Harmony Generation Driven by a Perceptually Motivated Tonal Interval Space

GILBERTO BERNARDES and DIOGO COCHARRO, Inesc Tec
CARLOS GUEDES, New York University Abu Dhabi, Inesc Tec
MATTHEW E. P. DAVIES, Inesc Tec

We present D'accord, a generative music system for creating harmonically compatible accompaniments of symbolic and musical audio inputs with any number of voices, instrumentation, and complexity. The main novelty of our approach centers on offering multiple ranked solutions between a database of pitch configurations and a given musical input based on tonal pitch relatedness and consonance indicators computed in a perceptually motivated Tonal Interval Space. Furthermore, we detail a method to estimate the key of symbolic and musical audio inputs based on attributes of the space, which underpins the generation of key-related pitch configurations. The system is controlled via an adaptive interface implemented for Ableton Live, MAX, and Pure Data, which facilitates music creation for users regardless of music expertise and simultaneously serves as a performance, entertainment, and learning tool. We perform a threefold evaluation of D'accord, which assesses the level of accuracy of our key-finding algorithm, the user enjoyment of generated harmonic accompaniments, and the usability and learnability of the system.

Categories and Subject Descriptors: H.5.5 [**Sound and Music Computing**]: Methodologies and Techniques, Modeling, and Systems

General Terms: Design, Algorithms

Additional Key Words and Phrases: Generative music, harmony, tonal pitch space

ACM Reference Format:
Gilberto Bernardes, Diogo Cocharro, Carlos Guedes, and Matthew E. P. Davies. 2016. Harmony generation driven by a perceptually motivated tonal interval space. Comput. Entertain. 14, 2, Article 6 (December 2016), 21 pages.
DOI: http://dx.doi.org/10.1145/2991145

1. INTRODUCTION

In today's creative economy, the role of media users is increasingly shifting from passive consumers to active consumer-producers [Bruns 2007]. This paradigm shift has been encouraged, among many factors, by the increased access to assistive technologies, which make it possible for people with little or no expertise in content creation to

Project "TEC4Growth - Pervasive Intelligence, Enhancers and Proofs of Concept with Industrial Impact/NORTE-01-0145-FEDER-000020" is financed by the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF).
This research is also supported by the Portuguese Foundation for Science and Technology under the post-doctoral grants SFRH/BPD/109457/2015 and SFRH/BPD/88722/2012.
Author's addresses: G. Bernardes, D. Cocharro, and M. E. P. Davies, Instituto de Engenharia de Sistemas e Computadores - Tecnologia e Ciência, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal; emails: {gba, diogo.m.cocharro, matthew.davies}@inesctec.pt; C. Guedes, New York University Abu Dhabi, PO Box 129188, Saadiyat Island, Abu Dhabi, United Arab Emirates; email: carlos.guedes@nyu.edu.

DOI: http://dx.doi.org/10.1145/2991145

engage in the production and dissemination of media assets. In this article, we seek to foster user-content creation by exploring assistive technology for music creation, in particular towards the automatic generation of tonal harmony.

Historically, tonal harmony has been subject to rigorous formalizations specifically adapted to two problems that are central to our research: the automatic generation of chord progressions and the automatic harmonization of a given melody. To this end, several approaches have been successfully applied, which according to Wiggins [1999], can be split into: (i) grammar-based systems [Steedman 1999], (ii) knowledge-based systems [Pachet 1994], (iii) genetic algorithms and genetic programming [Phon-Amnuaisuk et al. 1999], (iv) constraint satisfaction systems [Pachet and Roy 1998] and (v) neural networks [Gang et al. 1997]. To which we extend Wiggins' taxonomy with a new category: statistical learning, which includes systems that generate harmony based on representations learned from musical examples [Manaris et al. 2013; Eigenfeldt and Pasquier 2010]. Despite the wide range of proposed solutions, we argue that the problem of harmony generation has been restricted to a subset of the task, which is the four-voice harmonization of melodies encoded as symbolic music representations, and current solutions are too restrictive in the sense they offer users very little control over the generation process and a narrow scope for creative endeavor and experimentation.

In this article, our aim is towards an intelligent system that encompasses formal strategies for harmony generation within the tonal Western music syntax. We strive for a solution that encourages the rapid exploration of different musical results facilitated by the ability to robustly analyze, interpret, and generate music. By doing so, we exclude the need to manually select an appropriate set of chords to accompany a given musical input. Additionally, our approach for the automatic generation of musical harmony differs from, and extends previous research by: (i) targeting both symbolic representations and musical audio inputs with any number of voices, instrumentation, and complexity, (ii) offering more than one solution at a given time to extend the possibilities for live performance and creative experimentation, and (iii) presenting an adaptive user interface, which serves the threefold pedagogic, performative, and entertainment purposes.

A distinctive feature of our approach is the use of a perceptually motivated Tonal Interval Space [Bernardes et al. 2016], which is a geometric model of tonal pitch that follows research grounded in both music theory [Lewin 1987; Cohn 1997], cognitive psychology [Longuet-Higgins 1962; Shepard 1982; Krumhansl 1990] and more directly, the computational models by Chew [2000] and Harte et al. [2006]. In this article, the Tonal Interval Space acts as a framework in which we design algorithms for music analysis and generation. Specifically, we compute indicators of tonal pitch relatedness and consonance, as well as estimate the key of a melodic and/or harmonic musical structure following previous studies on this topic by Krumhansl and Schmuckler [1986], Temperley [1999], and Chew [2000].

Our generative model has been implemented as a piece of software for Ableton Live,[1] MAX,[2] and Pure Data.[3] The resulting application, D'accord, allows users to interactively generate m-note harmonically-compatible accompaniments in real-time based on three attributes computed in our Tonal Interval Space model: perceptual relatedness, level of consonance, and key. At each beat, the system suggests several key-related pitch configurations,[4] as well their compatibility to a beat input

[1]Ableton Live, https://www.ableton.com/en/live/, last access on 20 April 2015.
[2]MAX, https://cycling74.com/products/max/, last access on 20 April 2015.
[3]Pure Data, https://puredata.info/, last access on 20 April 2015.
[4]In this article, we adopt the term "pitch configuration" to denote (vertical) pitch structures with a variable number of notes.

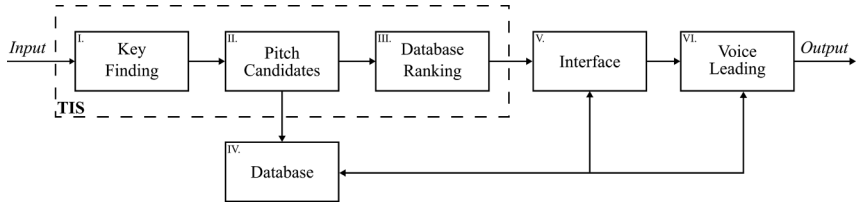

Fig. 1. Architecture of D'accord. The system modules are organized horizontally from left to right according to the information flow along with their input and output data.

by an adaptive interface designed for both novice and expert users alike, with the long-term goal of boosting the user's creative abilities through musical metacreation. The user is then responsible for selecting and triggering pitch configurations via one of several interaction modes and controllers (e.g., a computer keyboard, mouse, or MIDI controller). Prior to playback, the selected pitch configurations are further processed by an algorithm that aims to improve voice leading.

The remainder of this article is structured as follows: In Section 2, we present the architecture of D'accord, including the basic functioning of its component modules. In Section 3, we summarize the computation of the Tonal Interval Space and demonstrate its ability to measure the perceptual relatedness and consonance level of tonal pitch, as well as the possibility to estimate the key of a musical input. In Section 4, we describe the creation of a database of pitch configurations used during generation which is followed in Section 5 by an algorithm for ranking the database configurations. In Section 6, we present the user interface of D'accord and the interaction design behind the application. In Section 7, we detail an algorithm for improving the voice leading of selected database configurations. In Section 8, we present both objective and subjective evaluation of our system and finally, in Section 9, we draw our conclusions and outline areas for future work.

2. SYSTEM OVERVIEW AND DESIGN

Figure 1 shows the architecture of D'accord as well as the information flow between its six fundamental modules. In order to initialize the system, the user must first specify a MIDI or audio file as an input target. The first module of the system is responsible for splitting the input stream into beats, which provides the temporal segmentation for estimating the underlying key of the input. Based on the outputted key and whenever the key estimate is updated, the second module of the system generates a diatonic set of seven candidate m-note pitch configurations on the fly, which are available to the end-user of the system.

The third module of the system is then responsible for ranking the candidate pitch configurations by assessing how compatible they are with a target beat in terms of perceptual relatedness and consonance. Up to this stage, all processing depends on the 12-dimensional (12-D) Tonal Interval Space (see Section 3), which forms the backend of the three first modules of D'accord.

The fourth module of the system is a database that stores the information generated by the three first modules, in particular the collection of candidate pitch configurations in several representations. Its flexible structure allows the dynamic allocation of information on a beat-by-beat basis, which is accessed at runtime by the remaining modules of the system.

The fifth module is responsible for providing the end-user the information generated in the first three modules in an intuitive manner. The interface uses color coding to indicate the compatibility level between the database candidates and the target beats.

At each beat, more than one good solution is provided to the user, who can then select and trigger events (i.e., pitch configurations) using an external controller.

After a pitch configuration has been triggered, the system retrieves its pitch class components from the database and sends it to the last module of the system, which is then responsible for organizing the voice leading of the selected configurations. In this module, the pitch class set of the selected configuration is unfolded into several pitch spacings and inversions over the user-defined pitch range. Then, a cost function ranks all generated solutions and outputs the candidate with the minimal cost for playback.

3. TONAL INTERVAL SPACE

The 12-dimensional Tonal Interval Space [Bernardes et al. 2016] is a multi-level tonal pitch framework, which follows the line of perceptually motivated spaces in the context of the *Tonnetz*, such as Chew's [2000] Spiral Array and Harte et al.'s [2006] Tonal Centroid Space. The most salient tonal pitch levels, such as pitch classes, intervals, chords and keys are represented in the space by Tonal Interval Vectors (TIVs), $T(k)$, which result from aggregating their component pitch class, then converting them to a 12-D chroma vector,[5] $c(n)$, and finally computing their (L_1 normalized) Discrete Fourier Transform (DFT), such that:

$$T(k) = \frac{w(k)}{\bar{c}} \sum_{n=0}^{N-1} c(n) e^{-\frac{j2\pi kn}{N}}, k \in \mathbb{Z}, \tag{1}$$

where n is the chroma vector pitch class index up to $N = 12$, k corresponds to the particular interval in question, and $\bar{c}$ is the DC component of $T(0) = \sum_{n=0}^{N-1} c(n)$, which is responsible for normalizing the space. The TIV only includes the coefficients $T(k)$ for $1 \leq k \leq 6$, thereby excluding the DC component and symmetrical intervals resulting from the DFT computation.

To enhance tonal pitch relatedness and define a consonance measure in the space, we adjust the location of pitch classes in the Tonal Interval Space by weighting the contribution of each complementary interval, $T(k)$. To this end, we use empirical dissonance ratings of dyads derived from three studies [Malmberg 1918; Kameoka and Kuriyagawa 1969; Hutchinson and Knopoff 1978], whose composite index is detailed in Huron [1994]. Since the resulting weights, $w(k) = \{2, 11, 17, 16, 19, 7\}$, are known *a priori*, we incorporate them into the direct calculation of the $T(k)$ in Equation (1).

Following Harte et al. [2006], we visualize the 12-D space as 6 circles, each representing one complex DFT coefficient (see Figure 2), i.e., circle 1 has the real part of $T(1)$ on the x axis and the imaginary part of $T(1)$ on the y axis and so on. The weights assigned to the DFT coefficients correspond to the circles' radii, represented in Figure 2 as radius vectors.

The Tonal Interval Space is used in this article as a framework on which we design algorithms for computing indicators of tonal pitch relatedness, consonance, and for estimating the key of a musical passage (detailed in Sections 3.2, 3.3, and 3.4, respectively). Together these three components drive the selection of chord candidates in D'accord. However, before providing details about these, we detail the representation of symbolic and audio inputs in the space.

3.1. Pitch Representations

D'accord can process symbolic representations and audio signals, the strategies for each we address separately in the two following sections. For both input types, the musical

[5]Chroma vectors represent the distribution of energy across the chromatic pitch classes in a musical octave.

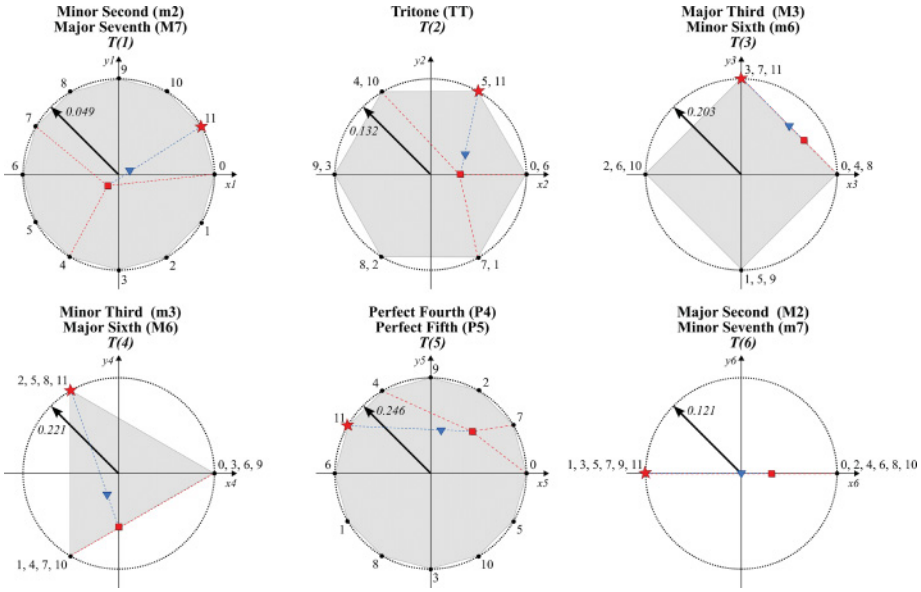

Fig. 2. Visualization of the Tonal Interval Space as six circles organized according to complementary intervals. Shaded gray areas denote the regions which TIVs can occupy for each circle. Three plotted TIVs correspond to the position of the C major chord (square), the pitch B (star), and the combination of both configurations (triangle)—which correspond to the pitch classes set {0, 4, 7}, {11}, and {0, 4, 7, 11}, respectively. The pitch configuration resulting from the combination lies on a segment line whose boundaries correspond to the location of first two TIVs. The radii correspond to the weights of each complementary interval, $T(k)$, which for visualization purposes are all represented by an identical size.

content can be either monophonic or polyphonic, for which D'accord first derives an appropriate chroma-based representation. Then it generates representations for the key and its seven related diatonic pitch configurations in the Tonal Interval Space. The rate at which D'accord updates the databases with new ranked configurations is linked to the tempo of the input files. Therefore, for both symbolic and audio inputs, we detail the strategies applied in the segmentation of the input into beats, as well as their representation as chroma vectors—a required pre-processing step for TIV computation.

3.1.1. Symbolic Input. The database of pitch configurations and input targets driven from symbolic music inputs are represented in the Tonal Interval Space by binary activations in a chroma vector $c(n)$. For example, the C major chord (pitch classes 0, 4, and 7) gives chroma vector $c = \{1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0\}$. In this sense, database configurations are represented by clean and robust chroma vectors, which assume equal temperament and enharmonic equivalence and discard chord inversions, the effect of the relative amplitude between pitch classes, note durations, and doublings. To generate a beat-synchronous chroma vectors driven from symbolic target input, we set all active chroma vector bins to 1, i.e., those corresponding to pitch classes occurring across the duration of each beat. To determine the beat locations, we rely on the metadata within MIDI files.

To represent keys in the Tonal Interval Space, we defined four different strategies, each driven by a set of chroma vectors for the major and minor modes. This collection of chroma vectors was originally presented in the context of key finding algorithms, where they are commonly referred to as key profiles, a terminology we also adopt. The C major and C minor vectors for each profile, p, are shown in Table I, and the chromas

Table I. Chroma Vectors Representing the c Major and c Minor Key Profiles
With the exception of the first theoretically derived profile, which results from the binary activation of the key diatonic pitch class set, the three remaining profiles were proposed in the context of key finding algorithms [Krumhansl and Kessler 1982; Temperley 1999; Chew 2000].

		C	C#	D	D#	E	F	F#	G	G#	A	A#	B
Major	c^{diat}	1	0	1	0	1	1	0	1	0	1	0	1
	c^{k-k}	6.35	2.23	3.48	2.33	4.38	4.09	2.52	5.19	2.39	3.66	2.29	2.88
	c^{temp}	0.748	0.060	0.488	0.082	0.670	0.460	0.096	0.715	0.104	0.366	0.057	0.400
	c^{chew}	2	0	1	0	1	1	0	2	0	1	0	1
Minor	c^{diat}	1	0	1	1	0	1	0	1	1	0	0	1
	c^{k-k}	6.33	2.68	3.52	5.38	2.6	3.53	2.54	4.75	3.98	2.69	3.34	3.17
	c^{temp}	0.712	0.084	0.474	0.618	0.049	0.460	0.105	0.747	0.404	0.067	0.133	0.330
	c^{chew}	2	0	1	0	1	1	0	2	1	1	1	1

of the remaining keys are obtained by rotating the vectors by 12 semitones. To compute the key TIVs, T^p, for each profile, p, we compute the DFT of the chroma vectors using Equation (1). The resulting TIVs are used to define the location of each of the 24 major and minor keys in the Tonal Interval Space.

The first key profile, c^{diat}, activates the diatonic pitch set of a key in a chroma vector. The second profile, $c^{k\text{-}k}$, by Krumhansl and Kessler [1982], was derived from the 'probe tone' method. The third key profile, c^{temp}, by Temperley [1999], resulted from adjusting the K-K profiles using music theory principles. The last key profile, c^{chew}, by Chew [2000], is widely used in her Center of Effect Generator algorithm for estimating the key of a musical passage in the Spiral Array.

3.1.2. Audio Input. Generating chroma vectors from audio poses substantially greater challenges, and hence demands a different approach than for symbolic inputs because robust note transcription from polyphonic audio stream remains unsolved [Benetos et al. 2013]. Furthermore, given that audio chroma representations could affect the robustness of D'accord, notably the key recognition stage, we examine three different strategies for audio chroma representation. As in the symbolic input, we first report the computation of chroma vectors on a regular and short-time interval basis and then their codification on a beat-by-beat basis (beat-synchronous chromas).

The first and second strategies use the QM chroma [Harte and Sandler 2005] and NNLS chroma [Mauch and Dixon 2010] plugins within Sonic Annotator [Cannam et al. 2010] with default parameters. While the QM chroma folds all spectral information into a 12-bin (octave) representation, the NNLS chroma performs an approximate note transcription, and typically provides a sparser representation of the input signal in the chroma domain. The third strategy applied to generate audio chroma vectors uses a more naïve method, referred to as "simple" chromas, and assumes that the most relevant information in the audio spectrogram is encoded in the most prominent peaks of its FFT analysis. We adopt this because of an expected need for sparsity in the chroma vectors to most closely match the symbolic input. Therefore, we encode the eight most prominent peaks resulting from a sliding window FFT analysis (window size set to 4096 samples and hop size of 2048 samples at a 44.1 kHz sampling rate). To extract the eight most prominent peaks of the spectra we use Pure Data's external sigmund~. To minimize the impact of transients, we exclude frequencies above 5 kHz.

To generate beat-synchronous chroma vectors from an audio stream, we compute beat locations using the QM-VAMP bar and beat tracker [Davies et al. 2009] within Sonic Annotator [Cannam et al. 2010] and then calculate the median value per chroma bin for all frames within each beat.

3.2. Measuring Pitch Relatedness

In earlier work [Bernardes et al. 2016], we showed that the relative location among multi-level pitch configurations represented in the Tonal Interval Space provides a strong indicator of the perceptual relatedness of tonal pitches. In the Tonal Interval Space, tonal pitch relatedness, D, can be computed as the Euclidean distance between Tonal Interval Vectors, such that:

$$D(T_i, T_j) = \sqrt{\sum_{K=1}^{6} (|T_i(k) - T_j(k)|)^2}, \tag{2}$$

where T_i and T_j are two different TIVs.

Moreover, the relative location of pitch configurations in the Tonal Interval Space concurs with theory principles at the three main levels of tonal Western music: pitch class/intervals, chords and key [Bernardes et al. 2016].

At the pitch class level, it places intervals that play an important role in the tonal system (e.g., octaves, fifths, and thirds) at smaller distances. Pitch class distances in the Tonal Interval Space preserve the pitch organization of the *Tonnetz*, meaning that, at the chordal level, the TIS favors chord progressions with minimal displacement of moving voices (known as voice-leading parsimony) by placing chords sharing common tones near to one another. Furthermore, while preserving the common-tone logic of the *Tonnetz*, the Tonal Interval Space expands the range of musical objects beyond triads, being able to map unique chroma vectors to unique points in the space.

At the key level, the Tonal Interval Space represents our expectancy of proximity between the 24 major and minor keys by placing the dominant, subdominant and their relative minor keys at close distances. Additionally, in our topology the degree to which pitch classes and chords relate to a particular region is dictated by their proximity to the key TIVs, or, in other words, the neighborhood of all key TIVs are occupied by their diatonic pitch class sets. Further details on measuring tonal pitch relatedness can be found in Bernardes et al. [2016].

3.3. Measuring Consonance

In the visualization of the 12-D Tonal Interval Space, pitch configurations occupy a limited area in each circle (gray shaded areas shown in Figure 2), which range from the center of the circle to the circumference as a result of the normalization in Equation (1). Within the restricted space that TIVs can occupy, the location of, and distance among, pitch class TIVs not only obeys music theory principles but is also determined by empirical consonance ratings of musical dyads. Therefore, we extrapolate the consonance measure $C(T)$ of the TIV by calculating the Euclidean distance to the center of the space. We divide the result by the individual pitch class consonance C_{max} to normalize it to the range [0, 1].

$$C(T) = \frac{\sqrt{\sum_k \left[T(k)\right]^2}}{C_{max}}, \tag{3}$$

where $C_{max} = 32.86$.

To compute the consonance level of composite pitch configurations, we sum the two chroma vectors to create a new vector that reflects their combined energy, then we compute the TIV of the resulting chroma vector using Equation (1), and finally apply Equation (3) to indicate their joint consonance. The location of the resulting TIV will fall in a line segment bounded by the original two TIV locations (see Figure 2). The exact location within this line segment is given by the total energy of each pitch configuration,

i.e., the configuration with lower energy pushes the resulting TIV from its location, and the configuration with more energy attracts the resulting TIV towards its location. Given this property of the space, we can save computation time while calculating the composite consonance level of two configurations by keeping track of their energy and computing the centroid of the two original TIVs, T_i and T_j, such that:

$$C(T_i, T_j) = \frac{1}{C_{max}} . \frac{\sqrt{\sum_{k=1}^{6} (|T_i.a_i - T_j.a_j|)^2}}{a_i + a_j}, \tag{4}$$

where a_i and a_j are the total energy of chroma vectors c_i and c_j. In this article, we hardcode the energy of both target and database TIV to unity, thus discarding the effect of the relative amplitude between combined chroma vectors, because the energy of symbolic and audio signals are not akin for comparison.

3.4. Key Finding

This section details our key-finding algorithm based on the Tonal Interval Space, which continuously estimates the key of a musical input over time. To estimate the key, our algorithm relies on two attributes of the Tonal Interval Space: (i) the set of diatonic pitch classes in a given key occupying a compact neighborhood around its key TIV and (ii) the 24 major and minor key TIVs being sparsely represented in the space and within each mode, equally spaced. Based on the assumption that a key-indicating element is the use of its diatonic pitch set, we estimate a key of a musical passage in the Tonal Interval Space by finding the nearest neighbor in the high-dimensional Euclidean space of a query TIV given a database of 24 major and minor key TIVs.

As musical events are sounded in sequence, their resulting TIV locations form a geometric shape, which becomes increasingly more complex over time. Following Chew [2000], instead of using this complex shape to compute key estimates, we collapse the temporal information to a single TIV. At each event, the algorithm only compares a unique TIV to the 24 major and minor TIV keys. Moreover, to avoid sudden changes in the estimated key, which is commonly stable over large temporal spans, as well as incorporating a simple model of a listener's short-term memory, which has been shown to play an important role in tonality perception [Farbood et al. 2013], the cumulative TIV, T_n, is calculated by the weighted average of the cumulative TIV of past events, $T_{n\text{-}1}$, and the current TIV, T_n, such that:

$$T_n = \begin{cases} T_n & n = 0 \\ \alpha T_n + (1 - \alpha)\, T_{n-1} & n \geq 1 \end{cases}, \tag{5}$$

where:

$$\alpha = \max\left(\frac{1}{n+1}, 0.01\right) \tag{6}$$

To rank the collection of 24 major and minor keys, the system computes the Euclidian distance between the accumulated TIV, T_n, and all key TIVs. The best key estimate, M, is the one which minimizes:

$$M = argmin_p \sqrt{\sum_{k=1}^{6} (|T_n(k) - T^p(k)|)^2}, \tag{7}$$

where T^p is a particular set of key TIVs derived from a collection of four different key profiles proposed in related literature and presented in Section 3.1. After finding the minimal Euclidian distance of the 24 key TIVs set from the cumulative input TIV, the system reports the best key estimate by a pitch class, which ranges between 0–11 for

major keys and 12–23 for minor keys. Both sets of values start by the pitch class C and follow an ascending chromatic sequence.

4. DATABASE CREATION

D'accord's database consists of seven diatonic pitch configurations which rely on two parameters: the number of notes, m, per pitch configuration and the key estimate. Depending on user input, the number of notes, m, per pitch configuration can range from 1 to 12 (i.e., the maximum number of notes in a chroma vector representation), but, in practice, we constrain this variable to a maximum of 4-note chords, since pitch configurations with 5 or more different notes are less typical in tonal music and require enhanced treatment of voice leading. Therefore we allow m to take values from [1, 4]. When $m = 1$, the system generates an accompanying melody, and when $m > 1$, m-note chords.

Given m, the system generates a diatonic set of seven pitch configurations at every key change. To generate the seven pitch configurations, the system first creates a list with the diatonic pitch class set of the estimated key, using the major scale and the harmonic minor scale for the major and minor modes, respectively. Then, for each pitch class it vertically stacks m-number of notes in intervals of thirds using the pitch classes from the set. For example, if our estimated key is C major and $m = 2$, the resulting key set is {0, 2, 4, 5, 7, 9, 11} and the database Z = {0 4, 2 5, 4 7, 5 9, 7 11, 9 0, 11 2}.

The order in which pitch configurations are generated follows a constant pattern based on the scale degrees of the estimated key. The first generated configuration always corresponds to the first degree of the scale, the second configuration to the second degree and so on. This order keeps track of the scale degree of each configuration or, when $m > 1$, the root of the chord. Finally, a database stores the generated pitch configurations as: (i) the set of their pitch class components, (ii) their chroma vector representations, and (iii) their TIVs, which are later accessed by the remaining modules of D'accord.

5. RANKING THE PITCH CONFIGURATION CANDIDATES

D'accord's database is composed of seven pitch configurations generated based on the target key estimate. Restricting the database candidates to seven pitch configurations related to the target key immediately discards some poor results. However, even though this limited set of configurations is assumed to better accompany a target beat than non-related key configurations, not all seven pitch configurations rank equally according to a specific target beat. Therefore, we rank the compatibility between the database configurations and each target beat.

To this end, we adopt two criteria: the degree of perceptual relatedness and level of consonance between target beat and database configurations. Both indicators have were already introduced and are computed using Equations (3) and (5). Since the score aims to minimize the Euclidian distance and maximize consonance between TIVs, we combine both indicators by inverting the normalized Euclidean distance computation by subtracting the result from 1. For each database configuration, Z_i, we compute a score, R_i, which indicates their joint perpetual relatedness and consonance level with an input target beat, T_{target}, such that:

$$R_i = [1 - \hat{D}(T_{target}, Z_i)] + \hat{C}(T_{target}, Z_i). \tag{8}$$

where $\hat{D}$ corresponds to the distances D normalized to occupy the range 0 to 1 for each specific pitch configuration Z_i, and $\hat{C}$ the consonance values for the database configurations Z_i, normalized to the range 0 to 1. The resulting rankings, R_i, indicate how compatible a target beat is with each database configuration, Z_i, in terms of perceptual relatedness and consonance, e.g., the best database candidate is $argmax_i$ (R_i).

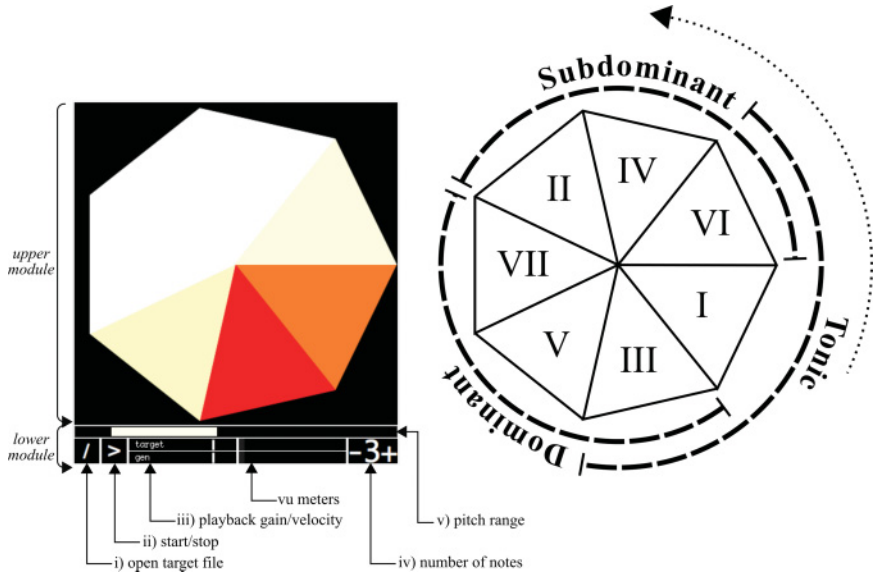

Fig. 3. D'accord interface developed for the PD programming environment (on the left) and the skeleton of the interface layers (on the right) showing the scale degree locations, the harmonic category function regions (dominant, subdominant, and dominant), and the typical motions between these categories (dotted line).

6. USER INTERFACE AND MODES OF INTERACTION

To allow users to interact with D'accord, we have created an interface for Ableton Live, MAX, and Pure Data – of which a screenshot is shown in Figure 3. Despite minor cosmetic differences, the interfaces for all environments have identical functionality.

The interface is composed of two main modules arranged vertically. The upper module contains a heptagon divided into seven equally sized triangles used for displaying the database rankings and triggering events for playback. At runtime, each triangle is dynamically mapped to one of seven database pitch configurations. The update rate of the mappings depends of the input file tempo and occurs at the onset of each beat. To expose the database configurations rankings we adopt a color code scheme ranging from continuous shades of white to yellow to red. The mapping between the color scheme and the database scores is linear. A zero score corresponds to white, a 0.5 score to yellow and a score of 1 to red. Between these values, the colors are gradually mixed, but in general, the closer to red, the better the score.

While pitch configurations are dynamically mapped to each of the interface's triangles, a fixed mapping between scale degrees and the seven triangles exists (see the rightmost image in Figure 3). The location of the seven scale degrees on the heptagon place pitch configurations sharing common-tones at close distances. Moreover, configurations sharing function categories (i.e., configurations that can substitute each other without altering the tonal function) are in adjacent triangles. For example the tonic, median, and submediant scale degrees, all related to the tonic harmonic function, are mapped to three adjoining triangles. The common-tone circular organization and harmonic function clusters places the best rated configurations at close distances and helps users anticipate typical motions between function categories which are most likely to run counterclockwise—assuming that Riemann's [1893] typical motions between categorical function applies, i.e., from the tonic to the subdominant to the dominant and back to the tonic.

Based on the interface, the user can make an informed decision about which pitch configuration to play by selecting one of the pitch configurations on the interface (i.e., one of the seven triangles in the heptagon). The decision of when to play and the duration is the responsibility of the user, who can trigger and release events by sending commands from external devices such as the mouse, keyboard or MIDI controllers. The two commands sent from the external devices to D'accord are then converted to MIDI note on and note off messages, to trigger and release note events, respectively.

The simplest mode of interaction with D'accord's interface is via the mouse. To trigger events with the mouse, the user must click on one of the triangles in the heptagon and release the mouse click to control the duration.

The second strategy is via the computer keyboard or any external MIDI controller. This mode eases the interaction between user commands and interface information by offering indirect control over the database configurations. To this end, while updating the interface's information, the system keeps track of the ranking database scores in decreasing order. Then it establishes a direct mapping between the keyboard keys 1 to 7 or the MIDI note messages 60–67 to the ranked database configurations. In this way, the best candidate solution is always mapped to the same location on the controllers. The duration of played events is controlled by a similar strategy as with the mouse interaction, i.e., by the release command of the keyboard or external MIDI controller.

In sum, D'accord's interface was designed to provide a simple, intuitive, and flexible experience for both expert and novice users. The mappings between database configurations and interface location provide three concurrent layers of information: (i) the compatibility score to a given target beat, (ii) categorical function of database configurations, and (iii) typical motion between harmonic function categories. Although, to some extent, the two latter layers may not be apparent to novice users, it can offer experts user the possibility to make more informed choices during runtime, as well as being used for the pedagogic purpose of learning tonal music structures in the context of projects such as McCarthy Music's Illuminating Piano,[6] Griffin and Jacob's [2013] adaptive digital music instruments, Dias and Guedes's [2012] Gimme da Blues, and Behringer and Elliot's [2009] and Bigo et al.'s [2012] manually driven generative music systems, which adopt the *Tonnetz* as a representation for designing tonal chord progressions.

The lower module contains the following control settings for D'accord: (i) open a target file, (ii) start/stop the playback, (iii) control the amplitude/velocity of the playback and/or generated material, (iv) define the number of notes per pitch configuration, and finally, (v) the pitch range which the generated output can occupy (defined on a range slider whose limits are the standard piano pitch range (i.e., A_1 to C_7 or the MIDI range note values 21–108)—which is quantized to multiples of 12 (i.e., musical octaves). Additionally, the lower module provides volume unit meters for both target and generated outputs.

7. VOICE-LEADING

Voice leading is an important aspect of harmony, which regulates both horizontal motion and vertical arrangement of the voices of a chord progression. Together with the rules of harmonic progression, voice leading establishes the ground rules for effective musical harmony writing. Its importance is fundamental in the training of the Western musician, backed by the claim that "good voice leading can take a simple chord sequence and transform it into a masterpiece" [Schonbrun 2011, 174].

[6]McCarthy Music's Illuminating Piano, http://www.mccarthypiano.com/software.aspx, last access on 20 April 2015.

Table II. Point Assignments for Voice Leading Rule Conditions

		Chord spacing	Melodic leap	Contrary motion	Parallel $5^{th}/8^{th}$	Hidden $5^{th}/8^{th}$
Cost	5	—	—	—	true	true
	2	>12	>8	—	—	—
	1	>3	>4	false	—	—
	0	=0	=0	true	false	false

Despite the large quantity of music theory literature documenting objective voice-leading rules, we should bear in mind that voice leading depends on several aspects including musical style, composer idiosyncrasies, orchestration, and density. Commonly, rules for voice leading found in music theory textbooks are drawn from, and meant for, canonic vocal four-part music and may not apply to music outside this category.

Here, we rely on a concise set of voice leading rules framed for vocal four-part music, which we adopt with relative freedom. It is outside of the scope of this article to present a comprehensive model for the automatic computation of voice leading, in the sense it implies many parameters that are either subjective or poorly addressed in the literature. Additionally, since D'accord is agnostic with respect to music style—the only constraint is the use of tonal harmony—we deliberately reduce the number of modeled rules to a smaller set, which conform to more general music contexts.

From the collection of voice-leading rules in Huron [2001], we select the following set based on a criterion of their adaptability to general music contexts other than vocal music:

(1) Chord Spacing. Intervals between voices should not exceed one octave, with the exception of the interval formed by the two lowest notes, in which no restriction is applied;
(2) Avoid Melodic Leaps. Conjoint chords should minimize the step distance in each voice, preferably maintaining common tones in the same voice or moving to the nearest possible pitch;
(3) Avoid Parallel Octaves and Fifths. Intervals of the octave and fifth should not be consecutive between the same voices;
(4) Avoid Hidden Octaves and Fifths. Intervals of the octave and fifth should not be consecutive between different voices;
(5) Outer Voices (Contrapunctual) Motion. Contrary motion between outer voices is encouraged.

The algorithmic strategy developed to improve voice leading is only applied to 3- or 4-note database configurations. It finds the best pitch configuration for playback from a large set of candidate configurations generated from all possible vertical spacings and inversions within the user-defined range of the selected pitch configurations. For example, if our selected chord is a 3-note chord, our database will include the chord in its fundamental state as well as its two inversions, i.e., with the third and fifth on the bass. Additionally, for each of the three sets, all possible chord spacings within the user-defined pitch range are instantiated.

All pitch configurations in the population are evaluated in relation to the previously played configuration and are ranked using a set of points shown in Table II if they meet particular conditions. Each pitch configuration is evaluated in terms of the five rule conditions, and for each condition the algorithm assigns a cost. Then, an overall cost per configuration is computed by summing the five rule conditions cost. Finally, the algorithm outputs the configurations with the minimal cost, which corresponds to the best voice leading.

Some additional voice leading properties can be manually specified by the user, such as the textural density or number of voices per pitch configuration as well as the

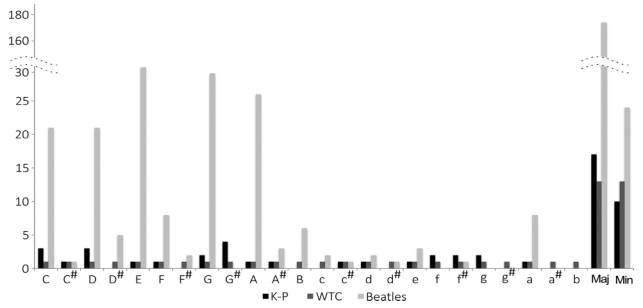

Fig. 4. Key distribution in the three audio and symbolic datasets (K-P, WTC, Beatles) used for evaluating our key finding algorithm (please note that for enhanced visualization the y axis scale is broken).

range the pitch configurations can occupy. The Tonal Interval Space and the interface design explicitly offer control over voice leading by placing configurations that share common tones at close distances. Retaining tones from one configuration to the next, commonly referred to as parsimony, improves the voice-leading algorithm to minimize the database costs.

8. EVALUATION

To evaluate D'accord we address three aspects of the system. First, we demonstrate the efficacy of the key-induction algorithm, which underpins the generation of an optimal set of candidate pitch configurations via an objective comparison against ground truth annotations and previous studies. Then we undertake two subjective experiments: (1) a listening test to assess the database rankings that support the user selection process and (2) a pilot user interaction experiment which aims to test the usability and learnability of the software.

8.1. Objective Analysis

8.1.1. Test Database. Our key finding algorithm was evaluated on three annotated datasets. Two datasets include both symbolic and audio classical music examples and correspond to Bach's 24 fugue subjects from Book I of the Well Tempered Clavier (WTC), and the Kostka-Payne (K-P) dataset [Temperley 1999]. The third is a large collection of pop/rock Beatles' songs assembled by Harte [2010]. Since we aim for a system capable of generating compelling accompaniments given a wide scope of music, we selected three datasets that cover distinct styles of music including monophonic/polyphonic and monotimbral/multitimbral examples. Furthermore, a comparison with existing key-finding methods can be established for the Beatles dataset [Gómez 2006; Mauch and Dixon 2010; Lindenbaum et al. 2015].

We excluded 19 musical examples from the K-P dataset, i.e., those in a mode other than the major or minor, and those containing a modulation in key. The entire Beatles' dataset was used. This gives a total of 230 musical examples: 27 from the K-P corpus, 24 Bach's fugue subjects, and 179 Beatles songs. The sources of the first two datasets provide musical examples in symbolic representation (MIDI file format), which were used to expand the datasets with audio examples by synthesizing them with piano samples using Logic.[7] We analyzed 51 musical examples in symbolic representation and 230 audio examples. All audio files were uncompressed single channel (mono) with 16-bit resolution and a sampling rate of 44.1 kHz.

Figure 4 shows the total distribution of musical examples per key, as well as the overall totals per mode and input type. In the Beatles dataset, there is a bias towards

[7]Logic, https://www.apple.com/logic-pro/, last access on 25 April 2015.

Table III. Point Assignments for Key Estimates Used to Evaluate Key-Finding Algorithms at the 2005 MIREX Competition

	Correct	Perfect 5th	Relative	Parallel	Others
Points	1	0.5	0.3	0.2	0

Table IV. Performance of Key Finding Algorithm for Symbolic Input

		c_{flat}	$c_{k\text{-}k}$	c_{temp}	c_{chew}
K-P	score (%)	57 (49)	80 (72)	**86 (79)**	78 (70)
	aBeats	6.7	5.2	**2.5**	3.7
	cRatio	23/27	**27/27**	26/27	25/27
WTC	score (%)	51 (46)	66 (48)	**86 (82)**	81 (77)
	aBeats	4.0	7.7	**2.1**	2.2
	cRatio	22/24	23/24	**24/24**	**24/24**

Score: Percentage of correct key estimates using the points assignments shown in Table III and (within brackets) the exact number of correct estimates. *aBeats*: average number of beats needed to reach a correct key estimate. *cRatio*: ratio between the number of correct key estimates to the total number of examples.

major over minor examples. However, in order to enable a direct comparison with previous research, we retained all examples.

8.1.2. Performance. For comparison with previous studies, we use a threefold evaluation method. First, we use the scoring strategy from the 2005 MIREX key finding competition as well as in Gómez [2006] and Izmirli [2006], which measures the percentage of correctly or closely related key estimates per beat segment using the score points shown in Table III. This evaluation method assumes that some incorrect key estimates are worse than others, because if the keys are related in some harmonic manner this may still be informative to a musician. Second, we compute the percentage of correctly estimated beat segment keys. The third evaluation criterion is the ratio of correctly estimated keys per musical example to the number of total examples.

Tables IV and V show the performance of our key finding method for symbolic and audio music input, respectively. The first row presents two values that correspond to the percentage of correct key estimates using the point assignments shown in Table III, and the second value (in brackets) gives the percentage of correct key estimates for the entire datasets. The second row presents a score of the model efficiency, which is computed as the average number of beat segments needed to reach the first correct key estimate. The last row presents the ratio of correctly estimated keys per musical example to the number of total examples.

The scores shown in Tables IV and V express the percentage of correct key estimates per beat segment for each dataset. The beat segmentation and update rate of key estimates aims to place D'accord as close as possible to a real-time performance. For the symbolic datasets, the Temperley key profile results in the best performance for our key finding method. The scores resulting from applying each of the profiles are shown in Table IV.

On the K-P dataset 79% of key beat segments were correctly estimated, and this score rose to 86% using the MIREX point assignments. On average, the algorithm needs 2.5 beat segments to output a correct key estimate, and overall it was successful at reaching the correct key in 96% examples of the dataset (26 out of 27). On the WTC dataset, our method performs equally well, with some minor differences: the score resulting from applying the MIREX point assignments is slightly higher (82%), and the convergence to correct key estimate was faster (2.1 beats on average). All examples in the WTC dataset converged to the correct key.

Table V. Performance of the Key Finding Algorithm for Audio Input

		c_{flat}			$c_{k\text{-}k}$		
		QM chroma	NNLS	Simple	QM chroma	NNLS	Simple
K-P	score (%)	27 (25)	72 (67)	62 (55)	33 (28)	81 (74)	76 (68)
	aBeats	13.1	7.1	12.0	4.9	4.1	11.3
	cRatio	10/27	24/27	22/27	16/27	25/27	26/27
WTC	score (%)	40 (31)	56 (50)	59 (55)	37 (28)	51 (29)	51 (32)
	aBeats	12.2	6.4	14.1	3.1	18.2	15.4
	cRatio	11/24	22/24	20/24	18/24	24/24	23/24
Beatles	score (%)	5.4 (2.1)	58 (42)	56 (36)	10 (7.1)	**76 (68)**	72 (62)
	aBeats	32.2	25.2	26.7	14.9	**14.8**	29.1
	cRatio	23/179	118/179	138/179	48/179	161/179	161/179
		c_{temp}			c_{chew}		
		QM chroma	NNLS	Simple	QM chroma	NNLS	Simple
K-P	score (%)	35 (31)	**85 (78)**	82 (75)	35 (31)	74 (67)	72 (67)
	aBeats	2.9	**2.2**	9.7	3.2	3.1	12.1
	cRatio	10/27	26/27	27/27	10/27	25/27	**27/27**
WTC	score (%)	43 (37)	**74 (63)**	70 (62)	45 (40)	48 (43)	68 (60)
	aBeats	3.4	3.9	3.9	1.5	2.7	4.4
	cRatio	16/24	24/24	24/24	12/24	23/24	24/24
Beatles	score (%)	9.0 (6.2)	74 (65)	72 (62)	12 (7.8)	51 (46)	65 (62)
	aBeats	32.4	15.2	22.9	17.1	16.7	27.4
	cRatio	33/179	156/179	**165/179**	26/179	133/179	160/179

A direct comparison with existing studies for key finding from symbolic inputs cannot be established because of small differences between the segmentation used in these studies and our model. While most systems update their estimates on a note-by-note basis our system processes information on a beat-by-beat basis.

In Table V, we compare the performance of our model using different input chroma representations and key profiles for three musical audio datasets. The best performance is achieved using the NNLS chroma for the three datasets. The K-K profiles performed better for the K-P and WTC datasets and the Temperley key profile on the Beatles dataset. The results achieved for these datasets and using the aforementioned chromas and profiles are: 78% (K-P), 63% (WTC), and 68% (Beatles). Applying the MIREX scoring system we obtain: 85%, (K-P), 74% (WTC), and 76% (Beatles).

Several studies have used the Beatles dataset to test their key finding methods. From these studies a direct comparison can be established with Gómez [2006], Mauch and Dixon [2010], and Lindenbaum et al. [2015], which also use the complete Beatles' dataset. They report 70.4%, 63%, and 66.5% of correct key estimates, respectively. Gómez and Lindenbaum et al. also report their results using the MIREX evaluation system obtaining scores of 76.2% and 75.6%.[8] A few songs in the dataset are in modes other than major or minor and therefore not modeled by most key finding methods including ours. However, we retain them to maintain a fair comparison with existing studies.[9]

The evaluation results in Table V also show that the chroma representation has a noticeable impact on the performance of the system with audio inputs, and considerably deteriorates in the presence of inharmonic spectra as in the Beatles' dataset examples resulting from the presence of drums (this being more evident when using the QM

[8]Please note that we did not run these algorithms ourselves. Instead, we use the results reported in their publications.

[9]A comparison to other key finding methods [Rocher et al. 2010; Papadopoulos and Tzanetakis 2012; Noland and Sandler 2006] using the Beatles dataset for evaluation was not considered because they used a reduced number of examples from the dataset.

chroma). The sparser chromas arising from the NNLS and "simple" chroma, obtained higher scores in comparison with the QM chroma. Additionally, even though the "simple" chroma is less effective than the NNLS chroma, it is computationally far more efficient. Therefore, depending on the application and computational power available, one may consider applying one of these different chroma representations.

As expected, the performance of our key finding method gives better results using symbolic inputs than audio in the K-P and WTC datasets. However, the results for both these datasets in the audio domain are still quite high, which we believe is due to the monotimbral nature of the audio renderings. Multitimbral datasets would likely lead to a deterioration of results. From audio signals the convergence to a correct key estimate is also slower than with symbolic inputs.

Based on the evaluation results, we utilize Temperley [1999] key profiles, c_{temp}, in Equation (7) to estimate the key of a musical input in our model, as these profiles generally lead to the best performance. For audio input, the NNLS chroma is used to create the beat-synchronous chromas.

8.2. Subjective Analysis

8.2.1. Listening Test Design. The aim of the listening experiment was to explore the relationship between user judgments of the fitness of the generated accompaniment and the database rankings, with the hypothesis that user judgments would be positively correlated with database rankings. To maximize the control over the experiment results we only tested generated accompaniments given a MIDI input melody. In order to retain as much control as possible over the stimuli, we restrict ourselves to a set of musical accompaniments driven from MIDI inputs. Therefore, accompaniments driven from audio were not tested in this experiment.

To create the dataset used in the listening experiment, we randomly selected a set of ten traditional melodies from the Essen corpus [Schaffrath and Huron 1995] and created 3-note accompaniments for each melody with different ranking scores. For each melody we sampled the highest, middle and lowest ranked database configurations (the first, fourth, and seventh ranked database configurations). In sum, our database comprised 30 musical examples based on 10 given melodies, for which three quite disparate accompaniments in terms of database rankings were generated per melody. Rhythmically, we automated the triggering of the pitch configuration to occur at each target beat onset. All generated notes had the same velocity and distinct pitch ranges for melody and accompaniment were adopted. We synthesized the generated output in Logic using vibraphone samples for the melody and piano samples for the accompaniment and perceptually balanced their loudness.

The experiment was run as follows: for each musical excerpt, the participants were asked to rate on a 7-point scale *how well the accompaniment fits the melody within the context of the tonal music system* as well as *how much they enjoy the melody* to control if the user enjoyment ratings of the melody had any significant impact on the accompaniment ratings. To allow the participants to familiarize themselves with the experiment and clearly identify the sounds used to synthesize the melody and the accompaniment, each experiment trial had a short preliminary training phase. In total, 14 participants were recruited to take the listening test. The population of the study included musicians with 6 years of musical training on average. To prevent order effects, the musical stimuli were presented in a random order at each experiment trial. Participants were not paid for taking part in the experiment.

8.2.2. User Interaction Experiment Design. The aim of the pilot user interaction experiment was to make initial observations about the usability and learnability of D'accord. To this end, we invited participants to experiment with D'accord under constrained

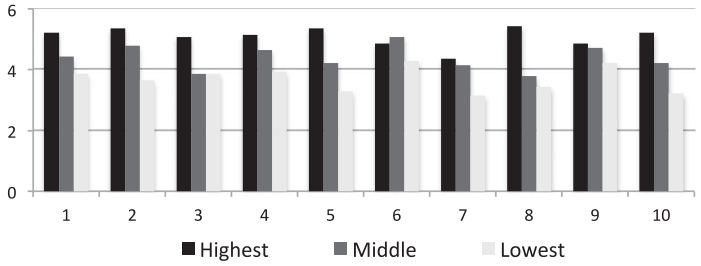

Fig. 5. Mean user ratings of the accompaniment per listening expert. Black bars correspond to the highest ranked examples by the system, the dark gray the middle ranked examples and the light gray the lowest ranked examples.

conditions. We invited participants who were minimally acquainted with technology and thus we restricted the pool of possible participants to university students or those with a degree already. Additionally, we used two participant target groups in order to test if the software could facilitate music creation regardless of the user's music expertise. In total, six participants took the experiment, of whom three were musicians and the remaining three non-musicians. All participants volunteered to take the experiment and were not paid.

The experiment was run individually for each participant as follows: First, we presented the general purpose of D'accord to the participants as a tool to assist users in generating musical harmony. Then, we explained the possible controllers they could use to interact with D'accord were the mouse and the keyboard (using the keys 1–7) as well as the role of the color coding to indicate the database rankings in the interface. Next, we asked the participants to listen to the input in order to familiarize themselves with it and finally we asked each participant to generate an accompaniment to that input file using D'accord. The last two steps were repeated twice in order to test two different input types: the first was a slow MIDI melody and the second a relatively fast polyphonic audio track comprising voice and chorus.

Precise observations and notes concerning the participants' interaction with D'accord were taken. The observations aimed to gather data about the participants' understanding of D'accord's interface usability and learnability. At the end of the experiment, participants were asked to describe their overall experience with D'accord in terms of learnability and usability/operability.

8.2.3. Results. To examine the results of the listening test we now inspect the mean ratings per stimuli, as displayed in Figure 5 ordered from the highest to the lowest ranked database configurations per listening excerpt. The overall user ratings across all excerpts for the highest, middle, and lowest ranked configurations concur with the order proposed by the system. The only exception is observed in excerpt 6, in which the highest ranked configuration obtained a lower rating than the middle one. Listening back to excerpt 6, we can identify that the middle ranked accompaniment is slightly more dissonant in relation to the highest ranked example, while still providing a compelling harmonization of the input melody. In fact, consonance or perceptual relatedness alone does not guarantee good artistic results, but rather a variable degree of dissonance, complexity, and information flow [Smith and Cuddy 1986].

Performing a paired T-test on the user ratings revealed a highly significant difference between both highest and middle, and middle and lowest ratings ($p < 0.001$). This result endorses the overall effectiveness of D'accord's ranking of the database configurations.

The scatter plot in Figure 6(a) shows a more general relationship between the user ratings and chord ranking order (i.e., assuming the values 1, 4, and 7 for the

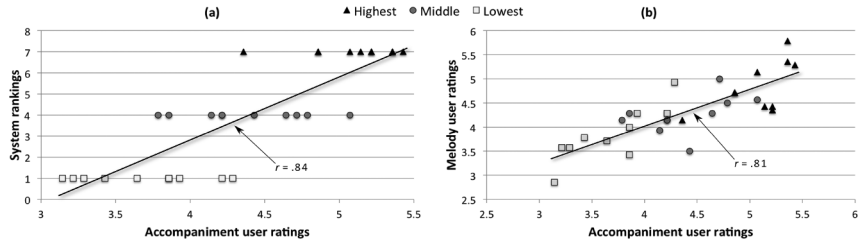

Fig. 6. Scatter plots of (a) overall pitch configurations ratings vs. user ratings and (b) the relationship between the accompaniment user ratings vs. the melody user enjoyment ratings. The best linear fit is shown for each plot which corresponds to the Pearson correlation coefficients $r = 0.84$ and $r = 0.81$, respectively.

lowest- middle- and highest-ranked accompaniments, respectively). A Pearson correlation coefficient of $r = 0.84$, ($p < 0.001$), indicates a statistically significant correlation between the variables on this dataset. In the context of D'accord, this ensures that the ranking order of the system conforms to the judgments of musicians. However, lower ranked results—meaning more dissonant and less related perceptually to the target beat—may still be acceptable in a performance context for expressive purposes.

The scatter plot in Figure 6(b) displays the relationship between the accompaniment user ratings and the melody user enjoyment ratings. A Pearson correlation coefficient of $r = 0.81$, ($p < 0.001$), indicates a statistically significant correlation which denotes that, even though the population of the test was entirely composed of trained musicians capable of distinguishing and evaluating the melody and accompaniment independently, the accompaniments do have an impact on the user ratings of the melody enjoyment.

The results of the user interaction experiment showed that the interface is easy to understand and operate. All participants (including those with and without musical training) agreed that the interface was appealing and the commands were simple. In terms of controllers, the majority showed a preference towards the keyboard interaction, which gave a larger degree of control over the generation. However, one of the participants from the musicians group noted the difficulty of having the same keyboard key give distinct scale degrees or chords. Surprisingly, a non-musician participant preferred the mouse interaction mode, reinforcing a possible game-like interaction. Having fully grasped the interface principles, the participants' attention was directed to the input file and how they could create expressive music. We noticed that all participants start directing their attention to the input file structure rather than the interface principles by the end of the experiment, enabling them to generate more compelling results by triggering events on salient points of the input metrical structure. The control of note durations was the poorest aspect in all participants' performance. Finally, two of the musician participants asked to experiment with their own music collection, and made some interesting remarks related to the database candidates used. Both noted that D'accord performed worse under inputs with complex harmonic structures because, in their opinion, the database candidates were too simple. Overall, the experiment results provided good indicators that support the ability to create meaningful harmonic structures regardless of the user's music expertise.

9. CONCLUSIONS AND FUTURE WORK

In this article, we have presented D'accord, a system for automatic harmonic accompaniment generation of a given MIDI or audio input. The primary contributions of our system are (i) the proposal of multiple ranked solutions for generating an

accompaniment of a target beat, thus widening the user scope for creative endeavor and experimentation, (ii) the possibility to dynamically vary the number of generated voices using the same algorithmic principles, (iii) an innovative adaptive interface composed of a threefold layer of information concerning database rankings, voice-leading parsimony, and typical motion between harmonic categories, and (iv) the processing of both MIDI and audio signals by a unique operation chain.

On the backend of our system is a Tonal Interval Space, in which we compute indicators of tonal pitch relatedness and consonance, as well as estimate the key of the musical input. While the latter underpins the dynamic creation of a database of key-related pitch configurations, the first two indicators are responsible for ranking the database configurations. Our ranking method not only excludes the need for a robust note transcription of the input, but it is also resilient to some errors in the input represenation without causing severe repercussions in the output.

Through our evaluation, we have shown that the proposed key finding method is able to reliably estimate the key from symbolic and musical audio to a degree which outperforms current systems. Additionally, our algorithm is not dependent on any training data as in Lindenbaum et al. [2015]—whose corpus-driven model presents many similarities with ours—but rather on a theoretical construct, which improves the efficiency and adaptability of our system. A subjective evaluation of the output of D'accord showed that the system rankings were highly correlated with user preferences for our dataset.

D'accord's interaction design is multifaceted in the sense it offers various solutions adapted to live performance or studio contexts and different levels of expertise in music and technology. When interacting with the interface via a mouse controller, we noted that the rate of information, which under fast tempos can be quite pronounced, made controlling the system challenging. To minimize this effect, we adopted two other strategies using the computer keyboard or an external MIDI device, which improved the control over the precise selection of the database configurations sorted according to the rankings. D'accord is available online and several examples generated driven from both symbolic and audio inputs and using different interaction modes are available at: https://smc.inescporto.pt/technologies/daccord.

The current implementation of our system is restricted to the generation of 'general' tonal music accompaniments. Stylistic instantiations are planned for future work in terms of generation and voice leading treatment. When restricting the system to a particular style we may be able to redefine the set of voice-leading rules to an idiosyncratic set (e.g., dissonance treatment in Baroque music).

We believe that a more robust key finding method based on the Tonal Interval Space can be developed if a more exhaustive comparison of chroma representations, key profiles and input segmentation strategies are studied outside of the constraints defined *a priori* by the application requirements.

While D'accord can generate interactive harmonic accompaniments in real-time, it currently relies on offline processing to extract the chroma and beat information from the musical input. In future work, we intend to pursue a fully real-time version able to cope with performed musical input through the use of online chroma estimation and beat tracking, as well as developing the interface for handheld devices with touchscreens.

ACKNOWLEDGMENTS

The authors would like to thank the editor and anonymous reviewers for their valuable comments and suggestions, which were helpful in improving the paper.

REFERENCES

Reinhold Behringer and John Elliot. 2009. Linking physical space with the Riemann Tonnetz for exploration of Western tonality. In *Music Education*, João Hermida and Mariana Ferrero (Eds.). Nova Science Publishers. Inc., Hauppauge, NY.

Emmanouil Benetos, Simon Dixon, Dimitrios Giannoulis, Holger Kirchhoff, and Anssi Klapuri. 2013. Automatic music transcription: Challenges and future directions. *Journal of Intelligent Information Systems* 41, 3, 407–434.

Gilberto Bernardes, Diogo Cocharro, Carlos Guedes, and Matthew Davies. 2016. A multi-level tonal interval space for modelling pitch relatedness and musical consonance. *Journal of New Music Research* 45, 4, 281–294. .

Louis Bigo, Jérémie Garcia, Antoine Spicher, and Wendy E. Mackay. 2012. Papertonnetz: Music composition with interactive paper. In *Proceedings of the 9th Sound and Music Computing Conference* (Copenhagen, Denmark). 219–225.

Axel Bruns. 2007. Produsage: Towards a broader framework for user-led content creation. In *Proceedings of the 6th ACM SIGCHI Conference on Creativity and Cognition*. ACM, New York, 99–106.

Chris Cannam, Michael Jewell, Christopher Rhodes, Mark Sandler, and Mark d'Inverno. 2010. Linked data and you: Bringing music research software into the semantic web. *Journal of New Music Research* 39, 4, 313–325.

Elaine Chew. 2000. *Towards a mathematical model of tonality*. PhD dissertation, MIT.

Richard Cohn. 1997. Neo-Riemannian operations, parsimonious trichords, and their "tonnetz" representations. *Journal of Music Theory* 41, 1–66.

Matthew E. P. Davies, Mark D. Plumbley, and Douglas Eck. 2009. Towards a musical beat emphasis function. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. 61–64.

Rui Dias and Carlos Guedes. 2012. Gimmedablues app for iOS: Overview and ongoing developments. In *Proceedings of the SIGGRAPH Asia 2012 Symposium on Apps (SA'12)*. ACM, New York, Article 2, 1 page.

Simon Dixon, Matthias Mauch, and Amélie Anglade. 2010. Probabilistic and logic-based modelling of harmony. In *Proceedings of the 7th International Conference on Exploring Music Contents*, Sølvi Ystad, Mitsuko Aramaki, Richard Kronland-Martinet, and Kristoffer Jensen (Eds.). Springer-Verlag, Berlin, 1–19.

Arne Eigenfeldt and Philippe Pasquier. 2010. Realtime generation of harmonic progressions using controlled Markov selection. In *Proceedings of the ICCC-X-Computational Creativity Conference*. 16–25.

Morwaread M. Farbood, Gary Marcus, and David Poeppel. 2013. Temporal dynamics and the identification of musical key. *Journal of Experimental Psychology: Human Perception and Performance* 39, 4, 911–918.

Dan Gang, Daniel Lehmann, and Naftali Wagner. 1997. Harmonizing melodies in real-time: The connectionist approach. In *Proceedings of the International Computer Music Association*. 27–31.

Emilia Gómez. 2006. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing* 18, 3, 294–304.

Garth Griffin and Robert Jacob. 2013. Priming creativity through improvisation on an adaptive musical instrument. In *Proceedings of the 9th ACM Conference on Creativity and Cognition*, Ellen Yi-Luen Do, Steven Dow, Jack Ox, Steve Smith, Kazushi Nishimoto, and Chek Tien Tan (Eds.). ACM, New York, 146–155.

Christopher Harte and Mark Sandler. 2005. Automatic chord identification using a quantised chromagram. *Audio Engineering Society Convention.* 118, 1–6.

Christopher Harte, Mark Sandler, and Martin Gasser. 2006. Detecting harmonic change in musical audio. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*. ACM, New York, 21–26.

Christopher Harte. 2010. *Towards automatic extraction of harmony information from music signals*. Doctoral dissertation, Queen Mary, University of London.

David Huron. 1994. Interval-class content in equally tempered pitch-class sets: Common scales exhibit optimum tonal consonance. *Music Perception: An Interdisciplinary Journal* 11, 3, 289–305.

David Huron. 2001. Tone and voice: A derivation of the rules of voice-leading from perceptual principles. *Music Perception* 19, 1, 1–64.

William Hutchinson and Leon Knopoff. 1978. The acoustic component of western consonance. *Interface – Journal of New Music Research* 7, 1, 1–29.

Özgür Izmirli. 2006. Audio key finding using low-dimensional spaces. In *Proceedings of the International Society for Music Information Retrieval Conference*. 127–132.

Akio Kameoka and Mamoru Kuriyagawa. 1969. Consonance theory. Part I: Consonance of dyads. *Journal of Acoustical Society of America* 45, 6, 1451–1459.

C. L. Krumhansl and E. J. Kessler. 1982. Tracing the dynamic changes in perceived tonal organization in a spatial map of musical keys. *Psychological Review* 89, 334–368.

Carol L. Krumhansl. 1990. *Cognitive Foundations of Musical Pitch*. Oxford University Press, New York.

Carol L. Krumhansl and Mark A. Schmuckler. 1986. Key finding in music: An algorithm based on pattern matching to tonal hierarchies. Paper presented at the 19th annual Mathematical Psychology meeting, Cambridge, MA.

Fred Lerdahl. 2001. *Tonal Pitch Space*. Oxford University Press.

David Lewin. 1987. *Generalized Musical Intervals and Transformations*. Yale University Press, New Haven, CT.

Ofir Lindenbaum, Arie Yeredor, and Israel Cohen. 2015. Musical key extraction using diffusion maps. *Signal Processing* 117, C, 198–207.

Hugh C. Longuet Higginsz. 1962. Two letters to a musical friend. *Music Review* 23, 244–248, 271–280.

Constantine F. Malmberg. 1918. The perception of consonance and dissonance. *Psychological Monographs* 25, 2, 93–133.

Bill Manaris, David Johnson, and Yiorgos Vassilandonakis. 2013. Harmonic navigator: A gesture-driven, corpus-based approach to music analysis, composition, and performance. In *Proceedings of the 9th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment.* 67–74.

Matthias Mauch and Simon Dixon. 2010. Approximate note transcription for the improved identification of difficult chords. In *Proceedings of the 11th International Society Music Information Retrieval Conference*. 135–140.

Katy Noland and Mark B. Sandler. 2006. Key estimation using a hidden Markov model. In *Proceedings of the International Society for Music Information Retrieval Conference.* 121–126.

François Pachet and Pierre Roy. 2001. Musical harmonization with constraints: A survey. *Constraints Journal* 6, 7, 7–19.

François Pachet and Pierre Roy. 1998. Formulating constraint satisfaction problems on part-whole relations: The case of automatic musical harmonization. In *Proceedings of the ECAI Workshop on Constraints for Artistic Applications*. 1–11.

François Pachet. 1994. The muses system: An environment for experimenting with knowledge representation techniques in tonal harmony. In *Proceedings of the 1st Brazilian Symposium on Computer Music*. 95–201.

Hélène Papadopoulos and George Tzanetakis. 2012. Modeling chord and key structure with Markov logic. In *Proceedings of International Society for Music Information Retrieval Conference*. 121–126.

Somnuk Phon-Amnuaisuk, Andrew Tuson, and Geraint Wiggins. 1999. Evolving musical harmonisation. *Artificial Neural Nets and Genetic Algorithms*. Springer, Vienna, 229–234.

Hugo Riemann. 1893. Vereinfachte harmonielehre, oder die lehre von den tonalen funktionen der akkorde. 1896. *Tr. H. Bewerunge*, Augener. London.

Thomas Rocher, Matthias Robine, Pierre Hanna, and Laurent Oudre. 2010. Concurrent estimation of chords and keys from audio. In *Proceedings of the International Society for Music Information Retrieval Conference*. 141–146.

Helmut Schaffrath and David Huron. 1995. The Essen Folksong Collection in the Humdrum Kern Format. Center Computer Assisted Research in the Humanities, Menlo Park, CA..

Marc Schonbrun. 2011. The everything music theory: Take your understanding of music to the next level. *Adams Media*.

Roger Shepard. 1982. Structural representations of musical pitch. In *The Psychology of Music*, Diana Deutsch (Ed.). Academic Press, 335–353.

Karen Smith and Lola Cuddy. 1986. The pleasingness of melodic sequences: Contrasting effects of repetition and rule-familiarity. *Psychology of Music* 14, 1, 17–32.

Mark Steedman. 1999. Categorical grammar. In *The MIT Encyclopedia of Cognitive Sciences*. MIT Press.

David Temperley. 1999. What's key for key? The Krumhansl-Schmuckler key-finding algorithm reconsidered. *Music Perception: An Interdisciplinary Journal* 17, 1, 65–100.

Geraint Wiggins. 1999. Automated generation of musical harmony: What's missing. In *Proceedings of the International Joint Conference in Artificial Intelligence*.

Received May 2015; revised October 2015; accepted January 2016