Interactive Sonification of 3D Swarmalators

Pedro Lucas
RITMO, Department of Informatics
University of Oslo
Oslo, Norway
pedroplu@ifi.uio.no

Stefano Fasciani
Department of Musicology
University of Oslo
Oslo, Norway
stefano.fasciani@imv.uio.no

Alexander Szorkovszky
RITMO, Department of Informatics
University of Oslo
Oslo, Norway
alexansz@ifi.uio.no

Kyrre Glette
RITMO, Department of Informatics
University of Oslo
Oslo, Norway
kyrrehg@ifi.uio.no

ABSTRACT

This paper explores the sound and music possibilities obtained from the sonification of a swarm of coupled oscillators moving in a virtual space called “Swarmalators”. We describe the design and implementation of a *Human-Swarm Interactive Music System* based on the 3D version of the Swarmalator model, which is used for signal analysis of the overall sound output in terms of *scalability*; that is, the effect of varying the number of agents in a swarm system. We also study the behaviour of autonomous swarmalators in the presence of one user-controlled agent, which we call the *interactive swarmalator*. We observed that sound frequencies barely deviate from their initial values when there are few agents, but they diverge significantly in a highly dense swarm. Additionally, with the inclusion of the *interactive swarmalator*, the group's behaviour tends to adjust towards it. We use these results to explore the potential of swarmalators in music performance under various scenarios. Finally, we discuss opportunities and challenges to use the Swarmalator model for sound and music systems.

Author Keywords

Human-Swarm Interactive Music Systems, Multi-Agent Systems, Swarm Intelligence, Swarmalators, Scalability

CCS Concepts

•Applied computing → Sound and music computing; •Computing methodologies → Modeling and simulation; •Human-centered computing → *Interactive systems and tools;*

1. INTRODUCTION

A human-swarm Interactive Music System (IMS) uses groups of agents, moving and interacting with each other according to simple rules, to generate novel sounds and musical patterns. Such systems are well suited to music improvisation due to their highly interactive and complex nature [2].

Our Human-Swarm IMS is based on the *Swarmalators* model [16], which describes a special type of coupled oscillator that moves in space, explained in detail in section 2.2. Sound and music applications using coupled oscillators, based on the widely known *Kuramoto* model [9], have been developed previously [10, 15, 11, 12, 17]. Still, the spatial properties of these oscillators have not been researched in the IMS context so far. We believe that having both spatial and temporal characteristics increases the possibilities for human interaction in the context of IMSs.

As such, we present the design and implementation for an IMS based on the 3D version of the Swarmalator model, in which the user can interact through the model's parameters and be one of the swarmalators. Moreover, due to the complexity of the behaviour resulting from the swarm dynamics, we are interested in analysing the sound output, known as “sound swarming”, that emerges from the sonification of the swarmalators through simple mapping rules. We intend to discover emergent properties from the individual interactions when we increase the number of agents (swarm *scalability*), as well as when one of the agents is controlled by the user. This paper addresses the following questions:

- *How can we implement a human-swarm IMS based on the Swarmalator model?*
- *What is the effect on the sound output when the number of agents increases?*
- *How do autonomous swarmalators behave in the presence of one interactive swarmalator that is controller by a user?*
- *What are the music performance possibilities of this IMS?*

Furthermore, the analysis of sound swarming as a macro-product contributes to the field of multi-agent systems, whose research goal is to connect the micro-scale behaviour with macro-scale properties and vice versa [20].

This paper is organized as follows: Section 2 illustrates the background regarding a Human-Swarm IMS and the Swarmalator model, section 3 lists works related to this research, section 4 describes the IMS based on swarmalators, section 5 shows the results for the sound analysis over recordings using the system, section 6 discusses our findings and implications for human-swarm interaction, as well as future work, and finally section 7 presents conclusions.

2. BACKGROUND

2.1 Human-Swarm IMS

A *human-swarm Interactive Music System (IMS)* is an improvisational system allowing users to interact with a swarm of *self-organised* artificial agents exhibiting *emergence* in a sound and music context [13]. From an individual perspective, *self-organization* refers to the local interaction between simple individuals; thus, as a collective intelligence, *emergent* properties can be produced from these interactions [3].

An agent in this type of swarm system can be related to a sound or music element that belongs to a particular perceptual time-scale, which is the time duration of the material or its complexity. In this work, the time-scale corresponds to *musical notes or sound objects* [2]. Particularly, every

NIME'24, 4–6 September, Utrecht, The Netherlands.

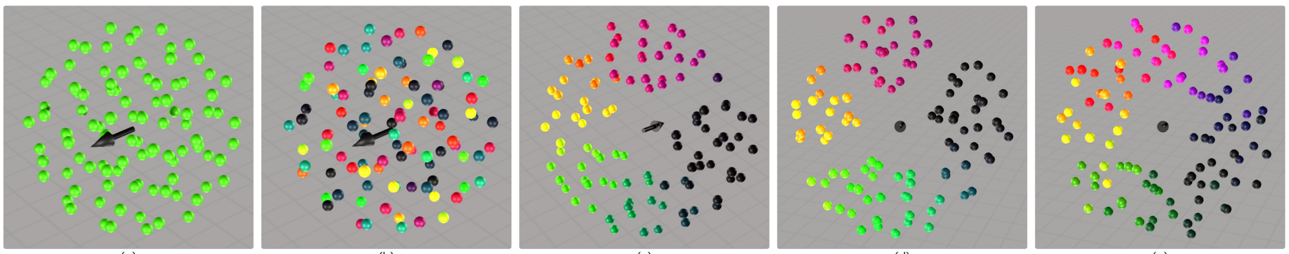


Figure 1: Swarmalators' States for 100 agents (step size $\delta t = 0.01$, speed = 4). (a) Static sync state $(J, K) = (0.1, 1)$, (b) Static async state $(J, K) = (0.1, -1)$, (c) Static phase wave state $(J, K) = (1, 0)$, (d) Splintered phase wave $(J, K) = (1, -0.1)$, (e) Active phase wave $(J, K) = (1, -0.75)$. An agent is a 3D sphere in space whose colour represents its current phase value. The black arrow is the normal vector of the plane that best fits the entire swarm from the centre of it.

agent is a sound oscillator played at a specific frequency as described in section 4.1.3. We call "sound swarming" the emergent collective result generated by the sound of individual agents.

For a Human-Swarm IMS, we usually associate the swarm dynamics behaviour with sound through mapping rules. In our case, the swarm dynamics is given by the *Swarmalators* model explained in the next section 2.2 and the mapping rules are defined in 4.1.3. This paper outlines the essential mechanisms required for facilitating this sonification task.

2.2 Swarmalators

The term "Swarmalator" combines "swarm", referring to a group of agents, and "oscillator", which is a system that exhibits periodic behaviour. Thus, the *Swarmalators* are oscillators whose phase and spatial dynamics are coupled. This model is proposed by O'Keeffe *et al.* [16] in which the spatial dynamics are explored in the 2D and 3D space. We use the 3D version given by the differential equations (1) and (2), for $i = 1, ..., N$, where N is the number of swarmalators, $\mathbf{x}_i = (x_i, y_i, z_i)$ is the position of a swarmalator i, and θ_i is its phase. There are two adjustable parameters (J, K), K is the *"phase coupling strength"*, and J measures the *"extent to which phase similarity enhances spatial attraction"*.

$$\dot{\mathbf{x}}_i = \frac{1}{N} \sum_{j \neq i}^{N} \left[\frac{\mathbf{x}_j - \mathbf{x}_i}{|\mathbf{x}_j - \mathbf{x}_i|} (1 + J \cos(\theta_j - \theta_i)) - \frac{\mathbf{x}_j - \mathbf{x}_i}{|\mathbf{x}_j - \mathbf{x}_i|^3} \right] \quad (1)$$

$$\dot{\theta}_i = \frac{K}{N} \sum_{j \neq i}^{N} \frac{\sin(\theta_j - \theta_i)}{|\mathbf{x}_j - \mathbf{x}_i|} \quad (2)$$

This work uses these differential equations in a discrete context to update each agent parameter within a small time frame δt with speed v. Thus, the update rule for the position and phase of the i-th swarmalator is given by (3) and (4) respectively.

$$\mathbf{x}_i(t + \delta t) = \mathbf{x}_i(t) + v\dot{\mathbf{x}}_i(t)\delta t \quad (3)$$

$$\theta_i(t + \delta t) = \theta_i(t) + v\dot{\theta}_i(t)\delta t \quad (4)$$

The internal oscillator for the swarmalator i can use $\mathbf{x}_i$ and θ_i depending on the specific application. In this work, we present a set of mapping rules for the *sound output* using these parameters in section 4.1.3, and we define an internal oscillator for *visualization* and *user interaction* purposes, as explained in section 4.1.2. We use the parameters (J, K, v) as user input to change the swarm behaviour.

In [16], five states were found and studied according to specific values of J and K. These states are: ***a.*** *static sync,* ***b.*** *static async,* ***c.*** *static phase wave,* ***d.*** *splintered phase wave*, and ***e.*** *active phase wave*. Their visual representation, rendered in the system described in this paper, is shown in Figure 1 and in a video[1]. The states and their (J, K) values are given in the image caption.

3. RELATED WORK

According to the *Web of Science* research database[2], there are several works derived from the Swarmalator model proposed by O'Keeffe *et al.* [16] involving fields different than physics (e.g. computer science, robotics, biology, neuroscience, etc.) with no specific works related to sound and music systems so far. Nevertheless, in the *Multimodal Interaction* field, Chakraborty and Timoney [6] investigated the swarmalators, together with the Kuramoto model [9], as predictors for musical phase synchronization amongst an ensemble having an oscillator representing a person. The authors highlight the importance of these models for the design of group musical interfaces in Human-Computer Interaction (HCI). Outside academia, we can find software components for creative proposes as the modules for *Ableton Live* known as *Swarmalators-t*[3] and *Swarmalators-n*[4] based on the 2D Swarmalator model with several sound and music mapping options.

However, works closely related to coupled oscillators for sound and music systems explore several properties of synchronization models. For instance, in [10], a population of non-linear oscillators are used to generate rhythms portraying *protomusicality*, which is a rhythmic but not necessarily creatively musical behaviour. Coupled-oscillator networks are employed for producing and analyzing sound in [11] and [12], suggesting that the generated product is best suited for types of music that are more oriented toward experimental music practices such as procedural, minimalist, and drone musical genres. Moreover, in [17], coupled oscillators are modelled for musical sequencing and synthesis, and their timbral effects are analysed.

From the "swarm" perspective, we can find references and suggestions for developing Human-Swarm IMSs in [13]. Besides, there are several works in the NIME community based

[1]`https://youtu.be/sDsLuTV6qLw` (accessed May 6, 2024))
[2]`https://www.webofscience.com/wos/woscc/analyze-results/52a8e2f7-0ef2-42fc-a0d5-56a8106618e7-ca9794f2` (accessed May 6, 2024))
[3]`https://dillonbastan.com/store/maxforlive/index.php?product=swarmalators-t` (accessed May 6, 2024))
[4]`https://dillonbastan.com/store/maxforlive/index.php?product=swarmalators-n` (accessed May 6, 2024))

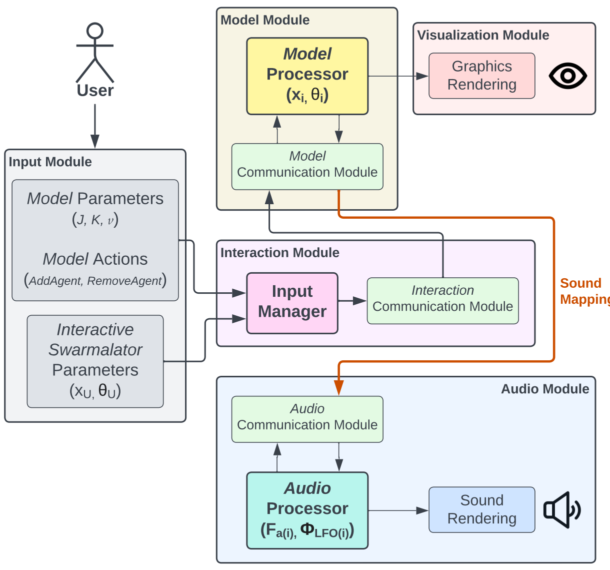

Model Module
Model Processor (x_i, θ_i)
Model Communication Module
Visualization Module
Graphics Rendering
User
Input Module
Model Parameters (J, K, ν)
Model Actions (AddAgent, RemoveAgent)
Interactive Swarmalator Parameters (x_U, θ_U)
Interaction Module
Input Manager
Interaction Communication Module
Sound Mapping
Audio Module
Audio Communication Module
Audio Processor ($F_{a(i)}$, $\Phi_{LFO(i)}$)
Sound Rendering

Figure 2: *System Architecture*. The control data flows from the users' input through the *interaction module* that feeds the *model* to modify the swarm properties and get the audio and visual outputs. The *sound mapping* link, illustrated in orange, emphasizes the relationship between the *model* parameters and the *sound* parameters.

on swarm intelligence and multi-agent systems. Some of these systems are software-based with agents hidden from the user, as in [4] and [18], which uses agent-based programming paradigms. Other works are more tangible, such as the music robotic systems found in [19, 7, 8]. Additionally, interaction with music swarm systems through global properties has been proposed; for instance, Burtner [5] explores the technique of *perturbation*, meaning the use of mitigated influence from one agent on the others, a concept that fits the purpose of the *interactive swarmalator* presented in section 4.1.2. Bisig and Schiesser [1] also use this technique implicitly through a physical control for a virtual swarm; furthermore, it can be seen as a physical-virtual music swarm application, a category that is currently explored with new immersive technologies (e.g. extended reality (XR), spatial audio, etc.) as the system proposed in [14].

This work considers both the "oscillators" and "swarm" perspectives through the 3D version of the Swarmalator model in a Human-Swarm IMS. Moreover, we explore ways for human interaction with autonomous and controllable agents in the context of sound and music, and demonstrate how these methods impact the resulting sound output as we vary the swarm size.

4. THE 3D SWARMALATORS SYSTEM

4.1 System Design

4.1.1 System Overview

The architecture depicting the main modules that compose this system is illustrated in Figure 2. The actions from the user are captured by the *Input Module* that sends the control data to the *Interaction Module*, which is in charge of transferring the parameters to the *Model Module* where the swarmalators' equations update the phase and position of every agent. The *Model Module* informs the *Visualization Module* when the agents' parameters change so that it renders the corresponding graphics as described in section 4.1.4. Moreover, this *Model Module* communicates the changes constantly to the *Audio Module*, which generates the sound from the mapping between the swarmalators' dynamics and the corresponding audio parameters as explained in section 4.1.3.

4.1.2 Agent and Swarm Behaviour

An agent can be added or removed by the user. When an agent is added to the environment, its phase θ and position $\mathbf{x}$ are assigned with random values; moreover, the agent needs to be *aware* of the rest of agents (by identifying its neighbours initially) so that it can access their phases and positions, and update its own parameters when staying in a `Moving and Pulsing` state. We identify these individuals visually as *moving coloured spheres* as depicted in Figure 1.

The pulsating behaviour of an agent i is given by an internal oscillator $S_i(t)$ defined by (5). This includes the phase θ_i from the Swarmalator model presented in section 2.2, t is the current value of a global timer used by all agents running during the system execution, and f an arbitrary frequency. $S_i(t)$ is relevant for both: the *visualization*, as explained in section 4.1.4, and the *interactive swarmalator (IS)*, a user-controlled agent explained below.

$$S_i(t) = \sin(2\pi f t + \theta_i(t)) \quad (5)$$

We introduce a user-controlled agent called the *interactive swarmalator (IS)*, with phase θ_U and position $\mathbf{x}_U$, represented visually by a cube as shown in Figure 3. It does not update its phase and position autonomously from others, as the Swarmalator model does; instead, these parameters are given by the user *interacting through the visualization* as follows:

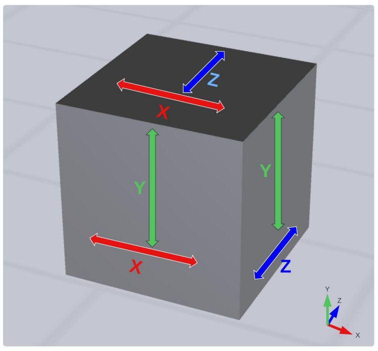


Figure 3: *Interactive Swarmalator (IS)*. A user can move this agent and change $\mathbf{x}_U$ in the plane where a face is selected (e.g. selecting and holding down the front face with the left mouse button allows the agent to be moved across the XY plane). Applying a selection with a different control (e.g. click with the right mouse button) sets the phase θ_U in its internal oscillator.

- **Phase θ_U:** We can change $\theta_U(t)$ at the moment that the user executes one concrete action over IS (e.g. in this system, it is a click with the right mouse button over the cube shape). When this action is performed on IS, the internal oscillator value $S_i|_{i=U}$ is reset to zero. This is achieved by setting the argument of the sine function to zero, that is, $2\pi f t + \theta_U = 0$. However, we want this phase in a range of $[0, 2\pi]$; therefore, $\theta_U = mod_{2\pi}(-2\pi f t)$, where $mod_{2\pi}$ denotes the modulo operation which gives the remainder of a division by 2π.

- **Position $\mathbf{x}_U$:** The user can move IS in the planes XY, XZ, and YZ as illustrated in Figure 3. That is the main reason for choosing a cube as the 3D visual (and interactive) representation of this special agent since the action is: selecting any face (e.g. holding the left mouse button) and dragging in the plane where the selected face belongs to change the position $\mathbf{x}_U$.

The *interactive swarmalator (IS)* does not execute the update rule as the autonomous swarmalators; however, the rest of agents *do* consider IS within their update mechanism. Therefore, IS is not influenced by the other swarmalators, but these others are, which is an important fact to consider when reading the results in section 5.2.

In terms of swarm behaviour, the system is represented under the *PQf+K* architecture explained in [13]. Here, we identify processes in terms of the *Swarmalators' Model (SM)*, *Audio Generation (AG)*, and the *Interactive Swarmalator (IS)*. The user interacts with the swarm through the *environment (E)* so that the *analysis (P)* module receives the parameters used by the agents, e.i, (J, K, v) for the Swarmalator model, the actions `(AddAgent, RemoveAgent)`, and $(\mathbf{x}_U, \theta_U)$ for the *interactive swarmalator*. This parameters are then used by the *patterning (f)* module to update the model and the sound mapping parameters based on the *knowledge (K)* provided by the agents' collection (the other phases and positions for one agent i); finally, the *synthesis (Q)* module generates the visual and audio outputs that return to the (E). This representation can be helpful to guide variations of the system proposed in this work and depicted previously in Figure 2.

Additionally, there are five presets the user can select to change (J, K): ***a.*** *static sync,* ***b.*** *static async,* ***c.*** *static phase wave,* ***d.*** *splintered phase wave*, and ***e.*** *active phase wave*. When a preset is recalled, (J, K) are updated accordingly

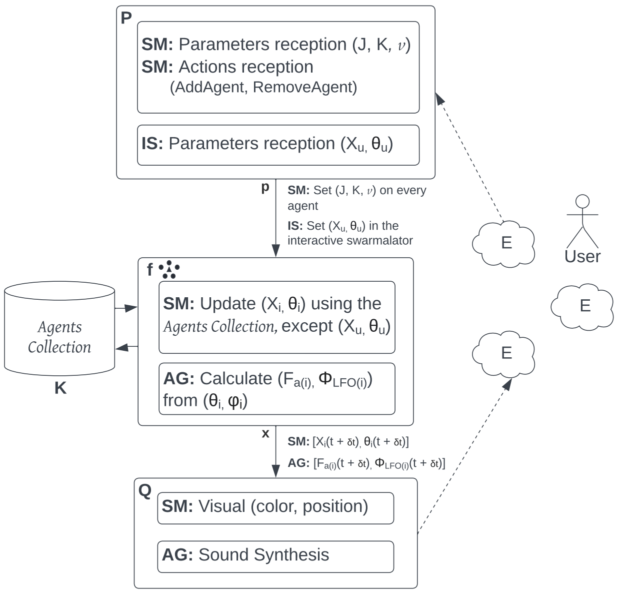

P
SM: Parameters reception (J, K, ν)
SM: Actions reception
(AddAgent, RemoveAgent)
IS: Parameters reception (Xu, θu)
p
SM: Set (J, K, ν) on every agent
IS: Set (Xu, θu) in the interactive swarmalator
E
User
E
E
f
SM: Update (Xi, θi) using the Agents Collection, except (Xu, θu)
AG: Calculate (Fa(i), ΦLFO(i)) from (θi, φi)
Agents Collection
K
x
SM: [Xi(t + δt), θi(t + δt)]
AG: [Fa(i)(t + δt), ΦLFO(i)(t + δt)]
Q
SM: Visual (color, position)
AG: Sound Synthesis

Figure 4: System representation in the *PQf+K* architecture. SM stands for *Swarmalator Model*, IS for *Interactive Swarmalator*, and AG for *Audio Generation*.

4.1.3 Sound Mapping and Synthesis

There is one audio signal for every swarmalator added to the environment. This signal is a `pulse wave` obtained by a `sine oscillator` with a sound frequency $F_{a(i)}$ in a range $[F_{a\_min}, F_{a\_max}]$, modulated in amplitude by a `low-frequency oscillator (LFO)` of frequency $f_{LFO(i)}$. The LFO works as a sine wave with the negative part clipped to zero. This signal can be visualized in the sound generation section of Figure 5, which also describes the mapping strategy.

To generate a signal i for a swarmalator i, we take the phase θ_i and the angle φ_i. To calculate this angle, we need to know the displacement vector between the swarm centre and the swarmalator position $\mathbf{x}_i$, and a reference vector as our `right vector` from the swarm centre as shown in Figure 5. The angle between those vectors is φ_i. Then, we apply the following two mapping rules:

- $\theta_i \rightarrow F_{a(i)}$**:** The phase θ_i given directly by the model is associated with the sound frequency $F_{a(i)}$. As θ_i can be within the range $[0, 2\pi]$, we establish a new range that maps $[0, 2\pi] \rightarrow [F_{a\_min}, F_{a\_max}]$.

- $\varphi_i \rightarrow \Phi_{LFO(i)}$**:** The angle φ_i is associated with the LFO's phase $\Phi_{LFO(i)}$ (normalized between 0 an 1). This mapping rule has a special characteristic since it relies on a *swarm property* (global centre), which means that the individual sounds are influenced directly by the collective. The mapping range is $[0, 360°] \rightarrow [0, 1]$.

The $\theta_i \rightarrow F_{a(i)}$ rule was selected for easier identification of the changes in phase from the model through pitched sounds. The $\varphi_i \rightarrow \Phi_{LFO(i)}$ is an arbitrary rule to give some sound variation to the output, making it perceptually (and relatively) interesting. Of course, there is a whole range of possibilities for mapping, and we encourage the readers to explore more of them.

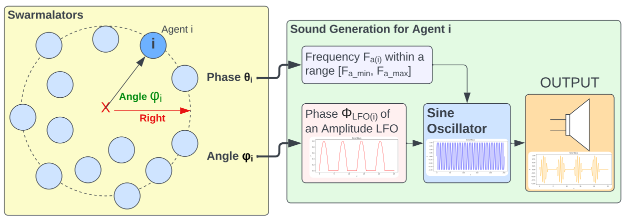

Swarmalators
Agent i
i
Angle φi
Right
Phase θi
Angle φi
Sound Generation for Agent i
Frequency Fa(i) within a range [Fa_min, Fa_max]
Phase ΦLFO(i) of an Amplitude LFO
Sine Oscillator
OUTPUT

Figure 5: Sound Mapping for a swarmalator i. The sound generation depends on the Swarmalator model through the rules $\theta_i \rightarrow F_{a(i)}$ and $\varphi_i \rightarrow \Phi_{LFO(i)}$.

The resulting sound output is the sum of N signals that go into one only mix, which is the "sound swarming" result. This global output can quickly increase in amplitude as the number of agents increases; thus, we can attenuate this amplitude by applying a rule that decreases the audio gain as the number of agents increases. In this system, we use a `gain attenuator` based on a custom curve that maps the number of agents to obtain the gain in decibels (dB). An example of this strategy is depicted in the *Max 8* patch in Figure 6, which maps the number of agents in the interval $[3, 100]$ to $[0, 1]$, then it is evaluated in a *drawable decaying curve* $y = g(x)$ having $x \in [0, 1]$ and $y \in [0, 1]$ to finally get the decibels (dB) from y to a value in $[-15\ dB, -30\ dB]$. The gain range and the curve are found through trial and error testing.

4.1.4 Visualization

To illustrate the agents' behaviour visually, we represent them with a *coloured sphere* shape as shown previously in

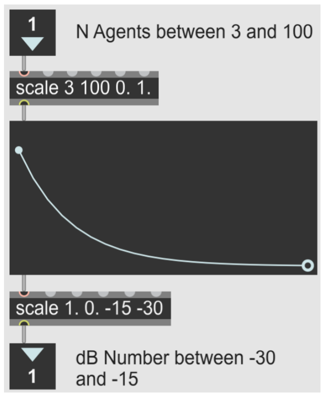


Figure 6: Gain attenuator for the "sound swarming" output. It shows how the signal gain in decibels (dB) changes based on the number of agents, using a drawable decaying curve.

Figure 1. Colours are mapped from the θ_i phase to a colour range represented by the HSV (Hue Saturation Value) scale. Each value in the scale is between 0 and 1; we keep S and V as 1 but change H according to the mapping rule $[0, 2\pi] \rightarrow [0, 1]$, giving vivid colours to represent the different phases.

Furthermore, the colour assigned to an agent i is visualized as a pulse that changes between this colour and a neutral colour, black in this case. This pulse is generated by the internal oscillator $S_i(t)$ explained in section 4.1.2. $S_i(t)$ is used for linear interpolation between both colours in a range $[0, 1]$ where 0 is black, and 1 is the assigned colour. As $S_i(t)$ can take values between -1 and 1, we map the colour interpolation using the mapping rule $[-1, 1] \rightarrow [0, 1]$.

4.2 System Implementation

The system was implemented in two environments. The *Unity*[5] game engine and *Cycling '74 Max 8*[6]. The *Model* and *Visualization* modules were implemented in *Unity*, as well as part of the *Interaction Module* corresponding to the *interactive swarmalator* control. It is possible to interact directly through the visualization by navigating the 3D scene with a computer mouse; moreover, the mouse is also used to control the *interactive swarmalator* as described in section 4.1.2. A view of the 3D rendering was shown previously in Figure 1.

The other part of the *Interaction Module*; that is, the actions for adding and removing agents (two buttons), the parameters (J, K, v) (three sliders), as well as presets to set the five swarmalators states based on these parameters (five buttons), are implemented in a *Max 8* patch. Any other parameter described previously is not changed by the user and is set directly (e.g. f, $[F_{a\_min}, F_{a\_max}]$). It is possible to interact through a MIDI controller. The *Audio Module* is also implemented within this patch.

The communication between *Unity* and *Max 8* is established through OSC messages within the same computer, running Windows 11, 64-bit, powered by a 12th Gen Intel(R) Core(TM) i7-12700H 2.30 GHz, with RAM 16 GB, and a laptop graphics processor NVIDIA GeForce RTX 3050 Ti.

5. RESULTS

[5]`https://unity.com/` (accessed May 6, 2024))
[6]`https://cycling74.com/products/max`(accessed May 6, 2024))

Table 1: General configuration for the swarmalators system.

Parameter	Value
a. static sync state (J, K)	(0.1, 1)
b. static async state (J, K)	(0.1, -1)
c. static phase wave state (J, K)	(1, 0)
d. splintered phase wave state (J, K)	(1, -0.1)
e. active phase wave state (J, K)	(1, -0.75)
speed v	4
step size (delta time) δt	0.01
frequency f (visuals and interaction)	1
$[F_{a\_min}, F_{a\_max}]$ (audio)	[50 Hz, 3000 Hz]

The analysis performed in this work used the parameters listed in Table 1. Additional configurations are detailed in the following sub-sections.

5.1 Effect of scalability on sound output

5.1.1 General Results

To observe the effect of increasing the number of agents through the different states in a resulting sound output, we recorded a long audio sample and applied a conventional analysis of the audio signal to examine how it changes over time in terms of amplitude and frequency. This recording was performed by increasing the number of agents from 3 to 100. For each number of agents, the parameters (J, K) are changed through the combinations corresponding to the five swarmalators states (*a. static sync, b. static async, c. static phase wave, d. splintered phase wave, and e. active phase wave*). Every state is recorded for 10 seconds, which means that the number of agents increases by one every 50 seconds until 100 agents are reached. The total recording time was 1 hour, 21 minutes and 40 seconds. Additionally, a video of the entire session with the corresponding visuals was recorded[7].

Figure 7 shows the time domain plot (waveform) depicting the amplitude of the sound output across the whole recording. It is possible to identify the attenuation effect in amplitude (explained in section 4.1.3), which is at its lowest at minute 30 and highest at the extrema of the signal.

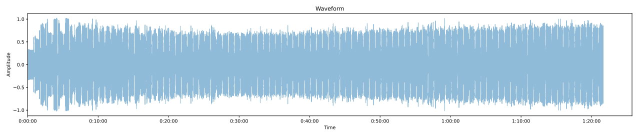


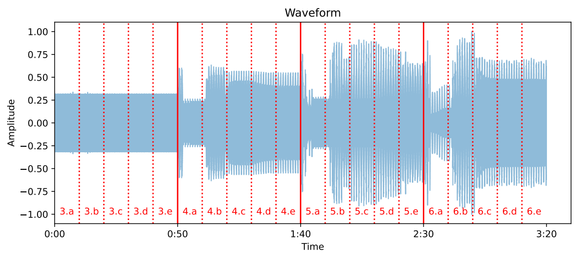
Figure 7: Waveform in the time domain regarding sound changes from 3 to 100 agents passing through the five swarmalators' states.

In the following sections 5.1.2, 5.1.3, and 5.1.4, we present the resulting waveforms and spectrograms in various segments of the recording.

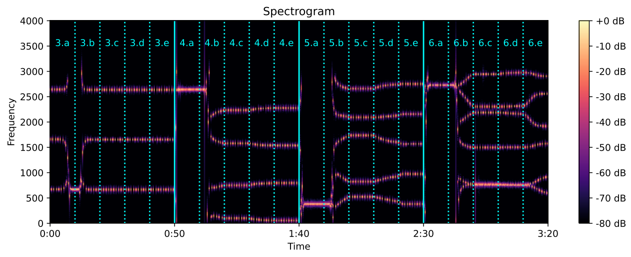
5.1.2 Effect of a low number of agents

Figure 8 shows the waveform and spectrogram of the recording for an excerpt covering from 3 to 6 agents. Note that these plots indicate the zones corresponding to different states (from a. to b.) for each agent increase. In Figure 8a, the sound waveform for this segment illustrates differences in shape and amplitude depending on the number

[7]`https://youtu.be/669TnNaJclA` (accessed May 6, 2024))



(a)



(b)

Figure 8: Sound changes from 3 to 6 agents passing through the five swarmalators' states (from a. to e.). (a) Waveform in the time domain. (b) Spectrum of frequencies over time (spectrogram).

of agents and states. Figure 8b plots the spectrogram where the changes in frequency are clear since the generated sound is a pulsed sine wave. The first 3 agents start in a random frequency, and it takes around 9 seconds to reach the *static sync state (a)*, while it is faster to change to the *static async state (b)*. Then, for the rest of the states, the 3 agents keep the same frequency similar to the initial ones before the first state change. However, as we increase the number of agents, different frequencies occur in the state zones but still do not show much variability within these zones except for the (b) zones and the 6.e (6 agents in the *active phase wave state*) zone.

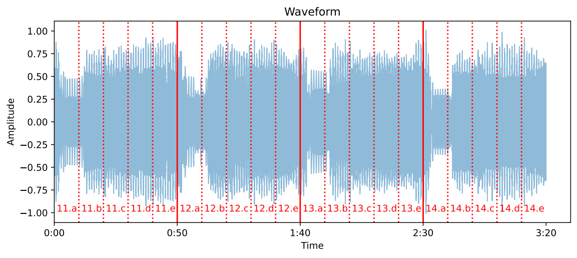
5.1.3 *Spatial movement effect*

Figure 9 illustrates a segment with agents from 11 to 14. In this particular range, the sound waveform looks similar between agents; nevertheless, the spectrogram marks clear differences in the frequency content over time. Note that, with more agents, the convergence to the *static sync state (a)* is faster; moreover, we start to see the effect of spatial motion for the *active phase wave state (e)* where frequencies intertwine with each other and appear slightly more diffused due to the coupled changes in phase and position from the model.

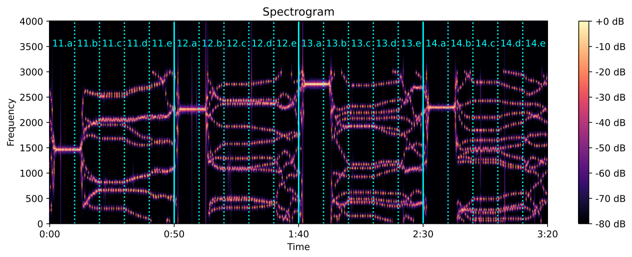
5.1.4 *Effect of a high number of agents*

The last segment of the recording with agents from 97 to 100 is shown in Figure 10. The shape of the sound waveform continues to look similar between agents, but we can notice how frequencies develop in time with a high density of agents. At this point, the *static async (b)* and the *active phase wave (e)* states look slightly similar as if they were uniformly random distributed across the frequency range of operation; however, differences are more apparent when hearing the sound output since several "frequency sweeping" effects can be perceived.

5.2 Influence of the interactive swarmalator

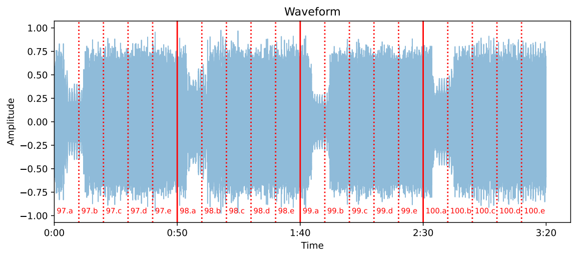


(a)

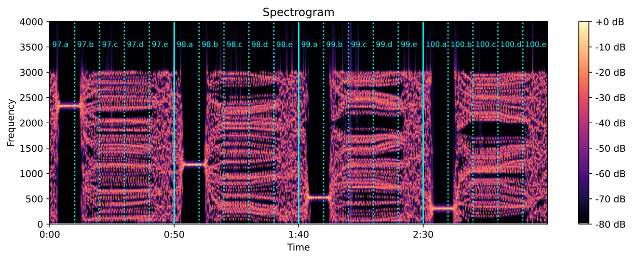


(b)

Figure 9: Sound changes from 11 to 14 agents passing through the five swarmalators' states (from a. to e.). (a) Waveform in the time domain. (b) Spectrum of frequencies over time (spectrogram).

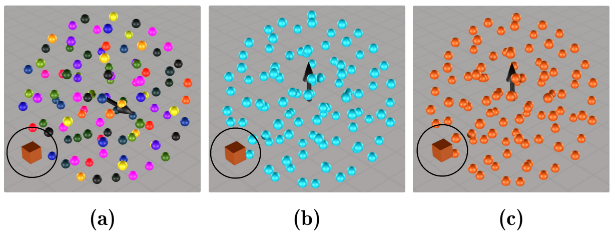


(a)



(b)

Figure 10: Sound changes from 97 to 100 agents passing through the five swarmalators' states (from a. to e.). (a) Waveform in the time domain. (b) Spectrum of frequencies over time (spectrogram).



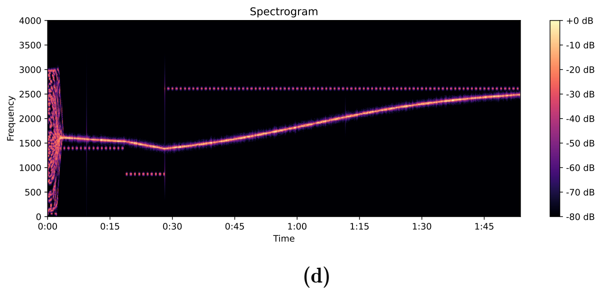


Figure 11: **In (a), (b), and (c), we visually have the influence of an *interactive swarmalator* when changing from the *static async* to the *static sync* state. (d) Spectrum of frequencies over time (spectrogram) for sound changes regarding 100 autonomous swarmalators converging to the *static sync state*. The *interactive swarmalator* has a specific frequency that changes twice, and the rest try to adjust to this one**

5.2.1 *User-changed "Phase"*

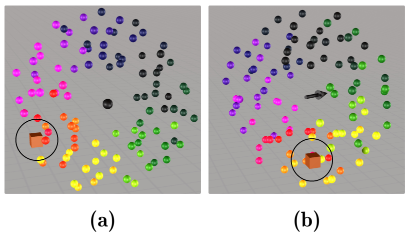
The *interactive swarmalator* described in section 4.1.2 and shown in Figure 3 was added to an environment composed of 100 autonomous swarmalators. We recorded 1 minute and 53 seconds starting from the 100 swarmalators in *static async state* and immediately triggering the *static sync state* and adding the user-controlled agent in a random position with an arbitrary phase. We changed the phase of the *interactive swarmalator* twice (meaning changing the sound frequency), one close to 18 seconds after starting the recording and the other close to 28 seconds, then we let the recording finish at that phase. A video showing the behaviour was also recorded[8].

Figure 11 (a), (b), and (c) show three different states when the *interactive swarmalator* is in place. Figure 11a is the initial *static async state*, then it converges to the *static sync state* in a frequency different to the one set on the *interactive swarmalator* as depicted in Figure 11b; however, the whole swarm approaches eventually to that frequency as in Figure 11c. This behaviour can be confirmed in the audio signal analysis plotted in Figure 11d. The spectrogram shows how the entire swarm agrees on a unique frequency that changes collectively until it matches the *interactive swarmalator's frequency* slowly and in a non-linear fashion.

5.2.2 *User-changed "Position"*

We analyzed the case of having 100 autonomous swarmalators in the *active phase wave state* and one *interactive swarmalator* to observe the behaviour when this agent is moved to a different position. We recorded 3 minutes 57 seconds in this state, starting when the *interactive swarmalator* is added to a random position (see the recorded video[9]). Then, we moved the agent to different positions in the recording. The visual result is shown in Figure 12 (a) and

[8]`https://youtu.be/UzPcXyVL8E8` (accessed May 6, 2024))
[9]`https://youtu.be/epYNfv9GNm4` (accessed May 6, 2024))



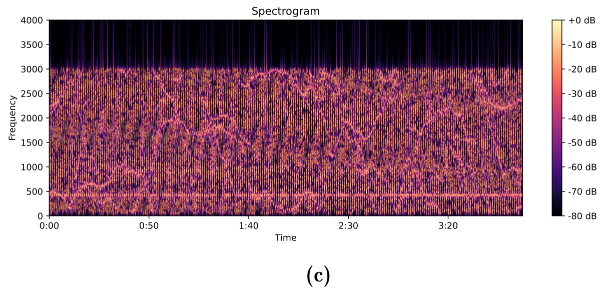


Figure 12: **The influence of an *interactive swarmalator* when it is moved (a) to a different position (b) and the swarm is in *active phase wave state*. Similar phases and positions tend to cluster together. (c) Spectrum of Frequencies over time (spectrogram) for sound changes regarding 100 automatic swarmalators in the *active phase wave state*.**

(b), which tells us that, no matter the position of the *interactive swarmalator*, the swarm always tries to match the phases and position of the closer area so that the overall pattern readjusts to the state of it (e.g. the orange area is closer to the orange *interactive swarmalator* dragging the whole swarm to it). In Figure 12c, we have a sound analysis where it is more difficult to see the changes that are happening in the visualization depicted in Figure 12 (a) and (b); however, we can observe the constant frequency of the *interactive swarmalator* close to 500 Hz, meaning that the "frequency sweeping" effect identified in section 5.1.4 does not affect that frequency section since the agent is not reacting to the other ones.

5.3 Behaviours in a Music Performance

The results presented in this work were applied in a musical setting by utilizing the mapping approach described in section 4.1.3. This mapping is used as a background musical layer, along with an extra mapping configuration as a foreground layer. The foreground layer comprises pitched sounds that have a fast attack and short duration, which vary according to the position of the agents; additionally, this layer only permits music scale frequencies, and its playback options include a pulse on every oscillation cycle or a rhythm pattern covering four cycles.

We created a video[10] showcasing several musical examples using a MIDI controller and a computer mouse. The video is divided into the following four parts, with prompts for actions executed under this configuration.

- **I. Async Rythm:** When the swarm is in *static async* state can generate rhythms with few agents.
- **II. Synced Euclidean Rhythm:** We explore the *static sync* state when every agent is associated with a specific rhythm pattern; in this case, we use a Euclidean rhythm of 10 beats out of 16 steps playing in 4 oscillation cycles. Variations in the number of agents allows different musical patterns to emerge.

[10]`https://youtu.be/rgXaFE6npD8` (accessed May 6, 2024))

- **III. Scalability with a high speed v:** We apply quick changes to the number of agents with a high speed v in the model to demonstrate musical changes when new agents enter to the scene and synchronize rapidly.
- **IV. Interactive swarmalators and more states:** This is the longest section which shows different actions and state changes considering several interactive swarmalators, demonstrating a more complete music performance.

The music possibilities are not limited to these examples, and further mapping strategies can be explored by taking into account the dynamics of the swarmalators.

6. DISCUSSION

6.1 Scalability Effect

The results presented in section 5 support further understanding of the swarmalators' behaviour in the sonic context under the mapping strategy described in section 4.1.3. In terms of scalability, the emergent sound output ("sound swarming") tends to keep the same starting random frequencies when there are three agents in the system on any state except the *static sync* one; however, as we increase the number of agents, the frequencies diverge depending on the different states. For a swarm size of 11 agents, we start to see and hear the effect of motion when the swarmalators are in *active phase wave*, which is more evident for a high density of agents.

Given these results, we can use less than 10 agents for situations that require stable sound frequency values across time, especially when using 3 or 4. As we add more agents, we can take advantage of the temporal variability in these frequencies, which can be more prominent when the swarm is in *active phase wave* and closer to the 100 number of agents.

Several applications can potentially take advantage of such behaviours; for instance, the *static sync state* can be used for synchronization applications in which agents are adjusted in the presence of disturbances; for the *static phase wave*, we can generate rhythms according to how the phase wave travels across the swarm, which could also be the case for the *splintered phase wave* if we group the rhythms according to the emergent clusters spatially created; for the *active phase wave* we can increase the dynamics of the sound through the agents' motion and thus enable a means for granular synthesis. Moreover, the transition between states, which can be perceived more clearly when hearing the sound output, can be useful to denote changes in a sound artwork.

6.2 Swarm-Based and Individual-Based Control

We can expand the interaction space by adjusting J and K to explore additional variations of the five studied states, which, together with v and f, can be parameters controlled by a range of values (i.e. suitable for a "knob-type" control each). These parameters allow a *swarm-based* control, meaning that their changes affect the overall behaviour of the collective; nevertheless, another way of interaction is an *individual-based* control, explored through the *interactive swarmalator* analyzed in section 5.2. Since the *interactive swarmalator* is fully controlled by the user and is not affected by other agents, it can influence the whole swarm as shown previously; that is, when the swarm is in the *static sync state*, the phase to converge will be the one that the *interactive swarmalator* is set, which implies aligning to the same frequency in the sound mapping. Likewise, the swarm tends to adapt to the behaviour of the *interactive swarmalator* in any given state. For instance, we studied how the swarm behaves when it is in *active phase wave*; as a consequence, the swarm arranges itself in such a way that the *interactive swarmalator* remains in the location where its phase and position should be within the swarm.

Having both *swarm-based* and *individual-based* control enriches the human interaction possibilities for this type of IMS, but one should take into consideration that there could be factors that might affect the user experience significantly, such as the limited computational resources and unpredictability in the behaviour when there is a high density of agents.

6.3 The Human-Swarm IMS

In Section 4 we described the design and implementation for a Human-Swarm IMS based on the Swarmalator model that effectively replicates the states studied in the original Swarmalator paper [16], suggesting a way-of-making for such a system in a sound and music context. A working prototype of this IMS was presented in the *Entrainment Workshop*[11] hosted by RITMO in August 2023, raising interest from music technologists and musicologists whose research is focused on synchronization and entrainment.

We continued to develop this prototype to conduct the research presented in this paper and explore the musical possibilities described in Section 5.3. The scalability, changes in state, and inclusion of interactive swarmalators demonstrate a degree of controllability in musical terms. However, we experience momentary deviations that might not feel congruent to the piece on some occasions, such as desynchronized rhythms, dissonant frequencies, or low harmonicity. Nevertheless, we believe these variations can help enrich a performance related to contemporary and experimental music genres.

6.4 Future Work

The model dynamics can be further explored by expanding the analysis presented in this work. For instance, we can focus on the state transitions. Note that we have analyzed a specific order of states (a.b.c.d.e.). Thus, we can change the order to identify significant differences in these transitions. Moreover, we can assess the swarm behaviour in terms of scalability when the *interactive swarmalator* is added since a faster adjustment might be achieved with fewer agents. Additionally, we can analyze rigorously the effect of having more than one *interactive swarmalator* in the environment for a deeper understanding of multiple controllable agents.

In musical terms, analyzing these dynamics would help us to develop new ways of mapping sounds and increase control during improvisational sessions. For example, a performer would be able to determine when a mapped rhythm is synchronized within the group or intentionally desynchronized to create melodic sequences. They could also explore variations in harmony when working with a specific number of agents.

Furthermore, this work is suitable to be explored through immersive technologies. The use of spatial audio to map positions from the model is potentially a way to increase immersion, as well as the use of Extended Reality (XR) devices (e.g. augmented, mixed, and virtual reality equip-

[11] `https://osf.io/stx3e` (accessed May 6, 2024))

ment). Robotic systems provide another means of visualization, merged with a physical environment, and can allow multimodal interaction. The problems to solve in these aspects are related to platform-specific challenges and HCI research to find optimal experiences.

We plan to continue these directions, explore different mapping strategies, and include users to discover emergent affordances over novel Human-Swarm IMSs.

7. CONCLUSIONS

This work presents the design and implementation of a Human-Swarm IMS based on the 3D Swarmalator model whose "sound swarming" output is analysed in terms of *scalability*. We identified specific sound behaviours visible in the frequency domain throughout several swarm states depending on the number of agents.

We added an *interactive swarmalator* to the model and studied its effect on the other autonomous agents. We observed that a swarm of 100 agents, slowly and non-linearly, adjusts to the behaviour of this single individual in terms of phase and position.

Moreover, we use the results of the presented analysis in a music performance that includes examples showing the performative possibilities of the system.

These findings address the research questions stated in the introduction and increase our understanding of the Swarmalator model for music systems. The results we present can inform design decisions that affect the sonic and human interaction experience when combining *swarm-based* and *individual-based* control.

8. ETHICAL STANDARDS

This work was partially supported by the Research Council of Norway through its Centres of Excellence scheme, project number 262762. It does not involve human participants.

9. REFERENCES

[1] D. Bisig and S. Schiesser. Coral – a Physical and Haptic Extension of a Swarm Simulation. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 385–388, Daejeon, Republic of Korea, May 2013. ISSN: 2220-4806.

[2] T. Blackwell. Swarming and Music. In E. R. Miranda and J. A. Biles, editors, *Evolutionary Computer Music*, pages 194–217. Springer London, London, 2007.

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems.* Oxford University Press, Oct. 1999.

[4] P. Bottoni, S. Faralli, A. Labella, and M. Pierro. Mapping with Planning Agents in the Max/MSP Environment: the GO/Max Language. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 322–325, Paris, France, 2006. ISSN: 2220-4806.

[5] M. Burtner. Perturbation Techniques for Multi-Performer or Multi- Agent Interactive Musical Interfaces. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 129–133, Paris, France, 2006. ISSN: 2220-4806.

[6] S. Chakraborty and J. Timoney. Multimodal Synchronization in Musical Ensembles: Investigating Audio and Visual Cues. In *International Conference on Multimodal Interaction*, pages 76–80, Paris France, Oct. 2023. ACM.

[7] A. Eigenfeldt and A. Kapur. 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, 2008. ISSN: 2220-4806.

[8] V. E. Gonzalez Sanchez, C. P. Martin, A. Zelechowska, K. A. V. Bjerkestrand, V. Johnson, and A. R. Jensenius. Bela-Based Augmented Acoustic Guitars for Sonic Microinteraction. In T. M. Luke Dahl, Douglas Bowman, editor, *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 324–327, Blacksburg, Virginia, USA, June 2018. Virginia Tech. ISSN: 2220-4806.

[9] Y. Kuramoto. Self-entrainment of a population of coupled non-linear oscillators. In H. Araki, editor, *International Symposium on Mathematical Problems in Theoretical Physics*, pages 420–422, Berlin, Heidelberg, 1975. Springer Berlin Heidelberg.

[10] A. Lambert. A Stigmergic Model for oscillator synchronisation and its Application in Music Systems. In *Proceedings of the 38th International Computer Music Conference.* Michigan Publishing, 2012.

[11] N. Lem. Sound in Multiples: Synchrony and Interaction Design using Coupled-Oscillator Networks. *16th Sound and Music Computing Conference*, May 2019. ISBN: 9788409085187 Publisher: Zenodo.

[12] N. Lem and Y. Orlarey. Kuroscillator: A Max-MSP Object for Sound Synthesis using Coupled-Oscillator Networks. In *14th International Symposium on Computer Music Multidisciplinary Research*, page 924, 2019.

[13] P. Lucas and K. Glette. Human-Swarm Interactive Music Systems: Design, Algorithms, Technologies, and Evaluation. *Proceedings of the 16th International Symposium on Computer Music Multidisciplinary Research*, Nov. 2023. Publisher: Zenodo.

[14] P. P. Lucas and S. Fasciani. A Human-Agents Music Performance System in an Extended Reality Environment. In M. Ortiz and A. Marquez-Borbon, editors, *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 10–20, May 2023. Place: Mexico City, Mexico.

[15] K. Nymoen, A. Chandra, K. Glette, and J. Torresen. Decentralized harmonic synchronization in mobile music systems. *2014 IEEE 6th International Conference on Awareness Science and Technology, iCAST 2014*, 257906(257906), 2014. Publisher: IEEE ISBN: 9781479973736.

[16] K. P. O'Keeffe, H. Hong, and S. H. Strogatz. Oscillators that sync and swarm. *Nature Communications*, 8(1):1–12, 2017. arXiv: 1701.05670 Publisher: Springer US.

[17] J. G. Sibley-Schwartz. Modeling coupled oscillators: Applications for musical sequencing and synthesis. *The Journal of the Acoustical Society of America*, 153(3_supplement):A231–A231, Mar. 2023.

[18] N. J. W. Thelle and P. Pasquier. Spire Muse: A Virtual Musical Partner for Creative Brainstorming. In *NIME 2021*. PubPub, 2021.

[19] Y. Uozumi, M. Takahashi, and R. Kobayashi. Bd : A Sound Installation with Swarming Robots. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 426–426, New York City, NY, United States, 2007. ISSN: 2220-4806.

[20] M. Wooldridge. *An Introduction to MultiAgent Systems.* Wiley Publishing, 2nd edition, 2009.