Theory Controller: A Silent IMS

Thibault Jaccard

Master's programme in Music, Communication and Technology

Department of Music
Norwegian University of
Science and Technology

Department of Musicology
University
of Oslo

June 2021

Abstract

This paper explores music theory real time controllability through the design of an interactive musical interface. More precisely, the instrument described here lets the user select any standard heptatonic scale - including all twelve keys and seven modes of the four main scale classes, diatonic, acoustic, harmonic minor and harmonic major - using a joystick-like gesture input device. The program presents a complex visual interface, and multiple use cases are explored. The system as it is does not produce any sound, it is thus not playable in performance settings on its own. It may however be used in conjunction with other compatible custom apps, such as an arpeggiator or a dynamic MIDI scale mapper, as well as virtual instruments.

The Theory Controller opens a discussion about chord progression awareness as well as inter-plugin scale data synchronisation. The typical user is portrayed, and possible enhancements for the system are envisioned. The latter is also compared with other existing music theory enabled interfaces.

Contents

1 Introduction

Music theory is often seen as an impediment to intuitive musicking. Few musicians feel interested in going deep into the maths behind the music, and those who do risk to be considered as elitist nerds. Moreover, many musical genres do not require one to understand scale theory, extended chords, or modes fully.

The number of possible notes in chromatic instruments may seem overwhelming. Most scales consist of seven notes taken as subsets of twelve possible notes. Apart from the piano centred around the C major scale, most instruments do not offer a hierarchy among notes. An interesting exception is the pedal harp. This diatonic instrument has seven strings, one for each scale degree, and pedals to alter them and thus select a scale. It is this concept that inspired this research. Using modern technology, what interactive music system design could afford music theory navigation? Or more precisely, what music theory parameters are to be assigned to a physical controller in order to intuitively select a mode? Such an interface may have several applications, such as syncing a dynamic MIDI scale mapper. This system could be considered as an electronic version of the harp's pedal/string dissociation. More on this in section 4.4.

Similar research has been sparse, the most noticeable one being The Scale Navigator [18]. The shared concept between the latter and the Theory Controller is having someone controlling the scale selection and having this selected scale inform a sound generating application. We propose a novel kind of dissociation, where the devices selecting the scale and triggering sound are not the same. In this paper, we present the design of a system that affords scale navigation with one hand, while the other hand may be used to play music within this scale in different manners.

degree	**scale class**			
	diatonic	acoustic	harmonic minor	harmonic major
I	ionian	melodic minor	harmonic minor	harmonic major
II	dorian	dorian ♭9	locrian 13	locrian 9 13
III	phrygian	augmented lydian	augmented ionian	phrygian ♭11
IV	lydian	lydian ♭7	dorian ♯11	minor lydian
V	mixolydian	mixolydian ♭13	mixolydian ♭9 ♭13	mixolydian ♭9
VI	aeolian	half-diminished	lydian ♯9	super augmented
VII	locrian	altered	diatonic diminished	diminished locrian

Table 1: All scales used in the Theory Controller

type	core notes			function
	3^{rd}	5^{th}	7^{th}	
augmented	maj	aug	maj	augmented
major	maj	per	maj	tonic, subdominant or other
dominant	maj		min	dominant
minor	min	per		tonic, subdominant or other
half-diminished	min	dim	min	subdominant or other
diminished	min	dim	dim	dominant (dim)

Table 2: Chord types and harmonic functions

2 Model

For this interactive system to exist, we need both a good understanding of western music theory and music interface design. Indeed, in order to have a good and accessible instrument, we need it to be intuitive (well designed, low floor) and comprehensive (affording all scales, high ceiling) [5].

2.1 Music Theory

As the focus is made on western tonal music, we may restrict ourselves to using the chromatic equal-tempered scale. We are, therefore, not including the infinity of possible microtonal tunings in our system. This paper will use the pitch class model [1, p. 776], which numbers the 12 semitones from $C = 0$ to $B = 11$, independently of the octave. For example, the pitch class of a MIDI note number N is calculated as $N \bmod 12$.

According to Ian Ring [16], there are 1490 possible scales. However, not all are used nor useful in our case. We decided to select 28 heptatonic modes, presented in table 1. We omitted other common cases such as diminished octatonic, whole tone or augmented scales, because of their non harmonically structural nature in tonal music. They may nevertheless be considered as exceptions in future research.

We use $12 \cdot 28 = 336$ different scales in our system by adopting the pitch class model. They consist of four groups of seven modes starting on twelve possible root notes, and the same degree of each group is often highly substitutable. For example: On a dominant chord, one could play in a Mixolydian, Mixolydian ♭13, Mixolydian ♭9 ♭13, or a Mixolydian ♭9 scale, which are all the sixth mode of their group. The Mixolydian scale and its counterparts are the default (and defining) scales of the dominant function [10, p. 18-19].

Note that the names are not canonical. Almost each of these scales has more than one name, and it is a matter of preference to choose one or another. However, most of these denominations are descriptive. For instance, Lydian ♭7 is a Lydian scale with a minor seventh instead of a major one.

In composing music, a very common and helpful interval is the perfect fifth. Indeed, it is at the same time the most consonant interval (besides the unison and the octave) and the distance between a dominant and its relative tonic. These qualities explain why keys, chords and scales and often organised in fifths. The most canonical example is the circle of fifths [4][10, p. 23]. It is going to be used in our system.

The concept of chord and scale type is also important here. We use it as defined in AutoScale [8] and reminded in table 2, but only include heptatonic scales and add all harmonic major modes. The resulting listing is shown in table 3. In the latter, the order in which scales appear is justified by the fact that only one note changes between a scale and the next. This may be useful to find a new scale that is close to the current one, as exposed in The Scale Navigator [18].

The chord type concept is a not exactly the same as the chord function (see fig. 2). Instead of characterizing the chord's harmonic role, it roughly describes its core degrees' alteration, which are the third, fifth and seventh. It then allows to group all scales in six subsets, each containing modes that are interchangeable in most harmonic contexts.

type	**scale**
augmented	augmented ionian
	augmented lydian
	super augmented
major	harmonic major
	ionian
	lydian
	lydian ♯9
dominant	mixolydian ♭9
	mixolydian ♭9 ♭13
	mixolydian ♭13
	mixolydian
	lydian dominant
minor	phygian ♭11
	phrygian
	dorian ♭9
	dorian
	dorian ♯11
	minor lydian
	melodic minor
	harmonic minor
	aeolian
half-diminished	altered
	locrian
	locrian 13
	locrian 9 13
	half-diminished
diminished	diatonic diminished
	diminished locrian

Table 3: Scales organised by type

2.2 Instrument Design

The main framework used for this project is MINUET [13]. This paper proposes to define a list of criteria about the goal of the instrument. This process was done early in the design process, and helped clarify many elements, particularly by who this instrument is aimed to be used. At first it was intended to create a perfect instrument for both beginners and advanced musicians, but following

MINUET framework made clear that it was not feasible yet. We decided to focus on musicians that already know music theory, and therefore this interface has a steep learning curve for total beginners. Nonetheless the curve has a very high ceiling, which means that the user may learn continuously for a long period of time, and hardly go around all possibilities. Virtuosity is therefore needed in order to take full advantage of the Theory Controller.

Another very important concept in design process was early testing. Indeed, one of the first iterations was confronted to an external musician, who took time to informally evaluate the system. As it will be rapidly discussed in section 3.2, this proved to be essential for the implementation of multiple new navigation means into the system.

3 Design

The Theory Controller has been prototyped using different platforms and programming languages. Pure Data [15] has been used at first for quick testing, but the current version consists of a JavaScript browser app implemented using p5.js [14], p5js-osc [9] and the Gamepad API [11].

In this section, the technical details of the instrument are presented. First, we will discuss the gesture acquisition and sensor choice. Then, the mapping will be described as well as the general behaviour of the main app. Next, the visual feedback is going to be presented and explained. Finally, we will see how the resulting scale data may be formatted to be used by other apps.

3.1 Gesture Input

For this instrument, different input devices have been considered. The main requirement was to be able to operate it entirely using only one hand. The reason for this is that in most planned applications, the other hand had to be available. Indeed, as the Theory Controller aims to navigate the scale space, it does not produce any sound. In order to play within the selected scale, another device has to be used, such as a Launchpad. More about this in section 4.

As one hand must potentially control the whole system, multiple degrees of freedom (DoF) have to be afforded by a single device. Different solutions exist, such as a joystick or an XY pad, both having both 2 DoF. Nevertheless, the interface calls for extensive controllability, and merely two DoF seem inadequate. As will be discussed in section 3.2, more than two parameters need to be controlled to offer a high learning curve ceiling. Thereby the instrument takes more time to master, but more complex and custom chord progressions are made possible.

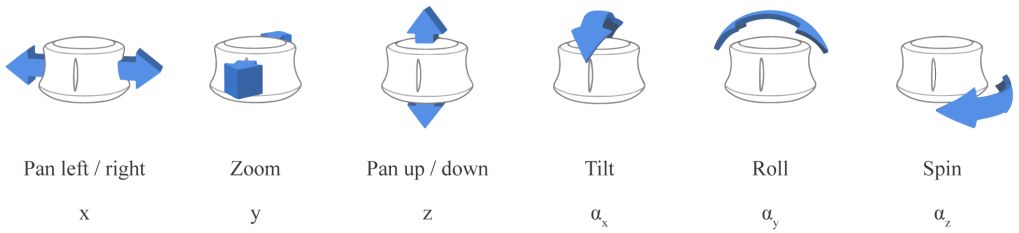

Figure 1: The six SpaceMouse dimensions

The device that appears to meet these requirements is the SpaceMouse from 3Dconnexion (figure 1). It is meant to be used for CAD drawing and offers six DoF (three translations and three rotations). As it is mass-produced, it is not very expensive. Moreover, unlike a generic accelerometer, it is exact, as it measures the rotation angle and the translation displacement relatively to a central position.

The SpaceMouse is recognized as an HID device by any OS. Thanks to the Gamepad API, it can be used directly in the browser.

3.2 Mapping

Scales are organised and accessed across three parameters, the **scale class**, **key** and **degree**. With four scale classes, twelve keys and seven degrees, all scales introduced in section 2.1 are included. Unlike omitted scales (diminished, whole tone or augmented for instance), tonal music mainly borrows chords from heptatonic scales. The conceptual difference between a scale and a chord is blurry, because both are "two forms of the same thing" [10, p. 33]. In this paper, we will consider that a chord is equivalent to a scale, both defined by the tonality to which they belong and the degree from where they begin. When talking about the different scales or chords within a key, we also use the term mode.

The **scale class** parameter ($s \in [0, 3] \subset \mathbb{N}$) represents the main scale family. The numbering is shown in table 4a.

The **key** parameter ($k \in [0, 11] \subset \mathbb{N}$) informs us about the pitch class (pc) on which starts the tonic scale. However, its value is not a pitch class. Keys are organised in fifths and identified as presented in table 4b.

The **degree** parameter ($d \in [0, 6] \subset \mathbb{N}$) corresponds to the note the mode start on, within the chosen key. Changing this parameter does not change the scale notes but relative their order and harmonic function. Changing the degree is equivalent to a chord change within a tonality. As we consider heptatonic keys, there are seven degrees. They are also organised in fifths, as shown in table 4c.

These three parameters have discrete values, and interpolating between those does not make sense. The SpaceMouse is thus used as a trigger rather than a continuous controller.

s	scale class
0	diatonic
1	acoustic
2	harmonic minor
3	harmonic major

k	pc	key
0	1	D♭
1	8	A♭
2	3	E♭
3	10	B♭
4	5	F♭
5	0	C
6	7	G
7	2	D
8	9	A
9	4	E
10	11	B
11	6	F♯

d	degree
0	I
1	V
2	II
3	VI
4	III
5	VII
6	IV

Table 4: The *scale class* (a), *key* (b) and *degree* (c) parameters

gesture	dimension	parameter
rotation	tilt	scale class
	spin	degree
	roll	key
translation	zoom	mode in type
	pan left/right	type
	pan up/down	semitone
click	button	minor third

Table 5: SpaceMouse to scale navigation parameters mapping

The user moves the joystick, and once a specific displacement or rotation value is reached, the parameter is changed, and the user has to go back below a particular value to be able to trigger the parameter change again. The SpaceMouse's three rotation dimensions (tilt, spin and roll, see fig. 1) are mapped to the three scale navigation parameters in a one-to-one manner [5], according to table 5. The motivation for this exact mapping is discussed in section 3.3. All three parameters are cyclic, and multiple parameters may be changed simultaneously.

Besides these three parameters that form the axes of our scale space, other means of navigation are proposed. During early testing and evaluation with a skilled jazz musician, it appeared important to implement shortcuts to quickly access modes that are more than two or three gestures away from the current one. One example is the tritone substitution, which is a common harmonic tool in jazz harmony. Using only the three parameters presented above, the tritone substitution is a six gestures (key changes) movement. Another widely used modal trick is playing in the key a semitone above or under the previous one, and then going back. On the circle of fifths, the semitone is a five steps move. To address these, other SpaceMouse's dimensions have been assigned to bigger key changes. The two

buttons on the sides of the device are used to move the key a **minor third** away (±3 keys), while the pan up/down (see fig. 1) dimension is mapped to a **semitone** harmonic movement (±5 keys). The tritone substitution is now a matter of two clicks on any of the two side buttons, because two minor thirds make a tritone.

Despite all this, some basic chord changes still appeared uncomfortable using our system. A typical example is the V/II-II-V-I progression in a major key. Even though it is a very common progression, V/II and II are far apart from each other. The issue that is revealed here is that replacing a scale by another one that has the same root note but is of a different type is not an easily afforded substitution. In the example above, the simple progression in our system would be VI-II-V-I, where VI is a minor chord. However, replacing the latter by a dominant chord is not intuitive. It became clear that we had to include a different kind of navigation, putting the chordal type at the center. This opens two new navigation dimensions, which are the **type** and the **mode in type**. The first one is simply one of the six types shown if table 2, while the second one corresponds to which scale we select within that type. This last parameter is a bit different than the others, as its maximum value depends on another parameter, the type. Indeed, some types contain more scales than others. The mapping for these two new parameters is also detailed in table 5.

An interesting feature to notice is that we now have two independent ways of accessing the same 336 scales. Using the rotation gestures on the SpaceMouse, scales may be accessed in a symmetrical manner, where navigation is closely related to how the scales are constructed. The degree and key parameters are stacking fifths, and all three (with scale class) have a fixed maximal value. On the other hand, translation gestures are more useful in real world context, where for example you would quickly want replace a scale by another one that fits the same chord, or change a chord's type to modulate. The rotation navigation requires to deeply understand the mathematical structure of music theory, and think in advance before doing modulations. The translation navigation works on the current chord, and affords shortcuts that would be highly counter intuitive without it. Given all this, we now have a strong many-to-many mapping.

The app is implemented in JavaScript using p5.js [14]. The communication between the app and any other app is done using the OSC protocol [3]. While MIDI has been considered an option, OSC has proved to suit our needs better, each OSC message allowing as many integers as needed to communicate all notes of a scale. More on this in section 3.4.

The Theory Controller app is accessible online [7]. It may be tested without a SpaceMouse, using the buttons on the bottom right corner for the three main parameters, and the four arrow keys for the type and mode in type navigation. The semitone and minor third shortcuts are not currently available without a SpaceMouse.

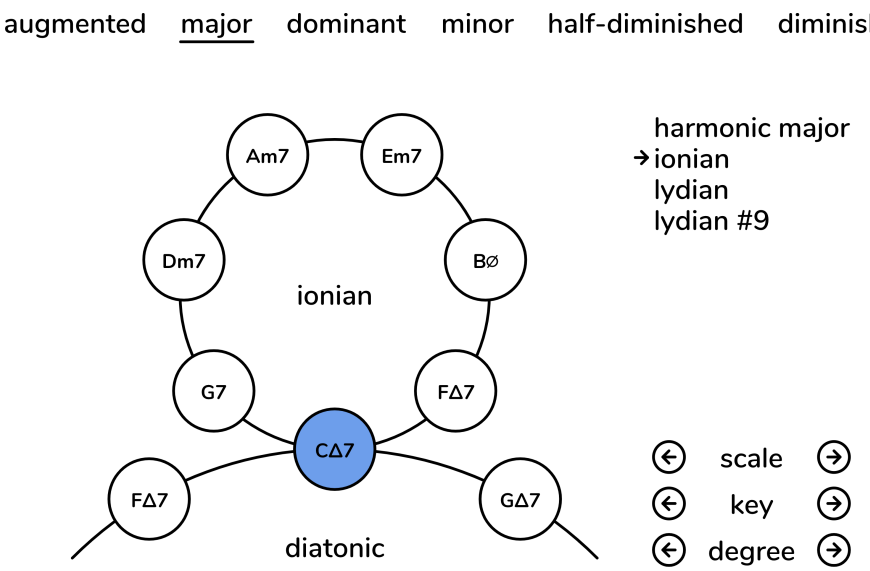

Figure 2: The Theory Controller visual interface

3.3 Visual Interface

Visual feedback is a central part of the system. As we deal with theoretical concepts, representing them in a relevant geometry is essential to make the instrument as intuitive as possible. At the same time, the system aims to be comprehensive, so all standard scales have to be accessible as neutrally as possible.

As our target user will have a basic understanding music theory, we have decided to build the visual design upon the circle of fifths, a widespread tool in composition and teaching of harmony. However, to avoid overfilling, only a quarter of the circle is visible on the screen. The current key is visually placed centre screen. The key a fifth above is placed to the right, and the one a fifth below to the left. A snapshot of the visual interface (set in C major) is presented in figure 2. Each scale is identified by its corresponding seventh chord, written in a tiny circle. The current key is always on top, and opens up in a tangent smaller circle that contains the seven modes of the selected key. Under the key is written the scale class (diatonic in fig. 2), and in the modes (or degrees) circle is written the mode (Ionian in fig. 2). The degree circle is itself organised in fifths. The tonic scale (or chord) is at the bottom, so it is simultaneously on the degrees circle and on the keys circle. At its left (clockwise on the degrees circle) is the dominant chord (V, a fifth higher), then the II, and so on, according to table 4c. This geometry affords simple II-V-I progressions as they appear consecutively on the modes circle.

When changing the key, an animation closes the modes circle inside its key scale, opens the new key's modes circle, and at the same time rotates everything so that the newly selected key and its modes circle are at the top. A snapshot is shown in figure 3a. It is quite a complex animation that helps the user understand that all seven modes

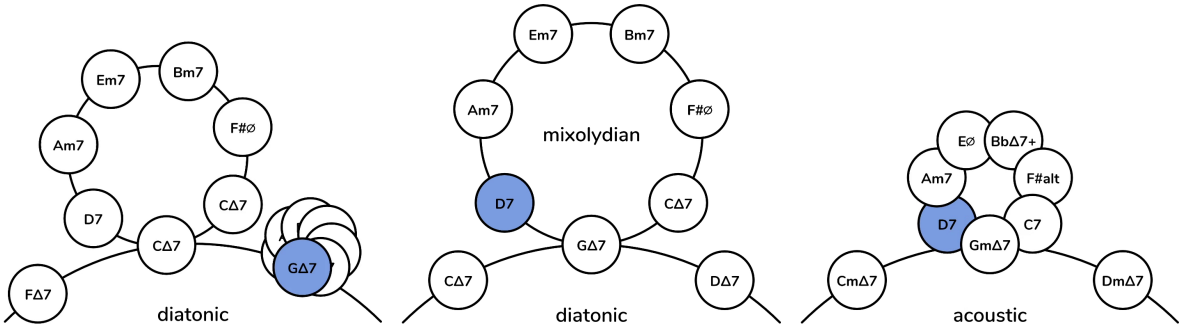

Figure 3: Key change (a), degree change (b) and scale class change (c)

are related to one key or derived from it. Changing the key offers seven new scales, so if everything was shown, we should have twelve modes circles around a vast keys circle of fifths. The animation helps keep the representation compact while still implying this idea. When changing the degree, the blue circle simply moves around the modes circle (fig. 3b). It may hard to imagine what these animations look like. You may either watch a quick demonstration on the related blog post [6] or directly test the app yourself [7].

The mapping of the key and degree parameters (tab. 5) makes sense using this representation. Indeed, the roll (fig. 1) gesture may intuitively be used to rotate the big keys circle because, with the SpaceMouse, this movement is centered around a position at the top. After the user rolls the joystick, it automatically returns to its initial position. On the other hand, the blue circle moves around the degrees circle. The spin (fig. 1) gesture makes more sense as the entire joystick moves around the Z-axis.

Finally, the scale class change is done using the tilt (fig. 1) gesture, which goes beyond the 2D plane in which those circles are drawn. It opens up a new set of $12 \cdot 7 = 84$ scales. The animation closes the degrees circle and opens it again, but the keys circle does not move. See figure 3c.

As said in section 3.2, the interface offers a way of navigating across these three parameters without a SpaceMouse, using arrow buttons in the bottom right corner (see fig. 2). Six buttons are organised in three rows, one for each parameter. Left arrow buttons decrement the parameter's value, while right arrow buttons increment them.

On top of the screen, the six types are written in line, which justifies the use of the horizontal arrow keys to change it. The type of the current chord / scale is underlined. Above, on the right, the modes sharing this type (tab. 3) are listed in a column, which explains the use of the vertical arrow keys to navigate. This list is dynamic, when selecting a scale from another type, the list is replaced by the modes of the new type. The current mode is shown with a small arrow to its left (see fig. 2).

3.4 Scale Data

As we will discuss in section 4, to take full advantage of this system, we may use it in conjunction with another app that makes something with the selected scale. In fact, the Theory Controller's true purpose is to generate scale data, according to what the user selects.

At first, the MIDI protocol was considered an option to transport the scale information. It may be an intuitive choice, as it is a widely used music transfer protocol. However, sending a heptatonic scale with MIDI notes means sending multiple message, in a given order. One lost message may mess up the entire scale. Moreover, in addition to the pitch class information, the note degree has to be communicated as well, to comply with the theory introduced in AutoScale [8]. The velocity has been considered an option to encode the degree, but it seemed to be a misusage of MIDI, too far from its initial aim. MIDI SysEx [12] would have been a better option, but OSC [3] appeared quite convenient, with its versatility and ease of implementation.

A scale data is then transported by a single OSC message, containing the address */scale* and two integer per scale note, its absolute degree ($ad \in [0, 6] \subset \mathbb{N}$, C = C♯ = 0, D = D♭ = D♯ = 1, ...) followed by its pitch class ($pc \in [0, 11] \subset \mathbb{N}$, C = 0, C♯ = D♭ = 1, ...). The generic heptatonic scale is thus encoded as follow:

$$/\text{scale } \text{ad}_\text{i}\ \text{pc}_\text{i}\ \text{ad}_\text{ii}\ \text{pc}_\text{ii}\ \text{ad}_\text{iii}\ \text{pc}_\text{iii}\ \text{ad}_\text{iv}\ \text{pc}_\text{iv}\ \text{ad}_\text{v}\ \text{pc}_\text{v}\ \text{ad}_\text{vi}\ \text{pc}_\text{vi}\ \text{ad}_\text{vii}\ \text{pc}_\text{vii}$$

Here is as example the OSC message generated for a C Ionian scale:

$$/\text{scale } 0\ 0\ 1\ 2\ 2\ 4\ 3\ 5\ 4\ 7\ 5\ 9\ 6\ 11$$

A D Dorian scale, which contains the same notes in a different order, is encoded as:

$$/\text{scale } 1\ 2\ 2\ 4\ 3\ 5\ 4\ 7\ 5\ 9\ 6\ 11\ 0\ 0$$

As you can see, the order in which notes (pair of integers) appear in the message is important, as it indicates the notes' order in the scale. The scale's root note is always first.

As we use heptatonic scales only, only the first degree would be necessary, as we basically have one note per degree. However, to make the app future-proof, the degree information is primordial for any other kind of scale. In the altered dominant scale, for instance, there are two second degrees, one minor and one augmented, while there are no fifth [10, p. 70-72]. This very concept would be impossible to communicate without the degree information for each note, thus erasing the difference between the half-diminished and the dominant versions of the altered scale. In octatonic scales, the eighth note has also a degree that must be communicated in order to fully determine the scale's type and nature.

4 Application

The system as it is may be a satisfying contraption for harmony nerds, but it has no musical use. In this section we will cover different use cases that have been explored by the author. In fact, the Theory Controller aims to be a versatile tool that adapts to many situations. Among navigating the scale space alone and jamming together, the possible applications are diverse and numerous.

4.1 Scale Navigation

A primary application is scale navigation. As they are organised in a relevant manner, discovering where chords are situated may help learning underlying structures of music theory. For this purpose, the app may be used without the SpaceMouse (see section 3.2), as the latter enhances the interactivity for performance settings.

The interface itself does not clearly expose the selected scale exact composition. In order to see what notes populate a scale, another app may be used in conjunction with the first, using the OSC based scale data format (see section 3.4) to communicate from one to another. Such a complementary app would plot the scale notes on a staff, a guitar fretboard, or a piano for example. However, we did not implement any of these in this project. What scale representation we did use is the chromatic circle, which we will discuss further in section 4.4.

When using a scale representation synchronised with the Theory Controller, the system may be used in education. Indeed, when teaching scales and chords, instead of showing static images of scores or the circle of fifths, one could take advantage of the dynamism of our app to illustrate a course when speaking. In instrument teaching, the professor always shows concepts in the instrument. The same could be done in music theory classes, instead of using a piano, which is often the case. Most instruments do not remotely resemble the piano keyboard. Our representation seems to offer a more neutral way of showing some musical concepts.

4.2 One Pad Chord

The first sound-producing application that was tested is the one pad chord. As indicated by its name, this setup consists of a singe MIDI pad, mapped to trigger the selected chord. It is a straightforward but also limited way of making music with the Theory Controller.

This system may be implemented in different manner, but what we propose here is using Pure Data. The patch and its two subpatches are shown in figure 4. In the latter, you can see that the scale data is received from a server in OSC format. It is then unpacked in fourteen numbers (two per note, see section 3.4). Only i-iii-v-vii tetrads are

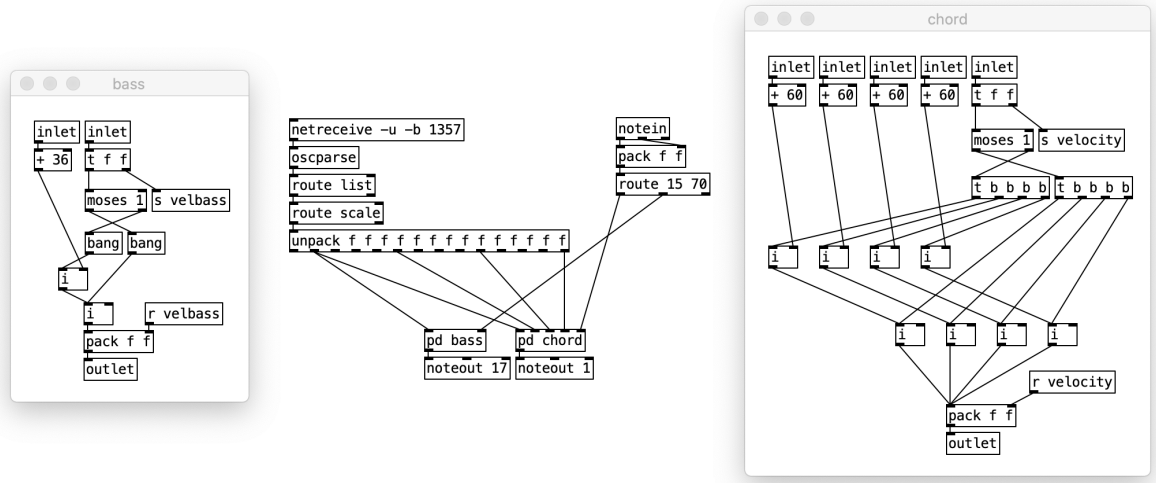

Figure 4: Pure Data patch for the one pad chord application

extracted from the scale, and only the pitch classes are relevant, as we generate MIDI messages. The root note is used to generate a bass message an octave lower that the rest. The four notes are used to generate the chord messages. Both the bass and the chord notes are stored in subpatches, in order to send the correct note off messages even if the scale has changed, and thus avoid hanging notes.

The generated MIDI messages may be used to trigger any kind of melodic instrument, virtual or physical (MIDI compatible of course). The bass and chord buses should be routed to different instrument, in order to obtain the best suited sound. This application could be enhanced in many different ways. Chord inversions may be a interesting thing to implement, leading to multiple pads triggering different versions of the same chord.

This application is suitable for musical performance, but it is also ideal for scale navigation, as it affords a quick hearing of the selected chord. Is does however not allows you to listen to the whole scale, as stacking the seven notes in thirds would be less usable in most context. The number of stacked thirds could also be an optional parameter in the one chord pad application, that could let the user define the extension complexity she wants to hear. But being able to recognise a whole scale when played all notes at once is not innate in everyone.

4.3 Arpeggiator

Another use case for the Theory Controller is an arpeggiator. Instead of playing all notes at the same time as with the one pad chord application, one could implement an arpeggiator receiving the OSC scale data and iterating over a melodic pattern built within the selected scale. That way, the user would only take care of the chord choice and let

the system play notes.

The main musical limitation of this system is the fact that the pattern has to be determined in advanced. If there is no way of making it evolve during the play, it quickly becomes repetitive. However, the arpeggiator could be enhanced in a jazz voicing manner, such as the automatic backing track algorithm found in iReal Pro [17]. Using such a voicing generator in conjunction with the Theory Controller could be very relevant for jazz musicking, or composition.

Another issue with the automatic melodic pattern generator is the synchronisation. Changing from one scale to another in our system may take some time, and unwanted transitioning scales could be heard. To address this, three solutions are suggested:

- Having a silence in the melodic pattern at the end of each bar, leaving time for the user to change scale
- Using a button to validate the selected scale, remapping one of the SpaceMouse's buttons, using a MIDI controller or a computer key
- Implementing an intelligent scale validation within the arpeggiator, where any incoming scale would only update the pattern at the end of a bar

This last solution seems to be the most useful in most musical contexts, but it requires a clock signal and some beat information in order to locate the transition from a bar to another. It may be relevant to implement this in the future as a plugin in a DAW, the latter giving the clock information according to the defined BPM.

4.4 AutoScale

AutoScale [8] was the main motivation for the Theory Controller. It is a music improvisation system that synchronises a dynamic MIDI scale mapper with a predefined chord progression. Here, instead of using a chord progression to deduce scales, we use the Theory Controller to configure the scale mapper. As the AutoScale web app has a chromatic circle visual interface, it may be used to visualise the selected scale, as mentioned in section 4.1. But it also enables MIDI scale mapping within the browser, and color coding on the Launchpad Pro. The user can then freely improvise in the selected scale with one hand on the Launchpad, and change the scale with the other hand on the SpaceMouse. Figure 5 displays the two apps synced together. As you may deduce, AutoScale's chromatic circle shows C on top, and increases the pitch class for every $\pi/6$ angle clockwise. It also has a color code highlighting the root note, third, fifth and seventh of the scale. More on this in the dedicated paper [8].

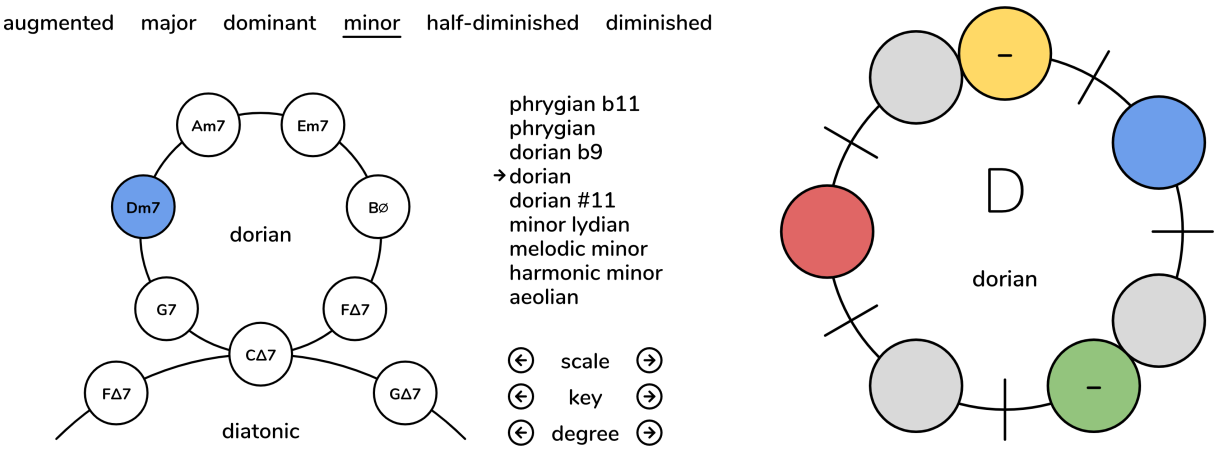

Figure 5: AutoScale used in conjunction with the Theory Controller

This application is way more versatile than the other presented, as all the notes are at hand, and the user is able to play chords, melodic pattern, make them evolve, and choose whatever voicing he wants. The system becomes a full instrument that can be used in any settings.

In this case, the Theory Controller acts as the harp's pedals, while the Launchpad offers the scale notes, as the harp's string. Like the harp, and unlike almost all other instruments, there is a scale dissociation happening. Not all twelve notes are accessible at the same time, as it is impossible to play out of scale. However, the musician has to choose a scale, and some chord progressions sound better than others.

Of course, as the Theory Controller broadcasts OSC data, multiple applications may be used at the same time, syncing many musicians together. A demonstration is available on the related blog post showing both the one pas chord (sec. 4.2) and AutoScale being used in conjunction with the Theory Controller [6].

5 Reflection

We will now discuss some design flaws, observations and envisioned further development of this project. The system was tested with an external musician during its development, however a larger scale user testing will bring more in depth analysis of the quality of the interface. Nonetheless, the author has been able to test the instrument himself a lot and draw important conclusions.

5.1 Design

A balance had to be found between a totally new representation and more traditional music theory concepts. Indeed, even if we wanted to propose an optimised scale space

visualisation, forcing experimented musicians to relearn things they already use may not be a good option to invite them to try out the system. For instance, partially representing scales by a corresponding chord name appeared to be an efficient way of giving an overview of the available modes in a scale class. Most musicians recognise those chord names instantaneously. Only showing modes of the current key, or scales of the current type, are compromises we made to offer a clean interface. Writing six columns of modes and drawing twelve circles around a huge circle of fifths would have been visually too aggressive. To compensate for this simplification, the key change animation appeared to be helpful, as it shows the user she is moving around the big key circle. The downside of it is that it may be hard to grasp the number of possible scales in the system. Only nine are shown, which leaves 327 hidden scales. The only way to figure out the size of the scale space is to navigate it in all directions, and it may take some time.

The main research question introduced in section 1 opens a reflection about which music theory parameters should be controlled in order to afford intuitive scale navigation. During iterations, more dimensions were added, and a many-to-many mapping emerged naturally. The fact that the same chord progression may be achieved in multiple distinct manner is an important feature of the design. Depending on the kind of harmonic progression, the user can select a navigation method, and combine them. Even more shortcuts could be added. Music is way more complex and comprehensive than what a simple circle of fifth could explain alone. It is now clear that being able to change scale both by modulating (circle of fifths) and by using modal interchange (mode in type) is essential to afford the fastest and the most intuitive music theory control.

5.2 User experience

Using the Theory Controller in the intended manner requires a deep awareness of the chord progression. It is a very rational approach of music, and it may not suit everybody's way of thinking. At first, the system aimed to be an intuitive interface for beginners, affording complex harmonic movements with no music prerequisites. It was supposed to be easy to understand yet comprehensive in its features, in order to maximise the target user pool. Nevertheless, while adding more options and functionalities to the system, it shifted to this highly complex interface, with a lot of elements present on screen. Instead of a purely geometrical and color coded representation, many concepts are written in plain text. This makes the usage of the software rely on previously acquired knowledge.

The ideal music theory interactive system would have the same amount of controllability and versatility, while being self-explanatory enough to avoid using any music convention. Such an instrument would have a gentler learning curve (low floor) and thus be appropriate for a larger user pool. An envisioned manner of implementing such a

system is to have some kind of scale proposition based on the current one. In a way, any movement would lead to a coherent chord progression. Proposed following scales could be prioritized by probability of presence, using a machine learning model trained on a huge chord charts corpus data, such as iReal Pro by Technimo. A similar algorithm already powers Klimper by Polydigm. However, the risk with this approach is to favor common chord progressions over others, thus impairing creativity. The Theory Controller focuses on scale neutrality, trying hard to expose each scale class, key and mode as equally as possible. The choice is completely up to the user.

5.3 Contribution

You may wonder, if the target user is an experienced musician knowing music theory, why bother learning this new musical interaction instead of using his already mastered instrument. Here are two reasons why. First, a lot of instruments are not harmony capable. These include for instance bowed string instruments, most wind instruments and some percussive instruments. Still, musicians playing those may know about chords and be frustrated of not being able to fully explore and play them. Traditionally, such cases require learning the piano or the guitar. The Theory Controller aims to be a new solution for this particular situation. Indeed, it is a good option if you know the theory, but do not want to spend the time learning scale patterns on the piano keyboard or chord positions on the guitar fretboard. You can quickly hear and compose chord progressions using this system. Secondly, in a performance context, the Theory Controller is only part of the instrument, as discussed in section 4. If you're a music nerd, you may encounter a situation where you want to jam with other musicians that were not as diligent as you during theory lessons. You can try to expose the exact composition of each chord and scale you want to use, but it is time consuming. Using instruments synced with the Theory Controller, you can focus on the chord progression while other musicians jam freely in the scale you selected. This scale syncing is a new paradigm for harmonically advanced jamming.

5.4 Future

Playing with others using scale syncing requires either to plug multiple MIDI controllers into the same computer, or to use a local network and multiple computers connected together. Given the networking nature of OSC, it may be envisioned to connect musicians non-locally, in a telematic musicking setting. It requires the use of a low latency audio streaming system, such as LoLa [2] or JamKazam [19] to name a few. Both audio and music theory could then be synced in real time, thus affording longer distance jamming using the Theory Controller. To do so, either port forwarding or the use of a VPN is

needed in order to ensure the OSC communication.

Other chord and scale related pieces of software exist, most of which are audio and MIDI plugins. For example, Captain Chords by Mixed In Key, Scaler by Plugin Boutique or AutoTheory by Mozaic Beats are chord and scale generator plugins that are already available. However, most of them are meant to be used for basic chord progression composition, and are not suitable for jazz live performance. Moreover, they are closed, in the sense that no universal scale communication protocol is proposed. We bring here this new concept of scale syncing, that could, if adopted in a similar manner as the MIDI standard, make all these software programs communicate with each other. Instead of reinventing the wheel every time, including a chord generator, MIDI scale mapper, scale representation, sequencer and so on in the same plugin, we could imagine in the future having these brands focusing on a specific aspect, and create a music theory plugin ecosystem together. The Theory Controller aims to explore this opportunity.

6 Conclusion

This project has been an opportunity to explore a totally new way of interacting with music. As we have seen in section 5.3, no other interface is really similar to this one, so it is hard to compare it to anything existing. The simple fact that it doesn't produce sound directly makes it stand appart from other musical interface. The research was not extensive enough to derive a clear framework for music theory enhanced instruments, but some observations might be useful for anyone trying to achieve something similar. One thing is certain, we do not have a perfect mapping strategy yet. Even without a proper large evaluation, it is quite sure that the inaccessibility of the system for most novices makes it improvable.

The small networked ecosystem developed here gives a glimpse of what could be feasible soon with a little theory coordination between creators. There are so many music producer that do not want to spend the time learning chords and scales, but the rise of chord generator plugins reveals a crave for interesting progressions. Perhaps the approach of a total theory controllability presented in this paper may inspire musical technologists and digital musicologists to further explore this interaction paradigm. And hopefully developers will specialize in different aspects of real time music theory software plugins. Anyway, the journey that the author began with AutoScale and continued with the Theory Controller is far from finished. Plethora of interactions with the scale space remain to be considered and implemented.

References

[1] W. Apel. *The Harvard dictionary of music*, volume 16. Harvard University Press, 2003.

[2] C. Drioli, C. Allocchio, and N. Buso. Networked performances and natural interaction via lola: Low latency high quality a/v streaming system. In *International Conference on Information Technologies for Performing Arts, Media Access, and Entertainment*, pages 240–250. Springer, 2013.

[3] A. Freed. Open sound control: A new protocol for communicating with sound synthesizers. In *International Computer Music Conference (ICMC)*, 1997.

[4] R. F. Goldman. *Harmony in Western music*. W. W. Norton & Company, 1965.

[5] A. Hunt and R. Kirk. Mapping strategies for musical performance. *Trends in gestural control of music*, 21:231–258, 2000.

[6] T. Jaccard. Blog post. https://mct-master.github.io/master-thesis/2021/06/03/theory-controller.html.

[7] T. Jaccard. Theory controller. https://thiblejack.ch/theory-controller/.

[8] T. Jaccard, R. Lieck, and M. Rohrmeier. Autoscale: Automatic and dynamic scale selection for live jazz improvisation. In *NIME*, pages 423–427, 2020.

[9] G. Kogan. p5js-osc. https://github.com/genekogan/p5js-osc.

[10] M. Levine. *The jazz theory book*. O'Reilly Media, Inc., 2011.

[11] C. McAnlis, P. Lubbers, B. Jones, D. Tebbs, A. Manzur, S. Bennett, F. d'Erfurth, B. Garcia, S. Lin, I. Popelyshev, et al. Playing around with the gamepad api. In *HTML5 Game Development Insights*, pages 163–175. Springer, 2014.

[12] R. A. Moog. Midi: Musical instrument digital interface. *Journal of the Audio Engineering Society*, 34(5):394–404, 1986.

[13] F. Morreale and A. De Angeli. Musical interface design: An experience-oriented framework. In *NIME*, pages 467–472, 2014.

[14] Processing. p5.js. https://p5js.org/.

[15] M. S. Puckette et al. Pure data. In *ICMC*, 1997.

[16] I. Ring. A study of scales. https://ianring.com/musictheory/scales/.

[17] Technimo. ireal pro. https://www.irealpro.com/.

[18] N. Turczan, A. Kapur, N. Ho, C. Honigman, and D. Shepherd. The scale navigator: A system for networked algorithmic harmony. In *NIME*, pages 101–104, 2019.

[19] D. Wilson. Jamkazam. https://jamkazam.com/.