Emergent Orchestras: A Modular Framework for Musical Robot Swarms

Lluc Bono Rosselló[1]*, Muhanad Alkilabi[2,3], Elio Tuci[2], Hugues Bersini[1], and Andreagiovanni Reina[4,5]*

[1]Institute for Interdisciplinary Studies on Artificial Intelligence (IRIDIA), Université Libre de Bruxelles, Brussels, Belgium
[2]Department of Computer Science, Université de Namur, Namur, Belgium
[3]Department of Medical Instruments Techniques Engineering, AlSafwa University College, Iraq
[4]Centre for the Advanced Study of Collective Behaviour (CASCB), Universität Konstanz, Konstanz, Germany
[5]Department of Collective Behaviour, Max Planck Institute of Animal Behavior, Konstanz, Germany
*Lluc.Bono.Rossello@ulb.be; andreagiovanni.reina@gmail.com

Abstract

Can music emerge from a swarm of robots each playing a single note and coordinating its behaviour with the others? We explore this idea by proposing a modular framework for the emergent generation of music, representing a novel intersection between robotics and artistic creation. We move beyond the works that link sound to robot movements or that allocate robots to musical roles. In our system, despite being limited to playing the atomic musical element, i.e., a single note, robots self-organise to play musical creations collectively. We illustrate the modular architecture of our framework by presenting three independent modules that run in parallel to enable the swarm to reach (i) temporal coordination so that robots play in synchrony, (ii) harmonic consensus so that notes are harmonically coherent, and (iii) beat distribution so that notes are distributed throughout time. We implement algorithms for the three modules building upon and extending existing swarm robotics solutions. Our bottom-up and modular approach also enables the use of cheap and accessible robots, hence fostering applicability, scalability, and robustness. Finally, combining the robot's physical embodiment with the swarm's plurality brings a unique dimension to the musical performance. We showcase our collaborative music creation framework with simulations and a real robot performance comprising 12 robots. This study shows the potential of combining music with swarm robotics to create musical complexity from simple robotic actions.

Introduction

Current generative artificial intelligence (AI) accomplishes tasks previously believed to be exclusively within the domain of human capability and, during the last months, we are witnessing a revolution in the automatic generation of high-quality texts, images, videos, and music. We are amazed by AI's successes because computers can now perform tasks that we believed required human creativity. The field of computational creativity (Colton et al., 2012) studies computational methods that allow us to understand and mimic human creativity. Most research efforts are now devoted to what we can call 'individual' computational creativity where a single (advanced) algorithm/method processes some input to generate a creative output, e.g., ChatGPT, DALL-E, Sora (Achiam et al., 2023; Ramesh et al., 2022; Brooks et al., 2024). Conversely, collaborative creativity consists of having a group of agents contributing to a single creative product through repeated interactions and collaboration (Mamykina et al., 2002). This is one of the aspects studied in computational social creativity (Saunders and Bown, 2015), which considers creativity as a process embedded into the society where interaction among agents is a key component. In our study, we focus on a group of artificial agents interacting and collaborating to produce a creative outcome. More precisely, we focus on the collective generation and playing of music, akin to what we describe as group improvisation among human musicians, or more informally said a jam session.

Our goal is to port the idea of musical improvisation to collective robotics. To do so, we build a generative collective AI framework that enables a group of robots to create and play music collaboratively. None of the robots is necessary or fundamental to the task, in fact, none of the robots could achieve alone what the group is capable of. This principle aligns well with what we can observe in social insect societies that, in turn, have inspired the field of swarm robotics (Dorigo et al., 2021). Swarm robotics studies how groups of robots can perform tasks collaboratively without any central leader, i.e., in a decentralised way. Thanks to decentralisation, robot swarms have the potential to be more scalable, more flexible, and more robust than centralised multi-robot systems (Hamann, 2018). Our idea is to design decentralised algorithms for robot swarms to enable them to play music that is collaboratively composed in real time. In this study, we focus on designing the basic rules upon which more advanced generative models can be built. We follow a bottom-up approach of incremental creativity through a modular framework where each module handles one aspect of automatic music generation and works independently from and in parallel to the other modules. By inheriting swarm robotics's properties, the proposed system can be a scalable, robust, and flexible generative collective AI framework for music.

Our research is different from and goes beyond previous work on AI-generated music (Tatar and Pasquier, 2019).

Different from traditional generative AI methods for music, e.g., (Yang et al., 2017), we study collaborative generation of music through interactions between autonomous artificial agents. Previous works that generated music using robot swarms defined a mapping between the robots' attributes, such as their velocity or position, and the musical parameters (Blackwell, 2007; Albin et al., 2012). In this way, the observed movement or collective behaviour of the robot swarm was translated into music – a so-called sonification – and any change or perturbation to the swarm had a direct influence on the resulting music. Recent work has conceptualised sonification through Hilbert spaces, improving the mapping between swarm and musical spaces (Mannone et al., 2023). While in sonification robots are 'unaware' of how their behaviour influences the resulting music, in our work robots are autonomous active agents that act towards the goal of creating music and change their behaviour according to the musical dynamics. Other works studied how groups of robots could actively play music in unison, allocating each robot to specific roles, e.g. music director, drummer, or pianist (Martins and Miranda, 2007; Eigenfeldt and Kapur, 2008). Similarly, it has been explored the idea of having a network of devices or robots that move and interact to synchronise a predefined audio file that they play together (McLurkin, 2022; Nymoen et al., 2014). In our work, synchronisation is only one aspect of the musical performance since our framework also includes the collaborative generation of music. This is similar to Dr Squiggles system (Krzyżaniak, 2021; Krzyzaniak, 2022) which comprises robots specifically designed to generate rhythms on percussion instruments. Similar to our work, their system collaboratively generates music, however Dr Squiggles is limited to the space of rhythmic creativity and both the harmony and the interaction network are predefined as their robots are programmed with a predefined sequence of notes to play and cannot move (static positions). In our system, each robot only plays a single note which is chosen depending on its social context. Hence our system aims to create music starting from the atomic level where, for us, the 'atoms' of music are single notes. What note to play and when to play it are decisions made by each robot based on what others are playing.

Employing robots to play the generated music is an important artistic aspect of our proposed framework. The tangible presence of robots as performers not only enriches the audience's experience but also emphasises the importance of embodiment in artistic expression. Embodied agents, in our case robots, with physical forms that interact in real-world settings, bring a unique dimension to performances, blending the lines between digital creation and physical manifestation. This concept of embodiment is crucial for enhancing audience engagement and emotional impact, as it provides a visual and auditory spectacle that purely digital systems cannot. Our system has several aspects that suit particularly well its employment in live performances of various kinds. It runs on generic (cheap) robotic platforms with minimal requirements, scales naturally (without needing algorithmic changes) to different group sizes, is robust to real-time malfunctioning, and keeps self-healing by adapting to perturbations. In addition to making the framework more widely applicable, running on generic – potentially cheap – robots also promotes accessible science as any research institute can afford to reproduce our system and build upon it.

Our framework develops along multiple musical dimensions through distinct behavioural modules. Hence, instead of synthesising music by crafting elements that embody multiple characteristics simultaneously – such as note, duration, offset, and timbre – each of those characteristics is generated independently yet concurrently, governed by its own set of interaction rules. In particular, in our experiments, rather than creating a chord progression conventionally, robots independently coordinate to 1) adhere to a unified tempo, 2) produce harmonically compatible notes, and 3) distribute these notes across the beats within a bar. Through robot simulation and experiments with 12 autonomous real robots, we demonstrate that through parallel interactions across various musical dimensions, the swarm can achieve a cohesive musical state akin to one generated by elements with multiple integrated features. An overview of the system and our results are presented in the summary video available at `https://youtu.be/ZM-gT9RWz80`. Our incremental approach allows for the introduction of additional dimensions through the design of new specific behavioural modules, such as the local adaptation of timbre, for example, following the method presented in (McLurkin, 2022). We discuss this aspect and other future research directions in the concluding discussion.

Methodology

In this study, we develop a framework to make music emerge from a swarm of robots. These robots are very simple in terms of both hardware and software. We assume that these robots are capable of (i) locally communicating small messages to nearby robots (i.e., they can broadcast a limited amount of data in a limited range); (ii) moving in their environment (i.e., they can change their position over time); and (iii) playing the audio file of a note. Our work takes inspiration from (natural and artificial) swarm systems where complex collective behaviour emerges from repeated interactions and self-organisation among numerous simple individuals (Camazine et al., 2001). Our robots are also simple individuals in that they are limited to only playing a single note – the most atomic element of music – and thanks to their self-organising behaviour and repeated interactions, the music emerges as a collective result. Each note corresponds to a pitch – i.e., the perceived frequency of a sound – played for a given amount of time. In our case, robots always play pitches for the same duration of 1 second, however, our framework offers the possibility to be personalised and play a pitch for shorter or longer durations. In summary, at the individual level, each robot can move, communicate in a short range, and play one note, and at the group level, the swarm generates and plays music.

How do we create music? To put this abstract goal into tangible objectives, we derive a set of subgoals that the swarm must achieve to create music. The behaviours to achieve these subgoals form the modules of our framework. In this work, we specify two foundational modules – temporal coordination and harmony – that are the basis for building any music, and a third higher-level module – beat distribution – that we tailor to create a specific type of music and is meant to be changed and personalised in future research.

Through temporal coordination, the robots can synchronise their actions in time, a crucial component to the emergence of rhythm. Temporal coordination allows the swarm to start and stop playing notes in a coordinated manner, following a common tempo – or notion of shared time – which is a foundational element of music composition. A shared temporal dimension (i.e., achieving synchronisation) allows bars (or measures) to divide the music into segments of equal duration, with the number of beats in each bar defined by the time signature (e.g., 4 beats per bar in a common 4/4 time signature). Consequently, beats are the basic time units within music, and their arrangement within bars gives rise to various rhythmic patterns. The length of musical units, such as notes or rests, is also determined relative to these beats, creating the rhythmic structure of a piece.

Through harmony, the robots play a musically coherent range of notes (called the harmonic context). In this way, robots do not need to play the same note but each robot can play a different note covering a range of pitches, however, all pitches should fit in the same harmonic context. In simple terms, the harmonic context defines which notes "sound good" when played together and which do not. The harmonic context relates to the frequently-used concept of musical scales which are ordered sequences of pitches that serve as the foundational structure for melodies and harmonies. For simplicity and clarity, in this study, we will use the set of major scales as our harmonic context, known for its sequence of whole and half steps that create a distinctively "happy" or "bright" sound.

Through beat distribution, we can move from notes to chords. Obtaining chords requires that the two foundational subgoals of temporal coordination and harmony are in place. When notes belonging to the same scale are played together, chords can be formed. In this work, we implement a simple beat distribution mechanism that enables us to show how it is possible to extend our framework in such a direction.

An important aspect of our framework is its modularity. Each subgoal is achieved through a mechanism implemented in an independent module that runs in parallel to the others. These modules consist of rules – i.e., algorithms – that the robot follows based on the local and partial knowledge they acquire from their neighbours. In the next subsections, we give a description of the mechanisms to achieve the three subgoals and the respective algorithms that we implemented to achieve them, taking inspiration from existing solutions in the swarm robotics literature.

Temporal Coordination

Each robot is an independent unit of hardware without any centralised unit – e.g., an orchestra director observable by all musicians – to keep a common reference of time. Therefore, robots need to achieve global synchronisation through local interactions with each other. Once synchronisation is achieved, it can be used to play notes simultaneously or with a controlled time difference (as further discussed in the module for beat distribution).

The robots have a CPU clock that allows them to work with the same frequency. In our case, we set a frequency of 0.25 Hz per bar, i.e., every 4 seconds a new bar starts. As robots share the same frequency, they only need to reach an agreement on phase. To achieve decentralised phase-synchronisation we use the Kuramoto Model (Kuramoto, 1975), which has already been successfully implemented on swarms of robots, e.g.,(Perez-Diaz et al., 2015). This simple yet effective model allows multiple oscillators with the same natural frequency to synchronise in phase through local interactions. The Kuramoto model is implemented on the robots, by using a counter θ, expressed in radians, that tracks the elapsed time from the start of each bar. Our robots have as their natural period $T = 4000$ ms, therefore the counter's maximum value T corresponds to 2π radians. Every millisecond, each robot i increments its counter by one step as

$$\theta_i(t+1) = \left(\theta_i(t) + \frac{2\pi}{T}\right) \mod 2\pi,$$

where $\mod$ is the modulo operator and $\theta_i(t)$ is robots i's phase counter at time t. Robots periodically broadcast their current phase θ. Every time a robot receives a neighbour's θ value, it uses it to update its internal phase counter. At each interaction with a robot j, robot i updates its phase as

$$\theta_i(t) = (\theta_i(t) + K \sin[\theta_j(t) - \theta_i(t)]) \mod 2\pi, \quad (1)$$

where $K = 1$ is the coupling strength and $\theta_i(t)$ and $\theta_j(t)$ are the phase counters of robots i and j, respectively.

Harmonic Consensus

In musical improvisations, human musicians normally agree on the scale to use before starting to play. In our swarm, robots also need to agree on a common scale, however, this collective decision is made concurrently with the musical performance (while they play).

Selecting the common scale, or common harmony, is necessary to play pitches that are in good harmonic agreement with each other. In Western music, it is common to consider that there are 12 distinct pitches, which recur periodically with different sound frequencies (i.e., in higher or lower octaves). For example, the standard keyboard, namely the piano, encompasses 88 distinct keys, each playing one of the 12 pitches in a different octave. Using a mathematical notation similar to the MIDI (Musical Instrument Digital Interface) notation, we can represent each pitch by an integer $1 \leq p \leq 88$, where the recurrence of pitches can be represented as the modulo function. Since there are only 12 pitches, we can consider keys that differ by multiples of 12 as harmonically equivalent. Therefore, the robot plays one of the 88 possible pitches but when finding harmonic consensus, the mechanism can be simplified by only considering 12 unique pitches (set $P \in \{1, 2, ..., 12\}$). Each of these pitches can be the root note (tonic) $p_0 \in P$ of a major scale, thus we have 12 possible major scales. The harmonic consensus is found by agreeing on what scale to use.

Each of our robots only plays one of the 12 unique pitches and each scale only contains 7 among these 12 unique pitches. Therefore, robots repeatedly exchange information on what is their selected pitch and, when their pitch is not within the scale of its neighbours, the selected pitch is changed. In this study, we only use the major scales, which are the most known to the public, for example, the C major scale contains the pitches (C, D, E, F, G, A, B). To define a major scale, we introduce the concepts of tone interval (T) as 2 steps and a semitone interval (S) as 1 step in our modulo-12 pitch set. A major scale is defined by a specific interval pattern $I =$ [T, T, S, T, T, T, S], which translates to steps of size $I = [2, 2, 1, 2, 2, 2, 1]$ (e.g., see the scales shown in Figure 1). Given the interval array I, a major scale m starting from a pitch $p_0 \in P$ can be generalised by iterative addition of elements from I to p_0, modulo 12, as follows:

$$m = \left\{ (p_0 + \sum_{k=0}^{j} I[k]) \mod 12 \,|\, j \in \mathbb{N} \wedge 0 \leq j \leq 6 \right\} \quad (2)$$

where the set m is a major scale composed of 7 elements.

To reach a harmonic consensus, the selected pitch of a robot should be in harmonic agreement with the selected pitches of the other robots, meaning that all pitches belong to the same major scale (comprising 7 pitches). To reach this consensus we implement an algorithm inspired from the minimal naming game (Baronchelli and Diaz-Guilera, 2012), which has been previously used to study consensus formation in robot swarms (Trianni et al., 2016). In the minimal naming game, agents agree on the same word (or name) by updating a local vocabulary where they add and remove words depending on the words that their neighbours are currently using.

In our framework, each robot broadcasts its pitch to its neighbours and maintains a buffer for the last $L = 3$ received pitches. The robot scans through all possible 12 scales to identify whether a harmonic consensus is already

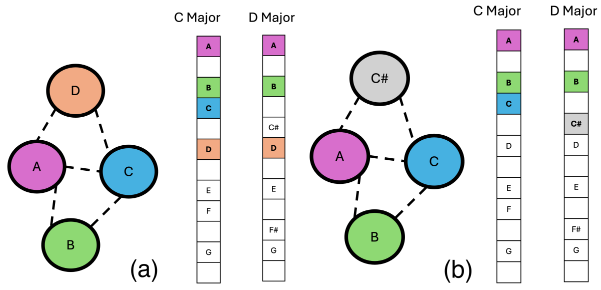


Figure 1: Illustration of one step of the harmonic consensus process. Each robot (shown as a circle) selects a pitch (shown as a colour and indicated within the circle) that exchange with its neighbours (connected by dashed lines). To simplify the representation, we only show two of the 12 possible major scales, each comprising 7 pitches. (a) The robots are in harmonic consensus as all robots' pitches belong to the C major scale. (b) There is no harmonic consensus as no scale includes all four robots' pitches.

in place, i.e., all the $L + 1$ pitches belong to a common scale. This situation of harmonic consensus is depicted in Figure 1a, where all robots' pitches belong to the C major scale. When there is no common scale, the robot probabilistically changes its selected pitch. This change happens with probability $r_c < 1$ to avoid that multiple robots deterministically change every time step in a perpetual loop never meeting each other. In our experiments, we set $r_c = 0.7$. This situation of harmonic disagreement is depicted in Figure 1b, where the robots' pitches do not belong to any shared major scale. Therefore, the robots change their selected pitch with probability r_c and select a new pitch by moving the current pitch by one step. For example, the grey robot randomly selects either C or D, or the blue robot randomly selects either B or C#. The selection of whether moving one step up or down (e.g., C or D for the grey robot) depends on which move maximises the probability of reaching an agreement. Assuming that the other robots will not change their pitches, the robot makes the change that would maximise agreement in the next step.

Local Distribution of Beats

Once the two foundational modules – synchronisation and harmonic consensus – needed to create music are in place, we can build more complex musical patterns by adding new modules to our framework. We add a third module to distribute beats within each bar, so that robots do not all play at the same time but they place their beats into subdivisions of time. In this example, as an arbitrary artistic choice, we implement a method to equally distribute notes among the available beats of a bar. We assume the bar (or measure) is divided into $N = 4$ sub-windows, thus each bar has $N = 4$ beats. The robots are initialised to play in a given beat W (in our case in the first beat $W = 1$) and throughout the process,

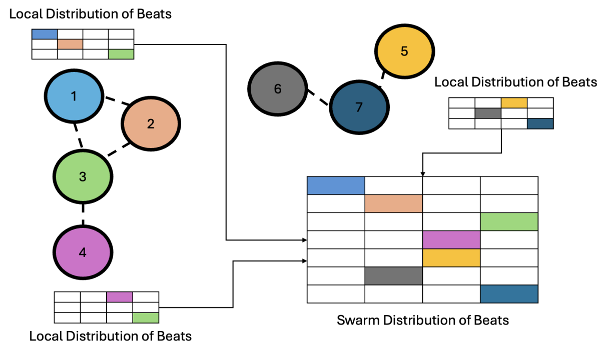

Local Distribution of Beats
1
2
3
4
5
6
7
Local Distribution of Beats
Local Distribution of Beats
Swarm Distribution of Beats

Figure 2: Illustration of one step of the local distribution of beats process. Each robot (shown as a circle) selects one of the $N = 4$ beats that minimises the overlap with its neighbours (shown as local tables of distribution of beats). The distribution at the swarm level (large table) corresponds to what we hear and is the sum of all the local distributions. On average, in a swarm of M robots, in each beat, there are $M/N = 7/4 = 1.75$ robots playing simultaneously.

they share W with their neighbourhood. To avoid playing simultaneously with other neighbours, each robot checks if its beat is already selected by another neighbour, and if so, it changes with probability $r_w = 0.7$ to another beat, selecting the one that is the least used in its neighbourhood. By setting $r_w < 1$, we avoid having robots that deterministically change their beats in a perpetual loop but can instead randomly break the symmetry.

By running this process, robots in communication range with each other distribute among the N sub-windows and as a result, the whole swarm (comprising M robots) plays on average simultaneously M/N notes per beat, creating then M/N pitch sets. This behaviour is summarised in the scheme of Figure 2. The distribution of beats, rather than uniform over the N sub-windows can be easily modified to obtain other distributions by changing the rules through which changes and allocation are selected (e.g., always keeping the third beat empty or using a Poisson distribution to have more notes played at the beginning of each bar and fewer at the end). Through these artistic design choices, the system can be configured to follow specific musical styles.

Robotic Implementation

Our framework has minimal requirements and can run on most robots employed in collective robotics research or educational projects, e.g., E-pucks or Thymio robots (Mondada et al., 2009, 2017). More precisely, to run our framework, robots must be capable of playing audio sound, exchanging small messages with each other, and (possibly) moving. Thus any robot equipped with a speaker, a communication system (e.g., Bluetooth or infrared transceiver), and wheels can run our framework. In our experiments, we implemented our framework on simple robots, initially designed for the art-science exposition Choeur Synthétique (Alkilabi et al., 2022). These robots are disks with a diameter of 30 cm with a 3D silhouette of a human head mounted on top, see Figure 3. These robots can move at a speed of 25 cm/s through differential-drive wheels, play sound, and broadcast infrared messages to other robots in line of sight within a short range of 50 cm.

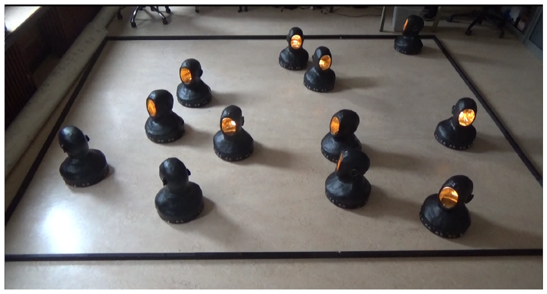
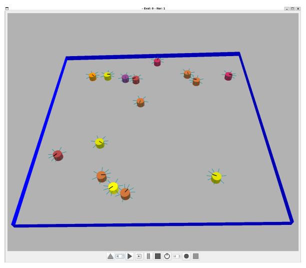


Figure 3: Robotic orchestra comprising 12 autonomous robots. A video of the experiment is available in the Supplementary Material and in the summary video `https://youtu.be/ZM-gT9RWz80`.

The communication bandwidth is relatively low as robots only exchange 1-byte messages twice per second. Given the small payload of each message (8 bits), the robot encodes information about its phase counter θ (for temporal coordination), its selected pitch (for harmonic coordination), and its selected sub-window W (for beat distribution) into two distinct messages sent in sequence twice per second. The most significant bit (MSB) of the 8-bit message indicates the message type. If the MSB is 0, the message encodes the selected pitch (4 bits for 12 possible unique pitches values) and the beat W (3 bits for 8 values). If the MSB is 1, the message encodes the phase counter - in radians - θ through 7 bits, which can however only represent 128 distinct values. Therefore, the counter θ in range $[0,2\pi]$ is approximated into those 128 levels and its accuracy is reduced. Robots also have limited computational capabilities and can only process three messages every 0.5 s. The limited communication bandwidth and the approximation mechanism are also included in our simulations. Further details on the communication board's hardware specifications, processing capabilities, and the IR transceivers' protocol can be found in Alkilabi et al. (2022).

Our modular framework is implemented as an algorithm that controls the robot behaviour. It is important to note that each robot executes the three modules (temporal coordination, harmonic consensus, and beat distribution) concurrently and independently from each other as three distinct functions of its control algorithm. Each robot operates on a musical cycle with a period of $T = 4000$ ms. Within this cycle, the robot is programmed to play its selected pitch (a note) for a duration of 1000 ms during the selected sub-


(a)

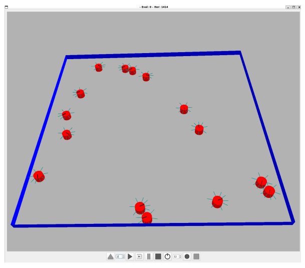

(b)

Figure 4: Two screenshots of the simulation at (a) the beginning of the experiment and (b) at the end of it. The colour of the 15 robots represents the scale they have in common.

window W. While running the musical algorithm, robots also move through an isotropic random walk, with fixed step length and turning angles chosen from a wrapped Cauchy probability distribution (Kato and Jones, 2013; Feola et al., 2023). The robot interrupts the random walk and starts an obstacle avoidance manoeuvre as soon as any of its seven proximity sensors, located around its base circumference, detects an obstacle – another robot or an arena wall – within a 12-cm range. Robots also use minimal memory to run their algorithm. They process up to three messages every 500 ms. Information regarding neighbours' phase counters θ and beats W is used immediately to update their state. Instead, information about neighbours' selected pitches is stored in a buffer that records the three most recent pitches received from the neighbours.

Experiments

We run a series of experiments with swarms of simulated and real robots. The simulations are run with a simple ad-hoc simulator (see Figure 4) that we designed to test system performance with a large number of repetitions in different configurations. More precisely, we ran 100 simulations per condition with swarms of 5, 15, and 30 robots. Instead, with the real robots, we only ran one experiment with 12 robots, which has the goal of demonstrating the validity of our results and the concrete possibility of porting our framework onto real robots.

The simulation updates the robot state every 100 ms. The behaviour and characteristics (e.g., sensors, actuators) of the simulated and real robots are kept as similar as possible, with the exception of the messaging frequency. While the real robots can send a message every 500 ms and read up to a maximum of 3 messages each 500 ms, in our simulations the robots send a message every simulated step (100 ms) and receive a maximum of 3 messages each 100 ms. This difference allowed us to test our framework under different communication constraints, having a more reactive simulated system and a slower, yet effective, real-robot system.

Initialisation and setup. The robots move in a squared flat environment enclosed by four walls. The environment is sized $4 \times 4\,\text{m}^2$ in the real-robot experiments and $6 \times 6\,\text{m}^2$ in the simulations (the simulation environment is larger to accommodate larger swarms). At the start of each experiment, each robot is initialised with the following parameters chosen uniformly at random: its position and orientation, its selected pitch in range [1,88], and its phase counter θ in range [0,2π]. The beat sub-window W is set to 0 for every robot to avoid starting with beats that are already uniformly distributed but test the correct functioning of our algorithm.

Metrics. We define three metrics to quantify the performance of each of the three independent modules that have been designed to achieve temporal coordination, harmonic consensus, and local distribution of beats.

Phase synchrony $\Delta\Theta(t)$ measures the phase synchronisation among all the robots at time t as the mean normalised absolute difference between robots' phase counters. Two robots are in maximum asynchrony when their phase is at the opposite values (e.g., robot i's phase counter is $\theta_i(t) = 0$ and robot j's $\theta_j(t) = \theta_i(t) + \pi = \pi$), hence the maximum phase difference between two robots is π. We compute the phase synchrony $\Delta\Theta$ as

$$\Delta\Theta(t) = \frac{2}{M(M-1)} \sum_{i=1}^{M} \sum_{j=i+1}^{M} (|\theta_i(t) - \theta_j(t)| \mod \pi) \tag{3}$$

where M is the number of robots. The initial factor normalises the value of $\Delta\Theta(t)$ in the range [0,1] with 0 indicating perfect synchronisation and 1 maximum asynchrony.

Harmonic agreement $H(t)$ measures the performance of the swarm in reaching an agreement on a common musical scale. The harmonic agreement H is defined as the proportion of robots whose pitches belong to – at least – a single common scale. The values of $H(t)$ are in the range $[1/M, 1]$, with $1/M$ indicating that every robot has a pitch belonging to a different scale from the others, and 1 indicating complete consensus on the same musical scale.

Evenness $\epsilon(t)$ measures how uniformly distributed robots' beats W are across the $N = 4$ sub-windows. If we denote $n_i(t)$ as the number of robots in the i-th sub-window at time t, the proportion of robots in each window is $p_i(t) = \frac{n_i(t)}{M}$, for $i = \{1, 2, 3, 4\}$ (where M is the total number of robots). The evenness $\epsilon(t)$ is calculated as the normalised variance of these proportions:

$$\epsilon(t) = \frac{\sigma^2(p_1(t), p_2(t), p_3(t), p_4(t))}{\sigma^2_{worst} - \sigma^2_{best}} \tag{4}$$

where $\sigma^2(p_1(t), p_2(t), p_3(t), p_4(t))$ is the variance of the proportions, and σ^2_{worst} and σ^2_{best} represent the variances in the worst and best-case scenarios, respectively. The measure has a value approaching 0 for uniform distributions and approaching 1 for highly uneven distributions.

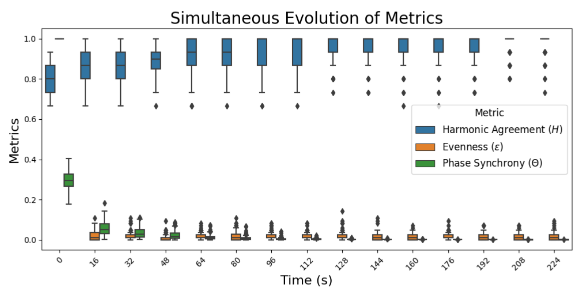


Figure 5: Concurrent evolution over time of the three metrics indicating the swarm performance of each of the three independent modules for 100 simulations with 15 robots.

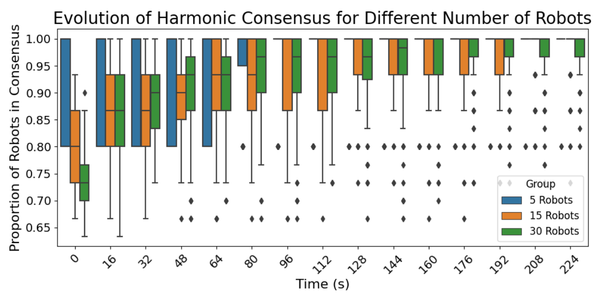


Figure 7: Harmonic agreement (i.e., largest proportion of robots with a common scale) for swarms with different sizes (100 runs per condition).

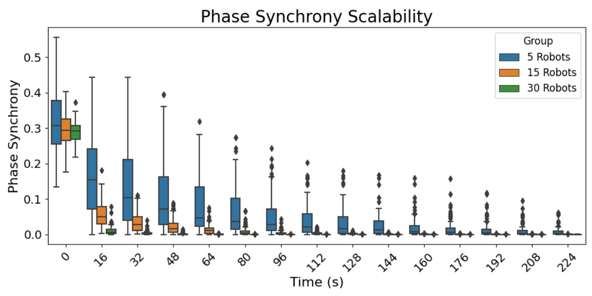


Figure 6: Phase synchrony for swarms with different sizes (100 runs per condition).

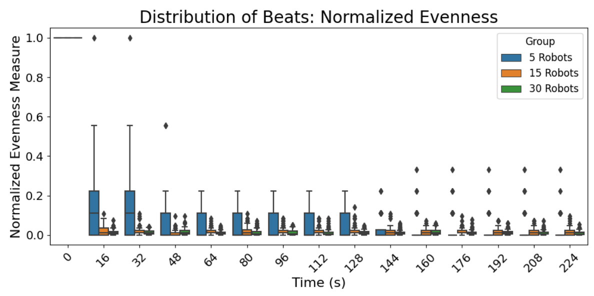


Figure 8: Evenness of the distribution of beats for swarms with different sizes (100 runs per condition).

Simulation results

One important aspect of our framework is its ability to run in parallel three modules (algorithm functions) that allow the swarm to achieve in parallel and independently the three subgoals of temporal coordination, harmonic consensus, and local distribution of beats. Figure 5 shows the three metrics $\Delta\Theta(t)$, $H(t)$, and $\epsilon(t)$ over time computed on 100 runs in a swarm of 15 simulated robots. In a relatively short time, the robots obtain good performance with respect to all three metrics by synchronising in time, establishing a harmonic consensus, and distributing their beats uniformly.

We further investigate the scalability of our system by reporting the results for swarms of 5, 15, and 30 simulated robots (average of 100 runs in each condition). Figure 6 shows the change in phase synchrony $\Delta\Theta(t)$ over time for the three simulated swarms. The results show that all swarms approach high levels of synchrony and most of them do it in a relatively short time (i.e., less than one minute). Only the smallest swarm, comprising 5 robots, is relatively slow in reaching a global consensus. This is due to lower connectivity among robots because the robot density (i.e., robots per square metre) is lower and thus, message exchange happens less frequently. The higher the robot density is, the quicker global synchronisation is achieved.

Figure 7 shows how harmonic agreement $H(t)$ generally increases over time for all tested swarm sizes. In this case, agreement is reached more rapidly in smaller swarms as there are fewer robots and therefore it is also statistically easier to have a common scale to which all selected pitches belong. Nevertheless, after four minutes, in most runs the swarm has reached a consensus on a suitable major scale.

Figure 8 shows that the robots can distribute uniformly among the four beats sub-windows, in this way, avoiding that all robots play concurrently. Similar to the synchronisation measure, smaller swarms are slower in distributing the beats, possibly due to lower connectivity among robots.

We can also qualitatively visualise the correct functioning of our framework in Figure 9 which shows two representative screenshots from a simulation with 15 robots. Figure 9a shows that at the beginning of the simulation, the robots play notes asynchronously (horizontal bars are not vertically aligned with each other) and mostly at the same time (horizontal bars are not uniformly distributed horizontally). Instead, Figure 9b shows that notes are played in synchrony and more uniformly spread on the horizontal space.

We also recorded a set of representative videos of simulation runs where it is possible to hear how the music generated by the robots evolves over time by reaching synchrony and harmonic coherence, and, in our opinion, getting musically better as the simulation progresses. The videos are available as Supplementary Material of this paper and in the summary video `https://youtu.be/ZM-gT9RWz80`.

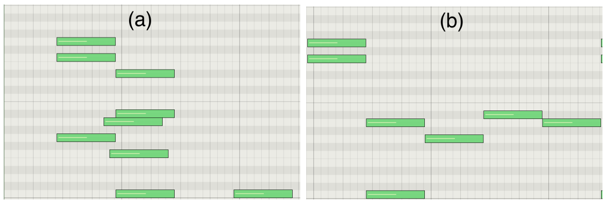


Figure 9: Visualisation of musical output for 15 robots. Each line indicates a different pitch and the green horizontal rectangles are the notes placed for 1 second on the different pitches over time. (a) At the beginning of the simulation the system is desynchronised with poorly distributed beats. (b) At the end of the simulation, the system is synchronised with uniformly distributed beats.

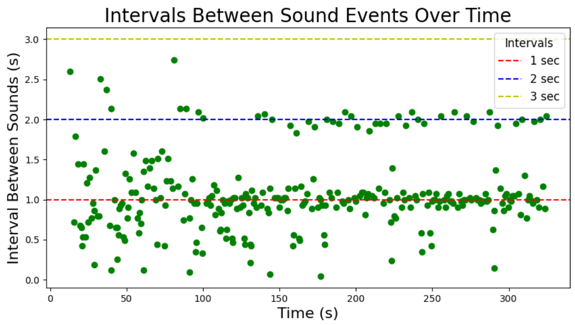


Figure 10: Evolution of intervals between two consecutive sounds in the audio of the experiment with the real robots.

Listening to the generated music in these simulations, we can appreciate how musical complexity increases with the number of robots. For example, with $M = 5$ robots and $N = 4$ beats, on average $M/N = 1.25$ notes are played at the same time and therefore we obtain a quite simple melody. Differently, with $N = 30$ robots, we obtain about $M/N = 7.5$ simultaneous pitches per beat.

Real-Robots Results

To validate the transferability of the results and produce an actual robot performance, we run one experiment with a swarm of 12 real robots. Figure 3 shows a screenshot of the video of the experiment which is available in the Supplementary Material of this paper. The red LED on top of the black head indicates the start of the 4-second period (i.e., when the phase counter $\theta_i(t) = 0$). It is interesting to see that robots can be activated asynchronously as anyway synchrony will emerge in a relatively short time (all red LED blink at approximately the same time), however, the sound that they produce is distributed consecutively throughout the four 1-second beats sub-windows. Additionally, as robots adapt to what they perceive locally, the video suggests that robots can be potentially added or removed at runtime (in our experiment, by activating them at different times).

To formally validate the experiment, we extracted the audio from the video, obtained the sound peaks, and analysed their frequency. Although the parameters of this audio processing have not been optimised, Figure 10 shows how the sound points are initially randomly spread in time (vertical axis) and after a couple of minutes, the points cluster around the 1-second and 2-second axes, indicating that the swarm as a whole mostly plays sounds with the desired structure, i.e., every 1 or 2 seconds.

Discussion and Conclusion

This paper proposes a modular framework for generating music through a robot swarm. The generated music is the emergent result of repeated interactions among simple robots which only play one note. The simplicity – and thus low requirements – of the robotic platform facilitates its potential employment in several artistic, educational, and scientific applications. In addition to being in line with the principles of accessible science thanks to its low cost and wide applicability, the results of our simulations and robot experiments indicate that this framework is a promising basis for a robust public engagement robotic orchestra.

While we report promising results, future research could exploit the full potential of this modular bottom-up framework for collaborative music generation. The modular architecture of our framework facilitates the integration of new mechanisms without disrupting existing modules, for example, new independent modules could use virtual forces of attraction and repulsion between robots to improve note harmony (robots that play notes sounding well together are attracted to each other), or dynamically change instruments to create suitable combinations. Future research can also investigate the reciprocal relationship between musical output and robot movement, where slow-moving robots with quasi-static neighbourhoods could lead to a stabilisation of the music, which can be perturbed by fast-moving robots with rapidly changing neighbourhoods. The complexity of musical output can be increased through new modules that include structural elements of musical composition, such as note hierarchies where interrelations between notes determine the relative positions in the beat sequence. Ultimately, the aim is to reach computational creativity where the musical output and performance of our swarm orchestra can be considered the result of a creative process. To achieve this goal, the framework needs to be expanded to include modules that create a more complex structure.

Acknowledgements

A.R. acknowledges support from DFG under Germany's Excellence Strategy - EXC 2117 - 422037984. The authors thank Timoteo Carletti for providing valuable feedback on this manuscript.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. *arXiv*, 2303.08774.

Albin, A., Weinberg, G., and Egerstedt, M. (2012). Musical abstractions in distributed multi-robot systems. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 451–458. IEEE.

Alkilabi, M., Eeckhout, A., Vitturini, M., du Chastel, M., Warzée, M., Rousseaux, J.-Y., Hubermont, A., Carletti, T., and Tuci, E. (2022). Choeur synthétique: An art installation based on swarm robotics. In *Swarm Intelligence Proceedings of ANTS 2022*, volume 13491 of *LNCS*, page 284–291, Cham, Switzerland. Springer.

Baronchelli, A. and Diaz-Guilera, A. (2012). Consensus in networks of mobile communicating agents. *Physical Review E*, 85(1):016113.

Blackwell, T. (2007). Swarming and music. In *Evolutionary Computer Music*, page 194–217. Springer, London, UK.

Brooks, T., Peebles, B., Holmes, C., DePue, W., Guo, Y., Jing, L., Schnurr, D., Taylor, J., Luhman, T., Luhman, E., Ng, C., Wang, R., and Ramesh, A. (2024). Video generation models as world simulators. *https://openai.com/research/video-generation-models-as-world-simulators*.

Camazine, S., Deneubourg, J.-L., Franks, N. R., Sneyd, J., Theraula, G., and Bonabeau, E. (2001). *Self-organization in biological systems*. Princeton University Press.

Colton, S., Wiggins, G. A., et al. (2012). Computational creativity: The final frontier? In *Proceedings of the 20th European Conference on Artificial Intelligence*, ECAI'12, pages 21–26. IOS Press.

Dorigo, M., Theraulaz, G., and Trianni, V. (2021). Swarm robotics: Past, present, and future [point of view]. *Proceedings of the IEEE*, 109(7):1152–1165.

Eigenfeldt, A. and Kapur, A. (2008). An agent-based system for robotic musical performance. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 144–149, Genoa, Italy.

Feola, L., Sion, A., Trianni, V., Reina, A., and Tuci, E. (2023). Aggregation through adaptive random walks in a minimalist robot swarm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '23, pages 21–29, New York, USA. ACM.

Hamann, H. (2018). *Swarm robotics: A formal approach*. Springer, Cham, Switzerland.

Kato, S. and Jones, M. C. (2013). An extended family of circular distributions related to wrapped cauchy distributions via brownian motion. *Bernoulli*, 19(1):154–171.

Krzyzaniak, M. (2022). How to build pipe organ robots. In *Proceedings of the 17th International Audio Mostly Conference*, AM '22, pages 121–128, New York, USA. ACM.

Krzyżaniak, M. (2021). Musical robot swarms, timing, and equilibria. *Journal of New Music Research*, 50(3):279–297.

Kuramoto, Y. (1975). Self-entrainment of a population of coupled non-linear oscillators. In *International Symposium on Mathematical Problems in Theoretical Physics*, volume 39 of *LNP*, pages 420–422, Heidelberg, Germany. Springer.

Mamykina, L., Candy, L., and Edmonds, E. (2002). Collaborative creativity. *Communications of the ACM*, 45(10):96–99.

Mannone, M., Seidita, V., and Chella, A. (2023). The sound of swarm. auditory description of swarm robotic movements. *ACM Transactions on Human-Robot Interaction*, 12(4):1–27.

Martins, J. M. and Miranda, E. R. (2007). Emergent rhythmic phrases in an a-life environment. In *Workshop on Music and Artificial Life (MusicAL 2007) at ECAL 2007*, pages 10–14.

McLurkin, J. (2022). The swarm orchestra: Temporal synchronization and spatial division of labor for large swarms of autonomous robots. Technical report, CSAIL, Massachusetts Institute of Technology, Cambridge, MA, USA. Retrieved from `https://groups.csail.mit.edu/mac/projects/amorphous/6.978/final-papers/jamesm-final.pdf`.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.-C., Floreano, D., and Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions*, volume 1, pages 59–65. IPCB: Instituto Politécnico de Castelo Branco.

Mondada, F., Bonani, M., Riedo, F., Briod, M., Pereyre, L., Retornaz, P., and Magnenat, S. (2017). Bringing robotics to formal education: The Thymio open-source hardware robot. *IEEE Robotics & Automation Magazine*, 24(1):77–85.

Nymoen, K., Chandra, A., Glette, K., and Torresen, J. (2014). Decentralized harmonic synchronization in mobile music systems. In *2014 IEEE 6th International Conference on Awareness Science and Technology (iCAST)*, pages 1–6. IEEE.

Perez-Diaz, F., Zillmer, R., and Groß, R. (2015). Firefly-inspired synchronization in swarms of mobile agents. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '15, page 279–286, Richland, SC. IFAAMAS.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. *arXiv*, 2204.06125.

Saunders, R. and Bown, O. (2015). Computational social creativity. *Artificial life*, 21(3):366–378.

Tatar, K. and Pasquier, P. (2019). Musical agents: A typology and state of the art towards musical metacreation. *Journal of New Music Research*, 48(1):56–105.

Trianni, V., De Simone, D., Reina, A., and Baronchelli, A. (2016). Emergence of consensus in a multi-robot network: from abstract models to empirical validation. *IEEE Robotics and Automation Letters*, 1(1):348–353.

Yang, L.-C., Chou, S.-Y., and Yang, Y.-H. (2017). Midinet: A convolutional generative adversarial network for symbolic-domain music generation. *arXiv*, 1703.10847.