SmartDrone: An Aurally Interactive Harmonic Drone

Travis MacDonald
St. Francis Xavier University
Computer Science
Antigonish, NS, Canada
x2015khn@stfx.ca

Barry MacKenzie
St. Francis Xavier University
History
Antigonish, NS, Canada
bmackenz@stfx.ca

James A. Hughes
St. Francis Xavier University
Computer Science
Antigonish, NS, Canada
jhughes@stfx.ca

ABSTRACT

Mobile devices provide musicians with the convenience of musical accompaniment wherever they are, granting them new methods for developing their craft. We developed the application SmartDrone to give users the freedom to practice in different harmonic settings with the assistance of their smartphone. This application further explores the area of dynamic accompaniment by implementing functionality so that chords are generated based on the key in which the user is playing. Since this app was designed to be a tool for scale practice, drone-like accompaniment was chosen so that musicians could experiment with combinations of melody and harmony.

The details of the application development process are discussed in this paper, with the main focus on scale analysis and harmonic transposition. By using these two components, the application is able to dynamically alter key to reflect the user's playing.

As well as the design and implementation details, this paper reports and examines feedback from a small user study of undergraduate music students who used the app.

Author Keywords

Interactive, Dynamic Accompaniment, Jazz Music, SmartPhone, Drone.

CCS Concepts

•Human-centered computing → Interaction design; •Applied computing → Performing arts; Sound and music computing;

1. INTRODUCTION

One of the dominant schools of modern jazz pedagogy is rooted in the Russellian theory of chord-scale relationships [1, 7]. In this theoretical framework, chords and scales are viewed as roughly isomorphic objects based on harmonic compatibility (an example is shown in Figure 1.) The popularity of the chord-scale approach for jazz pedagogy is no surprise: it provides an easily teachable framework for melodic improvisation.

Due to the prevalence of chord-scale theory, it is common for jazz musicians to dedicate significant practice time to the execution of scales. However, since practicing is commonly an unaccompanied activity, musicians are most often hearing scales without any harmonic context, thereby missing out on the chord half of the relationship.

In order to provide a solution for this problem, SmartDrone generates harmonically dynamic accompaniment for the user. It does so by playing sustained backing chords, which can be customized in order to meet the requirements of the user.

Another primary consideration for this project is the inconvenience of interacting with technology while practicing. Of course, it is necessary that applications require some form of interaction in order to function (e.g., mouse click, screen touch); however, constant direct interaction can be disruptive while practicing. In order to reduce this, SmartDrone will, without additional user intervention, adaptively change keys in order to reflect what the user is playing on their instrument.

We believe that the combination of these features will provide accessibility for individuals with different abilities, including those with disabilities and neurodiverse musicians. Ultimately, this is achieved by giving the user more freedom in how they practice their instruments. For example, it allows the user to interact with the tool with minimal physical movements. Further, it provides individuals with harmonic accompaniment without needing other musicians, which makes practice more accessible and gives them more freedom to make music where they feel most comfortable.

2. JAZZ THEORY

The most common scales in jazz theory are the major, melodic minor, and harmonic minor. Six additional modes can be derived from each of these scales, resulting in 21 unique scales. These scales are common material in chord-scale jazz theory as they provide a foundation for melody and harmony.

Chord-scales can be applied to jazz improvisation, particularly melodic improvisation over the chords of jazz compositions. Using the mapping technique of chord-scale theory, musicians can determine which scales to play in correspondence with the chord changes of a composition, thus providing a jumping-off point for melodic improvisation.

One could decompose the goal of jazz performance education into two main parts: developing the ability to accurately discern melodies and harmonies, and developing the technical capacity to play one's instrument. The first enables the musician to translate the music they hear onto their instrument, while the second allows them to execute it in real-time.

Since technical ability correlates with muscle memory, unaccompanied practice is not a primary concern. However, a musician's capacity to hear scales is directly affected by this factor since the underlying harmony influences the sound of the scale.

NIME'20, July 21-25, 2020, Royal Birmingham Conservatoire, Birmingham, UK.

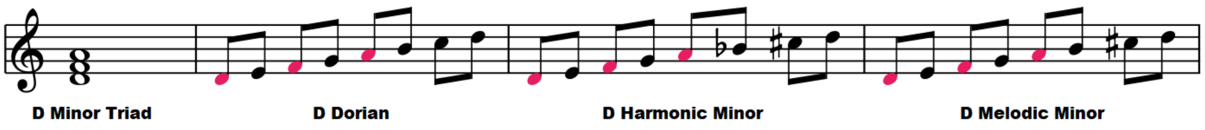


Figure 1: Three scales that can be mapped to a D minor triad.

For example, the sound of a C major scale is perceived differently depending on the harmonic context. If the scale is played with a C in the bass, the resulting sound will be C Ionian (first mode of the major scale.) If the same scale is played with a D in the bass, the resulting sound will be D Dorian (second mode of the major scale.) Thus, it is worth considering the usage of harmonic accompaniment when practicing scales in order to realize chord-scale relationships.

3. RELATED WORKS

One simple form of accompanied practice can be achieved with the use of a reference pitch. This refers to a sustained note that is played in order to establish a key center. A piano player can do this by playing a chord with one hand and a melody with the other, allowing them to experiment with different chord-scale combinations. But since most instruments can only produce one note at a time, a reference pitch must be provided in some other way.

One such solution is *DroneToneTool*[1] [6]: an app designed for reference pitch practicing. Users can choose the key by interacting with the onscreen keyboard, and like standard drones, *DroneToneTool* will play the root of the key. By playing only the root, the key is stabilized, yet nothing is implied about the tonality of the chord.

One advantage of only playing the root is that the user isn't restricted to a specific tonality and can change freely between scales (e.g. from C Ionian to C Dorian.) However, if the user has the intention of practicing a specific scale then the restriction of one tonality isn't necessarily a problem. For this reason, SmartDrone allows users to use chords, as well as single notes, as a reference pitch.

The area of generative music has had a wide range of applications in recent years. In the context of jazz, there are genetic algorithm based applications that generate chord sequences [9], melodies [5], and solos [3]. There are also applications that explore the area of real time accompaniment based on musical interaction from the user [10]. This application learns to accompany users based on example performances in order to achieve a more human-like accompaniment.

4. DEVELOPMENT

SmartDrone is a mobile application implemented for Android devices. The functionality of this application consists of four main components: pitch processing, scale analysis, harmonic transposition, and midi synthesis. The way these components work together in order to achieve hands free accompaniment is discussed in the following section.

Since the pitch processing and midi synthesis components both come from external libraries (*TarsosDSP*[2] [8] and *MidiDriver*[3] [4], respectively), the only scale analysis and harmonic transposition components are discussed.

[1]http://www.dronetonetool.com/
[2]https://github.com/JorenSix/TarsosDSP
[3]https://github.com/billthefarmer/mididriver

4.1 Application Flow

The process of this application begins with a continuous stream of data from the user's microphone. First, that information is transformed into pitch results and sent to the scale analysis model. This model functions by adding recent pitch results and discarding old ones, while recalculating the weight of each key after every operation. If a change in key is detected, then a chord in the new key is constructed and sent off to be played by the midi synthesizer.

4.2 Scale Analysis

The objective of the scale analysis component is to derive a key from a collection of pitch results. Although the current version of this app is limited to just major and melodic minor modes, future iterations will have a larger palette, including harmonic minor, bebop, and diminished scales.

4.2.1 Implementation

Keys are weighted based on the number of notes they share with the user's playing. Since only current notes are relevant in this context, newer pitch results are processed while older ones are discarded. An active key is chosen after it has maintained a threshold weight for the required amount of time.

The central piece of data in the scale analysis model is the note object. The purpose of this object is to store the index of the note so that it can be uniquely identified (the index of the note is the same as a MIDI number: 128 enumerated values, starting with C_{-1} at index 0, up to G_9 at index 127.) This allows differentiation between the same notes of a different octave, such as C_0 at index 12 and C_1 at index 24.

To track notes that have been recently heard, this model uses an *active note list*. Although there are 128 possible note values, this list contains only 12 indices; one index for each note's active status. Because of this design, one occurrence of any note, regardless of octave, denotes that note as active. Alongside this, each active note has a timestamp containing the last time it was registered. If the time since it was heard reaches a certain threshold, it gets discarded. If it is heard again, its timer gets reset.

Similar to the *active note list*, there is a *key weight list*. Like the *active note list*, this one also has a size of 12: one index for each key (since keys are not differentiated by octave.) In this list, each key has an initial value of zero and a max value of the number of notes it contains per octave. When a note becomes active, every key containing that note has its strength incremented by 1, and decremented by 1 when it becomes inactive.

In order to find all keys that contain a target note, a sequential search through all keys could be performed. However, certain scales can be used to find all keys containing a target note. In the major scale, for example, the Phrygian scale (third mode) can be used to find every key containing a target note. Using the note C as an example, the notes of C Phrygian (C, D♭, E♭, F, G, A♭, B♭) correspond with the exact same keys that contain the note C (for melodic minor and harmonic minor, both of their second modes can be used.) Although only a slight optimization, this relationship seemed worthy of mention.

One more consideration for the scale analysis model was to allow a short buffer when changing keys. This is in place to improve the accuracy of key detection. Along with the weight attribute, each key is given a *contender status*. A key is considered a contender based on three parameters: it has a weight greater than zero; it has the max weight of any key (can be tied); and its weight is greater than the current active key. If a key maintains a contender status for the required amount of time (five seconds by default) then it becomes the active key.

4.2.2 Scale Ambiguity

One important consideration for this method of analyzing scales was to address the issue of scale ambiguity. This problem occurs when there isn't enough information from the pitch results to yield a single contender key. With an insufficient number of notes found in the pitch results, numerous keys contain max weight, and the model will guess which key the user is playing in.

Take, for example, that the pitch results indicate the user played only the note C. Drawing from the example in Section 4.2.1, there are a total of seven major keys that contain this note. Based on this result alone, it is ambiguous as to which key the user is in. There will be seven keys that share the max strength of one, and thus this component will choose randomly between the contender keys.

A more interesting case of scale ambiguity involves the major pentatonic scale. This scale is similar to the major scale except that it contains only five notes per octave instead of seven (scale tones 1, 2, 3, 5, and 6.) This scale also has prominent usage in jazz music, notably because of its diversity of melodic functionality [2].

Using C major pentatonic as an example, it is most commonly used in the key of C for a major chord quality. However, there are several other uses: it can be used in the key of D♭ for a Lydian chord quality (fourth mode of major scale), or over D for a Dorian or Mixolydian chord quality (second and fifth mode of major scale), to name a few.

The reason that the pentatonic scale has these functionalities comes back to the general idea of chord-scale relationships: mapping scales to chords based on their harmonic compatibility. Since the pentatonic scale only contains five notes it is more harmonically flexible, and therefore can be mapped to many chords. In each of the previous examples, the pentatonic scale contained every note in the key it was used in. For this reason, the pentatonic scale also has some ambiguity with regards to the key it is in.

This raises an issue when it comes to the implementation of the scale analysis model. If the pitch results show a C major pentatonic scale (C, D, E, G, A), then the module would have three scales that share the max weight of five: C, F, and G major. There could be an argument made that the pentatonic scale is most commonly in its root key, so C major should be chosen in this case. However, the problem with this is that it limits the user to specific usage of this scale and therefore will not respond correctly if their intention is otherwise. Considering this issue, it seemed best to not make any assumptions about the user's intention.

Major, melodic minor, and harmonic minor scales, however, are not inherently ambiguous in this implementation. If the pitch results show that the user has played every note of a major scale, then that scale will be the only contender key and therefore chosen as the active key.

There are common ways of practicing scales in jazz education: ascending and descending, intervals (3rds, 4ths) to more complex variations (e.g. 7th chord arpeggios.) However, practicing in this manner does not demand anything special from the user; pitch results indicating these scale variations will suffice in determining the correct key, since every note of the scale is being played.

One last final point to touch on in terms of scale ambiguity comes from discerning different modes of a scale. This method of analyzing scales is not able to differentiate between modes of the same scale (e.g. D Dorian and E Phrygian), since it only tracks active notes. The only information that the model is sure of is that the user is in the key of C major. For this reason, the app requires the user to select which mode they are going to practice in.

4.3 Harmonic Transposition

Whenever a key change is detected the harmonic transposition component's job is to construct a chord for playback. Rather than programming specific chord voicings, this model was designed so that different harmonic components can work together to construct chords based on a few parameters.

4.3.1 Chord Components

The components that go together in order to make a playable chord (referred to as a *voicing*), are the *voicing template*, *mode*, and *key*.

The *voicing template* object contains a specific organization of chord tones. Chord tones only indicate the degree of a scale and do not imply any specific harmonic context (e.g. a degree of three indicates the third note of a scale.) Because each scale has a different intervallic makeup, the actual transformation from chord tone to note depends on the scale it is applied to.

In order to structure a group of chord tones, a *voicing template* object is used. This object differentiates between chord tones and bass tones so that they can be played in the proper harmonic register. As with the chord tone objects, *voicing templates* are also dependant on the harmonic context.

Take for example a *voicing template* made up of four chord tones with degrees 1, 3, 5, and 7 (this makeup resembles a closed 7th chord in root position, common in music theory.) Dependant on the scale, this template will result in a different voicing. If C Ionian is applied to it, then the resulting voicing will be C, E, G, and B. If C Dorian is applied to it, then the resulting voicing will be C, E♭, G, and B♭.

The *voicing template* is the only data that the user is required to configure. This means that if the user wants to practice with any different quality of a voicing, then they can program the template once and select the scale that will result in the desired voicing.

As for the remaining parameters required for the construction of a voicing, the *key object* in this case is quite simple. The sole purpose of this object is to store the index of the key (similar to the note index in Section 4.2.1.) The only difference is that there are only 12 values (one index for each note.)

With regards to the diatonic scales used in SmartDrone, major and melodic minor, each mode is made up of seven intervals. In the *mode object* the seven intervals are represented by a sequence of integers, where each integer contains the number of semitones the scale degree is away from the root. For example, the sequence for the Phrygian mode is 0, 1, 3, 5, 7, 8, and 10. From observing this example, the second degree is one semitone away from the root, the third is three semitones away, and so on.

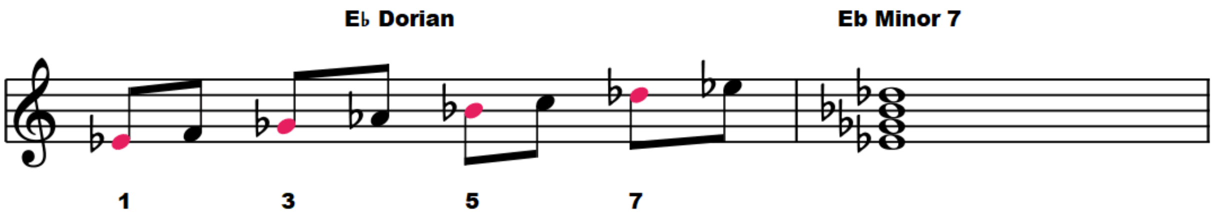
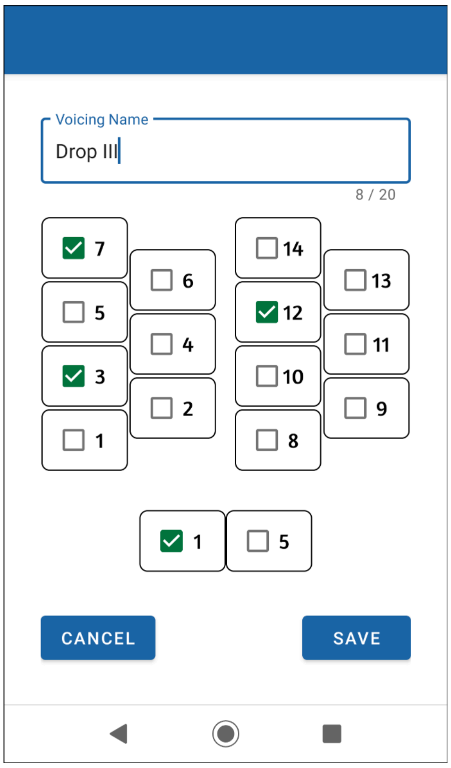


Figure 2: Transforming a voicing template into a voicing.



✓ 7	6	14	13
5	4	✓ 12	11
✓ 3	2	10	9
1		8	

✓ 1	5

Figure 3: Screenshot of the voicing template creator activity.

4.3.2 *Chord Construction*

With the three components from the previous section, a chord voicing can be constructed. The voicing object that can be played by the MIDI synthesizer consists of a collection of note objects.

A chord tone is transformed into a note by finding the mode's interval at the index of the tone's degree. After the interval is found, the index of the key is added to it, resulting in the corresponding note. Lastly, in order to transpose the note to the proper octave, the current result is summed with 12 multiplied by the number of octaves.

To demonstrate an example, the voicing template from the previous section (1, 3, 5, and 7) is used in combination with the Dorian mode and the key E♭. First, each tone finds the interval at the index of its degree, resulting in indices 0, 3, 7, and 10. Following this, the index of the key (index of E♭ is three) is added to each of the indices. By default, this application transposes chord tones up four octaves, so 48 is added to each index (four octaves was chosen so that the voicing will be in the typical range of harmonic jazz accompaniment.) The final result is a voicing that contains the indices 51, 54, 58, 61, which translates to the notes E$\flat_4$, G$\flat_4$, B$\flat_4$, and D$\flat_5$ (E♭ minor 7, see Figure 2.)

Thus, through this process of chord construction the user only has to program one voicing template. If the user decides to practice with a different chord quality, then they are only required to change the scale parameter.

It is worth noting that a skillful jazz accompanist would utilize voice leading techniques, as opposed to relying on one single voicing template. Voice leading allows for more creative accompaniment, which in turn influences the musical choices of the soloist. Although there is no voice leading in the current version, future iterations may implement this feature.

5. USAGE

In order to best meet the user's intention, SmartDrone has settings that allow custom specifications of a few parameters. These parameters mostly affect the operations of the scale analysis and harmonic transposition model.

Using the device's speaker for audio playback may cause inaccurate pitch detection due to the microphone receiving the phone's output as input. Because of this it is best to use either headphones or an external speaker with SmartDrone. To further improve pitch detection the device's microphone should be placed close to the source of the instrument's output.

5.1 Configuration

In the settings screen of SmartDrone, the user can choose the scale/mode, plugin, and voicing template. The settings screen also has audio playback so the user can hear what the current configuration sounds like.

The first parameter is the mode the user is going to practice in. In the current version of the application, all modes of the major scale and melodic minor are supported. The plugin is simply the sound of the drone, with the choices of a brass section, string section, or choir.

A more important feature is the voicing creator screen which allows the user to create their own voicing template (shown in Figure 3.) Each number represents the degree of the scale, with the bottom row representing the bass tones and the upper columns representing the chord tones. This screen also has playback so the user can hear how each change to the voicing alters the sound (the current mode in the key of C is used in this case.) The settings page also contains a list of all of the user's voicings so that they may save and change between voicing templates.

Alongside the settings that affect the playback of the drone, there are two additional settings when configuring the drone: active key sensitivity and note length filter.

The active key determines how long a contender key must be heard before it becomes the active key. A higher sensitivity means the keys require less time to change, and are therefore better suited for practicing that involves more key changes.

The note length filter determines how long a pitch must

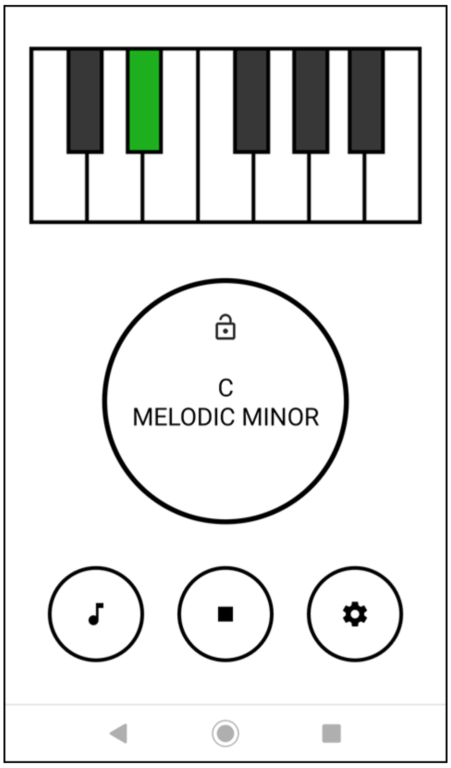


Figure 4: SmartDrone in action: the onscreen keyboard indicates an E♭ has been detected by the microphone.

be consecutively registered for before it is added to the active note list. A longer filter means that a note must reach a higher threshold before it is considered an active note. This filters out noise and inaccurate frequencies that may lead to inaccurate key detection.

5.2 User Interface

Upon starting the app, the drone is in an inactive state. Users can start the drone by pressing the play button and the application will begin listening for input. The current note that is heard will be displayed in green on the onscreen piano. This provides the user with feedback about the accuracy of the pitch detection. Figure 4 shows a screenshot of the user interface.

One additional feature of SmartDrone allows the user to lock the key they are in. This is useful when playing outside of a key, which is sometimes done in jazz music.

An example of outside playing could be playing a C major scale over a G♭ major chord. Although the resulting sound is dissonant, it is a tension that is sometimes utilized in the music. If a user tried to practice notes outside the key without the lock, the application would assume the user is changing keys. Using a lock solves this problem as the user is free to play any scale without the worry of changing the active key. The lock is activated by pressing the active key button in the center of the screen.

6. EVALUATION

6.1 Methodology

Once the application was complete a few students from the music program at St. Francis Xavier University were invited to participate in a study, using the version of the app described in this paper. The demographic of these musicians consisted of one female vocalist and two male guitarists. Ultimately, the goal of this study was to gauge the application's impact on a typical music student.

To start this process the participants were given the app on their smartphone, as well as a brief explanation as to how the application works. Participants were encouraged to use the application in any way that suited their individual practice, so that they were not restricted to one specific method of practice. The application was used by the individual during their own time, wherever it fit into their own practice schedule.

After the participants had sufficient experience with SmartDrone, individual interviews were conducted. These interviews consisted of open-ended questions so that the narrative of the discussion would be directed by the user's feedback. Participants were encouraged to discuss any aspect of the app that had an effect on their practice session.

6.2 Results

The interviews indicated that each user had used the drone in a way that best served their own musical goals. The main themes of the interview questions consisted of the harmonic accompaniment, dynamic key changing, and the overall user interaction with the app.

6.2.1 Dynamic Key Changing

Although one of the primary consideration for this application was harmony that could change on its own, the participants reported that they mostly stayed in one key while using the drone. One user even reported that they would choose a specific sound (e.g. C Dorian) and practice different variations of scale exercises (3rds, 4ths, and so on) for 20 minutes. Therefore, the dynamic key change feature seemed more underutilized than we had originally anticipated.

Although this feature was not extensively used, one participant reported that when they did use it, the application worked correctly most of the time. They did note, however, that the application seemed to have difficulty changing between similar keys; they noted that keys that were a 4th or 5th apart would have to be "coaxed" in order to change. This makes sense considering that keys that are this far apart only have a difference in weight of one, hence one wrong note is enough to cause an inaccurate key detection. This participant also noted that sometimes the application would change keys even though the user had not, which was distracting to their practice session.

The first participant reported that since they used the application primarily for the harmonic accompaniment features, they felt an option to manually choose the key, instead of playing a scale, would be beneficial. They expressed that with this feature the application would become more user friendly, especially for vocalists, since singing exact pitches is more difficult than playing them on an instrument.

6.2.2 Harmonic Accompaniment

The harmonic functionality seemed to provide the most utility for the user. Each participant reported that they used the drone in some way to sharpen their ability to recognize intervals. More specifically, two of the participants reported that they experimented with different harmonic tensions, as a way to gauge exactly which sounds they preferred.

One of these two said that they used Dorian chords in order to emulate a specific jazz artist, and then experimented by playing different notes on top of the chord. They reported moving between different chord-tones and non-chord tones in order to find the specific harmony that matched the sound they were looking for.

Another user reported similar usage, saying how they picked a single chord from the drone, and experimented with playing different tensions. The user referred to this method as trying out different "colors".

The final participant reported using the settings page as a form of harmonic accompaniment. Since the settings page has harmonic playback, this participant would manually change the scale and voicing template of the drone, and use the resulting voicing as accompaniment. The audio playback on the settings page was intended to give the user an idea of how the drone would sound during the practice session, so it was interesting to note that they had used the feature for ear training purposes.

6.2.3 User Experience

Based on the interviews, the primary usage of the application was for scale practice accompaniment and chord-scale ear training. Since this was also the intended area of usage for SmartDrone, the participants feedback seemed to target the key areas of concern for this project.

One common point made about the application was that it requires a certain level of musical experience to use. One participant suggested that the voicing creation feature might be confusing to less experienced musicians. This participant also stated the less experienced musicians may not understand the purpose of practicing with a drone.

Along these lines, one participant reported that the application would have been more beneficial to them if they had more musical experience. Although they felt there was value in using the app, their own experience with it seemed to mostly highlight "holes" in their musical knowledge, which they described as frustrating. This participant repeatedly expressed how much of a factor their own musical experience was, and thought of this application as a long term solution to ear training, rather than a short term fix.

From the other two participants the general idea that it made practicing a bit more fun was brought up. One of the two expressed that it influenced them to work on modal materials, an area in which they claimed to not practice much. They said that the application encouraged them to stay with one chord and see how many interesting musical ideas they could come up with, without repeating themselves.

The other of the two found that the drone made it "fun to gain fluency" in the technical aspect of scales. They felt that playing scales up and down on their instrument without any harmonic accompaniment was sometimes "just a blur". They found by having the drone accompanying them that it made regimented practice, such as different scale exercises, more "interesting" and "beneficial".

The lack of rhythm in the application was also brought up amongst the participants in the feedback, noting the absence of a metronome or drum track. Although one participant found that by focusing on the just the harmony they had the freedom to "fumble" around, the participants generally felt that some sort of rhythmic feature would be beneficial.

7. CONCLUSIONS & FUTURE WORKS

The overall goal of this project was to reduce the amount of direct interaction required by the user, as well as provide a solution to chord-scale ear training. SmartDrone contains just one approach to address this issue and it is clear that there is still more room for improvement.

It seems that the dynamic key changing feature was not as utilized as we had imagined, so it would be interesting to explore different solutions that could improve this feature. Manually changing keys still has some advantages over this application since it is faster and more accurate. SmartDrone's implementation uses a delay in key change to avoid scale ambiguity and even with this delay it can still guess wrong.

One solution would be to give note objects a weight attribute, instead of an active or inactive status. The scale analysis model could use this attribute to choose an active key when there are multiple contender keys, instead of choosing randomly between them.

Another solution would be to have the user play a specific phrase that dictates a key change. The phrase could be something simple, such as the same pitch one octave apart, and whatever key the phrase was played in would become the new active key. Even though this solution would make the scale analysis model obsolete, it could get very close to the speed and accuracy of manually choosing the key.

Lastly, the valuable user feedback, such as a metronome feature, or manually selecting active keys, will be incorporated into future versions of the application.

8. ACKNOWLEDGEMENTS

The authors acknowledge and thank the research participants for their time and effort.

The authors thank Dr. Matea Drljepan for her artwork.

The authors also thank Dr. Mark Daley for his contributions to the project.

9. COMPLIANCE WITH ETHICAL STANDARDS

Research methods were approved by the appropriate research ethics review board. All research subjects gave informed consent before participation.

10. REFERENCES

[1] J. Aebersold. *How to play jazz and improvise.* Jamey Aebersold, 1967.

[2] J. Bergonzi. *Inside Improvisation. Vol. 2: Pentatonics.* Advance Music, 1994.

[3] J. A. Biles et al. Genjam: A genetic algorithm for generating jazz solos. In *ICMC*, volume 94, pages 131–137, 1994.

[4] B. Farmer. Midi Driver, April 2019. v1.17. `https://github.com/billthefarmer/mididriver/releases`. Accessed: 2019-6-6.

[5] G. Papadopoulos and G. Wiggins. A genetic algorithm for the generation of jazz melodies. *Proceedings of STEP*, 98, 1998.

[6] J. F. Roberts IV. Drone Tone Tool, July 2017/2019. v1.2.3/1.2.5. Google Play Store and App Store Accessed: 2020-1-29.

[7] G. Russell. *The Lydian chromatic concept of tonal organization for improvisation.* Concept Publishing Co., 1959.

[8] J. Six, O. Cornelis, and M. Leman. TarsosDSP, a Real-Time Audio Processing Framework in Java. In *Proceedings of the 53rd AES Conference (AES 53rd)*, 2014.

[9] M. J. Steedman. A generative grammar for jazz chord sequences. *Music Perception: An Interdisciplinary Journal*, 2(1):52–77, 1984.

[10] G. Xia and R. Dannenberg. Improvised duet interaction: Learning improvisation techniques for automatic accompaniment. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 110–114, Copenhagen, Denmark, 2017. Aalborg University Copenhagen.