Multimedia Tools and Applications (2023) 82:4749–4771
https://doi.org/10.1007/s11042-022-13372-3

A rhythm-aware serious game for social interaction

Filippo Carnovalini[1] · Antonio Rodà[1] · Paolo Caneva[2]

Received: 13 April 2021 / Revised: 17 February 2022 / Accepted: 15 May 2022 /
Published online: 12 July 2022

Abstract

Making music with others is both an artistic act and a social activity. Music therapists can leverage the social aspects of music to increase the well-being of their patients by interacting with them musically, improvising rhythms and melodies together on shared musical instruments. This activity requires highly trained professionals and is therefore expensive for the clients. We propose a serious game that can help people without musical training interact by collaboratively creating a rhythm using MIDI drum pads. The gaming system analyzes the rhythm in real-time and adds musical feedback that is synchronized to what the users play, enhancing the aesthetical experience that is crucial to the musical interaction and its therapeutic effects. We assessed our system through quantitative metrics showing its capability of following a user-established tempo. Test players also completed a questionnaire, which showed they found the experience pleasant and engaging, and that the musical augmentation was helpful to their interaction.

Keywords Serious games · Sound and music computing · Music generation · Tempo detection · Human-computer interaction

1 Introduction

Music can be played alone, and solo pieces are often valuable parts of any virtuoso's repertoire. Despite this, music playing is mainly a collaborative activity. Music students spend a lot of time exercising in groups and orchestras, knowing that keeping a shared tempo and shared musical expression is a fundamental skill for a professional musician. When there is no written score, as with jazz improvisation, the interactivity reaches another dimension: each musician needs to pay close attention to what the others are playing to keep the tempo

✉ Filippo Carnovalini
filippo.carnovalini@dei.unipd.it

[1] Università degli Studi di Padova Computational Sonology Centre, via Giovanni Gradenigo 6, Padova, Italy

[2] Verona Conservatory of Music, Department of Music Therapy, via Abramo Massalongo 2, Verona, Italy

and make meaningful contributions to the session. Musical and social interactions are fundamental in music-making for non-professionals as well: even a garage band of teenagers requires the members to collaborate both when they play and when they stop if they want to make music effectively [29].

Interaction also leads to educational benefits, strengthening social skills and self-esteem, and giving musicians a sense of belonging as well as higher satisfaction [25]. The emotional value of music is also a critical factor in the emergence of social benefits [28] and music can be helpful even with subjects with social difficulties, for example, those suffering from autism spectrum disorders (ASD), that can express feelings through music more effectively than with words [4, 36]. While playing in groups can realize these social benefits even if the involved players are not professional-grade musicians, reaching an effective interaction and pleasing musical results still requires months (and sometimes years) of practice with an instrument. Not everybody wishes to spend that much time learning how to play, and some cannot even hope to achieve such technical abilities, for example, due to motor impairments.

Professional music therapists can help their clients obtain these benefits despite the lack of training [9]. One of the tools available to music therapists is that of improvisation. Even if the client is not a musician, therapists can share musical instruments and create music and sounds together. Alternatively, the client could play a simpler instrument, like a drum, that can be played by anybody to some extent, while the therapist contributes to the improvisation using more complex instruments, such as a piano or a guitar, creating music that follows what the client is doing but is more elaborate and pleasing. Doing so requires the help of a highly trained professional and is therefore hard to practice as a daily and inexpensive activity. Moreover, when applied to children, these improvisation sessions are usually a dialogue between the (adult) therapist and a single child. The therapist could interact with two or more children, but it would be impossible to have a “peer to peer” improvisation between children alone. One option towards this goal would be reducing the quality of the music, which is normally controlled by the therapist. This would mean, for example, using only drums, which are less demanding in terms of musical training. The result would be a rhythmic improvisation, devoid of melodic and harmonic material, but an improvisation nonetheless. This could seemingly be a good compromise since the interactivity is maintained, which is one of the principal features of music therapy. However, the aesthetic value of the interaction is a crucial factor for the therapeutic effects of a musical dialogue [3, 45], and should therefore not be overlooked.

In this paper, we propose a gaming system inspired by this kind of musical interaction: it requires two players that collaboratively create a rhythm, having the experience of creating a musical interaction, while the system ‘augments’ the interaction by adding a chord progression and melody. The game is meant to recreate the social benefits of group music-making and music therapy sessions, even for people that do not have musical expertise nor training. We propose this as a serious game since the goal of the game goes beyond mere entertainment [38]. In this sense, this contribution falls within the intersection of technology, music, and health/well-being. Using digital technologies to facilitate music therapy interactions and create personally-tailored experiences is a field that has seen a rise in interest lately but still has significant unexplored territory and potentially profitable interdisciplinary approaches [2]. In our case, the game can be seen as a way in which technology can use music to empower patients to reach therapeutic goals.

Regardless of these potential benefits, from the player’s perspective, the game’s objective is the interaction itself and the two players must collaborate rather than compete to obtain more rewarding feedback, both in the form of "points" seen on screen and with more pleasing musical augmentation. Having a higher aesthetic value in the musical output should give a more engaging experience to users and incentivize better interactions between the players.

1.1 Contributions

The main contribution of this paper is the description of a software system that detects and follows a user-established rhythm. The techniques that we describe here could be useful to other interactive multimedia applications, as the user-defined rhythm represents a novel kind of human-computer interaction. This is different from other already existing systems (such as those described in the next section) that use rhythm but generally impose a predetermined rhythm to the users. Audio games could also integrate this approach as a further way to give players feedback for their gaming performances and to spur the players to play more.

We present the rhythm following system embedded in the aforementioned serious game. While it takes inspiration from music therapy in its design, the goal is not to recreate a music therapy session with a serious game, but rather to create a game that leverages the benefits of music in a similar way to those that are commonly used by music therapists. This represents a first example of how musical augmentation can improve a multimedia application, in this case a game, to make it more engaging. The code for the presented software is openly available at the following address: https://gitlab.dei.unipd.it/facoch/sympaddy/

This article has the goal of presenting the project and its potential applications, describing the technical implementation and evaluating its ability to detect and follow a rhythm in real-time. This work also includes a questionnaire to assess whether the game is found to be engaging by the first testers, but other aspects of user interaction are not assessed here, and a longer-term evaluation is needed to precisely assess the social benefits that can result from the use of this game. These further assessments will be left for future studies, while this paper will exhaust the technical aspects of the system to allow others to reuse and re-implement the ideas described here.

1.2 Related works

Music is a fundamental element of multimedia applications, but systems that use music and rhythm as the main element of interaction are rarer. Some notable examples are commercially acclaimed video games that focus on rhythm.

The most common way in which rhythm is used as a game mechanic is to require the player to perform a series of actions at the right moment, following a predetermined sequence that is shown to the player aligned with a musical soundtrack. The actions can vary from dance moves (as in *Dance Dance Revolution* (Konami, 1998) or *Just Dance* (Ubisoft, 2009) or clicking buttons on a keyboard or touchscreen (*Frets on Fire* (Unreal Vodoo, 2002) or on special controllers (usually shaped like musical instruments, like guitars in *Guitar Hero* (Harmonix Music Systems/RedOctane, 2005), drums in *Rock Band* (Harmonix Music Systems/MTV Games, 2007) or *Taiko no Tatsujin* (Namco, 2001) or even DJ consoles in *Beatmania* (Konami, 1997). In all of these games, the rhythm is predefined and follows the chosen songs. The player can influence the musical output as making correct moves or mistakes can trigger visual and aural feedback, but the song does not account for improvisation nor adapts to the player's performance. Other games have made different use of rhythm by forcing a rhythm on otherwise “normal” game mechanics. *Otocki* (SEDIC/ASCII Corporation, 1987) is a shooter game and one of the first games to have a procedurally generated soundtrack. Each time the player shoots a bullet, a note is randomly generated using rules that ensure the note fits the background accompaniment. Since the notes must fall on beats of the background music, the bullets are also shot only on those beats, resulting in delayed shots if the player does not follow the rhythm. An even stricter imposition of rhythm is found in *Crypt of the NecroDancer* (Brace Yourself Games/Klei Entertainment, 2015), a fast-paced turn-based RPG where turns are tied to the soundtrack’s beats, meaning that if the player fails to act within a beat, the turn is lost. An interesting feature of this game is that the soundtrack is fully customizable by the user: a tempo and beat detection algorithm is used to align the gameplay to any music file the user selects from his personal library. This feature makes the game adaptive to the user selection, but once the music is selected it is still the player that must follow the rhythm of the game and not vice-versa as we wish to do with our system.

Rhythm is also used in academic research to create serious games. One application is to use rhythm games to teach the sense of rhythm, which is usually learned early in infancy and strongly influences auditory processing and speech learning [40]. In particular, infants with hearing impairments or neurological disorders do not learn rhythm as effectively as others. This makes the use of serious games that help to build a rhythmic sense useful to support their educational and neurological development. One such game is *Rhythm Workers* [6], where the player must either tap a touchscreen following the beat of a song or recognize whether a percussive sound is aligned to the background music’s beat. This game is designed for children with neurological impairments but requires good hearing since the rhythm is presented through sound. Another mobile game called *El Misterio de Armonisia* [34] uses both sound and visual cues as it is intended for children who have hearing impairments. The player must tap certain areas in the touchscreen when a symbol enters those areas, and the symbol’s movements are aligned to the musical cues, similarly to more famous games like *Piano Tiles* (Umoni Studio, 2014).

Another context in which rhythm games were used in serious context is that of motor rehabilitation. Rehabilitation of fine movements requires repeating certain movements many times, a task that can quickly become tiring making the patients less willing to do the prescribed exercises, hindering the benefits of the therapy [22]. One serious game with the goal of overcoming this difficulty [43] uses the Leap Motion Controller, a sensor made of two infrared cameras, to detect hand gestures and use them as input for a rhythm game that is similar to many games described above, but in which the notes are triggered by gestures. The presence of the music (to which the notes are aligned) and the challenge of the game makes the experience more pleasing and less tiresome. Other similar proposals using other motion capture systems exist in the literature [1].

All the applications described above are games that use some kind of prerecorded music to which the gameplay is aligned, rather than allowing the players to improvise and use their own pace. One application that is more similar to ours is *D-Jogger* [30], which detects the user’s walking or running pace and warps the music they are listening to make the musical tempo match the movement pace. This application has the goal of making an exercise session more pleasant by matching the tempo of the music with the user’s pace, making it similar to our therapeutic goal. Despite the similarity, the strong simplification of the rhythmic features considered (i.e., time between steps) in their system makes the tempo detection algorithm they implemented not useful for a music improvisation application. One application that deals with full improvisation instead is *Wiimprovisation* [5], which uses Wii controllers to allow the players to create music collaboratively. In this case, there is no background music, but the players can move their controllers as if they were drumsticks to produce sounds. A few different sounds are available by pressing different buttons on the controllers, and different instruments can be selected to allow the players to use different sets of sounds. The serious goal of this game is to provide mediation and improve social interactions in children suffering from behavioral disorders, making this proposal the most similar to ours. The main difference, in this case, is that the music is entirely generated by the players, who have access to very limited sounds and expressive potentialities, making the resulting music arguably less aesthetically pleasing than the augmentation we propose in this paper. We are not aware of other works in literature that have the aim of augmenting a rhythmic improvisation in the context of a serious game to make it more aesthetically pleasing.

Without considering the gaming aspects, the main technical requirement of the system we propose is being able to understand and follow a rhythm established by its users, synchronizing musical events generated by the software with those performed by the players.

Many works within Music Information Retrieval have focused on detecting rhythm meant as tempo and meter. In that case, there is no interest in retrieving such information in real-time [20, 21, 24, 50]. Requiring the system to work in real-time is instead more frequently considered by research relating to score-following and automatic accompaniment [19, 31, 37].

In particular, those works that focus on following a human improvisation best cover our requirements, but often make assumptions on the input performance, requiring some prior knowledge before the improvisation starts. One example is the MIDI Accompanist by Toiviainen [47] that uses oscillators to adapt to a non-predefined beat. This work has a complicated mathematical basis, and it only focuses on the adaptation to a beat, not considering the meter. Beatback [26] offers another approach to follow drummers in their improvisation, but the synchronization of the system is based on having the tempo decided before the improvisation, thus making it not adaptive. A similar approach is that of B-keeper [39], which analyzes the audio of a kick drum to perform synchronization, but again requires the prior knowledge of the approximate tempo and does not consider meter. The improvisation follower by Xia and Dannenberg [52] is instead based on melody but is limited to improvisations over a given melody, and the system needs to be trained over examples of the target accompaniment. Other famous systems that are capable of augmenting human improvisation include Pachet's *Continuator* [32] and Biles' *GenJam* [7], but both systems require a fixed tempo (GenJam also requires a chord progression and a scaffold of the song structure), and the musical augmentation works by exchanging musical solos with the user, and not giving a contemporary musical background to the user improvisation.

2 Architecture

The system's architecture is divided into three main parts, which reflect the three main aspects of the game. A game starts without any music, and the two players must start playing the MIDI pads that constitute the gaming system's input. In the beginning, they can only hear the sound they make with the percussion. As the game goes on, the system evaluates how well the two players are interacting and assigns a score to their playing. This is done by the "Listener" module, which collects the inputs from the users and uses those to infer low-level features like tempo and measure duration, in cooperation with the "Scorer", that uses these low-level features to evaluate the interaction between the players and assign them their score. The computed score influences the musical augmentation generated by the "Generator" module, that uses the information from the other two modules to generate augmented music and synchronize it to what the users are playing. We will now better explain each of these modules in the following three subsections while we will add some remarks on the implementation of the system at the end of this section. Before reading the details of each module, the reader might find it useful to watch a video example execution of the game, available at the following URL: https://mediaspace.unipd.it/media/0_g90zoo2n

2.1 Listener

The MIDI pads (the black squares in Fig. 1) represent the only input interface of the system used by the users. In the current implementation, each player uses a single pad, but this could easily be changed in the future. Every time someone hits a pad, it sends a note_on message to the computer running the gaming software, which saves a timestamp when the message is received, along with the MIDI velocity included in the message, which represents the force with which the user struck the pad. Being a strongly time-dependent application, the system currently does not allow for network play and the pads must be connected to the same computer. The rhythm played by each user is saved as a list of events, which can then be represented as the topmost plots in Fig. 2 show.

The players are not required to follow a precise tempo, and their rhythm is completely improvised: at first, they can only hear the percussive sounds they generate by hitting the pads. The system must then infer the tempo they produce to add music that is synchronized with what the users play. To do so, a list of events collected from the pads is transformed through a process called *gaussification* [21]. Gaussification constructs an integrable function from a list of timing points by taking the linear combination of the Gaussians centered on the input points, as can be seen in the lower plots in Fig. 2. More precisely:

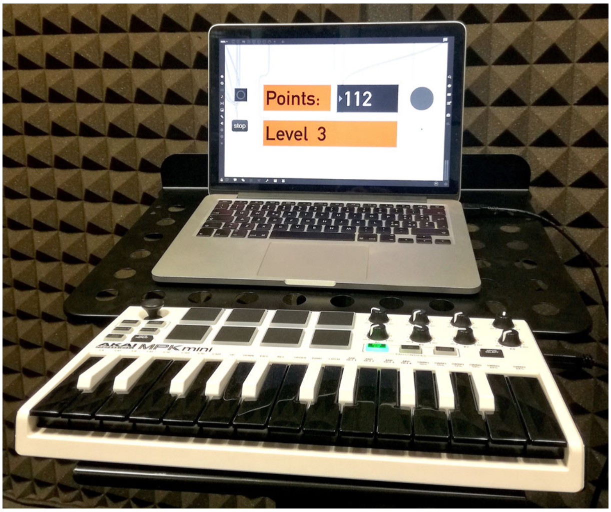


Fig. 1 The interface of the system: a computer running the software connected via USB to a MIDI keyboard with drum pads

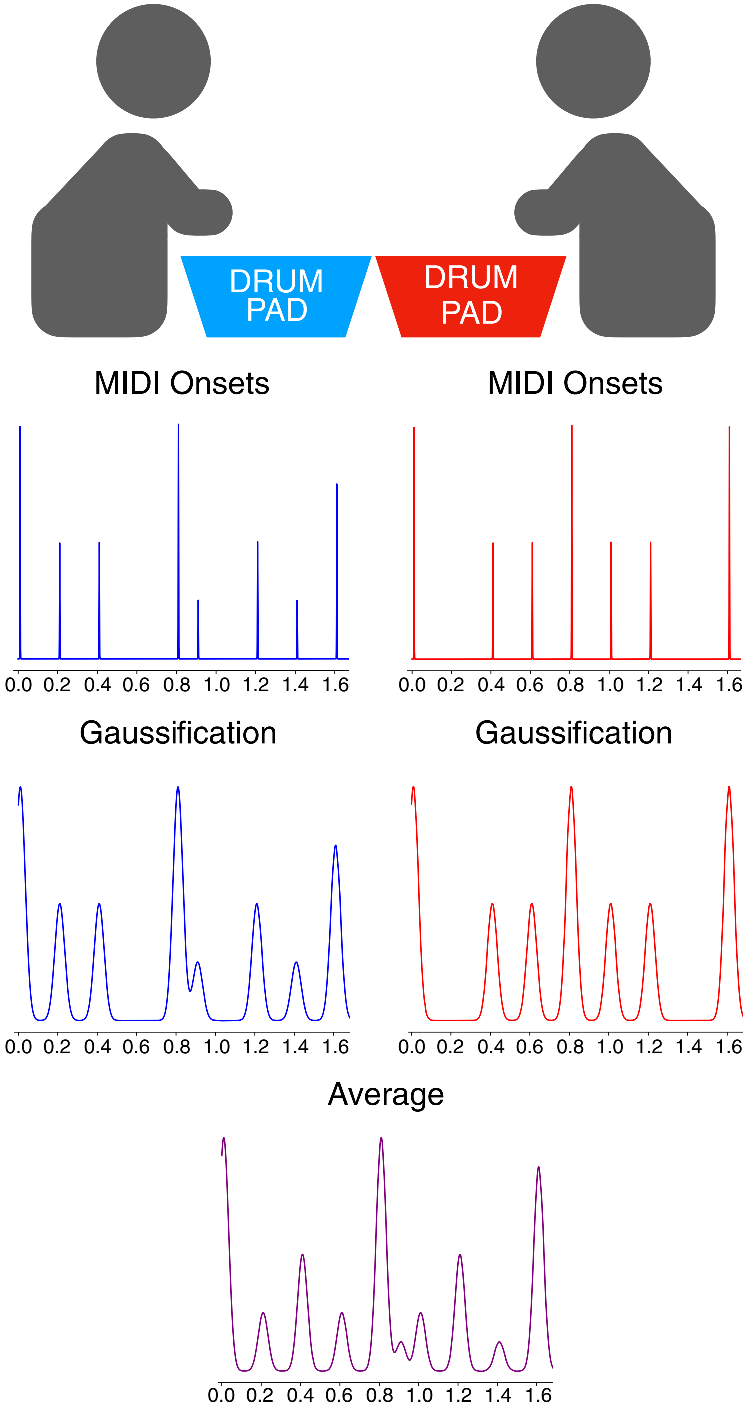


Fig. 2 Data flow in the Listener module: the onsets and velocities are collected from the drum pads and then gaussified, and an average of the two signals is computed

Let $R = \{t_i\}_{1\leq i\leq N}$ be a set of timestamps and $V = \{v_i\}_{1\leq i\leq N}$ the MIDI velocities associated with those timestamps. R and V together represent the input taken from the MIDI pads. Let σ be the standard deviation of the Gaussian kernel. Then

$$G_{R,V}(t) = \sum_{i=1}^{N} v_i e^{-\frac{(t-t_i)^2}{2\sigma^2}}$$

is the gaussification of R, V. The value for σ, as well as the tuning of the other parameters introduced below, will be discussed in Section 3.

Once this function is constructed, it is possible to infer the beat of the rhythm by computing the autocorrelation of the function (i.e. the correlation of the function with itself when shifted by a certain amount of time). The value of time-shift t that results in the highest correlation is the candidate beat duration. This can be computed with:

$$AG_{R,V}(t) = \sum_{i,j=1}^{N} v_i v_j e^{-\frac{(t-(t_i-t_j))^2}{2\sigma^2}} \tag{1}$$

The algorithm also uses a normalization based on perceptive features to ensure that the most appropriate tempo is chosen when there are ambiguities, derived from Parncutt's pulse-period salience [33]:

$$P(t) = e^{-\beta \log_2^2 t/t_s} \tag{2}$$

where t_s is the "spontaneous tempo" expressed in milliseconds, that is the tempo that humans are more likely to tap to if instructed to regularly tap to a tempo of their own choice. Throughout the research, t_s was fixed to 500 ms (120 bpm), following Parncutt's results. The other parameter, β, is a damping factor that describes the strength of the preference for the spontaneous tempo.

The meter (the number of beats in each measure) is computed by comparing "prototypical" functions representing different meters with the input signal. While doing this, the algorithm also determines the "phase" of the signal, that is the timing of the beginning of the measure, which does not necessarily coincide with the beginning of the signal. Additional details on this can be found in other works [14].

Frieler's Gaussification requires a finite set of timestamps in order to function. In practical terms, this requires selecting the latest events registered by the MIDI pads using a time window. The Gaussification is then performed on the set of events that happened within this time frame, regardless of the number of events and not considering the result of prior computations, which can result in less efficiency. Algorithm 1 instead allows computing a progressive result each time a single new input is received.

Algorithm 1 The algorithm for incrementally computing the autocorrelation of the input signal at a given time-shift t. The call to function $now()$ should output the current timestamp.

```
$events \leftarrow$ list of all prior events, added to the list with the following function. Each
element of this list contains three fields: $timestamp$, $velocity$ and $sum_t$;
$window \leftarrow$ the width of the time window in milliseconds;
Function Add_Event ($timestamp, velocity, t$)
    Output: $AG_{R,V}(t)$
    for $a$ in events do
        $a.sum_t \leftarrow a.sum_t + velocity * a.vel * e^{-\frac{(t-(timestamp-a.tstamp))^2}{2\sigma^2}}$;
    end
    $events \leftarrow$ events.append(event(timestamp, velocity, 0));
    $result \leftarrow 0$;
    for $a$ in events do
        if $a.timestamp > now() - window$ then
            $result \leftarrow$ result + a.sum$_t$;
        else
            remove a from events;
        end
    end
    return result;
end
```

The autocorrelation is computed using a single set of time points, which is the average of the two lists of events registered from the two users. Since the two players perceive the tempo as a feature emerging from the sounds produced by both, this approach is more appropriate for the estimation of the tempo the two players perceive, rather than computing the autocorrelation on both the signals and keeping a different tempo estimation for each user.

This module outputs the estimated tempo and meter of the average signal and a timestamp representing the moment in which the system estimates the next measure will begin. This timestamp is computed from the estimated duration of a measure and the estimated beginning of the measures in the signal (the phase) [14]. The beginning-of-measure prediction is fundamental for the Generator module, which uses it as a synchronization point between the music generated by the system and the sounds produced by the pads played by the users.

2.2 Scorer

The features computed by the Listener module are needed for the correct generation of music and to ensure the synchronization of the generated music with the rhythm played by the users. The Scorer module instead considers features related to the gaming aspects of the system to give feedback to the users about the quality of their interaction. The theoretical basis for this module derives from music therapy, and in particular from the "Improvisation Techniques for Music Therapy" devised by Kenneth Bruscia [9]. Bruscia's book describes a wide set of techniques available to music therapists to improve their interaction with the client. These are divided into different categories and do not focus solely on the music production but also on the physical/visual interaction with the client, which is a kind of information that is not available to our system. We selected five of the main basic features that only rely on the rhythm produced by the players. Here we report the definitions given by Perret in his comment on Bruscia's work [35] for the chosen techniques:

Imitation: Echoing or reproducing a client's response after the response has been completed;
Synchronization: Doing what the client is doing at the same time;
Incorporating: Using a musical motif or behavior of the client as a theme for improvising or composing, and elaborating it;
Pacing: Matching the client's energy level (i.e., intensity and speed);
Rhythmic grounding: Keeping a basic beat or providing a rhythmic foundation for the client's improvisation.

These techniques are meant to be used by the music therapist to help the client during the improvisation and do not immediately fit our situation where two peers are playing together and must be evaluated by a computer system. Nonetheless, they were useful in designing more precise and measurable features for our system. In particular, the system distinguishes four possible levels, describing the quality of the interaction:

Level 0: The system is incapable of clearly following the users, as they are not making a clear enough beat (no Synchronization or Rhythmic grounding);
Level 1: The system is capable of following the users, but one is dominating the rhythm and the other is not contributing (no Pacing);
Level 2: The interaction is considered normal: the two players have established a rhythm together;
Level 3: The interaction also includes imitations between the two players (Imitation and Incorporation).

The algorithm computes the level of interaction according to Algorithm 2, that requires as input the gaussified signals (the ones of the two players as well as the average one), the duration of beat and measure in milliseconds, and the list of the timestamps of the notes produced by the two players. The algorithm uses three functions: *correlation(a,b)*, that computes the correlation of the signal *a* with the signal *b*; *now()*, that returns the current timestamp; and *shift(a,b)* that moves along the x-axis all the points of the signal *a* by *b* units.

Algorithm 2 The algorithm for the computation of the current interaction level.

```
**Input**: $signal_1, signal_2, signal_a, beat, measure, notes_1, notes_2$
**Output**: Level of interaction
$clarity \leftarrow$ correlation($signal_a$, shift($signal_a$, $beat$))/ correlation($signal_a$, $signal_a$);
**if** $clarity < 0.4$ **then**
    **return** $0$;
**end**
$density_1 \leftarrow |\text{el} : \text{el} > \text{now()-10000} \;\&\; \text{el} \in notes_1|/10$;
$density_2 \leftarrow |\text{el} : \text{el} > \text{now()-10000} \;\&\; \text{el} \in notes_2|/10$;
**if** $density_1 < 0.5 \;||\; density_2 < 0.5$ **then**
    **return** $1$;
**end**
$crossCorr \leftarrow$ (correlation($signal_1$, shift($signal_2$,measure)) + correlation($signal_2$, shift($signal_1$,measure))) /2;
**if** $crossCorr < 15$ **then**
    **return** $2$;
**else**
    **return** $3$;
**end**
```

The algorithm computes three significant features to distinguish the levels. The *clarity* represents how confident the system is in estimating the current beat. The two *density* values represent the notes per second each user plays. Finally, *crossCorr* is how similar the two signals are at the distance of one measure, i.e., how much the users imitate each other measure per measure. The threshold values for the various levels were chosen empirically by computing the average values obtained by a "metronome" interaction, i.e. where the two users were substituted by a software sending a beat at regular intervals. It would be possible to use machine learning to determine better thresholds if enough data is collected from real users' interactions.

The final score is given to the users by adding a number of points each second depending on the current level. Notice that, since the goal of the game is the interaction between the two players rather than a challenge, the score is shared. The values of these features are not directly transformed into points given to the user, but rather to compute the current multiplication level. To ensure that the scoring system is stable, the level shown in the interface is not the latest computed level, but the median of the fifteen last computed levels. As well as being necessary for the visual feedback, the levels also influence the musical output as the music generation method is the same at every level (as described in the next section), but the volume of the generated instruments is proportional to the level reached by the players.

2.3 Generator

In this section, we describe the algorithms we used to generate music that is then synchronized with the rhythm established by the users. However, there are potentially no constraints on the music that can be added to the system since the Listener module computes all that is necessary to synchronize music to the beat produced by the users. For example, it would be possible to playback a pre-selected audio file (e.g. the users' favorite song) and synchronize it via time warping [19, 30, 39]. We decided to use simpler generated music to avoid taking the focus away from the interaction between the users. If the players are too captured by the music, they might start following it by tapping on its beats, becoming a sort of 'human metronome'. This would strongly reduce the interaction between the players and distract them from each other's rhythm. The Generator receives as input all the results of the other modules' computations. The most important information is the prediction of the beginning of the next measure. The Generator saves all the predictions obtained from the Listener, and each time a pad is struck the current time is compared with the saved predictions. If one is compatible with the current time (with a 50 milliseconds tolerance), the system estimates that the received input is the beginning of a new measure. The time tolerance is based on Parncutt's studies [33]. Every time this happens, the tempo and the meter of the Generator are updated to match the ones computed by the Listener, and the internal metronome of the Generator is reset and started. The Generator could potentially start a measure at the predicted moment without waiting for input from the user, but the above approach was preferred since it increases the feeling of control over the output. Having the system react to specific actions performed by the user is important to obtain the feeling of "I made this happen", which is considered crucial for the effectiveness of music therapy [46].

2.3.1 Harmonic accompaniment generation

At the beginning of each measure, a chord is selected using a first-order Markov chain. This chain can be manually crafted or can be generated via a Python script that uses the music21

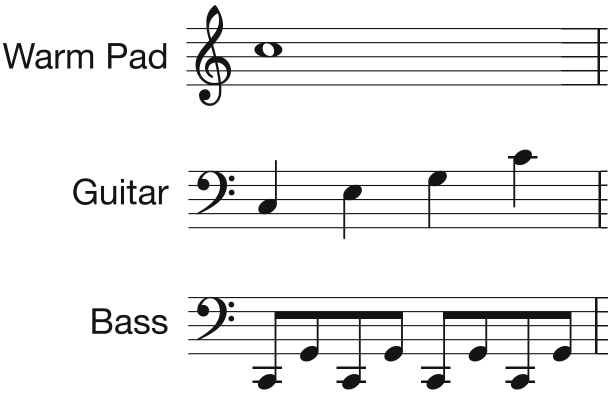


Fig. 3 A generated measure in 4/4 for each of the accompaniment instruments, based on a C major chord

library[1] to read a lead sheet in MusicXML format. The chain is constructed by taking note of the frequency of moving from one chord to another in the input file. The script is written so that the chords are analyzed in a way that does not depend on the key, i.e. by using roman numerals instead of actual chord names. The software only generates chord progressions in the key of C major, but it would be easy to transpose the generated music to other tonalities.

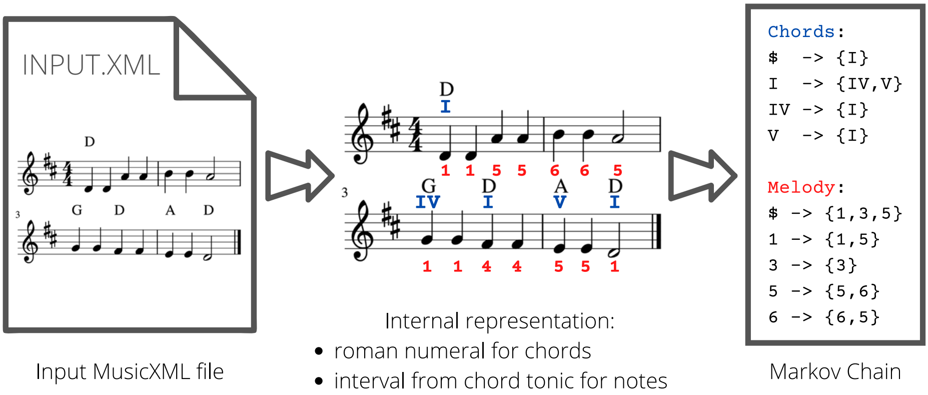
Three accompaniment instruments are used to play the chords: an example is shown in Fig. 3. The actual notes depend on the current chord, but the warm pad (General MIDI instrument 90) always plays the tonic while a guitar (General MIDI instrument 25) plays the chord arpeggio, and a bass (General MIDI instrument 34) alternates between the tonic and the fifth of the chord every eighth note. A new chord is selected each time there is synchronization between a note_on event and the beginning of a measure predicted by the Listener, but also when the internal metronome of the Generator reaches the 1st beat. This means that the chord changes with every measure unless the same chord is selected again.

2.3.2 Melody generation

To generate melody, a second Markov chain was constructed using the same Python script. Since the generated melodies should depend on the underlying chord being played at the moment, the script saves intervals between the tonic of the chord and the note being considered rather than the name of the note itself. The chain is restarted every time a chord changes, both in the learning and in the generation phase. Moreover, since some chords are major and others are minor, the intervals are saved as diatonic intervals and become a chromatic interval depending on the current chord during the generation phase. This process is briefly illustrated in Fig. 4.

A chromatic percussion instrument is associated with each player: a Music Box (General MIDI instrument 11) and a Vibraphone (General MIDI instrument 12). Chromatic percussions are used to keep the direct interaction of users focused on the rhythmic aspect, but these instruments can add pitch to the played notes. The two players follow two parallel chains that are both aware of the chord being played by the system. The Markov chain only controls the pitch of the generated melody, while the player determines the rhythm. Moreover, the Markov chain does not output the final pitch but rather an interval relative to the

[1] https://web.mit.edu/music21/



```
Chords:
$  -> {I}
I  -> {IV,V}
IV -> {I}
V  -> {I}

Melody:
$ -> {1,3,5}
1 -> {1,5}
3 -> {3}
5 -> {5,6}
6 -> {6,5}
```

Fig. 4 Data flow for the creation of the Markov chains from an input lead sheet. In the example chains, the likelihood of each transition is not represented

current chord. Therefore, each time a player hits his pad, the actual new note is generated following these steps:

1. Current chord is retrieved;
2. If the chord has changed, the seed is set to “$” (empty seed), else, the seed is the last generated interval;
3. The seed is used on the chain to generate the next interval;
4. The generated note is computed by following the interval chosen by the chain starting from the tonic of the current chord.

There is no control of the harmonicity of the notes generated by the two chains, but the use of diatonic intervals converted to chromatic intervals based on the chord ensures that it is impossible to generate strong dissonances. This approach of using intervals from the chord tonic to limit the output to “safe” notes was used by other notable works in Music Generation, such as Biles’ GenJam [7].

2.4 Implementation

The modules described above represent the abstract functionalities of the system rather than actual software modules. The real implementation we used for this system uses a Max/MSP[2] patch for the collection of the input from the user and the generation of the music. The patch communicates via Open Sound Control with a Python script that receives the list of inputs from the patch and computes both the features described in the Listener module and those related to the Scorer module. Practically, the Generator is implemented in Max/MSP and the Scorer in Python, but the Listener is shared between the two systems. This subdivision was chosen because Max/MSP is not fit for computations like those needed for the autocorrelation, but on the other hand Python is not ideal for real-time computing. Having a server invoked by the patch ensures that the computations done by Python are not critically dependent on timing (all the time labeling is handled by Max/MSP) while keeping the advantage of using Python and NumPy for the computation of the correlation of signals.

[2] https://cycling74.com/products/max/

3 Evaluation

The main goal of this study was to create a system that would help the players develop a rhythmic improvisation with music added by the software. The added music should help increase interpersonal synchronization, resulting in an engaging experience that can lead to better affiliation [27]. To help obtain this goal, a tempo tracking system was designed as described in Section 2.1. The evaluation of the system is twofold: we first describe a quantitative corpus-based evaluation of the system's ability to follow the rhythm as established by users, and then we include a questionnaire-based evaluation of the user experience to measure the general appreciation and engagement of the users with the system.

3.1 Quantitative evaluation

3.1.1 Setup

To verify the tempo tracking abilities of the system, a corpus of rhythmic performances is needed, having a known ground truth tempo. Moreover, since the system uses symbolic information (timing and velocity) based on the MIDI protocol, a corpus of MIDI files (or other symbolic formats) must be used for the assessment, rather than a corpus of audio files (such as `.mp3` or `.wav` files). Many such corpora exist and are available freely on the internet since such datasets are also used for music production. However, we found that most datasets were mostly (if not entirely) comprised of beats in 4/4 meter, and were generated from scores or other sources that use quantized timings, representing an ideal perfect tempo. Instead, real human performances (the ones we are most interested in) include subtle variations from the established tempo due to human expressiveness or to imprecisions and other physical constraints.

The dataset we chose is the Groove Dataset [23], created as part of Google's Magenta Project[3]. This dataset includes more than thirteen hours of drum MIDI files, recorded directly from human performances. The problem of not having many samples in meters other than 4/4 is present in this dataset as well: of its 1150 files, only twelve are not in 4/4 meter. Therefore, we decided to eliminate those twelve files and focus on 4/4 for this evaluation. This dataset also includes many "fills" samples, i.e., short phrases meant to join different parts of a song. These are not fit for tempo tracking because of the short length that would not allow the algorithm to find a rhythmic similarity. Those were thus excluded, along with all the files that lasted less than two measures, leaving a total of 448 files that were used for the evaluation of the system.

Since our system is not designed to simply tell the tempo of a given MIDI file, for the goal of this evaluation each input file was processed as follows:

- all MIDI note_on messages were considered a single hit on one pad of our system;
- notes that happened at the same moment (with a 25 milliseconds tolerance) were joined to be considered as a single event, summing their velocity and keeping the timestamp of the first note;
- only the first 15 seconds of each file were kept: the system considers only the notes within a defined time window, which was less than 15 seconds in all experiments, so the remaining part was disregarded for this evaluation.

[3] https://magenta.tensorflow.org/datasets/groove

Table 1 The results obtained by our system with various window settings, averaged across all β and σ values tested

Window (ms)	Correct	Imprecise	Total
2500	58.80%	15.80%	74.60%
5000	66.33%	14.71%	81.05%
7500	69.47%	13.65%	83.12%
10000	70.31%	13.27%	83.58%

The tempo output given by the system represents what the system estimates as the tempo for that particular moment, as our system never considers the file as a whole. This means that it uses less information than what is available, but this is a better simulation of what happens during the actual improvisation game, where the system must evaluate the tempo only based on the last few seconds of improvisation. We used two metrics in this evaluation: the percentage of files for which the tempo was correctly identified, and the percentage of files for which the tempo estimate is double or half the ground truth tempo. Since playing double or half tempo still allows synchronization (while the "feel" is different, the beats of a halved tempo always fall on a beat of the regular tempo), this is considered a minor mistake, and we thus consider the sum of these percentages as an indication of the precision of the system. These metrics were also used as a way to tune the three parameters of the tempo tracking system: σ, used to compute the autocorrelation of the signal (see (1) and Algorithm 1), β, a damping factor for the Parncutt function (2), and the time window of events to be considered when computing the tempo.

3.1.2 Results

We tested various values for those parameters, as reported in Tables 1, 2 and 3, testing all the combinations between these parameter settings. The tables report the percentages of estimates that were correct, as well as the percentage of estimates that were double or half the correct tempo ("Imprecise" guesses), and the total of times which the system guessed either correctly or imprecisely. Table 1 shows a first result which was to be expected: lengthening the window makes the estimates more accurate. These results would make us choose the longest window possible, but, in this context, the tempo does not vary from the established one. Having longer windows also means that the system will be slower reacting to tempo changes, a feature that is not tested here but is considered important to the system. Therefore, we fixed the window to 7500 ms in the following tables. Tables 2 and 3 show the results when changing the β and σ parameters, which are useful for tuning the system. From both tables, we can see correct estimates are not directly correlated to imprecise estimates. This means that tuning the parameters can make a difference in how the system interprets

Table 2 The results obtained by our system with various β settings, with a time window fixed to 7500 ms, averaged across all σ values tested

β	Correct	Imprecise	Total
0.25	63.77%	21.91%	85.68%
0.5	70.15%	15.80%	85.95%
0.75	71.82%	13.07%	84.89%
1	71.80%	11.84%	83.63%
1.5	70.85%	10.13%	80.98%
2	68.40%	9.19%	77.59%

Table 3 The results obtained by our system with various σ settings, with a time window fixed to 7500 ms, averaged across all β values tested

σ	Correct	Imprecise	Total
7.5	66.16%	17.36%	83.53%
10	68.15%	16.32%	84.47%
12.5	68.92%	15.59%	84.51%
15	69.73%	14.79%	84.51%
17.5	69.75%	14.43%	84.18%
20	70.53%	13.84%	84.37%
22.5	70.66%	13.67%	84.33%
25	70.95%	13.42%	84.37%
27.5	71.24%	13.05%	84.29%
30	71.11%	12.76%	83.87%
32.5	70.78%	12.34%	83.13%
35	69.66%	11.80%	81.46%
37.5	68.20%	11.22%	79.43%
40	66.67%	10.56%	77.22%

different tempo candidates (especially the β, used in (2) to distinguish between promising tempo candidates given by the autocorrelation computation). When choosing the parameters, a tradeoff must be found between favoring the correct estimate (that will result in a better experience) and tolerating more imprecise estimates that allow for synchronization even if the algorithm is not accurate. We fixed $\beta = 0.75$ and $\sigma = 25$, a combination that gave correct results in 74.06% of the cases and an additional 12.47% of imprecise results, for a total of 86.53% of "acceptable" estimates.

3.1.3 Comparison

It would be useful to compare these results to those obtained by algorithms for tempo tracking to give this data more meaning. Within the field of Music Information Retrieval (MIR), MIREX[4] is the primary collection of challenges and datasets relating to MIR tasks, so it is customary to compare new results to those obtained in those challenges. The challenge that comes closest to our goal is that of "Audio Tempo Estimation", which, as the name suggests, involves detecting the tempo of audio files. While being the most similar, being geared towards audio files makes this challenge very different from our goal of detecting tempo in a symbolic setting, but MIREX does not include any challenge or dataset for the estimation of tempo in symbolic files. This may be because, in most scenarios, a symbolic file will come with a tempo annotation (which is sometimes necessary to play the file) making it useless to use algorithms to infer the tempo. To have a benchmark, we choose nonetheless to compare our results with those obtained by Tempo-CNN [41] an open-source algorithm that is freely available online[5] and that performed well on the 2018 MIREX Audio Tempo Estimation challenge, which is the latest at the time of writing as the challenge was not repeated in 2019. Instead of using the MIREX dataset for the comparison, which is composed of audio files and cannot function with our system, we rendered the MIDI files from

[4] Music Information Retrieval Evaluation eXchange, www.music-ir.org

[5] https://github.com/hendriks73/tempo-cnn

Table 4 Comparison of the results of our system (window=7500, $\beta = 0.75$, $\sigma = 25$) and those obtained by Tempo-CNN

	Correct	Imprecise	Total
Our proposal	74.06%	12.47%	86.53%
Tempo-CNN	70.31%	21.43%	91.74%

the Groove dataset into audio files. The Groove dataset already includes an audio rendering of all its files, but using a standard rendering would not have allowed a fair comparison with our system, which only uses timestamps and velocity. To avoid including too much information (mainly timbre variation, due to using a full drumkit rather than a single pad), we rendered the MIDI using only a single sound for all the note_on events in the files. We choose the Hi Bongo sound (note 60 on General Midi channel 10) because it is a hand-played drum about the same size as our pads, making it most similar to the playing style we would expect on our system. The conversion from MIDI to .wav was performed using the Command Line tool TiMidity++[6]. For each of the 448 files in our corpus, Tempo-CNN estimated the tempo, and once again, we computed the number of correct guesses and also guesses that are half or double the correct tempo.

Table 4 shows the results obtained by our system in the ideal setting, compared to those of Tempo-CNN. The two systems performed similarly: our system guessed correctly more often, but Tempo-CNN had more acceptable guesses in total. It is also interesting comparing Tempo-CNN's result with the MIREX one, even if they are not directly comparable. MIREX requires submitted algorithms to guess two tempos rather than one, as the dataset includes two tempo annotations for each song. These two annotations are almost always one the double of the other, making the metric used by MIREX "At least one tempo correct" similar to our "Total" guesses, which accounts for double/half tempo estimates. Tempo-CNN guessed more than 97% of the times at least one tempo correctly on both the datasets used by MIREX. This suggests that there is a performance drop between what registered at MIREX and our evaluation due to the lack of timbrical information in our dataset, which highlights that it is harder to guess the tempo using the limited information available to our system. It is also to be noted that our system is extremely efficient: using Algorithm 1 for the computation of the autocorrelation incrementally, it is possible to calculate the autocorrelation at every new note in a couple of milliseconds, making this system extremely apt for real-time applications like the one described here. It is not immediately clear if Tempo-CNN could be adapted to operate in real-time.

All the evaluations described in this section focused on estimating the tempo on a given file that uses a single tempo throughout the execution. Meter detection was willingly left out of this evaluation, as well as the ability of the system to adapt to tempo and meter changes during the execution. The reason for this is that such mid-execution changes are hard to find in available datasets of human performances. These additional aspects were evaluated through simulations in a separate work: by simulating a human rhythm, we were able to vary both the tempo and the meter during the simulated execution. The results of those experiments can be found in a separate publication [14], but it is worth reporting some of the main findings. When the simulated tempo changed gradually, the system quickly adapted. Abrupt changes required instead more time to adapt, depending on the length of the used window. Since only the events that are within the window are considered, longer windows consider more events with the tempo/meter before the change, making it harder for

[6] http://timidity.sourceforge.net

the system to adapt to the new tempo. For this reason, the window should be kept relatively short to allow the system to adapt.

3.2 Qualitative evaluation

Twenty-four participants were collected among university students who volunteered to test a musical game. Nine were females and fifteen males with an average age of 24.17 years (standard deviation 3.48 years). In pairs, the participants who agreed to take part in the study and signed the informed consent were told they would use a game to create a “rhythmic interaction”. Before starting the experiment, they were shown the drum pads and how hitting them would produce a percussive sound. They were instructed to collaboratively create a rhythm, only using the pads and not talking or communicating if not through the percussions of the pads. They were also told that the system would add music to their rhythmic performance based on how they interacted. Immediately after the explanation, the users were left alone, while the researcher listened to the interaction from the next room, and the users could start playing as soon as the researcher had left. For this evaluation, the system for the generation of melodies was left out, leaving only the harmonic accompaniment described in Section 2.3.1. This was done because while informally testing the system, we found that the melody can distract the users from the rhythmic interaction, which we wanted to be the main focus in this evaluation.

After playing with the system for three minutes, the researcher returned to the room and each of the participants was asked to fill a questionnaire consisting of ten 7-points Likert items asking their level of agreement with the presented sentences, on a scale from 1 (Completely Disagree) to 7 (Completely Agree). The questionnaire also asked what their level of musical expertise was using a single question [53], but except for one classically trained musician, all reported little amateur experience or no experience at all. All the questions were posed in Italian, as it was the native language of all the participants. The questions in English, as well as the results of the questionnaire, are reported in Table 5. The Cronbach’s alpha of the collected data is 0.83, showing that the questionnaire has reasonable reliability. The first four questions were meant to assess whether the participants liked to play with the system, while the other questions were more intended to assess if they felt the system helped them interact musically. These questions were chosen to test whether the players considered

Table 5 The questions posed to the participants of the evaluation of the system with their average response, standard deviation, and median respons on a scale from 1 (Completely Disagree) to 7 (Completely Agree)

Question	Avg.	S. D.	Median
I found the game experience pleasant.	5.92	1.00	6
I think the music was aesthetically pleasing.	5.04	1.17	5
I wished to keep playing.	5.58	1.22	6
I wish to play again with this system in the future.	5.25	1.20	5
I was actively interacting with the other participant.	5.33	1.72	6
I was actively interacting with the system.	5.08	1.32	5
The system was interacting actively with me and the other participant.	5.50	1.29	5
I felt I had control over what was happening musically.	4.17	1.34	4
The system reacted to what I and the other participant were playing.	5.25	1.27	5
The music produced by the system was a stimulus to interact with the other player.	5.58	1.32	6

the game engaging, as this was the main aspect to be assessed via user-testing since it could not be directly assessed via quantitative measures by the researchers.

On average, the rating for the first questions was between 5 (Somewhat Agree) and 6 (Agree), indicating that the participants found the system to be pleasing to play with and would have liked to play again, meaning that the system is considered engaging by the participants. The results regarding the interactivity of the system are also positive, with averages between 5 and 6 except for question eight. The participants felt there was an interaction both between the players and with the system, and that the generated music was helpful to their interaction. Despite the perceived sense that the system reacted to their inputs, the participants did not feel like having full control over the produced music (the median response is 4, Neither Agree nor Disagree). This is probably due to the fact that the rhythmic dimension of the music, the one that is directly controlled by the users, is only a fraction of the whole musical output. This indicates that future iterations of this game should focus on enhancing the musical augmentation to consider more features of the input, so that the augmentation can vary mimicking the users' interaction more closely. This would in turn increase the feeling of control of the players, fundamental for the "I made this happen" feeling mentioned in Section 2.3.

Aside from the questionnaire, qualitative observations collected during the experiment by listening to the players' interactions show that while the system is both capable of following an established rhythm and quickly adapting to changes in tempo, the users are not driven to experiment with more complex interactions. This is probably because the system takes a few seconds to adapt to quick and abrupt changes (this is to be expected since the system needs at least a complete repetition of a period before being able to adapt), and the immediate feedback (before the system adapts) is negative. This induces the players to stick to the first common rhythm they can establish, which is not necessarily the best situation. Moreover, it was noticed that while the system is capable of handling meters different from 4/4, all the participants only used this meter. The trained musician seemed to try exploring more complex rhythms at times but settled for simpler ones possibly to accommodate the other player as well as not having immediate positive feedback from the system. This suggests that further studies only using musicians might show a different usage of the system. In general, the results collected from the questionnaire are not very strongly detached from the neutral response, showing that there is still much room for improvement in the system. Nonetheless, considering that the evaluation was carried out on what is the very first version of the system, the results are very encouraging. Further developments could improve the interaction to give a better feeling of control over the produced music, for example by adding more complex musical variations to the output mimicking the users' input more closely.

4 Conclusions

In this paper, we described a musical serious game that allows non-musicians to have social musical interactions. The system does not require any musical expertise, nor does it require the intervention of a music therapist. The game requires two players to improvise a rhythm by playing MIDI drum pads. Their improvisation is analyzed to determine rhythmic features (namely tempo, meter, and the onset of a measure) that are then used by the system to generate a musical background that is synchronized with the rhythmic interaction. The musical augmentation serves both as a reward for good interactions between the players, and to make the experience more engaging for the users. We described how the architecture of the system works and how it infers the features it needs by computing the correlation of signals constructed from the timing and velocities of the players' beats. From the computed correlation, it is possible to infer the beat kept by the players as well as the musical meter they are producing, and it is possible to predict the timing of the beginning of the next measure. We also described some metrics extracted from the same signal which were used to evaluate the interaction between the two players.

The system's ability to estimate tempo was evaluated using a dataset of human rhythmic songs, and compared to another work in literature. The results show that our algorithm performs comparably to other state-of-the-art proposals. A questionnaire was used to evaluate the gaming aspects of the system. Twenty-four students played with it and then responded to ten questions with a Likert scale. The collected answers show that they generally enjoyed the game and felt it helped them interact with each other.

The system was not designed to address a specific category of users, although the design was inspired by the social benefits that children with social difficulties can gather from musical interactions. It is more generally a tool that anyone can use to musically interact with someone else, e.g. music therapists, who could add this game to their toolbox. Other categories of users that could benefit from this kind of musical game include patients recovering from conditions that can impair motor skills, such as strokes. Recovery therapies include many repetitive exercises that can be made less fatiguing with the help of musical augmentation [1, 16, 22].

We evaluated the tempo-following system we propose and gave an indication of user engagement through a questionnaire. This data is useful as a first indication of the appreciation of the game, but there are still more aspects to be investigated. From the gaming perspective, usability tests should be performed to tell the quality of the system as a game. Focus groups consisting of music therapists and musicians can also give more insights on the quality of the musical and therapeutic aspects of the system and could be the occasion to implement a Heuristic Evaluation of the system. Regarding the therapeutic benefits of the game, a more in-depth study is needed using a larger pool of users, possibly more diverse especially in age, and collecting psychophysiological data along with their written impressions, gaining more reliable indicators of engagement and emotional activation. These studies should also use a control group using pre-existing technologies and different settings such as no musical augmentation versus harmonic augmentation and melodic augmentation. Moreover, if certain health benefits are desired from this game, they need to be established and assessed through a long-term evaluation, possibly asking professionals to test this game in their daily practice.

As pointed out in Section 3, the game would benefit from more complex variations in the generated music, to be more capable of mimicking the users' inputs. This requires advancements both in the Scorer module, which would need to extract more and more meaningful features, and in the Generator module, which should be capable of adapting the generated music to those features. For example, the articulation and the velocity of the generated notes are two clear indicators of emotional intent and could vary according to the force used by the users [10, 11, 48, 49].

We described a simple algorithm for the generation of melodies, but this was not used in the user assessment. Further investigation is needed to find out whether the melody is nothing more than a distraction from the rhythm or if it can make the experience more engaging and effective. We designed the melody generation to make it possible to influence it by choosing melodies the users like. It would be useful to assess if using user-selected melodic seeds can make the experience more enjoyable to users. The chosen melodies should also be able to adapt to the changes in the input, possibly defining more hierarchical levels of different note densities [44] and embedding expressive performance generation to leverage such hierarchical descriptions [13, 18].

Finally, an interesting investigation would be the perception of creativity from this system, which can be seen as a compositional tool shared between the users which retains a certain level of independence while being influenced by them, who also create social interactions between themselves [8, 17]. In particular, the emotional content of this interaction could give the users a different feeling of creativity, as they would be able to relate to the generated music better than they would if the music was simply generated by the computer on its own [12, 15, 42, 51].

Funding Open access funding provided by Università degli Studi di Padova within the CRUI-CARE Agreement. The first author is funded by a Doctoral Grant by Università degli Studi di Padova. No further funding was received for conducting this study.

Code Availability The code for the presented serious game is available as open source code at the following address: https://gitlab.dei.unipd.it/facoch/sympaddy/

Declarations

Competing interests The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Agres K, Herremans D (2017) Music and motion-detection: A game prototype for rehabilitation and strengthening in the elderly. In: 2017 International conference on orange technologies (ICOT). IEEE, Singapore, pp 95–98, https://doi.org/10.1109/ICOT.2017.8336097. https://ieeexplore.ieee.org/document/8336097/
2. Agres KR, Schaefer RS, Volk A, van Hooren S, Holzapfel A, Dalla Bella S, Müller M, de Witte M, Herremans D, Ramirez Melendez R, Neerincx M, Ruiz S, Meredith D, Dimitriadis T, Magee WL (2021) Music, computing, and health: A roadmap for the current and future roles of music technology for health care and well-Being. Music Sci 4:205920432199770. https://doi.org/10.1177/2059204321997709
3. Aigen K (2007) In defense of beauty: A role for the aesthetic in music therapy theory. Nord J Music Ther 16(2):112–128
4. Allen R, Heaton P (2010) Autism, music, and the therapeutic potential of music in alexithymia. Music Percept 27(4):251–261
5. Benveniste S, Jouvelot P, Lecourt E, Michel R (2009) Designing wiimprovisation for mediation in group music therapy with children suffering from behavioral disorders. In: Proceedings of the 8th International conference on interaction design and children, IDC '09. Association for computing machinery, New York, pp 18–26 https://doi.org/10.1145/1551788.1551793
6. Bégel V, Seilles A, Bella SD (2018) Rhythm workers: A music-based serious game for training rhythm skills. Music Sci 1:2059204318794369. https://doi.org/10.1177/2059204318794369
7. Biles JA (2013) Straight-ahead jazz with genjam: A quick demonstration. In: Musical metacreation: papers from the 2013 AIIDE workshop. Association for the advancement of artificial intelligence, p 4

8. Brown AR (2012) Creative partnerships with technology: How creativity is enhanced through interactions with generative computational systems. In: Proceedings of the 2012 AIIDE workshop. AAAI technical report WS-12-16, p 7
9. Bruscia KE (1987) Improvisational models of music therapy. Thomas, Springfield, IL, OCLC: 246139778
10. Canazza S, De Poli G, Rodà A (2015) CaRo 2.0: An interactive system for expressive music rendering. Adv Hum Comput Interact 2015:1–13. https://doi.org/10.1155/2015/850474. http://www.hindawi.com/journals/ahci/2015/850474/
11. Canazza S, De Poli G, Rodà A, Vidolin A (2012) Expressiveness in music performance: analysis, models, mapping, encoding. In: Steyn J (ed) Structuring music through markup language: designs and architectures. IGI Global, Hershey, PA, pp 156–186
12. Carnovalini F (2019) Open challenges in musical metacreation. In: Proceedings of the 5th EAI International conference on smart objects and technologies for social good, GoodTechs '19. ACM, New York, pp 124–125. https://doi.org/10.1145/3342428.3342678. Event-place: Valencia, Spain
13. Carnovalini F, Rodà A (2019) A multilayered approach to automatic music generation and expressive performance. In: 2019 International workshop on multilayer music representation and processing (MMRP). IEEE, Milano, Italy , pp 41–48. https://doi.org/10.1109/MMRP.2019.00016. https://ieeexplore.ieee.org/document/8665367/
14. Carnovalini F, Rodà A (2019) A real-time tempo and meter tracking system for rhythmic improvisation. In: Proceedings of the 14th International audio mostly conference: A journey in sound, AM'19. Association for computing machinery, New York, pp 24–31. https://doi.org/10.1145/3356590.3356596. Event-place: Nottingham, United Kingdom
15. Carnovalini F, Rodà A (2020) Computational creativity and music generation systems: An introduction to the state of the art. Front Artif Intell 3:14. https://doi.org/10.3389/frai.2020.00014
16. Chen JL (2018) Music-supported therapy for stroke motor recovery: theoretical and practical considerations. Annals of the NYAS 1423(1):57–65
17. Corneli J, Pease A, Stefanou D (2018) Chapter 6 social aspects of concept invention. In: Confalonieri R, Pease A, Schorlemmer M, Besold TR, Kutz O, Maclean E, Kaliakatsos-Papakostas M (eds) Concept invention: foundations, implementation, social aspects and applications, computational synthesis and creative systems. Springer International Publishing, Cham, pp 153–186. https://doi.org/10.1007/978-3-319-65602-1_6
18. Cristani M, Pesarin A, Drioli C, Murino V, Rodà A, Grapulin M, Sebe N (2010) Toward an automatically generated soundtrack from low-level cross-modal correlations for automotive scenarios. In: MM'10 - Proceedings of the ACM multimedia 2010 international conference, pp 551–559
19. Dannenberg RB (1984) An on-line algorithm for real-time accompaniment. In: ICMC, vol 84. Michigan Publishing, Ann Arbor MI, pp 193–198
20. Dixon S (2001) Automatic extraction of tempo and beat from expressive performances. JNMR 30(1):39–58
21. Frieler K (2004) Beat and meter extraction using gaussified onsets. In: ISMIR. Universitat Pompeu Fabra, Barcelona, Spain, p 6
22. Fujioka T, Dawson DR, Wright R, Honjo K, Chen JL, Chen JJ, Black SE, Stuss DT, Ross B (2018) The effects of music-supported therapy on motor, cognitive, and psychosocial functions in chronic stroke. Annals of the NYAS 1423(1):264–274
23. Gillick J, Roberts A, Engel J, Eck D, Bamman D (2019) Learning to groove with inverse sequence transformations. In: International conference on machine learning (ICML), p 11
24. Gouyon F, Herrera P (2003) Determination of the meter of musical audio signals: seeking recurrences in beat segment descriptors. In: AES Convention 114. AES, Amsterdam, Netherlands, p 8
25. Hallam S (2010) The power of music: Its impact on the intellectual, social and personal development of children and young people. Int J Music Educ 28(3):269–289
26. Hawryshkewich A, Pasquier P, Eigenfeldt A (2010) Beatback: A real-time interactive percussion system for rhythmic practise and exploration. In: NIME '10. University of Technology Sydney, Australia, pp 100–105
27. Hove MJ, Risen JL (2009) It's all in the timing: interpersonal synchrony increases affiliation. Soc Cogn 27(6):949–960
28. Koelsch S (2015) Music-evoked emotions: principles, brain correlates, and implications for therapy. Annals of the NYAS 1337:193–201. https://doi.org/10.1111/nyas.12684
29. Kokotsaki D, Hallam S (2007) Higher education music students' perceptions of the benefits of participative music making. Music Educ Res 9(1):93–109. https://doi.org/10.1080/14613800601127577
30. Moens B, Muller C, Van Noorden L, Franěk M, Celie B, Boone J, Bourgois J, Leman M (2014) Encouraging spontaneous synchronisation with d-jogger, an adaptive music player that aligns movement and music. PLoS ONE 9(12):e114234. https://doi.org/10.1371/journal.pone.0114234

31. Muller M, Kurth F, Roder T (2004) Towards an efficient algorithm for automatic score-to-audio synchronization. In: ISMIR. Universitat Pompeu Fabra, Barcelona, Spain, p 8
32. Pachet F (2002) Interacting with a musical learning system: The continuator. In: Music and artificial intelligence. Springer, pp 119–132
33. Parncutt R (1994) A perceptual model of pulse salience and metrical accent in musical rhythms. Music Perception: An Interdisciplinary Journal 11(4):409–464. https://doi.org/10.2307/40285633
34. Pérez-Arévalo C, Manresa-Yee C, Beltrán VMP (2017) Game to develop rhythm and coordination in children with hearing impairments. In: Proceedings of the XVIII International conference on human computer interaction, Interacción '17. Association for computing machinery, New York, p 4. https://doi.org/10.1145/3123818.3123853
35. Perret DG (2005) Roots of musicality: Music Therapy And Personal Development. J. Kingsley Publishers, London
36. Quintin EM, Bhatara A, Poissant H, Fombonne E, Levitin DJ (2011) Emotion perception in music in high-functioning adolescents with autism spectrum disorders. J Autism Dev Disord 41(9):1240–1255. https://doi.org/10.1007/s10803-010-1146-0
37. Raphael C (2002) A bayesian network for real-time musical accompaniment. In: Advances in neural information processing systems 14. MIT Press, Cambridge, MA, pp 1433–1439
38. Ritterfeld U, Cody M, Vorderer P (2009) Serious games: mechanisms and effects. Routledge, https://doi.org/10.4324/9780203891650
39. Robertson A, Plumbley M (2007) B-keeper: A beat-tracker for live performance. In: NIME '07. ACM Press, New York, pp 234
40. Santolin C, Russo S, Calignano G, Saffran JR, Valenza E (2019) The role of prosody in infants' preference for speech: A comparison between speech and birdsong. Infancy 24(5):827–833. https://doi.org/10.1111/infa.12295
41. Schreiber H, Müller M (2018) A single-step approach to musical tempo estimation using a convolutional neural network. In: Proceedings of the 19th International society for music information retrieval conference (ISMIR). Paris, France, p 8
42. Scirea M, Eklund P, Togelius J, Risi S (2017) Can you feel it?: evaluation of affective expression in music generated by metacompose. In: Proceedings of the genetic and evolutionary computation conference, GECCO '17. ACM, New York, pp 211–218. https://doi.org/10.1145/3071178.3071314. Event-place: Berlin, Germany
43. Shah V, Cuen M, McDaniel T, Tadayon R (2019) A rhythm-based serious game for fine motor rehabilitation using leap motion. In: 2019 58th Annual conference of the society of instrument and control engineers of Japan (SICE), pp 737–742
44. Simonetta F, Carnovalini F, Orio N, Rodà A (2018) Symbolic music similarity through a graph-based representation. In: Proceedings of the audio mostly on sound in immersion and emotion - AM'18. ACM Press, Wrexham, pp 1–7. https://doi.org/10.1145/3243274.3243301
45. Stige B (1998) Aesthetic practices in music therapy. Nordisk Tidsskrift for Musikkterapi 7(2):121–134
46. Swingler T (1998) The invisible keyboard in the air: An overview of the educational, therapeutic and creative applications of the EMS Soundbeam. In: 2nd European conference for disability, virtual reality & associated technology. University of Reading, Skövde, Sweden, pp 253–259
47. Toiviainen P (1998) An interactive MIDI accompanist. Comput Music J 22(4):63–75. https://doi.org/10.2307/3680894
48. Turchet L, Rodà A (2017) Emotion rendering in auditory simulations of imagined walking styles. IEEE Trans Affect Comput 8(2):241–253
49. Turchet L, Zanotto D, Minto S, Rodà A, Agrawal SK (2017) Emotion rendering in plantar vibro-tactile simulations of imagined walking styles. IEEE Trans Affect Comput 8(3):340–354
50. Whiteley N, Cemgil AT, Godsill S (2006) Bayesian modelling of temporal structure in musical audio. In: ISMIR. University of Victoria, Canada, pp 29–34
51. Williams D, Kirke A, Miranda ER, Roesch E, Daly I, Nasuto S (2015) Investigating affect in algorithmic composition systems. Psychol Music 43(6):831–854
52. Xia GG, Dannenberg RB (2017) Improvised duet interaction: learning improvisation techniques for automatic accompaniment. In: NIME '17. Aalborg University, Copenhagen, Denmark, p 5
53. Zhang JD, Schubert E (2019) A single item measure for identifying musician and nonmusician categories based on measures of musical sophistication. Music Percept 36(5):457–467. https://doi.org/10.1525/mp.2019.36.5.457