A Few Good Samples: Shape & Tone Depiction for Hermite RBF Implicits

Emilio Vital Brazil[2, 1] Ives Macêdo[2, 1] Mario Costa Sousa[1] Luiz Velho[2] Luiz Henrique de Figueiredo[2]

[1]University of Calgary, Canada [2]IMPA – Instituto Nacional de Matemática Pura e Aplicada, Brazil

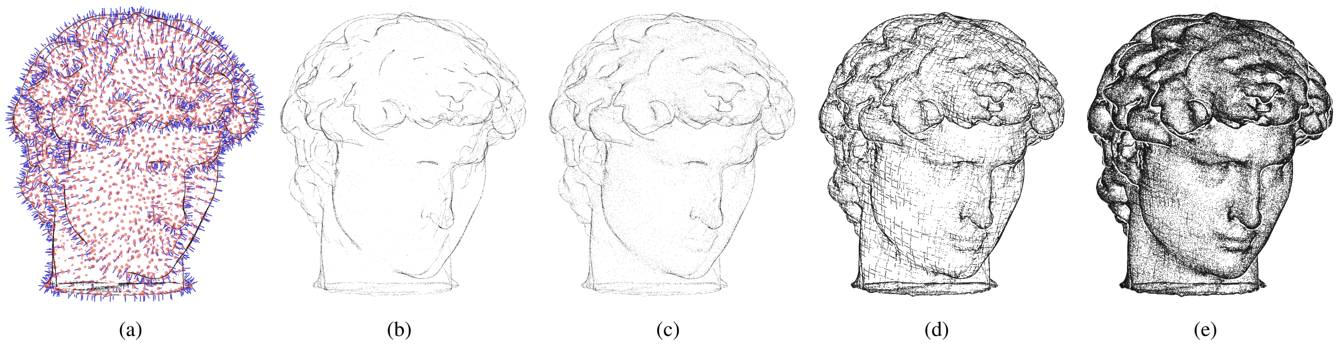


Figure 1: *Drawing steps of David's head with our system. (a) Given a set of point and normal samples, a Hermite Radial Basis Function is reconstructed and directly rendered, depicting shape and tone in different styles: (b) silhouettes with hidden-line attenuation; (c) adding a small number of stippling marks; (d) increasing number of stippling marks and adding curvature-based cross hatching, and (e) completing the tone depiction by adding more stippling marks and enhancing interior contours with a white halo. The model has 4096 samples, 700K render points and with CPU rendering at 16 fps.*

Abstract

We present techniques for rendering Hermite Radial Basis Function (HRBF) Implicits in different pen-and-ink styles. HRBF Implicits is a simple and compact representation, providing three fundamental qualities: a small number of point-normal samples as input for surface reconstruction, good projection of points near the surface, and smoothness of the gradient field. Our approach uses these qualities of HRBF implicits to generate a robust distribution of points to position the drawing primitives. The resulting implicit model is then rendered using point-based primitives to depict shape and tone using silhouettes with hidden-line attenuation, drawing directions, and stippling. We present sample renderings obtained for a variety of models.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms;

Keywords: Non-photorealistic rendering (NPR), variational implicit surfaces, Hermite RBF implicits, point-based NPR, computer-generated stippling.

1 Introduction

Pen & ink illustrations, whether with traditional or computer-generated techniques, provide a number of perceptual cues such as relationships between light and dark, shape, pattern and edge depiction, drawing direction, focus, and gradients of detail and texture. Three key elements are essential for effectively conveying these perceptual cues: *where* to place drawing primitives, *how many* to place, and *how to draw* them [Andrews 2006; Lohan 1978; Smith 1992; Deussen 2009; Kim et al. 2009]. In this paper, we present methods approximating traditional ink-based rendering techniques for depicting shape and tone perceptual cues, suitable for non-photorealistic rendering (NPR) applications using implicit surfaces as the primary object representation (Figure 1).

Implicit surfaces provide important, mathematically precise information about surface properties, useful for answering *where* and *how many* primitives to draw across the surface. Implicit surfaces allow global calculations such as point pertinence (i.e. whether a point is within the surface volume) and distance evaluation, and at the same time, also allow obtaining local differential properties, such as curvature. This brings advantages over other types of geometric models. We use the recently introduced Hermite Radial Basis Function (HRBF) Implicits which interpolate point-normals to reconstruct an implicit surface [Macêdo et al. 2009]. HRBF Implicits is a simple and compact representation, requiring only a few number of point-normal samples to reconstruct quality implicit surfaces. In addition, the good behavior of HRBF Implicits allows performing all the general implicit surface operations using simpler and more efficient algorithms, even for complex models.

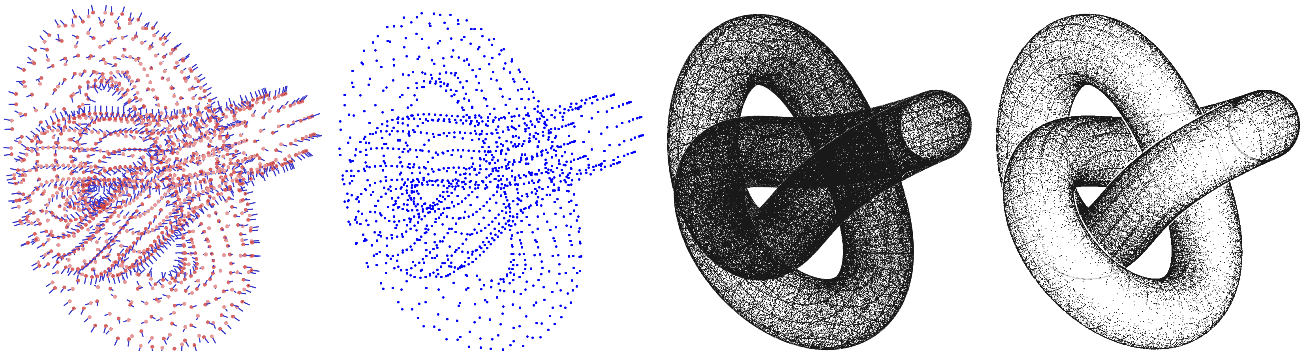
The main contribution of this paper is on applying NPR techniques directly over HRBF Implicit models, bringing important benefits such as consistent and good projection of points, placement and distribution of drawing primitives (for shape and tone depiction), and real time interaction with the rendered model.

2 Related Work

Different works have proposed NPR techniques for implicit surfaces, addressing the problem of extracting contours (silhouettes, feature curves) and approximating different traditional rendering styles including pen & ink stylized rendering, hatching and stip-

[1]email: smcosta@ucalgary.ca

[2]email: { emilio | ijamj | lvelho | lhf } @impa.br



(a) Surface samples (b) Seed points (c) All render points (d) Final rendering

Figure 2: *Overview of our system pipeline. (a) Given surface samples (points and normals), (b) seed points are placed across the surface, (c) allowing further placement of render points; (d) these render points are then used to modulate different tone and shape depictions.*

pling [Bremer and Hughes 1998; Elber 1998; Foster et al. 2005; Jepp et al. 2006; Jepp et al. 2008; Schmidt et al. 2007], feature line extraction and drawing [Ricci 1973; Rosten and Drummond 2003; Burns et al. 2005; Plantinga and Vegter 2006; Stroila et al. 2008; Proença et al. 2007; Proença et al. 2008], painterly rendering [Akleman 1998], tone-based clip art [Stroila et al. 2008], and mixed media [Jepp et al. 2009].

Bremer and Hughes [1998] presented an approach to extract and trace silhouettes incrementally from analytic implicit functions. Short interior ink-based strokes are also positioned using successive ray intersection tests, including hidden-line removal (HLR). Foster et al. [2005] extended these tracing and particle-based techniques by providing additional options for stroke stylization and specific interior stroke placement strategies on complex hierarchical implicit models. Techniques for rendering sudden blends and CSG junctions are also presented. Jepp et al. [2006, 2008] have further extended this NPR framework using flocking techniques to manage particle distribution and render additional surface contours in different pen & ink stippling and curvature-based hatching. Proença et al. [2007, 2008] also extended the approach presented by Foster et al. [2005], by extracting and rendering suggestive contours over point-set MPU implicits. One particular strategy is to project particles from a base mesh onto the implicit surface and model the strokes using this particle distribution. This approach was used by Elber [1998] who also presented several methods for ink-based stroke rendering effects. More recently, Schmidt et al. [2007] adapted an approach where low-resolution silhouette and suggestive contours are extracted from a coarse base mesh approximating the smooth surface and incrementally refined and projected to the implicit surface. Stippling and HLR are also provided by adapting surfel techniques.

In our approach, we use a new representation, Hermite Radial Basis Function (HRBF) implicits. Our point distribution does not require relaxation techniques, given that HRBF provides a good projection framework. All our rendering primitives (silhouette contours, stippling, hatching) are points. Rendering is performed directly over the implicit model without requiring any intermediate representation. We provide a Hidden Line Attenuation (HLA) method, which approximates some of the visual elements of an artist-generated visual construction, or *scaffolding* [Schmidt et al. 2007].

3 System Overview

Initially, a HRBF is fitted to a given set of surface samples (points and normals). The level 0 of this function is an implicit surface which interpolates the samples (Figure 2(a)). Next, we need to place seeds over the generated surface (Figure 2(b)). If the samples are already well distributed, they may be used as seeds. Otherwise, seeds are created within the bounding box of the samples, regularly spaced, jittered and projected on the implicit surface. Then, the refinement phase starts, which consists of increasing three sets of points (Figure 2(c)). The first type is point stippling, which does not follow a specific direction and achieves a well spread coverage. The second type follows the principal directions of curvature, being useful to depict volume or represent guiding lines (Figure 7(d)). The third type, as the former, creates a line drawing effect, now following a discontinuous spherical combing pattern (a tiled direction field) (Figure 7(b, c)), proper for depicting object masses. Finally, given a fixed camera position, we classify the generated points as either front, back, or silhouette. We use this classification to define how the points will be rendered (Figure 2(d)).

4 Implicit Representation

In order to place our point-based primitives over a surface, we rely on a few basic geometric operators, namely, projection of a point onto a surface and the computation of its normals, curvatures and principal directions. By employing a suitable representation, these operations can be made fast and implemented using simple approximate algorithms, yet still giving very good results.

Our representation of choice is based on implicitly-defined surfaces computed from points and normals using the HRBF Implicits method of [Macêdo et al. 2009]. This representation has many desirable properties which allow us to employ off-the-shelf linear algebra packages together with very simple iterative algorithms to both compute the implicit function and implement our basic geometric operators robustly enough.

In the following, we briefly review HRBF Implicits and our scheme for placing points onto a surface.

4.1 Hermite RBFs

Recently introduced in [Macêdo et al. 2009], HRBF Implicits provide a powerful tool to reconstruct implicitly-defined surfaces from points and normals. They present many desirable properties of which we take advantage in designing our method. At first, the reconstructed surface is guaranteed to interpolate the given points; in addition, the unit normal at those points equals the gradient of the function without the need of creating artificial offset samples.

Since the gradient of the implicit function has unit norm at the samples, the function does not vary too wildly close to the surface, a property useful for simple iterative projection algorithms. Also, the implicit function is guaranteed to be at least $\mathcal{C}^1$ at the sample points and, by properly choosing the RBF, $\mathcal{C}^\infty$ everywhere else, hence the reconstructed surface is typically $\mathcal{C}^1$ at the samples and $\mathcal{C}^\infty$ otherwise. This is a useful property in estimating the local curvatures and principal directions at a given point on the surface. Experiments indicate that their Hermite interpolation property allows good behavior of both the reconstructed surface and the implicit function even under nonuniform and coarse samplings, thus filling holes and capturing local geometric details 'hidden' in the normals.

Since the main focus of this work is on rendering, we use a HRBF Implicits fitter as a black-box for which the points $\{\mathbf{x}^j\}_{j=1}^N \subset \mathbb{R}^3$ and normals $\{\mathbf{n}^j\}_{j=1}^N \subset \mathbb{S}^2$ samples are the input and a function $f : \mathbb{R}^3 \to \mathbb{R}$, implicitly defining a surface by $\mathcal{S} = f^{-1}(0)$ with the properties mentioned above, is the output. For the sake of completeness, we briefly review the form of a HRBF Implicits interpolant and how to fit its coefficients from given points and normals.

The HRBF Implicit Interpolant

Macedo et al. [2009] introduced HRBF Implicits as an interpolatory method for recovering implicitly-defined surfaces from points and normals. By making use of a theoretical framework for generalized interpolation using radial basis functions, a concrete expression for the implicit function $f : \mathbb{R}^3 \to \mathbb{R}$ was derived as follows:

$$f(\mathbf{x}) = \sum_{j=1}^{N} \left\{ \alpha_j \psi(\mathbf{x} - \mathbf{x}^j) - \langle \boldsymbol{\beta}^j, \nabla\psi(\mathbf{x} - \mathbf{x}^j) \rangle \right\} + p(\mathbf{x}) \quad (1)$$

where $\alpha_j \in \mathbb{R}$, $\boldsymbol{\beta}^j \in \mathbb{R}^3$, $p : \mathbb{R}^3 \to \mathbb{R}$ is a trivariate polynomial and the scalar field $\psi : \mathbb{R}^3 \to \mathbb{R}$ is defined by a radial basis function $\phi : \mathbb{R}_+ \to \mathbb{R}$ as $\psi(\mathbf{x}) := \phi(\|\mathbf{x}\|)$. Although the original paper does not contain the polynomial term, this augmentation is possible by introducing appropriate side-constraints as we explain later.

Tthe authors [Macêdo et al. 2009] provide sufficient conditions and examples of suitable choices for ϕ attending the assumptions made in their theoretical considerations. Most notably, the Gaussians $\phi_\sigma(r) := \exp(-\frac{r^2}{2\sigma^2})$ and suitable Wendland's compactly supported functions [Wendland 1995], of which $\phi_\rho(r) := (1 - \frac{r}{\rho})_+^4(4\frac{r}{\rho} + 1)$ is the one they employed. It was shown that, by enforcing the interpolation conditions $f(\mathbf{x}^j) = 0$ and $\nabla f(\mathbf{x}^j) = \mathbf{n}^j$ at each sample point, the coefficients in the expression above (without the polynomial term) are uniquely determined and can be recovered by solving the induced symmetric positive definite linear system.

In order to introduce augmenting polynomial terms, which is useful when employing compactly-supported RBFs, we need to fix a basis $p_1, \ldots, p_M : \mathbb{R}^3 \to \mathbb{R}$ for these trivariate polynomials, where $M = \binom{d+3}{3}$ and d is their degree; in addition, we need to be sure the only polynomial with at most that degree whose value and gradient are zero at all sample points is the constant zero; we also need the additional side-constraints on the coefficients α_j and $\boldsymbol{\beta}^j$,

$$\sum_{j=1}^{N} \left\{ \alpha_j p_k(\mathbf{x}^j) + \langle \boldsymbol{\beta}^j, \nabla p_k(\mathbf{x}^j) \rangle \right\} = 0, \quad \forall k = 1, \ldots, M. \quad (2)$$

Together with the interpolation conditions, these constraints result in a symmetric (indefinite) linear system with $4N + M$ variables which is guaranteed to have a unique solution for every (pairwise-different) sample points and any prescribed normals.

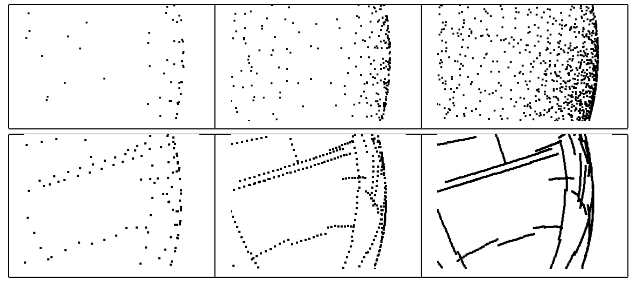


Figure 3: *Multi-Level Sample Refinement. left to right, levels of refinement: one, two, and three.*

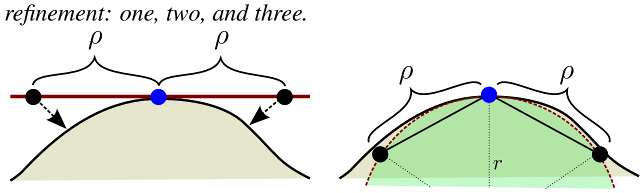


Figure 4: *Placement of render points (black) near the surface using the given seed (blue). Left: using the tangent plane. Right: using the osculating circle.*

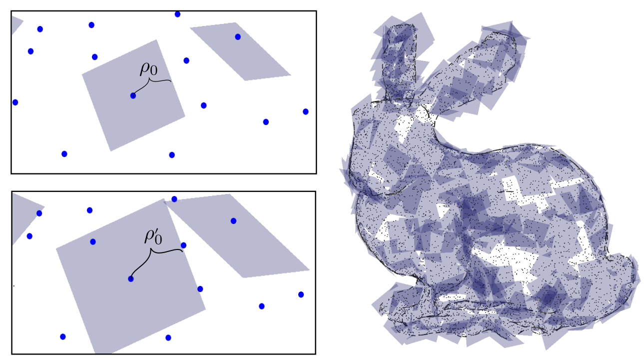


Figure 5: *Using square patches to choose ρ_0. Upper left, the first approximation to ρ_0; bottom left, the user's choice; right, the visual feedback after 2 steps of subdivision*

In this work, we use a radial function which does not attend the strict conditions presented in [Macêdo et al. 2009], the triharmonic $\phi(r) := r^3$. However, it was shown by Duchon in his seminal paper [Duchon 1977] that, for this choice of basis function and linear augmenting polynomials ($d = 1$), the resulting Hermite interpolation system is well-posed for any set of (pairwise-different) sample points. Moreover, the recovered implicit function above will minimize a suitable generalization of the *thin-plate energy* for Hermite problems in $\mathbb{R}^3$.

4.2 Seed Placement

The seed placement step is important to define the final drawing, since all points' placement will be derived from the seeds' position. We use two different techniques to place seeds. The first one is using the samples themselves as seeds. This strategy is good, since it does not require the seeds to be projected over the surface (which is an expensive process). However, it is recommended only when the samples are well distributed over the surface, which typically happens when data is sampled from a regular mesh. On the other hand, when the samples are not well distributed, we need a different strategy to obtain a good seed placement and capture all model aspects.

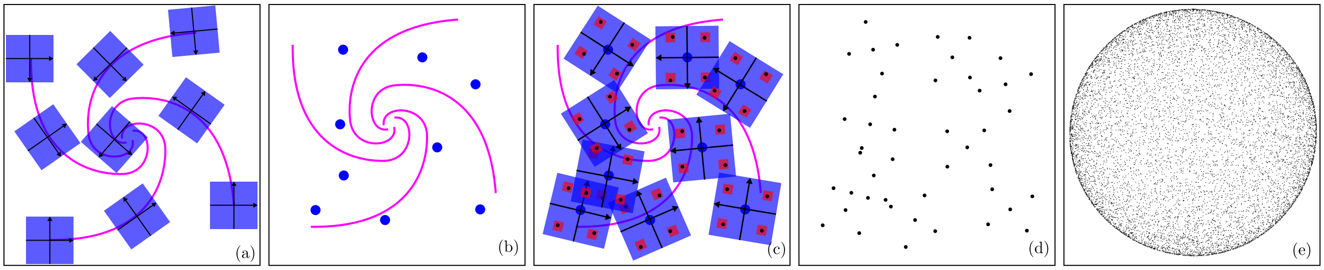


Figure 6: *Placement of stippling points: top view (a, b, c, d) and covering the entire model (e). (a) the tangent plane basis created using the steps describes in Section 5.2 near the discontinuity point; (b) seeds over the surface; (c) points generated, with red square representing the jitter range; (d) results of one subdivision step; (e) starting with 6 seed points, after 5 subdivision steps, the final results over a sphere;*

The second seed placement technique relies on HRBF's properties allowing good point projections. We fill the samples' bounding box with points and then project them on the surface. In order to do that, we define the resolution R which will be used in its largest dimension, defining the resolution of the other two dimensions to create a regular grid with cubic cells. In the center of each cell, we place a point which is randomly moved to a distance up to αL, where L is the side length of the cell (in this work, we use $R = 8$ and $\alpha = 0.25$). This random displacement reproduces the effect of jittering.

To approximate the projection of one point p onto $\mathcal{S}$, we use the optimization method of steepest descent with Armijo rule to minimize the function $f(x)^2$ and, $x^{(0)} = p$ as the initial guess (for more details see [Bertsekas 1999]). It is worth noticing that this simple method provides a good enough approximation of the projection of p on the surface as long as the function f has properties similar to those of a signed distance function in p.

5 Multi-Level Sample Refinement

After the seeds have been placed on the surface, their positions are ready to be used to generate rendering points. We divide the rendering points in three groups: stippling, principal directions of curvature, and combing directions. Stippling points are placed in a scattered fashion, focusing on covering $\mathcal{S}$ uniformly, while the two other groups provide linear mark depictions by clustering points along a directional field. All three groups share the same recursion idea. We use the actual seed positions to place a new point near the surface, located at a distance ρ from its seed. After that, we project the new points onto the surface using the same method describe in Section 4. In the next step, all the recently generated points will become seeds of its own group, and $\rho = \rho/3$ (Figure 3). The process goes on until the desired visual effect is achieved. It is important to notice that, by using this $1/3$ rule, the distance between the seed and all its descendants is limited; in fact, after k steps, the distance between the original seed and any descendant will be less than $1.5\rho_0$ and two points with the same original seed will be at most $1/3^k\rho_0$.

5.1 Sampling near the Surface

Since the projection method will be faster and more precise when the point is near the surface, we try to place new points as close as possible to it. Basically we have two approaches to place the new points: the first one uses the tangent plane of the surface, and the other uses curvature estimation (Figure 4). Finding the tangent plane of a projected or sample point of the HRBF Implicit is virtually costless, since these points already have their gradients calculated. In contrast, in our approach, the use of curvatures to estimate the new point position may be an expensive process, with a higher computational cost than to project a point a bit farther from the surface. As a result, this approach is only used when we place render points at distances ρ along one of the principal directions of curvature on the osculating circle (Section 5.3).

The initial step size ρ_0 defines whether the points will be well spread or clustered over the surface. We use two semi-automatic approaches to pick a good estimate of ρ_0. In the first approach, when placing seeds from the bounding box, the first ρ_0 approximation will be the voxel diameter. In the second approach, when seeds are placed directly from the samples, we use the average of their empty ball diameter. However, these two approaches can either underestimate or overestimate ρ_0, thus creating clusters or visually broken lines, respectively. To avoid these cases, our system allows the user to explictly set ρ_0. In order to provide a good visual feedback, our system randomly places square patches with sides equal to $2\rho_0$ over the surface (Figure 5).

5.2 Stippling Points

To generate the stippling points we use the seeds' tangent plane. We need to create a basis to the affine plane to place these points. There are many possibilities for building the basis using the normal vector as input, with all choices having at least one point of discontinuity [Stark 2009]. For this step in our pipeline, we select a method which has two points of discontinuity, after observing they avoid patterns (Figure 6(e)). To create our basis, we rotate the normal to a fixed axis $\mathbf{r}' = R\mathbf{n}$ and then compute the cross products $\mathbf{u} = \mathbf{n}\times\mathbf{r}'$ and $\mathbf{r} = \mathbf{n} \times \mathbf{u}$. After that, $\mathbf{n}$, $\mathbf{u}$ and $\mathbf{r}$ are unitized. Observe that, on the fixed axis, this method is not well defined; however, we observed this was not a problem when placing stippling points. Four points are placed near the surface by using the local coordinates of the affine plane: $p_{i,j} = i \cdot \rho\mathbf{r} + j \cdot \rho\mathbf{u}$, where $i, j = \pm 1 + u$, and $u \sim \mathcal{U}([-0.125, 0.125])$, meaning that u is a random variable with uniform distribution within the interval $[-0.125, 0.125]$ (Figure 6).

5.3 Drawing Direction

To create the perception of short continuous lines, we use the smoothness property of the HRBF and a method to create smooth directions. The idea is as follows: since we have a smooth variation of normals and a piecewise smooth function $F : \mathcal{S}\times\mathbb{S}^2 \rightarrow \mathbb{S}^2\times\mathbb{S}^2$, $F(p, \mathbf{n}) = (\mathbf{r}, \mathbf{u})$, we use these properties to create a sequence of points $p_i = \pm\rho\mathbf{w}$, where $\mathbf{w}$ could be either $\mathbf{u}$ or $\mathbf{r}$. As previously mentioned, the points will be closer to each other at each step of the subdivision; therefore, after a few steps, we have the visual perception of a line being defined (Figure 3, bottom row). At the first subdivision step, the original seeds throw points along both directions ($\mathbf{r}$ and $\mathbf{u}$). After this first subdivision step, we separate the points in two sets, generated using $\mathbf{r}$ and $\mathbf{u}$, respectively. As a re-

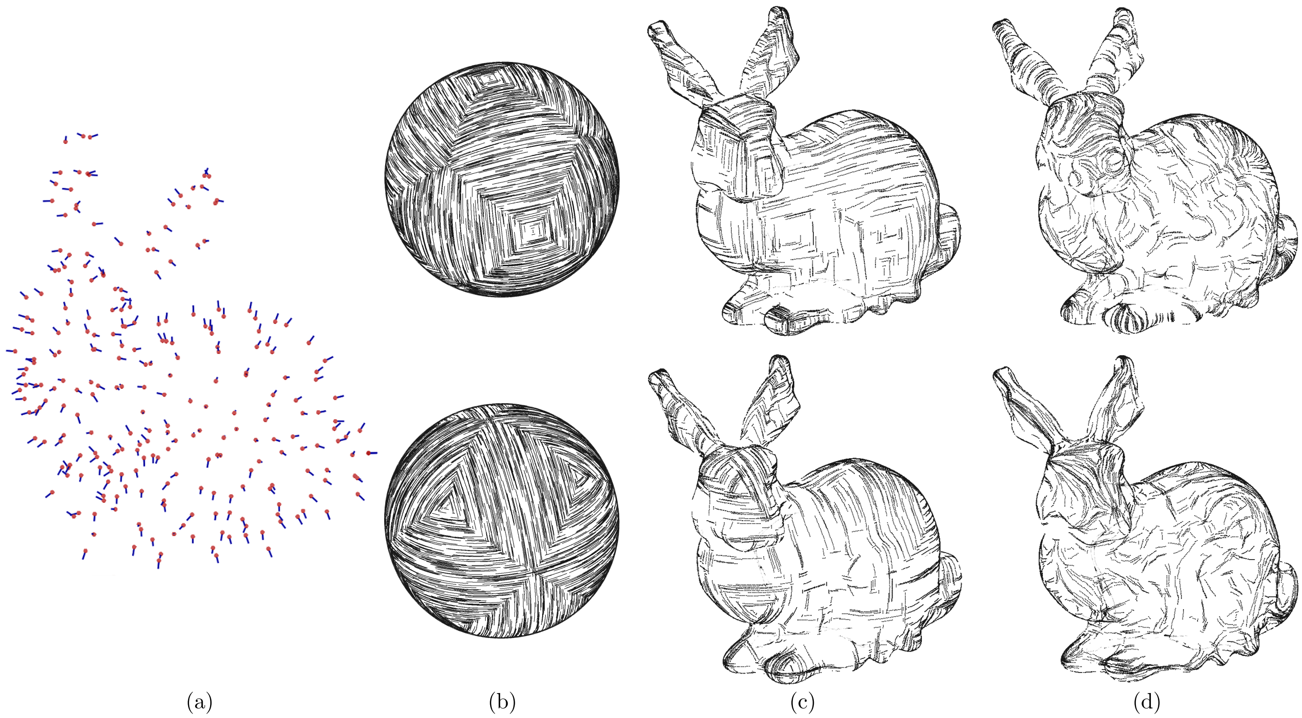


Figure 7: *Comparing drawing directions. (a) 256 point-normal samples over the Stanford bunny model. Combing directions over (b) the sphere and (c) the bunny model. (d) First and second principal directions of curvature (top and bottom bunny, respectively).*

sult, each set will only generate points along its original direction.

We work with two different functions F to create the perception of lines. The first one is the Principal Directions of Curvatures, using the method described by Kindlmann et al. [2003] to calculate a reduced Hessian matrix, followed by its eigenvalues and eigenvectors to get the principal directions and values of curvature. The second function is the *Combing* Directions, using the method described in [Stark 2009] to create the basis. This approach splits the sphere into 12 regions of directions. This partition creates a pattern which is curvature-independent. In our experiments, we observed these directions provide good perceptual cues for shape depiction and are particularly good when the surface has large variations on the curvature. In Figure 7, we compare the combing directions with principal directions of curvatures.

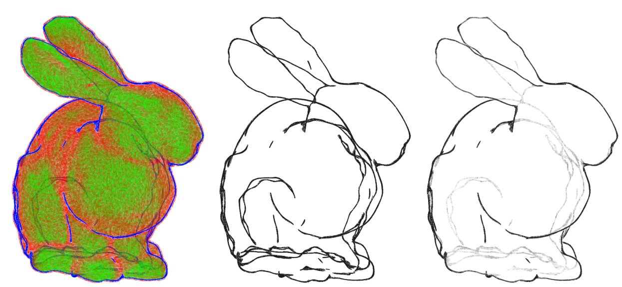


Figure 8: *Left: the α decay when the point gets closer to the silhouette: back points (red) and front points (green). Middle: without hidden line attenuation. Right: final result.*

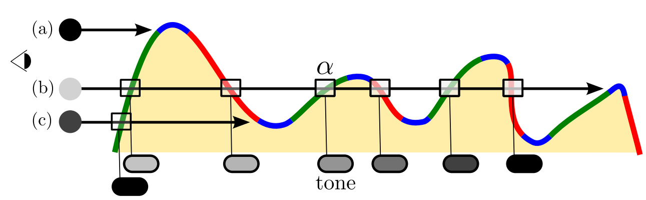


Figure 9: *Different levels of hidden-line attenuation accumulated along the viewing direction: (a) none (b) full (c) partial. Ellipses correspond to the tone value at the intersection point (between surface and viewing directions). Boxes correspond to the alpha attenuation at the point. Line colors correspond to front-faces, back-faces and silhouettes (red, green and blue, respectively).*

6 Rendering

At this stage, we are ready to use the render points already placed over the surface to visualize the implicit model in different styles. The render points are classified in three sets: front, back and silhouette. We calculate $\nu = \mathbf{n} \cdot \mathbf{v}$, where $\mathbf{n}$ is the normal at the point and $\mathbf{v}$ is the viewing vector. Using a threshold $\delta > 0$, we identify front points when $\nu < -\delta$, back points when $\nu > \delta$, and silhouette points otherwise. After classifying all points, different rendering effects are created, as described next.

6.1 Silhouettes and Hidden-Line Attenuation

In our system, the silhouette points are always displayed; however, occluded points could appear, thus creating artifacts. We would like to provide different visual effects instead of simply removing occluded points (Figure 8, middle and right). We attenuate hidden-lines by displaying the back and front points in the same color as

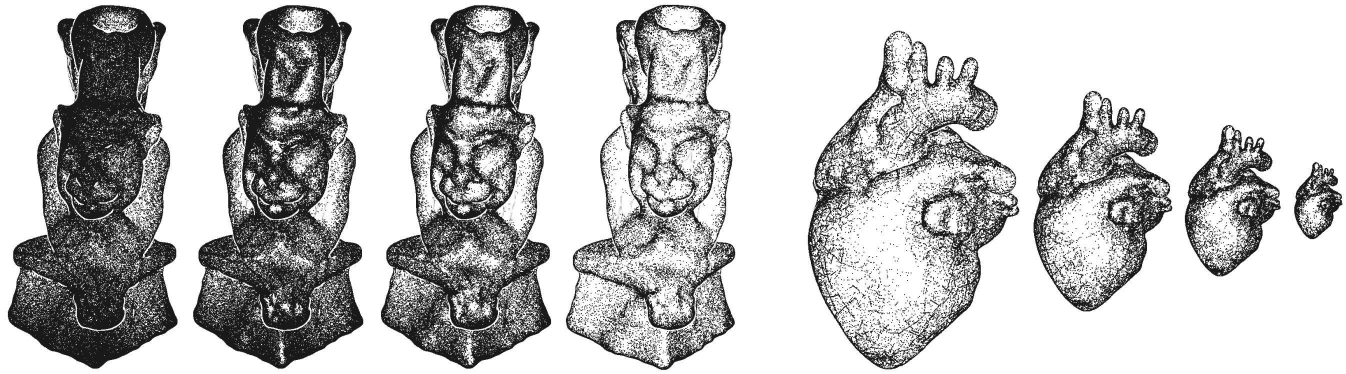


Figure 10: *(from left to right) Tone depiction by removing render points proportionally to the light intensity. Scaling a shaded heart model by removing render points.*

the background and with an opacity value $\alpha \in [0,1]$ (Figure 9). The tone of the silhouette point will be closer to the background's as much as its depth-complexity. To be sure the silhouette points will not be occluded by other points, we use a decay function for α. The α values of the front points have a quadratic decay function $\alpha_f = \nu^2\lambda$, and we use $\alpha_b = 0.2(\log(\nu - .05) + 5)\lambda$ to the back points (Figure 8, left), where λ is a parameter controlled by the user. If $\lambda = 0$, all silhouettes are displayed (Figure 8, middle); if $\lambda > 0$, then we have line attenuation (Figure 8, right). The size of the back points and its α-decay function allows to create a halo effect on the silhouette cusp points, but we need to control the point size to achieve the same visual effect independent of scale. In the next section, we provide more details regarding how to control the scaling effect.

6.2 Tone Depiction

In our approach, tone is depicted by removing front-points from the surface to create three main types of effects: shading, depth attenuation and tone scaling. The front points are removed randomly, using different probability density functions. The back points are plotted following the same rules of the former section. Lighting effects are achieved by calculating the tone $\tau \in [0,1]$ at the point, using any choice of illumination model (Figure 10). Let us define a random variable $u \sim \mathcal{U}[0,1]$. A render point is displayed only if $u \geq \tau$. In this work, lighting effects were generated using $\tau = (\mathbf{n} \cdot \mathbf{l})^\lambda$, where $\mathbf{l}$ is the unit light vector and the parameter λ allows the user to control the light intensity. Depth attenuation is achieved by removing both silhouette and front points. Similarly to lighting effect, points with $\tau \geq u$ are removed. We use the approach presented in [Barla et al. 2006], $\tau = 1 - \log(d/d_{min})/\log(d_{max}/d_{min})$, where d is the distance between the point and the camera and d_{min} and d_{max} is the depth where we start the attenuation and the far visible depth, respectively (Figure 13). Tone scaling preserves shading coherence when the model is scaled up or down due to camera motion (Figure 10). The chance of a front-point being displayed has as an exponential probability density function with the zoom factor as a variable; To control the Halo-effect, we use the same function, but now to affect the size of the back-point.

7 Results and Discussion

Our NPR techniques successfully depict shape and tone of HRBF Implicits by extracting and rendering silhouettes with hidden-line attenuation, stippling, and hatching following principal curvatures and combing directions. All the results were generated on an 2.66 GHz Intel Xeon W3520, 4 gigabyte of RAM and OpenGL/nVIDIA Quadro FX 3800 graphics. Timings are presented in Table 1 for models representing a variety of subjects. Our results were generated with point samples from 3D meshes (*Stanford bunny, Heart, Gargoyle, Lamp,* and *David*), parametric surfaces (*Tori* and *Knot*), height-maps (*Terrain*) and implicit surfaces (*Sphere*). All preprocessing and run-time rendering were computed on the CPU only.

Model	Dir.	Samples	Stippling	Hatching	FPS
Sphere(f. 7)	C	6 (0.0s)	3.7K (0.1s)	600K (8.2s)	20
Bunny (f. 7)	C	256 (0.1s)	1.4K (1.8s)	75K (45.8s)	80
Bunny (f. 7)	P	256 (0.1s)	1.4K (1.8s)	75K (70.2s)	80
Heart (f. 10)	P	512 (0.9s)	63k (97s)	81K (120s)	69
Trrn (f. 13)	P	512 (0.9s)	63k (105s)	335k (635s)	26
Grgl (f. 10)	–	1024 (5.3s)	360k (950s)	–	27
Bunny (f. 8)	–	1024 (5.1s)	630k (1831s)	–	19
Tori (f. 11)	P	1024 (5.3s)	126k (399s)	53k (172s)	55
Knot (f. 2, 11)	P	1440 (13s)	178k (707s)	74k (343s)	36
Lamp (f. 12)	–	2469 (60s)	308k (1976s)	–	39
David (f. 1)	C	4096 (267s)	510k (5383s)	212K (1747s)	16

Table 1: *Time (in seconds) given a drawing direction (C for combing, P for principal directions of curvature); number of samples, stippling and hatching marks, and frames per second.*

Table 1 shows that all models used in our experiments are rendered in real-time. Also, the pre-processing time depends on the number of points and samples. As expected, more samples result in more complex HRBF computations. In addition, placing render points along principal directions of curvature takes longer than along the combing directions.

Our approach produces promising results approximating pen-and-ink styles as found in line drawings executed by hand on models reconstructed from a given small set of point-normal samples. We evaluated our results by observing how close they approximate traditional pen & ink drawings.

Figure 1 illustrates the steps for rendering pen-ink drawings using our system, in a similar way as found in traditional drawing production (i.e. from initial sketch to finished rendering). Given a 3D set of point-normal samples, (a) our system initially reconstructs the model and allow the user to select rendering techniques providing different levels of visual abstractions for shape and tone depiction using lighting. For instance, adding silhouettes with hidden-line attenuation (b, c), hatching along combing direction (d), and then adding stippling (e). **Figure 2**(d) illustrates tone depiction of a knot model rendered using a combination of curvature-based hatching, stippling and slight hidden-line attenuation (enhancing the shape depiction). **Figure 7** provides a comparison between hatching along combing directions and principal directions of curvature on the Stanford bunny fitted from 256 samples and using the bounding-box approach to place seed points onto the model (Section 4.2). Observe the different shape depiction abstractions. **Figure 11** shows a combination of the principal directions of curvature rendered over two models. Notice that strokes placed along the first principal direction of curvature depict volumes, while strokes placed along the second direction of curvature direct our eyes along the length of the model [Rawson 1987; Goldstein 1999; Hertzmann and Zorin 2000; Sousa et al. 2004]. **Figure 12** illustrates a garden lamp model with planar regions. Note that sharp features are adequately rendered. **Figure 13** shows a Canyon terrain model (512 point-normal samples) rendered in different ink-based styles, properly depicting both shape and tone.

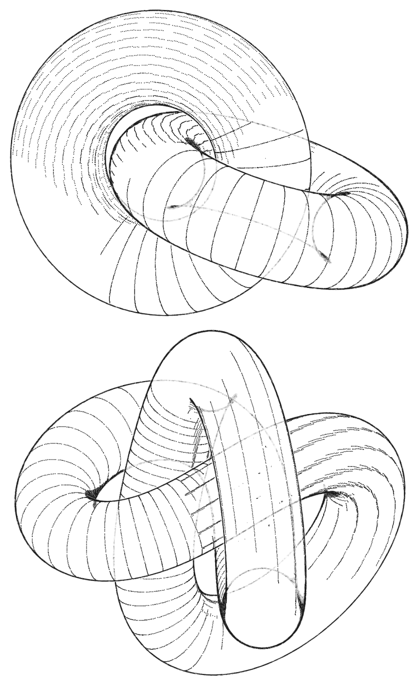


Figure 11: *First and second principal directions of curvature applied locally on two tori and a knot.*

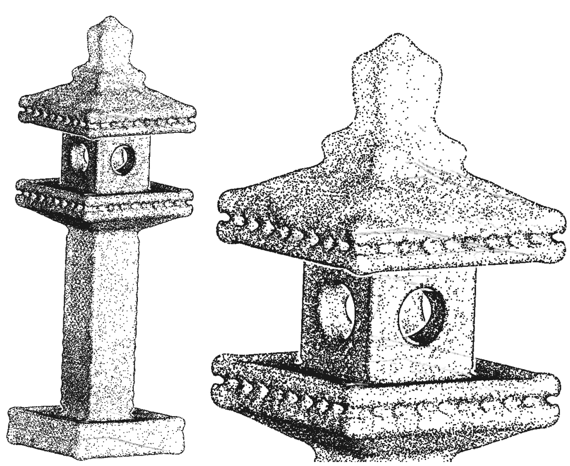


Figure 12: *General and detailed rendering of a garden lamp.*

8 Conclusions and Future Work

In this work, we present a completely point-based approach for depicting shape and tone in models represented as implicit surfaces. For this representation, we employ the recently introduced HRBF Implicits for their good properties in reconstructing implicit models from few samples consisting of points and their associated normals. Among the features of our approach, the most salient issues regard our strategies for placing initial seeds, a multilevel refinement of points over the surface, choices of refinement directions suggesting lines drawn along principal directions of curvature, a tiled direction field (the combing direction), a smoother curvature-independent choice of directions, and also new approaches to depict shape and tone by implementing, in a simple manner and directly over the HRBF implicit model, ink-based NPR techniques including silhouettes, attenuated hidden-lines and lighting tones by stippling and hatching. Our approach demands an initial pre-processing which densely samples the implicit surface, however, after this step, both camera and lighting parameters can be changed at interactive rates on the CPU.

There are still many avenues for further improvement one may explore: the pre-processing step might be made faster by exploiting the parallelizability of the seed placement and point refinement. The evaluation of the HRBF Implicits interpolant could be implemented in graphics hardware while its fitting made in CPU. Also, even though the basic geometric operators on which we rely in designing our pipeline are general enough to be implemented for different representations, we have only experimented with models based on HRBF Implicits. We plan to experiment with data and representations from different application domains to further evaluate the suitability of our method. Since we tried to design a method as automatic as possible, we left aside concerns of controllability in the final rendering. Research on more flexible and artist-driven tools for rendering and content creation is an important and relevant area for improving and integrating our method. Also, specific stylization effects for individual drawing primitives is an interesting and important topics to investigate and integrate in our system [Xu and Chen 2004; Kim et al. 2009]. Finally, a more formal evaluation with trained artists and illustrators should be performed and might indicate directions for further usability investigation.

Acknowledgements

We would like to thank the Digital Michelangelo Project, Stanford University for the David model, the Stanford Computer Graphics Laboratory, NTU 3D Model Database, and Rich Pito (University of Pennsylvania, GRASP Lab) who made the other models available for use in this paper. Many thanks to Nicole Sultanum and Patricia Rebolo Medici for their useful discussions and advice. We also thank the anonymous reviewers for their careful and valuable comments and suggestions. This research was supported in part by the iCORE/Foundation CMG Industrial Research Chair in Scalable Reservoir Visualization, by Discovery Grants Program from the Natural Sciences and Engineering Research Council of Canada, and grants from the Brazilian funding agencies CNPq and CAPES/PDEE.

References

AKLEMAN, E. 1998. Implicit painting of CSG solids. In *Proc. of CSG '98, Set-Theoretic Solid Modelling: Techniques and Applications*, 99–113.

ANDREWS, W. M. 2006. Introduction to perceptual principles in medical illustration: lines & illusions. *Tutorial Notes, Illustrative Visualization for Medicine and Science (Eurographics '06)*.

BARLA, P., THOLLOT, J., AND MARKOSIAN, L. 2006. X-toon: an extended toon shader. In *Proc. of 4th international symposium on Non-photorealistic animation and rendering (NPAR '06)*, 127–132.

BERTSEKAS, D. P. 1999. *Nonlinear Programming*, 2nd ed. Athena Scientific.

BREMER, D., AND HUGHES, J. F. 1998. Rapid approximate silhouette rendering of implicit surfaces. In *Proc. of Implicit Surfaces '98*, 155–164.

BURNS, M., KLAWE, J., RUSINKIEWICZ, S., FINKELSTEIN, A., AND DECARLO, D. 2005. Line drawings from volume data. *ACM Transactions on Graphics (SIGGRAPH '05) 24*, 3, 512–518.

DEUSSEN, O. 2009. Aesthetic placement of points using generalized Lloyd relaxation. In *Proc. of 5th Intl. Symposium on Computational Aesthetics in Graphics, Visualization and Imaging (CAe'09)*, Eurographics Association, 123–128.

DUCHON, J. 1977. *Splines minimizing rotation-invariant seminorms in Sobolev spaces*, vol. Constructive Theory of Functions of Several Variables, 571 of *Lecture Notes in Mathematics*. Springer Berlin Heidelberg, 85–100.

ELBER, G. 1998. Line art illustrations of parametric and implicit forms. *IEEE Transactions on Visualization and Computer Graphics 4*, 1, 71–81.

FOSTER, K., JEPP, P., WYVILL, B., SOUSA, M. C., GALBRAITH, C., AND JORGE, J. A. 2005. Pen-and-ink for blobtree implicit models. *Computer Graphics Forum (Eurographics '05) 24*, 3, 267–276.

GOLDSTEIN, N. 1999. *The Art of Responsive Drawing*. Prentice-Hall.

HERTZMANN, A., AND ZORIN, D. 2000. Illustrating smooth surfaces. In *Proc. of SIGGRAPH '00*, 517–526.

JEPP, P., WYVILL, B., AND SOUSA, M. C. 2006. Smarticles for sampling and rendering implicit models. In *Proc. of 4th Theory and Practice of Computer Graphics (TPCG'06)*, 39–46.

JEPP, P., DENZINGER, J., WYVILL, B., AND SOUSA, M. C. 2008. Using multi-agent systems for sampling and rendering implicit surfaces. In *Proc. of XXI Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI '08)*, 255–262.

JEPP, P., ARAUJO, B. D., JORGE, J., WYVILL, B., AND SOUSA, M. C. 2009. Style nodes for hierarchical tree-based implicit surface modelling. In *Proc. of 5th Intl. Symposium on Computational Aesthetics in Graphics, Visualization and Imaging (CA'09)*, 41–48.

KIM, S. Y., MACIEJEWSKI, R., ISENBERG, T., ANDREWS, W. M., CHEN, W., SOUSA, M. C., AND EBERT, D. S. 2009. Stippling by example. In *Proc. of 7th Intl. Symposium on Non-Photorealistic Animation and Rendering (NPAR '09)*, 41–50.

KINDLMANN, G., WHITAKER, R., TASDIZEN, T., AND MOLLER, T. 2003. Curvature-based transfer functions for direct volume rendering: Methods and applications. In *Proc. of 14th IEEE Visualization (VIS '03)*, 513–520.

LOHAN, F. J. 1978. *Pen & Ink Techniques*. Contemporary Books, Inc.

MACÊDO, I., GOIS, J. P., AND VELHO, L. 2009. Hermite interpolation of implicit surfaces with radial basis functions. In *Proc. of XXII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI '09)*, 1–8.

PLANTINGA, S., AND VEGTER, G. 2006. Computing contour generators of evolving implicit surfaces. *ACM Transactions on Graphics 25*, 4, 1243–1280.

PROENÇA, J., JORGE, J., AND SOUSA, M. 2007. Sampling point-set implicits. In *Proc. of 4th IEEE/Eurographics Symposium on Point-Based Graphics (PBG '07)*, 11–8.

PROENÇA, J., JORGE, J., AND SOUSA, M. 2008. Suggestive contours over point-set implicits. In *Proc. of 3rd Intl. Conference on Computer Graphics Theory and Applications (GRAPP '08)*, 171–180.

RAWSON, P. 1987. *Drawing*. University of Pennsylvania Press.

RICCI, A. 1973. A constructive geometry for computer graphics. *The Computer Journal 16*, 2, 157–160.

ROSTEN, E., AND DRUMMOND, T. 2003. Rapid rendering of apparent contours of implicit surfaces for realtime tracking. In *British Machine Vision Conference*, 719–728.

SCHMIDT, R., ISENBERG, T., JEPP, P., SINGH, K., AND WYVILL, B. 2007. Sketching, scaffolding, and inking: a visual history for interactive 3D modeling. In *Proc. of the 5th Intl. Symposium on Non-Photorealistic Animation and Rendering (NPAR '07)*, 23–32.

SMITH, J. A. 1992. *The Pen and Ink Book: Materials and Techniques for Today's Artist*. Watson-Guptill Publications.

SOUSA, M., SAMAVATI, F., AND BRUNN, M. 2004. Depicting shape features with directional strokes and spotlighting. In *Proc. of Computer Graphics International (CGI '04)*, 214–221.

STARK, M. M. 2009. Efficient construction of perpendicular vectors without branching. *Journal of Graphics, GPU, & Game Tools 14*, 55–62.

STROILA, M., EISEMANN, E., AND HART, J. 2008. Clip art rendering of smooth isosurfaces. *IEEE Transactions on Visualization and Computer Graphics 14*, 1, 135–145.

WENDLAND, H. 1995. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics 4*, 1, 389–396.

XU, H., AND CHEN, B. 2004. Stylized rendering of 3D scanned real world environments. In *Proc. of the 3rd Intl. Symposium on Non-Photorealistic Animation and Rendering (NPAR'04)*, 25–34.

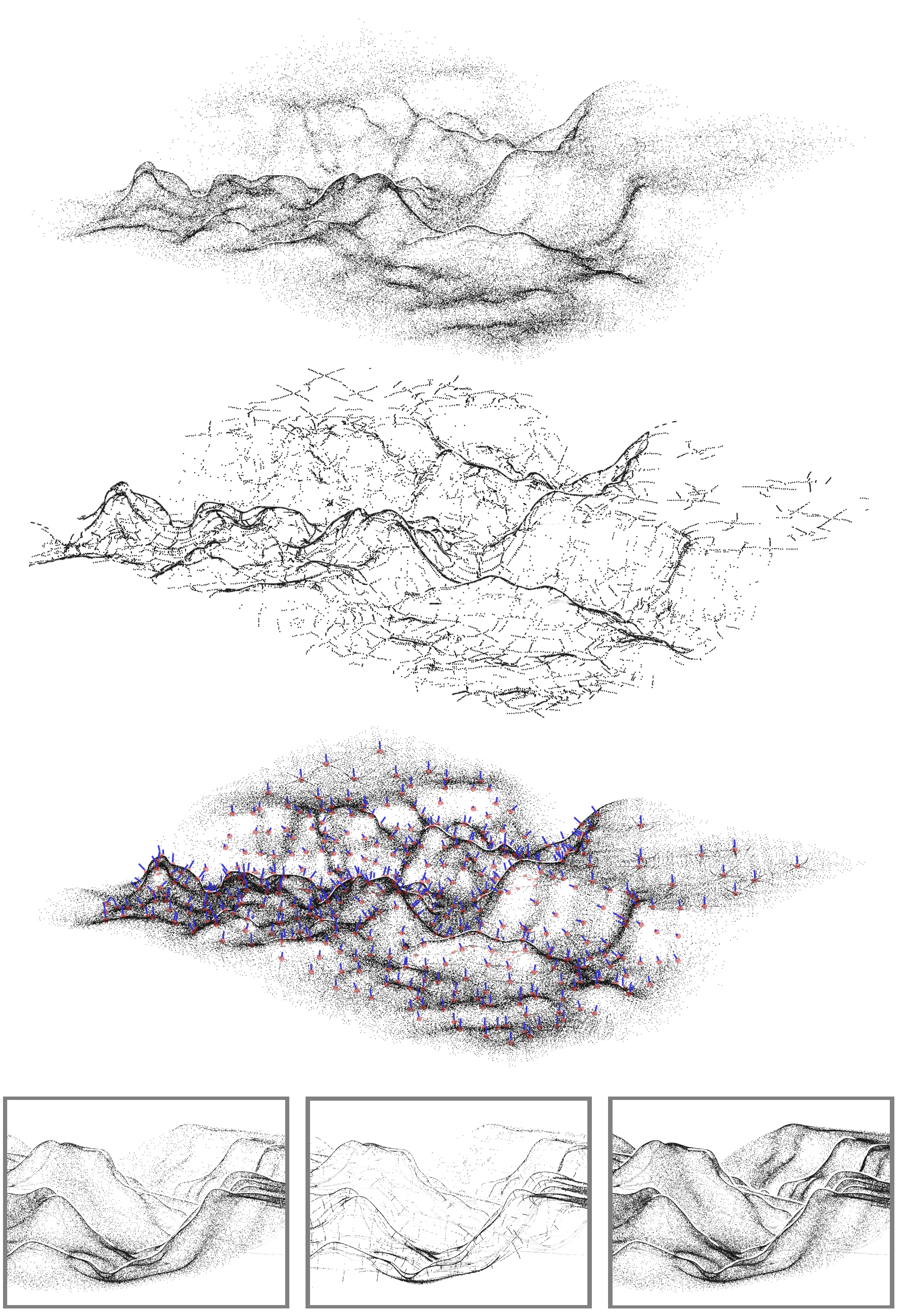


Figure 13: *Shape and tone depiction of the Canyon terrain model in different pen & ink styles using our system. From top to bottom: using stippling with depth modulation; Cross-hatching the principal directions of curvature; the 512 point-normal samples used for the HRBF reconstruction; three different visual abstractions of the model. The model has 512 samples, 400K render points and with CPU rendering at 26 fps.*