MediaFlies
An Interactive Flocking Based Tool for the Remixing of Media

Daniel Bisig and Tatsuo Unemi

Artificial Intelligence Laboratory, University of Zurich, Switzerland
Daniel Bisig, University of Zurich, Andreasstr. 15, 8050 Zurich, Switzerland
Email: dbisig@ifi.unizh.ch

Department of Information Systems Science, Soka University, Tokyo, Japan
Tatsuo Unemi, Soka University, 1-236 Tangi-cho, Hachioji-city, Tokyo, 192-8577, Japan
Email: e-mail: unemi@iss.soka.ac.jp

Abstract

The project MediaFlies implements an interactive multi-agent system that incorporates flocking and synchronization in order to generate a constantly changing visual output. It relies on prerecorded or life video material, which is fragmented and recombined via the agents' activities. Users influence these visuals via video based tracking and thereby affect the amount of disturbance of the original material. The project draws its inspiration from the biological phenomena of flocking and synchronization. Simulations of these phenomena form the basis for the generative behavior of MediaFlies. In its current implementation, MediaFlies does not support audio. Future improvements will address this shortcoming and will also implement feedback mechanisms by which the agents' behaviors are influenced by qualitative properties of the media material itself.

Introduction

Research in the field of Artificial Life and Artificial Intelligence forms an important conceptual and technical background for people who produce algorithmic and generative art. In these fields, art and science share a common interest in issues such as emergence, self-organization, complexity, autonomy, adaptivity and diversity. Complex and self-organized systems have a great appeal for art, since they possess the capability to continuously change, adapt and evolve (Sommerer and Mignonneau 2000). In addition, such systems tend to respond to user input in surprising and non-trivial ways and can therefore be very rewarding for users who engage in exploratory forms of interaction.

A group of animals such as a school of fish or a flock of birds constitutes a typical example of a self-organized system. Flocking algorithms, which model these behaviors, are based on the seminal work of Craig Reynolds (Reynolds 1987). Such algorithms enjoy an increasing degree of popularity among artists. Flocking algorithms give raise to lifelike and dynamic behaviors that can easily be adapted to control a variety of visual or acoustic parameters. In addition, flocking algorithms lend themselves to interaction. This doesn't come as a surprise when taking into account the joyfulness, which for instance young children express when scaring away flocks of real birds. For this reason, interaction with a flocking based artwork often seems very intuitive and natural. Examples of flocking based generative systems include robotic works such as "ARTsBot" (Ramos et al. 2003), interactive musicians (Blackwell 2003) or orchestras (Unemi and Bisig. 2005), and interactive video installations such as "SwarmArt" (Boyd et al. 2004) and "swarm" (Shiffman 2004). The project MediaFlies is similar to the works of Blackwell and Shiffman, in that it employs a sample driven approach to generative art. The system doesn't create it's output entirely de novo but rather by combining human generated visual input with the generative activity of the flock. We are convinced that this approach yields a very interesting potential for generative art, in that it tries to combine human and machine creativity and thereby escapes some of the pitfalls of aesthetical inaccessibility that occasionally haunts generative art.

Concepts

As stated in the introduction, algorithmic and generative art has been influenced by research in the field of Artificial Life and Artificial Intelligence since quite some time. One of the possible roles an artist can assume in this relationship is to mix and match formerly fairly unrelated scientific methods in order to explore novels forms of interactivity and conduct aesthetic experiments. In this spirit, the MediaFlies project attempts to combine flocking and synchronization simulations.

It also tries to explore novel ways in which algorithmic variety and human based aesthetic decisions could be combined. One of the classical approaches to this issue is interactive genetic algorithms, in which the user takes over the part of a fitness function in an evolutionary system. MediaFlies abandons this somewhat tedious and slow approach in favor of a system that is dynamic and open to

more flexible forms of interaction. MediaFlies realizes an automated and interactive copy & paste type of system, which puts the computer into the role of a video jockey. MediaFlies recombines fragments of existing video material into a new artwork, which may or may not be totally dissimilar from its origins.

In summary, the MediaFlies project tries to realize the following two conceptual ideas:

- To combine flocking and synchronization algorithms in order to mimic the behavior of a large aggregate of individuals whose changing spatial relationship lead to the formation and breakup of clusters of concerted activity.
- To create a generative system, which produces a remix of existing media material that, depending on user interaction, constantly shifts in between a faithful reproduction and a total defamiliarization of this material.

Implementation

The software has been developed in C++ under Mac OS X 10.3 & 10.4. It relies on a variety of libraries (see table 1), which are available for different operating systems. Porting MediaFlies to different platforms should therefore be relatively easy.

Library	URL	Comment
TrollTech Qt	www.trolltech.com	GUI Development
Quicktime	developer.apple.com/ quicktime/	Video Capture / Playback
OpenGL	www.opengl.org	3D Graphics
OpenCV	www.intel.com/technology/ computing/opencv	Computer Vision

Table 1: Software Libraries. A list of all the libraries on which MediaFlies relies.

The implementation of MediaFlies involves the following aspects, which will be discussed in the next subsections: agent system, flocking simulation, synchronization, visual feedback, and interaction.

Agent system

In MediaFlies, agents populate a 3D world that is continuous in time and space and which exhibits periodic boundary conditions. The agents are organized within agent groups. These groups comprise all agents that share common properties. The agent groups update the agent neighborhood relationships by employing a "Loose Octree" space partitioning scheme (Thatcher, 1999). The behavior repertoire of an agent is also managed by its encompassing agent group. This repertoire consists of a list of basic behaviors, each of which conducts a single activity (such as evasion or velocity alignment) and whose output is a force vector that is added to the agent's overall force vector. At the end of a simulation step, the agent's theoretical acceleration is calculated from this summed force. By comparing the theoretical acceleration with the agent's current velocity, linear and angular acceleration components are derived and subsequently clamped to maximum values. Finally, the agent's velocity and position are updated by employing a simple explicit Euler integration scheme.

For performance reasons, two agent groups are implemented in the current system. The relatively few agents (100 – 400, depending on computer power), which are part of the so-called master agent group conduct true flocking. The second group is named slave agent group and contains a much larger number of agents (500 – 4000). These slave agents possess a much more limited behavioral repertoire and rely on the master agents to mimic flocking. By this method, a larger number of agents can be simulated than if all agents would conduct proper flocking. The master agents are usually not visible to the user. It is the slave agents, which control the simulation's visual feedback and engage in synchronization behavior.

Flocking simulation

The flocking algorithm is closely related to the original Boids algorithm (Reynolds 1987). It consists of the following three basic behaviors:

- Cohesion: agents move towards the perceived center of their neighbors
- Evasion: agents move away from very close neighbors
- Alignment: agents adapt magnitude and direction of their velocity to the average velocity of their neighbors

The master agent group possesses additional behaviors, which determine how the agents respond to user interaction. These behaviors are described in the interaction subsection.

The movement of agents, which belong to the slave agent group, results from only one basic behavior: master following. At the beginning of a simulation run, each slave agent is assigned a particular master agent. Since there are a larger number of slave agents than there are master agents, several slave agents share a common master agent. The agent following behavior guarantees that the slave agents possess a similar spatial distribution as the master agents. In order to prevent slave agents from clustering right on top of their respective master agents, each slave agent follows the location of it's master agent at a fixed but initially randomized offset.

Synchronization

Slave agents synchronize by changing the texture coordinates and correspondingly the video frame region, which is displayed at the agent's position (see figure 1). The video frames are derived either from live or prerecorded video. The video source material is continuously fed into a ring buffer, which forms a 3D movie space (see figure 1, top left). The content of the ring buffer is transformed into a set of textures out of which random texture coordinates are assigned to the slave agents at the beginning of a simulation. In order to cope with the fact that agents may be assigned not only different regions within a texture but also different textures representing different frames along the time axis of the movie space, MediaFlies employs texture coordinates with three components: x, y, and time. The x and y components represent standard texture coordinates, whereas the time component indexes a video frame within the ring buffer.

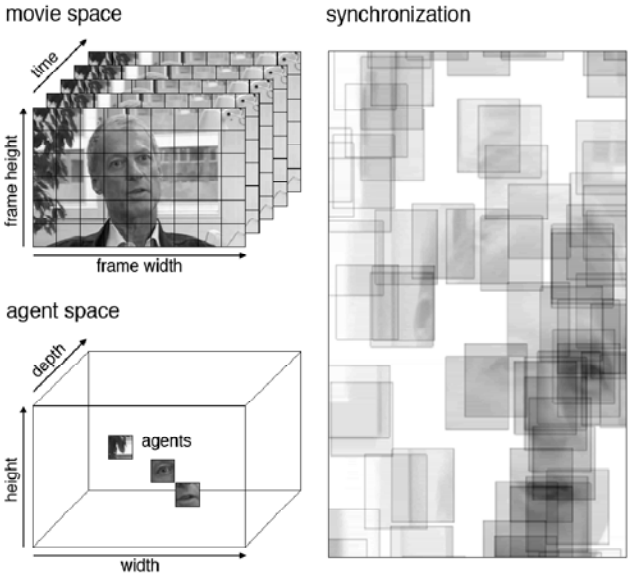

Figure 1.

Top Left: Movie Space. Captured frames are stored in a ring buffer according to their temporal order.

Bottom Left: Agent Space. Agents move within a 3d space possessing periodic boundary conditions.

Right: Synchronizing Texture Fragments. Agents compare the texture coordinates of their texture fragments with those of their neighbors and attempt to recreate the original relative spatial relationships of these texture coordinates.

While the slave agents move within their agent space (see figure 1, bottom left), they continuously compare the texture coordinates of their fragments with those of their neighbors. Synchronization is based on the following principle. Each agent possesses two sets of distance vectors that represent the spatial relationship between the agent and it's neighbors: agent space vectors and movie space vectors. Agent space vectors are calculated from the agent's position within agent space whereas movie space vectors are derived from the agent's texture fragment position in movie space. By transforming the agent space vectors into movie space, a set of target vectors is created for the movie space vectors. Synchronization attempts to match the movie space vectors and target vectors but shifting the former towards the latter. The amount of shifting is determined by a fixed synchronization rate. The effects of synchronization strongly depend on the flocking behavior of the master agents. As long as the agent space vectors change very little (e.g. movement velocity and direction of neighboring agents are coherent) and the group is fairly dense, synchronization succeeds eventually in recreating parts of the original video image. Since synchronization acts on a local level only, these recognizable regions may show up at different locations than in the original video and they may represent multiple copies of a single original feature (see figures 3 and 4, right). On the other hand, as soon as the coherence in the flock's movement diminishes, the effects of synchronization vanish and a cloud of unrelated texture fragments starts to form (see figures 3 and 4, left).

Interaction

Users are able to influence the flock's movement and thereby affect the amount of synchronization that takes place. A camera-based system is employed in order to track the positions and movements of one or several users. Tracking is mostly based on standard image segmentation techniques. The running average method (Picardi 2004) is employed to discriminate foreground and background. In order to detect movement, a motion history image is created based on which a motion gradient is calculated.

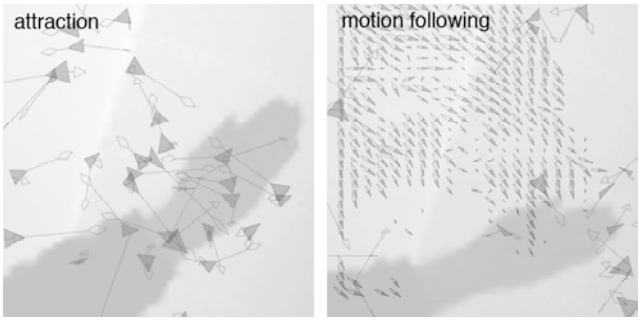

Figure 2

Left: Attraction. Master agents move towards the author's arm. Agents are depicted as triangles. The line extending from the agent's body into a rhombus shape represents the agent's velocity. The line, which ends into a triangular shape, represents the forces acting on the agent.

Right: Motion Following: Master agents follow a vector field that resulted from the author's arm movement. Motion vectors are represented as stylized compass needles whose darker sections point into the direction of movement.

In order to derive not only motion direction but also motion magnitude, a custom-processing step has been included, which extends the motion vectors along regions of constant brightness in the motion history image. The results of video tracking lead to the formation of an attraction scalar field and motion vector field (see figure 2). These 2D fields are coplanar with the x and y axis of the agent world and their scalar values and vectors have identical effects along the z-axis of the agent world. Master agents react to these fields via corresponding basic behaviors.

The "attraction" behavior allows agents to sense within their perception range the scalar values that result from object tracking. High values in this field correspond to the current foreground and low values to the current background. Intermediate values result from the diminishing effect of previous for- and background data. The force vectors produced by the "attraction" behavior causes agents to move towards regions of high value as well as towards the front region of the agent world (see figure 2, left). The "motion following" behavior creates force vectors that point into the averaged direction of the movement vectors that result from motion tracking (see figure 2, right). Once more, agents can only perceive movement vectors within their perception range.

The effects of attraction caused by relatively stationary users lead to an increased flock density and decreased flock speed at the corresponding position. Consequently, the effects in synchronization are no longer counteracted by diverging agent movements and therefore lead to the formation of dense regions that exhibit high similarity to the original video material. User movement leads to the opposing effect of disrupting highly synchronized agent groups and causes shifts and disturbances in the video feedback.

Visual feedback

The visual feedback of the MediaFlies software consists of scattered fragments of imagery that continuously shift their position and content according to the implemented flocking and synchronization algorithms. Depending on the flock's density and movement, these fragments form a highly transparent fog like amorphous mist or coalesce into distinct and opaque clumps of original video content (see figures 3 and 4). The fragments are associated with slave agents, who control their position, texture coordinates, size and opacity. Each fragment is implemented as a billboard and therefore always faces the viewer regardless of the agent's orientation. The number and size of slave agents has a large impact on the visual discrepancy between unsynchronized and synchronized agent groups. The visual appearance of the fragments can be further altered by changing the image source of their texture masks. These static masks are combined with video textures via multi-texturing. High contrast masks cause flickering and interference like effects when fragments shift across each other (see figure 4). Low contrast and smooth masks lead to "painterly" effects (see figure 3).

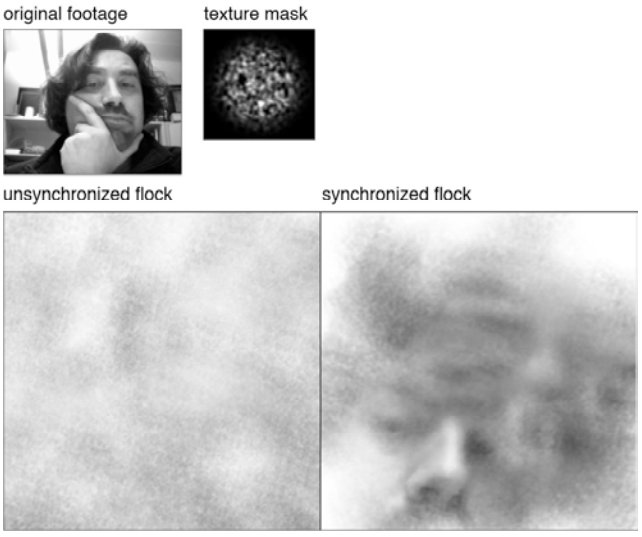

Figure 3: Visual Feedback Based on Life Video Input

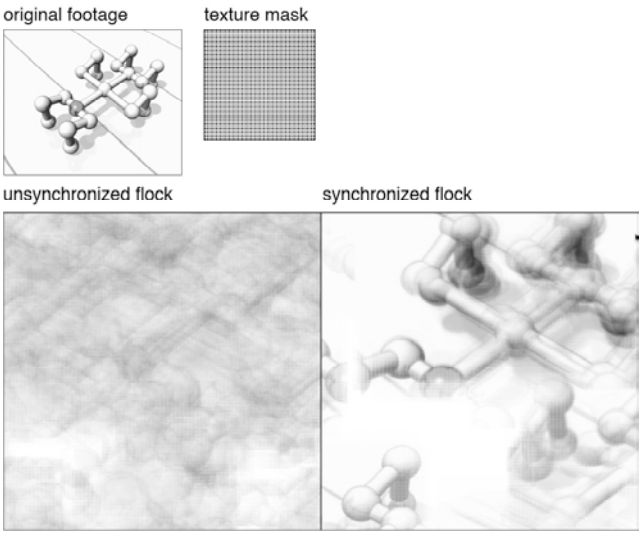

Figure 4: Visual Feedback Based on Prerecorded Video Input.

Results and Discussion

The first implementation of the MediaFlies system has been presented as an installation during the Tweakfest festival in Zurich, Switzerland (www.tweakfest.ch), which took place on November 9 and 10 in 2005. The visual feedback was projected via a video beamer. The video input to the system was provided by a miniDV camera, which pointed to the reception and cafeteria area of the festival space. The video input was periodically changed to that of a series of movie files. An iSight webcam pointed to a highly frequented corridor section and provided the means of interaction. The software ran on a Dual 2.5 GHz

G5 Apple computer. Due to the fact that the webcam wasn't positioned right in front of the projection area but somewhat offset and rotated, many visitors didn't realize that they could actually interact with the installation. On the other hand, most visitors with whom I talked were surprised by the way the visuals seemed to reflect the situation and mood in the space. During bustling moments shortly before talks, people rushed through the corridor, thereby creating a fast paced shifting of conglomerates of seemingly unrelated image fragments. These moments contrasted with more tranquil situations during breaks in the festival program, when people sat together in the cafeteria. The MediaFlies installation responded to this type of situation by creating slowly moving clusters of image fragments that clearly depicted visual details of the cafeteria situation.

The current version of the MediaFlies system suffers from a few shortcomings, some of which will be further addressed in the "conclusion and future work" section. The computational demands of the system are somewhat daunting. In order to achieve a high degree of fracturing and blending of the video input, a large number of slave agents needs to be simulated. This of course conflicts with the desire to display very fluid and smooth fragment motions, which requires a high frame rate. The way texturing is handled also puts a significant load on the system. Textures are steadily updated based on new video input and therefore need to be continuously transferred to the video card. At the moment, the video tracking system also needs improvement since it is not very robust and requires a very long background calibration phase. Finally, we consider the fact that MediaFlies can only deal with video but not audio as highly unsatisfactory. The capability of the system to grasp and reproduce moods in the environment will be significantly improved by adding audio capabilities.

Conclusion and Future Work

We would like to start the conclusion section by expressing a few thoughts on interactive and self-organized installations in general. Based on our experience with such installations (Unemi and Bisig, 2005. Unemi and Bisig, 2004, Bisig, 2003) we come to the conclusion that these installations possess very promising potentials but still pose significant challenges with regard to appropriate forms of interactivity. Most visitors are unfamiliar with installations whose interactivity doesn't possess a clear and always reproducible input-output relationship (contrary to purely reactive installations). Depending on the presentation situation, visitors are hardly willing to take their time in order to explore and learn new and more adequate forms of interaction.

On the other hand, these types of installations are capable of responding in a much more refined and surprising way than reactive installations. The diversity of reactions and their capability to sustain particular behaviors even in the absence of the initial triggering inputs usually baffles and impresses visitors. We believe that frequented spaces outside of an exhibition context constitute the most promising setting for interactive and self-organized installations. In these settings, the variety of interaction possibilities can match the diversity of the systems behavior. Short and sustained forms of interaction can coexist and interaction can be deliberated or simply become a byproduct of people's normal activities. In addition, visitors are free to choose any attention level they like, ranging from blissful ignorance to concentrated observation.

Based on the feedback we received from visitors of the Tweakfest festival, we believe, that the MediaFlies system has succeeded in providing interesting and aesthetically fascinating forms of visual feedback. This positive feedback was mostly based on two aspects of the system: it's capability to continuously display and destroy recognizable fragments of video material (in particular live video material), and its way of responding to (mostly unintentional) forms of interaction. In order to obtain further feedback, we would like to test MediaFlies in a somewhat different setup, which is hopefully more encouraging for direct and intentional forms of interaction.

We believe that the concept and current realization of the MediaFlies system has sufficient potential to justify further development. First and foremost, we are currently working on an audio extension of the system, which will possess similar functionality as the video processing part. Live or prerecorded audio material will be cut into fragments both in the time and frequency domain. These fragments will be rearranged into a new audio stream. The technique to achieve these effects will be based on Fast Fourier Transformation and Granular Synthesis.

A second and more fundamental extension of the system deals with the relationship between agent behavior and media material. Currently, the agents are entirely unaffected by the quality of the media material they represent. We would like to introduce a feedback mechanism between the visual and acoustical output of the system and the agent's behavior. By allowing agents to change their behaviors depending on the media fragments, which they and their neighbors present, novel forms of algorithmic media recomposition could be explored. One possible way to achieve this could consist of letting the media material affect some of the physical or behavioral properties of the agents. This effect can be based on structural (e.g. spatial and temporal color distribution) or semantic properties (e.g. emotional quality) of the media material. Using such a system, agents may for instance learn structural and statistical properties present in one type of input media and try to reconstruct these properties when presenting a different type of media.

References

Bisig, D. 2003. *BioSonics - Interactive Growth System*. In Proceedings of the Generative Art Conference.

Blackwell, T.M. 2003. *Swarm music: improvised music with multi-swarms*. Artificial Intelligence and the Simulation of Behaviour, University of Wales.

Boyd, J.E., Hushlak, G., and Jacob, C.J. 2004, *SwarmArt: interactive art from swarm intelligence*. Proceedings of the 12th annual ACM international conference on Multimedia. 628-635. New York, NY, USA.

Picardi, M. 2004. *Background subtraction techniques: a review*. University of Technology Sydney Workshop on Computer Vision & Image Processing.

Ramos, V, Moura, L. and Pereira, H. G. 2003. *ARTsBot – Artistic Swarm Robots Project*. http://alfa.ist.utl.pt/~cvrm/staff/vramos/Artsbot.html

Reynolds, R. W. 1987. *Flocks, herds, and schools: A distributed behavioral model*. Computer Graphics, 21(4):25-34.

Shiffman, D. 2004. *swarm*. Emerging Technologies Exhibition. Siggraph, Los Angeles, LA, USA.

Sommerer, C., and Mignonneau, L. 2000. *Modeling Complex Systems for Interactive Art*. In Applied Complexity - From Neural Nets to Managed Landscapes, 25-38. Institute for Crop & Food Research, Christchurch, New Zealand.

Thatcher, U. 1999. *Notes on Spatial Partitioning*. http://tulrich.com/geekstuff/partitioning.html

Unemi, T. and Bisig, D. 2004. *Playing music by conducting BOID agents.* In Proceedings of the Ninth International Conference on Artificial Life IX, 546 - 550. Boston, USA.

Unemi, T. and Bisig, D. 2005. *Music by Interaction among Two Flocking Species and Human*. In Proceedings of the Third International Conference on Generative Systems in Electronic Arts, 171-179. Melbourne, Australia.