Rendering with Parallel Stripes

Gershon Elber
Computer Science Department
Technion
Haifa 32000, Israel
Email: gershon@cs.technion.ac.il

February 20, 2001

Abstract

This paper presents an artistic rendering scheme that is based on parallel stripes and inspired by Victor Vasarely’s art work. The rendering process is conducted using parallel planar curves that are warped and translated in the projection plane an amount that is a function of the depth of the object, at that location.

In this work, the parallel stripes are derived as a set of isoparametric curves out of a warped injective B-spline surface that is derived from a Z map of a Z-buffer of the scene.

Key Words: Injective mapping, B-spline curves and surfaces, Victor Vasarely, Artistic rendering, Nonphotorealistic rendering.

1 Introduction

In recent years, line art rendering has played a major role, in the computer graphics community. Going beyond the efforts toward photorealism, the attempt to create aesthetic and pleasing synthetic line art pen-and-ink illustrations has been surprisingly successful. The origin of line art techniques could be found long before the computer graphics era, in wood carving and classical art [Hasl77, Ivin92]. In one of the first computer based, electronic, attempts, [Pnue94] constructs line art illustrations from images, a work that was inspired by Dürer [Ivin92]. Another example of an image based rendering system is Piranesi [Lans95] that captures and exploits depth information from the three dimensional model as a Z-map, allowing the user to select certain regions in the scene and paint them with prescribed surface normals and material types. Earlier, in [Sait90], derivatives of the Z-map image of the scene were also exploited in the enhancement of feature lines such as silhouette and boundary edges. In several locations, for example [Velh91], space filling curves are also used in the image plane to produce digital half-toning. The classical space filling curves of Peano, Hilbert, and Sierpinsky are considered.

Clearly, synthetic construction of line art is not limited to image based input. The additional information in the three dimensional geometry can improve the final, line art, result. Shadows can be
computed, texture maps can be added, and obviously shading can be set for arbitrary positions of light sources. In [Dool90a, Dool90b], feature lines, such as silhouettes and discontinuities, are employed in constructing illustrations that combine shaded display and line drawings. Levels of visibility are also conveyed in [Dool90a, Dool90b] by using different line styles such as dashed and dotted line types. In [Leis94], ray tracing and various texture mapping based techniques are employed to hatch three dimensional models, hatching that is produced using thresholding on the result. In [Wink94], the possibilities of applying strokes, tones and textures, and outlines or silhouette curves for polygonal models, are considered. In [Elbe95b, Wink96], similar algorithms to construct line art renderings of freeform parametric surfaces are presented.

This paper has been inspired by Victor Vasarely’s [Vasa1, Vasa2, Vasa3] early work during the thirties, on illusions that are created by deformed curves. Vasarely presented a method to create the illusion of a three dimensional scene by distorting parallel curves in the plane (See Figure 1). Examining this work of Victor Vasarely, the parallel curves are warped and translated an amount that reflects on the shape and more so on the depth of the scene at the local neighborhood. Even more interesting is the fact that no two curves cross each other in Figure 1, creating the sensation of continuity. This stripes based technique was also used in a more realistic drawings with shading and gray levels, for example in the Martian, 1946 of Vasarely [Vasa3].

Figure 1: An illusive illustration by Victor Vasarely, 1938.
In this work, we aim at creating an automated system that would produce similar illusive drawings from three dimensional geometry. Toward this end, we first establish a scheme that would prohibit the crossing of warped curves, a scheme that is based on planar warping that is injective or one-to-one.

In [Lee95], a condition for a planar uniform bicubic B-spline surface to be injective is derived, motivated by the image morphing application, where the mapping of the metamorphosis process is required to be injective. A sufficient injective condition is derived in [Lee95] via bounds on the displacements of the control points of the B-spline surface.

A tighter condition than [Lee95] for a planar uniform bicubic B-spline surface to be injective is derived in [Choi99] by examining the upper bounds on the partial derivatives of $S(u, v)$. This condition is also sufficient only. Both conditions of [Lee95] and [Choi99] do not permit the control mesh to self intersect, even if the final B-spline surface continues to be self intersection free and one-to-one.

In [Good96], the condition for a Bézier surface to be injective is derived, a condition that is reformulated as $O(n^2)$ linear inequalities, where the surface is of degrees $(n \times n)$. In [Carnicer99], linear conditions for a determinant of a matrix to be positive are derived, extending [Good96] to higher dimensions. In this work, we extend the result of [Good96] to B-spline surfaces with floating end conditions and uniform knots.

This paper is organized as follows. In Section 2, we derive a tighter condition than [Choi99] for a uniform bicubic B-spline surface with floating end conditions [Far93] to be injective. In Section 3, the algorithm to create renderings of parallel stripes that follow Victor Vasarely’s style is presented. Examples are portrayed in Section 4 and we conclude in Section 5.

The implementation as well as all the examples shown as part of this work were created with the aid of the IRIT solid modeling system [Irit20] that is developed at the Technion, Israel Institute of Technology.

2 Injective Mapping

Let

$$S(u, v) = (x(u, v), y(u, v)) = \sum_{i} \sum_{j} (x_{ij}, y_{ij}) B_i(u) B_j(v),$$

be a continuous planar B-spline surface of degrees $(n \times n)$ [Far93]. Let the partial derivatives of $S(u, v)$ be denoted as $\frac{\partial x(u, v)}{\partial u}$, $\frac{\partial y(u, v)}{\partial u}$, $\frac{\partial x(u, v)}{\partial v}$, $\frac{\partial y(u, v)}{\partial v}$.

Due to the continuity of $S(u, v)$, $S(u, v)$ is locally injective if and only if the determinant of the
Figure 2: A surface can have a non-vanishing Jacobian and yet fail to be one-to-one.

Jacobian of the mapping $S$ never vanish. Denote by $J$ the determinant of the Jacobian:

$$J[S](u,v) = \det \begin{pmatrix} \frac{\partial x(u,v)}{\partial u} & \frac{\partial y(u,v)}{\partial u} \\ \frac{\partial x(u,v)}{\partial v} & \frac{\partial y(u,v)}{\partial v} \end{pmatrix} = \frac{\partial y(u,v)}{\partial u} \frac{\partial x(u,v)}{\partial v} - \frac{\partial x(u,v)}{\partial u} \frac{\partial y(u,v)}{\partial v}$$

(2)

By local, we say that for a sufficiently small neighborhood, if $J[S](u,v) \neq 0$, the surface is one-to-one. In contrast, examine the surface in the shape of an $\alpha$ that is shown in Figure 2. While for this surface, $J[S](u,v) \neq 0$ holds throughout, the mapping is clearly not injective. Hence after, we consider the injective condition in the local sense only.

If $S$ is of degrees $(n \times n)$, $J[S]$ is of degrees $(2n-1) \times (2n-1)$. Consider a single polynomial patch, $P_{kl}(u,v)$ of $S(u,v)$. $P_{kl}(u,v)$ has $(n+1)^2$ control points that affect it. Moreover, the Jacobian of $P_{kl}$, $J[P_{kl}]$, is another polynomial of degrees $(2n-1) \times (2n-1)$. Clearly, being a polynomial, $J[P_{kl}]$ could be represented as a Bézier and/or B-spline surface of degrees $(2n-1) \times (2n-1)$.

Having $J[P_{kl}]$ represented as a Bézier and/or B-spline surface of degrees $(2n-1) \times (2n-1)$, and due to the convex hull containment property of the representation, $J[P_{kl}]$ is never zero if all the coefficients of the function of $J[P_{kl}]$ in the Bézier and/or B-spline representation are positive or all of them are negative. A simple algorithm to verify the injectivity condition of a B-spline surface could therefore be derived and is presented in Algorithm 1.

A main computational difficulty in Algorithm 1 is found in the need to convert each individual polynomial patch of $S(u,v)$ into a Bézier form. In general, having a non-uniform B-spline function, the basis conversion procedure must be resolved for each individual patch. Nonetheless, for uniform knot sequences, that solution may be computed once for the entire surface. Furthermore, this single solution could be precomputed ahead of time.

Consider the case of a uniform B-spline surface. In the ensuing discussion, we will restrict ourselves to bicubic B-spline surfaces with floating end conditions. Then, the basis functions of all the different
Algorithm 1
Input:
\( S(u, v) \), A B-spline surface;
Output:
TRUE if \( S(u, v) \) is injective, FALSE otherwise;
Begin
For \( k \) runs over all patches of \( S(u, v) \) in \( u \) Do
For \( l \) runs over all patches of \( S(u, v) \) in \( v \) Do
\[ J[P_{kl}] (u, v) \Leftarrow \text{Jacobian of function of patch } P_{kl}; \]
\[ J[P_{kl}] (u, v) \Leftarrow J[P_{kl}] (u, v) \text{ as a Bezier function; } \]
if \( J[P_{kl}] (u, v) \) has coefficients with different sign; return FALSE;
End
return TRUE;
End

patches of \( S(u, v) \) are identical and one is required to solve for the basis conversion functions only once.
Let,
\[
P_{kl}^3 (u, v) = \sum_{p=k}^{k+3} \sum_{q=l}^{l+3} (x_{pq}, y_{pq})B_{p-k}^3 (u)B_{q-l}^3 (v),
\]
be such single bicubic patch, with \( B_i^j (u) \), \( i = 0, 1, 2, 3 \) denoting the four uniform bicubic B-spline basis functions. Further, let,
\[
\frac{\partial x(u, v)}{\partial u} = \sum_{p=k}^{k+3} \sum_{q=l}^{l+3} x_{pq}B_{p-k}^3 (u)B_{q-l}^3 (v),
\]
\[
\frac{\partial y(u, v)}{\partial u} = \sum_{p=k}^{k+3} \sum_{q=l}^{l+3} y_{pq}B_{p-k}^3 (u)B_{q-l}^3 (v),
\]
\[
\frac{\partial x(u, v)}{\partial v} = \sum_{p=k}^{k+3} \sum_{q=l}^{l+3} x_{pq}B_{p-k}^3 (u)B_{q-l}^3 (v),
\]
\[
\frac{\partial y(u, v)}{\partial v} = \sum_{p=k}^{k+3} \sum_{q=l}^{l+3} y_{pq}B_{p-k}^3 (u)B_{q-l}^3 (v).
\]

Then, from Equation (2),
\[
J[P_{kl}^3] (u, v) = \sum_{p=k}^{k+3} \sum_{q=l}^{l+3} x_{pq}B_{p-k}^3 \left( \frac{\partial}{\partial u} \right) B_{q-l}^3 (v) \sum_{r=k}^{k+3} \sum_{t=l}^{l+3} y_{rt}B_{r-k}^3 (u)B_{t-l}^3 \left( \frac{\partial}{\partial v} \right) (v)
\]
\[
- \sum_{p=k}^{k+3} \sum_{q=l}^{l+3} y_{pq}B_{p-k}^3 (u)B_{q-l}^3 \left( \frac{\partial}{\partial v} \right) (v) \sum_{r=k}^{k+3} \sum_{t=l}^{l+3} x_{rt}B_{r-k}^3 (u)B_{t-l}^3 \left( \frac{\partial}{\partial u} \right) (v).
\]
$J[P^5_{kl}]$ is a bivariate polynomial function of degrees $(5 \times 5)$ and hence could be represented as a bivariate Bézier function of degrees $(5 \times 5)$:

$$J[P^5_{kl}](u, v) = \sum_{i=0}^{5} \sum_{j=0}^{5} c_{ij}\theta^5_i(u)\theta^5_j(v),$$

where $\theta^5_i(u)$ and $\theta^5_j(v)$ are the quintic Bézier basis functions. Knowing that a unique biquintic polynomial surface exist, one can equate Equation (6) with Equation (5) at $36 = 6 \times 6$ different $u$ and $v$ values resulting in 36 equations and 36 unknown to solve for. By selecting the node point, also known as greville abscissa [FarI93], $u = i/5, i = 0, ..., 5$ and $v = j/5, j = 0, ..., 5$, maximal stability of these equations could be established.

The 36 equations are linear in the unknowns, $c_{ij}$, but are quadratic in $x_{kl}$ and $y_{kl}$ because given $u = u_0$ and $v = v_0$, each term of Equation (5) contains a product of the form $a x_{pq} y_{rt}$, $a \in \mathbb{R}$. Maple [MapI] solves this set of 36 equations in few minutes on a modern workstation and, for example, the solution of $c_{00}$ equals,

$$c_{00} = \frac{-x_{20} \cdot y_{10}}{36.0} + \frac{x_{10} \cdot y_{20}}{36.0} - \frac{x_{20} \cdot y_{00}}{72.0} + \frac{x_{00} \cdot y_{20}}{36.0} - \frac{x_{10} \cdot y_{00}}{9.0} + \frac{x_{00} \cdot y_{10}}{9.0} - \frac{x_{00} \cdot y_{01}}{9.0} + \frac{x_{00} \cdot y_{21}}{9.0} - \frac{x_{00} \cdot y_{12}}{9.0} + \frac{x_{00} \cdot y_{22}}{9.0} - \frac{x_{01} \cdot y_{02}}{9.0} + \frac{x_{01} \cdot y_{12}}{9.0} - \frac{x_{02} \cdot y_{11}}{9.0} + \frac{x_{02} \cdot y_{21}}{9.0} - \frac{x_{02} \cdot y_{22}}{9.0} + \frac{x_{01} \cdot y_{02}}{9.0} - \frac{x_{01} \cdot y_{12}}{9.0} + \frac{x_{02} \cdot y_{11}}{9.0} - \frac{x_{02} \cdot y_{21}}{9.0} + \frac{x_{02} \cdot y_{22}}{9.0}.$$

Having $c_{00}$ as one of the smallest expression, it is obvious we are unable to provide all 36 expressions as part of the paper. One can find the complete solution and implementation as part of the IRIT solid modeling system [Iri20].

With these 36 inequalities in $x_{ij}$ and $y_{ij}, i, j = 0, 1, 2, 3$, one can evaluate all the coefficients of the bivariate Bézier function form of $J[P^5_{kl}](u, v)$ in Equation (6) and examine and verify that all these coefficients share the same sign. Moreover, by applying refinement [Cohe80] to the Bézier patch and examining the signs of the refined control points, one can arbitrarily closely converge to a necessary condition for a one-to-one mapping.

While this condition continues to be only sufficient, it is tighter than the conditions presented in [Choi99, Lee95]. Moreover, the injectivity condition presented herein could be similarly employed.
Figure 3: Two examples of uniform bicubic B-spline surfaces with self-intersecting meshes, that are injective. These two surfaces are correctly identified as injective by Algorithm 1. The B-spline surfaces are shown in thick gray, whereas the (self-intersecting) control meshes are shown in thin black lines.

for arbitrary orders and could be exploited even for the case of nonuniform B-spline functions at the expense of the individual evaluation of the Bézier and/or B-spline polynomial patches for each interior knot interval. In addition, the derived condition allows the control mesh to self intersect and overlap while correctly detecting if the surface is a one-to-one mapping or not. Two examples for such extreme cases are presented in Figure 3.

3 Proposed Algorithm

Armed with an ability to tightly examine if a uniform bicubic B-spline surface is indeed injective, we are ready to derive our main algorithmic approach toward the emulation of Vasarely’s like drawings.

As already stated, the fact that non of the drawn curves should cross each other suggests a solution in the form of a set of input parallel lines that is combined with an injective mapping \( \mathcal{M} : \mathbb{R}^2 \rightarrow \mathbb{R}^2 \). The later maps the set of parallel lines into a new set of warped parallel curves that represent the final drawing. The amount of warping should locally depend on the depth, as prescribed by the Z-buffer, at that location. In this work, we are about to employ an injective uniform bicubic B-spline surface to represent this warping function and about to exploit adjacent isoparametric curves of the same surface as the output curves.

Given a geometric scene, \( \mathcal{S} \), \( \mathcal{S} \) is first rendered into a Z-buffer, keeping only the depth information.
Then, an iterative process translates one control point at a time the desired amount that is prescribed by the depth, subject to the preservation of the injectivity condition.

Let \( S(u, v) = \sum_i \sum_j P_{ij} B_i(u) B_j(v) \) be an injective bicubic uniform B-spline surface, such that \( P_{ij} = (x_{ij}, y_{ij}) \) and \( B_i(u) \) and \( B_j(v) \) are the uniform bicubic B-spline basis functions. We question the amount that one can translate control point \( P_{ij} \) such that the surface \( S(u, v) \) remains injective. Let \( V_{ij} = (v_x, v_y) \) be the translation amount that is locally prescribed by the depth of the Z-buffer. Then, the new location of \( P_{ij}, \hat{P}_{ij} \), equals,

\[
\hat{P}_{ij} = (x_{ij}, \hat{y}_{ij}) = (x_{ij} + \lambda v_x, x_{ij} + \lambda v_y),
\]

for some \( 1 \geq \lambda \geq 0 \). \( \lambda \) equals one if the new prescribed translation do indeed preserve the injectivity condition, and \( \lambda \) is between zero and one, otherwise.

Reexamining the \( c_{ij} \) coefficients in Equation (6), one can show (see side-bar on the coefficients of the Bézier patch) that no term of the form \( x_{kl} y_{kl} \) exist in \( c_{ij} \), as is evident, for example, in Equations (7). This property becomes significant when one substitutes \( \hat{P}_{ij} \) in, as the expression of \( c_{ij} \) remains linear in \( \lambda \), making it simple to solve for \( \lambda_0 \) that satisfies \( c_{ij}(\lambda) = 0 \). Assume, without lose of generality that \( c_{ij}(\lambda)|_{\lambda=0} > 0, 0 \leq i, j \leq 5 \). Then, having 36 constraints of similar form, \( c_{ij}(\lambda) = 0, 0 \leq i, j \leq 5, \) one is required to acquire the minimal positive real \( \lambda \) value from all these 36 equations. Alternatively, and given \( 0 < \lambda < 1 \), if exists \( c_{ij}(\lambda) < 0 \), one can conduct a binary search on \( \lambda \) between zero and \( \lambda \), examining all 36 constraints, rapidly converging to the allowable step size.

An interior control point \( P_{ij} \) of \( S(u, v) \) participates in 16 adjacent patches. A boundary, or close to a boundary, control point will participate in less. Therefore, the above test for a \( \lambda \) that preserves the injectivity condition, should be exercised in all (up to 16) patches that employ \( P_{ij} \) and again, the minimal positive real \( \lambda \) that results should be used.

Algorithm 2 presents the proposed rendering process. Starting with a prescribed scene that is rendered into a Z-buffer and a prescribed uniform bicubic B-spline surface that is injective, the surface is rotated to the desired orientation as prescribed by some angle \( \beta \) and a translation vector is constructed in the plane in an orthogonal direction, \( \beta + 90 \). Then, an iterative process automatically computes an injective warping of \( S(u, v) \) that would yield the necessary mapping, \( \mathcal{M} \), subject to an intensity control, \( I \), that affects the translation amount.

The convergence of the algorithm can be determined by the difference between the translation amount of each control point and (some function of) the depth at that location. In addition, one can put a bound on the number of allowed iterations.
The Coefficients of the Bézier Patch

**Proposition** The function

\[ f(u, v) = \sum_{p=0}^{3} \sum_{q=0}^{3} x_{p,q} B_p^3(u) B_q^3(v) \sum_{r=0}^{3} \sum_{t=0}^{3} y_{r,t} B_r^3(u) B_t^3(v) - \sum_{p=0}^{3} \sum_{q=0}^{3} y_{p,q} B_p^3(u) B_q^3(v) \sum_{r=0}^{3} \sum_{t=0}^{3} x_{r,t} B_r^3(u) B_t^3(v), \]

where \( B_p^3(t) \) denotes the uniform cubic B-spline basis functions, as in Equation (5), has no factor in the form of \( x_{ij} \) \( y_{ij} \).

**Proof:** Without loss of generality, let the uniform knot sequence be \( t_i = i \). Using the derivatives of B-spline functions, one gets,

\[ f(u, v) = \sum_{p=0}^{2} \sum_{q=0}^{3} (x_{p+1,q} - x_{pq}) B_p^2(u) B_q^3(v) \sum_{r=0}^{2} \sum_{t=0}^{3} (y_{r,t+1} - y_{rt}) B_r^3(u) B_t^2(v), \]

Examining the term in the square brackets inside the summation, for \( p = r \) and \( q = t \),

\[ (x_{p+1,q} - x_{pq})(y_{pq+1} - y_{pq}) - (x_{p+1,q} - x_{pq})(x_{pq+1} - x_{pq}) \]

or the factor of \( x_{pq} y_{pq} \) cancels out with no other factors in the form of \( x_{ij} y_{ij} \). A similar examination of this term inside the summation for the cases of \( p = r + 1 \) and/or \( q = t + 1 \) would yield similar results.

Now, because \( f(u, v) \) has no factor in the form of \( x_{ij} y_{ij} \), any linear combination of the coefficients of \( f(u, v) \) will also be free of such factors. In particular the linear combination that computes the conversion of the bi quintic function of \( f(u, v) \) into a Bézier function.

### 4 Examples

We start with a three-dimensional model of Beethoven rendered into parallel stripes in Figure 4 (a) and (b). Another example, of the well known Utah Teapot, is presented in Figure 4 (c) and (d). The entries in the (a) and (b) pair and in the (c) and (d) pair differ from each other in the intensity of the warping function, \( I \). This influence of the intensity factor on the resulting image, \( I \), is also demonstrated in Figure 5.

In most presented examples, the Z-buffer has been initialized to a depth of zero and then the scene was rendered above it, in the \( +Z \) direction. An additional example of this approach is shown in Figure 6 (a). Nevertheless, in Figure 6 (b), the Z-buffer is reversed. That is, the Z-buffer was similarly initialized to a depth of zero but rendering was conducted in the \( -Z \) direction. The result is a translation field in the reciprocal direction that yielded Figure 6 (b). It is quite surprising that the
Algorithm 2

Input:

- $S$, a scene to be rendered as a Vasarely style drawing;
- $S(u, v)$, a prescribed injective uniform bicubic B-spline surface to warp;
- $\beta$, orientation angle with respect to the $x$ axis, of the stripes;
- $I$, intensity factor;

Output:

- $\mathcal{L}$, a set of parallel stripes conveying scene $S$;

Begin

- $\mathcal{Z} \leftarrow$ Rendered $Z$-buffer depth map of $S$;
- $S(u, v) \leftarrow S(u, v)$ rotated $\beta$ degrees in the XY plane;
- $\mathcal{V} \leftarrow (0, 1)$ rotated $(\beta + 90)$ degrees in the XY plane;

While not converging Do

For all control points $P_{ij} = (x_{ij}, y_{ij})$ of $S(u, v)$ Do;

- $V_{ij} \leftarrow \mathcal{V}$ interpolated $Z(x_{ij}, y_{ij})$;
- $\lambda \leftarrow$ Maximal translation allowed while $S(u, v)$ is injective, $0 \leq \lambda \leq 1$,
  using Algorithm 1;

- $P_{ij} \leftarrow P_{ij} + \lambda V_{ij}$;

Od

Od

- $\mathcal{L} \leftarrow$ Set of isoparametric curves of $S(u, v)$ at a prescribed density and width;

End

Figure 4: Stripes rendering of a three dimensional model of Beethoven ((a) and (b)) and of the Utah Teapot ((c) and (d)). The entries in each pair differ from each other in the intensity of the warping.
mind still attempts to capture the scene in Figure 6 (b) as a protrusion \(^1\). Being able to render the stripes in arbitrary orientation, one can clearly merge two or more such renderings into a fabric of stripes, possibly even of different styles (density, width, gray level, etc.). In Figure 7, two such renderings of a pawn chess piece have been created at different orientations (Figures 7 (a) and (b)), only to be merged in Figure 7 (c).

An example of stripes rendering of a chess piece of a queen at different resolutions of the B-spline surface is presented in Figure 8. Three different mesh sizes are employed for the underlying B-spline surface. The denser the control mesh, the sharper the resulting drawing will be and vice versa.

The control over the density of the stripes and their width can also affect the drawing. One example of such an effect is shown in Figure 9. The only difference between Figure 9 (a) and Figure 9 (b) can be found in the number of extracted isoparametric curves and their width. The underlying warped B-spline surface is identical in both cases.

Since the local translation of the stripes is mostly controlled by the values presented by the Z-buffer, one can apply a filter to the Z-buffer before the rendering process takes place. Borrowing ideas from [Sait90], in Figure 10, first and second order derivative operators have been applied to the Z-buffer before the rendering. It is interesting to note the (not so appealing) quality of the resulting illusion drawings, compared to all the other, regular Z-buffer renderings.

\(^1\)at least the author’s mind.
Figure 6: Self portrait. In (a), we look out of the page, whereas in (b), we look into the page.

Figure 7: Stripes rendering of a three dimensional model of a pawn chess piece. (a) and (b) shows two renderings with different angles, density and thickness. (c) presents a merge of the two.
Figure 8: Three Stripes renderings of a three dimensional model of a chess piece of a queen. In (a), the mesh size of the bicubic B-spline surface is $300 \times 300$, in (b) the mesh size is $200 \times 200$ whereas in (c) the mesh size is $100 \times 100$.

Figure 9: Two stripes renderings of a three dimensional model of a cow at different line width and density. The underlying warped B-spline surface is identical in both cases.
Rendering with Parallel Stripes

Figure 10: Two Stripes renderings of a pawn chess piece, after a first ((a) and (b)) and second ((c) and (d)) derivative operator has been applied to the Z-buffer. The entries in each pair differ from each other in the intensity of the warping. Compare with Figure 7 (a).

So far, the underlying B-spline surface was in the form of a uniform grid but need not be. In Figures 11 and 12, some already presented scenes are rendered over a disk-like and wiggle-like underlying surface, have the potential secondary effect of a background with varying shades.

Finally, one can compose several such stripes renderings as an overlapping process, applying a sequel of Z-maps to the B-spline surface. Two results of such an attempt are portrayed in Figure 13.

All the stripes renderings presented in this work were created using a modern workstation in few seconds to several minutes. In most presented drawings, fifty iterations were employed during the warping function derivation before convergence was declared.

5 Conclusion

We have presented a stable and automatic scheme to create stripes renderings that mimic the style of the “Zebra” drawing of Victor Vasarely.

In this work, we have selected to employ bicubic B-spline surfaces due to the $C^2$ continuity that they offer. The approach taken to derive the one-to-one conditions of the mapping, $\mathcal{M}$, could clearly apply to arbitrary orders and/or non uniform B-spline surfaces as well, at the cost of more intensive computation. Higher order surfaces might result in smoother stripes rendering yet would also lose details. Moreover, the injective condition presented herein could also serve in other applications such as image morphing. In fact, and following a comment of one of the reviewer of this paper, it would be interesting to try the application of the computed warping in the plane to actual images, with the possible hope that the resulting warped image will better convey three dimensional information.
Figure 11: Stripes rendering of a three dimensional model of Beethoven over a wiggle-like (a) and disk-like (b) underlying surface. Compare with Figure 4.

Figure 12: Stripes rendering of a three dimensional model of a cow over a wiggle-like (a) and disk-like (b)-like underlying surface. Compare with Figure 9.
Figure 13: One can compose several Z-maps in a sequences into a one stripes rendering. In (a), several cows are portrayed whereas in (b), few pawn chess pieces are presented.

In this work, we have offered one way to implement this drawing style of rendering with stripes. One could consider other schemes such as structured lighting [Bala82], yet stripes that are free of intersections are much more difficult to achieve. Examining the vertical walls in the Z-map of, for example, figures 5, 7 and 8, any method that is based on structured lighting would have hard time figuring a proper illumination direction that will result in intersection free stripes, and if such direction exists at all. Having the benefit of an injective mapping, non of the adjacent isoparametric curves could locally cross each other.

Being discrete, the employed Z-buffer introduced jagged curves in the presented drawings. Two ways could be considered to reduce this phenomena. First, one could consider Z-buffers with sufficient resolution that would reduce this effect to an acceptable level. A second approach could employ low pass filtering on the output curves smoothing out these noisy details.

Due to the nature of the drawings, the highly dense curves could result in Moire [Fole90] patterns if the curves are not drawn at the proper quality. For example, some Moire patterns could be observed in the middle of Figure 12 (a).

This work has presented a simple nonphotorealistic rendering scheme that could not only be intermixed with other rendering schemes but also could spawn a whole set of artistic varieties. Such varieties might include different line styles, use of colors, exploitation of different basic primitive shapes other than lines and circles, etc. The full potential of this rendering scheme and its evolutionary path for
and in the computer graphics community will be revealed, in the coming years.

6 Acknowledgment

The models of beethoven and the cow, were donated to the public domain by Viewpoint [Viewpoint]. The self portrait was scanned with the aid of Gil Zigelman, Technion.

My search after Vasarely’s drawings had tremendous aid from Freddy Bruckstein, Dan Gordon, George Wolberg, Bob Shakib, and Michele Vasarely. I thank them all.

This research was supported in part by the Fund for the Promotion of Research at The Technion, Haifa, Israel.

References


