# AUTOMATIC BASS LINE TRANSCRIPTION FROM STREAMING POLYPHONIC AUDIO

*Matti Ryynänen and Anssi Klapuri*

Tampere University of Technology
Institute of Signal Processing
P.O.Box 553, FI-33101 Tampere, Finland
{matti.ryynanen, anssi.klapuri}@tut.fi

## ABSTRACT

This paper proposes a method for the automatic transcription of the bass line in polyphonic music. The method uses a multiple-F0 estimator as a front-end and this is followed by acoustic and musicological models. The acoustic modeling consists of separate models for bass notes and rests. The musicological model estimates the key and determines probabilities for the transitions between notes using a conventional bigram or a variable-order Markov model. The transcription is obtained with Viterbi decoding through the note and rest models. In addition, a causal algorithm is presented which allows transcription of streaming audio. The method was evaluated using 87 minutes of music from the RWC Popular Music Database. Recall and precision rates of 64% and 60%, respectively, were achieved for discrete note events.

***Index Terms***— Music, Modeling, Hidden Markov models, Audio systems, Viterbi decoding

## 1. INTRODUCTION

Bass line transcription refers to the automatic extraction of a parametric representation (e.g., a MIDI file) for the bass notes in a polyphonic music signal. The term *bass line* refers to an organized sequence of consecutive notes and rests played with a bass guitar, a double bass, or a bass synthesizer. A *note* has a single pitch (a note name), a beginning (onset) time, and an ending (offset) time. Together with the melody, the bass line is an important characteristic of a song in several music styles, especially in popular music. Bass line transcription provides a useful tool for bass players and musicians, music information retrieval applications, and music transcription and analysis software.

Bass line transcription has been previously studied by Goto [1] and by Hainsworth and Macleod [2]. Goto considered the fundamental frequency (F0) estimation of the bass and melody lines. The method of Hainsworth and Macleod consisted of consecutive steps of note onset location, note hypothesis generation, and determination of bass notes from the hypotheses.

Figure 1 shows a block diagram of our transcription method. An audio signal is processed frame-wise with two feature extractors: a multiple-F0 estimator and an accent estimator. The acoustic model uses these features to derive a hidden Markov model (HMM) for bass notes and a Gaussian mixture model (GMM) for rests. The upper F0 limit is estimated frame-by-frame to reduce the interference of other instruments. A musicological model uses the F0s to estimate the musical key and to choose between-note transition probabilities which are modeled with a bigram or a *variable-order Markov model (VMM)*. Viterbi decoding finds a path through the models and produces a transcribed sequence of notes and rests. An optional post-processing step trains a VMM for the transcribed sequence and uses it to determine transition probabilities during the second pass of Viterbi decoding.

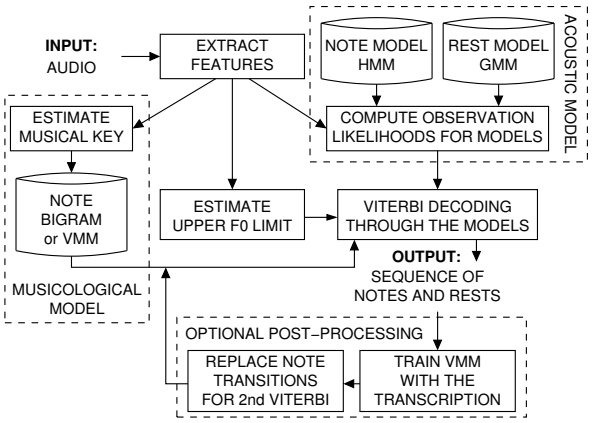


**Fig. 1**. The block diagram of the transcription method.

The architecture of the proposed method is similar to our method for melody transcription [3]. However, the acoustic and musicological models are tailored for bass line transcription and we investigated the suitability of VMMs for capturing the repetitive nature of bass note patterns. In addition, a causal algorithm is presented to enable transcription of streaming audio.

For training and evaluating the method, we use the RWC (Real World Computing) Popular Music Database consisting of 100 acoustic recordings of typical pop songs [4]. For each recording, the database includes a MIDI file which contains a manual annotation of the bass notes, hereafter referred to as the *reference notes*. We use audio segments between 30 and 90 seconds of 87 recordings including bass notes, resulting in 87 minutes of audio in total. We limit the possible F0 range to 36–254 Hz, and remove reference notes outside the range both from training and from evaluation (1.6% of the original reference notes). On average, a song contains approximately 170 reference notes with a standard deviation of 80 notes.

## 2. FEATURE EXTRACTION

The front-end of the method consists of two frame-wise feature extractors: a multiple-F0 estimator and an accent estimator. Stereo input audio is mixed to mono before feature extraction.

This work was supported by the Academy of Finland, project No. 213462 (Finnish Centre of Excellence program 2006 - 2011)

### 2.1. Multiple-F0 Estimation

We use the multiple-F0 estimator proposed in [5]. Briefly, the method calculates a harmonic transform of the Fourier spectrum and then picks multiple F0 values by iterative detection and cancellation. The estimator analyzes audio in overlapping 92.9 ms frames with 23.2 ms interval between the beginnings of successive frames. The F0 search range is limited to 36–254 Hz (MIDI notes 26–59). As an output, the estimator produces four F0 estimates and their saliences in each frame. For the $i$:th *F0 estimate* $x_{it}$ in frame $t$, $i \in [1, 4]$, the *salience value* $s_{it}$ roughly expresses the strength of $x_{it}$. The F0 values are expressed as unrounded MIDI note numbers by $69 + 12 \log_2(\mathrm{F0}/440)$. The salience values are normalized to zero mean and unit variance over the training data.

### 2.2. Accent Signal for Note Onsets

The *accent signal* $a_t$ measures the degree of phenomenal accent in frame $t$ based on incoming spectral energy, and in practice indicates potential note onsets. The accent estimation method proposed in [6] is used in a manner similar to [3]. The accent values are normalized to zero mean and unit variance over the training data.

## 3. UPPER F0 LIMIT ESTIMATION

The upper F0 limit estimation aims at setting the maximum F0 for possible output MIDI notes in each frame. This reduces both the interference of higher notes from other instruments and the computational load due to a smaller search space.

The upper limit is estimated by calculating salience-weighted mean of F0s in frame $t$, $y(t) = (\sum_i x_{it} s_{it})/(\sum_i s_{it})$, where salience values $s_{it}$ are used before normalization. The function $y(t)$ is smoothed to obtain $\tilde{y}(t), t > 0$ using attack-release averaging by Eqs. (1)-(3) after initializing $\tilde{y}(0) = 59$.

$$\tau = \begin{cases} \tau_{\mathrm{a}}\,, & \text{if } y(t) > \tilde{y}(t-1) \\ \tau_{\mathrm{r}}\,, & \text{otherwise} \end{cases}\,, \tag{1}$$

$$g = \exp(-1/(\tau f_r)), \tag{2}$$

$$\tilde{y}(t) = (1-g)y(t) + g\tilde{y}(t-1), \tag{3}$$

where $f_r$ is the frame rate. The time-constants $\tau_{\mathrm{a}} = 0.14$ s and $\tau_{\mathrm{r}} = 2.3$ s control attack and release of the averaging, respectively. We obtain the upper F0 limit by $\lceil \tilde{y}(t) - c \rceil$, where $\lceil \cdot \rceil$ is the ceiling function and $c$ is a constant offset. We found $c = 4$ to perform best in simulations.

## 4. NOTE AND REST MODELING

Bass notes are modeled with a 3-state left-to-right HMM in a manner similar to [3]. The note HMM state $q_\ell, \ell \in [1, 3]$, represents the typical values of the features in the $\ell$:th temporal segment of note events. One note HMM is allocated for each MIDI note. The observation vector $\mathbf{o}_{n,t}$ is defined for a note HMM with nominal pitch $n$ in frame $t$ as

$$\mathbf{o}_{n,t} = [\Delta x_{n,t},\ s_{jt},\ a_t]^T\,, \tag{4}$$

where $\Delta x_{n,t} = x_{jt} - n$ is the difference between the measured F0 estimate $x_{jt}$ and the nominal pitch $n$ of the modeled note. Index $j$ is obtained using Eqs. (5)-(6).

$$m = \arg\max_i \{s_{it}\}\,, \tag{5}$$

$$j = \begin{cases} m\,, & \text{if } |x_{mt} - n| \le 1 \\ \arg\min_i \{|x_{it} - n|\}\,, & \text{otherwise.} \end{cases} \tag{6}$$

For a note model $n$, the maximum-salience F0 estimate and its salience value are associated with the note if the absolute F0 difference is less or equal to one semitone, otherwise the nearest F0 estimate and its salience are used. An observation vector thus consists of the F0 difference $\Delta x_{n,t}$, its corresponding salience value $s_{jt}$, and the accent signal value $a_t$.

The note model is trained as follows. For the time region of a reference note with nominal pitch $n$, the observation vectors by Eq. (4) constitute a training sequence. The observation sequence is accepted for the training only if the median of the absolute F0 differences $|\Delta x_{n,t}|$ during the note is less than one semitone. The note HMM parameters are then obtained using the Baum-Welch algorithm, where the observation likelihood distributions are modeled with four-component GMMs.

The rest segments, i.e., the time segments where no bass notes are sounding, are modeled by a four-component GMM (analogous to a 1-state HMM). The observation vector for rest is the maximum salience in each frame $\mathbf{o}_{r,t} = \max_i\{s_{it}\}$.

## 5. MUSICOLOGICAL MODELING

The musicological model controls transition probabilities between the note models and the rest model. The musicological modeling is based on the fact that some notes are more probable than others given a musical key and the preceding notes. Therefore, we need a key estimator and a model for bass note sequences to give transition probabilities. If the musicological model is disabled, we use uniform transition probabilities.

### 5.1. Key Estimation for Bass

Key estimation aims at finding the *relative-key pair*, e.g., C major / A minor, from the F0 estimates. Let $k \in \{0, 1, \ldots, 11\}$ denote the relative-key pairs C major / A minor, D♭ major / B♭ minor, and so forth until pair B major / G♯ minor, respectively. Given the F0 estimates $x_{it}$ and the *key profiles* $P_{\mathrm{maj}}$ and $P_{\mathrm{min}}$ (see Fig. 2), the most probable relative-key pair $\widetilde{K}(t), t > 0$ is calculated as follows:

$$d(x, k) = \mathrm{mod}(\langle x \rangle - k, 12) \tag{7}$$

$$K(k,t) = K(k, t-1) + \sum_i \Big[ \log P_{\mathrm{maj}}(d(x_{it}, k)) + \log P_{\mathrm{min}}(d(x_{it}, k+9)) \Big] \tag{8}$$

$$\widetilde{K}(t) = \arg\max_k \{K(k,t)\} \tag{9}$$

Equation (7) gives the distance of an F0 estimate to the tonic note of the key $k$, where $\mathrm{mod}(\cdot,\cdot)$ is the integer modulo operator and $\langle \cdot \rangle$ is the nearest integer function. By initializing $K(k, t = 0)$ with a uniform distribution over $k$, Eq. (8) then recursively calculates the likelihood for each key based on the F0 estimates $x_{it}$ and the key profiles $P_{\mathrm{maj}}(d)$ and $P_{\mathrm{min}}(d)$. The profiles define the likelihoods for distance $d \in \{0, 1, \ldots, 11\}$ in major and minor keys, respectively. When calculating distances in minor keys, term $+9$ in Eq. (8) shifts the key index $k$ to the relative minor key. Equation (9) gives the most likely key $\widetilde{K}(t)$ in frame $t$.

For melody transcription [3], we successfully used the key profiles reported by Krumhansl in [7, p. 67]. For bass notes, however, we discovered that using artificial profiles consisting of only tonic, perfect fourth, and perfect fifth performed better. We refer to these as I-IV-V profiles. Bass lines usually contain lots of these notes with

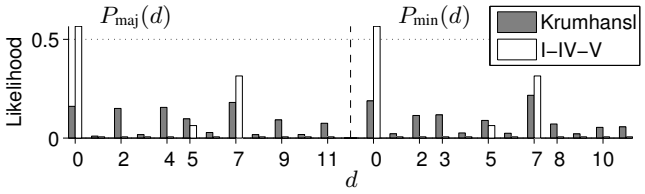


**Fig. 2**. The Krumhansl and the proposed I-IV-V key profiles.

respect to the key, and therefore lead to a better performing model. The Krumhansl and the I-IV-V profiles are illustrated in Fig. 2. The key estimation results for both profiles are reported in Sect. 7.

### 5.2. Modeling Bass Note Sequences

The bass lines usually contain a great deal of repetitions and note patterns which are conventionally used in bass playing. Therefore, it should be beneficial to use a predictor to give a probability for moving to a note given the preceding notes. For this purpose, we have successfully used a note bigram (i.e., only the preceding note affects the probability) in melody transcription [3]. Due to a more repetitive nature of bass lines, however, a better predictor would utilize several preceding notes. One possible solution is to use a note $N$-gram which requires a lot of memory for large $N$. A better solution is to use a VMM for which the context length $N$ (limited by a predefined maximum context length $N_{\text{max}}$) varies in response to the available statistics in the training data. This is a very desirable feature, and for note sequences, this means that both short and long note sequences can be modeled based on their occurrence in the training data within a single model. Begleiter *et al.* published an article on prediction with VMMs and provided a toolbox of algorithms for VMM training and prediction [8]. In our simulations, we use the prediction by partial matching (PPM) algorithm from the toolbox.

We train a note bigram and a VMM by using a collection of over 1300 MIDI files including bass lines. For each file, we estimate a relative-key pair using the Krumhansl profiles and all MIDI notes from the file. Based on the estimate, the bass notes are shifted both up and down to C major to obtain two note sequences one octave apart from each other. A rest is added between two consecutive notes if they are separated by more than 200 ms. We obtain a key-normalized predictor by training with these bass-note sequences. While calculating the transition probabilities during transcription, the key estimate of Eq. (9) is used to match the notes with the key-normalized predictor.

## 6. VITERBI DECODING AND POST-PROCESSING

The note models and the rest model constitute a network where the transitions probabilities between the models are given by the musicological model. Viterbi decoding finds a path through the models and produces a sequence of notes and rests as the transcription of the bass line. The entire method operates faster than real-time on a PC with a 1.7 GHz Pentium 4 processor. Figure 3 shows an example transcription.

### 6.1. Blockwise Backtracking for Streaming Audio

Conventionally, Viterbi decoding decides the best path by backtracking from the last frame of the song. We noticed that after some frames, however, the best path remains unchanged and it is possible

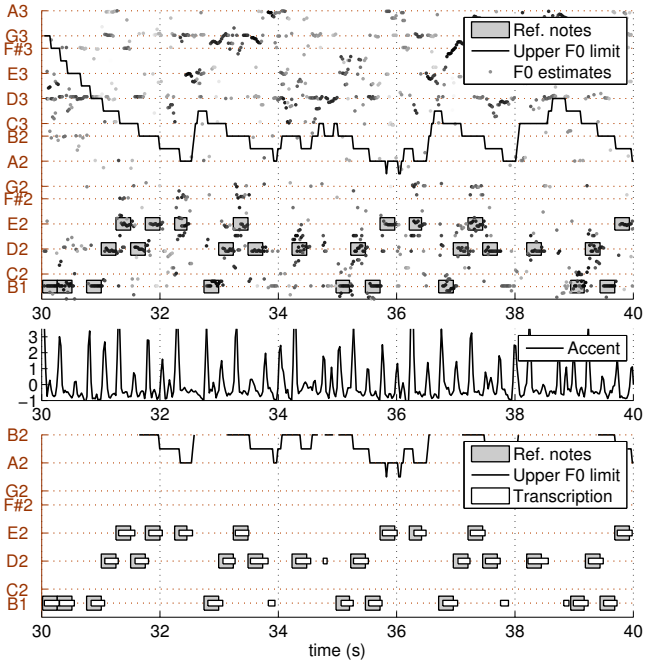


**Fig. 3**. An excerpt from song RWC-MDB-P-2001 No. 61 showing the reference notes, F0 estimates (salience indicated by intensity) and accent, the estimated upper F0 limit, the notes of estimated key G major / E minor, and the transcription.

to fix the the path (and the transcription) before processing the entire song. This enables causal transcription of streaming audio.

The backtracking is performed blockwise as follows: given the block size $b$, process frames $(t-b, t]$ and update the key estimate $\widetilde{K}(t)$ and the note transition probabilities accordingly. Then backtrack through frames $(t-2b, t]$ and fix the path in frames $(t-2b, t-b]$. Then replace $t$ with $t+b$ and continue. In other words, backtracking is performed in 50% overlapping blocks of $2b$ frames where the best path is fixed for the first $b$ frames. In the simulations, a block size of half a second ($b = 23$) was sufficient to produce similar results as the conventional backtracking.

### 6.2. Post-processing with a Retrained VMM

An optional post-processing step trains a VMM from the individual transcribed sequence of notes and rests, and uses it to give the transition probabilities during the second Viterbi decoding pass. This aims at adapting to the song by capturing the possible bass-note patterns present in the first transcription. The key estimation is not needed during the second pass since the trained VMM includes the key information in the actual note patterns.

## 7. RESULTS

The method was evaluated using three-fold cross-validation on the 87 songs in the database. We used the recall rate $R$ and the precision rate $P$ defined by

$$R = \frac{\#(\text{correctly trans. notes})}{\#(\text{reference notes})}, P = \frac{\#(\text{correctly trans. notes})}{\#(\text{transcribed notes})} \quad (10)$$

**Table 1**. Transcription results using the standard Viterbi (%).

| Method | $R$ | $P$ | $F$ |
|---|---|---|---|
| No musicological model: | | | |
| Acoustic models | 56.5 | 49.2 | 50.8 |
| Upper F0 limit estimation | 60.7 | 58.2 | **57.4** |
| Musicological modeling: | | | |
| Bigram note transitions | 63.1 | 58.5 | 58.8 |
| VMM note transitions | 64.0 | 59.7 | **59.9** |
| VMM post-processing | 62.8 | 61.3 | 60.2 |

**Table 2**. Transcription results using blockwise backtracking and changing key estimates (%).

| Method | $R$ | $P$ | $F$ |
|---|---|---|---|
| Bigram note transitions | 63.1 | 58.6 | 58.8 |
| VMM note transitions | 63.8 | 59.6 | 59.8 |

as performance criteria. A reference note is correctly transcribed by a note in the transcription if their MIDI note numbers are equal, the absolute difference between their onset times is less than 150 ms, and the transcribed note is not already associated with another reference note. We used the F-measure $F = 2RP/(R+P)$ to give an overall measure of performance. The recall rate, the precision rate, and the F-measure were calculated separately for the transcriptions of each recording, and the average over all the transcriptions is reported for each criterion.

### 7.1. Transcription Results

Table 1 summarizes the transcription results for different simulation setups when using the standard Viterbi decoding, where the backtracking starts from the last frame $t_{\max}$ and the key estimate $\widetilde{K}(t_{\max})$ is used throughout the transcription. When using only the acoustic models, the results are not very good due to disabling the musicological model. The upper F0 limit estimation greatly improves the results. The musicological model further improves results where the VMM for transition probabilities works better than the note bigram. For the VMM, the best performing maximum context length $N_{\max}$ was five preceding notes. The post-processing is simulated for the best performing method (VMM note transitions), and it slightly improves the results. For the post-processing VMM, the best maximum context length was $N_{\max} = 8$. The most common errors include octave errors, missing note onsets, and confusions with other instruments. For all setups, the average temporal overlap between transcribed and reference notes was 62%.

Table 2 shows the results when using the blockwise backtracking. The acoustic models both with and without the upper F0 limit estimation give exactly the same results as the standard Viterbi. The causal backtracking seems to work equally well with the standard Viterbi decoding although the key estimate and the note transition probabilities may change between each block of frames.

### 7.2. Key Estimation Results

We measure the performance of the key estimation method by comparing the estimated keys to the manually annotated reference keys in the database. The evaluation criterion is the key signature distance on the circle of fifths between the reference key and the estimated relative-key pair. Table 3 shows the key estimation results. The percents indicate the time spent in estimated keys over the database. The first two rows show the results with the last key estimate $\widetilde{K}(t_{\max})$ used with the standard Viterbi. The lower two rows show the results for blockwise updated key used in the causal method. In both cases, the proposed I-IV-V profiles estimate the correct key more often than the Krumhansl profiles. The blockwise updated key expectedly performs worse than using $\widetilde{K}(t_{\max})$ which gathers evidence regarding the key over the entire song.

**Table 3**. Key estimation results.

| Key signature distance | 0 | 1 | 2 | $\geq 3$ |
|---|---|---|---|---|
| Krumhansl profiles, $\widetilde{K}(t_{\max})$ | 57.2 | 34.7 | 2.0 | 6.1 |
| I-IV-V profiles, $\widetilde{K}(t_{\max})$ | 80.4 | 9.8 | 3.2 | 6.6 |
| Krumhansl profiles, $\widetilde{K}(t)$ | 49.4 | 36.3 | 3.8 | 10.6 |
| I-IV-V profiles, $\widetilde{K}(t)$ | 69.7 | 16.9 | 4.8 | 8.6 |

## 8. CONCLUSIONS

We have described a method for transcribing the bass line in polyphonic music. The method successfully employed VMMs in modeling bass note sequences. In addition, the method was shown to work also as a causal version, thus enabling the transcription of streaming audio. The method was evaluated using 87 minutes of polyphonic popular music and achieved good results. Audio demonstrations are available at
`http://www.cs.tut.fi/sgn/arg/matti/demos/basstrans`.

## 9. REFERENCES


[1] M. Goto, "A real-time music-scene-description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals," *Speech Communication*, vol. 43, no. 4, pp. 311–329, 2004.

[2] S. W. Hainsworth and M. D. Macleod, "Automatic bass line transcription from polyphonic music," in *Proc. International Computer Music Conference*, Havana, Cuba, 2001.

[3] M. Ryynänen and A. Klapuri, "Transcription of the singing melody in polyphonic music," in *Proc. 7th International Conference on Music Information Retrieval*, Victoria, Canada, Oct. 2006.

[4] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Popular, classical, and jazz music databases," in *Proc. 3rd International Conference on Music Information Retrieval*, Oct. 2002.

[5] A. Klapuri, "Multiple fundamental frequency estimation by summing harmonic amplitudes," in *Proc. 7th International Conference on Music Information Retrieval*, Victoria, Canada, Oct. 2006.

[6] A. P. Klapuri, A. J. Eronen, and J. T. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 342–355, Jan. 2006.

[7] C. Krumhansl, *Cognitive Foundations of Musical Pitch*, Oxford University Press, 1990.

[8] R. Begleiter, R. El-Yaniv, and G. Yona, "On prediction using variable order Markov models," *J. of Artificial Intelligence Research*, vol. 22, pp. 385–421, 2004.