ANNOTATING MUSICAL SCORES IN ENP

Mika Kuuskankare
Department of Doctoral Studies
in Musical Performance and Research
Sibelius Academy
Finland
mkuuskan@siba.fi

Mikael Laurson
Centre for Music and Technology
Sibelius Academy
Finland
laurson@siba.fi

ABSTRACT

The focus of this paper is on ENP-expressions that can be used for annotating ENP scores with user definable information. ENP is a music notation program written in Lisp and CLOS with a special focus on compositional and music analytical applications. We present number of built-in expressions suitable for visualizing, for example, music analytical information as a part of music notation. A Lisp and CLOS based system for creating user-definable annotation information is also presented along with some sample algorithms. Finally, our system for automatically analyzing and annotating an ENP score is illustrated through several examples including some dealing with music information retrieval.

Keywords: Music representation, annotating, symbolic notation.

1 OVERVIEW

Expressive Notation Package (ENP, [1, 2]) is a music notation program that is designed for displaying scores using the common Western music notation. ENP has been used in several research projects ranging from computer aided composition to controlling virtual instruments. A special focus has been given to compositional and computer-assisted music analysis applications.

ENP has a graphical user interface that allows musical objects to be edited directly with the mouse. It supports two fundamental notational styles, i.e., mensural and non-mensural notation, and a number of special notational styles such as time notation, frame notation, etc.

Representation of musical units must offer ways of making annotations, giving names, making comments, adding images or diagrams, providing links to informative resources on the web, etc [3]. ENP provides a collection of standard and non-standard notational attributes (e.g, articulations) called ENP-expressions. Furthermore, it offers a set of attributes that can be used to represent analytical information or other user-defined annotations as a part of a musical texture. In addition to their traditional use, ENP-expressions can be used in wide range of applications: (1) display music theoretical analysis information, e.g, annotate motives, harmonic progressions, etc; (2) visualize specialized analytical information, such as Schenker graphs, or pitch-class set theoretical information; (3) attach arbitrary textual annotations, names, or comments to objects; (4) dynamically inspect and visualize data contained by notational objects, i.e., duration, velocity, start-time, etc.; (5) add instructions to tutorials, documentation, or presentations, etc.

All ENP-Expressions can access the data contained by the notational objects they are associated with. This allows to design dynamic expressions that can automatically display relevant information about themselves and their musical context. It is also possible to use a scripting language called ENP Script [4] as an algorithmic complement to the manual approach where the user inserts ENP-expressions by hand. This is useful when building, for example, computer-assisted music analysis applications or automatically annotating a musical score.

In the following, we present and discuss the annotation possibilities of ENP in more detail. We start with a brief introduction of ENP-Expressions and the notational output of ENP. Section 3 is the main part of this study. Here we present a wide range of annotation devices along with some example scores. Section 4 gives a brief look at the possibilities of automatic music information retrieval and annotation. Section 5 presents some conclusions and ideas for future work.

2 ENP-EXPRESSIONS IN BRIEF

Every expression is attached to some musical object or to a group of objects. Currently these objects can be either notes or chords (see [5] for a description of the object hierarchy in ENP). All the expressions are aware of their musical context and can automatically adjust their position and graphical appearance accordingly. New expressions can be created through a textual interface using Lisp and CLOS or by using a set of specialized editors inside ENP.

Figure 1 gives a concise overview of the notational possibilities of ENP in a modern context. The example is written in non-mensural notation (time notation) and contains various expressions, special note heads and playing styles.

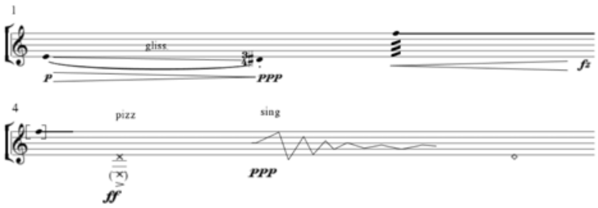

Figure 1: An example of the notational output of ENP.

3 ANNOTATIONS OVERVIEW

3.1 Groups

Groups are multi-purpose expressions that can be used to mark continuous passages in music. The visual appearance of the groups can be adjusted to suit the current application. Groups can also contain user definable textual information. The next example (Figure 2) shows one use for groups to annotate motives in a score. The groups, in this case, are represented as brackets with labels.

Figure 2: Motivic analysis information inserted in the score with the help of groups (J.S. Bach: Invention no. 1).

3.2 Text Expressions

Textual information can be attached to both chords and notes. The expressions can be either static or dynamic. These expressions typically contain some written instructions or annotations made by the user. Dynamic ones, in turn, can adjust the visual appearance and printed information by analyzing their musical context.

3.2.1 Static Text Expressions

Next, we give two examples of static text expressions. In Figure 3 the user has inserted text expressions to the notes and entered the note names accordingly. This kind of behavior is acceptable when the data contained by the notes is not expected to change (e.g., through transposing). In this case the expressions show their association with the notes by drawing a dotted line connecting the expression to the corresponding note.

Figure 3: Some static text expressions inserted in a score. The association between the expression and the notational object is displayed by a dotted line

Another example of a static expression is the *window expression*. This device is useful for giving focus to an abject at any position on the screen. These expressions apply only to the current window so they are suitable for inserting markings only to one static page at a time, as in case of tutorials or presentations.

Figure 4: Window expressions can be used to mark any position in the score.

3.2.2 Dynamic Text Expressions

Dynamic expressions can adjust their visual representation (e.g, position, shape, color, text) automatically. This ensures that the score is always up-to-date. This is also useful when dealing with analytical problems: the analysis information remains correct even if the music is edited. For this purpose we introduce a special expression called *annotation*. Dynamic expressions are also useful to display relevant information accessed from the objects they are attached to. Almost any attribute of the notational objects can be made visible in this way. The attribute can be selected from an automatically generated list that is specialized for both chords and notes. The current set includes, for example, *midi*, *start-time*, *diatone*, *duration*, *velocity*, *channel*, etc.

Figure 5 gives one example of the behavior of dynamic annotation. The data displayed is read directly from the objects. In this case the data shown is the value found in the *midi* slot of the note.

Figure 5: Dynamic expressions can adjust their visual appearance automatically.

The user can also write additional methods for displaying the result of some user-definable code with the help of annotation expressions. ENP provides an *add-to-annotation-library* macro that can be used to add user definable annotations to a special library. These annotations are immediately usable and can be inserted into the score through context sensitive menus. The parameters of add-to-annotation-library are: (1) the type of the object the expression is associated with, (2) the name of the expression, and (3) the function that constructs the displayed data.

Below we give two relatively simple examples of dynamic text annotations created with the help of the add-to-annotation-library macro.

First, we define a new annotation that can be attached to notes (1). In (2) we define the name of the annotation (*note-name*) and in (3) we give the function that constructs the textual information depending on the midi value of the note. Figure 6 gives the corresponding score.

```
(add-to-annotation-library
 :note                                               ;(1)
 note-name                                           ;(2)
 #'(lambda(note)                                     ;(3)
     (format () "note name: ~a"
             (midi-to-note-name (midi note)))))
```

Figure 6: A dynamic text annotation, displaying the note name, created with the help of the add-to-annotation-library macro.

As can be seen in Figure 7, the needed user-interface components (context sensitive menus) are also added automatically. The annotations can be inserted in the score as any built-in ENP-expression (see the ”User methods” menu in Figure 7). The two menus marked as ”Methods” and ”Slots” contain a list of available slots and predefined methods applicable to the object in question.

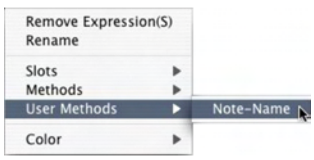

Figure 7: Menus for user definable annotations are added automatically by the system.

The second example deals with displaying some pitch-class set theoretical analysis information. Here we use a dynamic text expression to mark the pitch-classes above the individual notes in a twelve-tone row. In Figure 8 we have two twelve-tone rows. The former one is the row used in Alban Berg's Lyrische Suite in original form, and the latter one, in turn, is its transposed version.

The expression definition is given below. There are only few changes when compared to the previous example. The most important one is that in this case we take the modulo 12 of the midi value contained by the note, as can be seen in (1).

```
(add-to-annotation-library
 :note
 pc-name
 #'(lambda(note)
     (format () "~a"
             (mod (midi note) 12))))           ;(1)
```

Figure 8: The twelve-tone row of Alban Berg's Lyrische Suite (above) and its transposed version (below). Note, that the pitch classes shown by the dynamic annotations (pc-name) are displayed correctly in the transposed row form.

3.3 Score Expressions

Score-expressions can be used to visualize discontinuous information in the score (Figure 9). This is a convenient way to display information that is scattered across different parts. Arbitrary vertical relations can be made visible for analytical purposes. The shape of the score expression can also be selected. The next example illustrates the use of the box-shaped score-expression to draw a focus around two chords in the score. This kind of an expression is useful for revealing groups containing several entries.

Figure 9: Focus drawn around two chords using a box-shaped score-expression (Arnold Schoenberg: Sechs Kleine Klavierstücke, op. 19).

A line-shaped score-expression is best suited to indicate a relation between two notational objects as can be seen in the Figure 10. Here some music analytic information (cross relation) is inserted in the score. In addition to the line connecting the two notational objects the expression can contain some user definable text.

Figure 10: Voice-leading analysis information displayed using the line-shaped score-expression.

3.4 Pitch-class Clock Face

The Pitch-class clock is a highly specialized expression that can be used to visualize set theoretical information as a part of the musical texture. It is a dynamic graphical object that displays the pitch-class set of a selected chord as a pitch-class clock face. The pitch classes that are present in the set are highlighted (Figure 11).

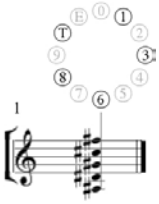

Figure 11: A pitch-class clock face is a specialized expression for visualizing pitch-class set theoretical information.

3.5 Highlighting

Highlighting is a visualization device that can be used for demonstration purposes. Relevant information in the score can be highlighted to focus the attention to a specific detail in the score (Figure 12). The areas outside the highlighted area are dimmed but still clearly visible, thus preserving the view to the whole musical context at the same time. The highlighted area can be moved and re-sized by the user.

Figure 12: A highlighted area created in the bassoon part of Rite of Spring by Igor Stravinsky.

4 AUTOMATICALLY ANNOTATING A SCORE

In this section we briefly describe how to automatically analyze and annotate an ENP score by using a built-in scripting language, ENP script. A script is usually used to automate complex or advanced tasks within the program. In a music notation program it is useful to allow frequently repeated operations (e.g., applying an articulation pattern) to be automated by using scripts.

The scripting examples in this section are explained only briefly. It is not our intention to explain the syntax of ENP Script but to illustrate the potential of the system in general.

In our first example we examine the vertical aspect of a piece of music by performing a simple harmonic analysis.[1] The result of the analysis, the name of the pitch-class set of each prevailing harmony, is printed on the bottom of the score. The naming of the pitch-class sets follows the conventions introduced by Allen Forte [6]. Figure 13 gives a score with the set theoretical analysis information. The script used to generate the information is shown above the score. This kind of use of a script resembles the container-iterator idea found, for example, in CPNView [7, 8]. Here we iterate through every harmonic situation in the score (see the :harmony keyword in (1) in the example script). A special Lisp function *add-expression* (2) is used to insert into the score a group expression displaying the analysis information. The expression is attached to the lowest note of the chord (3). In (4) the built-in set-theoretical function *sc-name* is used to identify the pitch-class set name of the current harmony.

[1] Here we consider the term harmony to be a result of one, two or more notes being sounded simultaneously. Usually the smallest possible harmony contains at least two notes. For simplicity, monophonic situations are regarded as special cases.

As can be seen in Figure 13 there are three pitch-class set names below the first note in the second measure ($E^\flat$). This is due to the fact that there are three different harmonies that include the note in question. The set-class names are, from the bottom up: (1) 1-1, which is the result of the $E^\flat$ sounding alone, (2) 2-5, which is the result of the $E^\flat$ and two-line $B^\flat$ in the right hand staff sounding together, and (3) 2-6, resulting from the low $E^\flat$ and the one-line A in the right hand.

```
(* ?1 :harmony                        ;(1)
   (?if
    (when (complete-chord? ?csv)
      (add-expression                 ;(2)
       'group (give-bass-item ?csv)   ;(3)
       :info (sc-name (m ?1))))))     ;(4)
```

Figure 13: Vertical pitch-class set theoretical analysis information displayed in the score with the help of ENP Script and groups (Arnold Schoenberg: Suite für Klavier, op. 25).

The second example deals with horizontal aspects of music. The script in this case is used to calculate and display the interval chain of a melodic line. The script is given below and the resulting score can bee seen in Figure 14. In this case we are interested in the relation between two consecutive notes (see (1) in the script). An *analysis-text-group* expression is used in this case (2). It is positioned above the associated objects (see Figure 14). The expression is also displaced horizontally (3) so that it is positioned between the notes. The printed information is calculated by subtracting two consecutive pitches (4).

```
(* ?1 ?2                                          ;(1)
   (?if
    (add-expression 'analysis-text-group ?1      ;(2)
                    :user-position-X-correction  ;(3)
                    (- (/ (width ?1) 2.0) 1.0)
                    :info (- (m ?2) (m ?1)))))   ;(4)
```

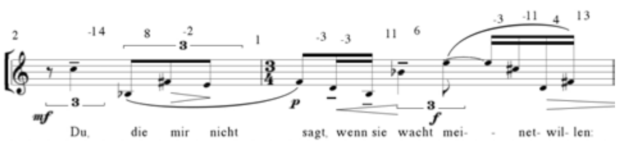

Figure 14: The interval chain of the melodic line displayed above the staff (Anton Webern: Zwei Lieder, op. 8).

Finally, we give an example dealing with n-grams. N-grams have been widely used in text retrieval, where the sequence of symbols is divided into overlapping constant-length sub-sequences [9]. The use of n-grams in MIR has also been considered by several authors (see for example [10]).

Let us examine the script in more detail. It is divided into two parts. The first part (A) calculates di-grams and the second part (B) calculates tri-grams. The Lisp function, add-expression, is again used to insert the analysis information into the notes (2). This time the information is attached to multiple notes (see the di-gram information and the corresponding brackets in Figure 15). Also a more subtle bracket style is utilized (3). In (4) and (5) the interval information is calculated and formatted (the specifics of the Lisp's format function are not in the scope of this paper. See [11] for detailed information). The second part of the script (B) is otherwise identical, when compared to the first one, except that we now consider four consecutive notes (6) forming three intervals. The printed information is again calculated in (7-9).

```
A.

(* ?1 ?2 ?3                                   ;(1)
   (?if
    (add-expression 'group ?1 ?2 ?3           ;(2)
         :kind :bracket-at-end                ;(3)
         :info (format () "~3,@d |~3,@d"      ;(4)
                       (- (m ?2) (m ?1))      ;(5)
                       (- (m ?3) (m ?2))))))

B.

(* ?1 ?2 ?3 ?4                                ;(6)
   (?if
    (add-expression 'group ?1 ?2 ?3 ?4
         :kind :bracket-at-end
         :info (format () "~3,@d |~3,@d |~3,@d"
                       (- (m ?2) (m ?1))      ;(7)
                       (- (m ?3) (m ?2))      ;(8)
                       (- (m ?4) (m ?3))))))  ;(9)
```

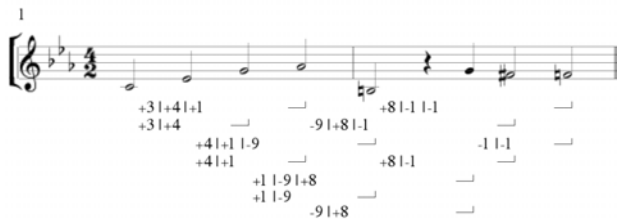

Figure 15: Di- and tri-grams displayed as a part of a musical texture (J.S. Bach: Musikalisches Opfer).

5 CONCLUSIONS

A comprehensive set of different annotation devices in ENP were discussed. Also a scheme to automatically analyze a score and to visualize analytical data was introduced.

Currently the ENP-expressions can be attached only to notes or chords. The present scheme should be extended to apply all the objects in the score. This way it would be possible to attach annotations to parts, voices, measures, bar lines, clefs, etc. It is possible, however, to insert comments to measures, for example, but these annotations have different status than the regular ENP-expressions. By addressing this problem it would also make the ENP object representation scheme more unified and coherent.

Displaying MIR results in a score is suitable only for certain types of data. To bring ENP closer to practical MIR applications it would be beneficial to integrate a statistical package, such as R [12], into the environment (R is a language and environment for statistical computing and graphics). This would allow to represent, for example, pitch histograms and other similar information retrieved from an ENP score.

ACKNOWLEDGEMENTS

The work of Mikael Laurson has been supported by the Academy of Finland (SA 105557).

REFERENCES

[1] Mika Kuuskankare and Mikael Laurson. ENP2.0 A Music Notation Program Implemented in Common Lisp and OpenGL. In *Proceedings of International Computer Music Conference*, pages 463–466, Gothenburg, Sweden, September 2002.

[2] Mika Kuuskankare and Mikael Laurson. Expressive Notation Package - an Overview. In *International Symposium on Music Information Retrieval*, 2004.

[3] Tillman Weyde. Case Study: Representation of Musical Structure for Music Software.

[4] Mika Kuuskankare and Mikael Laurson. Intelligent Scripting in ENP using PWConstraints. In *Proceedings of International Computer Music Conference*, pages 684–687, 2004.

[5] Mika Kuuskankare and Mikael Laurson. Recent Developments in ENP-score-notation. In *Sound and Music Computing '04*, October 2004.

[6] Allen Forte. The Structure of Atonal Music. *Journal of Music Theory*, 1973.

[7] Donncha Ó Maidín. Common Practice Notation View: a Score Representation for the Construction of Algorithms. *Proceedings of International Computer Music Conference*, pages 248–251, 1999.

[8] Donncha Ó Maidín and Margaret Cahill. Score Processing for MIR. *International Symposium on Music Information Retrieval*, pages 59–64, 2001.

[9] Shyamala Doraisamy. *Polyphonic Music Retrieval: The N-gram Approach*. PhD thesis, University of London, 2004.

[10] J. S. Downie. *Evaluating a Simple Approach to Music Information Retrieval: Conceiving Melodic N-grams as Text*. PhD thesis, University of Western Ontario, 1999.

[11] Guy L. Steele. *Common LISP : The Language*. Digital Press, 2nd edition, 1990.

[12] Francisco Cribari-Neto and Spyros G. Zarkos. R: Yet another econometric programming environment. *Journal of Applied Econometrics*, 14:319–329, 1999.