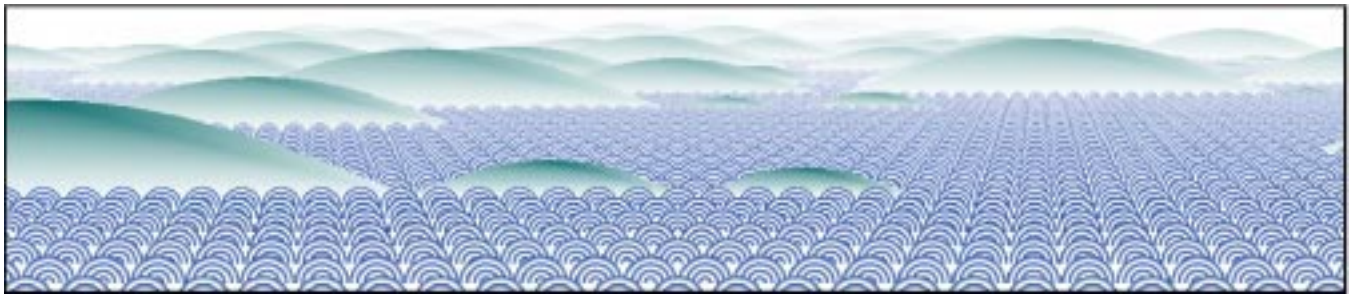
Eastern Perspectives and the Question of Personal Expression

Kaye Mason, Sheelagh Carpendale, Peter MacMurchy, Brian Wyvill
Department of Computer Science
University of Calgary
Calgary, Alberta, Canada
{katherim,sheelagh,peterm,blob}@cpsc.ucalgary.ca



Abstract

Methods in non-photorealistic graphics have tended to place emphasis on algorithms, instead of artists. In contrast, we present a conceptual framework that has been designed to put the control back in the hands of the artist. Combining ideas from non-photorealistic graphics and artificial intelligence, we examine a new method of supporting alternative artistic styles. The details of this model are described using displays based on stylised wave patterns called "seigaiha" in Japanese. In keeping with the Japanese style we provide interactive access to alternate styles of perspective. Finally, we animate our images within this framework, presenting flicker-free non-photorealistic animation.

Keywords: NPR, modelling, self-expression, seigaiha, perspective, graftals, complexity systems, animation, multi-agent systems

1 Introduction

As with their photorealistic (PR) predecessors, early systems for non-photorealistic (NPR) graphics generated expressive effects under algorithmic control. Many NPR approaches have consisted of rendering techniques, which create an image from a simple artist-generated model, or post-production techniques, which manipulate a flat image to create a desired effect. While these techniques are useful for the rapid production of images, they have an undesirable side-effect: much expressive control is taken away from the artist, and assumed by an algorithm. In these systems, the artist is left with only some indirect ability to alter a few parameters. The non-photorealistic effects that are intended to make the image more 'expressive' are under the direction of a deterministic system.

Our research begins with the premise that an artist cannot be entirely replaced with an algorithm. We present an interactive paradigm for graftal-based NPR that endeavours to give creative decisions to the artist. Since algorithmic control is largely dominant in the rendering phase of the graphics pipeline, we shift the creation of NPR effects to the modelling phase, allowing the artist to manipulate the effects directly. Because of the large number of image elements in a scene, we give the artist tools taken from multi-agent and complexity systems with which to direct the model. Our approach, which is based on the ideas used in complexity systems such as Swarm [10] and SELES [5], shifts the balance between algorithmic support and personal expression by combining the rendering and modelling stages of the graphics pipeline. In addition to the potential for personal expression, a significant advantage to our approach is that, since we are modelling these effects directly, inter-frame consistency in animation is no longer an issue.

Section 2 explains motivation, and Section 3 presents the concept behind our approach and the basic structure of our system. In Section 4 model Japanese stylised water patterns or seigaiha. Sections 5, 6 and 7 describe the balance between system support and personal expression during the modelling of seigaiha. This is followed by a discussion of future directions and conclusions.

2 Motivation

Research in computer graphics has come to understand just how important it is to provide support for a broad range of artistic styles. Work has been done in trying to simulate traditional media, such as sketching (see [27, 33, 35]), watercolour (see [4, 9]) and sculpting [22, 23, 24]. Many alternate rendering styles have been developed (see [2, 12, 14, 21]) as well as alternate post production systems (see [9, 25]). A few techniques have looked at non-standard geometries (see [18, 26]).

For many non-photorealistic techniques, the system has an initial set-up period in which the artist does not take part, followed by an interactive phase which the artist controls ([14, 2, 21, 9, 25, 30]).

This interactive phase usually consists of modelling, but could involve image creation by another means. When this is complete, the artist can make choices as to the rendering or post-production algorithm and set various parameters. While there are a growing number of options and algorithms, once these choices are made, the process becomes a 'black box'. If, on completion, the image does not prove satisfactory, the artist has two choices: return to the modelling phase and try again, or revert to pixel-by-pixel touch-up.

The conceptual solution to this problem lies in finding the right balance of direct artistic control, and algorithmic assistance. Removing algorithmic support would render the process untenable. Most non-photorealistic algorithms function by creating image artefacts. These artefacts can be strokes, perturbations, or stipples, and can range from something as small as a single point to something as large as a sketch of a leaf. In most cases, they are distributed over the image in vast numbers. Asking artists to create their image on what could amount to a pixel by pixel basis is equivalent to asking them to work with individual grains of sand. It is impractical to impossible to consider placing them into an image one at a time. This is especially true for creation of animations. While an artist may wish to place or manipulate an individual artefact, it cannot be expected that they should do this with every artefact.

The question of achieving this balance in practice is not as simple. The amount of control an artist might wish to hand off is going to be entirely dependent upon on the individual artist. Some artists might wish to control nearly everything by hand. Others might wish to instruct an algorithm, and have it do the majority of the work. In this latter case, the algorithm should still reflect the desires of the artist as much as possible, and the assumptions of the system architects as little as possible. Ideally a system would:

- create models and images from simple instructions,
- be transparent instead of a 'black box' method, allowing the image creation to be watched,
- be interruptible in that the creation of an image could be stopped and re-directed at any moment, and
- support both explicit and general re-direction.

We attempt to address these issues by modelling with rendering primitives using techniques from complexity and multi-agent systems, and by giving artists controover perspective.

3 Modelling with Rendering Primitives

To create a such a system requires a re-examination and alteration of the graphics pipeline. Within this pipeline, one is creatively involved only during the modelling phase. After the model has been created, one has less input. Artists can choose the rendering algorithm and set parameters, but at this stage the algorithm takes control. If the result is undesirable, they can re-render or post process, but interaction choices do not improve; they can only choose a new algorithm and new parameters. To alter this pipeline, we model with rendering primitives. To create a system that is continuously interruptible, we borrow ideas from complexity systems. To provide algorithmic support, we use ideas from multi-agent systems.

3.1 Borrowing from Complexity Systems

The basic concept of complexity systems is individual components in the system understanding only their own manifestation and behaviour. The overall complexity is emergent. Since the individual elements act without global knowledge, it is possible to model systems and situations that would be dauntingly complex to model explicitly. This concept has been used in ecological modelling [5] and animation [28]. Perhaps the best known example is Swarm [10], a complexity-based simulation environment. Swarm has been used to create simulations with a large number of individuals that interact dynamically within their environment. Individual-based systems can represent many natural complexities, spatial processes, and cumulative effects. Basing the system on individuals makes the system easier to understand and parameterise.

Our system is similar to a complexity system in that it is composed of individual entities that have only local knowledge. They know only how they manifest visually and how they behave in a local immediate sense. That is given their current situation they can deduce what to do next. The system contains is no global knowledge and the resulting complexity of the whole system is emergent. That is the complexity is built up gradually through the actions of many entities. Also, because all actions are local and simple in themselves the resulting system can be interrupted at any time.

Complexity systems are usually used to model a dynamic process such as the hive behaviour of ants or bees, thereby creating a simulation of the process. Since we use a complexity system to create images it might not be immediately thought of as a simulation. However, we are modelling the creative process of making an image, including indecision, interruption and re-direction. As a result we consider the dynamic model of an image during creation as a simulation. The original complexity systems have been 2D systems. Our system operates in a 3D environment. However, we are as yet only making partial use of the 3D space and consider in its current form that our system is operating as 2D+. Further, we have structured our system to be hierarchical. It can contain different types of entities and these entities belong to tribes. In turn, the tribes exist in the environment.

3.2 Using Multi-Agent Ideas

In a multi-agent system, agents are intelligent and have a rule set that governs their behaviour on several levels. Through this rule set they can interact with other agents. For instance, they can cooperate, compete, breed, ignore or even destroy other agents. In the computer graphics arena, Reynolds[28] pioneered animation of flocks and herds using rules based on strictly local knowledge. This is similar to complexity systems in that the agents have largely local knowledge, however, the agent's rule sets can incorporate more global knowledge and/or goals.

We utilize multi-agent ideas. Our entities have rules for interaction that can involve cooperation, competition, avoidance, etc. These rules allow entities to interact in one way with members of their own tribe and other ways with members of other tribes. They can have tribes they like and are attracted to or that they dislike and avoid. The tribes can have tribal interests, rules of behaviour, and goals. These rules also allow us to propagate information up and down the hierarchy. For instance, tribes can be informed of satisfaction at an environmental level and then in turn inform their entities. Further rule sets are used to support communication between the artist and the elements in their creation whether they be entities, tribes or the environment. Since rules can have sets of constraints which they endeavour to satisfy, this degree of satisfaction is also an avenue for artistic control. For further discussion of the use of artistic satisfaction or happiness with the resulting image as a driver for the creative simulation see Mason and Carpendale [19].

3.3 Establishing our system

This system is based on complexity concepts and is an interrupt-driven constraint satisfaction engine. In contrast to static scene-elements in traditional graphics, each scene-element in this system is a dynamic semi-autonomous entity. Individual-based systems can represent many natural complexities, spatial processes,

and cumulative effects. Basing the system on individuals makes it easier to understand and parameterise. This engine differs from other complexity systems in that it operates in a 3D environment and uses a hierarchy. Our rendering primitives are called entities and all entities know how they manifest and how they behave. Rendering primitives that manifest or behave similarly can be grouped together and belong to a tribe.

At the top of any complexity system is a space, or an environment in which the simulation occurs. In a traditional complexity or multi-agent system the environment does very little other than accounting for its various layers or components. This is not the case in our system. We provide a number of environment-level tools that the artist can use to direct the simulation.

The environment contains a grid of hash-buckets based on location. This grid exists to help entities to understand their position in the world quickly. An entity can simply look into its grid cell to see who it is sharing space with, and into neighbouring cells to discover its neighbours. This reduces exhaustive searches to local searches.

In our system the environment is populated by tribes. Each tribe is a group of similar entities, with similar rules for expressing themselves. This conceptual grouping improves artistic control by allowing hierarchical editing, discussed further below.

4 Modelling Seigaiha

Eastern art forms have captured the imagination of many artists and authors in both the West and the East. Batterberry [1] and Fenollosa [6] provide a general background on Eastern Asian art, while Screech [29] and Kobayashi [13] focus on Japanese art. Of particular inspiration to our work is a video exploration of perspective used in Chinese art by Hockney and Haas [11].

We chose to model a stylised artistic approach that can be used very simply and has scope for gradually increasing visual complexity. Stylised 'Japanese' water is such a visual system. The form we look at is referred to as seigaiha and is used in the design of fabric and pottery. Figure 1 shows an example of its use in Imari pottery. Seigaiha, like most Japanese representations of water, is created through repetitive use of a geometric design.

There are many issues to be addressed in achieving even partial visual complexity, as regards the seigaiha tradition. We first describe the representation of seigaiha as entities, tribes and environment in our system. We then present an initial exploration of placement, perspective and animation. In all three we explore the balance between algorithmic support and personal control.

4.1 The Entities

We will start by looking directly at the entities and their role in the simulation. There are three types of entities, water entities, land entities and a plane entity. A basic physical manifestation of the water and land entities is shown in Figure 2. The plane entity does not have a physical manifestation – it exists only as a reference for the other entities, as described below.

Our research draws on the work of Kowalski et al. [14], Markosian [17, 16] and Kaplan et al [12]. These researchers use the idea of *graftals*. The term graftal was coined by Smith [32], who determined that Mandelbrot's fractals [15] could be extended beyond mathematical definitions to definitions by a formal grammar. Although we are not yet drawing intensively on the grammar behind our graftals, we are relying on rule sets to generate and define our entities, and will do so more in the future.

The Plane

The plane includes a scalar field where each scalar represents the type of terrain that exists at that point in the environment. The plane could hold more information in the future: a vector-field, flow vectors, or height values. The plane is not an active entity in the simulation. It doesn't alter itself; it simply responds to queries and directions. Another entity might ask it, "Should there be water or land, at location (x, y, z)?". It might also direct, "Change the value of the terrain at location (x, y, z) to land," if the artist specifies such.

Each water and land entity knows about the plane. When deciding whether or not to draw itself, an entity can query the plane as to what terrain type is supposed to exist in its current location. In addition, when the artist is painting water or land entities, these entities direct the plane to change the terrain type of a region.

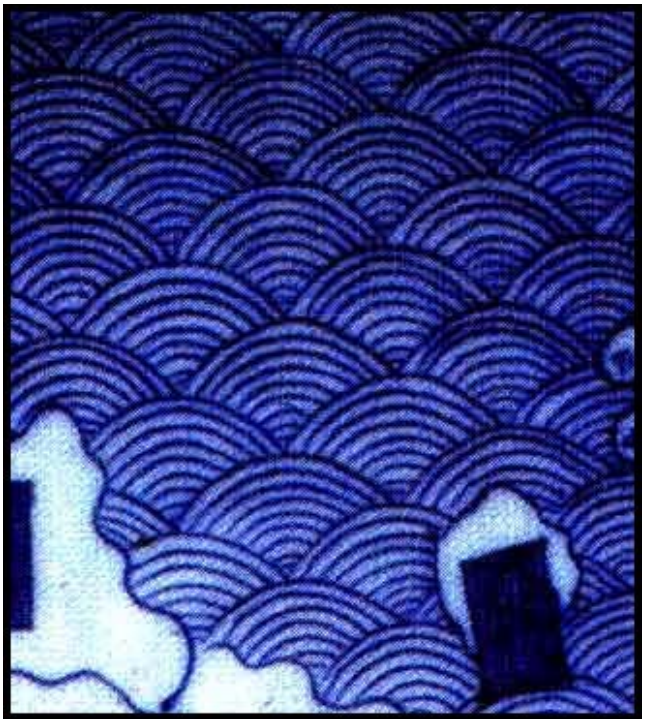


Figure 1: Imari Seigaiha Pattern. Copyright of Zuzu Creations.

Figure 2: The land and water entities.

The Water

The water entities are manifested as four nested hemispheres that have been slightly shaded to suggest watercolours (Figure 2(b)). They are currently spread algorithmically through the entire world-space. Having the water decide when to hide itself was more expensive than simply drawing the water everywhere, and it also led to unacceptable gaps in the landscape.

A water entity may modify its height (within user-defined parameters), or move itself to another location in the scene during the simulation, as directed by the artist. In order to make decisions about how far to move, or how much to scale, it uses information that it gathers from its neighbours, unless it is specifically directed.

The Land

A land entity manifests as a sinusoidal curve. Its height is modified by two factors – a scale factor provided to it by the tribe (and to the tribe by the artist), and a factor relating to its length. Land entities that are longer in length will draw themselves slightly higher.

The length of the land entities is determined by a greedy algorithm operating in a competitive environment. When a land entity is given the opportunity to form itself, it will try to claim as much space as possible from other neighbouring land entities. The first land entity to become dominant in a particular location can turn off all other entities in that location. In this way, the land will continue to advance, until it either finds the edge of the land terrain area, or until it reaches the limit of its own greed. This limit can be set for all entities, or specifically for each entity.

4.2 The Tribes

In the seigaiha simulation there are three tribes: a water tribe, a land tribe, and a plane tribe. The water tribe is responsible for passing directions from the artist to the water entities. Though the entities are largely responsible for their own physical manifestation, the tribe is able to give hints. It coordinates between the entities so that they all animate in the style determined by the artist, either moving downstream, or scaling in their place. It can also give hints to the entities in the form of scaling factors provided by the artist as amplitude and velocity.

Similarly, the land tribe is also responsible for passing on directions from the artist. These directions can include placement, length and height.

There are very few direct controls in the plane tribe. Most of the changes that the artist can make to the entities in this tribe are made through the aegis of the land and water entities. The artist can declare what they would like to paint by selecting an entity, either land or water, as the representation of the terrain type. The entities selected then tell the plane to alter the underlying terrain.

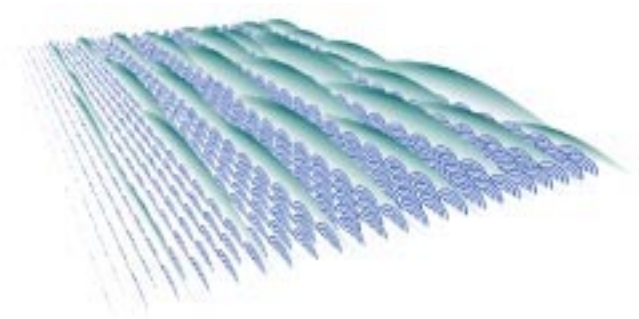


Figure 3: Environment in world space.

4.3 The Environment

Although we use alternate perspective transformations to view the seigaiha pattern, the actual entities in the simulation are laid out in a three-dimensional space, as shown in Figure 3. This gives an advantage in that we can do computations on the basis of (x, y, z), and also that the artist can work in whichever of the spaces is more comfortable, or more appropriate for the particular simulation.

5 Placement

Scene construction in our system is achieved through algorithmically-supported interactive painting. The initial placement of entities is currently done algorithmically: entities are distributed evenly across the space. Such initial placement algorithms are one way that we can offer assistance to artists. It would be laborious for an artist to place all of the entities by hand. Our system will allow this, but we also offer supportive algorithms. If the artist does choose to have an algorithm place entities into the scene, the system still makes it possible for the artist to intervene and move, remove, or alter entities.

The fill algorithm we use for the seigaiha application simply adds alternate, offset rows. The offset in the rows, in conjunction with our perspective transformations, allows us to create the traditional seigaiha pattern.

The land entities use a different algorithm, described previously. In essence, each land entity will reach greedily within its row for all the space it can get, stepping on the nearby entities and shutting off their ability to draw. Once land entities have established their size and position, they remain there, awaiting artist intervention.

The artist can interact with the scene by painting with algorithms. Basically, the artist attaches an algorithmic tool to the brush, and uses the tool in the scene. To show how this works, we construct an example in our seigaiha simulation whereby the artist can change the terrain type by painting with the mouse. The result is the ability to paint land or water over the scene interactively, if the algorithmic placement is unsatisfactory. This allows for artistic determination of the scene composition.

The algorithm for painting works as follows. The artist selects the type of terrain to paint and enters paint mode. The artist then moves the mouse over some entities. Touched entities communicate with the plane entity, ordering a change in terrain type for their local area. If water is being painted, any land entities, which only draw themselves if they are on land terrain, will disappear from the scene. If land is being painted, the land entities that are in the area and not drawing themselves will begin to do so. They will also try to fill any available area, as described above. During painting, the simulation can optionally be running, if the artist so desires.

Hierarchical editing capabilities are built into this system, though they require further development to be fully realised. Hierarchical editing is a simple but powerful concept. It would be an onerous task to edit an image with thousands of entities without some algorithmic support. We thus allow the artist to edit groups of similar entities together. While changes can be made on an entity-by-entity basis where desired, they can also be made to a group of water entities, or to all water entities, over the entire scene.

6 Perspectives on Perspective

While several forms of perspective have been explored in computer graphics, we know of no other graphics model that incorporates eastern ideas on perspective. A thorough exploration of western perspective, in the context of computer graphics, can be found in Carlbom and Paciorek [3]. Zorin and Barr [37] developed a perspective refinement. These ideas were taken up by Wood et al.[36] and furthered into the non-photorealistic realm in an image-based rendering approach to the creation of panoramic views for cel animation.

In our system, we explore eastern ideas on depth representation and take variations of perspective further. As we have stated, one of our primary premises is the concept of artistic control over issues that have been formerly controlled algorithmically. Perspective is one such issue. It is of great significance in the eastern context, where explorations and studies in perspective have been broad and varied [11]. We must therefore allow the artist more flexibility than the simple western notions of one, two, or three point perspective.

Chinese and Japanese artists had developed a rich and distinctive eastern style of perspective before the arrival of western missionaries in the late seventeenth century. The reason for the diver-

gence in eastern and western development is quite apparent from Figure 4, which shows an aerial view of the medium that was typically used in the eastern context. At seventy-two feet in length, this scroll would have provided a series of artistic challenges that were entirely different from challenges commonly faced in the western context. These challenges required the development of different techniques for problem solving in this context.

Figure 4: Southern Inspection Tour – Wang Hui. Photo: Jerry Sohn. Courtesy Program for Art on Film. This 72-foot long canvas is typical in eastern art, but is almost never used in the western context.

The sense of depth that is created in eastern art is intended to give the viewer a sense that the space is expanding in front of them [11]. This is achieved through several means. One of these is a clever use of space and occlusion. Objects are placed either separately, allowing the viewer to imagine the connection between each object, or so as to use occlusion to make the relative position in depth explicit. Occlusion is used in both eastern and western paintings; however, from our observations and the discussions we have read, there seems to be several distinctively eastern notions for conveying the impression of depth.

Figure 5 shows just how complicated these perspective techniques can get. Each house in this leaf fragment is drawn in perspective, and yet the lines of perspective are not the same for any two of the houses. By this technique, the artist is able to open up the canvas and reveal much more to the viewer, showing parts of the scene that would be hidden from a traditional western perspective. Perspective projections that have been used in computer graphics come from the western school of thought, and are largely inadequate to express the eastern tradition. In this section we will briefly discuss traditional approaches for conveying depth information, before considering the ways in which we allow artists to use this information in our system.

6.1 Traditional Graphics Perspective

For comparison purposes, Figure 6 shows our seigaiha model with water entities only, rendered with a traditional computer graphics perspective, as described in [3]. The matrix used for this transformation is:

$$M_{per} = \begin{bmatrix} x_s & 0 & 0 & 0 \\ 0 & y_s & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1/z_s & 1 \end{bmatrix}$$

In this matrix, x_s, y_s and z_s represent a scale on the values of a point, in proportion to the viewing box, and the aspect ratio of the final image.

Figure 5: Seasons – Gong Xian. At the University of Durham, Oriental Museum.

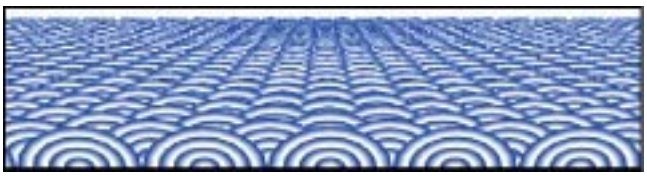


Figure 6: A traditional perspective projection, as by M_{per}.

The projection produced by M_{per} is not particularly useful for our seigaiha model, for the same reason it is not the style of choice used in the eastern context. It is difficult to fill a canvas with landscape, as is often done in the eastern context, with this projection. Attempts to do this could be made by having landforms that are taller in the background. This does not work well with flat areas like oceans, farms, or plains, all of which occur frequently in the context of early modern Chinese and Japanese landscapes.

Another method would involve moving the eye point close to the ground. This is undesirable in that the front landforms eclipse everything else. We are left only with tilting the viewplane, as shown in Figure 7 and diagrammed in Figure 8a. Where the entities near the bottom are almost orthogonal to the view vector. The reason for this is clear from Figure 8a: the projection near the bottom of the view plane is nearly orthogonal to the ground plane. The image is drawn without land to make the projection more clear.

While this latter orthogonality problem could be corrected by billboarding all of the entities in the scene, it is conceptually cleaner to deal with this by altering our view of perspective projection to be understood as in Figure 8b, and as described below. This conceptual shift has practical applications – it allows us to give the user dynamic control over the projection, as will be discussed at the end of this section.

6.2 Farther Up in the Picture Plane

In general, a canvas does not have a depth field – it exists as a two dimensional plane. This applies equally to a raster display, and to a painting surface made of bamboo, cloth or cotton. We can visualise a three dimensional representation, model it, and even do calculations based on it. But in the end, if we are working with a two dimensional representation, we must somehow arrange to have a two dimensional image.

One way to do this is to translate distance information into height on the painting surface. To transfer a point from world space into canvas space with this method, we simply take a proportion of the

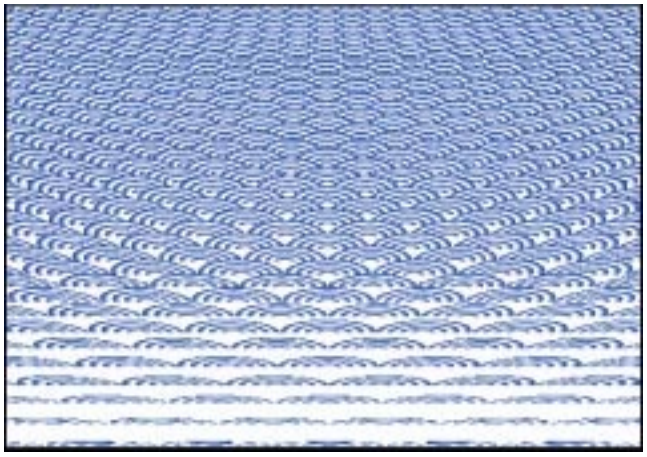


Figure 7: Attempting to fill the view window with a traditional perspective projection, by tilting the viewing angle. The thin entities at the bottom occur when the viewing angle is orthogonal to the ground plane.

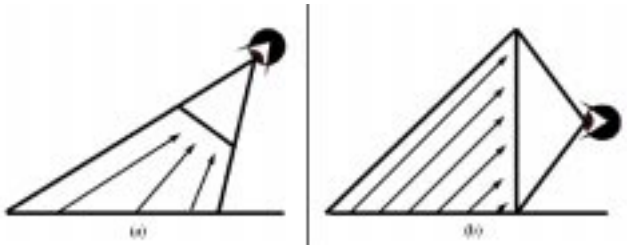


Figure 8: The difference between a traditional CG perspective projection, and our perspective projection.

point's height, and add to it a proportion of its depth. Proportions are used to take into account the height, width and aspect ratio of the final canvas, but they also allow us to give artistic control over the projection.

We accomplish projection in our system by taking the traditional perspective matrix out of the graphics pipeline, and replacing it with the following.

$$M_{sta} = \begin{bmatrix} x_s & 0 & 0 & 0 \\ 0 & y_s & F_s z_s & 0 \\ 0 & 0 & z_p & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

As with the matrix M_{per} above, x_s, y_s, and z_s represent scaling in the x, y and z components of the point to make it conform to the viewbox and image aspect ratio. The z_p component preserves a small amount of z to allow depth buffering. The F_s factor now represents stacking, or the proportion of the z component that is added to the y component before the image is drawn to the canvas.

By allowing artists to control the F_s factor, we give them control of how much of the depth is added to the height value to determine the local position in canvas space. The artist can choose a large amount of depth, no depth, or even a negative depth, which results in objects that are further away being lower down on the canvas.

Figure 9 shows an example of our seigaiha simulation with no land, drawn with this projection. The entities at the top of the canvas are farther away in the z-plane. Those at the bottom are closer to the camera. Figure 16, in our results section, shows another example of this projection with land added. With the addition of occlusion, as described above, we feel the depth of the various objects is quite readily determined, as it is in the traditional scroll paintings.

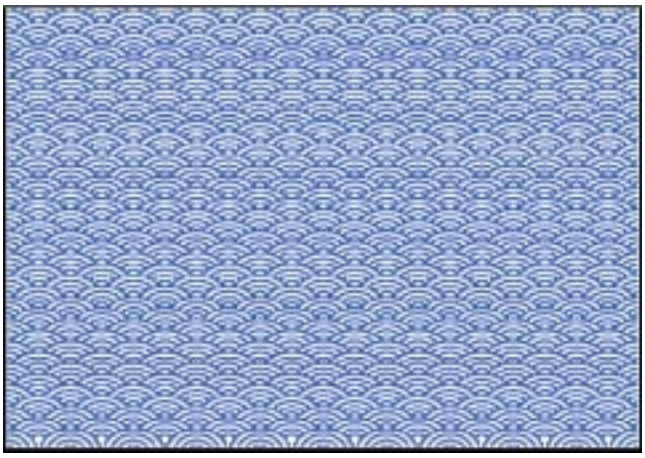


Figure 9: Perspective Projection created by the matrix M_{sta}. Entities that are further away are drawn higher up on the view plane, creating an effect similar to that achieved in Chinese and Japanese handscrolls.

6.3 Recession with Distance

Some eastern artists did use the technique of scaling distant elements to invoke a sort of recession with distance, though largely in cooperation with the stacking approach. We have provided for this in our system, by using the projection matrix M_{rec}. The variables x_s, y_s, z_s, z_p, and F_s retain the same meaning as above. This matrix has an additional recession factor F_r, which is multiplied by the scaled z value, and then placed to affect the homogeneous coordinate. This factor then divides all point values in canvas space, thus resulting in a scaling down of object sizes with distance.

$$M_{rec} = \begin{bmatrix} x_s & 0 & 0 & 0 \\ 0 & y_s & F_s z_s & 0 \\ 0 & 0 & z_p & 0 \\ 0 & 0 & F_r z_s & 1 \end{bmatrix}$$

As with the stacking, we allow the artist to determine how much recession may or may not be desirable in any particular case. This allows for a range of effects to be created, and within this range, the artist can adjust the perspective projection to whatever is appropriate to the work of art in question. Figure 10 shows us an example of this recession used in combination with the stacking, in our seigaiha model. Again, we remove the land, so that one can more clearly follow the lines of the water entities, and understand just what is happening.

At first glance, this approach seems more familiar to those of us who are used to traditional graphics approaches to perspective. And in a way, it is. By incorporating our approaches to perspective as a continuous system, we can achieve a traditional CG perspective. We do this by lowering the stacking constant F_s in M_{rec} to zero, and by ensuring that the recession constant F_r is in the range from $(0, 1)$. Ranges of F_s and F_r outside of these values, open up the possibilities for differing perspectives.

The reason for the difference between Figure 7 and Figure 10 is illustrated Figure 8. In the traditional CG version of the perspective projection in 8(a), the view vector is nearly orthogonal to the ground plane in the lower part of the view plane. By contrast, in our projection, illustrated in 8b, this is not the case. The lines of projection are parallel. Every image element is projected in the same

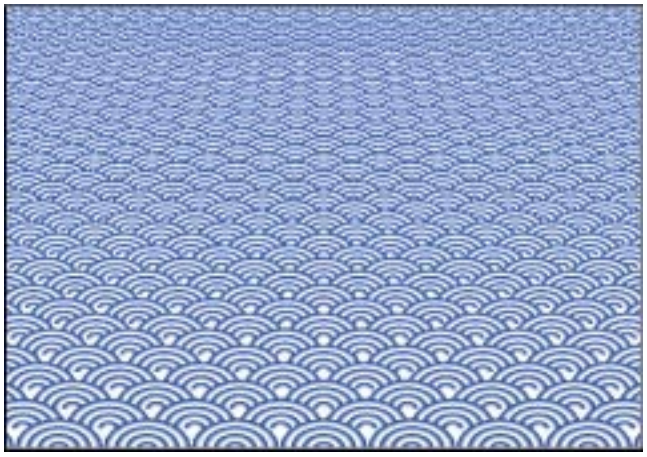


Figure 10: Perspective Projection created by the matrix M_{rec}. Entities that are further are drawn higher up on the view plane, and smaller. Later handscrolls (post-western missionary influence) follow this pattern.

manner, and is viewed from its 'front', though overall impression is of an aerial view.

6.4 Diagonal Perspective

Many Chinese and Japanese paintings incorporate a type of diagonal perspective. There are several resulting visual effects. There is often a fairly apparent criss-crossing diagonal grid to the layout. This diagonal grid has the effect of creating what could be termed several vanishing points. These vanishing points are nested in that they recur between each pair. This creates the effect of space opening out in front of the viewer.

Figure 11 shows a fragment of a wider handscroll painting by Japanese artist Hishikawa Moronobu that illustrates this diagonal perspective well. The viewer's position with respect to the right side of the image is on the right side, and the viewer's position with respect to the left side is on the left.

We accomplish a smooth transition between perspectives providing a combination of the effects just described. When the recession constant F_r in M_{rec} is in the range $[1, 2)$, the projection gives us a type of diagonal perspective as seen in Figure 12. Notice the self-similarity of the vanishing points, which recur.

Figure 13 shows this perspective with the stacking constant F_s at a higher value, so that the stacking effect is spread over the canvas. This gives us a close approximation to the sort of perspective that we see in the eastern scrolls, as in Figure 11.

6.5 Expansion in the distance

A final perspective approach that we have observed in eastern art involves objects that are farther away actually increasing in size. In this case, we set the constant F_r in M_{rec} to be in the range $[-1, -2)$. While this effect can seem confusing to an eye trained in the western perspective, it has been explored quite extensively in the eastern context, particularly in the seventeenth century.

6.6 Dynamic Perspective Control

One of the greatest achievements of our work with perspective has been the ability to fluidly blend these different approaches to perspectives in a way that gives the user greater control over the projection. The matrix M_{rec} shows the stacking constant F_s, and the recession constant F_r. By replacing the standard projection matrix with the matrix shown, and by giving control of these factors directly to the artist, we are giving control over the projection to the artist.

Figure 11: Scenes in a Theatre Tea-House (fragment)– Hishikawa Moronobu. Copyright The British Museum, used with permission. The lower view shows the lines of perspective drawn over the picture. The picture contains multiple viewpoints – one off to the right, and one off to the left.

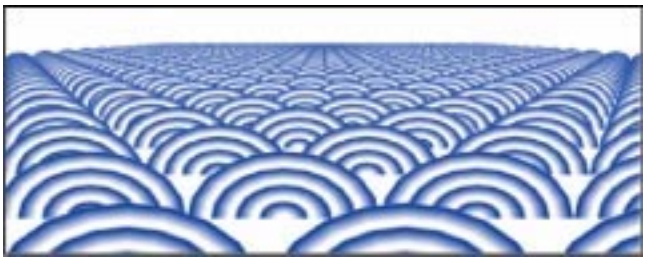


Figure 12: Diagonal Perspective. There are multiple vanishing points, and a criss crossing pattern.

While these projections do not necessarily represent the only views of perspective possible, and while they do not yet address the fact that eastern artists often use different perspective approaches in different parts of the same work, they do begin to address the issue of perspective in an eastern context, and the application of artistic control over its parameters.

7 Toward Animation

A principle advantage of our approach is inter-frame consistent animation. Since the rendering primitives exist in the actual model, they maintain frame to frame coherency. The potential exists within this framework for the properties of each entity to be adjusted by the artist, to whatever level of detail is desired, either on a frame-by-frame basis, or in the overall environment.

Our initial attempts to introduce animation into the seigaiha model involved moving the entities to show flow, as down a river.

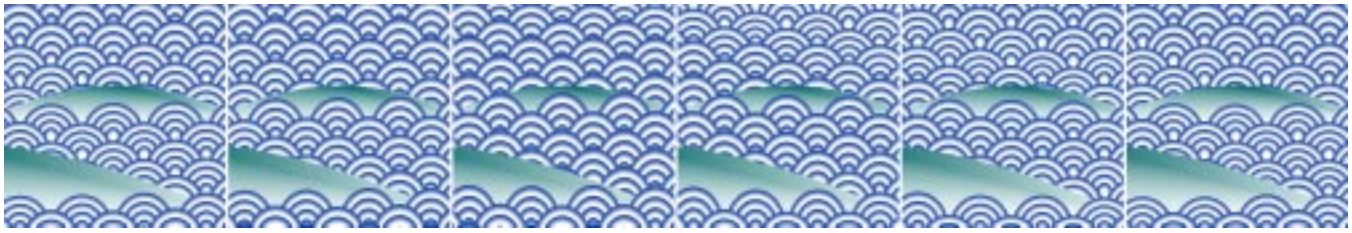


Figure 15: Animation

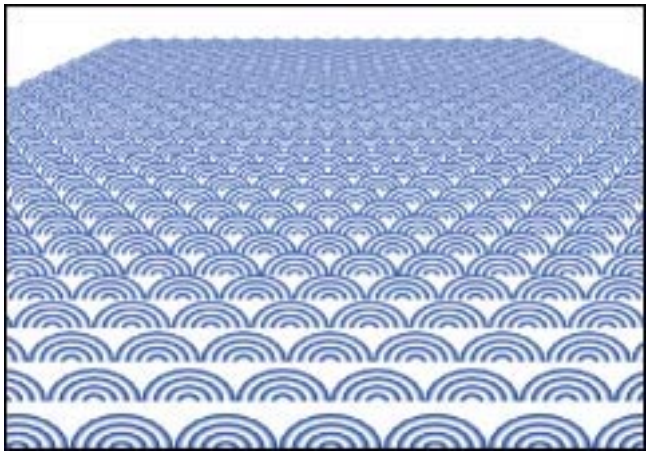


Figure 13: Diagonal Perspective Stack. Similar to Figure 12, but with the stack constant F_s at a higher value.

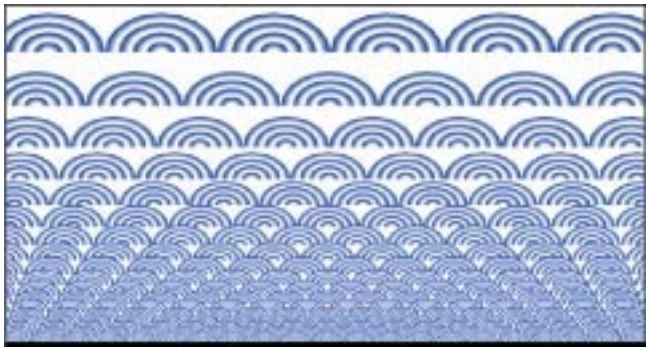


Figure 14: Dish Perspective, showing expansion into the canvas. Entities that are farther are actually drawn larger.

This is physically what is happening with water flowing down a river, and it has been done successfully in particle system simulations of water [31, 34]. However, with the non-photorealistic elements, it did not create the effect we had been hoping for.

It is useful to note this, because when we are are working with image elements that do not resemble reality, and within a perspective projection that is not physically based, it is necessary to approach problem solving from a different angle. Our final animation came from a suggestion that the entities should do the wave, as though they were in a hockey rink – each one standing up in its own turn, then passing the wave on to the next entity.

With this in mind, we turned to research in modelling and animating of water. Modelling and visualisation of fluid flow and water have been objects of research for many years in the fields of both computer graphics and physics. As Alain Fournier points out in his famous paper on generating waves for computer graphics[7], there is a large body of work on waves. Unlike Fournier, we use a Eulerian method for tracking points, as it is better suited to our local control paradigm. In particular, we refer the reader to Matthews[20], who describes the wave equation as a hyperbolic partial differential equation. We use central differencing to solve both a one dimensional formulation, with waves travelling horizontally or vertically, and the following two-dimensional formulation:

$$\begin{aligned} x_i &= x_{i-1} + (v_{i-1} \cdot \delta t) \\ v_i &= v_{i-1} + m \cdot (4x_{i-1} + x_{t_{i-1}} + x_{b_{i-1}} + x_{l_{i-1}} + x_{r_{i-1}})\delta t \end{aligned}$$

In these equations x represents the entity's position, m its mass, and v its velocity. The current time step is reprsented by i, and the increment by δt. The constructs x_* represent thte postion for the entity's neighbours, where x is the neighbour at the top, x_b at the botom, x_l at the left x_r at the right.

In this way, each entity computes its position (interpreted as a scaling factor in y) at each time step by looking at its previous position and velocity. The current velocity is in turn computed based on the entity's previous position and velocity, and the position and velocity of its immediate neighbours. This method lends itself very well to a complexity system, where each entity is making the decision about what size to draw itself. The results are remarkably effective. Figure 15 shows a few frames from an animation using this system. If fortune favours, you will be able to trace the wave as it moves across the water. The land is included in this sequence to give a static point for the eye to compare between frames.

To allow for greater user control and to encourage stability of the system, the velocity is then capped to a user-defined limit, which has the effect of controlling the amplitude of the wave function. The absolute value of the x_i is used, with a fixed offset, so that the water entities never scale to, or below, zero.

In addition, particle system approaches to the visualisation of water are particularly relevant to this research. Most notable is research by Sims[31], Goss[8], van Wijk[34].

8 Future work

Through use of techniques from artificial intelligence, we have begun to build a framework to support artistic direction in non-photorealistic graphics. In this framework, the addition of non-photorealistic effects takes place in the modelling phase of the graphics pipeline, and is under the control of the artist with algorithmic support.

This research opens up new areas for consideration, such as the development of a formal grammar for modelling of individual entities, to expand on the notion of entities as graftals. The development of a visual language for the specification of the rules for the constraint-calculus goes hand in hand with this. The largest missing piece from this is the creation of a visual language for the building of tribes and entities. Much of this is still done in code, which

Figure 16: West Coast Millennial Angst.

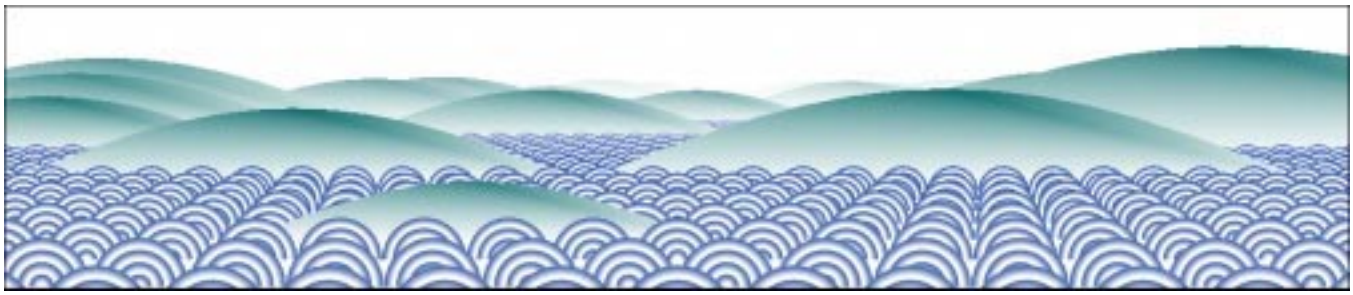


Figure 17: Where's Mount Fuji?

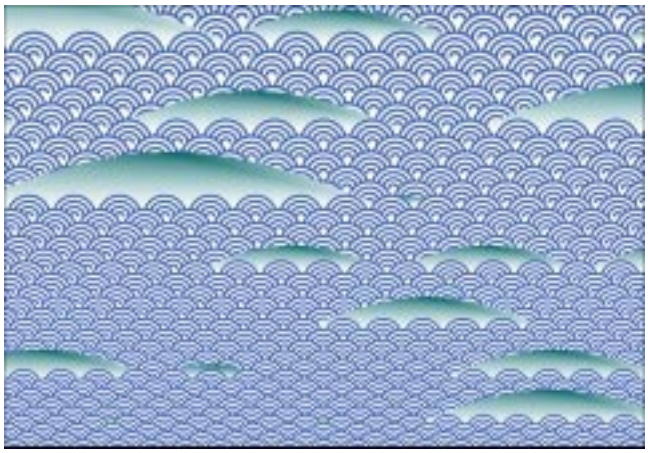


Figure 18: Under the waves at Cowtown. Demonstrating dish perspective.

should not be the case. Requiring artists to be programmers in order to use the system is highly restrictive. In addition to this, more tools are necessary to allow the artist to manipulate large numbers of entities at once, and further development is necessary on the hierarchical editing that is present in a nascent form.

More thought is also warranted in the area of non-photorealistic animation. It is clear that techniques that are not physically based require further exploration. Further, a naive approach to keyframing is far too cumbersome. The consideration of other approaches might uncover patterns in successful techniques.

9 Conclusions

We have used complexity and multi-agent systems to support artist-control in non-photorealistic graphics. Using the seigaiha example from Japanese tradition, we show how the use of a complexity system, where each entity is an intelligent, semi-autonomous agent, can be used to give algorithmic support for non-photorealistic methods with minimal loss of artist control.

On the journey to this goal, we have encountered and explored a number of other issues with respect to non-photorealistic graphics and artistic control. The first of these relates to perspective. Approaches that have been traditionally taken in computer graphics with regard to perspective are inadequate to the task of drawing in a traditional Japanese style. The second issue is in regards to animation, where again, traditional approaches had to be exchanged for less conventional means.

Acknowledgements

The research is supported in part by NSERC research grants. We would like to thank Mark Matthews his help with physics equations, and state our appreciation to all the graphics lab at the University of Calgary for their ideas, encouragement and support. And Red Carpet, for the free coffee and hot chocolate!

References

[1] Michael Batterberry. *Chinese and Oriental Art*. McGraw-Hill, 1968.

[2] David J. Bremer and John F. Hughes. Rapid approximate silhouette rendering of implicit surfaces. In *Implicit Surfaces Proceedings*, 1998.

[3] I. Carlbom and J. Paciorek. Planar geometric projections and viewing transformations. *Computing Surveys*, 1978.

[4] Cassidy J. Curtis, Sean E. Anderson, Joshua E. Seims, Kurt. W. Fleisher, and David H. Salesin. Computer-generated watercolor. In *SIGGRAPH '97*. SIGGRAPH, ACM, 1997.

[5] J. Fall and A. Fall. Seles: A spatially explicit landscape event simulator. In *Proceedings of the GIS and Environmental Modeling Conference, Santa Fe, New Mexico, Jan, 1996*, 1996.

[6] Ernest Francisco Fenollosa. *Epochs of Chinese and Japanese art : an outline history of East Asiatic design*. sl, 1913.

[7] Alain Fournier and William T. Reeves. A simple model of ocean waves. In *SIGGRAPH '86*. ACM, 1986.

[8] Michael E. Goss. A real time particle system for display of ship wakes. *IEEE Computer Graphics and Applications*, 1990.

[9] Aaron Herzman. Painterly rendering with curved brush strokes of multiple sizes. In *SIGGRAPH '98*. ACM, 1998.

[10] David Hiebeler. The swarm simulation system and individual-based modeling. In *Decision Support 2001*, 2001.

[11] David Hockney and Philip Haas. A day on the grand canal with the emperor of china: or surface is illusion but so is depth. Video, 1988.

[12] Matthew Kaplan, Bruce Gooch, and Elaine Cohen. Interactive artistic rendering. In *Non-PhotoRealistic Rendering and Animation 2000*. NPAR, 2000.

[13] Todashi Kobayashi. *Ukiyo-e*. Kodansha International, 1982.

[14] Michael A. Kowalski, Lee Markosian, J.D. Northrup, Ludomir Bourdev, Ronen Barzel, Loring S. Holden, and John F. Hughes. Art-based rendering of fur, grass, and trees. In *SIGGRAPH '99*. SIGGRAPH, ACM, 1999.

[15] Benoit Mandelbrot. *The Fractal Geometry of Nature*. W.H. Freeman and Compnay, 1983.

[16] Lee Markosian. *Art-Based Modeling and Rendering for Computer Graphics*. PhD thesis, Brown University, May 2000.

[17] Lee Markosian, Barbara J. Meier, Michael A. Kowalski, Loring S. Holden, J.D. Northrup, and John F. Hughes. Art-based rendering with continuous levels of detail. In *Non-PhotoRealistic Rendering and Animation 2000*. NPAR, ACM, 2000.

[18] D. Martin, S. Garcia, and J.C. Torres. Observer dependent deformations in illustration. In *Non-PhotoRealistic Rendering and Animation 2000*. ACM, 2000.

[19] Kaye Mason and Sheelagh Carpendale. Artist-driven expressive graphics. In *Eurographics 2001 Short Presentations*. EG, 2001.

[20] John H. Matthews. *Numerical Methods for Mathematics, Science and Engineering*. Prentice Hall, 1992.

[21] Barbara J. Meier. Painterly rendering for animation. In *SIGGRAPH '96*. ACM, 1996.

[22] Shinji Mizuno, Minoru Okada, and Jun ichiro Toriwaki. Virtual sculpting and virtual woodcut printing. *The Visual Computer*, 14, 1998.

[23] Robert Noble. *Intuitive Sculpting of Flexible Objects for Coherent Animation*. PhD thesis, Robert Gordon University, 1998.

[24] Robert Noble and Gordon J. Clapworthy. Improving interactivity within a virtual sculpting environment. In *Information Visualization '98*. Robert Gordon University, IEEE Press, 1998.

[25] Victor Ostromoukhov and Roger D. Hersch. Multi-color and artistic dithering. In *SIGGRAPH '99*. ACM, 1999.

[26] Paul Rademacher. View-dependent geometry. In *Proceedings of the 26th annual conference on Computer graphics*. ACM, 1999.

[27] Ramesh Raskar and Michael Cohen. Image precision silhouette edges. In *Symposium on Interactive 3D Graphics*, April 1999.

[28] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *SIGGRAPH '87*, 1987.

[29] Timon Screech. *The Western Scientific Gaze and Popular imagery in later Edo Japan : the lens within the heart*. Cambridge University Press, 1996.

[30] Michio Shiraishi and Yasushi Yamaguchi. An algorithm for automatic painterly rendering based on local source image approximation. In *Non-PhotoRealistic Rendering and Animation 2000*. NPAR, ACM, 2000.

[31] Karl Sims. Particle animation and rendering using data parallel computation. In *SIGGRAPH '90*, 1990.

[32] Alvy Ray Smith. Plants, fractals, and formal languages. In *SIGGRAPH '84*. ACM, 1984.

[33] Mario Sousa and John W. Buchanan. Observational model of blenders and erasers in computer-generated pencil rendering. In *Graphics Interface*, 1999.

[34] Jarke J. van Wijk. Flow visualization with surface particles. In *IEEE Computer Graphics and Applications*, July 1993.

[35] Lance Williams. Ink jets, level sets and silhouettes. In *Workshop on Image-Based Modeling and Rendering*, March 1998.

[36] Daniel N. Wood, Adam Finkelstein, John F. Hughes, and Craig E. Thayer. Multiperspective panoramas for cel animation. In *SIGGRAPH '97*. ACM, 1997.

[37] Denis Zorin and Alan Barr. Correction of geometric perceptual distortions in pictures. In *SIGGRAPH '95*. SIGGRAPH, ACM, 1995.