# GimmeDaBlues: An Intelligent Jazz/Blues Player And Comping Generator for iOS devices

Rui Dias[1], Telmo Marques[2], George Sioros[1], and Carlos Guedes[1]

[1] INESC-Porto / Porto University, Portugal
ruidias74@gmail.com
gsioros@hotmail.com
cguedes@fe.up.pt
[2] CITAR / Universidade Católica do Porto, Portugal
telmomarques33@gmail.com

**Abstract.** This paper describes an application for iPhone/iPod Touch/iPad devices that allows anyone to play jazz keyboard and solo instruments along a predefined harmonic progression, using the multi-touch properties of the iOS devices. While the user plays keyboard and/or solo instruments, the application automatically generates the bass and drums parts, responding to the user's activity.
Dynamic mapping of the notes and chords available in the graphical interface provides an intuitive and natural way to play otherwise complex chords and scales, while maintaining a physical playability that will be familiar to experienced keyboard players, and provides an entertaining, yet challenging experience for non-musicians.

**Keywords:** Automatic Music Generation, Blues, Jazz, iOS, Entertainment.

## 1 Introduction

### 1.1 GimmeDaBlues Overview

It is widely accepted that Jazz – and this includes the Blues – is a musical style directly related with improvisation and interaction. But how do Jazz musician interact? How do they improvise? It is beyond the extent of this study to answer to such questions, but they trace a new demand (this time pivotal to our application): Could this be transferred to a human-computer environment in a way that either a competent jazz musician or an inexperienced layman could reach the interaction and improvisation proficiency?

9th International Symposium on Computer Music Modelling and Retrieval (CMMR 2012)
19-22 June 2012, Queen Mary University of London
All rights remain with the authors.

Existing applications apparently respond partially to these demands. Apps like ThumbJam [1] have an instrument-like approach, providing an interface for expressive playing of sample-based sounds and allowing and flexible mapping of the selected sound. This approach, however, is passive, in that the app doesn't have any kind of information, intelligence or participation on the resulting musical content.
Knowledge from well-known jazz algorithmic improvisation systems like Genjam [2], and Bob [3], provide the base for the creation of intelligent and informed music systems.
*GimmeDaBlues* — from now on GdB — was developed to fulfil all the purposes for such question: an application for jazz and blues lovers, whether musicians or non-musicians, having playability and ease of use in mind.

For musicians the application can be entertaining and funny. People interact with the bass and the drums while playing the piano or the Hammond-like organ. Improvising simultaneously with the trumpet will give a sense of leadership and control of the entire combo. For non-musicians this can also be challenging and instructive: although GdB has a jazz theory background that contributes to a clean academic performance, timings and groove are entirely up to the user, so that to sound jazzy, the user has to be familiarized with jazz.

### 1.2 Multi-touch surfaces musical controllers and iOS

In computing interaction within hardware and software, *multi-touch* means a 'touch sensing surface's (trackpad or touchscreen) ability to recognize the presence of two or more points of contact with the surface [4]'. This plural-point awareness allows multiple fingering functionality – surely important in our application – like the use of both hands, both thumb manipulation on the keyboard region, and the *solo-like* fingering on the soloist instrument region defined by the trumpet image. This multi-touch surface method is nowadays widely used by almost every touch screen phones and notebooks, but what makes the iPhone, the iPod Touch, and the iPad devices unique is their quick response to "swipes, pinches, and finger presses [4]. In rhythmical styles such as jazz and blues this features are absolutely primordial for a musical controller. Once these three surfaces are compatible in terms of iOS the application extents to all of them three.

## 2 Interface

The GdB interface is divided in two main areas in the center, plus the volume controls for the bass and drums on the left and right edges of the screen (Fig. 1). The main areas on the center are the instrument areas. The upper half is the solo instrument while the lower half is the keyboard (chord) instrument. It is not expected to play the right keys especially on such a messy keyboard. Specific keys are not important but moving from left to right on the keyboard results on chords or notes from low to high according to the zone pressed. The small button on the top left drops a menu with options for *Record*, *Setup*, *Library* and *About* screens.

**Fig. 1** Main Window

**Recording.** The Record button starts recording the current session as the user plays. Both the user's instruments and the automatically generated bass and drums will be recorded. When recording is on, a stop button emerges on the top right corner permitting to stop recording whenever we want. Stopping the recording will show a menu with options to save the session in the library or discard it.

**Setup.** In the *Setup* menu, the user can choose the *Instruments*, *Style*, *Root Key* and *Tempo*. Each option will show the corresponding selection list.
The currently available instruments for the solo (upper) instrument are *Trumpet*, *Bright Trumpet* and *Scoop Down Trumpet*. For the keyboard (lower) instrument the user can choose a *Piano* or a *Blues Organ* sound.
The *Style* option shows a list of the available song styles. Each song style is a well-known jazz/blues chord structure, like a *Classical* (major) *Blues*, or *Minor Blues*.
The *Root Key* changes the song key, transposing all the events in the current style to the selected key.
The *Tempo* option sets the speed in beats-per-minute values.

**Library.** In the library the user can see and play the available recorded sessions. Each stored recording is a MIDI type 1 file that can easily be exported to the computer using iTunes File Sharing feature. These files can then be opened in any external sequencer or notation program. The file sharing also allow to import files into the GdB app, so the user can easily manage his own session files in the computer and select the ones to send to the mobile device to, for example, go for a given jam session.

# 3 Technical Description

## 3.1 Metronome

The underlying beat source is a metronome running at three times the bpm setting. This accounts for the typical triplet subdivision of the beat in traditional jazz and blues styles. The clock is then divided in two different pipelines, corresponding to two different timelines, synchronized but slightly "out of phase". This is due to a very common practice in jazz playing, which is the anticipation of the first beat. Very often, the musician will anticipate the next first beat while still in the last beat of the current measure, usually by playing a triplet before the beat. This anticipation is not only rhythmical but also harmonic, if the harmony in the next beat changes.

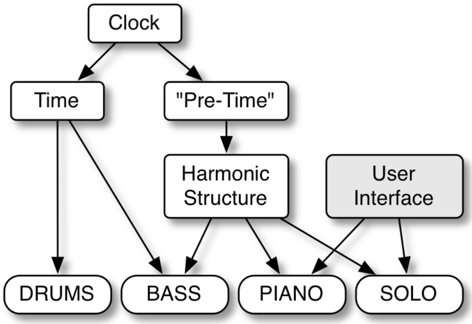


**Fig. 2.** Control diagram.

The anticipated timeline, the *pre-time*, is used to trigger the harmonic sequence, so if the user plays an anticipated chord up to an eight before the end of the measure, the harmony will correspond to the one in the next measure. The second timeline - the real or *current time*, is the perceived beat and is used to drive the bass and drums algorithms. Technically, this second timeline is a delayed version of the first one. The bpm value can be set in the setup page of the app.

### 3.2 Harmonic Structure

Each song, called *style* in the app, has a different harmonic structure, drawn from well-known traditional jazz/blues standard chord progressions. The internal sequencer reads the harmonic contents for each measure in real time, triggered by the *pre-time* clock, sending them to the pitched instruments (bass, solo and keyboard). Each instrument then uses the harmonic content information according to their specific algorithm.

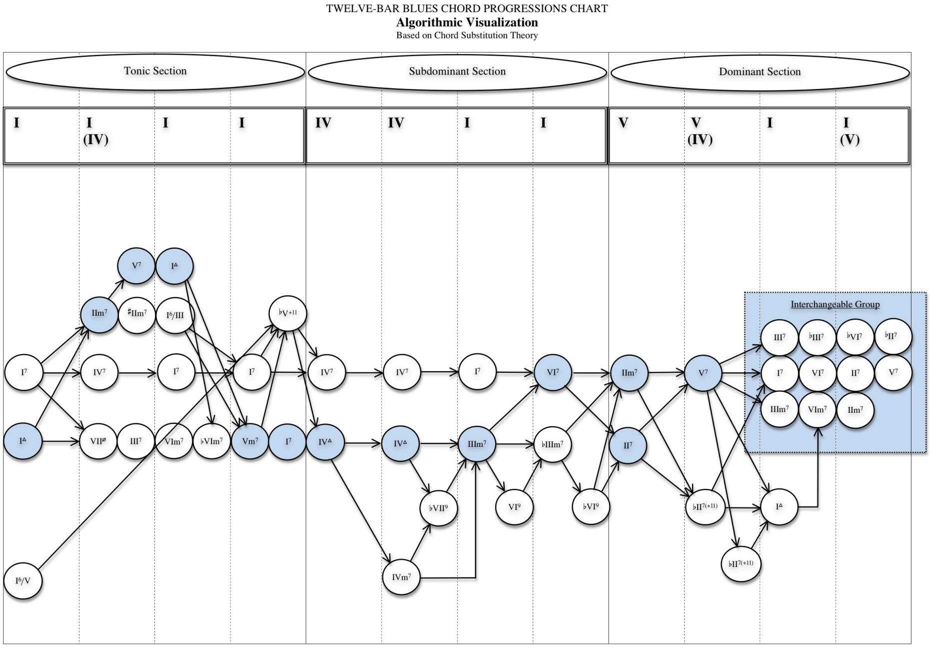


**Fig. 3.** Interchange chord substitution chart.

The study of the co-relation between common harmonic base progressions in several well-known Blues structures and common interchange and chord substitution procedures was the base of the harmonic structure selections. The chart presented in Fig. 3 was built, developed from jazz theories based on Levine [5], Nettles [6], Steedman [7] & [8], Pease [9] & [10], and Felts [11]. This chart will also be used in the development of an intelligent algorithm for harmonic variation in a future version of the app.

Each *style* defines not only a generic chord progression, but also the *voicing* for each instrument. A *voicing* is the way a given chord is played. In typical jazz and blues performance, the player has complete freedom over the combination of notes he uses to play a chord, as well as their distribution along the instrument's range, using inversions, tensions, and extensions, as long as he maintains a certain coherence with the base chord and/or chord progression. In the GdB app, this notion applies mainly to the keyboard instrument. As for the bass and solo instruments, the *voicing* defined for each song sets a scale that can be played with the corresponding harmony.

### 3.3 Solo and Keyboard Harmonic Mappings

Each one of the four instruments in GdB has different algorithms, whether relating to the interaction or to the generation method. The pitched instruments use the harmonic contents differently.

**Solo instrument.** As said before, the solo instrument's voicing data is used to define a scale. Each time a new chord arrives, this scale is mapped dynamically to the horizontal axis of the instrument's corresponding area of the screen.

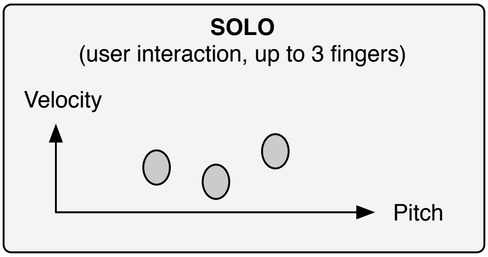


**Fig. 4.** The solo instrument interface mapping.

Each finger touch in the solo area will produce one single note, corresponding to a given note of the scale associated with the current harmony, that will be sustained while the finger is pressed. Dragging the finger will skim through the scale notes. The vertical position of the finger will determine the attack velocity of the note.

Also, using the multi-touch capabilities of the iOS devices, this area allows for polyphonic events as the user can play with up to three simultaneous fingers. This extra feature is especially interesting for playing short brass section-like riffs in the middle of a solo, or while comping for external soloists.

**Keyboard instrument.** The keyboard instrument (the lower half of the screen) is built to play block chords. Each finger touch in this area plays a two or three note chord, depending on the position on the vertical axis. By using two fingers, the user can play two chords, like a keyboard player would do, simulating piano playing in a familiar and intuitive way. The combination of the two fingers with two or three notes in each chord allows for the creation of two to six note chords.

The horizontal position in this area also determines the pitch, going from low pitches on the left to high pitches on the right, just like on a piano or organ keyboard, and determines also the chord inversion, so that moving to close positions will have a musically interesting melodic feeling to the chord successions.

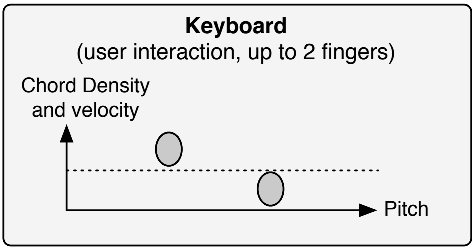


**Fig. 5.** Keyboard instrument interface mapping

### 3.4 Bass and Drums

The bass and the drums are generated automatically, according to the harmonic and metrical situation on the twelve bar blues. Once again this is grounded on that model presented in Figure 3**.**
The probabilities change dynamically with the user activity: more events being triggered by the user results in more interaction generated by bass and drums.

**Bass.** Every time a new chord arrives from the harmonic structure, the chord notes are spread throughout a range of almost three octaves, corresponding loosely to the useful range of a normal bass. These notes form an indexed list of useful notes, which can be played by the bass generator.
The note played in the beat following the chord change will be the chord root note. In the beats where there are no chord changes, the bass will play random notes from the chord.
The note events are triggered according to a probability table, setting each beat of the measure separately. The probabilities change dynamically again based on user activity.

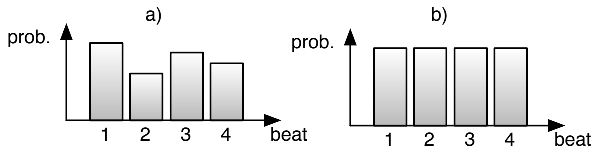


**Figure 6.** Bass probability table state for each beat at a) minimum user activity; b) maximum user activity

**Drums.** The drums part is generated automatically by combining a small number of elementary rhythmic patterns, typical for the Blues and Jazz styles. The elementary patterns are played back one at a time but their order is stochastically generated. Each time a pattern finishes playback a new one is selected. The patterns are represented as a binary sequence of triggering or no triggering sound events (see Figure 7). In order to be able to control the density of the resulted rhythm, the elementary patterns are sorted in advance, according to the number of the events each one contains. The probability of a pattern to be selected for playback depends on its position in the ordered list of elementary patterns and on the desired density for the resulted rhythm. The density can vary continuously during performance; however, in practice, only the value at the moment a new pattern is chosen is affecting the resulted rhythm.

Two separate rhythmic patterns are generated, one for the *ride* section, which includes the hi-hat cymbal, the ride cymbal and the crash cymbal, and one for the *snare* section, which includes the snare drum and the kick drum. The elementary patterns have a length either of a whole bar, for the *hi-hat* section, or equal to the beat duration, i.e. a quarter note, for the *snare* section. Since the *snare* patterns are relatively short, it is not allowed for a pattern to be selected for playback twice in a row. The *snare* patterns must always alternate.

In the *snare* section, only one sound is triggered at a time. The triggered sound is decided stochastically according to relative probabilities that are predefined for the various metrical positions. The sound triggered in most metrical positions is the snare drum sound. Nevertheless, for specific metrical positions, there is a finite probability which ranges between 30 to 80% of replacing the snare drum sound with a kick drum sound. In the *ride* section, the ride cymbal and the crash cymbal are triggered according to a similar algorithm to that for the *snare* section, with the ride cymbal being more often triggered and the crash cymbal being triggered only in specific positions in the 12 bar structure. A hi-hat cymbal sound is triggered according to a fixed pattern.

The amplitudes of the triggered sounds are randomly generated according to predefined MIDI velocity ranges. A different velocity range corresponds to each sound at each metrical position.

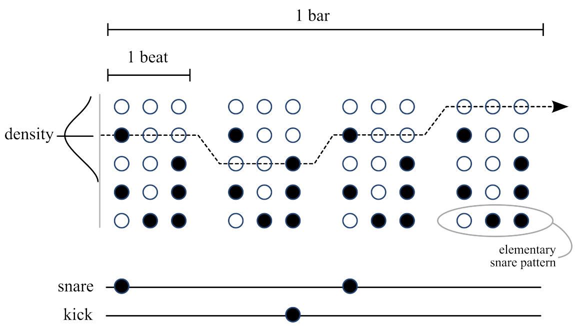


**Figure 7.** Example of how the *snare* section rhythmic patterns are generated. On every beat a mew elementary pattern is selected for playback according to the density control. Some of the snare drum sounds get replaced by kick drum sounds.

## 4. Conclusion and Future Developments

Having in mind the use of knowledge from automatic music generation and machine musicianship techniques in the development of an interactive application for music playing led to the development of an iOS application, using the multitouch capabilities to provide a very simple and intuitive, yet powerful, interface.

This interface has to accomplish three fundamental conditions: It must be efficient, user-friendly, and make people want to use it. Dix et al. [12] resumes it in three ‘use’ words: **Useful**, **Usable**, and **Used**. Several people tested the application throughout two months, musicians and non-musicians, in order to understand which ‘use’ words had to be improved. We notice that the application is easily usable, but the ‘efficiency’ and the ‘want to use it’ aspects could be improved.

Future developments will include the development of an algorithm to produce harmonic variation, according on the afore mentioned theories of chord substitution rules, the improvement of the walking-bass algorithm and the improvement of the piano voicing mapping algorithm. The possibility of synchronizing several devices wirelessly in order to allow for group playing will also be addressed in a future version.

## 5. Acknowledgments

This research was done as part of the project "Kinetic controller, driven, adaptive and dynamic music composition systems" funded by the ERDF through the Program COMPETE, by the Portuguese Foundation for Science and Technology (FCT), Project ref. FCOMP-01-0124-FEDER-011414, UTAustin/CD/0052/2008, and partly supported by the European Commission, FP7 (Seventh Framework Programme), ICT-2011.1.5 Networked Media and Search Systems, grant agreement No 287711 (MIReS).

## 6. References

1. http://thumbjam.com (accessed on June 4th 2011)
2. Biles, J. (1994) “Genjam: A genetic algorithm for generating jazz solos”. In *Proceedings of the 1997 ICMC*.
3. Thom, B. (2000) “BoB: an Interactive Improvisational Music Companion” in *Fourth International Conference on Autonomous Agents (Agents-2000)* Barcelona, Spain.
4. Brandon, John (2004) “How the iPhone works” http://www.computerworld.com/s/article/9138644/How_the_iPhone_works (accessed on June 10th 2011)
5. Levine, Mark (1989) *The Jazz Piano Book,* Petaluma: Sher Music
6. Nettles, Barrie & Graf, Richard (2002) *The Chord Scale Theory and Jazz Harmony.* Rottenburg: Advance Music
7. Steedman, Mark (1984) “A Generative Grammar for Jazz Chord Sequences, *Music Perception* 2 (1), pp. 52-77
8. Steedman, Mark (1996) “The Blues and the Abstract Truth: Music and Mental Models” in A. Garnham & J. Oakhill (eds.), *Mental Models In Cognitive Science*. Mahwah, NJ: Erlbaum 1996, 305-318
9. Pease, Ted & Pullig, Ken (2001) Modern Jazz Voicings: Arranging for Small and Medium Ensembles Boston: Berklee Press
10. Pease, Ted (2003) *Jazz Composition: Theory and Practice.* Boston: Berklee Press
11. Felts, Randy (2002) *Reharmonization Techniques.* Boston: Berklee Press
12. Dix, Alan et al. (2004) *Human-Computer Interaction* (3rd. ed). Essex: Pearson and Prentice Hall, p. 5