

# A Model for the Perception of Tonal Melodies

Dirk-Jan Povel

Nijmegen Institute for Cognition and Information (NICI)  
P.O. Box 9104  
6500 HE Nijmegen, The Netherlands  
[Povel@NICI.kun.nl](mailto:Povel@NICI.kun.nl)

**Abstract.** This paper presents a concise description of OPTM, a computational model for the *On-line Processing of Tonal Melodies*. It describes the processes a listener executes when transforming a tone sequence into a tonal melody. The model is based on the assumption that a tone sequence is represented in terms of a chord progression underlying the sequence. Chord tones are directly coded in the harmonic frame, while non-chord tones can sometimes be linked to a subsequent chord tone. The model implements three primary mechanisms: *key finding*, *chord recognition*, and *anchoring*. The operation of the model and its associated output are displayed while the tone sequence is entered incrementally. The output consists of the evolving harmonic framework, the assimilation of non-chord tones, the arising expectations, the tones that do not fit, and an overall indication of the ‘goodness’ of the melodic percept.

## 1 Introduction

A tonal melody is a sequence of tones that is perceived as well-formed or acceptable within the repertoire of tonal music. Thus, a melody is the result of a perceptual process. It is this process that we have described formally in the OPTM model.

### 1.1 Background

The OPTM model is founded upon theoretical and experimental research in the perception of tonal music. This research has provided the insight that a listener when listening to tonal music activates a mental framework consisting of a metrical schema and a harmonic schema that serve as the context in which the input is coded [e.g. 1, 2, 3, 4, 5, 6]. This conception implies that the perceptual process has a bottom-up aspect in which the incoming tones are registered and pre-processed, and a top-down aspect in which the listener activates a pertinent metrical and harmonic framework that guides further coding of the input.

The global form and structure of the model was inspired by [7, 8, 9, 10] among others, while its specific architecture is based on experimental work in [11, 12]. The latter research has shown that a tone sequence is perceived as a tonal melody *only* if the listener succeeds in discovering the underlying harmony and if occurring non-

chord tones can be linked to closely following chord tones (see section 2.1 for an example). That research has also shown that the judgment of melodic quality is hardly influenced by characteristics of the pitch-height configuration such as number of contour changes, pitch reversal tendency, mean interval dissonance, mean interval size, or the variation of interval sizes.

These combined findings suggest that the activation of a harmonic framework is an indispensable step in the processing of tone sequences, and, more importantly, that a melody is represented in terms of its harmonic implications, i. e., as a pattern of stabilities, attractions between tones and chords, and expectancies for future events, rather than in terms of properties of the pitch-height configuration. This is a significant result because it indicates that listeners represent a melody in terms of its ‘musical meaning’ rather than in terms of its surface detail, just as a listener will usually only be able to paraphrase the meaning of a spoken sentence but not the actual wording. Independent evidence supporting this conception has been reported in [13].

As the main goal is to describe the perceptual processes that take place while the input sequence is made available incrementally, the OPTM model is designed as a series of processes unfolding in time<sup>1</sup>. For this purpose the model is provided with a flexible interface that displays the incremental build up of a mental representation while the sequence unfolds.

The OPTM model is still under construction. At present there are limitations regarding the size and type of sequences that can be processed, and some of the algorithms will need further elaboration and fine-tuning. The overall structure of the program, however, is fairly well established.

## 1.2 Domain

The domain of the model consists of monophonic (single) tone sequences comprising tones of equal duration and segmented into groups of three tones. The tones in a sequence are specified by the following parameters: *Onset* (moment in time at which the tone is turned on), *Note* (pitch height as a MIDI number); *Duration* (duration of the note in ms); *Velon* (the ‘loudness’ of the note on a scale of 1 – 127); and *Beat* (a Boolean, if *Beat* = true the note is the first of a group or segment). Due to graphical restrictions of the interface the maximum number of tones in a sequence is presently limited to 13.

## 2 Global description of the model

The model accepts as input a sequence of tones upon which a number of processes are executed that simulate the formation of a musical (tonal) representation by a human listener. These processes attempt to discover the *key* of the piece, to establish a *harmonic framework*, and to *link* tones to this framework.

---

<sup>1</sup> In this respect the model differs from the Melisma model of Temperley [9] that only arrives at an interpretation after the entire sequence has been analyzed.

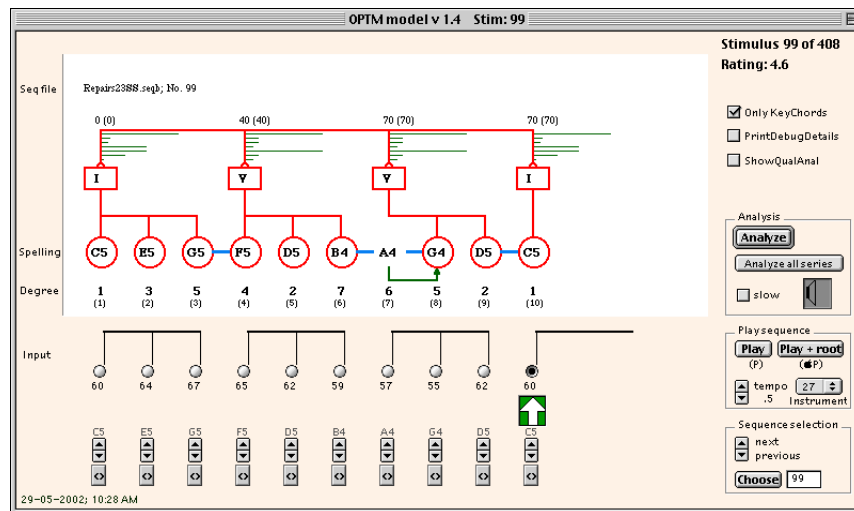
At each moment these processes render a hierarchical structural analysis that describes which notes are integrated in the analysis and which expectations arise about future tones and chords. For some sequences the model will find an appropriate harmonic progression that allows the accommodation of all tones, whereas for other sequences the model fails to find a suitable harmonic progression or may not be able to account for all notes. By comparing the output of the model with perceptual judgments of human listeners the veridicality of the model will be examined.

OPTM version 1.4 consists of three primary processes: *KeyFinder*, *ChordFinder*, and *Anchoring*. These processes are described in section 3. A description of the additional processes as well as other details of the system can be found at <http://www.socsci.kun.nl/~povel>.

The model is programmed in REALbasic, an object oriented language for the MacIntosh computer (from which applications for the Windows platform can be built). Both the elements (tones, chords, keys, etc.) and the processes are implemented as objects.

## 2.1 An example

Figure 1 presents an example of how the results of the model are displayed in the main interface of the system. The interface has been designed with the aim to efficiently enter and select input sequences, and to optimally visualize the operation of the model: the incremental build up of the underlying structure and the coding of the input sequence.



**Fig. 1.** Illustration of the processing of a sample sequence  $\{C_5 E_5 G_5 F_5 D_5 B_4 A_4 G_4 D_5 C_5\}$  on the main interface of OPTM version 1.4. See text for explanation

The interface shows the input sequence in a pseudo-score notation. Underneath each input-note three buttons are placed that serve to alter the sequence and to play a

fragment; on the lower right there is a column of buttons for setting 1) analysis parameters, 2) play parameters, and 3) stimulus selection. The actual output of the model, resulting from the above mentioned analyzers *KeyFinder*, *ChordRecognizer* and *Anchoring*, is depicted in the large area above the input (the stepwise construction of the output can obviously not be shown here). It shows the chords that the model yields for the four segments of the sequence (I, V, V, I) and the linking of the chord tones to those chords. Segment 3 (A4, G4, D5) contains a non-chord tone (A4) that is not linked to the chord, but to the subsequent chord tone G4.

The seven thin horizontal lines drawn just above each chord rectangle indicate the expectations for the next chord (represented in terms of transition probabilities from the current chord to each of the chords on the 7 degrees) which are part of the top-down input to the *ChordRecognizer* (see sections 3.1.1 and 3.1.2)

### 3 Description of the Main Analyzers and Knowledge Bases

#### 3.1 KeyFinder

The function of the *KeyFinder* is to establish the key which supplies the overall musical context and its associated knowledge bases.

The *KeyFinder* takes as input one or more notes from the first segment, expressed as pitch classes (PC's). A key is represented as consisting of 12 notes, seven diatonic notes and five chromatic notes, each having a certain degree of stability (a measure of the relative prominence of the note in the key). These stabilities are based on the tonal hierarchy profiles collected in experiments reported in [14]. The stabilities of the 12 tones in the major key are: 6.35, 2.23, 3.48, 2.33, 4.38, 4.09, 2.52, 5.19, 2.39, 3.66, 2.29, 2.88; and of those in the (harmonic) minor key: 6.33, 2.68, 3.52, 5.38, 2.60, 3.53, 2.54, 4.75, 3.98, 2.69, 3.34, 3.17. Next, the activation of each of the 24 keys (12 major and 12 minor keys) is determined as the sum of the stabilities of the input notes (represented as PC's) in those keys. In calculating the key the occurrence of a PC is only counted once. The key with the highest activation is selected as the key of the piece.

The algorithm is a simplified version of the algorithm developed by Krumhansl & Schmuckler [3, p. 77–110]. Our version does not take into account the duration of the notes, nor the order of the notes in the segment, both identified factors in key finding. However, the key finder in the present model is not meant to be a robust algorithm as it is only used to get the analysis started. For simple tonal melodies in which the key is clearly established by the first few tones it works well enough. It should be noted that since the key is only based on the notes in the first segment, the model cannot handle modulation.

### 3.1.1 The KeySpace

As soon as a key has been induced, a *KeySpace* is constructed that describes the diatonic tones (as PC's) and the chords that constitute the key. After, for instance, the key of C major has been recognized, a *KeySpace* will be configured as shown in Table 1.

**Table 1.** Key space for the Key of C major

Degree	I	II	III	IV	V	VI	VII
Diatonic tones	0	2	4	5	7	9	11
Major triad	+	+	+	+	+	-	-
Minor triad	-	+	+	-	-	+	-
Dominant 7th	-	+	-	-	+	-	-

The *KeySpace* represents part of the knowledge that the listener brings to bear in perceiving the input. It is top-down information that helps to determine the degree of a tone, and that supports the process of chord recognition. In a later phase, other chords may be added to *KeySpace*.

### 3.1.2 The Piston Table

The *PistonTable* is another aspect of the knowledge that the listener uses in processing the input. The *PistonTable* represents the transition probabilities of chords in tonal music as specified in the 'Table of usual root progressions' by Piston [15]. Given the root of a chord, it shows the likelihood that it will be followed by one or more other chord(s). See Table 2.

**Table 2.** The table of usual root progressions from Piston & DeVoto [15]

Chord	Is followed by	Sometimes by	Less often by
I	IV or V	VI	II or III
II	V	IV or VI	I or III
III	VI	IV	I, II or V
IV	V	I or II	III or VI
V	I	VI or IV	III or II
VI	II or V	III or IV	I
VII	III	I	-

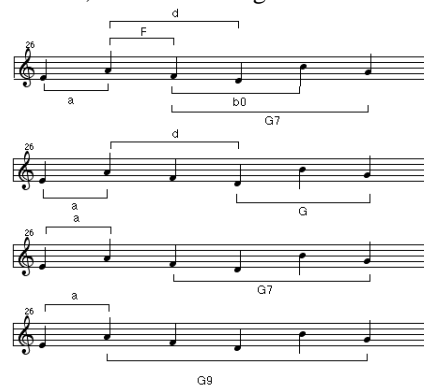
A Table of Root-transition-probabilities is derived from the *PistonTable*. Like *KeySpace*, the *PistonTable* plays an important role in the process of chord recognition. It is a top-down constraint that affects the activation of chord candidates in the *ChordRecognizer*. It specifies the chance that some chord is followed by some other chord. As such, the expectations that are derived from this table are based on a first-order approximation in which only the preceding chord is taken into account. This approach is clearly rather limited as a typical listener probably creates expectations based on longer chord progressions (e.g. cadences). Therefore, in future developments, we intend to base the harmonic top-down constraints on more sophisticated theoretical notions like those proposed in [8, 16, 17].

## 3.2 ChordRecognizer

### 3.2.1 Introduction

*ChordRecognizer* is the most important analyzer in the model; it is also by far the most complex one. Its purpose is to establish the harmonic interpretation of a segment, thereby providing the frame of reference for that segment. The main problems encountered in chord identification are:

1. to determine the window in which one is going to search for a chord. Depending on the partitioning of a sequence different harmonic analyses will tend to be generated, as shown in Figure 2.



**Fig. 2.** Different harmonic analyses for a melodic fragment depending on the applied partitioning

2. the fact that in a key a variety of chords (each consisting of 3 or 4 tones, if we reckon the most common chords) are formed using only a very limited (7 or 8) number of PC's. This results into considerable overlap between the constituents of different chords leading to serious identification problems. See Figure 2.
3. the overlap problem is often aggravated by the phenomenon of underspecification: not all elements of a chord are played but a listener still identifies a chord.
4. not all tones within the identification window belong to one chord; i.e., non-chord tones occur that are usually anchored to chord tones (passing tones, neighbor tones, appoggiatura's).

To solve the chord identification problem as a bottom-up problem without using the segmentation of the sequence induced by meter is virtually doomed to fail (see however [18]). Only if the chord recognition process is guided by the segmentation of the sequence (effected by the meter), and if the expectations for specific chord progressions suggested by the key and the chords already given are taken into

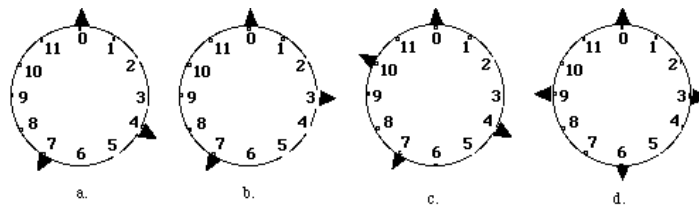
account, it will appear feasible to arrive at a veridical harmonic interpretation. Moreover we believe that the human listener also works along these lines.

For these reasons, the present chord identification algorithm uses a) the successive segments as windows within which chords are being identified, and b) the expectations for the current harmony derived from the Piston Table associated with the current key. The successive steps in *ChordFinder* are discussed below.

### 3.2.2 Steps in the chord recognition process

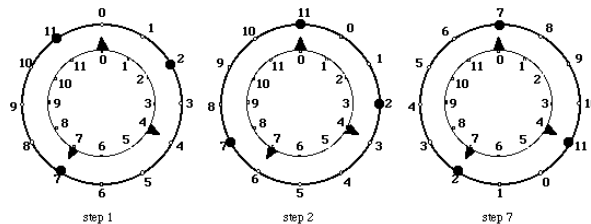
#### 3.2.2.1 Step 1: Bottom-up processing: Activation of chord templates

First the notes in the segment are converted into Pitch classes and mapped on four types of chord templates: major triad, minor triad, dominant seventh chord, and diminished seventh chord. These templates can be represented on a chromatic circle comprising all 12 pitch classes as shown in Figure 3.



**Fig. 3.** Templates for a Major triad (a), a Minor triad (b), a Dominant seventh chord (c), and a Diminished seventh chord (d).

The bottom-up activation of each of the chord types is calculated by establishing to what degree the input activates each of the different instances (e.g. C major, C# major, D major etc.) of the four chord types. In terms of the above circles, this can be represented as rotating the input, also displayed on a chromatic circle, around a chord template and determining at each step the sum of the weights of the elements in the template that correspond to the notes in the input. See Figure 4.



**Fig. 4.** Three stages in the process of establishing the activation of a major triad by rotating the input (PC's 2 7 11) around the major triad template: 1) activation of C major; 2) activation of C# major; 3) activation of G major. G major (PC 7) yields the

highest activation, because all tones in the input correspond to elements in the template.

The code that calculates the activation is:

```
for j = 0 to 11          // rotate in 12 steps
  for i = 1 to Length   // number of elements in the input
    Activation(j)
      =Activation(j)+Template(((PCsInFragment(i)-j)+12)Mod12)
  next
next
```

First the net activation is computed. This activation is caused by the unique PC's in a segment, i.e. irrespective of the number of times a PC occurs in a segment. Total bottom-up activation is defined as a function of the net activation, the number of repeated chord tones, and the number of non-chord tones according to formula (1).

$$\text{BottomUpActivation}(i) = \text{NetActivation}(i) + \text{NrofRepeatedChordTones}(i) * .5 \quad (1) \\ + \text{NrofNonChordTones}(i) * .1$$

It should be noted that a dominant-seventh chord is only activated if the seventh is present, and that a diminished seventh chord is only activated if both the root and the seventh are present. In a later stage, other chords (e.g. minor seventh, half diminished seven, etc.) may be added, making the selection process inevitably more difficult.

#### **3.2.2.2 Step 2: incorporating top-down information**

Next, top-down information is added to the bottom-up information of all chord instances of all types. Specifically, the total activation is calculated by multiplying the bottom-up activation with the expectation for the chord derived from the Piston Table. In this way the Total Activation for all chords is obtained.

#### **3.2.2.3 Step 3: determining the maximally activated chord**

In this step the chord(s) with the maximal activation, are searched for by finding the chord(s) with the highest activation. These chord candidates are put in the list MaxChords. If this list contains more than one chord, which occurs rather frequently due to the small number of tones in a segment, further steps are taken to arrive at a unique identification. If it only contains one chord, this is the chord underlying the current segment.

#### **3.2.2.4 Step 4: towards the identification of a single chord**

If MaxChords contains more than one chord, it is determined whether there are non-chord tones within that chord interpretation and whether these can be anchored (see below) to a succeeding tone in the segment. If there is a non-chord tone that can *not* be anchored, that chord will be removed from MaxChords. In virtually all cases MaxChords will then contain only one chord candidate: the *Recognized Chord* representing the harmonic interpretation of the current segment.

### 3.3 Anchoring

The perceptual mechanism of *Anchoring* was first described in [7] that also specified the circumstances needed for a non-chord tone to be anchored to a subsequent chord tone. Two variants of the analyzer *Anchoring* have been implemented. The first, *Anchoring1*, takes a non-chord tone as input and determines if one of the following tones in the segment is a chord tone at an interval of less than 2 semitones. If this is the case the non-chord tone is anchored to that chord tone. Figure 5 shows two examples of this type of anchoring: in the first example the F# is anchored to its neighbor G, while in the second example the A is anchored to the G (named ‘delayed’ anchoring [7]).

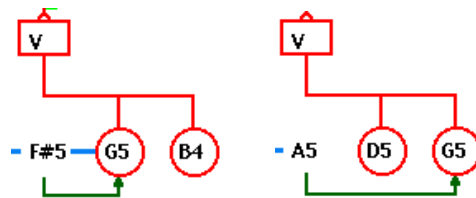


Fig. 5. Two examples of Anchoring1. See text for explanation.

The other variant, *Anchoring2*, determines whether an unanchored nonchord tone in the previous segment can be linked to a current chord tone. If this is the case the two tones will be linked. Figure 7 shows two examples.

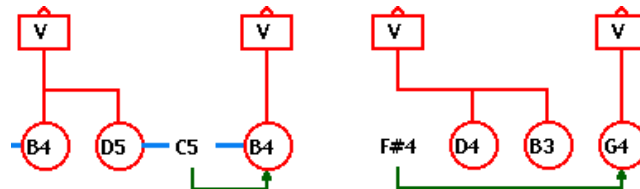


Fig. 6. Two examples of Anchoring2. See text for explanation.

A note can only be anchored once, so that the A in the segment  $\{A_4 B_4 G_4\}$ , (underlying harmony G) will anchor to the B and not to the subsequent G.

## 4 Output: the Percept

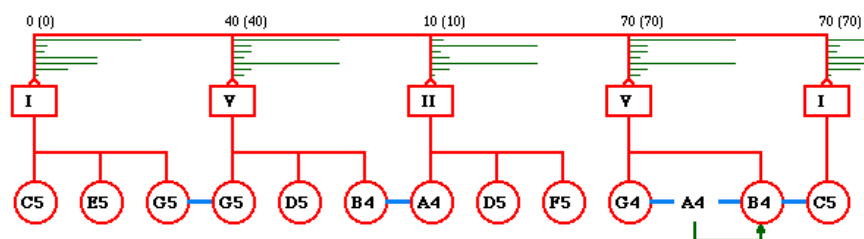
The output generated by the model is expressed in the class *Percept*. This class is viewed as representing aspects of the mental representation that has been formed during the processing of the input. These include perceptually relevant characteristics

associated with the representation that could show up in listener reports, and that will be used to test the model. The specific contents of Percept are:

1. The *structural description* displayed in the main window, showing
  - a. the harmonic progression assigned to the melody
  - b. The chord tones and non-chord tones
  - c. Which non-chord tones are/are not anchored
  - d. The expectations arising at different points in the sequence
2. A number of properties derived from the structural description, namely:
  - a. The progression strength (the sum of the expectations of the successive chord transitions, indicating how 'regular' or expected the harmonic progression is)
  - b. The number of anchored non-chord tones
  - c. The number of non-anchored non-chord tones
  - d. Progression is hampered (a Boolean which is 1 if ChordRecognizer has failed at some point)
  - e. Sequence ends on beat (a Boolean)
  - f. Sequence ends on I or VI (a Boolean)
  - g. Finally, a quantitative overall score *Goodness* is computed as a weighted sum of the features a - f above.

#### 4.1 A few examples

Figure 7 shows the output of OPTM 1.4 for three selected sequences. Sequence 1 is a well-formed sequence with one anchored non-chord tone in the penultimate segment. Sequence 2 is an ill-formed sequence (at least the second half) for which the model indeed does not succeed in creating an appropriate analysis: there are two unconnected tones, the assigned chord progression is uncommon, the last tone is not an element of I or VI, and the sequence does not end on a down beat. Sequence 3 shows that the model is unable to recognize the (rare) modulation from C-major to Eb-major, as a result of which it generates a highly inappropriate parsing.



Sequence 1. Progression strength = 17; Goodness = 9.9 (max = 10)

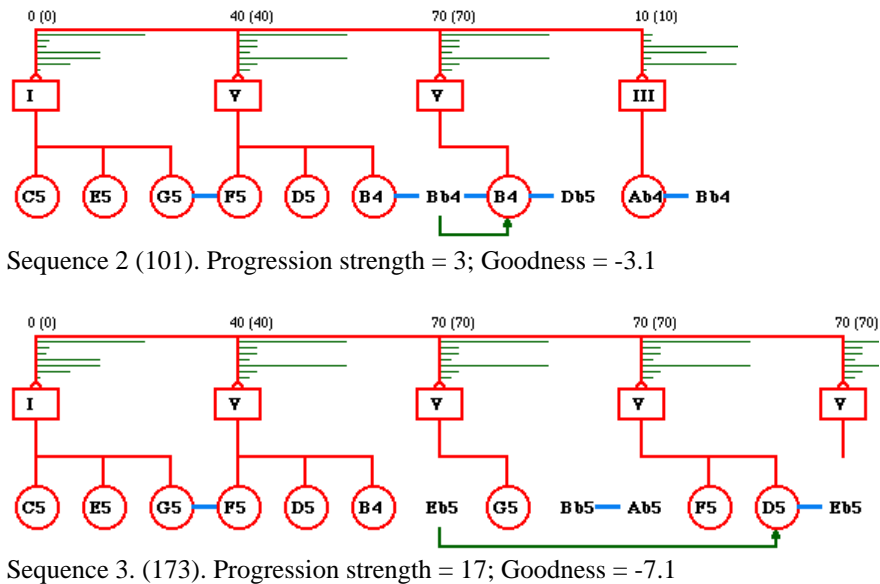


Fig. 7. The output of OPTM 1.4 for three sample sequences. See text for explanation.

## 5 Outlook

The present form of the OPTM model was obtained by adjusting the parameter values and the interaction between the modules until the performance of the model approximated that of a human listener. Although the model at present appears to work reasonably well, it still needs to be tested formally.

The various types of output that the model generates, listed in section 4, yield ample possibilities for testing: predictions from the model can be compared to the responses of human listeners. Currently we are in the process of testing the model using 1) existing tunes and melodies selected from <http://www.themefinder.org/>, and 2) a set of over 400 sequences, collected in a study in which experienced musicians completed unfinished melodic fragments, that greatly differ in the degree of well-formedness. Proceeding in this way, we expect to gain a reliable evaluation of the model as well as suggestions for future improvements.

## Acknowledgment

I thank Erik Jansen and Olivier Povel for textual suggestions that have considerably improved the readability of this paper.

## References

1. Bharucha, J. J.: Music Cognition and Perceptual Facilitation: A connectionist Framework. *Music Perception* 5 (1987) 1-30
2. Cuddy, L. L., Cohen, A., Mewhort, D. J.: Perception of structure in short melodic sequences. *Journal of Experimental Psychology: Human Perception and Performance* 7 (1981) 869-883
3. Krumhansl, C.L.: *Cognitive Foundations of Musical Pitch*. Oxford University Press, New York (1990)
4. Lerdahl, F., Jackendoff, R. *A Generative Theory of Tonal Music*. MIT Press, Cambridge MA (1983)
5. Longuet-Higgins, H. C., Steedman, M. J.: On interpreting Bach. In B. Meltzer, D. Michie (Eds.) *Machine Intelligence*. Edinburgh University Press, Edinburgh (1971)
6. Povel, D. J.: Internal Representation of Simple Temporal Patterns. *Journal of Experimental Psychology: Human Perception and Performance* 7 (1981) 3-18
7. Bharucha, J. J.: Anchoring Effects in Music: The Resolution of Dissonance. *Cognitive Psychology* 16 (1984) 485-518
8. Lerdahl, F.: Tonal Pitch Space. *Music Perception*, 5 (1988) 315-350
9. Temperley, D.: *The Cognition of Basic Musical Structures..* MIT Press, Cambridge MA (2001)
10. Zuckerkandl, V. *Sound and Symbol*. Princeton University Press, Princeton (1956)
11. Povel D. J., Jansen, E.: Perceptual Mechanisms in Music Perception. *Music Perception* 19 (2001) 169-199
12. Povel, D. J., Jansen, E.: Harmonic Factors in the Perception of Tonal Melodies. *Music Perception* (2002) Fall issue
13. Sloboda, J. A., Parker, D. H. H.: Immediate Recall of Melodies. In P. Howell, I. Cross, R. West (Eds.) *Musical Structure and Cognition*. Academic Press, London (1985)
14. Krumhansl, C. L., Kessler, E. J.: Tracing the Dynamic Changes in Perceived Tonal Organization in a Spatial Representation of Musical Keys. *Psychological Review* 89 (1982) 334-368
15. Piston, W., Devoto, M.: *Harmony*. Victor Gollancz, London (1989 originally published 1941)
16. Sutcliffe, T.: Syntactic Structures in Music. <http://www.harmony.org.uk/>
17. Steedman, M. J.: A Generative Grammar for Jazz Chord Sequences. *Music Perception* 2 (1984) 52-77
18. Pardo, B., Birmingham, W. P.: *The Chordal Analysis of Tonal Music*. The University of Michigan, Department of Electrical Engineering and Computer Science Technical Report CSE-TR-439-01 (2001)