of the time of the next beat. The motion between the two latest known beats is sampled in equal intervals. The motion since the last known beat is sampled from the last known beat to the current time. This sampling scheme is demonstrated in figure 4.11.

The class of the next beat cannot be estimated well long before the beat. The distance between best matching unit of the SOM and the feature vector indicates how reliable the beat class estimation is. The class of the winning SOM unit indicates the expected class of the next beat. As the beat ends matching error diminishes and the beat classification vector stabilizes. Figure 4.12 illustrates how the feature matching error relates to the part of beat.

One could also create a separate SOM for this purpose rather than recycle the beat classifier SOM. While a plausible approach it has not been tested as the current implementation works well enough.

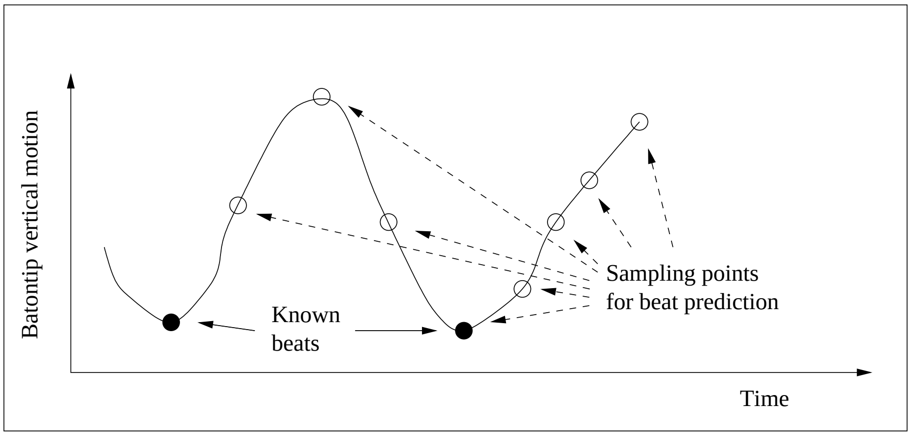

Figure 4.11: Sampling motion for beat class prediction

4.4 Beat pattern classification

In concert situation musicians know the beat pattern in advance. Making radical changes to the expected pattern is likely to cause problems for the musicians.

The expected beat pattern is represented as a list of possible beats. The known beats have certain parameters associated with them: Beat class vector, expectation (not probability) and score position.

4.4.1 Matching observed beats to expected beats

All observed beats should be associated with a beat in the list of possible beats. Various combinations of associating observed beats to expected beats are tested and the best matching permutation wins.

If one tries to test all possible combinations of matching observed beats to beat knowledge the number of permutations is usually too large. For example matching 4 observed beats to 100 possible beats would result in about 100^4 permutations.

Only part of the knowledge base is used for pattern matching at a time. The beginning of the matching section is determined from the older beats matching estimations. The new match area begins from the last “certain” beat and ends within n beats.

After the matching has been performed the oldest beat that was matched is used to indicate the beginning of search segment when the next beat is observed. An example of beat pattern matching can be found in figure 4.13.

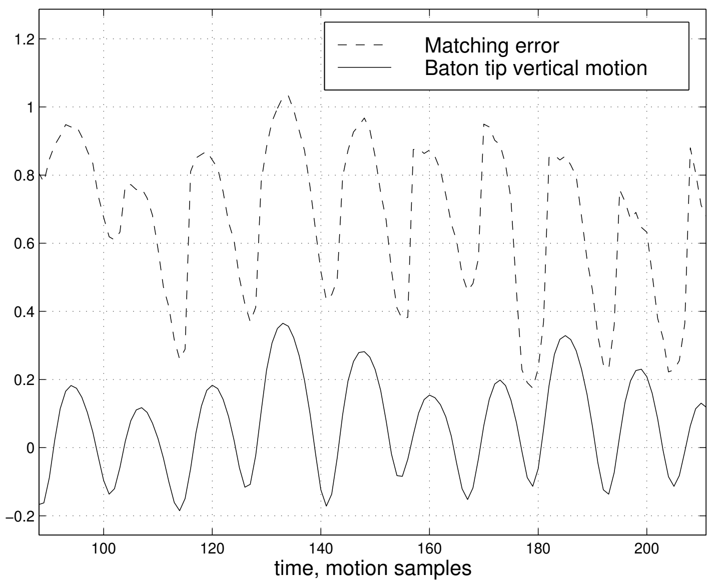

Figure 4.12: The class prediction feature matching error as a function of time (real results).

4.4.2 The matching error

The most direct way to match beats is by comparing the class vectors of observed and expected beats. This method ignores the dependencies between beats and the timing of beats. Both are important parameters when trying to match beats to knowledge and predicting what kind of beat the conductor will make next.

Adding an extra cost functions (besides beat-classification error) makes the system more robust. Figure 4.14 displays how the projected difference between two beats can be used to add cost of choosing particular beat-matching combination.

Finding the proper balance between beat-classification cost (error) and other cost functions is currently a matter of hand-tuning. The optimal balance can be expected to vary depending on the conductor and the piece.

4.4.3 Results

In good circumstances beat pattern is matched perfectly. When the beat classification fails the beat pattern matching becames a lot more difficult. The beat pattern matching algorithm can make false estimation. For this very reason there algorithm tries to match many beats at once. Even if one beat is misplaced the system can recover from it.

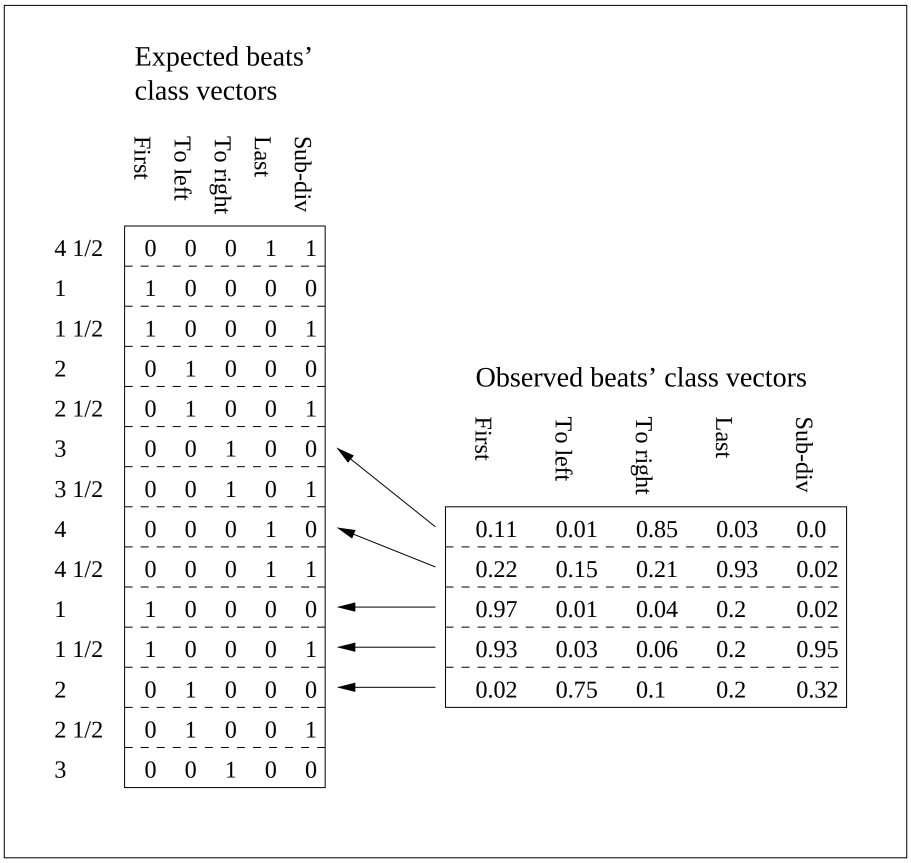

Figure 4.13: Matching observed beats to the knowledge of the beats. A segment of known beat pattern list is used for finding the best fit for the latest observed beats.

4.5 Controlling tempo

Multiple tempo control methods have been tested and implemented. The primary tempo generation method has used heuristics. The follower adjusts tempo according to certain hard-coded rules.

Finding the proper heuristics for tempo adjustments is not a trivial problem. A tempo change can be done in different ways and one should test them to find the best. An alternative approach was also tested briefly - ANN for tempo control. While not currently used in the system I hope ANNs can be tuned to control tempo in the future.

4.5.1 Heuristics

The heuristic tempo control method is implemented as a state machine. The player model has an internal state that consists of current score position and

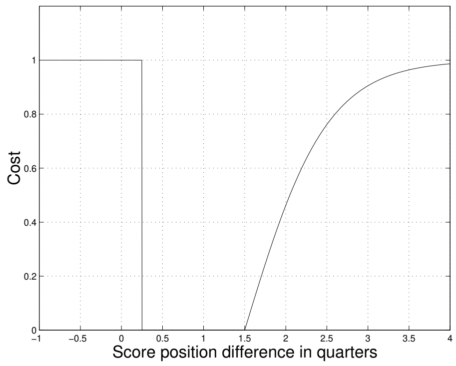

Figure 4.14: A cost function for beat pattern matching. This function penalizes physical beats placed very close to each other in the score (or far from each other).

tempo. The system inputs has a number of input parameters: the expected and estimated tempo, expected and estimated nuances, the score and the desired expression style.

The base tempo estimation is taken from beats - the time and score difference between down-beats indicates the tempo. This is a reliable measure as long as the tempo does not change too quickly and beats are recognized correctly.

Finer adjustments are done constantly depending on the estimated beat phase (section 4.2). If the follower appears to be lagging behind (or advancing too much) both tempo and score position are adjusted to counter the effect. The strength of the adjustment can be varied depending on the score, tempo, conductors gestures and musical hints.

When many notes are played quickly in succession tempo changes become more audible. As an opposite, if a long pause is about to end the first note-on event can be passed without concern about the past tempo. Reactivity is automatically decreased as note density increases to compensate for this.

It is important that we have explicit state, not a hidden state in a control filter (PLL etc). The explicit state can be controlled easily. If the tempo control was done with a filter the parameters would be more difficult to change in real-time - modifying the state of an operational DSP filter is not a straightforward operation.

The beat pattern tracking results are used to verify that the system is playing the correct score position. If the score position of the player does not equal pattern-matching results the score position of the player is corrected.

The heuristic algorithm leaves data unused. The nuancing information is ignored. In future one might use the algorithms by Bresin and Friberg

(1998) to create more expressive output.

4.5.2 ANN method

Preliminary tests were performed to see if the heuristics could be replaced with an ANN (only MLP was tested). Ultimately the aim is to mimic actions of real musician with ANN. The ANN could be trained with played passages of music and it should learn the rules that govern musical expression. Thus one would not need heuristic algorithm for expression synthesis. As far as research interests are concerned an ANN-based approach would be much more interesting than a collection of ad hoc rules.

The training data must come from real world - interaction between conductor and musician. Since the only thing one can measure directly is the timing of notes that knowledge must be used. The time difference of ANN-estimated note output time and human timing is the measure of error.

This scheme attempts to measure event timing error, a rather rare error function in ANN science. An alternative error measure would be the tempo used. Event timing is more realistic measure, since the events make up the tempo. With no events, tempo cannot be perceived.

Often ANNs are used to minimize the sum of errors over the training set (global error). Defining the global error is difficult in this case. Summing the error of all event timings would give some indication. The relationship between events is in that case ignored. Figure 4.15 displays two estimations with similar sum of errors, but different local tempo changes.

The absolute time difference is a valid measure only for the first note in test run. Figure 4.16 shows that the new context provided by mistiming a note should be reflected in the timing of following notes.

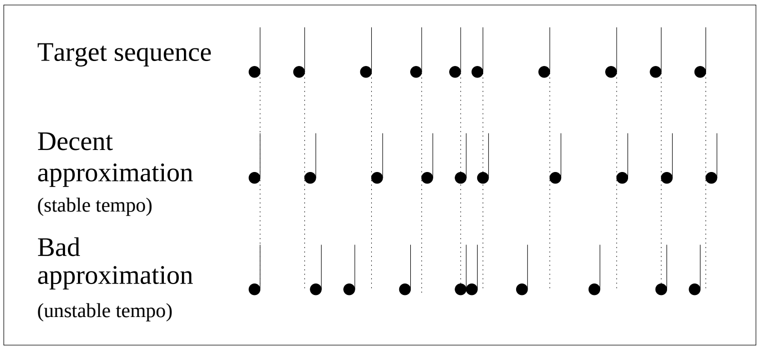

Figure 4.15: Comparing two approximations with similar sum of local errors.

The actions of the musician depend on past actions (feedback). To simulate this the ANN needs to have a context describing the current situation. I have used an explicit context: The new input vector to the ANN is created based on the output of previous rounds.

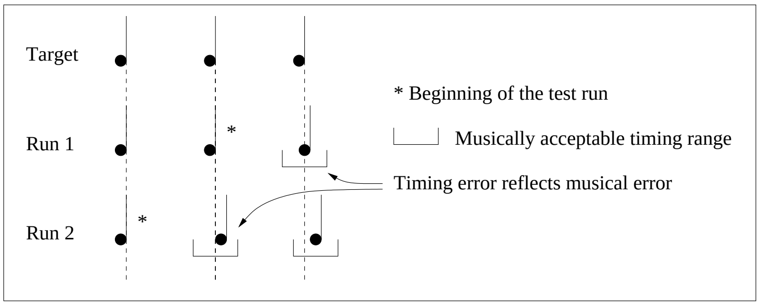

Figure 4.16: Only the first note in the run will automatically have valid timing error.

Getting a measure for local musically relevant errors can be done using limited timesteps as opposed to typical unlimited timesteps (figure 4.17). Short runs only use short segment of the training seqeuence at once. When measuring total error multiple short runs are performed to cover the whole training data. When short runs are used one takes the context (history) from the target performance (the training target) and uses the net for a limited number of samples. Then the ideal context is taken again from the next starting point.

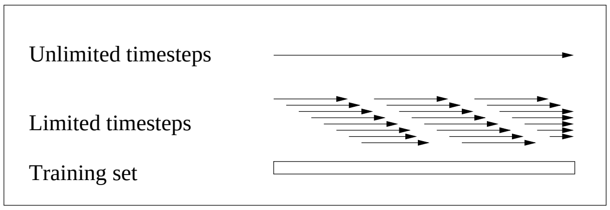

Figure 4.17: Unlimited timesteps vs. limited timesteps.

The assumptions are:

- The context parameters of the target performance imply correct musical behaviour.
- Feeding this context to the ANN at the beginning of each short run gives it an ideal context.
- The timing error for the first note of ANN-generated output will be musically relevant.
- The error for the next few events must not be too far from the ideal error either.

The next pseudocode illustrates how the recursion is carried out in the current scheme.

```
error_sum = 0
for(i=0; i < short_run_length; i = i+1) {
  context = idealcontext[i]
  for(j=0; j < short_run_length; j = j+1) {
    ann_input  = [gestures context]
    net_out    = ann.value(ann_input)
    error_sum  = error_sum + error(net_out, context)
    context    = new_value(net_out, context)
  }
}
```

The aim of the ANN is to estimate time until uopcoming event. The training setup is in figure 4.18.

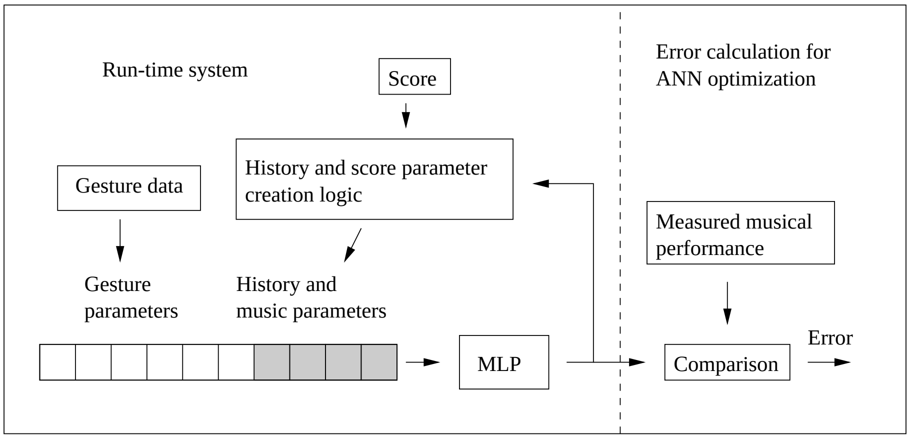

Figure 4.18: ANN player model and the training environment.

The ANN input vector contains three sets of parameters: history, music and gesture parameters. Gesture parameters represent the estimated gestures - beat phase, motion size, nuancing etc. Music parameters represent the current upcoming score. New notes are presented by the score position difference between the upcoming and last played note. History parameters refer to results of past actions. If a note was just played it is manifested in the history parameters. The tempo implied by playing that note is also in the history parameters. Since history parameters are calculated explicitly they can be recalled at any point in the training set. This is necessary if one wants to be able to compare event timing to measured timings with short test runs to guarantee short-term behaviour will get optimized.

The network output indicates the time till the next event. If the network predicts the next event should be played within the next sample the event is "triggered". Once an event is triggered by the network the error is calculated. Absolute time difference between target time and the ANN triggered time is used as the basis of error.

At first I simply squared the absolute value to get error measure. That seemed to work badly - often the network failed to converge. Eventually the timing error was determined with: $error = abs(t)^{1.5}$ where t is the timing difference between network output and training target. Basically this means that big absolute errors do not dominate the total error. If the event should have been triggered, but the network leaves it untriggered the error is also calculated. This is still ad-hoc error measure. The musically valid error measure is unknown. It may well be far more complex than simply event time difference.

When testing the setup I first fabricated training data that represented a simplistic piece of music. Fabrication was used since it is easy and controllable way to get data: one can directly affect the parameters - tempo, accelerandos etc. The results were promising, but not fabulous. The system could not learn to trigger events even in simplistic case. Having deadlines to meet I could not study this approach further. Examining this method thoroughly is probably the next direction in developing the player model.

Often recurrent Elman MLPs (figure 4.19) are used for time series-processing. With recurrent network the context is represented by the state of the net. There is no way one could take one context and replace with another since the context format is different (hidden in the net) for different nets. If one cannot set the context for each run the will cannot be expected to perform well, if at all.

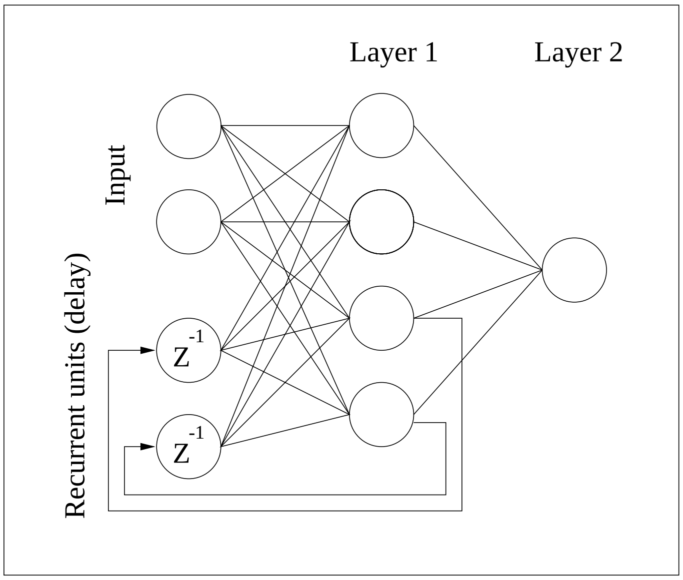

Figure 4.19: Recurrent Elman network.

Chapter 5

Special gestures

Certain gestures have been deemed important enough to warrant a specific gesture detector. In current implementation all of these are tracked with heuristic methods.

5.1 Beginning stroke

All musical pieces start. It is highly desirable to start the piece in the right time.

The first note is not played immediately when the beginning is detected. The system waits for the proper time to play the note - typically first down-beat or second up-beat.

The motion tracking mechanism may limit the way beginning can be detected.

Two alternate methods have been made. The choice of method can be selected as program is started.

5.1.1 Observe position

Figure 2.7 shows an example of piece start. It is typical example and can be used as starting point. The algorithm to detect beginning is constructed by hand since the situation is simple enough.

The method for finding the beginning is a state machine that has four states:

- Baton lowered - not began yet
- Ready - baton raised to waist and still
- Starting - baton moving and raised to shoulder-level
- Began

Once the state machine reaches the fourth state the music can start. There are various parameters that need to be specified, the treshold values for waist and shoulder hieght etc.

As long as the conductor behaves as expected the system will recognize the beginning of the piece reliably. It has been found that the system can work with the same parameter settings for multiple conductors.

5.1.2 Observe activity

With accelerometers it is impossible to know the absolute position of any sensor (section 3.5). This alone renders position-based start impossible. An alternative we have used is measuring conductor activity. To detect beginning we simply decide that once certain minimum acceleration is exceeded the conductor is active and playing is began (figure 5.1).

Any strong movement will trigger the playing. This is rather crude method, but when the system needs to react easily (not realistically) to user input it can be used. I call this method "Heureka beginning", since it was made for the Heureka demonstration (appendix A).

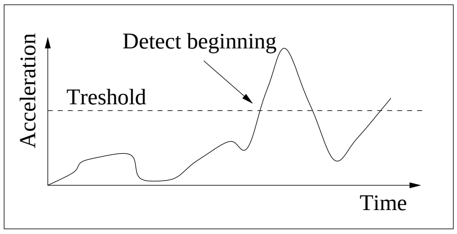

Figure 5.1: Determining the beginning time by activity.

5.2 End stroke

Often a conductor can perform a specific end stroke. This is typically a wide stroke that begins from left/high and ends to right/low. It is relatively easy to detect them based on the beginning and end position of a stroke. Since finishing the piece prematurely is one of the worst mistakes a musician can do the triggering must be very reliable.

The follower requires multiple end strokes in a row to make it quit playing prematurely. In fact this resembles real life. If conductor wants to stop piece suddenly the band usually will go on playing for several seconds after the first end stroke has been shown. Yelling is often necessary to make the most stubborn musicians notice what is happening.

5.3 Fermata

Fermata means the duration of a particular note is extended far beyond normal rhythmic variations. After the fermata is passed the playing should continue as usual.

Typically conductor stops movement abruptly to indicate fermata. The piece is continued when conductor starts moving again.

Detecting fermatas reliably early enough without a priori knowledge is very difficult. If the fermata is detected too late the playing cannot be halted. If any slowing of movement is considered fermata one gets a lot of misdetected fermatas (movement can be slowed for other reasons as well).

The system can detect fermatas on the fly (if required). This is typically a difficult problem for any musician. In usual circumstances a musician would know when the fermata should be. The follower can be instructed to stop for fermata in certain score positions. In these cases the fermata is forced on the follower. Once fermata is finished the playing will continue as usual.

5.4 Inactivity

The conductor may also be inactive during the piece. This may mean the playing of the piece should stop. It could also mean that the band should play on their own without taking too much notice to the conductor. When the maximum acceleration of the baton falls below certain threshold for long enough inactivity is detected.

Currently inactivity is only used to suggest piece end.

5.5 Staccato and legato

Staccato (shortened notes) and legato (notes glued together) are typical performance hints embedded in motion. Staccato is usually indicated by making angular, jerky movements. This can be detected by measyring the relationship between local maximum and average/median acceleration. This simple method yeilds fairly good results.

The legato can be detected with the same method, small difference between maximum and median acceleration indicates the smoothness of the motion.

Chapter 6

Emotion parameters

There are no specific gestures for emotion like there are for rhythm. The emotional content is embedded in other gestures. One must find general movement features that imply emotion.

The most direct way to extract emotion would be to gather large amount of motion data with given emotional content. Later one could match measured motion data against the data base. This is probably not a possible solution. Two motion curves may resemble each other very much, but minor details will reveal the true meaning. Besides, any new radically new motion shapes will be unclassified.

Rather one must seek features that work on any motion. Given these features one should be able to gain knowledge of what is the emotional content embedded in the motion.

The results of non-verbal communication studies could be useful for this purpose since we are trying to understand the body language.

General features can be extracted with the Fourier transform and histogram analysis. These methods have been tested (visualized), but the actual analysis based on the features has not been done.

During the work these features have not been used for detecting the emotional state. This chapter presents groundwork for such research.

6.1 Motion signal spectrum

One of the most usual methods to analyze any signal is measuring the spectrum of the signal with Fourier series (Fourier, 1822). Since we have discrete samples discrete Fourier Transform is used.

The signal is the motion curve as a function of time. Short period of samples are taken (less than four seconds). These samples are windowed (Hamming or Hanning) and zero-padded. FFT is used to calculate spectrum of the signal. The Fourier transform is performed on three position dimensions, but not the rotation dimensions. Figure 6.1 shows the sum of the axis-wise spectrograms for one conducted passage. FFT window length 1.5 seconds. In this sample conducting style alternates between happy and tense (first happy). Halfway thru, the dynamics are changed from soft to loud.

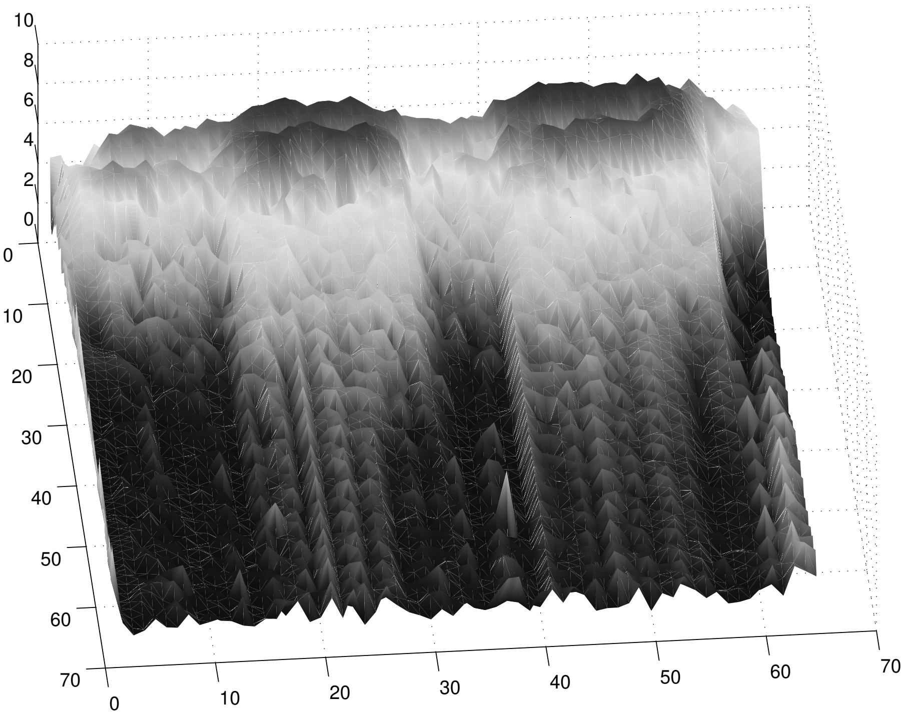

Figure 6.1: Spectrogram of baton tip speed. X-axis display time,Y-axis (depth) frequency and Z-axis (height) amplitude). With color shading these images can be very pretty.

Limitation of using the Fourier transform is that short sample vectors provide little information. In practice this means that very fast mood changes (less than half a second) cannot be tracked by Fourier methods. This limitation may not be a problem; musicians seldom identify mood changes instantenously. Their reaction speed is also limited.

6.2 Histograms

Histogram can display features that cannot be inferred from the spectral data. One maintains collection of samples and calculates histogram of those values. The histrogam implies distribution of certain parameter in the motion. The parameter can be - for example - velocity, acceleration or curvature. As with Fourier transform short analysis periods yield little information. There are a number of parameters that need to be adjusted, the number of bins, scaling of bins and number of samples to use. The best parameter settings are probably context-dependent. The follower should know how to scale them during operation.

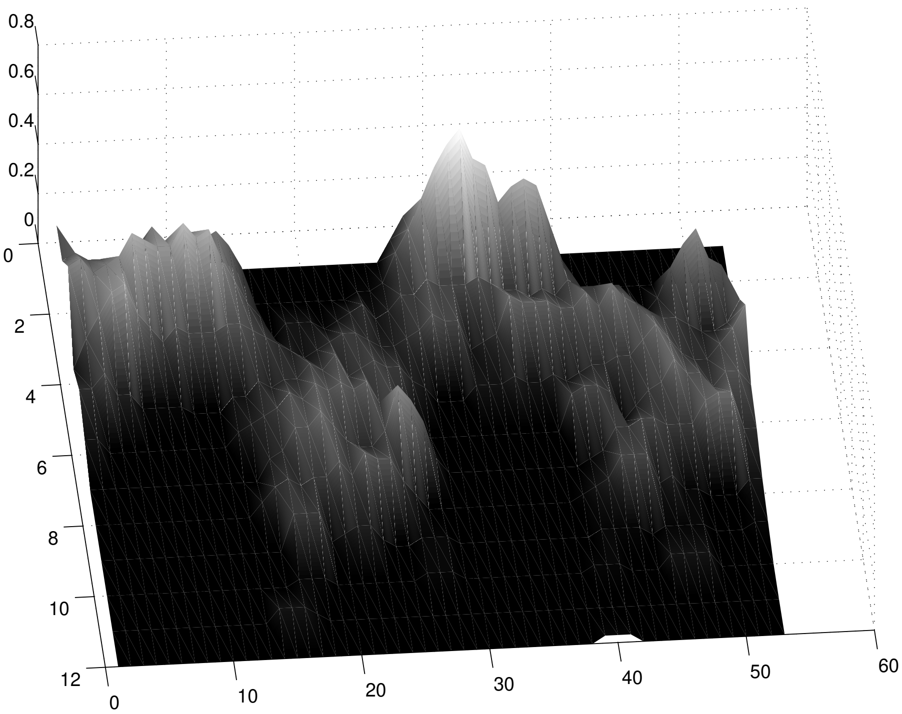

Figure 6.2: Histogram of baton tip acceleration. Motion is the same as in figure 6.1.

Chapter 7

Expression - the feedback

The gesture analysis tools are useless unless the information can be used effectively. Rhythm generation was discussed in tempo tracking - section 4.5. Other issues are discussed here.

7.1 More than one musician

One musician is not an orchestra. When one global tempo estimation rules all virtual players they make the same mistakes in the same time. The opinion of all listeners has been that the non-existing group behaviour of the system is a marked shortcoming.

This calls for multiple player models that react to each other musically. Such systems have not been developed so far. This requires multiplying the player logic and adding methods to get knowledge of the state of other players.

7.2 Hints

In an orchestra a musician knows the piece, it has been played many times before it is performed for the audience. The follower is given a list of hints regarding the music. The hints indicate when a fermata should be expected, times when the tempo might change rapidly, time signature and other features not indicated in MIDI file.

The data is stored in plain text ASCII file. Each song is assigned a hint file. Multiple hint files can be specified for each song (for different conducting styles).

7.3 Creating nuances

The limitations of modern synthesizers and MIDI affect the methods that can be used. Typically, when you want a louder sound only the volume of sound changes. An acoustic instrument changes also the sound color. This can give a feeling of stress and energy. How do you make music more tight, lively or happy ?

In practice when more energy is needed one can increase the note-on velocity and hope for the best. If you want staccato, then shorten the notes. These are blunt methods, but that is what you get with MIDI. If one is willing to spend enough time with editing the sounds and designating which MIDI-parameters affect particular property of the sound better results can be achieved. Such work has been done is Satoshi's system.

One can use rule-based methods to make music more expressive. Creating more musical output is field of study; Canazza has shown a way to shape arbitrary sound sources towards more musical output (Canazza Sergio, 1998). Tested sound sources were standard sample-playback synthesizers, FM synths and music recordings.

7.3.1 The score

For our purposes MIDI file is a problematic tool. MIDI file contains only events. As an example, it does not indicate whether a note is a short note or staccato. MIDI contains real-time message types, note-on and off, pitch change etc. This information is insufficient for the conductor follower. One would like to know the actual score. As a band-aid solution the follower parses a score from the MIDI file. Figure 7.1 shows the difference between the MIDI event flow and actual score.

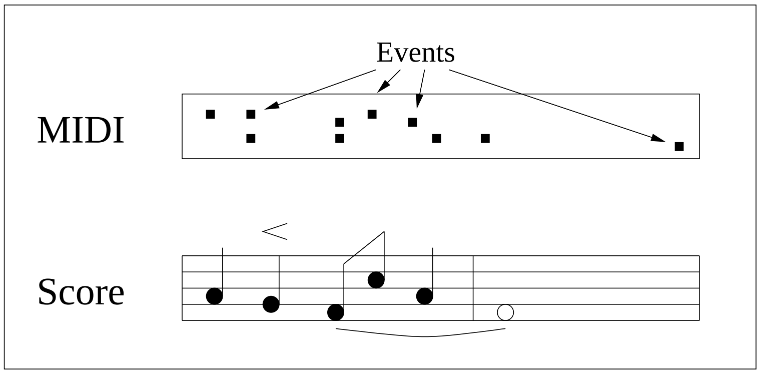

Figure 7.1: MIDI vs. score.

The follower transforms the MIDI file to a score. Note-on/off pairs are sought from the MIDI data. These pairs are then transformed to note structures which define both end and beginning time of the notes. Any control events that take place during the note (pitch bend, aftertouch) are attached to the note structure as extra information.

The player model modifies the score data in real-time according to the gestures conductor makes and also according to the hints (section 7.2).

7.3.2 Staccato

Staccato makes notes shorter and (often) more piercing. Few choices exist for shortening the note. The direct route is to scale note duration by some coefficient. An alternative is to decide maximum staccato note length. The maximum length can be described in logical time (quarters) or real-time (seconds). The follower can use any of these three methods to control staccato note duration.

A particular problem with limiting the duration of the note is abruptness of the effect. Once staccato is detected all notes are shortened aggressively. The scaling approach can be used to adjust to staccato dynamically.

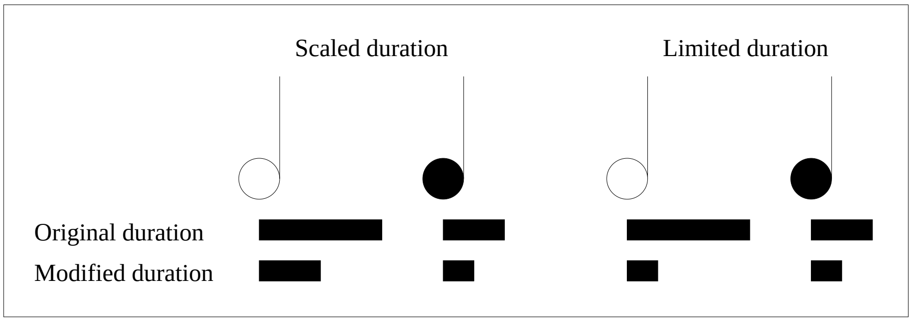

Figure 7.2: Staccato schemes

A musician would note the acoustics when playing staccato. In more reverberant hall the staccato notes (and other notes in lesser degree) are played slightly shorter to counter the effect of reverb. This behaviour has not been mimicked the follower.

Whether one should use the staccato is a demanding question. It has often been commented that a musician should know the nuances in advance. Should the follower react to staccato gestures in full or use them when directed ? This probably depends on situation. If the MIDI-file has written to include staccato (in form of short notes) adding another layer of staccato is not necessary.

Chapter 8

Extensions to ANN algorithms

Some extension the basic ANN algorithms have been used to improve the performance of the networks in this application.

The ANN work began using Matlab's neural network package. Since I needed to extend the algorithms and no fitting ANN package was found in the web a small C++ ANN library was written.

8.1 SOM with output

Self-organizing map has been used as a classifier in the conductor follower. The most usual way to determine the class of a SOM neuron is by labeling the neurons manually, each neuron having a specified label (Kohonen, 1995). The most severe limitation of this approach is that the classification result is completely quantisized and the vagueness between class borders cannot be expressed.

Supervised learning of input-output pairs can be added to the architecture. This is a well-known technique ((Ritter H. & K., 1992) page 130). The units (vectors) of the map have input and output parts (figure 8.1).

The input vector for SOM during training consists of the input features

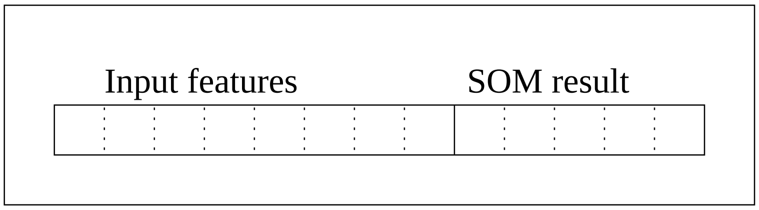

Figure 8.1: SOM vector consisting of input parameters and output parameters.

and output values. The best matching unit search ignores the output of the net. The whole map vector is trained with the input vector.

During usage the output vector is not known a priori. The best matching unit is searched from the map and the output parameters from that unit are used as outputs. It can be used like MLP. This algorithm maps certain areas of input space to output space by directly assigning average properties of the target area to aprticular SOM unit.

Classification can be performed by assigning each class a binary output vector as is typical with MLP classifiers. The SOM output vector will have values between zero and unity. Beat classification (section 4.3.3) is performed with this ANN.

8.2 Multiple MLPs

When a multi-layer perceptron is used to map a very complex space to another the number of internal parameters has to be increased. The increased number of parameters causes difficulties in estimation of the parameters - the numeric optimization.

Using multiple MLPs is a way to simplify the mapping an MLP has to perform. The MLPs can be assigned certain area of the input space. It is hoped that this simplifies the input/output mapping and hence makes finding optimal parameter values is easier. Additionally the parameters in the MLP are easier to optimize.

One way to divide the input space is by using SOM. A small SOM (2 * 2 etc.) is trained with the input data. Then the training data is split based on where in the SOM the input vector is. Each MLP is trained independently with its set of input/output vector pairs. The SOM is first used to divide input space to smaller areas. The same input vector is then used with MLP to form the final network output. Figure 8.2 illustrates this. One can also use different input vectors for MLP and SOM. The first vector can be optimized for good SOM clustering while another presents more information to MLP.

Since each MLP has its own parameters the number of total parameters increases. This may cause over-learning. To counter this one must either gather more training data or make the MLP nets smaller. In the context of conductor follower both possibilities have been successfully used.

A minor tweak is to ignore the map units that do not gather enough training data. This sets lower limit to the number of training examples for a particular MLP. Otherwise some SOM neurons may attract too few feature vectors and the corresponding MLP will be easily over-trained.

One would expect the output of the network could suffer discontinuity when sub-net changes. If a particular MLP covers only limited input/output space the local optimization may lead to anomalies in the edges of input space. In our applications effects like this have not been found. For example,

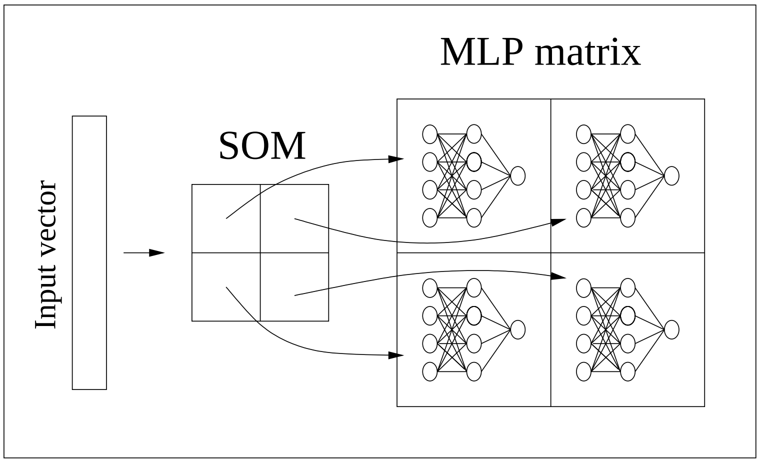

Figure 8.2: Using SOM to select the MLP to be used.

the results of figure 4.4 were created with SOM-distributed MLPs (SOM size was 2 x 2). One can hardly see discontinuous output from the network as MLP is switched (all four MLPs were used in the visible sequence).

Since each MLP optimization is an independent problem the work could be parallelized. The calculation could take full advantage of multiprocessor hardware.

8.3 MLP with cyclic output

When one is using ANN to estimate phase the discontinuity at cycle edge is often a problem. With little change in the input the output value should switch from lower bound to higher bound (figure 4.2). Some people have used two network outputs (sine and cosine) to indicate the phase (like in (Backman & Karjalainen, 1993)). This method was used in the earlier MLPs of conductor follower.

One wants to monitor the phase, not the sine and cosine of the phase. It was often found that wide amplitude movement created wider output than smaller movements. Since target outputs are of fixed amplitude the network training error does not reflect the phase error.

With minor modifications to network topology and training error function one can get the phase angle directly from the net.

The final layer of the net has two inputs. The layer contains a single arctan neuron. This means it has two inputs from the previous layer and a single output. The output is calculated as arctan of the two input values (fig 8.3). The error calculation must not compare the output of the net directly with the desired output, but use modulo calculus; eg. given a cycle in range 0-1 values 0.001 and 0.999 are very close to each other.

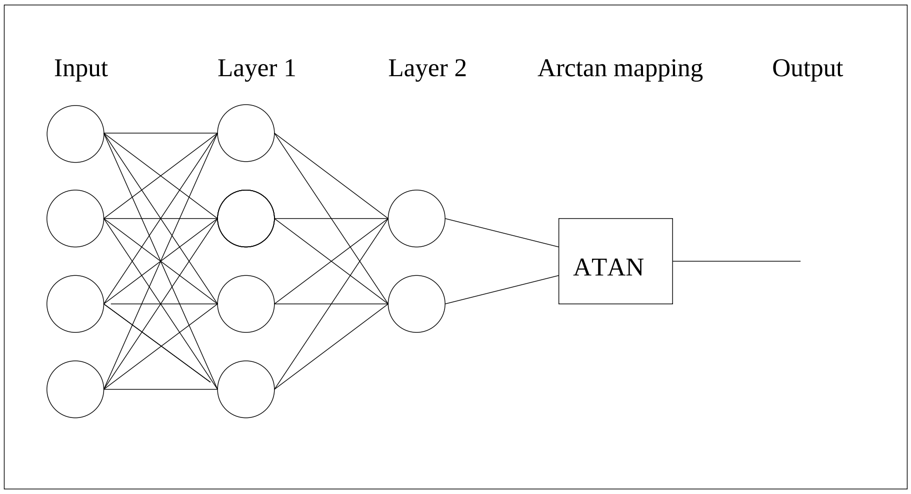

Figure 8.3: Adding a trigonometric final layer to make network output cyclic.

Using this MLP architecture yielded better phase stability in the beat prediction process. The figure 4.4 shows the output from an multi-MLP net that has been configured with arctan in the output layer.

Chapter 9

Software and hardware tools

9.1 Software

Developed software can be grouped into three parts. First group contains dozens of small scripts that are used to examine conductor motion and follower run-time results. Second group of programs and scripts is used to generate the ANNs required in the real-time usage. Finally there is the following software that performs real-time conductor following.

Programming languages are C++ and Matlab. All C++ code is based on Rami Hänninen's *LibR* virtual reality class library (Hänninen, 1999). This library provides a lot of basic functionality that the program needs - matrix operations, connection to system MIDI libraries, MIDI files, 3D graphics tools and data structures.

9.1.1 Overview of the real-time software

A flow graph of the follower was in figure 1.5. One library takes care of reading motion data. Another applies filtering to the motion data. The third stage involves objects that interpret the gestures, process their information and synthesize musical output. If multiple player models are to be used this block can be multiplied to simulate the effect of multiple musicians.

The program has a tree-like structure as in figure 9.1. The tree applies to ownership relations. The low-level modules (gesture detectors etc.) have pointers to other modules (resulting in a graph). It has been found that extending the structure is straightforward and its maintenance is not too difficult. (As historical comment: The first version of follower was in many ways a dirty hack and very difficult to manage or keep stable.)

Sometimes a given object may not exist or it can be dysfunctional. For example certain gesture detector may lack the necessary neural network to operate properly. In these cases these objects are ignored and rest of the program works the best it can. Usually crippling the program will cause

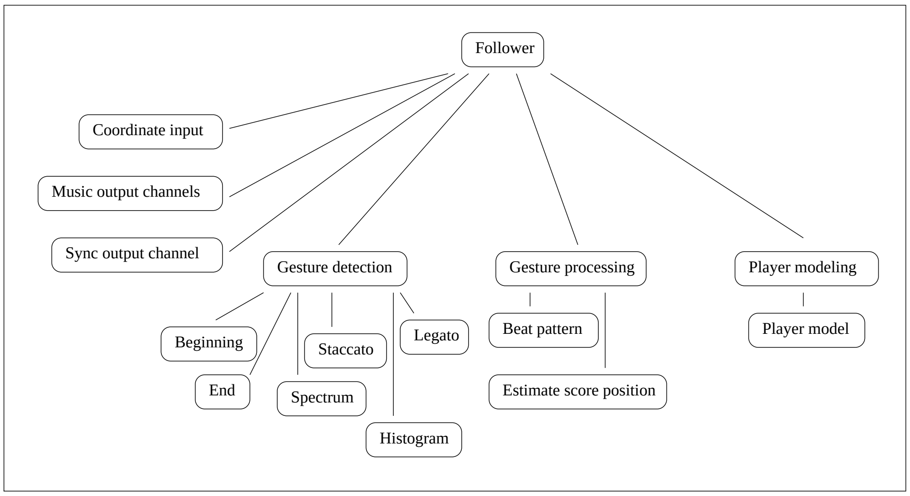

Figure 9.1: Program hierarchy. Ownership between objects is shown, but links between objects are not. There are more gesture detectors than included in this figure.

bad following results. When gathering data from the system - the feature vectors for classification for example - this setup is useful.

9.1.2 Visualization

Understanding the program operation can be greatly aided by visualization. Motion animation, graphs that display neural network performance and other parameters can be presented at the same time. Using this method one can examine how the neural networks and other analysis tools co-operate.

There are both 3D and 2D visualization tools (figs. 1.4 and 9.2) that can be used together. The 3D visualization takes place in a separate process. This guarantees that potentially slow visualization will not slow down the follower.

The visualization could also be used for other purposes. One can test how well a musician can understand a conductor if he can see only the motion of hands.

9.1.3 Operating system concerns

The follower must run with steady “frame rate”. This means that it must get CPU-time fairly often. Psychoacoustic studies indicate that timing resolution of 2-5 milliseconds is required (Lunney, 1974). This forces the same timing resolution on the scheduler of the operating system.

Our programming platform being Unix this can be a problem. Usually Unix scheduler tries to maximize program performance by allocating

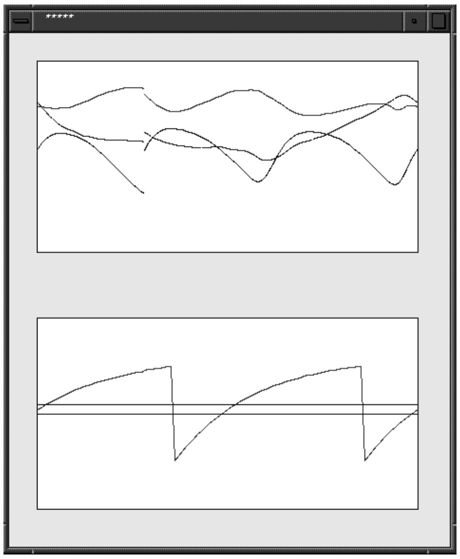

Figure 9.2: Real-time 2D visualization snapshot. Upper plot displays baton motion. The lower plot displays performance parameters, ANN outputs, tempo etc.

CPU time in fairly big chunks. Typical Unix scheduler can be instructed to allocate CPU-time using real-time priorities (POSIX1.b).

This scheme has been found very reliable. The follower is practically never late in delivering the output (all scheduler interrupts greater than 10 millisecond hick-ups are reported). Even when the system is heavily loaded (tested with 5 parallel C++ compilations) the follower gets the resources it needs.

Recent study about computing latencies indicates that IRIX (our main platform) is powerful real-time operating system when timing resolution of few milliseconds is required (Brandt Eli, 1998). Our own experience shows that the same can be said about Linux, the second supported platform (section 9.3).

9.2 Computational performance

In PC with 233 MHz Pentium processor the real-time usage takes less than 1 % of machines processor time. This metric was measured with standard system tools, command "top" and system CPU timing function "clock". For unknown reason the comparable load of an Silicon Graphics (SGI) O2 with MIPS R10000 processor is over 7 % (the SGI should be faster than the PC). Thorough performance analysis has not been made, since lack of performance has not been a problem.

In neural network calculations the SGI is a lot faster. Program profiling does not show any obvious performance sinks in SGI. Calculating the neural networks and finding proper parameter values may take several hours of hand-tuning and computation.

9.3 Hardware

The original programming platform was Silicon Graphics. It was chosen because of its ease of programming and powerful media API. A Linux port was made because a low-cost version of the system was needed for Heureka (appendix A).

The work-station and magnetic tracker communicate via standard local area network. This can be a problem when there is a lot of traffic in the LAN. Luckily the network in TML lab is usually able to handle the IP traffic gracefully.

The sound may be synthesized with any synthesizer. Both custom algorithms and commercial hardware have been used for sound synthesis.

Chapter 10

Conclusions

Making of a conductor follower has been presented. The follower uses many standard tools of information technology. 3D geometry manipulation transforms the coordinate systems to desired format. 2D and 3D graphics are used to visualize the motion and internal state of the follower. DSP tools are used to calculate motion features. Normal FIR filters as well as median, minimum and maximum filters are applied on the motion data for feature extraction. Further feature extraction is done using FFT and histograms. The gesture identification is based on ANNs and heuristic rules. Matching the identified gestures to expected conducting style is performed with heuristic matching rules and explicit computation. Heuristic rules mimic the musician.

Detailed evaluation of all methods cannot be done in such short time. The focus was on the whole - interplay between components - not on individual components themselves.

It has been found that the system is capable of tracking conductors who use clear gestures. The system has built-in implicit expectation of how conducting is performed, there are beats, beat patterns and fermatas. Should a conductor choose a completely different method the system will fail to react musically. It is easily misled if the motion does not match the training data of the ANNs. In optimal circumstances it can react faster than a musician would, but with lesser musicality. In comparison humans can adapt to new gesture immediately. The rich base of common knowledge makes it possible for conductor to perform gestures that would be impossible for an automatic system to comprehend instantly.

During the testing I received mixed response from conductors and conductor students. Some were enthusiastic and would have liked to have a conductor follower at home. Some doubted whether a system like this would be worth anything. Many conductors did find the visualization (figs. 1.4 and 9.2) interesting and useful. While the tracking does not cover full body the visualization does show explicitly many errors a conductor can make.

10.1 Further work

Project like this will not reach level of perfection in ten years. The most obvious place for improvements is the player modeling section. More experiments with ANN-controlled playing will be made.

Group behaviour is the next logical step to make in order to make the system more orchestra-like. Representing the musical intelligence with artificial neural networks could be a major improvement.

The system could also react to other factors besides the conductor. Such input could be the music physical musicians play. Thus the system could react to real musicians in performance situation.

10.2 Workload

One would hope to work on crafting new algorithms and approaches to the problems. In practice coding the algorithms takes significant amount of time. As January 1999 ends I have about 20000 lines of C++ and 7000 lines of Matlab code (or rather, this many lines in the files - including blank lines and documentation). Of 20k lines of C++ code half are directly used by follower and the rest contain neural network libraries etc.

A lot of effort was spent on testing a particular approach eg. verifying the results. Any problem could have many sources. Typically one would wonder if the problems are caused by bad algorithm, parameters, or software bugs.

Debugging took probably more than third of the time. There are several ways one can go wrong, code bugs (shell scripts, C++ and Matlab), component interaction problems (faulty resource files, incorrect call methods between scripts), faulty programming environment (Unix paths, links etc) and mistaken coordinate tracking (faulty 3D transformations).

Due to time limitations few components of the follower could be examined in detail. The large number of components that make up the follower effectively limits the time one can spend on each.

10.3 Plan vs. results

Target timetable was outlined in section 1.4. The schedule was altered on the fly more than once. Some parts arrived before expected while some features came late. As expected the work finished later than planned.

Building the system for Heureka demonstration stand (appendix A) took some additional time.

Since the I was familiar with the work the beginning was painless and I could start at full speed. A lot of code (about 10 000 lines) existed before the work for this masters thesis started. No time needed to be spent learning new tools or working environment.

10.4 Credits

This work has been funded by the Academy of Finland. The work has been done in Telecommunications Software and Multimedia Laboratory (TML) in Helsinki University of Technology (HUT).

The Heureka demonstration was funded by Heureka, VTI Hamlin, Tekes and Veikkaus.

The author is part of DIVA group (DIVA, 1999) in Helsinki University of Technology.

References

Ascension. 1999. *Ascension Technology Corporation.* Company's WWW home page, URL: http://www.ascension-tech.com/.

Backman, J., & Karjalainen, M. 1993 (April). Modelling of human directional and spatial hearing using neural networks. *Pages 125–128 of: Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, vol. 1.

Brandt Eli, Dannenberg Roger B. 1998 (October). Low-Latency Music Software Using Off-the-shelf operating ststems. *Pages 137–141 of: Proceedings of the International Computer Music Conference.* International Computer Music Association, Ann Arbor, Michigan USA.

Brecht, B., & Garnett, G. 1995. Conductor Follower. *In: Proceedings of the International Computer Music Conference.* San Francisco: Computer Music Asociation.

Bregler, C. 1997. Learning and Recognizing Human Dynamics in Video Sequences. *In: IEEE Conf. on Computer Vision and Pattern Recognition.*

Bresin, R., & Friberg, A. 1998. Emotional expression in music performance: synthesis and decoding. *Pages 85–94 of: KTH, Speech, Music and Hearing, Quarterly Report*, vol. 4. Stockholm, Sweden: Kungl Tekniska Högskolan - Royal institute of technology.

Canazza Sergio, De Poli Giovanni, Di Sanzo Gianni Vidolin Alvise. 1998 (October). A model to add expressiveness to automatic musical performance. *Pages 163–170 of: Proceedings of the International Computer Music Conference.* International Computer Music Association, Ann Arbor, Michigan USA.

Carosi, P., & Bertini, G. 1992. Light Baton: A System for conducting computer music performance. *In: Proceedings of the International Computer Music Conference.* San Francisco: Computer Music Asociation.

Chafe, C., & O'Modhrain, S. 1992. Musical Muscle Memory and the Haptic Display of Performance Nuance. *Page 428 of: Proceedings of the Inter-*

national Computer Music Conference. San Francisco: Computer Music Asociation.

Coda. 1998. *Coda Music Technology Inc. SmartMusic*. Document named sm.htm available via internet from the server "www.codamusic.com." URL http://www.codamusic.com/sm.html.

DIVA. 1999. *DIVA WWW home page*. Document named index.html available via internet from the server "www.tcm.hut.fi". URL http://www.tcm.hut.fi/Research/DIVA/.

Fourier, Jean-Baptiste Joseph. 1822. *Théorie analytique de la chaleur (Analytic Theory of Heat)*.

Friberg, A. 1995 (May). *A Quantitative Rule System for Musical Performance*. Ph.D. thesis, Royal Institute of Technology, Stockholm, Sweden. Available vie URL: http://www.speech.kth.se/music/publications/ruleref.htm.

Friberg, A., & Bresin, R. 1997. Automatic Musical Punctuation: A Rule System and a Neural Network Approach. *Pages 159–163 of: Proceedings of KANSEI - The Technology of Emotion AIMI International Workshop*. AIMI - Associazione di Informatica Musicale Italiana.

Hamlin., VTI. *VTI Hamlin Acceleration Sensors*. Document named sensors.html available via internet from the server "www.vti.fi". URL: http://www.vti.fi/Products/Sensors/sensors.html.

Hänninen, Rami J. 1999. *LibR - An object-oriented software library for real-time virtual reality (to be published), Licentiate Thesis*. Helsinki University of Technology.

Hiipakka J., Hänninen R., Ilmonen. T Napari. H Lokki T. Savioja L. Huopaniemi H. Karjalainen M. Tolonen T. Välimäki V. Välimäki S., & Takala, T. 1997. Virtual Orchestra Performance. *Page 81 of: Visual Proceedings of SIGGRAPH'97*. Los Angeles: ACM SIGGRAPH.

Ilmonen, T. 1998 (7-9 September). Tracking Conductor of an Orchestra Using Artificial Neural Networks. *In: Human and Artificial Information Processing. Finnish Conference on Artificial Intelligence, STeP'98.*

Keane, D., & Gross, P. 1989. The MIDI Baton II. *In: Proceedings of the International Computer Music Conference*. San Francisco: Computer Music Asociation.

Kohonen, Teuvo. 1995. *Self-Organizing Maps*. Berlin: Springer.

Large, E. W., & Kolen, J. F. 1994. Resonance and the perception of musical meter. *Connection Science*, **6**(1), 177–208.

Lunney, H. M. W. 1974. Time as Heard in Speech and Music. *Nature*, **249**, 592.

Marrin, Teresa. 1996. *Toward an Understanding of Musical Gesture: Mapping Expressive Intention with the Digital Baton.* M.Phil. thesis, MIT.

Marrin Teresa, Picard Rosalind. 1998 (October). The "Conductors Jacket": A Device for Recording Expressive Musical Gestures. *Pages 215–219 of: Proceedings of the International Computer Music Conference.* International Computer Music Association, Ann Arbor, Michigan USA.

Mathews, M. 1991. The Radio Baton and Conductor Program, or: Pitch, the Most Important and Least Expressive Part of Music. *Computer Music Journal*, **19**(4).

McElheran, B. 1989. *Conducting Technique for Beginners and Professionals.* Oxford/New York: Oxford University Press.

Morita H., Watanabe H., Harada T. Ohteru S., & S., Hashimoto. 1990. Knowledge information processing in conducting music performer. *In: Proceedings of the International Computer Music Conference.* San Francisco: Computer Music Asociation.

Parncutt, R. 1997. Modeling piano performance: Physics and cognition of a virtual pianist. *In: Proceedings of the International Computer Music Conference.* San Francisco: Computer Music Asociation.

Ritter H., Martinez T., & K., Schulten. 1992. *Neural Computation and Self-Organizing Maps - An Introduction.* Computational and Neural Systems Series. Addison-Wesley.

Schalkoff, J. 1992. *Pattern Recognition: Statistical, Structural, and Neural Approaches.* J. Wiley.

Takala T., Hänninen R., Välimäki V. Savioja L. Huopaniemi J. Huotilainen T., & M., Karjalainen. 1996. An Integrated system for Virtual Audio Reality. *In: Preprint 4229 (M-4), Copenhagen: 100th AES Convention of Acoustic Engineering Society: 46.*

Tobey, F., & Fujinaga, I. 1996. Extraction of Conducting Gestures in 3D Space. *In: Proceedings of the International Computer Music Conference.* San Francisco: Computer Music Asociation.

Toiviainen, P. 1997. Modelling the Perception of Metre with Competing Subharmonic Oscillators. *In: Proceedings of the Third Triennial ESCOM Conference.*

Toiviainen, P. 1998. Intelligent Jazz Accompanist: A real-time system for recognizing and accompanying musical improvisations. *In: Human and Artificial Information Processing, Finnish Conference on Artificial Intelligence.*

Usa, S., & Yasunori, M. 1998 (October). A Multi-modal conducting simulator. *Pages 25–32 of: Proceedings of the International Computer Music Conference.* International Computer Music Association, Ann Arbor, Michigan USA.

Välimäki, V., & Takala, T. 1996. Virtual Musical Instruments – Natural Sound Using Physical Models. *Organised Sound*, **1**(2), 75–86.

Watanabe T., Chil-Woo L., Tsukamoto A. Masahiko Y. 1996. A Method of Real-Time Gesture Recognition for Interactive Systems. *Pages 473–477 of: International Conference on Pattern Recognition.* IEEE.

Appendix A

The DIVA multimedia system

The Digital Interactive Virtual Acoustics (DIVA) group aims at creating virtual environment with powerful audio-visual impact.

Currently the system includes a band of four animated virtual musicians. The band can be controlled by conducting with the baton. The follower interprets the motion and instructs the animation engine to behave accordingly. The animation can be presented via monitor, screen (data projector), or head-mounted display.

The sounds are synthesized with physical models or traditional sample-based synthesizers depending on the computational resources. The sound can be further processed by applying auralization algorithms (for 3D sound) in real-time. The auralization can be listened with headphones, two loudspeakers or loudspeaker array depending on the application and resources.

The animation mechanism is described in detail in Rämi Hänninens work. The components of DIVA are presented in greater detail in (Takala T. & M., 1996). More information and up-to-date list of publications can be found in WWW (DIVA, 1999).

A.1 The SIGGRAPH'97 demo

The DIVA system has been presented in SIGGRAPH'97 (computer graphics) conference (Hiipakka J. & Takala, 1997). It was demonstrated for one week in maximum configuration with three SGI workstations.

People from ordinary folk to choral conductors conducted the follower. Most musically inclined people were quite happy with it and even ordinary people could get the virtual orchestra playing. Normal musicians were pleased to find out how to control the virtual orchestra without realizing that the control differs from real band.

Figure A.1: The DIVA band in a subway station.

Note that this event took place long before the work for this masters thesis began.

A.2 The Heureka demo

A DIVA demo was installed in Finnish science center Heureka November 1998. The demonstration stand will be there for more than one year and the system may begin world tour after that time.

The system uses a single PC to run band animation and perform conductor following. Any show visitor can step up, take the baton and begin conducting. Since most show visitors are children the system must withstand considerable stress. At the same token it need not be very realistic since the average user does not have a clue of how to conduct an orchestra.

Being a standard installation we needed to use cheap off-the-shelf components. Thus hardware failures can be fixed easily. Rather than using magnetic motion tracking accelerometers (see 3.5) were used. The computer was a standard Linux-PC.

The PC had 3D acceleration card for graphics, multichannel A/D card for accelerometers and sound card for MIDI output. The sound was synthesized with a commercial synthesizer. A baton was made of plastic pipe and the accelerometers were inside the pipe mounted on an ice-cream stick.

For me the Heureka demo meant experimenting with the accelerometers and pushing the follower software to a stable production version quickly. Additional time was spent building the system and fixing final bugs in Heureka.

The Heureka staff has been very pleased with the popularity of the stand. The system has been up for more than three months now without major problems. We have been positively surprised by the stability of the system.