# Swarm Based Computer Music - Towards a Repertory of Strategies

**Daniel Bisig**
*Institute for Computer Music and Sound Technology*
*Zurich University of the Arts*
*e-mail: daniel.bisig@zhdk.ch*

**Martin Neukom**
*Institute for Computer Music and Sound Technology*
*Zurich University of the Arts*
*e-mail: martin.neukom@zhdk.ch*

## Abstract

Algorithms, simulations and data collections from the sciences constitute an important source of inspiration and manipulation for generative art. One of the main challenges in adapting such material for artistic purposes consists in the creation of aesthetically meaningful correspondences between the underlying data and the appearance and behavior of the artwork. This paper focuses on the application of swarm simulations for the creation and control of computer music. The authors propose different categories of relationships that can be established between a swarm's behavior and musical processes. These categories include: parameter mapping, proximity based events, procedural patching, physical representation. We hope that the identification and evaluation of these different categories will contribute to the establishment of a repertoire of strategies that help musicians to assess and harness the aesthetic potential of swarm based music.

## 1. Introduction

Generative and Algorithmic Art emphasize the employment of processes for the automated creation of artworks [1]. Very often, these processes result from algorithms, simulations and data collections that have originally been created with non-artistic goals in mind. Accordingly, artistic realization involves the challenging task of establishing meaningful correspondences between these underlying principles and the appearance and behavior of the artwork. These correspondences should take into account the perceptual peculiarities of the selected feedback modalities and need to balance innate algorithmic properties with stylistic constraints and embellishments in the creation of an aesthetic result. It is rare that a direct relationship between an algorithm and the artwork's behavior and appearance leads to satisfactory results. For example an algorithm's dynamics may be too varied and complex to perceive recurring patterns. The range and density of its output may not be adequate for the selected output modality. The algorithm's long term behavior may fail to create a dramaturgy and aesthetic tension that maintains a visitor's interest. This paper discusses several categories of relationships between algorithmic processes and aesthetic feedback. These categories have been identified based on a set of experiments that employ swarm simulations as prototypical examples of

autonomous, self-organized and emergent processes and that explore methods of transforming their behaviors into acoustic feedback.

## 2. Swarm Music

Swarm simulations possess a variety of properties that render them particularly promising for musical and artistic applications. Swarms can exist within spaces of arbitrary dimension and topology that can be matched to the artworks feedback mechanism. Agents within a swarm tend to move in clusters. This fact can be exploited to create correlations and coherence between parameters. Agent behaviors can be easily customized to meet desired levels of complexity and autonomy for an artwork. Finally, swarm simulations support natural and intuitive forms of interaction. The application of swarm simulations for the creation of music has been explored by several artists such as [2-9].

The research that led to this publication employs software tools that have been developed as part of a project entitled Interactive Swarm Orchestra (ISO) [10]. The activities within this project are twofold. They include research that addresses theoretical and practical aspects of swarm based computer music [11]. In addition, they deal with the development and distribution of generic software and hardware tools and their documentation in order to facilitate the realization of swarm based computer music [12-13].

## 3. Relationship Categories

This publication describes a series of experiments that served to identify and evaluate different categories of relationships between swarm simulations and musical creation. These experiments have been realized by creating software applications that combine several ISO related software libraries such as ISO Flock (a library for swarm simulation) and ISO Synth (a library for sound synthesis, signal processing and sound spatialization). The applications have been tested on a mobile installation (ISOkaeder) that consists of 20 speakers positioned within the corners of a Dodecahedron structure. Within the interior of this structure, the audience perceives spatially distributed sounds via three-dimensional ambisonic sound projection. All examples employ swarm simulations both for the creation and positioning of sounds. The individual examples differ with regard to the method of sound creation, and the relationship between swarm behavior and acoustic result. All examples are identical with respect to the mapping of agent positions to sound positions. The position of sound sources is directly derived by normalizing the position of individual agents. The only exception to this rule applies when sound sources are placed very close to the origin of the coordinate system that defines the installation's speaker positions. In this case, the position of the sound sources is offset to the surface of a sphere whose center lies at the position of the origin and whose radius corresponds to a predefined proximity threshold between the sound sources and the origin (see figure 1).

The examples that are described throughout the following sections have been categorized based on the relationship between swarm behavior and acoustic result. These categories include: parameter mapping, proximity based events, procedural patching, physical representation.

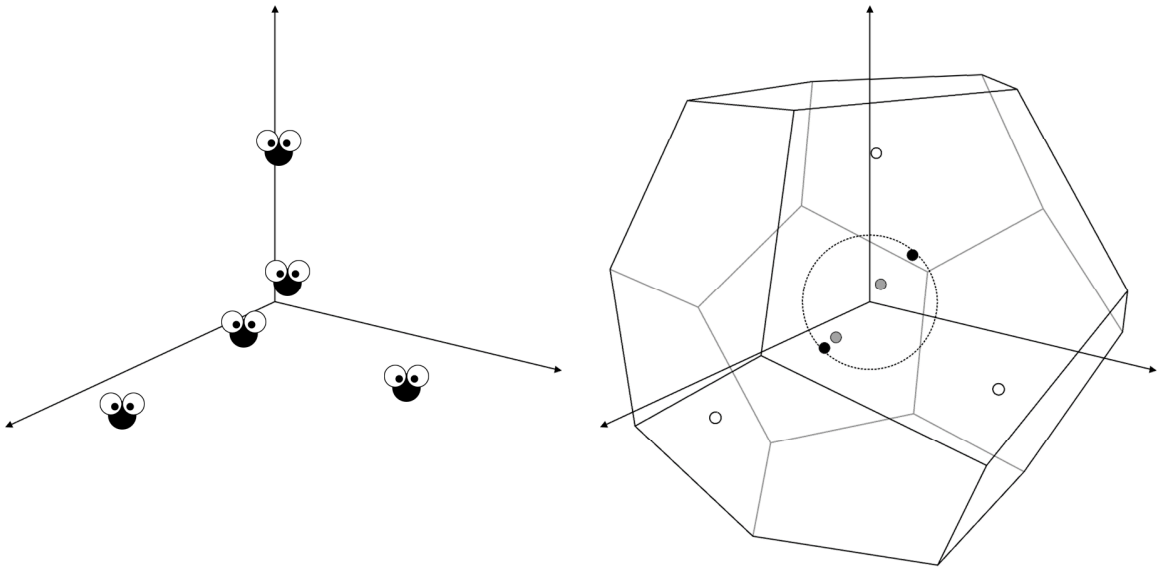

***Figure 1:*** *Swarm Based Sound Spatialization. Left side: swarm of agents. Right side: sound sources placed within a dodecahedron structure. Filled gray circles represent sound sources whose proximity to the origin is below a predefined threshold (depicted as central circle). Filled black circles represent sound sources that have been moved onto the surface of the threshold circle. Empty circles represent sound sources whose position has not been altered.*

### 3.1 Parameter Mapping

The simplest and most commonly applied correspondence between swarm behavior and musical result employs a fairly direct mapping from agent properties to musical parameters. To match the number of musical parameters that should be controlled by a swarm, the dimensionality of individual agent properties is adapted or several agent properties are combined. Subsequently, the values of these agent properties are scaled and possibly discretized to limit the range and diversity of values that the musical control parameters can assume. Regardless of the specifics of the mapping, this type of correspondence renders the characteristics of swarm behavior clearly perceptible in the musical result. The music is usually made up from a collection of acoustic elements whose number matches the number of agents and whose properties cluster more or less strongly depending on the strength of attractive and repulsive behaviors among agents. In addition, the musical dynamics exhibits correlated acoustic movements whose velocity and diversity matches those of the underlying swarm.

#### 3.1.1 Additive Synthesis

In this example, several sine oscillators, whose number equals the number of swarm agents, sum their output to create the final acoustic feedback (see figure 2). Each agent controls its respective sine oscillator by mapping the y component of its position to the oscillator's frequency and the z component to the oscillator's amplitude. Depending on the spatial distribution of the agents, the acoustic contributions of the individual oscillators are clearly discernible or their output fuses into a single tone. The oscillators' frequencies and amplitudes sweep according to

the agents movements. The velocity of these sweeps depends both on the movement velocity of the agents and the update interval of the swarm simulation. This update interval is much slower than audio rate and leads to temporal discretization. This discretization becomes audible at a simulation update rate lower than about a 1/20th of a second.

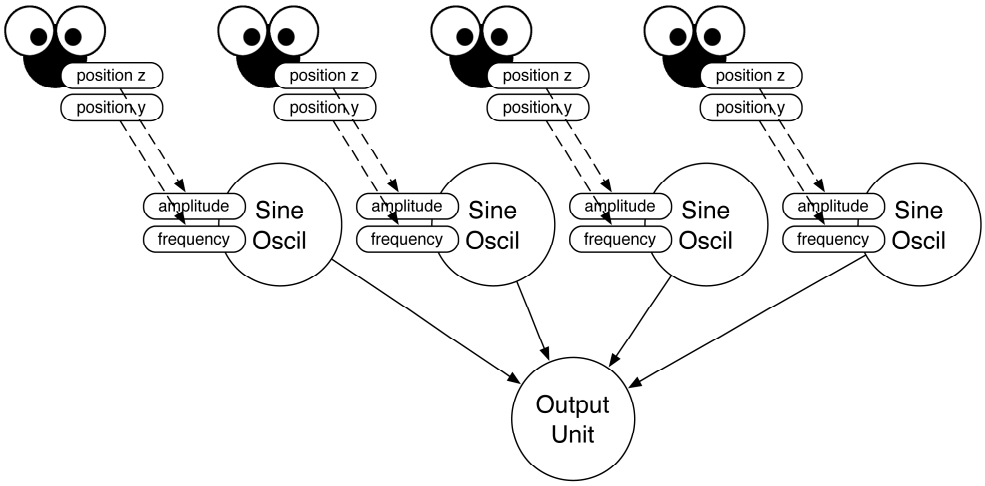


***Figure 2:*** *Swarm Controlled Additive Synthesis. Labelled circles represent audio units. Rounded text boxes represent agent properties and audio control ports, respectively. Solid arrows correspond to signal paths between audio units. Dashed arrows represent mappings from agent properties to audio unit control ports.*

### 3.1.2 Granular Synthesis

This example also employs parameter mapping but assumes a slightly more sophisticated approach. First of all, it maps those agent properties that reflect the behavior of the entire swarm particularly sensitive well. In addition, musical parameters at updated more frequently than the simulation rate. This effect is achieved my mapping the properties of several agents at slightly different times to a single musical parameter. More technically, the agent parameters are written into control buffers at positions that are specific for each agent (see figure 3). The synthesis engine reads these buffers and applies their content at audio rate to control the corresponding musical parameter. The write position within these control buffers is determined by discretizing the x-component on an agent's position. The resulting index represents the temporal position when the agent's properties take musical effect. The more agents a swarm contains and the further they spread apart, the more frequent the acoustic feedback changes.

This example employs granular synthesis to generate sound. Grains are created by extracting short stretches of audio samples from a ring buffer into which the content of an audio files is continuously fed. When the grains are played back, the samples they contain are multiplied by a bell shaped amplitude envelope. The output of all grains that play simultaneously is summed. Each agent triggers a single grain and controls the characteristics of this grain during one update interval of the sound synthesis engine. The x component of an agent's position controls the trigger time of

the sound grain. The amplitude of an agent's velocity determines the playback rate of the grain's content. The direction of the agent's velocity determines the position within the ring buffer from which the grain's content is created. The average distance between an agent and its neighbors determines the duration of the grain. The shorter this distance, the longer the duration of the grain. Finally, the more neighbors an agent possesses, the larger is the amplitude of the corresponding grain.

This parameter mapping creates a musical result whose perceived similarity to the original audio material is strongly dependent on the properties of the swarm. Coherent agent movement is represented by little variation in the agents' relative positions and velocities. Accordingly, grains are chosen and played in the same sequence as the audio was stored in the audio file. In addition, compact swarms result in the formation of clearly audible and continuous audio output whereas agents that are scattered far apart lead to the creation of faint and short audio fragments that fail to merge into a continuous audio stream.

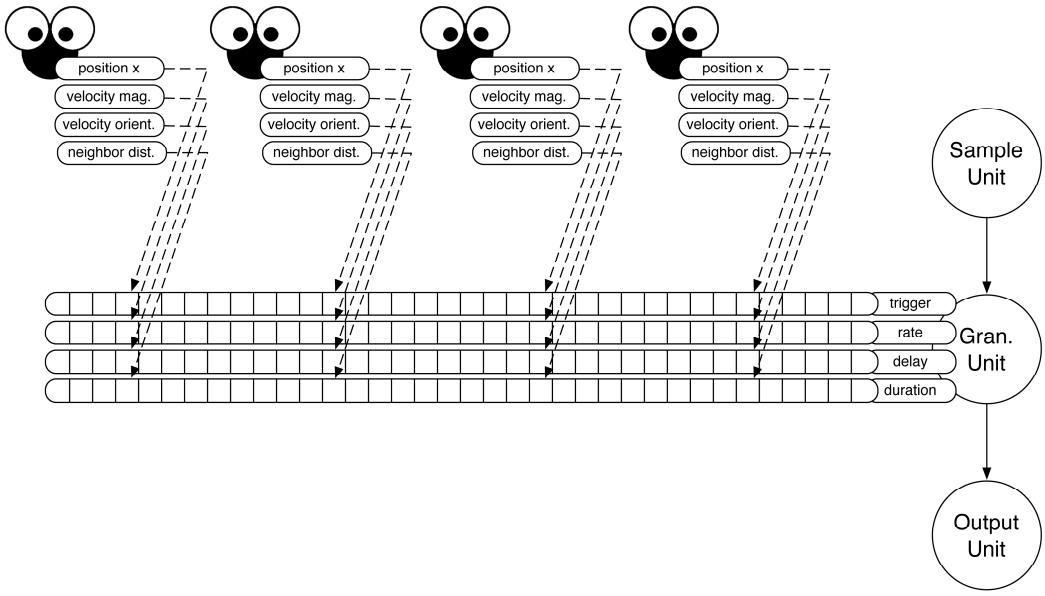


***Figure 3:*** *Swarm Controlled Granular Synthesis. The one dimensional grid patterns represent audio unit control port buffers. Dashed arrows represent mappings from agent properties to particular positions within these buffers.*

### 3.2 Proximity Based Events

The distinct spatial properties of swarms can be exploited by establishing proximity relationships between agent properties and musical control parameters. Both the agents' properties and musical control parameters are projected as points into a shared space. The euclidian distance between these points may then be employed to change musical properties of trigger musical events. This approach is attractive because it takes into account the inherently spatial nature of swarms. The properties of the shared spaces can be adapted to deal with different musical topologies. Since the number of agent properties does not need to match the number of control parameters, this approach offers good flexibility when the number of agents or musical control parameters changes throughout the course of a musical piece. On

the other hand, this approach may introduce some computational overhead due to the possibly many distance calculations that need to be performed. This problem can be addressed by employing appropriate space partitioning techniques. The ISO Space programming library offers a small variety of such techniques such as fixed spatial grids or hierarchical region trees that are applicable for any dimensionality.

### 3.2.1 Sample Triggering

This example represents a very simple application of the previously described technique. The example employs an audio patch that consists of several sample playback units and whose output is combined to create audio feedback (see figure 4). Each of these units has a three-dimensional position associated to it which is mapped into a three-dimensional euclidian space. The agents' position parameters are also mapped into this space and updated continuously as the simulation progresses. Whenever the sound synthesis engine updates, it evaluates the distance between the positions of the agents and the sample playback units. If this distance falls below a certain threshold, the corresponding audio sample is triggered. A sample keeps playing as long as the distance remains below the specified threshold. The amplitude of sample playback is inversely proportional to this distance and causes samples to fade in and out when agents move close by. Despite its simplicity, this approach illustrates some of the possibilities of a spatial approach to swarm music. For example, the probability that similar audio files play concurrently can be increased by assigning close-by positions to the sample playback units. These positions may also be placed at regular intervals to facilitate the creation of rhythmic patterns. Both the audio files characteristics and the units' distribution patterns may vary across the entire space. An entire composition may then unfold as the agents move slowly through this space.

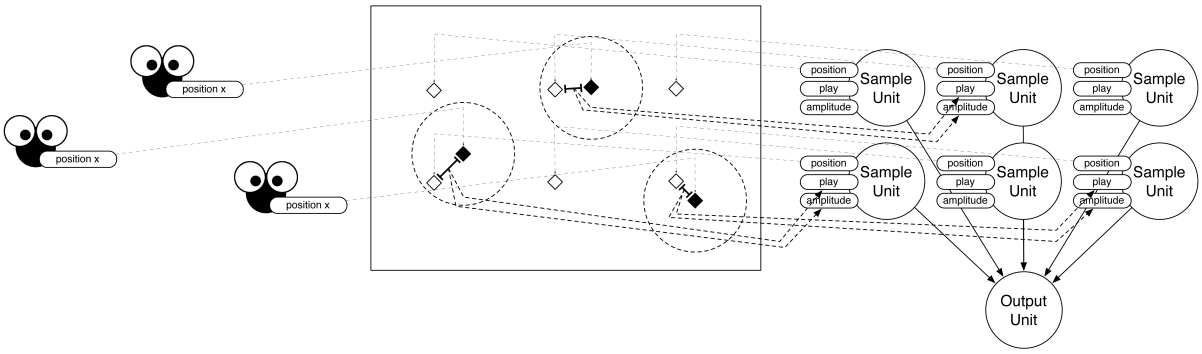


***Figure 4:*** *Proximity Based Sample Triggering. Both agents (depicted on the left) and sample playback units (depicted on the right) possess positions within a shared Euclidian space (filled black diamond shapes designate agent positions, empty diamond shapes designate audio unit positions). Dashed circles represent proximity thresholds among agent and audio unit positions. Black bars stand for neighborhood relationships among these positions.*

### 3.3 Procedural Patching

In all the categories that have been presented so far, the role of a swarm was limited in that it could only modulate pre-specified control parameters of a pre-existing musical patch. Procedural patching concedes an entirely different and much less constrained role to a swarm. Here, a patch is constructed on the fly according to rules whose execution depends on the behavior and properties of the swarm. As the swarm simulation progresses, the musical patch will continue to evolve as the rules are applied over and over again. This approach offers by far the greatest flexibility but also poses mayor challenges for finding satisfying technical and aesthetic solutions. The technical challenges include the following aspects: the sound synthesis engine needs to support life patching functionality (at the time of this writing, this functionality is only partially implemented in ISO Synth), the resulting patches produce output signals whose dynamic range is within the allowed boundaries for audio data (i.e. in between -1.0 and 1.0), the computational demands of the resulting patches stays below the processing power of the computer. The aesthetic challenges stem from the fact that the introduction of patch construction rules adds an additional levels of design decisions and complexity to swarm based computer music. The swarm in itself represents a generative system that is difficult to exploit musically. A rule based patch construction system represents a second generative layer. Accordingly, it is very hard to make aesthetically informed design decisions for such a two layer system. In addition, it is difficult to preserve the acoustic recognizability of a swarm's behavior across the level of patch construction rules. Despite these challenges, procedural patching is likely to become one of the aesthetically most rewarding forms of swarm based computer music.

### 3.3.1 Modulation Synthesis

In this example, a swarm simulation controls the topology of a musical patch that consists solely of sine oscillators (see figure 5). The patch exists in two versions, a silent version that is in the process of being created by the swarm and an active version that produces audio output. Whenever, the silent patch is finished, the two patches switch their roles and all the connections between oscillators in the formerly active patch are severed. Every agent possesses two oscillators (one in each patch version) that remain assigned to it as long as the agent exists. The connections among the oscillators are established based on the neighborhood relationships among agents. The first agent that is selected from a group of neighboring agents as well as all agents that possess no neighbors connect the output of their oscillators to the final audio output of the sound synthesis system. The frequency and amplitude of this oscillator is directly controlled by the agent's position. Agents that appear as neighbors of these first agents connect their oscillators to one of the control ports of these output oscillators. The modulation amount and modulation frequency are controlled by the neighbor's distance. In the sequence of increasing distance, the first neighbor attaches its oscillator to the amplitude control port, the second one to the frequency control port and the third one to the phase control port. Additional neighbors are treated as first agents and connect their oscillators to the output unit. Agents that are neighbors of neighboring agents connect their oscillators to the control ports of oscillators that are already connected to other control ports. By this way, a modulation tree emerges that represents the neighborhood relationship among the agents. If agents are scattered far apart, all the agent oscillators produce a directly audible output and an audio feedback that possesses a constant spectrum

results. At the other extreme, if all agents form part of the same closely packed swarm, only a single oscillator produces an audio signal but this signal is heavily modulated by a very large modulation three. Therefore, as a swarm's density varies, the method of sound synthesis will gradually shift in between additive and modulation synthesis.

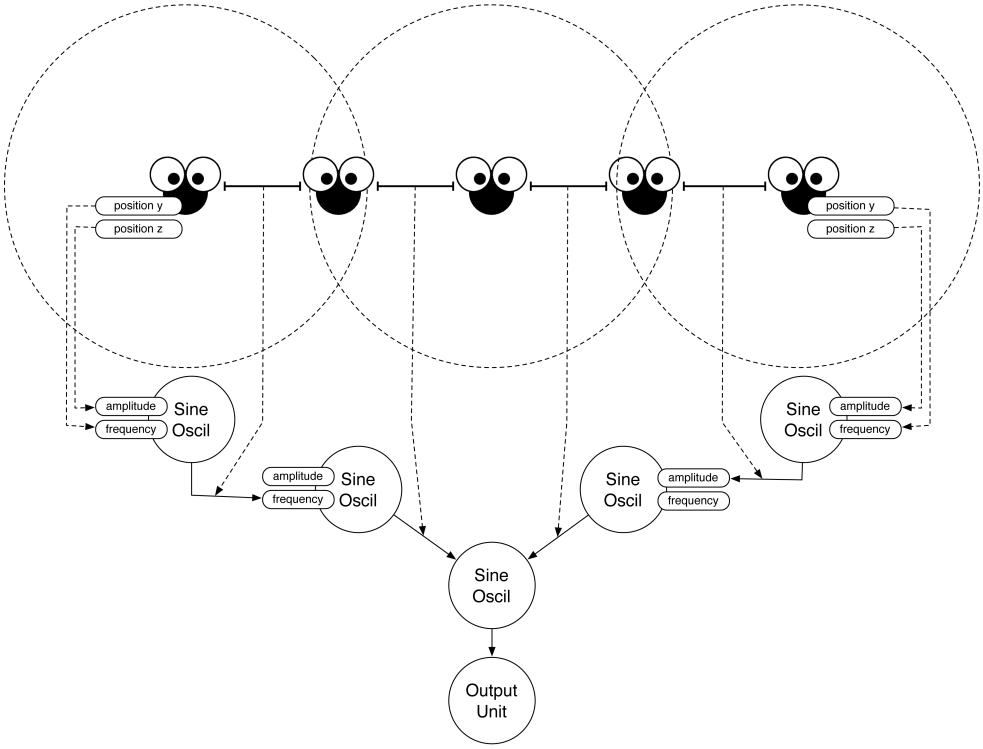


***Figure 5:*** *Agent Neighborhood Based Audio Patch Creation.*

### 3.4 Physical Representation

Most swarm simulations bear similarity with particle based physical simulations. Agents act as point masses that are subject to proximity dependent attraction and repulsion forces. Physical modeling plays an important role in sound synthesis. It therefore seems attractive to merge swarm simulations with those physical modeling techniques that simulate forces acting among particles in order to reproduce natural acoustic phenomena. This approach is promising since it combines the benefits of realistic sound models with a swarm's capabilities for self-organization and emergence. Accordingly, the emphasis of this approach lies on the creation of an acoustic output that is characterized by naturalness and familiarity in its sound texture and temporal dynamics. The design of this type of swarm based computer music is less prone to problems of arbitrariness and contrivance than those described previously. Here, design decisions involve the selection of swarm models and physical models that share similar characteristics as well as the weighting of their respective behaviors.

#### 3.4.1 Mass-Spring Networks

In this example, swarm agents assume the role of mass points in a mass-spring network (see figure 6). Springs are created and destroyed based on the neighborhood relationships among the agents. The oscillations of these springs generates the acoustic output. The damped springs are continuously excited via white noise. Springs are excited most strongly when they are created or severed. The swarm controls the amount of coupling between the springs. Uncoupled springs oscillate at unrelated frequencies. The resulting sine waves are perceived as individual sound sources. Coupled springs influence each other's oscillations. The larger the number of coupled springs and the more linear the arrangement of their connections the closer the mass-spring network approximates the behavior of a string. In this arrangement, the springs produce sine waves whose output merges and is perceived as a single sound that possess a harmonic spectrum.

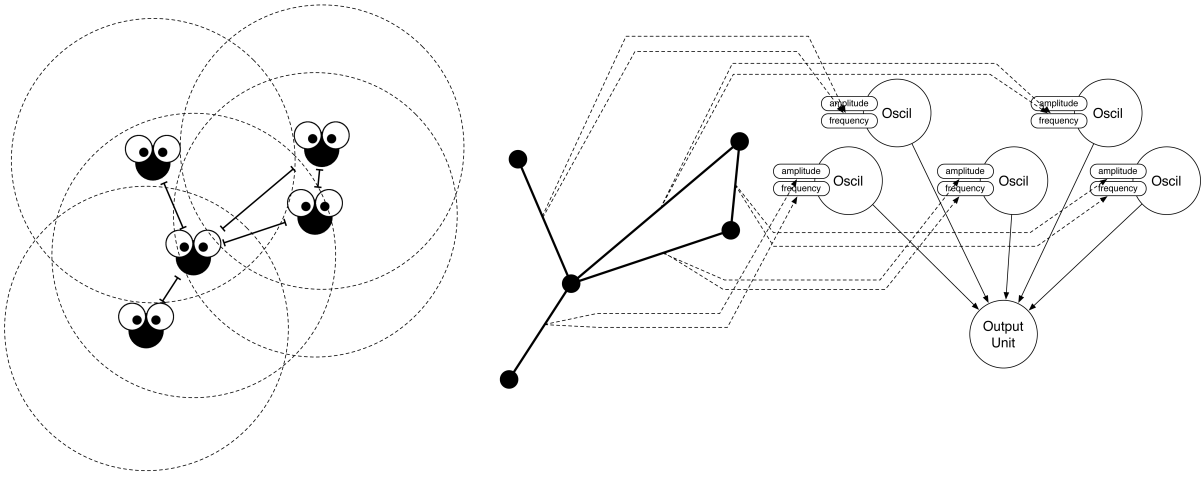


***Figure 6:*** *Agent Neighborhood Based Mass Spring Network. Filled black circles represent mass points, strong black lines represent damped springs.*

## 4. Results and Conclusions

The evaluation and establishment of design strategies for the creation of swarm based computer music forms an essential part of our research within the context of the ISO project. The identification of several categories of swarm music relationships represents an important step towards our goal to help musicians to assess and harness the aesthetic potential of swarm based music. We hope that this publication convinces the reader that swarm based computer music still possesses largely untapped potential and that its greatest promise is likely to lie beyond parameter mapping based approaches. We believe, that some of these results are of relevance outside the realm of swarm based computer music and therefore may inspire other research within the field of generative art.

Obviously, additional research and artistic experimentation is required to further advance swarm based computer music. So far, in all our acoustic experiments, sound spatialization has been restricted to a direct mapping from agent position to sound position. It is clear that future experiments need to explore more diversified approaches to swarm based audio spatialization and study their effects on immersion and spatial perception. Furthermore, in all our experiments, swarms have existed in

spaces of Euclidian topology. In our future research, we would like to substantiate our statement that a swarm simulation's spatial topology can be suitably adapted to match an artwork's specific feedback.

The relationship between swarm behavior and performer activities constitutes another important aspect of swarm based computer music. In a previous publication [Bisig et al. 2008a], we have outlined some aspects of human swarm interaction in a somewhat speculative manner. We are well aware that this issue deserves a more systematic and empiric approach. Accordingly, the next research phase within the ISO project focuses on conceptual, technical and artistic aspects of human swarm interaction.

Finally, we believe it is important to concede a more significant role to artistic creation within our future research. Both members of the ISO research team as well as independent composers and musicians should de-emphasize short technical and aesthetical experiments in favor of dedicated musical compositions and public performances. This approach constitutes the only viable method to assess the capability of swarm based computer music to generate musically fertile results that capture an audience's interest over sustained periods of time.

## 5. References

[1] Galanter, P. (2008). What is Generative Art - Complexity theory as a context for art theory", Proceedings of the Generative Art Conference, Milano, Italy.

[2] Blackwell, T. (2003). Swarm music: improvised music with multi-swarms. Artificial Intelligence and the Simulation of Behaviour, University of Wales.

[3] Blackwell, T. and Young, M. (2004). Swarm Granulator. EvoWorkshops, Coimbra, Portugal.

[4] Uozumi, Y. (2007). GISMO2: An Application for Agent-Based Composition. Lecture Notes in Computer Science, Springer Berlin/Heidelberg.

[5] Uozumi, Y., Takahashi, M., and Kobayashi, R. (2007). A Musical Framework with Swarming Robots. Proceedings of the International Symposium on Computer Music Modelinh and Retrieval, Copenhagen, Denmark.

[6] Unemi, T. and Bisig, D. (2004). Playing Music by Conducting BOID Agents. Proceedings of the Ninth International Conference on Artificial Life IX, Boston, USA.

[7] Unemi, T. and Bisig, D. (2005). Music by Interaction among Two Flocking Species and Human. Proceedings of the Third International Conference on Generative Systems in Electronic Arts, Melbourne, Australia.

[8] Unemi T. and Bisig D. (2007). Identity SA - an interactive swarm-based animation with a deformed reflection. Proceedings of the Generative Art Conference. Milano, Italy.

[9] Bisig, D. and Unemi, T. (2006). MediaFlies - A Video and Audio Remixing Multi Agent System. Proceedings of the Generative Art Conference, Milano, Italy.

[10] Bisig D., Neukom M., and Flury J. (2007) Interactive Swarm Orchestra. Proceedings of the Generative Art Conference. Milano, Italy.

[11] Bisig, D., Neukom, M. and Flury, J. (2008a). Interactive Swarm Orchestra, an Artificial Life Approach to Computer Music", Proceedings of the International Computer Music Conference, Belfast, Ireland.

[12] Bisig D., Neukom M., and Flury J. (2008b) Interactive Swarm Orchestra - A Generic Programming Environment for Swarm Based Computer Music. Proceedings of the International Computer Music Conference. Belfast, Ireland.

[13] Interactive Swarm Orchestra Project Website: http://www.i-s-o.ch/