Expressive Performance of Polyphonic Piano Music using Hierarchical Structure

Bastiaan J. van der Weij
5922151

Bachelor thesis
Credits: 15 EC

Bachelor Opleiding Kunstmatige Intelligentie

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervisor
Prof. dr. ir. R.J.H. Scha

Institute for Language and Logic
Faculty of Science
University of Amsterdam
Science Park 904
1098 XH Amsterdam

June 24th, 2010

Abstract

This paper introduces a system for expressively performing transcribed music (scores) using statistical learning from a dataset. The dataset that is used contains performances aligned to the corresponding score. Expression is considered to be the way in which the performance deviates from the score. Expressive actions are predicted from links found between musical structure and expression. Previous expressive actions are also incorporated in the predictions. Musical structure will be automatically extracted from the scores using a set of formal rules that assign a hierarchical grouping structure to the music. To system will be able to deal with polyphonic music by splitting the scores into melody and harmony.

Keywords: performance rendering, musical structure, constituent structure, polyphonic piano music

1 Introduction

A music performance involves a performer and an audience. The performer tries his best to perform music to the audience in a way that pleases the audience. In western art music tradition it is conventional that the performer provides his rendition of a musical score: a detailed transcription of the music in musical notation. A performer should be skilled enough to accurately reproduce the music in the score, but what sets a good performer apart from other performers and drives people to pay money to see a musical performance is his ability deviate from the score and interpret it in a way that pleases his audience. In fact, it is generally agreed on that for most western art music, a perfect reproduction of the score sounds very unpleasant. The performer makes the performance sound 'natural' and expresses his musical knowledge and perhaps feelings in the performance, the resulting performance is expressive.

It should be clear that performing a musical piece is not a trivial task. Theoretically it would be possible to construct robotic performers with perfect mastery of their instruments as well as perfect ability to reproduce scores. Yet there is still large gap between a perfect score reproduction by robots and a good sounding performance (the perfect reproduction sounds, indeed, robotic). Performance rendering systems attempt to bridge this gap.

Deviations from a score in a performance can be described as the end result of a number of processes, the most important of which is called expression. It is not immediately clear where expression comes for or even what it is. It is not the within the scope of this thesis to elaborate widely on the complex phenomenon of expression. Instead, it will be assumed that expression is strongly linked to musical structure and that a computational model using this structure can to some extend adequately model expression. The quality of the system should be an indication of the correctness of this assumption.

Early performance rendering systems used rule-based approaches to recognise small local structures that are used as indications to add certain kind of expression. A prominent ongoing rule based system, *Director Musices* uses 30 rules with weighting parameters that take music features as input and produce expressive actions as output. The rules where created with the help of musical experts. *Pop-E* is a more recent rule based system that uses aspects of Lerdahl and Jackendoff's Generative Theory of Tonal Music(GTTM) [Introduce this].

Gerhard Widmer is doing research using a large dataset of performances aligned to scores.

As datasets become more available, machine learning approaches are gaining popularity. However, very few approaches make extensive use of structure. Some machine learning systems like the *Music Interpretation System* make use of some aspects of GTTM and Widmer uses musical *closure* [Narmour, introduce this properly] as a feature in his succesful *YQX* system. This means that only the level of structure defined by closure is used by this system. It seems that higher level structure that would indicate for example repetition of parts plays an important [This argument is important and needs more elaboration] role in expression.

This thesis introduces machine learning system that makes of local structure as well as higher level structure to generate expressive performances of polyphonic piano music. Section X will introduce Y, section P will elaborate on Q and finally section Z will discuss the results.

Evaluation. Performance context, adaptability, creativity. Subjective listening, correlation.

1.1 Related Work

Move part of the introduction here?

2 Approach

2.1 Musical structure

When listening to music, the listener assigns a certain hierarchical structure to the music perceived: Notes make up phrases, phrases make up themes, themes make up parts and parts make up the piece. This structure is usually accentuated in a performance. The most obvious example of this is slowing down at the end of a piece. On a lower level, transitions between themes may be marked by slowing down, changing volume or changing speed. Although structure can be accentuated in a performance, listeners, and trained listeners even more so, will also be able to recognise structure in an expressionless reproduction of the score, or when quietly singing the music in their heads. This suggests that it is possible to extract structure from a merely score. It also seems likely that knowing the structure will help creating an expressive performance.

The delta framework

The delta framework is based on the idea that small differences indicate low level structural transitions and large differences indicate higher level structural transitions.

Combining delta trees In his MSc dissertation, van den Berg suggests using *yeild rules* to combine trees generated by different delta functions. The yeild rule states that nodes in a set of trees can be connected if they are *yeild equal*, that is, if the their set of leaf nodes they recursively contain is equal. The result

is a multi-tree, in which the top note is always connected and intermediate nodes may or may not be connected.

Unfortunately, a multi tree is hard to interpret since

Using repetition to improve structural analysis Repetition be formulated in terms of the delta framework to provide greater generality. It seems desirable that we consider a transposed repetition of an earlier theme to be a repetition nevertheless. The same holds for rhythmic repetition: it may be that a particular rhythm is repeated, we want to treat this as a repetition too. The delta framework allows this intuition to be implemented easily: we only look at repetition in the deltas between notes. If we take for example the inter ——— ——————— Consider the musical sequence $S = \{n_0, n_1, n_2, n_3\}$ is ambiguous and has two possible assignments of constituents: $S_0 = \{n_0\}, S_1 = \{n_1, n_2, n_3\}$ or $S'_0 = \{n_0, n_1\}, S'_1 = \{n_2, n_3\}$. The notion off repetition can help us here. Consider the case that S'_1 is a repetition of S'_0 but S_1 is not a repetition of S_0, this tells us that the split $S = \{S'_0, S'_1\}$ is more likely than the split $S = \{S_0, S_1\}$.

Using repetition to create constituent classes Repetition can also be used to make the constituent assignment more interesting. What follows is a method that identifies which consituents are repetitions of which other constituents.

Algorithm 1 Bottom up tree traversal and similarity relations

```
for depth ∈ {depth(tree), ··· , 0} do
  for c ∈ {c|depth(c, tree) = depth} do
    for class ∈ C do
      if similar(c, class) then
        c.class = class
      end if
    end for
    if c.class = null then
      C.append(createClass(c))
      c.class = createClass(c)
    end if
  end for
end for
```

A resulting tree could look like figure 1

2.2 Performance Model

2.2.1 Smoothness of expression

To avoid sudden changes in expression that sound weird, you would want a model that describes what a sensible sequence of expressive actions is. It may be sensible to train this model on all available data since this model should not describe what expressive actions are appropriate for one specific pianist or composer, it should rather describe what sequences of expressive actions are acceptable in general. To achieve this, a continuous hidden markov model, trained on all performances in the dataset, will be used.

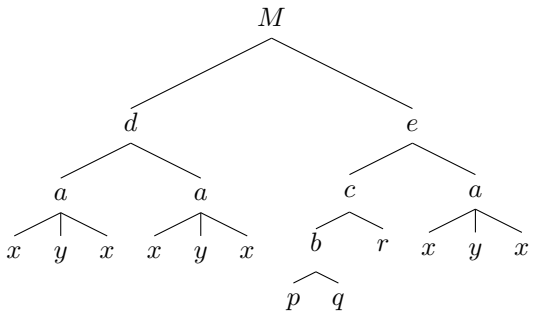

Figure 1: Example tree with similarities

Let D be a collection of compositions. Let a composition, c be a collection of expressive actions, a and let an expressive action be a tuple (l, t) describing the loudness deviation, timing deviation (and/or local tempo?) of the note. The hidden markov model can be described as follows:

$$
\begin{aligned}
D &= \{c_0, c_1, \cdots, c_n\} \\
c_i &= \{a_0, a_1, \cdots, a_n\} \\
a_i &= (l, t) \\
P(c) &= P(a_1|a_0) * P(a_2|a_1, a_0) * \cdots * P(a_n|a_{n-1}, a_{n-2}, \cdots, a_0))
\end{aligned}
$$

2.2.2 Expressive timing

Honing and Desain, Honing and timmers, Honing.

2.3 Learning

3 Method

4 Evaluation and Results

4.1 Subjective Listening

4.2 Correlation

5 Conclusion

6 Future Work

References

[1] H. Honing. Expresso, a strong and small editor for expression. *San Francisco: International Computer Music Association*, pages 215–218, 1992.