

Generating Musical Performance with Director Musices

Anders Friberg, Vittorio Colombo, Lars Friden,
Johan Sunberg

Leo Wang, ISE575, Spring 2006 1

Agenda

- ◆ Introduction
- ◆ System Infrastructure
- ◆ Input & Output
- ◆ Rules
- ◆ User Interface & Examples
- ◆ Conclusion

Leo Wang, ISE575, Spring 2006 2

Introduction

- ◆ Director Musices is a rule-based program
- ◆ Users define rules in the performance aspects such as phrases, articulation
- ◆ By manipulating rule parameters, the same music piece can have different interpretations

Leo Wang, ISE575, Spring 2006 3

Agenda

- ◆ Introduction
- ◆ System Infrastructure
- ◆ Input & Output
- ◆ Rules
- ◆ User Interface & Examples
- ◆ Conclusion

Leo Wang, ISE575, Spring 2006 4

System Infrastructure

```

graph LR
    Score --> Rules
    Rules --> Performance
    K_values[K values] --> Rules
    subgraph Rules_Box [Rules]
        direction TB
        R_text[e.g. intonation, ensemble]
    end
  
```

◆ Each rule has a global quantity parameter k that controls the magnitude of the rule

◆ The selection of rules, k values, and internal rule values decide the expression

Leo Wang, ISE575, Spring 2006 5

Agenda

- ◆ Introduction
- ◆ System Infrastructure
- ◆ Input & Output
- ◆ Rules
- ◆ User Interface & Examples
- ◆ Conclusion

Leo Wang, ISE575, Spring 2006 6

Input & Output

- ◆ Three Formats
 - Scores, a simple text-based custom format (.mus)
 - Performances, similar to the score format with performance variables added
 - MIDIs
 - ◆ Each track is assumed to contain only one voice
 - ◆ Key velocities are not considered

Leo Wang, ISE575, Spring 2006 7

Agenda

- ◆ Introduction
- ◆ System Infrastructure
- ◆ Input & Output
- ◆ Rules
- ◆ User Interface & Examples
- ◆ Conclusion

Leo Wang, ISE575, Spring 2006 8

Rules

- ◆ The rules are categorized to three types
 - Grouping rules that mark boundaries between smaller and larger tone groups
 - Differentiation rules that increase differences between categories
 - Ensemble rules for the interaction between musicians in an ensemble

Leo Wang, ISE575, Spring 2006

9

Rules

- ◆ The input is stored as a score object containing a list of track objects, which in turn contains a list of segments
- ◆ Track objects correspond to melodic parts, segments represent one note or one chord

Leo Wang, ISE575, Spring 2006

10

Rules

- ◆ Each rule has a set of performance variables, expressed in physical measure
 - E.g., interonset duration(dr), sound level(sl), vibrato amplitude(va)
- ◆ When several rules change the same performance parameter, the total change is the sum of changes from each rule.

Leo Wang, ISE575, Spring 2006

11

Rules

- ◆ Each rule can be applied to different level of objects
- ◆ Rules are written in Lisp

Leo Wang, ISE575, Spring 2006

12

Rules

- ◆ An example of the rule


```
(defun phrase-rule (k)
  (each-note-if
    (not (last?))
    (next 'phrase-start)
    (then
      (add-this 'dr (* 40 k))
    )))
  (each-track
    (set-this-dr ;
      (* (this-dr) 1.2) ))
```

Leo Wang, ISE575, Spring 2006

13

Agenda

- ◆ Introduction
- ◆ System Infrastructure
- ◆ Input & Output
- ◆ Rules
- ◆ User Interface & Examples
- ◆ Conclusion

Leo Wang, ISE575, Spring 2006

14

Agenda

- ◆ Introduction
- ◆ System Infrastructure
- ◆ Input & Output
- ◆ Rules
- ◆ User Interface & Examples
- ◆ Conclusion

Leo Wang, ISE575, Spring 2006

15

Conclusion

- ◆ A interesting application with an easy-to-use GUI
- ◆ Rules might be difficult to create

Leo Wang, ISE575, Spring 2006

16