CONTROLLING TEXTURE MAPPING ONTO IMPLICIT SURFACES WITH PARTICLE SYSTEMS

Ruben Zonenschein[1] *Jonas Gomes*[1] *Luiz Velho*[1] *Luiz Henrique de Figueiredo*[2]

[1]Instituto de Matemática Pura e Aplicada – IMPA
Estrada Dona Castorina 110, 22460-320 Rio de Janeiro, RJ, Brazil
ruben | jonas | lvelho@visgraf.impa.br

[2]Laboratório Nacional de Computação Científica – LNCC
Avenida Getúlio Vargas 333, 25651-070 Petrópolis, RJ, Brazil
lhf@lncc.br

ABSTRACT

We discuss the control possibilities of texture mapping for a new method to apply 2D textures onto implicit surfaces. The method uses a particle system associated with the gradient vector field of an implicit function to acquire texture coordinates at a support surface. It introduces geometric, parametric and temporal concepts to solve the problem of texture mapping, elements from which we create tools for local and global control.

1. INTRODUCTION

Texture mapping is a well established technique in computer graphics. It associates a texture source to a surface, improving surface detail without changing the geometry of the model.

A texture source, known as a *texture map*, may be described by an image, a look-up table or a mathematical function. The basic idea of texture mapping is to associate a texture space to the surface space, where the texture will be applied. This method is closely tied to surfaces described parametrically, such as patches, since the association of two parametric spaces is straightforward.

While new ideas and uses of this technique lends to powerful tools for increasing the visual complexity of parametrically defined models, little has been done to apply 2D textures onto implicit surfaces. The reason is very simple: implicit surfaces do not have a natural coordinate system defined on them, and the usual parametric association of surface points and texture coordinates cannot take place. Therefore, it seems natural to apply textures onto implicit surfaces using solid textures [1, 2, 3]: a 3D texture space is embedded in object space and texture coordinates values are defined with no additional cost at any point on the surface. Although 3D texture mapping is a powerful technique, it is limited to materials that have an intrinsic 3D structure, such as wood and marble, but not appropriate for materials that would wrap a surface, such as images. Moreover, 3D texture mapping leaves little room for control, since there is a fixed correspondence between points on the texture and points on the surface.

In fact, to effectively take advantage of the texture mapping technique, it is important to be able to control how the texture is mapped onto the surface. In traditional parametric texture mapping, a user may be given tools to place a texture map at any desired position and orientation on a surface, scale the map, tile it along the surface, stretch it etc. It is desirable to have similar tools for applying 2D textures onto implicit surfaces.

We have recently introduced a new method to apply 2D textures onto implicit surfaces [4, 5]. By generating a force field from the gradient vector field of an implicit function, we simulate a particle system that maps the implicitly defined model to a support surface for the texture. In this paper, we discuss the possibilities of controlling texture application with this method.

2. PREVIOUS WORK

The problem of applying textures onto implicit surfaces and the possibility of controlling texture mapping has been addressed before.

Much attention has been given to the possibilities of parameterization of implicit surfaces, either globally or in parts [6, 7, 8]. It is reasonable to follow this approach, since applying textures onto parametric surfaces is well understood. Unfortunately, there is no general conversion method to parameterize implicit surfaces.

The use of solid textures on implicit surfaces was first discussed by Wyvill et al. [3]. Their work shows clearly that 3D textures are strongly tied to an implicit object, with little possibility of control. Because of its 3D nature, solid textures are closer related to the volume of an implicit solid than to its surface.

Wyvill et al. were particularly interested in blobby objects whose shape changes in time, and how a 3D texture space should be set in order to follow a geometry change. By setting a separate 3D texture space for each key point, in the skeleton of a blobby object, their method performs a weighted sum of coordinates at the texture space level to assign texture values to points on the surface. Although they make an effort to keep continuity of the texture while a object changes shape, the texture distortion may not follow the geometric change, thereby creating texture artifacts.

One alternative for applying 2D textures onto implicit surfaces would be via a projection [9]. As in slide projection, a 2D texture source can be projected onto a surface, by computing the intersection of rays that leave the texture in the direction of the surface.

The projection approach presents a critical problem: the rays as a whole will never reach all points of a closed surface from the outside, making impossible to wrap the texture around the surface. Even if only part of the surface is to be textured, a ray may intersect it more than once, giving the same texture attributes to different points, which is not desirable (see Figure 1).

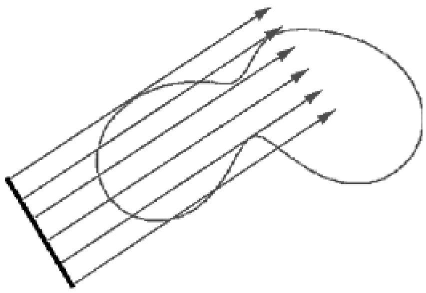

Figure 1: Rays intersecting the surface more than once.

From the ideas presented above, it became clear that we should search for some inherent characteristic of implicit surfaces to overcome the 2D texturing problem. We have found that the gradient vector field of the function that describes an implicit surface has a great potential for solving the problem. We do not abandon the nature of implicit surfaces, as the 2D projection technique does; and we are not restricted to viewing implicit surfaces as volumes, as in solid texturing. Moreover, the overall setup of our method has the advantage of creating a rich environment for controlling texture placement.

3. 2D TEXTURING IMPLICIT SURFACES WITH PARTICLE SYSTEMS

3.1. Overview

An implicit surface is defined as the set of points x in space that satisfy an equation $F(x) = c$, where $F\colon \mathbf{R}^3 \to \mathbf{R}$ and $c \in \mathbf{R}$. Thus, F defines a continuous family of *level surfaces*, one for each *isovalue* c. The level surfaces of F follow the gradient vector field ∇F orthogonally, as can be seen in the Figure 2. This observation is the starting point for a method for sampling implicit surfaces [10] and for the texturing method described in this section.

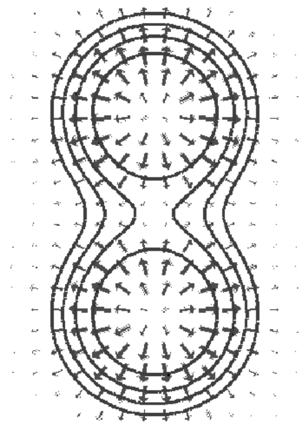

Figure 2: Gradient field follows level surfaces.

We interpret ∇F as a force field, which we use to drive a particle system. Particles initially at rest on an implicit surface move along the ambient space until they reach a support surface, where a texture is defined.

The use of a dynamic system to perform the correspondence between points on two different spaces — the surface and the texture — opens up various degrees of control. In fact, besides the mapping function, the method introduces geometric and temporal concepts to solve the problem, elements of which we can take advantage to control how the texture is applied onto the surface. In this paper, we explore this flexibility, discussing the possibilities of control of 2D texture mapping onto implicit surfaces.

3.2. Mathematical model

Let S be an implicit surface defined by a function F. We employ the gradient vector field ∇F to generate a force field defined in the ambient space, and we use it to guide particles that are initially at rest on S. The motion of a particle is governed by the differential equation

$$\frac{d^2x}{dt^2} + \gamma\frac{dx}{dt} + \nabla F = 0,$$

where x is the position of the particle, t is time and γ is a viscosity constant. This particle system establishes a correspondence between points on the implicit surface and points on a support surface T, where the texture is defined: the texture attribute for each point on the surface is “read” at the intersection of the corresponding particle trajectory with T. The support surface T should be simple enough so that it is very easy to define texture coordinates on it (e.g. cylinder, sphere, etc.). For this reason, we use support surfaces that have both a parametric and an implicit description. The parametric description allows us to define a texture on T. An implicit description helps us detecting whether a particle is inside, outside or on T. As we shall see, the availability of these two descriptions is useful when designing tools for texture mapping control.

3.3. Constructing the force field

The basic element of the force field is the gradient vector field of the implicit function: alone, it is enough to guide particles leaving the surface and reaching the support surface for the texture. Figure 3 illustrates the results of the algorithm in 3D.

Nevertheless, the method described makes it possible to introduce other elements to take part in the force field. These elements give us more freedom of manipulation over the force field, yielding different tools for texture mapping control. We distinguish them depending on their behavior over the force field, either global or local.

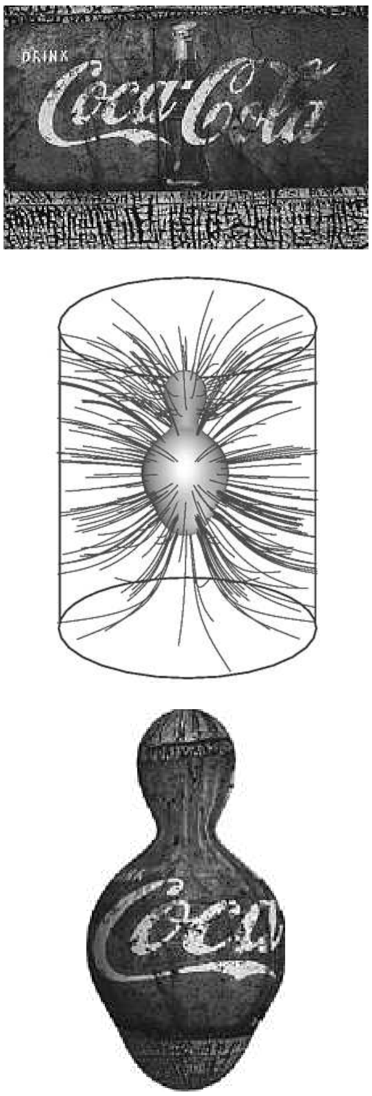

Figure 3: texture source (top), particles trajectories (middle), and textured blobby object (bottom).

Global control: The method described associates two geometric objects through a force field. It is generated from the gradient vector field which reflects the shape and neighborhood properties of the implicit surface. This force field is then used to induce particles away from the implicit surface, adapting them to the support surface. It would be intuitive if, besides this force, another force could attract them to the support surface T. As described above, T has also an implicit description, whose gradient vector field, as well, reflects the shape and neighborhood properties of the support surface. We use a modified gradient vector field ∇G of the support surface, by reversing its sense to behave as an attraction force to T [10]. The global force field is then computed by a linear combination of the two gradient vector fields, giving the following motion equation:

$$\frac{d^2x}{dt^2} + \gamma\frac{dx}{dt} + \alpha\nabla F - (1-\alpha)\nabla G = 0,$$

where $\alpha \in [0,1]$ is a user-controllable relative weight. Figure 4 shows how a linear combination may adapt the particles better onto the support surface.

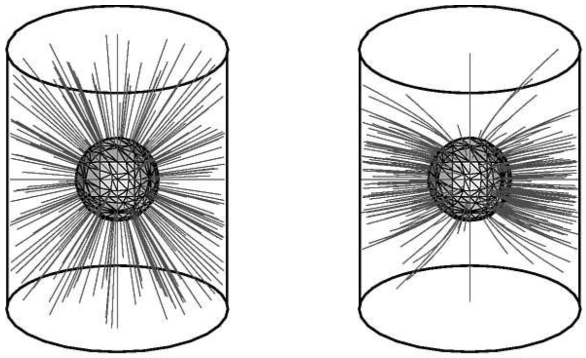

Figure 4: Particle trajectories based only on ∇F (left) and a linear combination of ∇F and ∇G (right).

Local control: The laws that govern a dynamic system can be very strict. In our case, its behavior depends on a very reliable source, the gradient ∇F. Nevertheless, an external force introduced in a dynamic system can modify its characteristics im many ways. In our method, an external force may be useful to interfere on a particular region of the particles trajectories. It will provide local control over the texture placement. Figure 5 shows an external force acting on a region of the force field.

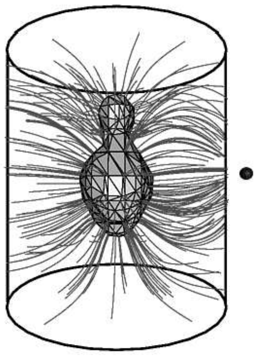

Figure 5: An external force attracting some particles.

3.4. Implementation

We use a polygonal approximation of the implicit surface as the input of our system. This simplicial representation is used on most graphics pipelines. Specially in our case, a polygonized model with texture coordinates at its vertices can be rapidly textured using graphics hardware.

We start the simulation with particles placed at the vertices of a polygon mesh that approximates the implicit surface. We use a simple adaptive polygonization algorithm to construct the mesh [11]. Each particle follows a trajectory along the force field; we integrate their motion equations numerically with Euler's method. At each step in the simulation, we test whether a particle has crossed the support surface T, using its implicit definition. If it has, then we compute the intersection point of the trajectory with T. A texture coordinate is then taken at this point, using the parametric definition of T. When the simulation has finished for all particles, we use linear interpolation of texture coordinates on each polygon, to complete the texturing of the implicit surface.

The timestep that governs each particle's step along the vector field plays a major role in the accuracy of the simulation. This discretization of the particle's position in the medium defines how fast the particles should travel. The faster it moves, the less accurate is its flow across the level sets of the function F. On the other hand, if the timestep is too small, then the simulation may be unnecessarily slow. Better methods employ an adapted timestep. We are currently working on an adaptive integration based on Runge-Kutta's method.

4. CONTROL

The method described in the previous section could be classified as a two-step texture mapping technique. Each step represents a different level where control tools may be created. Figure 6 illustrates this view of the method.

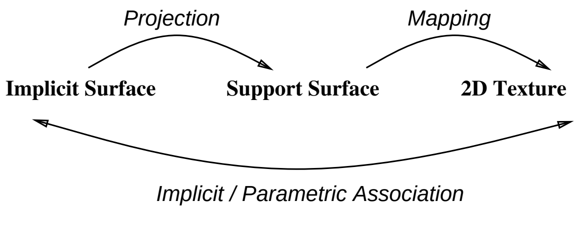

Figure 6: The method and its steps.

In summary, the method comprehends:

- *Projection step*: The first step deals with the particle system simulation that associates an implicitly defined surface to a support surface for the texture. The goal is to find the best adaptation of the implicit surface to the support surface via the construction of a force field.
- *Mapping step*: The second step refers to the mapping of a 2D texture source onto a parametric description of the support surface.

In combination, these two steps produce the final association of points in an implicit surface and points in a 2D texture source.

We take advantage of the characteristics of each step to create three different types of control tools:

- *Geometric*: geometric transformations applied to the objects and external forces.
- *Parametric*: varying attributes of the processes.
- *Temporal*: using the time component of the dynamic system.

The scope of each control tool will be associated to the global and local classification described in Section 3. Our goal is to create a variety of intuitive and effective tools in order to fill the diagram in Figure 7. In fact, it is important to have this variety of control tools. A user may want to change the texture mapping in a global and local manner, and with this system we are able to provide different types of tools to achieve such results.

	Geometric	**Parametric**	**Temporal**
Global			
Local			

Figure 7: Classification of texture control aspects.

In Sections 5 and 6 we analyze how we may control the final result creating tools and manipulating parameters at each method step. In Section 7 we overview how the method should be used in an interactive user interface.

5. CONTROL IN THE PROJECTION STEP

The projection step retains the basic elements of the method described in Section 3. At it, an implicit to support surface association is established. It is of no surprise that we found in this step a rich environment for controlling this association, including the creation of the force field and the simulation itself.

The possibilities of texture mapping control inherent to the projection step can be analyzed through the technique's overall setup: there is one coordinate system related to the support object for the texture; another coordinate system for the ambient space, where the objects are embedded; and a force field that associates these two coordinate systems. We are able to create tools for manipulating the elements that participate in this setup.

We create geometric tools taking advantage of the existence of the two coordinate systems, applying transformations onto the objects embedded in them. The parametric tools reflect the changes that can be made to the functions that participate in the force field. The temporal tools take advantage of the time component existing in dynamic systems.

5.1. Geometric tools

The relative positions of S and T and of any other external force source play a major role over the global placement of the texture. Any transformation applied to them will modify the overall force field. Figure 8 exemplifies.

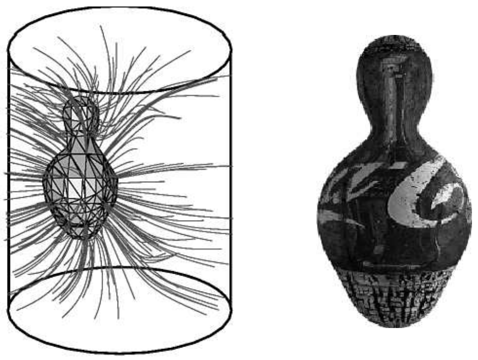

Figure 8: A transformation on T affecting the force field and its effects on texturing S.

The geometric position of an external force source creates variations over some particle trajectories. In Figure 9 we see how a repulsor may affect the texture in a specific part of the surface.

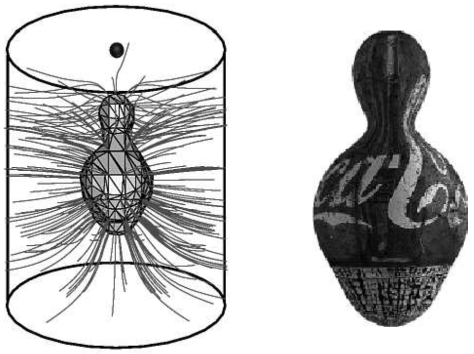

Figure 9: Placing a repulsor to squeeze the texture on the top part of the bottle.

It is important to notice that different support surfaces will lend to different results of texture mapping. In fact, the user should choose T depending on its general geometric similarity with S. In Figure 10, a sphere is chosen as the best support surface for spheric-like object.

Once a good support surface is chosen, the user can apply affine transformations and investigate the changes in the textured implicit surface. Figure 11 shows how scaling T may globally affect the texture on S.

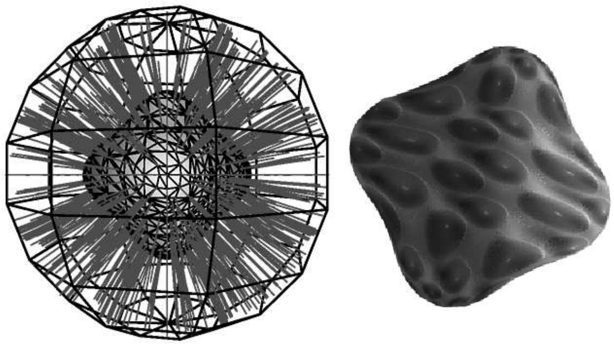

Figure 10: A spheric-like object textured using a sphere as support surface.

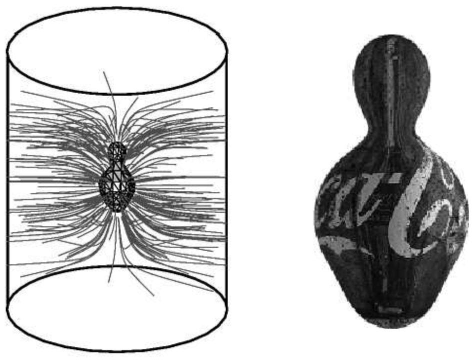

Figure 11: Particles taking a longer trajectory to reach a scaled T and the result on texturing S.

5.2. Parametric tools

The elements of the projection step that lend to parametric tools are the functions that generate the force field. By changing their parameters we can achieve global and local control.

As explained above, the force field may be globally controlled with a blend of the gradient vector field of S and T. Instead of a linear blend, a weighted blend could be used to enhance the proprieties of shape and neighborhood of a surface while a particle is closer to it. In the next section we use this parametric control in time to globally change the force field (see Figure 13).

Local control can be parametrically refined by varying any of the parameters, such as the area of influence. In Figure 12 we refine the squeezing effect introduced in Figure 9 by changing the attractor's magnitude. We resize the attractor on this illustration to better compare its effect over the force field.

5.3. Temporal tools

An important characteristic of a dynamic system is its time component. In our method, this feature can be depicted in the lifetime of particles: they are born on the implicit surface, move along a force field while they are alive, and die when they reach a support surface. We may introduce a time parameter in the motion equation that governs the particles along the medium. The α parameter, now as a

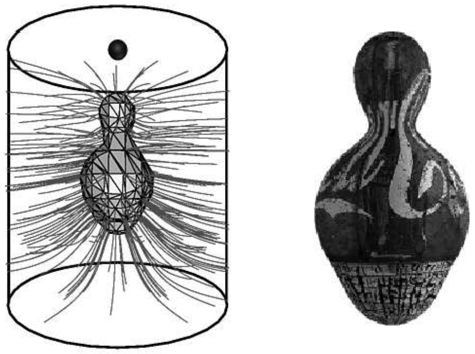

Figure 12: Increasing the squeezing effect on the top part of the bottle.

function of time, will yield the following motion equation:

$$\frac{d^2x}{dt^2} + \gamma\frac{dx}{dt} + \alpha(t)\nabla F - (1-\alpha(t))\nabla G = 0.$$

A parametric tool may be combined with temporal tools. Figure 13 shows a weighted blend of ∇F and ∇G as a function of time. The closer a particle is to an object, the higher is the contribution of the object's implicit function to the force field.

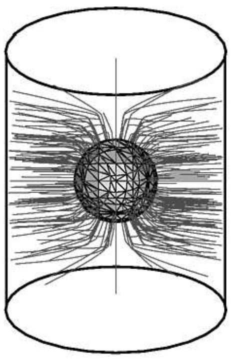

Figure 13: A weighted blend of ∇F and ∇G.

In fact, all tools introduced in the previous sections can be generalized to be used while the simulation is running. One can experiment with the results of applying linear transformations on the objects and external forces while the particles are moving. Figure 14 shows the result of rotating the support surface while the simulation takes place.

6. CONTROL IN THE MAPPING STEP

The mapping step establishes an association of a 2D texture and the support surface. This amounts to generating texture coordinates in a parametric domain, which is usually the identity transformation. Nevertheless, we shall see a wide range of handles for controlling the mapping of the texture onto the support surface.

As described, our method assumes that a simple support surface is used, which has both parametric and im-

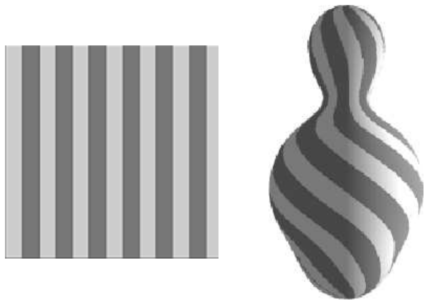

Figure 14: The texture source and a twisted mapping.

plicit descriptions. For the mapping step, this is important to be able to map the texture source onto the support surface using the usual parametric texture mapping technique. The tools available to this technique to control how the texture is mapped will increase the control over our method.

A global geometric tool may be used when applying transformations at the image source changing its position and size at the support surface. Figure 15 shows an image label being applied onto different parts of a cylinder.

Figure 15: The effect on S by a change of the image on T.

A parametric control can be manipulated changing the tiling parameter of a replicated texture (see Figure 16).

Figure 16: Tiling an image onto S.

Once run, the method solves the problem of associating implicit surface points to 2D texture points. As it is, we have already a good degree of texture mapping control when setting up the method and while the simulation is run. However, the resulting association is also very good information, from which we are able to create tools to control and adjust the texture mapping. Figure 17 shows the simplicial approximation of an implicit surface mapped onto a 2D texture source.

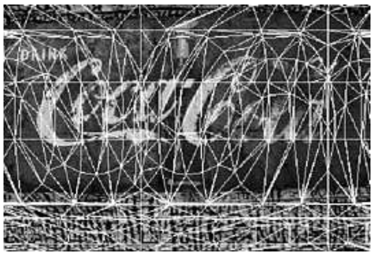

Figure 17: A simplicial approximation of an implicit surface mapped onto a 2D texture source.

Selecting one or more surface vertices, we are allowed to move their mapped counterparts at the texture source, warping it to locally change the texture mapping. This is important when trying to match a part of an image with a particular region of the surface. Figures 18, 19 and 20 exemplify such tool. An initially shaded object and a texture source are introduced in the particle system method (Figure 18). As the simulation is over, the object's simplicial approximation is mapped onto the texture domain. The texture source is then translated and warped in order to match the desired features (Figure 19). By re-rendering the scene, an exact association of object and texture features is achieved (Figure 20).

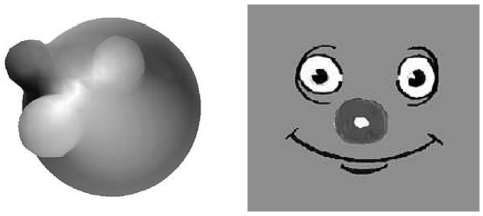

Figure 18: The object shaded and the texture source.

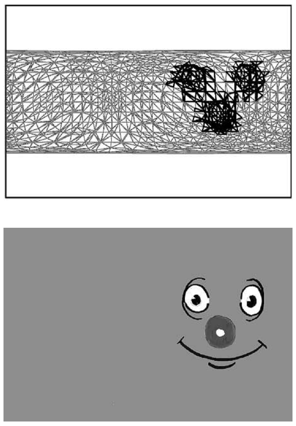

Figure 19: The object simplicial approximation mapped onto the texture and the texture source translated and warped to match the features.

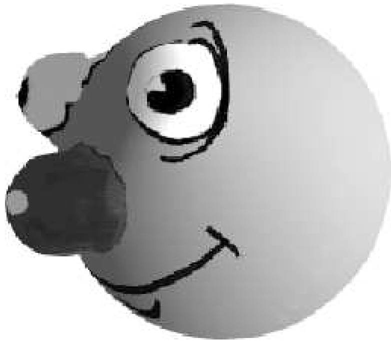

Figure 20: The resulting textured object.

7. OVERVIEW AND USER INTERFACE

We are currently working on implementing an fully interactive system that allows one to texture map an implicit surface with the particle system method in real time. Our goal is to be able to follow the steps listed below in a recursively and repeatedly way.

- Input an implicit surface file and select a support surface.
- Apply affine transformations onto the surfaces.
- Run and visualize the simulation.
- Visualize the textured implicit surface.
- Establish the best projection using the global tools.
- Refine particular areas using local tools.

To fully take advantage of the particle system method and its possibilities of texture mapping control, a specific user interface should be designed. Tools for visualizing the particle simulation, changing function parameters and picking mapped particles at the texture source, to name but a few, shows how diverse those tools are. For this reason, the implementation and the user interface we are designing exploit the particularities of the *Projection* and *Mapping* steps to create an integrated system. Also, in this interactive system, feedback is a very important issue. Therefore, we rely on hardware with texture-dedicated engines, such as Open GL. The system includes four canvas:

Simulation canvas: The *Projection* step must have a 3D interface, which is available through this canvas. On it, the implicit and support surfaces, the external forces and the particle system are manipulated and visualized. The canvas includes three orthogonal projections and a perspective projection. Once the implicit and support surfaces are selected, the *Simulation* canvas allows the user to apply affine transformations to the objects, visualize them, and run and visualize the simulation. Toggle buttons allows the user to hide or view each component in this canvas. This is important when the user wants to focus on a particular point. Also, different viewing options are available for each object: polygonized, shaded, and textured.

Parameter canvas: All text input, such as the implicit surface and the texture source files, are interfaced through the *Parameter* canvas. On it, sliders, buttons and menus allows the user to change the control parameters. Also, the temporal concept discussed in this paper can be used in this canvas. The particle system simulation may be "turned on" so that it becomes a non interrupted process. In this case, all geometric and parametric manipulation may happen while the system is "on". This will lend to different situations where an animated texture can be created.

Texture canvas: The *Mapping* step is essentially 2D. Its main interface is through a side by side canvas. While in one side is placed the texture source, the other one has two coordinate systems embedded in it:

- The first one is the parametric domain of the support surface. It allows one to place the 2D texture at different positions and scale at the support surface.

- The second one is the projection of the simplicial approximation of the implicit surface onto the support surface. This is suitable to visualize and manipulate the result of the projection mapping in the vertex or face level. Associating vertices, faces, and regions of the implicit surfaces mapped on the support surface and their counterpart at the texture source allows one to have a fine control of where things are.

The visualization of these tools is integrated with the *Simulation* canvas, where picked vertices, faces and regions are highlighted and where the texture placement at the support surface is visualized. Also, the warping tools described in Section 6 is interfaced with this canvas.

Render canvas: Integrating the interface, this canvas permits the user to visualize the textured implicit surface using a high quality rendering software. Using a camera paradigm, this canvas allows one to zoom in and out and rotate the camera and the textured implicit surface.

8. CONCLUSION

We have described a method to associate a 2D texture source to an implicitly defined surface. An extensive set of tools for controlling the texture placement has been presented. The different classifications of these tools, depending on scope of action and on intrinsic elements such as geometric, parametric and temporal, allowed us to sketch an integrated user interface.

We are currently implementing such a system, integrating with objects defined in multiresolution representation, allowing it to run in real time.

A generalization of this method to apply textures onto each individual primitive of a composite deformable implicit object is presented in [12].

9. REFERENCES

[1] D. Peachey, Solid texturing of complex surfaces. *Proceedings of SIGGRAPH'85*, pp. 279–286.

[2] K. Perlin, An image synthesizer. *Proceedings of SIGGRAPH'85*, pp. 287–294.

[3] G. Wyvill, B. Wyvill, C. McPheeters, Solid texturing of soft objects. *IEEE Computer Graphics & Applications*, 7(12), 1987, pp. 20–26.

[4] R. Zonenschein, J. Gomes, L. Velho, L. H. de Figueiredo, Textura de superfícies implícitas com sistemas de partículas. *Proceedings of SIBGRAPI'95 (Brazilian Symposium on Computer Graphics and Image Processing)*, pp. 305–306.

[5] R. Zonenschein, J. Gomes, L. Velho, L. H. de Figueiredo, Texturing implicit surfaces with particle systems. *SIGGRAPH'97 Visual Proceedings*, p. 172.

[6] H. K. Pedersen, Decorating implicit surfaces. *Proceedings of SIGGRAPH'95*, pp. 291–300.

[7] H. K. Pedersen, A framework for interactive texturing operations on curved surfaces. *Proceedings of SIGGRAPH'96*, pp. 295–302.

[8] J.-P. Smets-Solanes, Vector field based texture mapping of animated implicit objects. *Computer Graphics Forum*, 15(3), 1996, pp. C-289–300. (Proceedings of Eurographics'96).

[9] A. Barr, Decals. In *State-of-the-Art of Image Synthesis*, Course Notes, SIGGRAPH'83.

[10] L. H. de Figueiredo, J. Gomes, Sampling implicit surfaces with physically-based particle systems. *Computer & Graphics*, 20(3), 1996, pp. 365–375.

[11] L. Velho, Simple and efficient polygonization of implicit surfaces. *Journal of Graphical Tools*, 1(2), 1996, pp. 5–24.

[12] R. Zonenschein, J. Gomes, L. Velho, L. H. de Figueiredo, M. Tigges, B. Wyvill, Texturing composite deformable implicit objects. *Submitted to SIBGRAPI'98*. http://www.visgraf.impa.br/Projects/dtexture