Goiaba: a software to process musical contours

Marcos S. Sampaio[1], Pedro Kröger[1]

[1]Genos—Grupo de pesquisa em Computação Musical
Escola de Música da Universidade Federal da Bahia

`mdsmus@gmail.com, pedro.kroger@gmail.com`

Abstract. *Contour is the shape or format of objects. Contours can be associated to musical parameters such as pitch and time, representing one in function of another. Contours help to give coherence to musical piece and can be used to analyze and to compose music. Contour theories provide many operations that demand precise mathematical calculation. In this article we present the current state of Goiaba, a software that assists musicians in contour related tasks such as the calculation and plotting of operations, and a case study of a composition where Goiaba was used to generate the contour-related material.*

1. Introduction

Contour is the shape or format of an object. In music one can speak of a pitch contour, density contour, and so on. Contours can easily be recognized from graphic representation by professionals and laymen alike [Marvin, 1988]. For instance, Beethoven's Fifth Symphony's main motive and the corresponding pitch contour are represented respectively in figures 1a and 1b.
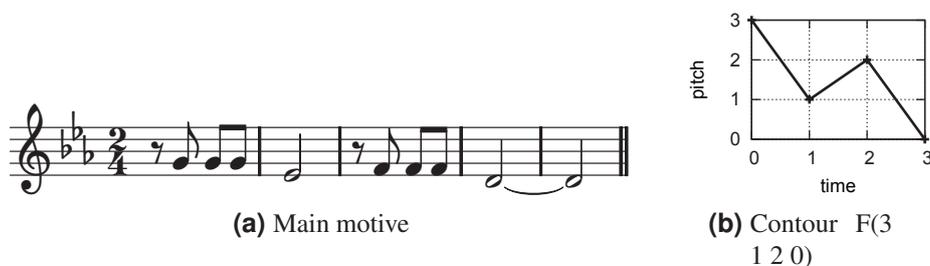

(a) Main motive **(b)** Contour F(3 1 2 0)

Figure 1: Fifth Symphony main motive contour

Technically, a contour is an ordered set of numbered elements [Morris, 1993]. Absolute values of contour elements are ignored, and only the high-low relationship between them is regarded. For instance, the music in figures 1a and 2 have the same pitch contour, graphically represented in figure 1b, and symbolically by F(3 1 2 0)[1]. Yet, both passages sound completely different. In our opinion that is a feature of using contour theory in composition, to have an underlining process providing coherence and musical variety at the same time.

The study of contour is important because contours can help to give coherence to a musical piece, like motives and pitch sets. They are structural devices that can be combined through operations like inversion and retrogradation, and can be approached by analytical or compositional points of view.

---

[1]The uppercase letter F names the contour.

Figure 2: A melody with the F(3 1 2 0) contour

Contour theories [Friedmann, 1985, Morris, 1987, Marvin, 1988, Beard, 2003] have been developed to organize the current knowledge about contour in a systematic way. These theories were developed primarily as analytic techniques for non-tonal compositions [Beard, 2003], and provide arrays, matrices and many operations to help the comparison of contours, like inversion, translation, comparison matrix, and contour interval array. It's out of the scope of this paper to provide a comprehensive review about contour theory. The reader can find a good literature review in [Beard, 2003].

A computer program to process contours can assist composers and analysts in tasks like the calculation of operations—avoiding human error and wasting time—, automated graphical plotting, and conversion from music score to contours and vice-versa. For these reasons we are developing *Goiaba*, a software to contour processing (described in section 2).

In this article we will present *Goiaba*, its data representation, examples of the software output, and a case study of a composition, in which *Goiaba* was used to compose most of a woodwind quintet.

2. The contour processing program Goiaba

Goiaba is a program written in Common Lisp developed by the authors of this paper to process and plot contours. It has many contour-related operations, like inversion, retrogradation, rotation, contour reduction [Adams, 1976], contour class, contour adjacency series, contour adjacency series vector, contour interval, contour interval array, contour class vector I and II [Friedmann, 1985], and comparison matrix [Morris, 1993]. Currently, *Goiaba* accepts and shows contours in a numeric format, but it can also plot contours in a pdf file.

Goiaba has two representations for contours; simple contours represents only the values of the contour elements, and contours with durations are basically ordered collections of cartesian points. For instance, the contour in figure 1b would be represented as a simple contour as `#s(3 1 2 0)` and as a contour with duration as `#d(#p(0 3) #p(1 1) #p(2 2) #p(3 0))`. The forms `#s(...)` and `#d(...)` indicate a simple contour and a contour with duration, respectively. The notation `#p(x y)` indicates a point with two values. So, from the example we can see that *Goiaba* really represents a contour with duration as a collection of points. The symbols #s, #d, and #p are user-defined lisp reader macros that expand into code to instantiate objects of types simple-contour, contour-with-duration, and point, respectively. For instance, `#p(0 1)` is expanded to `(make-instance 'point :x 0 :y 1)`, which is the usual way of instantiating objects in common lisp. Finally, *Goiaba* has a few constructor functions besides the reader macros to help the creation of contour objects. The functions, `make-point-list`, `make-simple-contourlist`, and `make-contour-with-duration-list`, map a list to the correspondent object.

Goiaba uses the Cl-pdf library to plot contours, allowing easy visualization of contour operations. For instance, the code in figure 3a generates a graph with the original contour Z, `#s(0 5 3 4 1 3)`, and its retrogradation, inversion, and rotation. The result can be seen in figure 3b.
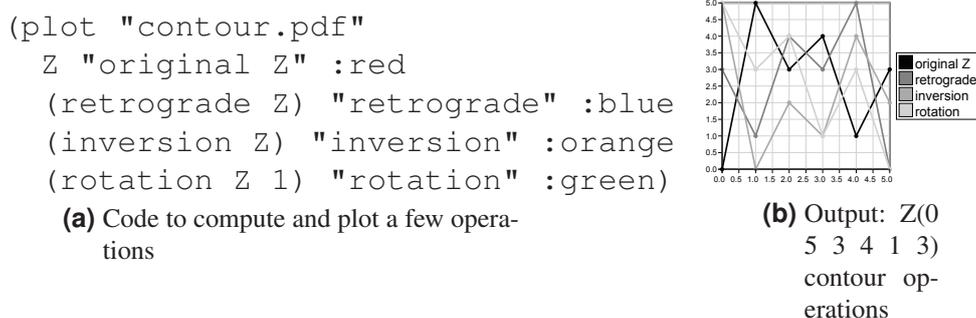
```
(plot "contour.pdf"
  Z "original Z" :red
  (retrograde Z) "retrograde" :blue
  (inversion Z) "inversion" :orange
  (rotation Z 1) "rotation" :green)
```

(a) Code to compute and plot a few operations

(b) Output: Z(0 5 3 4 1 3) contour operations

Figure 3: *Goiaba* **input code and graphic output**

Goiaba takes advantages of Common Lisp's multimethods capabilities. In Common Lisp a method is actually a generic function that specializes on the type of its arguments. For example, we have two definitions for a generic function called `rotate`, one specializes on the contour-with-duration object while the other on the simple-contour object. The advantage of this approach is that we can add more types of contour if necessary and write the appropriate generic functions that will specialize on those types, without disrupting the existing code.

3. The application of Contour in composition

Systematic studies about the usage of contour operations and combinations in musical composition are scarce, despite the possible coherence that contours can provide. For this reason we are researching the usage of contour in composition and its advantages. The first author of this paper, during his master's [Sampaio, 2008], composed a woodwind quintet, based on contour theories operations. We will use this piece as a case study for the use of contour theory in composition, and how *Goiaba* was useful in the compositional process.

This piece was composed entirely using *Goiaba* to simplify the calculation of contour operations and plotting. The piece is based on the contour P(5 3 4 1 2 0) and on combinations of contour operations associated to parameters such as pitch, tempo, density and texture. In figure 4a we can see the original contour P and its subsets and operations; retrogradation, inversion, rotation, and interpolation.

Goiaba was essential to compose a *fugato* session in the quintet because each part of the subject and countersubject were based on different combinations of operations of rotation and retrogradation. The subject is formed by the concatenation of P and its rotation by a factor of 3, as seen in figure 4b. Figure 5a shows *Goiaba*'s graphical output for both contours. The countersubject is formed by a sequence of three rotations (by the factor of 5, 4, and 3) of the retrograde of P (fig. 4c). We can see the graphical output of these contours in figure 5b.

4. Conclusion and future work

Contours help to give coherence to a musical piece, are easily recognized graphically by musicians and laymen, and can be used to analyze and compose music. Contour theories provide many operations that demand precise mathematical calculations and graphical representation, for this reason we are developing *Goiaba*, a contour processing software that helps the calculation and plotting of contour operations. *Goiaba* has been proven to be useful in composing music that uses contour theory extensively. Currently *Goiaba*
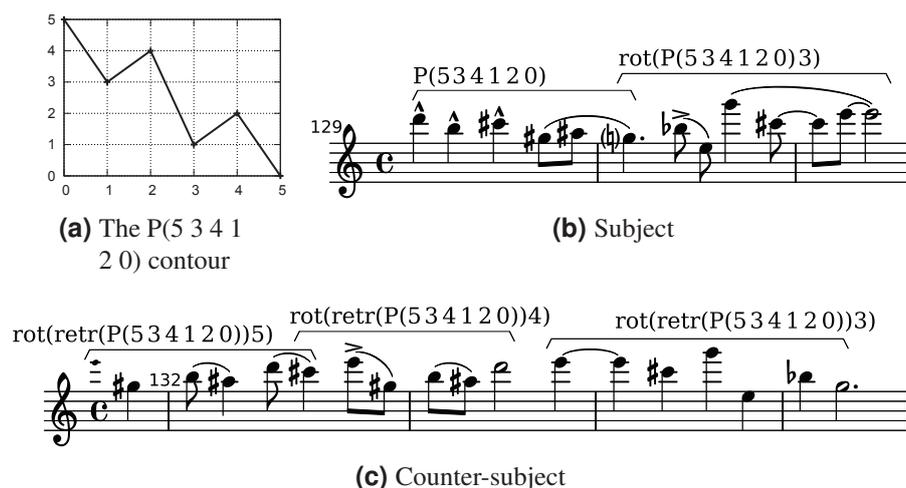
(a) The P(5 3 4 1 2 0) contour

(b) Subject

(c) Counter-subject

Figure 4: Structural elements of *fugato*

(a) Subject
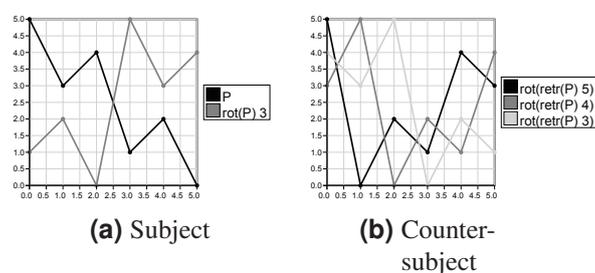
(b) Counter-subject

Figure 5: Software output for *fugato* contour operations

accepts only input in a symbolic format, but we have plans to add support for Lilypond, MIDI, ABC, and MusicXML formats as well. The next step in our research is to improve *Goiaba* user interaction, releasing a more friendly interface, possibly with a GUI.

References

Adams, C. R. (1976). Melodic contour typology. *Ethnomusicology*, 20(2):179–215.

Beard, R. D. (2003). *Contour Modeling by Multiple Linear Regression of the Nineteen Piano Sonatas by Mozart*. Phd diss., The Florida State University. School of Music.

Friedmann, M. L. (1985). A methodology for the discussion of contour: its application to schoenberg's music. *Journal of Music Theory*, 29(2):223–48.

Marvin, E. W. (1988). *A generalized theory of musical contour: its application to melodic and rhythmic analysis of non-tonal music and its perceptual and pedagogical implications*. PhD thesis, University of Rochester.

Morris, R. D. (1987). *Composition with Pitch-classes: A Theory of Compositional Design*. Yale University Press.

Morris, R. D. (1993). New directions in the theory and analysis of musical contour. *Music Theory Spectrum*, xv:205–28.

Sampaio, M. (2008). Em torno da romã: aplicações de operações de contornos na composição. Master's thesis, Universidade Federal da Bahia, Salvador, BA.