SwarmVis: a Tool for Visualizing Swarm Systems

Don Miner Niels Kasch

***Abstract*—In this paper, we provide an overview of SwarmVis, a tool for visualizing swarm systems. SwarmVis is able to create informative still images and videos of swarm systems moving in two- and three- dimensional spaces through the use of several visualization techniques. We discuss what properties of swarms need to be conveyed and then explain in detail how we tackled each of these individually in SwarmVis. Finally, we present a case stud to demonstrate how SwarmVis can be used to analyze both the low-level and swarm-level behaviors in a multi-agent system.**

I. Introduction

Swarm systems are comprised of a group of agents that exhibit some collective behavior. Examples of these systems include: social animals (e.g. ants[1], schooling fish[2], migrating geese[3]), and multi-robot systems[4][5]. Visualizing swarm systems (swarms) effectively is difficult due to the large number of individual agents contained. In addition, a typical swarm's high density and chaotic motions amplifies the complexity of swarm visualization. Effective swarm visualizations are needed to provide insight into swarm behavior and to inspect local interactions. A well defined set of visualization techniques would provide researchers with these insights.

Crude swarm visualization techniques plot the position of each agent in space, as seen with a Reynolds boid flock[3] in Figure 1 (left). A still image only shows the location of individual agents and does not convey important information such as the direction, velocity, and previous positions. We have implemented *SwarmVis*, a toolkit for visualizing swarms, that goes beyond the simple plotting technique to solve this problem. The toolkit aims to provide visualization techniques that allow researchers to interactively investigate swarms, making the study of interactions, fine-grained movements, and swarm behavior possible. Figure 1 (right) displays a visualization of a swarm system using SwarmVis' trails feature, which conveys much more information.

The goal of SwarmVis is to facilitate, through visualization, the detailed analysis of emergent behavior that results from swarm systems. In order to satisfy this objective, designed SwarmVis around the following three requirements:

- The toolkit shall visualize swarms using a series of still images (i.e. an animation) embedded in an interactive user interface that has features such as pausing, slowing down and rewinding of frames.
- Agent position, direction and velocity shall be conveyable through both still images and videos generated from the software.
- The toolkit shall provide a variety of built-in visualization techniques that are useful in visualizing swarm systems.

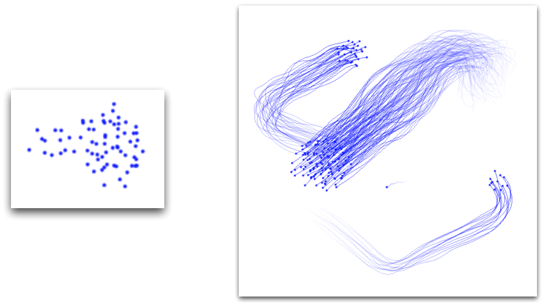

Fig. 1. A basic visualization of a swarm that only plots points in space (left). No information other than agent position and some structural features is provided. A basic visualization of a swarm that only plots points in space as well as trails of significant length (right). Important properties such as direction, velocity, structure, rotation, and previous positions are all naturally conveyed.

In this paper, we discuss previous work in swarm system visualization, some of which have provided inspirations for the techniques in SwarmVis. Then, we discuss the implementation details of the user interface and graphical visualization. We conclude with a demonstration of the effectiveness of SwarmVis by using it to explore and analyze a swarm system. A brief user guide for compiling, running and loading data in SwarmVis is provided in the appendix.

II. Related Work

Not much work has been done specifically tackling the problem of effective swarm visualization. Most visualizations are results of swarm intelligence research itself and the majority of articles do not discuss details relevant to our work. Therefore, we often pulled inspiration from figures in papers that had nothing to do with visualization and more to do with the swarm system itself. Comprehensive multi-agent and swarm system frameworks [6][7] have been created in the past, but the visualization aspect is only a component and not the main focus. Some researchers have implemented visualization techniques for specific swarm domains, such as the particle swarm optimization algorithm[8] and boid flocking[3]. There have been several projects that used a swarm system paradigm to visualize some sort of information or domain, such as data variations[9], art[10], evolutionary algorithms[11][12], flow[13][14], and source code commits[15]. We typically do not care about the domain, but we are often able to get inspiration from the articles' figures, even if they are not directly explained from a visualization point of view. In this section, we discuss two projects that have motivated particular

Fig. 2. A visualization created by Boyd et al. with SwarmArt[10].

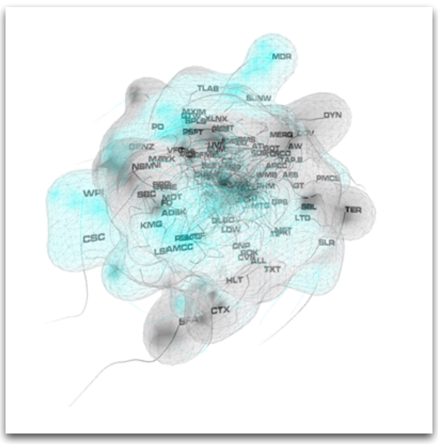

Fig. 3. An “information flocking” visualization created by Moere[9].

features in SwarmVis.

A. SwarmArt

SwarmArt by Boyd et al.[10] is a combination of science and art with the aim of this project was to create visually appealing swarms. SwarmArt visualizes the positional data of a swarm by projecting individual agents onto a 2D surface by representing agents as dots and distinguished from another by color. Agents are visualized in motion by plotting the agents' positions as they change over time by appending an agent trail that fades with time indicating current and past positions. A sample image created with SwarmArt is shown in Figure 2. This work, among others[15], was an inspiration for how we visualize agents as simple dots with fading trails showing previous positions.

B. Information Flocking

Moere demonstrates how self-organization and behavior as seen in Reynolds' Boids can be used to show the change of information over time[9]. This method is dubbed *Information Flocking* and combines the flocking behavior rules with clustering techniques to give users a unique perspective of changing data. The domain presented in this paper and the method of information flocking are not directly relevant to our project. However, the visualization of the agents this author uses is incredibly smooth looking and pretty to look at. Each agent is a single particle in 3D and its traveled path is depicted by a colored, curved line that has gradually decreasing opacity from the head. The tails are made smooth by using the *Catmull-Rom spline* algorithm, which generates smoother tails than linear interpolation through the previous points the agent has visited. The particles' color shows change in data: blue for a decrease and red for an increase.

In addition, the visualization may include an encompassing “blob” that shows vicinity (data similarity). That is, if particles are in the same block, they are in the same flock. This blob method also helps the user find outliers. These blobs are generated with Marching Cubes; specifically a method called “implicit surface polygonizer.” Unfortunately, computing the blob is too computationally intensive to be displayed every time step. The author claims that this simulation was able to run in real time with 500 agents.

This work also motivated us in having SwarmVis show agents as a particle with a fading trail. However, we decided against using a spline method to show smoother trails because the splines to not actually pass through the points the agent has, which in some cases may be misleading. Using color in the trail motivated us to as well, although SwarmVis does not in the same way. The “blob” shown in this work is interesting and informative; we may implement this feature in SwarmVis in the future.

III. Implementation

SwarmVis is mostly implemented in the C++ language and utilizes the Qt framework[16] to build the graphical user interface and to imbed an OpenGL window. The user interface is split down the center, the left side containing the majority of the user interface and the right side containing the graphical display. The visualization frame in the right side can be rotated and zoomed by dragging and mouse-wheeling, respectively. In general, we tried to keep the interface as simple and straightforward as possible. A more detailed description of the user interface is given in Figure 4. Changes made to the visualization through the interface happen in real-time, as the visualization is playing. Lists and parameters are populated when a data set is loaded (more information about data sets is available in the appendix).

Still images can be created by SwarmVis by dumping the graphic currently shown in the visualization frame. Also, videos can be made by having SwarmVis automatically dump every frame in the course of an animation as an image. These images can then be compiled into a video by a third-party program that is used to make stop-motion or time-lapse videos.[1]

In the remainder of this section, we list the available visualizations in SwarmVis, what they convey, how they are

[1]Boid flock and tetrahedron-forming swarm videos made by SwarmVis: http://maple.cs.umbc.edu/ don/projects/SAF/videos/#tetra

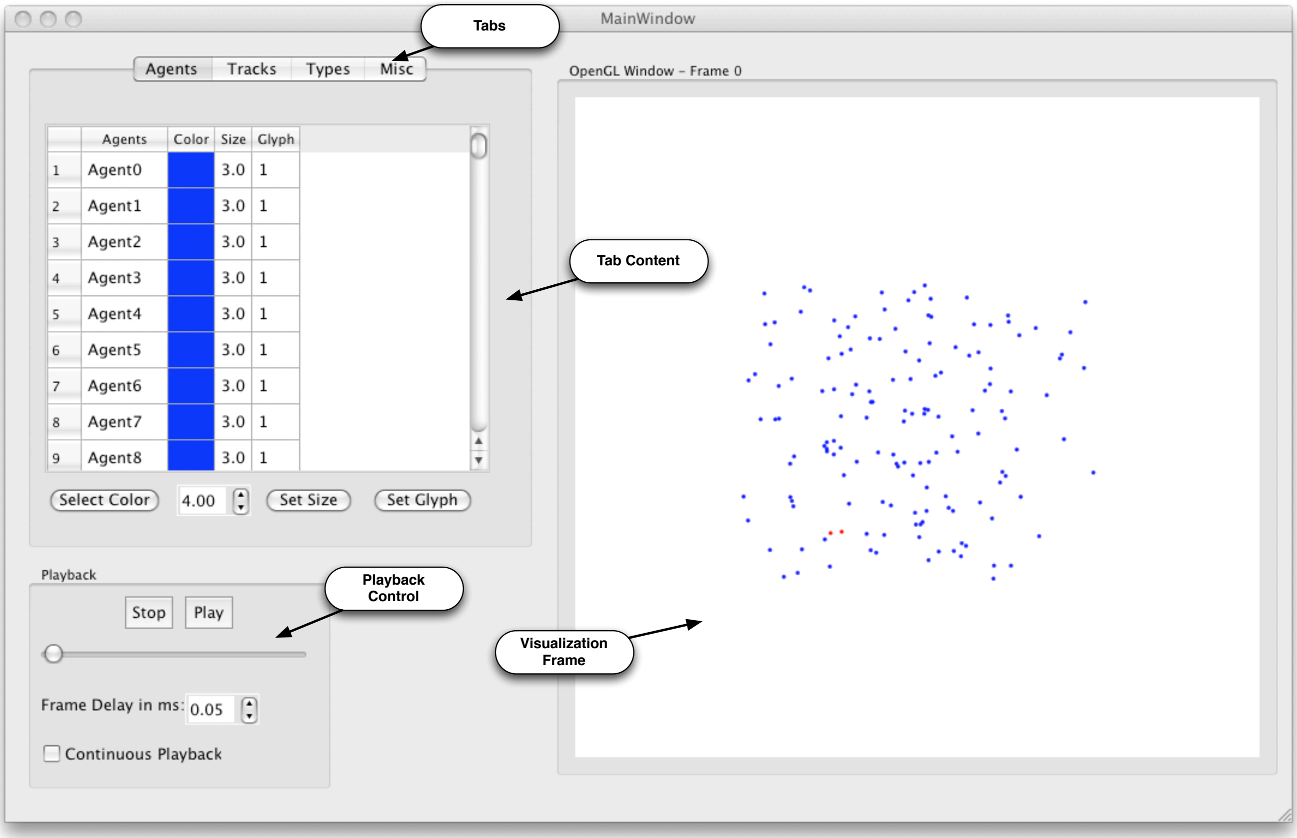

Fig. 4. The SwarmVis graphical user interface. here is only one window in our application and from it you can navigate to all commands and visualization options. The interface is split between the user controls and the visualization itself. There are tabs at the top left that give options for different visualizations in the tab content frame. The playback frame in the bottom left works similar to a standard media player. The visualization frame shows the swarm with all selected visualizations applied to it. In this screenshot, the user has two agents colored red with all other options standard.

Fig. 5. Three captured frames from a swarm animation of three-dimensional flocking boids[3]. When viewed in quick succession as frames in a video, the user is able to detect the flocks' motion and structural changes.

implemented, how they can be enabled, and how they can be modified.

A. Visualizations

1) Animation: The playback control (as seen in seen in Figure 4) gives the user the ability to specify what is being shown in visualization frame. The inspiration for this control is a standard media player with the ability to stop, play and manually move through frames (with the slider). In addition, the user can specify how fast the animation is created by adjusting the amount of delay between frames ("frame delay" box).

Viewing a swarm over several time steps conveys a great deal amount of information. It conveys direction, velocity and structure of the swarm by being able to see how it changes from time step to time step. SwarmVis manages to be incredibly smooth even with a fast frame rate and hundreds of agents on a modern computer.[2] An example of this is shown in Figure 5, in which a swarm is shown to be generally moving south with its sub-swarmsbeginning to merge. All of our visualizations that can be applied to the swarm (color, size, trails, tracks) can be seen in fluid animation.

2) Color and Size: An important feature of SwarmVis is the ability to track individual agents. This can be done by selecting any number of agents of interest in the list of agents (seen in the content frame of Figure 4) and changing their color or size with the button controls. For example, the corner agents have been made bigger to emphasize their importance in the tetrahedron swarm shown in Figure 7.

In addition, users can change the color of all agents that are members of a specific group in the Types tab, as seen in Figure 6. This feature can be used in addition to changing the size to

[2] 2.2 GHz MacBook Pro running Mac OS X 10.5

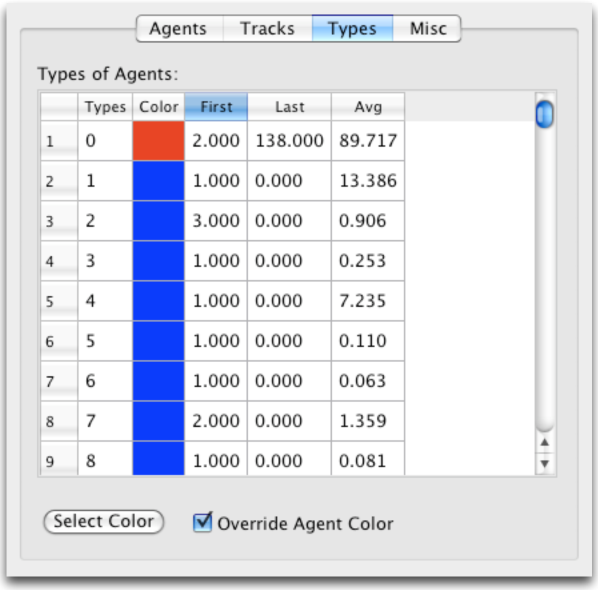

Fig. 6. The Types tab shows the user which different subgroups exist in the swarm and allows the user to assign a specific color to that group. Supplemental information helps the user select the groups he wants: First (number of agents in that group in the first frame), Last (number of agents in that group in the last frame), and Avg (average number of agents in that group during the entire animation).

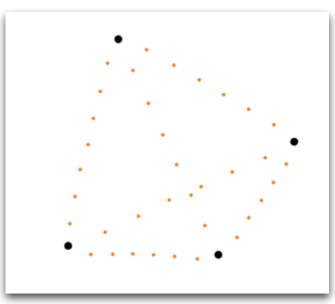

Fig. 7. A captured frame from a swarm of tetrahedra-forming boids. The corner agents have been given a larger size and a darker color than the edge agents.

emphasize particular agents. For example, in the swarm shown in Figure 7, we color the corners black and the edges orange. Another example is shown in Figure 8, in which we colored the largest swarm red and left the other swarms blue. Changing the color of the largest flock in the flocking domain is particularly useful because it shows when two groups merge by changing the agents' colors as they collide. Also, this feature is great for creating figures for published articles in swarm research because by changing the color and size, the writer can refer to specific agents as "the bigger agents" in the text instead of having to manually add labels to the images.

3) Trails: Trails are one of the most important features in SwarmVis because it conveys motion, previous positions, direction, and change in structure, both in still images and videos. Trails are lines that hang behind agents, tracing their

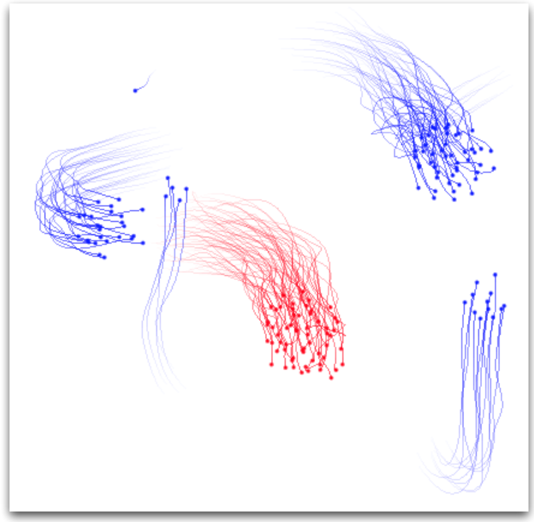

Fig. 8. A captured frame from the three-dimensional flocking swarm shown in Figure 5. The trails in this image are of length 35 time steps and agents of a particular group are colored red.

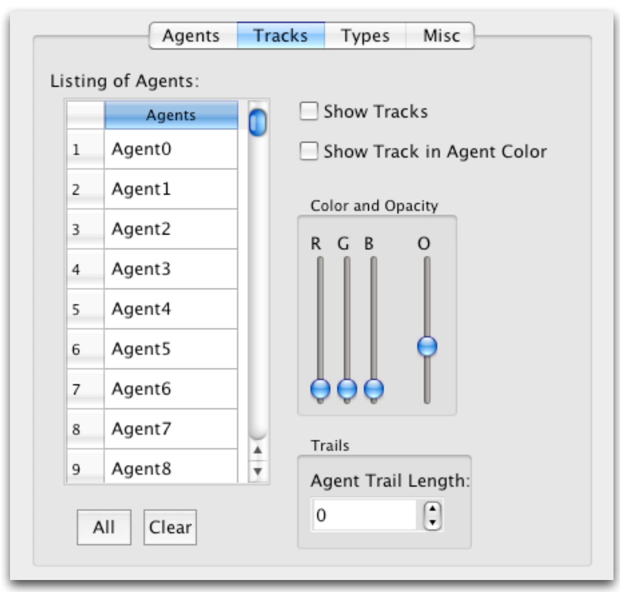

Fig. 9. The Tracks tab allows users to add tracks and trails to agents. The length of the trails can be adjusted with the "Agent Trail Length" box.

previous positions. The trails are created by connecting previous positions of the agents with successive lines that are more opaque near the current agent and less so further back in time. The length of trails can be adjusted by modifying the number in the "Agent Trail Length" text box, shown in Figure 9.

Trails can be used to view swarm behavior at an abstract level, such as tracking the motion of several boid flocks at once, as seen in Figure 8. One of the most powerful abilities of trails is to convey low-level behavior information in swarms. For example, in Figure 10, we can see that the rightmost corner

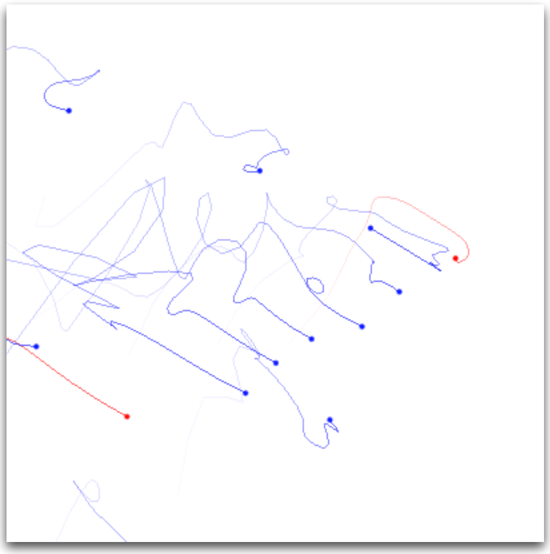

Fig. 10. A captured frame from a swarm of tetrahedra-forming boids shown in Figure 7. Close up views of the swarm can convey much information about how agents interact with one another. In this frame, we see how the rightmost red corner agent swapped places with the rightmost edge agent. Then, edge agents fill in the gap by moving to the right. The different times in which each adjustment happened is shown by how long ago a pattern happened in the trail.

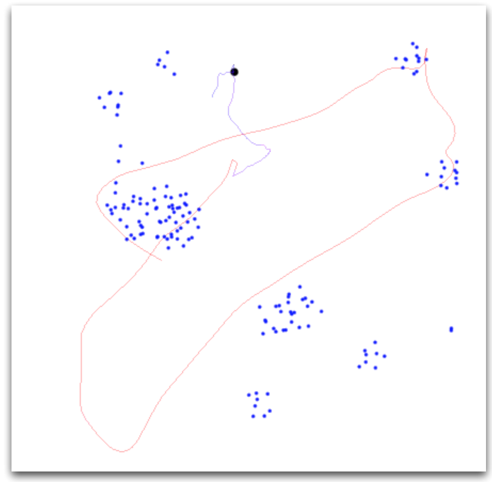

Fig. 11. A captured frame from the three-dimensional flocking swarm shown in Figure 5. The lonely agent (marked by being larger and darker) in the middle follows this path over its lifetime. The path is colored based on velocity, showing that it moved rather slowly while being solitary (blue portion). The agent then sped up after being picked up by a flock (red portion). This even is inferred by the "notch" as well as the change in color between the blue path and the red path. The structure of the line in three-dimensional space is seen much better when able to interactively rotate the view.

agent (colored red) swapped position with the rightmost edge agent (colored blue). Then, all the agents in that edge shifted down to accommodate. The time delay in behavior is shown by the "notch" in the path happened further ago for the left agents and more recently in the right agents. This visualization feature makes qualitative analysis of how the agents behave both at a low interaction level and a high swarm emergent behavior level possible.

4) Tracks: The Tracks feature is very similar to the trails feature (they are implemented the same way), except that it shows the position of the agent over the entire lifetime of the system. At any given frame, the agent will be somewhere on the track For example, a track of a single agent is shown in Figure 11. Several agents can be selected at once in the list of agents in the Tracks Tab (Figure 9) to have their trails all shown at once with the color specified by the "Color and Opacity" sliders. When tracks are used in conjunction with trails, the trails are overlaid on top of the tracks so that both trail and track information is conveyed.

5) Velocity Color: The velocity feature is useful for displaying the instantaneous or historical velocity of the agents. This visualization can be applied in two different ways: color the agent and its trail the agent's current instantaneous velocity and/or color the agent tracks based on the velocity at each time step on the track. SwarmVis takes the distance travelled in one time step to determine the velocity of an agent and then calculates the desired color from the gradient. The color gradient is determined by making pure red the greatest velocity seen in the data set and making pure blue a velocity of zero

Fig. 12. The velocity tab allows users to map agents' velocities to a color, shown in the gradient on the right. The "Show Agent Velocity" option will color the agent and its entire trail (if enabled) more red if the agent is moving relatively fast, or more blue if the agent is moving relatively slow (seen in the right image of Figure 13). The "Velocity in Tracks" option makes it so the agent tracks are colored according to speed at that time step (seen in the left image of Figure 13).

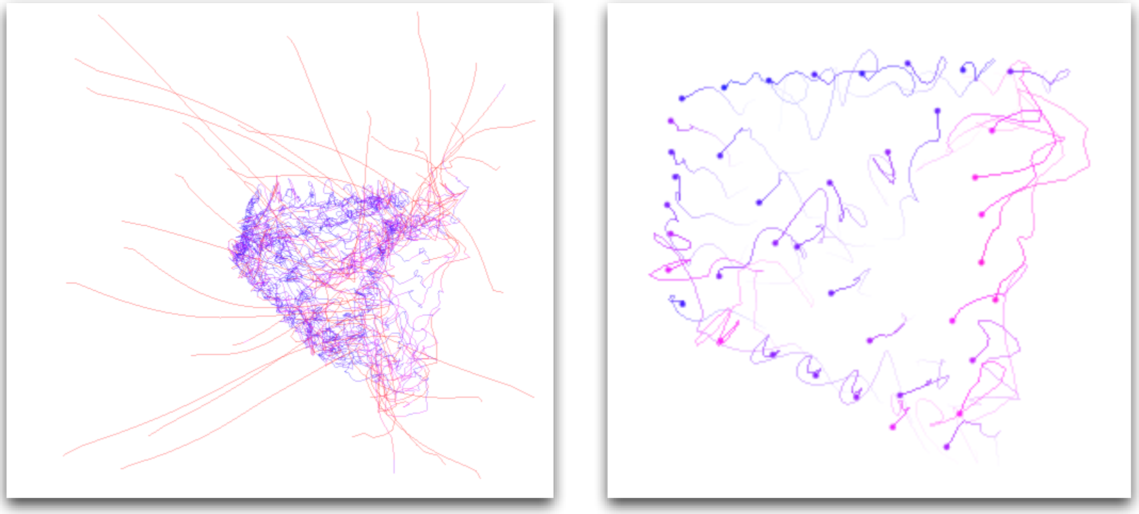

Fig. 13. A captured frame from a swarm of tetrahedra-forming boids shown in Figure 7. On the left, all agent tracks are displayed, showing that agents move faster (the color red) at the beginning. Once the agents organize into the tetrahedron they slow down (the color blue). On the right, the trails and the agents themselves are colored according to their instantaneous velocity. From this captured frame, it is obvious that the right edge is experiencing movement while the rest of the shape is calm.

(as seen in Figure 12).

Tracking velocity in the color of the agent is used to track the instantaneous velocity of agents in any time step. While viewing the swarm as an animation, the change in velocity is conveyed by seeing how the color changes. In addition, important information can be gleaned by viewing still images of agents colored by velocity because agents that are moving faster are colored differently than if they are moving slower. An example of this is shown in Figure 13 (right). We decided that coloring the entire trail the instantaneous velocity (opposed to its default color) makes the color more obvious. There is no actual information contained in the trail's color, as it is the same color as the agent.

Coloring tracks based on the velocity is a powerful feature that displays how an agents' or an entire swarm's velocity changes over time. Every line segment in the track is colored based on the velocity of the agent at the time step the line segment represents. A single agent's track is pictured in Figure 11, in which we can see how the velocity of this particular agent changes over the course of its lifetime. In contrast, in Figure 13 (left), every agent has its track shown. From this image, we can determine that the movement is very fast and chaotic in the initial phases but then calms to a slower speed once the agents have organized into the shape.

IV. Results: Analysis of the Tetrahedron-Forming Swarm

Throughout the course of this document, we have discussed the motivation behind the visualizations in SwarmVis by displaying images from two data sets: a boid flock and a tetrahedron-forming swarm. In this section, we will a cohesive and comprehensive walkthrough in how a user would analyze the tetrahedron-forming swarm displayed in previous images.

A. Background

The shape is formed by agents adhering to the following procedure. First, four corners are selected by the swarm via a simple election algorithm. The agents follow the following rules:

- If a corner, be attracted to the three other corners very slightly.
- If not a corner, be attracted to the two closest corners with great, but equal, strength. That is, the output force vector of this rule is the sum of two vectors of some constant magnitude. Note that when an agent is on the line formed by its two closest corners, the forces cancel each other out.
- Avoid any agents that get too close.

The end result of these rules are somewhat obvious: the agents will self-organize into a tetrahedron. However, many of the fine-grained details in the agent behavior is not clear. Some questions one may want to ask about this system are:

- Will the shape formed be equilateral?
- How stable is the shape once it is formed? Does it reach an equilibrium and stop moving? If not, what is changing?
- How do agents react to movement in the tetrahedron?
- How fast do agents create a shape that appears to be a tetrahedron?
- Are there any interesting "emergent strategies" that this swarm uses to create the shape?

These questions and are easily answerable by studying the rules or the raw position data. The purpose of SwarmVis is to provide users the power to answer these questions, as well as identify emergent behaviors not previously expected to exist in some systems.

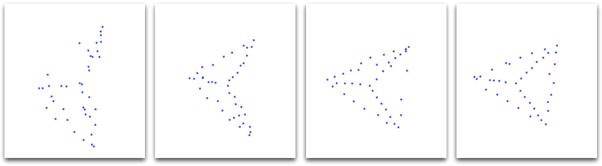

Fig. 14. An emergent strategy that the tetrahedron-forming swarm follows is to form two triangles with a shared edge, then close in on itself to form the last edge. This is easily seen by watching the swarm in animation.

The particular tetrahedron-forming swarm data set we will be analyzing in this section has forty agents and has 333 individual frames of agent coordinates.

B. SwarmVis Analysis

SwarmVis has several tools available to analyze swarms and to create informative videos and still images. To demonstrate the results of our work, we provide an example of how the tetrahedron-forming swarm can be analyzed. The following three general approaches discussed in this section are only a starting point in studying this domain. There is always something else to be found in a swarm system and only requirement is the user to looks for it. SwarmVis enables the user to find these points of interest.

1) Animation: Several interesting observations can be made by viewing the swarm through plotted points in animation, without any additional visualizations. From this animation, we can determine that the swarm starts out spread thin with no organization. Then, agents move towards the center and follow an emergent strategy (i.e., not explicitly part of the agent rules) to form the six corners of the tetrahedra: form two triangles with a shared edge (five edges) then close the last edge in on itself, forming the sixth. Agents executing this strategy are seen in Figure 14. After this, we can see the swarm moving towards the center of the screen in a stable manner. Then, all of a sudden, agents start bouncing back and forth from edge to edge, not permitting the tetrahedron to reach an equilibrium. Other than these obvious facts, not much more information can easily be inferred by watching this simple animation. In addition, it is difficult to convey what one sees in an animation in written format with figures.

Our first step in making more useful visualizations is to color the corners different than the rest of the agents so that they are easier to track. We can see in the Types tab (Figure 6) that there are two groups, labeled “corner” and “edge” (agents are labeled in the data set). Although the labels are intuitive, we can also infer that the second group in the list is contain the corner agents because it has an average of 3.988 agents over the lifespan of the swarm. Since a tetrahedron has four corners, it is safe to assume that this group contains the corners we wish to color. This is important because the corner agents follow different rules than the edge agents and are used as markers for the swarm. At this point, our visualization shows a tetrahedra similar to that in Figure 7.

Videos can be made by recording the visualization frame. Creating videos would be an appropriate way to show information conveyed by an animation in a setting that does not require the visualization to be printed (e.g., online or presentation).

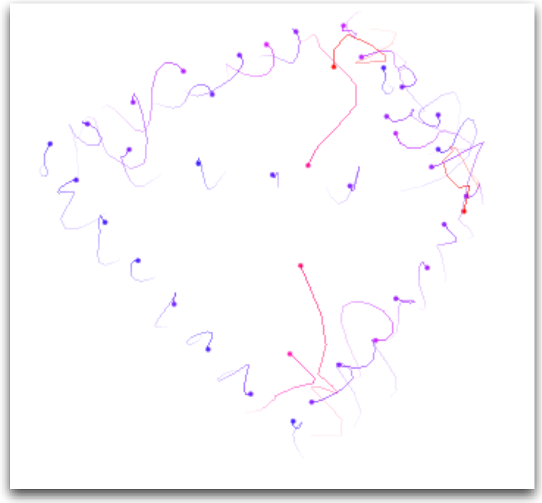

Fig. 15. The tetrahedron-forming swarm creating its sixth edge. Short trails are turned on with agent coloring based on velocity enabled.

2) Trails: Previously, we described how trails conveyed important information in the frames shown in Figures 10 and 13. We now add another example of how powerful of a tool agent trails are in creating compelling still images by exploring how the tetrahedra-forming swarm is completing the sixth edge of the shape by adding trails along with agents coloring according to velocity. The trails show where the agents are coming from and where they are going and with the addition of the velocity coloring we can easily see which agents are shifting significantly. If zoomed in close to the sixth edge as it closes and taking a still image at the appropriate time (which we found with moving our slider slowly), we receive the image shown in Figure 15. From this image, we can tell that the new edge agents move to the new empty edge are agents closest to the corners. These agents move very quickly, as seen by their red coloration. Meanwhile, agents from the edges these quick agents are coming from shift downwards to fill in the gap and are colored purple due to their slight movement. At this point, we feel we have a detailed understanding of how the last edge is formed through this emergent strategy. Without trails, this phenomena could not have been demonstrated in a single image (e.g., see Figure 14).

3) Tracks: Previously, we showed that velocity colored tracks can be used to get a high-level view of the swarm behavior, as seen in Figure 13. Colored tracks can also be used to infer information about an individual agents’ behavior as well. To do this, we cycle through several agents listed in the Tracks tab. Very quickly, one realizes that there is a significant difference between the behavior of a corner agent and an edge agent. A comparison of two agents is given in Figure 16. From the image showed in this figure, we see that the life of an edge agent is far more chaotic than that of a corner agent. The corner agent barely changes position

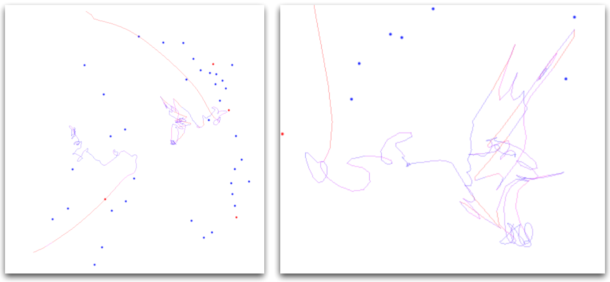

Fig. 16. Two velocity colored tracks compares the difference between a corner's track and an edge agent's track (left). The corner agent gets into position and remains relatively calm for the remainder of the time. Meanwhile, the edge agent goes through several high-speed moments and never reaches a steady state (right).

after reaching its desired location. On the other hand, the edge agents movements are both chaotic in terms of position, but also change in velocity. In some situations, selecting agents that end up near one another can convey information about how the change in velocity of one agent affects its neighbors.

V. Future Work

We have demonstrated that SwarmVis is a powerful and dynamic tool. It is a complete application that has accomplished all the goals that we set out to achieve. However, there are a few user interface changes that would make usability better. Also, we would like SwarmVis to visualize the instantaneous velocity and the depth of agents since this information is not explicitly present in the graphical display. These future improvements are discussed in this section.

In the future, we would like to make it easier to select agents and determine which agents are which in the graphical display. To facilitate this, we hope to be able to show text labels that are adjacent to the agent in the visualization window. These labels can show the name of the agent in the agent list. Also, the ability to click agents in the visualization window will allow for greater interactivity and easier modification of agent-level colors and effects.

Currently, each visualization effect is controlled by a separate procedure. Therefore, separate agent lists are kept in the Agents tab, the Tracks tab and the Types tab. To make changes to an effect requires using that effect's tab and any changes made (e.g., color) are not reflected in the other lists. Having an interactive global agent list that displays all necessary information will enable users to modify the colors and effects on groups of agents more effectively.

We also plan on adding the feature to change the representation of the agent itself to better display direction. Currently, direction is inferred by the user by viewing the agents' trail, which may not be effective in every situation. If agents were represented as three-dimensional arrowheads or perhaps other glyphs, still images would convey very clearly the direction the agent is going on.

To further enhance the information conveyed in still images, we plan on determining a sufficient way to display depth. We are not sure what the best technique for this would be, and no previous work to our knowledge has specifically dealt with this problem.

Finally, we plan on making SwarmVis extendable through a plug-in type framework. Any new visualization effects that are added to SwarmVis must be hard coded into the source. This is not intuitive for a user that does not have knowledge of the source code who wishes to add his own visualization to SwarmVis. Plug-ins could be simple programs that take the agent data as input and return what the plug-in would like to have drawn on the screen. For example, the agent trails effect could be implemented as a plug-in, such that it returns a list of lines to be drawn. A plug-in user interface will have to be implemented that allows users to manage plugins, as well as use them. There should be some way for the plug-ins to interface with the global agent list to add informations to columns and access information.

References

[1] I.D. Couzin and N.R. Franks, "Self-organized lane formation and optimized traffic flow in army ants", in *Proc. R. Soc. Lond. B*, 2003, vol. 270, pp. 139–146.

[2] J.K. Parrish, S.V. Viscido, and D. Grunbaum, "Self-organized fish schools: an examination of emergent properties", *Biological Bulletin, Marine Biological Laboratory, Woods Hole*, vol. 202, no. 3, pp. 296–305, 2002.

[3] C. W. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model", in *Computer Graphics, 21(4) (SIGGRAPH '87 Conference Proceedings)*, 1987, pp. 25–34.

[4] F. Mondada, G.C. Pettinaro, A. Guignard, I.W. Kwee, D. Floreano, J.L. Deneubourg, S. Nolfi, L.M. Gambardella, and M. Dorigo, "Swarm-Bot: A New Distributed Robotic Concept", *Autonomous Robots*, vol. 17, no. 2, pp. 193–221, 2004.

[5] J.D. McLurkin, *Stupid Robot Tricks: A Behavior-Based Distributed Algorithm Library for Programming Swarms of Robots*, PhD thesis, Massachusetts Institute of Technology, 2004.

[6] S. Luke, C. Cioffi-Revilla, L. Panait, Sullivan K., and G. Balan, "Mason: A multiagent simulation environment", *Simulation*, vol. 81, no. 7, pp. 517–527, 2005.

[7] J. Klein, "breve: a 3d environment for the simulation of decentralized systems and artificial life", in *ICAL 2003: Proceedings of the eighth international conference on Artificial life*, Cambridge, MA, USA, 2003, pp. 329–334, MIT Press.

[8] B.R. Secrest and G.B. Lamont, "Visualizing particle swarm optimization - gaussian particle swarm optimization", *Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE*, pp. 198–204, April 2003.

[9] A.V. Moere, "Time-varying data visualization using information flocking boids", *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pp. 97–104, 0-0 2004.

[10] J. Boyd, G. Hushlak, and C. Jacob, "Swarmart: interactive art from swarm intelligence", in *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, New York, NY, USA, 2004, pp. 628–635, ACM.

[11] L. Spector, J. Klein, C. Perry, and M. Feinstein, "Emergence of Collective Behavior in Evolving Populations of Flying Agents", *Genetic Programming and Evolvable Machines*, vol. 6, no. 1, pp. 111–125, 2005.

[12] L. Spector and J. Klein, "Evolutionary dynamics discovered via visualization in the brave simulation environment", in *Eds.), Workshop Proc. of ALife VIII, UNSW*. 2002, pp. 163–170, Press.

[13] J. Kruger, P. Kipfer, P. Kondratieva, and R. Westermann, "A particle system for interactive visualization of 3d flows", *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 6, pp. 744–756, 2005.

[14] W. Merzkirch, *Flow Visualization Second Edition*, Academic Press Inc. (London) Ltd., 1987.

[15] M. Ogawa, "Code swarm", Website last accessed September 2008, http://vis.cs.ucdavis.edu/ ogawa/codeswarm/.

[16] "Qt", Website last accessed November 2008, http://trolltech.com/.

APPENDIX
USER GUIDE

A. Compiling and Running

SwarmVis was created to run on Unix-like systems, such as Mac OS X and GNU/Linux. The following programs are required to compile SwarmVis:

- Qt 4.2
- gcc/g++ 4.2.2
- make

Note that some libraries from Qt4.2 may be needed to run SwarmVis in binary form. We have tested SwarmVis on Mac OSX 10.5 and openSUSE Linux 11.

Run `qmake`, then run `make`, both in the root directory, to compile SwarmVis. A binary will be created in the `bin/` directory. Execute this binary to run SwarmVis.

B. Data File Format

SwarmVis requires a specific file format for data sets that are to be loaded. The agents' position data is segmented into separate files that each represent a single time step. These files are space-delimited data, with each row representing an agent. For example, a swarm system with 100 agents depicted over 500 time steps will have 500 files in a folder, each with 100 lines.

Each row entry in a file follows a format as well. The first two or three columns (depending on dimensionality) are the position data (X, Y) or (X, Y, Z), respectively. The last column is reserved for the group label, which may be used to pass group membership data to SwarmVis. For example, a well-formed line that conveys a three-dimensional position with group information could be:

```
10.15 5.24 84.85 CornerAgent
```

The listing of agents in each file should be stable. That is, the third line in one file and the third line in another file should represent the same agent.

A plain-text information file containing important meta-data must accompany the frame files in the same directory. The following variables must be defined (i.e., `VARNAME = VALUE`) in this file in order for the data to be loaded appropriately:

- `DIMENSIONS` (`2` or `3`)
- `AGENTS` (the number of agents)
- `FRAMES` (the number frames/time steps)
- `RANGEX` (the maximum X value)
- `RANGEY` (the maximum Y value)
- `RANGEZ` (the maximum Z value)
- `AGENTTYPES` (`1` to track agent types, `0` if not)

At the bottom of this file, the keyword `FILES` must appear, followed by a list of frame files, in temporal order. The number of files listed here must equal the number specified by the `FRAMES` variable. Also, the number of lines in every file must match the number specified by the `AGENTS` variable. Below is a sample info file:

```
DIMENSIONS = 3
AGENTS = 150
FRAMES = 446
RANGEX = 600
RANGEY = 600
RANGEZ = 600
AGENTTYPES = 1

FILES
frame000001.txt
frame000002.txt
...
frame000446.txt
```

To load a data set in the SwarmVis application, navigate to "Load Data" and select the info file.