42. Music Learning: Automatic Music Composition and Singing Voice Assessment

Lorenzo J. Tardón, Isabel Barbancho, Carles Roig, Emilio Molina, Ana M. Barbancho

Traditionally, singing skills are learned and improved by means of the supervised rehearsal of a set of selected exercises. A music teacher evaluates the user's performance and recommends new exercises according to the user's evolution.

In this chapter, the goal is to describe a virtual environment that partially resembles the traditional music learning process and the music teacher's role, allowing for a complete interactive self-learning process.

An overview of the complete chain of an interactive singing-learning system including tools and concrete techniques will be presented. In brief, first, the system should provide a set of training exercises. Then, it should assess the user's performance. Finally, the system should be able to provide the user with new exercises selected or created according to the results of the evaluation.

Following this scheme, methods for the creation of user-adapted exercises and the automatic evaluation of singing skills will be presented. A technique for the dynamical generation of musically meaningful singing exercises, adapted to the user's level, will be shown. It will be based on the proper repetition of musical structures, while assuring the correctness of harmony and rhythm. Additionally, a module for singing assessment of the user's performance, in terms of intonation and rhythm, will be shown.

In this chapter, we present several methods and techniques to implement a complete educational tool for learning to sing. Typically, singing skills are improved by rehearsing a set of appropriate exercises under the supervision of a music teacher. The role of this music teacher is to evaluate the user's performance, and to recommend new exercises according to the user's evolution. Therefore, the presented methods allow the creation of user-adapted exercises and evaluation of the singing skills of the user automatically right after the performance. The goal of this combined approach is to create a virtual environment that partially resembles the music teacher's role, leading to a faster self-learning process.

Two main submodules are presented in this chapter: (1) an automatic generator of singing exercises, and (2) a singing assessment module, which analyses the user's voice in order to rate the quality of the singing performance. In the first, the generated singing exercises are musically meaningful, based on repeated structures, and can be adapted to suit the level of the user. In the second, the module for singing assessment compares the user's performance with respect to the automatically generated singing exercise, and rates the user's performance with two criteria: intonation and rhythm. In Fig. 42.1, a block diagram of the complete system is shown.

Using these methods, the singing learning process becomes an iterative self-guided process. First, the system provides a set of exercises. Second, the user sings the suggested practices. Third, the system asseses the user's performance. And finally, it suggests new scores according to the grade obtained. Note that the scores generated are not precomposed but dynamically generated according to the current level of the user. In this way, a gradual and fully adapted learning process is assured.

This chapter is organized as follows. In Sect. 42.1 and 42.2, we present the related work on melody com-

R. Bader (Ed.), *Springer Handbook of Systematic Musicology*, https://doi.org/10.1007/978-3-662-55004-5_42

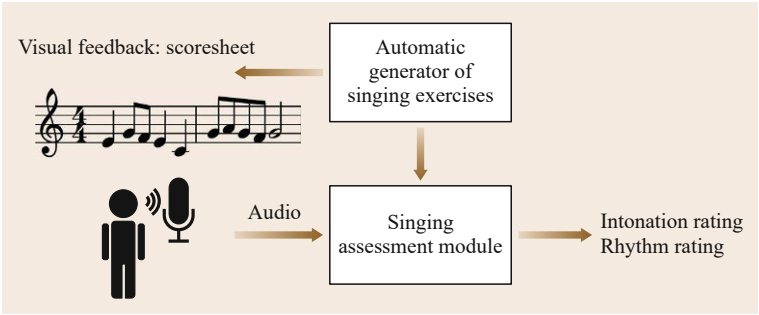

Fig. 42.1 Block diagram of the complete system

position and automatic singing assessment respectively. Then, in Sect. 42.3 an automatic generator of singing exercises is described. A scheme for automatic singing assessment is described in Sect. 42.4. Finally, Sect. 42.5 draws some conclusions about the tool for singing learning presented in this chapter.

42.1 Related Work on Melody Composition

The incorporation of different skills in the field of music information retrieval related to the computational analysis and description of musical pieces allows us to face different tasks like automatic music transcription [42.1], or the identification of relations between songs [42.2], among others. Furthermore, the computational model of the human experience in the musical field and the human brain processes in this field are of great interest for psychology and musicology [42.3].

In this context, the automatic generation of musical content is the topic considered. Often, music is defined as *organized sound* [42.4] with order and structure. Thus, systems to generate music must be trained beforehand to learn a logic composition style as stated in [42.5]. For this reason, an algorithm for learning composition rules and patterns is outlined.

A set of descriptors must be analyzed in order to model the style of the melodies. Concerning the temporal descriptors, tempo and time signature, previous works can be found. The work presented in [42.6] is focused on the onset estimation based on the spectral analysis. In [42.7], histograms to find the most repeated interonset values are employed. Recently, methods for structural analysis such as [42.8] based on a timespan tree for the structural similarity detection, which can be addressed making use of the autosimilarity matrix [42.9], were found.

In this chapter we consider an innovative approach for tempo estimation based on the interonset interval (IOI) histogram inspired by [42.6], followed by a fine adjustment stage.

Regarding music composition, apart from composition schemes based on pattern reallocation and variations, other methods are described in the bibliography. In [42.10–12] Markov models are used for the modeling and composition processes. The use of genetic algorithms such as *Biles*' GenJam system [42.13] are also present in the field of automatic music composition. Methods based on probabilistic approaches such as *Cope*'s experiments in musical intelligence (EMI) [42.14, 15] also focus on the creation of an automatic music composition framework. Inmamusys [42.16] is another composition scheme based on probabilistic structures. This method is very similar to the one considered here, since both use previously learned patterns to generate a new composition from the reallocation of them. However, we considered the presence in the composition system of a postprocessing stage, intended to make all the motives in the database learned fit in the composition. Note that Inmamusys restricts the combination of motives to subsets previously tagged as compatible.

42.2 Related Work on Voice Analysis for Assessment

Regarding the evaluation of singing voice, the literature reports a number of schemes for automatic singing assessment [42.17–27]. These schemes are able to provide feedback about the user's singing performance.

Commonly, in order to attain the desired objectives, the audio is processed according to the following steps. First, a low-level feature extraction process is performed to find a set of frame-level vectors with

meaningful information about the input. In the case of singing analysis, the most important feature is fundamental frequency F_0, although most of the approaches also use other features such as energy, aperiodicity, zero-crossing rate or certain auditory-based features. In the literature, a wide set of approaches for F_0 estimation have been proposed, some of which are based on the time domain, whereas others are based on the frequency domain (see [42.28] for a comprehensive review). One of the most-used approaches is the Yin algorithm [42.29], since it is simple, effective and easily accessible. The Yin algorithm was developed by de *Cheveigné* and *Kawahara* in 2002 [42.29], and it has been found to be effective in many monophonic music transcription systems [42.26, 30–32].

Then, the feature(s) extracted is postprocessed in order to identify voiced regions (the *voicing* process) and, in many cases, a later note-level segmentation is also performed. The estimation of voiced sounds can be performed using a wide variety of descriptors at frame-level: F_0 stability [42.33], root-mean square (RMS) [42.34], aperiodicity [42.35], or zero-crossing rate [42.36], etc.

Additionally, a note-level segmentation process of the singing voice (also called singing transcription) must be performed. To this end, some systems analyze the low-level feature(s) using heuristic rules and a set of thresholds [42.34, 37], whereas other approaches are based on probabilistic models, especially hidden Markov models [42.35, 38].

Finally, the assessment of singing skill is performed by analyzing the postprocessed low-level features and/or the note-level segmentation of the audio input. Prior works have led to various solutions for automatic singing rating. In general, all these systems focus on intonation assessment with visually attractive real-time feedback. Some of these systems use a reference melody (considered the target performance) in order to assess the user's performance, whereas other approaches are melody independent.

42.3 Music Composition for Singing Assessment

In order to be able to accurately design automatic music composition methods, it is necessary to know the parameters involved in the composition. In this section, we present both the parameters used by a novel autonomous music compositor that generates new melodies using a statistical model and the composition scheme itself. Different aspects related to the traditional way in which music is composed by humans such as harmony and structure repetitions will be considered.

The approach is focused on an educational context. The student should be able to automatically generate reinforcement melodies according to a particular musical level enlarging the number of available training exercises.

42.3.1 Learning Musical Parameters

The approach designed for the generation of contents is based on the music theory method called *ostinato* [42.39]. This method considers the composition of music on the basis of pattern repetition with harmonic variations in such a way that the repetition of the motives creates the melody structure.

Thus, rhythm patterns, pitch contours, harmonic progressions and tempo structures must be learned [42.40].

Thus, a database of musical parameters can be used to model the training level of certain musical pieces, as in [42.41]. Since the main objective is to develop a music model for the automatic creation of compositions with style replication, the discovery of this type of information and the development of specific procedures to make use of the different pieces of information to model music corresponding to different training levels are considered. This can be done on the basis of a probabilistic analysis of rhythm and pitch patterns stored in a database filled with music samples of different complexity levels. In Fig. 42.2, a diagram of a suitable analysis system is presented.

According to the characterization parameters required, the database can be divided into three levels

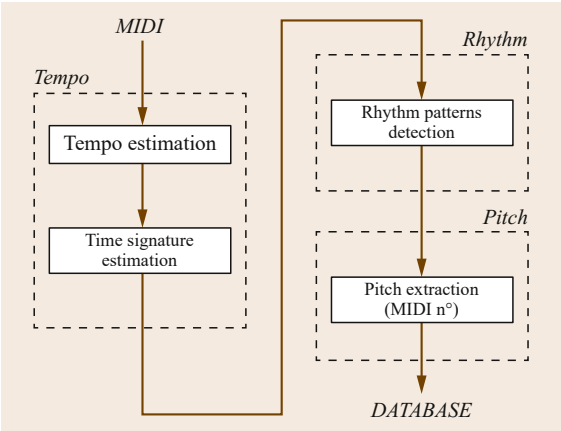

Fig. 42.2 Illustrative scheme of the music analysis system

hierarchically organized corresponding to: (L-1) time signatures, (L-2) rhythm patterns and (L-3) pitch contours (Fig. 42.3).

The measures the training samples will be split into will be the elements stored in the database. These elements will be used for the composition of novel music scores. In order to achieve this goal, the bar length has to be established for proper measure splitting. The bar length can be manually set but also a system to perform this task automatically can be devised. Thus, tempo estimation is required in first place.

The tempo can be extracted easily by analyzing Musical Instrument Digital Interface (MIDI) metadata messages [42.42], if present. But this information can also be incorrectly stored. In order to develop a robust estimation scheme, an algorithm to estimate the tempo and time signature from MIDI files can be used. Note that the availability of correct tempo information is critical in order to relate the duration of the notes obtained by means of the analysis of Note On and Note Off messages [42.42] to musical figures.

Note that according to [42.43], the target parameters in this work are the basic ingredients for the composition of music: rhythm, pitch motives (melody) and harmony (which is considered at the score composition stage).

Now, we consider the specific estimation stages.

Temporal Estimations

The tempo and the time signature have to be estimated in order to correctly perform bar separation and properly split rhythmic patterns. Note that here we consider a rhythmic pattern to be equivalent to a complete measure from the input training data.

Tempo Estimation. The algorithm considered for tempo estimation is inspired by the work presented in [42.7]. However, in our scenario, the analyzed IOIs are directly extracted from the melody. Initially, the most repeated IOI value can be considered a candidate pulse, or tactus [42.44]. This pulse is related to tapping or dancing while listening to a piece of music [42.45]. However, some considerations must be taken into account:

- Resting periods are not explicitly extracted. MIDI files contain information about the notes solely (Note On and Note Off events [42.42]). However, resting periods can be indirectly extracted and must be used to properly estimate the tempo.
- The tactus extracted can be a multiple or a divisor of the actual tactus. By setting a valid tempo range, this value can be corrected.
- The tactus estimated will not be the exact one due to the discrete nature of the histogram. The value can be finely corrected in a postprocessing stage.

The objective of the fine adjustment of the tactus is to find the value that causes the lowest displacement from the constant beat and the input file onsets. This can be achieved by defining a specific model to interpolate the histogram of the IOIs.

Then, the tactus has to be associated with a certain rhythmic element to define the duration of the quarter note and for the estimation of the tempo. Since the range of valid tempos in music is large – from *Largo* ($40 \sim 60$ quarters per minute) to *Presto* ($180 \sim 200$ quarters per minute) [42.46] – the tempo range must be manually reduced in order to establish an accurate relation between durations and musical elements. A suitable hypothesis is that the tempo of the training data is *Moderato* ($76 \sim 108$ quarters per minute). Anyway, a mapping can always be defined by calculating the duration of each rhythm figure in the range selected.

As described in [42.47], the tempo estimation algorithm often estimates doubled or halved tempos. This is a normal behavior caused by the tempo concept itself [42.48]. The tempo is actually subjective, which means that some editors may use shorter rhythm figures reducing the selected tempo, while other can do the opposite, to represent the same performance speed.

Finally, observe that the duration of the metric figures is well known after the tempo is estimated. The duration of each measure can be obtained by multiplying the duration of the pulse by the number of pulses that fit in one bar. So, the next goal is estimate the bar duration. The approach for this purpose can be based on the evaluation of different bar split scenarios for several tentative lengths. Then, some features like the number of repeated bars, the number of bars and the number of split notes can be considered.

Time Signature Estimation. The estimation of the time signature can be based on the analysis of bar repetitions, which can be done using a multiresolution analysis scheme [42.49, 50] to obtain the bar length that best suits the input melody among a set of candidates.

The rhythm self-similarity matrix (RSSM), as described in [42.51], is useful for the purpose at hand. The input melodies can be split into the k candidate bar lengths in order to build the RSSM using the tactus as a unit. Note that the analysis system will create k RSSM matrices, one for each of the candidates. Using those

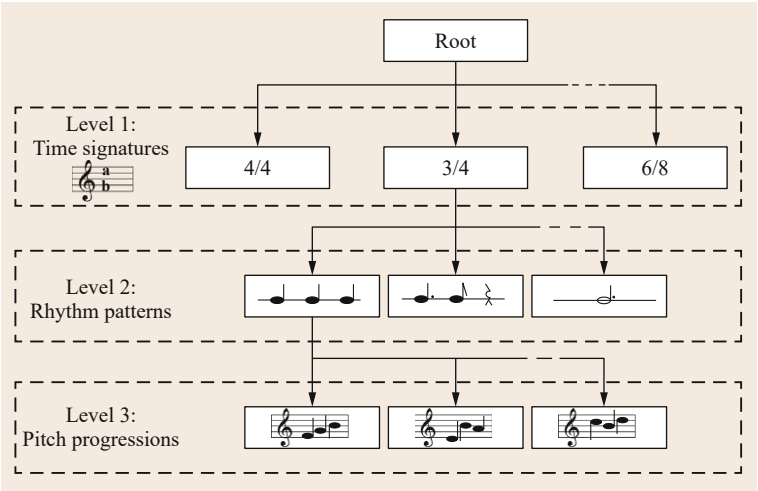

Fig. 42.3 Representation of the hierarchical database considered to store and organize music parameters for music composition based on pattern repetitions

RSSMs, the following descriptors can be extracted and considered in the estimation process:

- Number of repeated bars: the amount of different bars repeated along the split performed for a certain length candidate. The most repeated bar will, more probably, perform a proper separation.
- Number of repeated bar instances: the average number of instances per repeated bar. The larger this number, the more probable the separation will be.
- Number of ties between bars: the number of notes divided between two bars for the candidate length. The lower the number of ties between bars, the more probable the separation will be.
- Number of detected bars: the number of bar splits in the original stream. The number of bars tends to be a power of two.

Note that the number of repeated bars, bar instances and ties must be normalized by the total number of bars detected to give rise to comparable measures. In order to classify the time signature of the input melodies using these descriptors, different classification schemes can be considered, such as the J48 decision tree classifier [42.52], or another one based on sequential minimal optimization (SMO) [42.53], which are available in the Weka machine learning software suite [42.54].

Rhythm Patterns

Rhythm is probably the musical feature more closely related to the structure of a musical composition. The parameters obtained by the tempo estimation stage (see previous section) can be used to quantize the duration information extracted from the input MIDI file and relate the intervals to the corresponding figure duration.

Observe that splitting into measures is accomplished by applying thresholds to the accumulative sum of measure durations of the input.

If the accumulation of durations equals the threshold, the measure splitter gets the measure and stores it in the database since the measure is complete. If the accumulation of durations overpasses the threshold, then a tie between bars exists and the estimated time signature at the current point is assumed to be correct, although a note is between two bars. Also, the note that overpasses the measure duration must be split into two notes: one with the proper duration to complete the previous bar, and another one with the remaining duration that will be part of the following bar, to remove the tie.

The pitch contour of the rhythmic patterns obtained by the splitting scheme are stored (Fig. 42.3). Later, patterns with more contour versions will be selected with higher probability than others by the composition system. This choice is oriented to the replication of the probabilistic model of the rhythmic patterns in the melodies composed.

Pitch Progression

The pitch contour [42.46] is more important for the generation scheme than the notes themselves since, in order to maintain the personality and the style of the reused motives, the pitch contour must be preserved [42.55]. Note that the notes are specified by the MIDI messages.

Summing up, the actual notes are not necessary if a harmony corrector is used to adapt the melody to the chord progressions. Also, the use of variations instead of the unmodified pitch patterns provides flexibility so that the patterns can be adapted to the harmonies and, additionally, the output melody can be set up to any desired key signature.

42.3.2 Melody Generator

The melody generator will use the rhythmic and pitch information and the predefined chord progression stored in the database (Fig. 42.3) to create new melodies that replicate the style or complexity of the songs previously analyzed. The melody generation can be performed by means of the concatenation of rhythmic patterns according to composition rules defined by the analyzed music samples and by previously selected musical parameters (time signature, tempo for a chosen complexity level).

First of all, the initial tonality, the time signature, the number of bars and the dataset of parameters corresponding to a certain training level can be selected beforehand. Then, other specific parameters can be selected or modified: all these are presented in Table 42.1.

Then, some rules have been considered to be automatically applied in order to guide the pattern selection, to ensure the proper harmony adaptation and to guarantee the continuity of the pitch contour. In order to define the rules, *Schellenberg*'s simplification [42.56] of *Narmour*'s Realization-Expectation model [42.57] is perfectly suitable. A specific algorithm based on music theory concepts can be used for harmony adaptation at each measure [42.58]. Figure 42.4 shows a schematic representation of the stages of the melody generation algorithm.

In the next subsections, the steps performed by the melody generator proposed will be described in detail.

Pattern Selection

The items *i* the dataset that fulfill the time signature requirement (database level 1) chosen by the user beforehand will be selected. Then, among these patterns, stored in the database after the analysis stage, the ones required for the creation of the rhythmic structure will be selected. For example, if the melody structure is defined as A-B-B-A, then two rhythmic patters (from the database, level 2), will be acquired.

Table 42.1 Music generation: selectable parameters

Global level
• Initial tonality
• Time signature
• Number of bars
• Style database
Phrase level
• Predefined rhythmic pattern
• Predefined harmonic pattern
Measure level
• Tonality
• Chord
• Rhythmic pattern

After linking each measure in the structure to a particular rhythm pattern, a pitch contour is selected randomly among all the pitch version for each of the motives (from database, level 3).

Note that at this stage, the pitch progression selected may not be in accordance with the harmony set up. At a later stage, a chord transposition system should adapt the pitch curve to fulfill the given harmony progression keeping the continuity of the melodic curve.

Harmony Progression

A user can design a particular chord progression. However, note that the chord progression is a very important parameter for the musical success: there are combinations of chords that do not sound well together while others do [42.59]. So, in order to guide the selection of the chord progression, a set of harmonic progressions that sound well together can be defined [42.60].

The reason why some harmonic progressions sound well while others do not is related to the listener expectation [42.57], which is linked to the cultural environment and the preference of the listener for some chord transitions rather than others. The predefined progressions considered follow the Western music theory [42.60]. These progressions are I-ii-V-I, I-vii-I-V, I-I-IV-V or I-IV-V-I, among others.

Finally, in order to adapt the patterns selected in the previous stage, a melody transposition scheme based on music theory rules must be employed. This method will be described in next section.

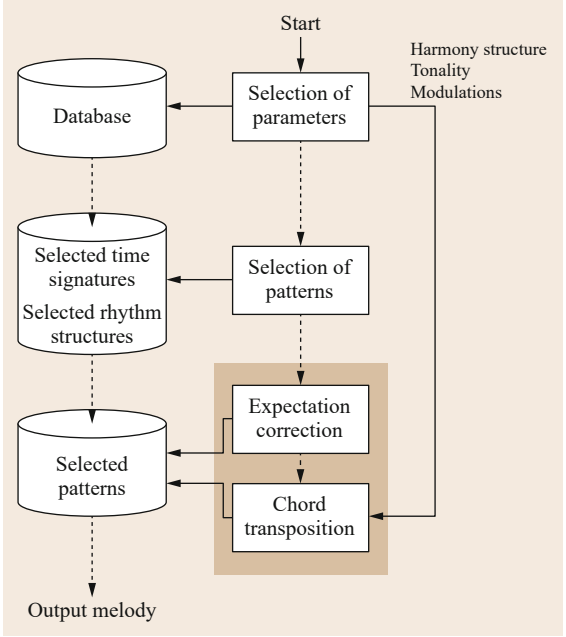

Fig. 42.4 Scheme of the melody generator

Chord Transposition

The chord transposition system must perform the proper changes to the sequence of notes to ensure that the harmony attained is the one selected and to guarantee the continuity of the melodic line according to the expectation model [42.56, 57]. This can be achieved by applying a musical harmony adaptation method based on level changing [42.58] together with additional constrains derived from the expectation model.

As the simple concatenation of patterns causes the appearance of transitions that do not sound natural, the idea to fix this issue is to generate a Narmour candidate [42.57] that properly follows the melodic line in the posterior measure. This candidate should fulfill the rules regarding the musical expectation.

The system analyses the last two notes of each measure to generate a new note. These two notes (called implication) are used to evaluate a third note (called realization), which will be the candidate note [42.56]. The following items are observed for the generation of the candidate notes [42.56]:

- Interval: A small interval [42.61] (less than a tritone) implies that the next note should follow the direction of the pitch progression. Otherwise, it would not achieve the expectation.
- Pitch jump: The pitch jump after a small interval should be similar to the previous one and in the same direction, according to the previous rule.
- Progression of the intervals:
  - If the implication interval is less than two semitones, then the third note should be back closer to the first note of the implication.
  - After a change in the direction or a large interval, the realization interval should be smaller than a tritone.

Recall that the position of the notes in the measures is key for the chord transposition stage. So, first, the chord notes, considered responsible of the harmony definition, and the nonchord notes, commonly called passing notes, are identified [42.59]. This process can be based on the analysis of the position of each note within each measure. The notes in downbeats will be considered chord notes, whilst those placed in upbeats will be considered nonchord notes.

Then, the chord transposition subsystem applies two different procedures to these two types of notes:

- Accented notes must belong to the chord
  - First chord note (or Narmour candidate [42.57]): This note is assigned to the closest pitch of the chord.
  - Secondary chord notes: Following the original pitch contour, secondary accented notes are moved to the closest pitch in the contour direction.
- Unaccented notes: The original interval between the previous note and the current note is replicated.

When the pitch and harmony adaptation processes are finished for every measure, the creation of a new melody is completed. Then, the performance on the melody by the user must be assessed.

42.4 Singing Assessment

In this section, we consider the problem of singing assessment for music learning. The descriptions will be based on the algorithm described in [42.62]. This algorithm evaluates the user's singing performance by comparing the processed audio against a reference melody.

In our case, the reference melody corresponds to the final output of the methods described in previous sections.

The main steps often required for the task at hand are illustrated in Fig. 42.5. These steps include the following global tasks: fundamental frequency (F_0) extraction and singing assessment based on F_0 alignment.

Next, we will briefly describe these steps following the approach selected, although other relevant schemes can be found in the literature [42.19, 25, 63].

42.4.1 F_0 Extraction

The Yin algorithm [42.29] has been found to be a good choice to extract the F_0 vector. This evolves from the idea of the autocorrelation method [42.64] to introduce relevant improvements. The modifications are based on the definition of the so-called cumulative mean normalized difference function $d'_t(\tau)$. This function peaks at the local period with lower deviations than the conventional autocorrelation function [42.29]. The cumulative mean normalized difference function is defined upon the squared difference function $d_t(\tau)$ given by

$$d_t(\tau) = \sum_{j=t}^{t+W} (x(j) - x(j+\tau))^2 \ , \qquad (42.1)$$

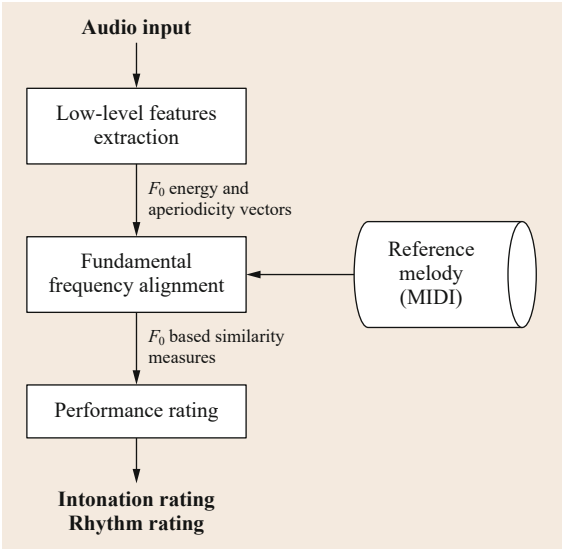

Fig. 42.5 A block diagram of the method for automatic singing assessment proposed in [42.62]

where $\tau \in [0, W)$ is an integer lag variable, W is the window size, $x(\tau)$ is the amplitude of the input signal x at time τ and t is the time index. Then, this function is normalized to give rise to the cumulative mean normalized difference function

$$d'_t(\tau) = \begin{cases} 1 & \tau = 0 \\ \dfrac{d_t(\tau)}{\frac{1}{\tau}\sum_{j=1}^{\tau} d_t(j)} & \text{otherwise} . \end{cases} \quad (42.2)$$

The Yin algorithm finds the local minimum in $d'_t(\tau)$ with the smallest τ'. Afterwards, a parabolic interpolation stage is performed using $d'_t(\tau')$, $d'_t(\tau'-1)$ and $d'_t(\tau'+1)$ to obtain an accurately estimated local minimum at τ_p. This value can be used to calculate the F_0 with $F_0 = f_s/\tau_p$, where f_s stands for the sampling rate.

The aperiodicity measure or voicing parameter is given by $ap = d'_t(\tau_p)$. This parameter is useful to identify voiced/unvoiced frames [42.62].

Note that the original Yin algorithm, implemented in Matlab, can be found in [42.65].

42.4.2 Assessment of Singing Voice

Once the F_0s of the user's performance and the reference melodies are extracted, they must be compared. A suitable method to align the functions for comparison is dynamic time warping (DTW) [42.66, 67]. This technique is useful for finding an optimal match between two sequences under certain restrictions. Note that the definition of the optimality criterion of the match strongly affects the performance of the alignment.

In [42.62], the cost matrix M for the DTW algorithm is defined as (other choices could be considered)

$$M_{ij} = \min\left\{(F_{0T}(i) - F_{0U}(j))^2, \alpha\right\} , \quad (42.3)$$

where $F_{0T}(i)$ is the F_0 of the target melody in the frame i, and $F_{0U}(j)$ represents the F_0 of the user's performance in the frame j. M_{ij} is the cost and α is a constant. Note that using this scheme, when the squared difference between F_0s becomes larger than α, the situation is considered to correspond to a spurious value and its contribution to the cost matrix is bounded.

The DTW algorithm uses the cost matrix to provide an optimal path $[i_k, j_k]$ for $k \in 1 \ldots K$, where K is the length of the path, matching the two input signals. Figure 42.6 illustrates the alignment performance.

In [42.68], a Matlab implementation of the DTW algorithm can be found.

DTW as a Similarity Measure

The path for the alignment between the user's performance and the reference melody conveys relevant information for singing evaluation. Actually, the DTW is suitable for assessing both the intonation and the rhythmic performance [42.62].

DTW to Assess Intonation. The cost matrix M provides information about the instantaneous deviation of the sung note with respect to the reference, as well as information about the overall F_0 deviation. Consequently, the total cost of the optimal alignment path found can be used as the similarity measure for intonation assessment. Then, the total intonation error (TIE) can be computed as

$$TIE = \sum_{k=1}^{K} M_{i_k j_k} , \quad (42.4)$$

where M is the cost matrix previously defined, and $[i_k, j_k]$, with $k \in 1 \ldots K$, represents each of the steps of the optimal path, K being the length of the path.

DTW to Assess Rhythm. DTW is also a powerful procedure for automatic rhythm assessment. The specific shape of the optimal path contains the necessary information about the rhythmic performance.

In the cost matrix of the DTW, a diagonal straight line represents a perfect rhythmic performance (no de-

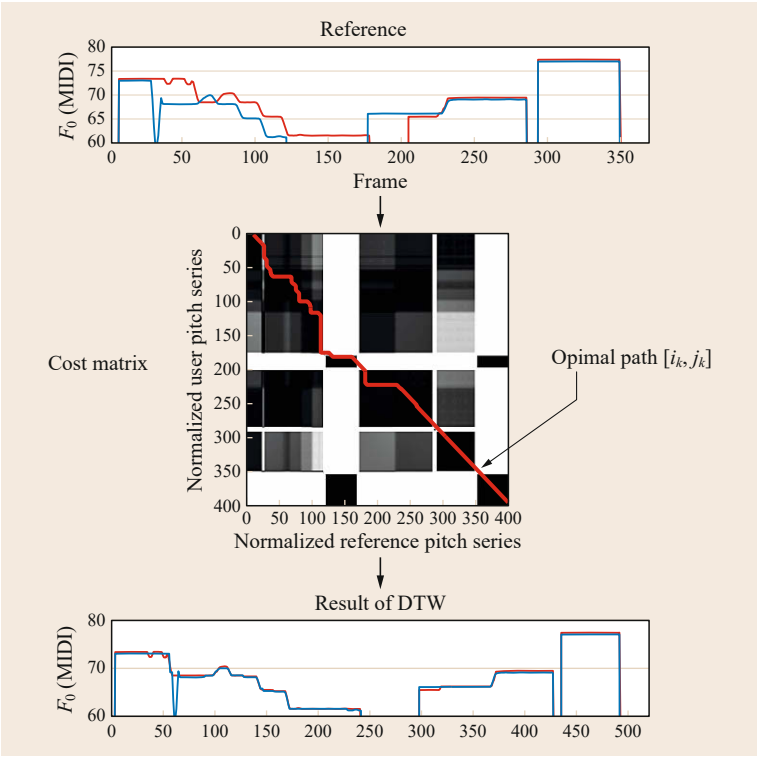

Fig. 42.6 F_0 alignment between a user's performance and the reference melody using dynamic time warping (DTW)

viation with respect to the target melody). A poor rhythmic performance would yield large deviations with respect to such a straight line. Figure 42.7 illustrates this idea.

The analysis of the deviations of the alignment path found with respect to the ideal path provides relevant rhythm assessment information. Specifically, a straight line with a slope different from the ideal one represents good rhythmic performance in a different tempo. On the other hand, the straightness of the path reveals the presence of erratic rhythmic errors. The straightness can be quantified by performing an ad hoc linear approximation to the path found, and then analyzing the error.

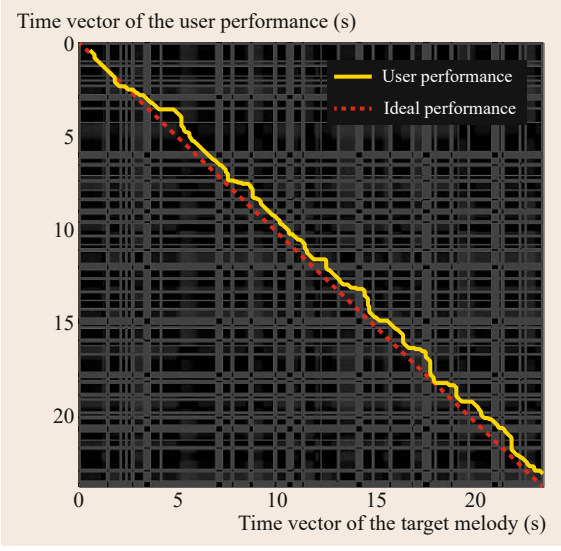

Fig. 42.7 Sample of the usage of DTW with F_0 signal for rhythm assessment. Rhythmically stable user's performance (*solid line*) and ideal rhythm performance (*dotted line*) ►

42.5 Summary

In this chapter, a complete approach to the development of computational tools for singing learning has been proposed.

Two main subsystems are required for the singing learning purpose: a module for the automatic generation of singing exercises with selectable complexity levels,

and a module for the automatic assessment of the user's singing performance.

A complete melody generator scheme, including the required analysis stages, has been presented. The generator described is able to automatically generate new melodies adapted to a certain music level selected beforehand.

An approach for the automatic assessment of singing voice has also been described. The method selected compares the F_0 of the user's performance against the reference F_0 of an automatically generated melody. The scheme provides an evaluation of both intonation and rhythm.

Acknowledgments. This work has been funded by Ministerio de Economía y Competitividad of the Spanish Government under Project No. TIN2016-75866-C3-2-R. This work has been done at Universidad de Málaga, Campus de Excelencia Internacional Andalucía Tech.

References

42.1 M.P. Ryynänen, A.P. Klapuri: Automatic transcription of melody, bass line, and chords in polyphonic music, Comput. Music J. **32**(3), 72–86 (2008)

42.2 J. Serrá, E. Gómez, P. Herrera: Audio cover song identification and similiraty: Background, approaches, evaluation, and beyond. In: *Advances in Music Information Retrieval*, Vol. 274, ed. by Z.W. Ras, A.A. Wieczorkowska (Springer, Berlin, Heidelberg 2010) pp. 307–332

42.3 S. Koelsch, W.A. Siebel: Towards a neural basis of music perception, Proc. TRENDS Cogn. Sci. **9**(12), 578–584 (2005)

42.4 R.F. Goldman: Ionisation; Density, 21.5; Integrales; Octandre; Hyperprism; Poeme Electronique, Musical Q. **47**(1), 133–134 (1961)

42.5 G. Nierhaus: *Algorithmic Composition: Paradigms of Automated Music Generation*, Vol. 34 (Springer, Wien 2010)

42.6 C. Uhle, J. Herre: Estimation of tempo, micro time and time signature from percussive music. In: *Proc. Int. Conf. Digital Audio Effects (DAFx)* (2003)

42.7 F. Gouyon, P. Herrera, P. Cano: Pulse-dependent analyses of percussive music, Proc. ICASSP **4**, 396–401 (2002)

42.8 S. Tojo, K. Hirata: Structural similarity based on time-span tree. In: *Proc. 9th Int. Symp. Comput. Music Model. Retriev. (CMMR)* (2012) pp. 645–660

42.9 M. Müller, D.P.W. Ellis, A. Klapuri, G. Richard: Signal processing for music analysis, IEEE J. Sel. Top. Signal Process. **5**(6), 1088–1110 (2011)

42.10 A. Van Der Merwe, W. Schulze: Music generation with Markov models, IEEE Multimed. **18**(3), 78–85 (2011)

42.11 M. Pearce, G. Wiggins: Towards a framework for the evaluation of machine compositions. In: *Proc. AISB'01 Symp. AI Creat. Arts Sci* (2001) pp. 22–32

42.12 D. Conklin: Music generation from statistical models. In: *Proc. Symp. Artif. Intell. Creat. Arts Sci. (AISB)* (2003) pp. 30–35

42.13 E.R. Miranda, J.A. Biles: *Evolutionary Computer Music* (Springer, London 2007)

42.14 D. Cope: Computer modeling of musical intelligence in EMI, Comput. Music J. **16**(2), 69–83 (1992)

42.15 D. Cope: *Computer Models of Musical Creativity* (MIT Press, Cambridge 2005)

42.16 M. Delgado, W. Fajardo, M. Molina-Solana: Inmamusys: Intelligent multiagent music system, Expert Syst. Appl. **36**(3), 4574–4580 (2009)

42.17 D.M. Howard, G. Welch, J. Brereton, E. Himonides, M. Decosta, J. Williams, A. Howard: WinSingad: A real-time display for the singing studio, Logop. Phoniatr. Vocology **29**(3), 135–144 (2004)

42.18 Barcelona Music and Audio Technologies: *SKORE Performance Rating*, http://skore.bmat.me (2008)

42.19 O. Mayor, J. Bonada, A. Loscos: The singing tutor: Expression categorization and segmentation of the singing voice. In: *Proc. AES 121st Convention* (2006)

42.20 D. Rossiter, D.M. Howard: ALBERT: A real-time visual feedback computer tool for professional vocal development, J. Voice Off. J. Voice Found. **10**(4), 321–336 (1996)

42.21 Sony Computer Entertainment Europe: *Singstar* (SCEE London Studios 2004)

42.22 T. Nakano, M. Goto, Y. Hiraga: An automatic singing skill evaluation method for unknown melodies using pitch interval accuracy and vibrato features. In: *Proc. INTERSPEECH (ICSLP)* (2006) pp. 1706–1709

42.23 J. Callaghan, P. Wilson: *How to Sing and See: Singing Pedagogy in the Digital Era* (Cantare Systems, Surry Hills 2004)

42.24 D. Hoppe, M. Sadakata, P. Desain: Development of real-time visual feedback assistance in singing training: A review, J. Comput. Assist. Learn. **22**(4), 308–316 (2006)

42.25 S. Grollmisch, E. Cano Cerón, C. Dittmar: Songs2see: Learn to play by playing. In: *41st Int. Audio Eng. Soc. Conf. (AES)* (2011)

42.26 Z. Jin, J. Jia, Y. Liu, Y. Wang, L. Cai: An automatic grading method for singing evaluation, Rec. Adv. Comput. Sci. Inf. Eng. **5**, 691–696 (2012)

42.27 C. Dittmar, E. Cano, J. Abeßer, S. Grollmisch: Music information retrieval meets music education, Multimod. Music Process. **3**, 95–120 (2012)

42.28 E. Gómez, A. Klapuri, B. Meudic: Melody description and extraction in the context of music content processing, J. New Music Res. **32**(1), 23–40 (2003)

42.29 A. De Cheveigné, H. Kawahara: YIN, a fundamental frequency estimator for speech and music, J. Acoust. Soc. Am. **111**(4), 1917 (2002)

42.30 T. Viitaniemi, A. Klapuri, A. Eronen: A probabilistic model for the transcription of single-voice

melodies. In: *Proc. 2003 Finn. Signal Process. Symp. FINSIG'03* (2003) pp. 59–63
42.31 M. Ryynänen, A. Klapuri: Modelling of Note Events for Singing Transcription. In: *Proc. ISCA Tutor. Res. Workshop Stat. Percept. Audio Process. (SAPA)* (2004)
42.32 G.E. Poliner, D.P.W. Ellis, A.F. Ehmann, E. Gómez, S. Streich, B. Ong: Melody transcription from music audio: Approaches and evaluation, IEEE Trans. Audio Speech Lang. Process. **15**(4), 1247–1256 (2007)
42.33 E. Molina: *Automatic Scoring of Signing Voice Based on Melodic Similarity Measures* (Universitat Pompeu Fabra, Barcelona 2012)
42.34 R.J. McNab, L.A. Smith, I.H. Witten: Signal processing for melody transcription, Proc. 19th Australas. Comput. Sci. Conf. **18**(4), 301–307 (1996)
42.35 M. Ryynänen: Singing transcription. In: *Signal Processing Methods for Music Transcription*, ed. by A. Klapuri, M. Davy (Springer Science/Business Media LLC, New York 2006) pp. 361–390
42.36 J.J. Mestres, J.B. Sanjaume, M. De Boer, A.L. Mira: *Audio Recording Analysis and Rating*, US Patent 8158871 (2012)
42.37 G. Haus, E. Pollastri: An audio front end for query-by-humming systems. In: *Proc. 2nd Int. Symp. Music Inf. Retriev. (ISMIR)* (2001) pp. 65–72
42.38 W. Krige, T. Herbst, T. Niesler: Explicit transition modelling for automatic singing transcription, J. New Music Res. **37**(4), 311–324 (2008)
42.39 E. Molina: Hacer música... para aprender a componer, Eufonia, Didáct. Músic. **51**, 53–64 (2011)
42.40 M.K. Shan, S.C. Chiu: Algorithmic compositions based on discovered musical patterns, Multimed. Tools Appl. **46**(1), 1–23 (2010)
42.41 P.J. Ponce de León: Statistical description models for melody analysis and characterization. In: *Proc. Int. Comput. Music Conf.*, ed. by J.M. Iñesta (2004) pp. 149–156
42.42 Association MIDI Manufacturers: *The Complete MIDI 1.0 Detailed Specification* (The MIDI Manufacturers Association, Los Angeles 1996)
42.43 R.S. Brindle: *Musical Composition* (Oxford Univ. Press, Oxford 1986)
42.44 F. Lerdahl, R. Jackendoff: *A Generative Theory of Tonal Music* (MIT Press, Cambridge 1983)
42.45 W.T. Fitch, A.J. Rosenfeld: Perception and production of syncopated rhythms, Music Percept. **25**, 43–58 (2007)
42.46 W. Appel: *Harvard Dictionary of Music*, 2nd edn. (The Belknap Press of Harvard Univ., Cambridge, London 2000)
42.47 K. Seyerlehner, G. Widmer, D. Schnitzer: From rhythm patterns to perceived tempo. In: *Int. Soc. Music Inf. Retriev. (ISMIR)* (2007) pp. 519–524
42.48 M.F. McKinney, D. Moelants: *Ambiguity in Tempo Perception: What Draws Listeners to Different Metrical Levels?* (Univ. of California Press, Oakland 2006) pp. 155–166
42.49 M. Gainza, D. Barry, E. Coyle: Automatic bar line segmentation. In: *123rd Convent. Audio Eng. Soc. Convent. Paper* (2007)
42.50 M. Gainza, E. Coyle: Time signature detection by using a multi resolution audio similarity matrix. In: *122nd Convent. Audio Eng. Soc. Convent. Paper* (2007)
42.51 J. Foote, M. Cooper: Visualizing musical structure and rhythm via self-similarity. In: *Proc. 2001 Int. Comput. Music Conf.* (2001) pp. 419–422
42.52 J.R. Quinlan: *C4.5: Programs for Machine Learning* (Morgan Kaufmann, San Francisco 1993)
42.53 J. Platt: *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines* (Microsoft Research, Redmond 1998)
42.54 M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten: The WEKA data mining software: An update, SIGKDD Explor. **11**(1), 10–18 (2009)
42.55 W.J. Downling, D.S. Fujitani: Contour, interval and pitch recognition in memory for melodies, J. Acoust. Soc. Am. **49**, 524–531 (1971)
42.56 E. Schellenberg: Simplifying the implication-realization model of musical expectancy, Music Percept. **14**(3), 295–318 (1997)
42.57 E. Narmour: *The Analysis and Cognition of Melodic Complexity: The Implication-Realization Model* (Univ. of Chicago Press, Chicago, London 1992)
42.58 D. Roca, E. Molina (Eds.): *Vademecum Musical* (Enclave Creativa, Madrid 2006)
42.59 B. Benward: *Music: In Theory and Practice*, Vol. 1, 7th edn. (McGraw-Hill, New York 2003)
42.60 R.W. Ottman: *Elementary Harmony: Theory and Practice*, 5th edn. (Prentice Hall, Englewood Cliffs 1989)
42.61 A.E. Yilmaz, Z. Telatar: Note-against-note two-voice counterpoint by means of fuzzy logic, Knowl.-Based Syst. **23**(3), 256–266 (2010)
42.62 E. Molina, I. Barbancho, E. Gomez, A.M. Barbancho, L.J. Tardon: Fundamental frequency alignment vs. note-based melodic similarity for singing voice assessment. In: *IEEE Int. Conf. on Acoust. Speech Signal Process. (ICASSP)* (2013) pp. 744–748
42.63 J. Wapnick, E. Ekholm: Expert consensus in solo voice performance evaluation, J. Voice **11**(4), 429–436 (1997)
42.64 L.R. Rabiner, R.W. Schafer: *Digital Processing of Speech Signals*, Prentice-Hall Series in Signal Processing No. 7, Vol. 25 (Prentice Hall, Englewood Cliffs 1978) p. 290
42.65 A. De Cheveigné: *Matlab Implementation of YIN Algorithm*, http://audition.ens.fr/adc/sw/yin.zip (2012)
42.66 H. Sakoe: Dynamic programming algorithm optimization for spoken word recognition, IEEE Trans. Acoust. Speech Signal Process. **26**, 43–49 (1978)
42.67 C.A. Ratanamahatana, E. Keogh: Everything you know about dynamic time warping is wrong. In: *3rd Workshop Min. Tempor. Seq. Data, 10th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. (KDD-2004)* (2004)
42.68 D. Ellis: *Dynamic Time Warp (DTW) in Matlab*, http://labrosa.ee.columbia.edu/matlab/dtw (2003)