

Artificial Texturing: An Aid to Surface Visualization *

Dino Schweitzer

Computer Science Department
University of Utah
Salt Lake City, Utah 84112

Abstract

Texture is an important surface characteristic which provides a great deal of information about the nature of a surface, and several computationally complex algorithms have been implemented to replicate realistic textures in computer shaded images. Perceptual psychologists have recognized the importance of surface texture as a cue to space perception, and have attempted to delineate which factors provide primary shape information. A rendering algorithm is presented which uses these factors to produce a texture specifically designed to aid visualization. Since the texture is not attempting to replicate an actual texture pattern, it is called "artificial" texturing. This algorithm avoids the computational limitations of other methods because this "artificial" approach does not require complex mappings from a defined texture space to the transformed image to be rendered. A simple filtering method is presented to avoid unacceptable aliasing.

CR Categories and Subject Descriptors: I.3.3 [Computer Graphics]: Picture/Image Generation – display algorithms; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism – color, shading, shadowing and texture

General Terms: Algorithms

* This work was supported in part by the National Science Foundation (MCS-8004116 and MCS-8121750), the U.S. Army Research Office (DAAG29-81K-0111 and DAAG29-82-K-0176), and the Office of Naval Research (N00014-82-K-0351).

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

© ACM 0-89791-109-1/83/007/0023 \$00.75

1. Introduction

Almost all physical surfaces contain some microstructure or texture visible to the human eye. This texture gives us information such as the type of material which composes the object and the relative smoothness or coarseness of the surface. Computer generated images which portray such texture provide the same types of information and appear extremely realistic. Unfortunately, the algorithms for generating such images are generally time consuming and require knowledge of how to map the texture onto the surface.

Perceptual psychologists have analyzed a different type of information provided by texture: that of space perception [7] [3] [8]. The microstructure of a surface provides a somewhat regular pattern for visualization. The changes in this pattern, or texture gradient, give strong cues as to the orientation and depth of the surface. Extensive studies and experiments have shown that there are three characteristics of texture which provide this perceptual information: size, shape, and density. Changes in these components due to standard perspective and projective transformations provide knowledge about surface depth and changes in the orientation of the surface.

To avoid confusion in terminology, "texture" in this paper will not refer to the intuitive meaning of changes in the normal direction due to the surface microstructure. Rather, it is the pattern on the surface which is of interest. This pattern may be caused by different colors as in a tiled floor, or different intensities because of changing normals such as the skin of an orange. As will be discussed, it is the changes in this pattern which provide a basis for perception.

This paper describes a texturing technique which approximates the texture changes caused by distance from the viewer and orientation of a surface without attempting to exactly render a realistic texture. This approach supplies several of the same visualization cues as provided by exact texture rendering algorithms without the complexity of generating a realistic texture. Thus, "artificial" texturing is a means for providing an inexpensive aid to visualizing the shape of a shaded surface.

2. Background

To understand the objectives and advantages of this approach, some background will be provided on the role of textures in perception and conventional algorithms to simulate such texture.

2.1. Texture in Perception

Visual perception has been studied by different groups over many centuries. Early artists were interested in making paintings appear realistic, and thus developed the concepts of perspective, shading, and several other "cues" which could be used to imply depth. Physiologists and psychologists have studied vision and perception in an attempt to understand the workings of the eye and the brain. The results of these studies have been applied to computer generated images producing very realistic results.

The classic paradox in the study of perception is how one perceives depth from a two-dimensional retinal image. Cue theory attempts to explain this dilemma by suggesting that it is additional pieces of information, or cues, which when combined with the flat image provide the depth knowledge. Traditional primary cues were attributed to the physiological effects necessary to focus on an object while the cues used by painters such as perspective, occlusion, and relative size were termed secondary [3]. More recently, texture and movement have been studied as important visual cues [7].

Surface texture provides a fairly regular pattern over a large surface. For example, the size of individual blades of grass in a field may be random, but taken as a whole, creates a constant textural pattern. There are also textures which are created regularly such as bricks in a wall or a checkerboard. As a surface is slanted away from the eye the texture pattern gradually changes due to perspective and projection. This gradual change is called the texture gradient [8]. Perception researchers isolate three separate gradients which influence perception: texture size, shape, and density. Several experiments have shown that each of these gradients strongly influence the perception of surface depth and slant [3]. Figure 2-1 is a simple example of a textured surface, and illustrates the effects of texture size, shape, and density to perception.

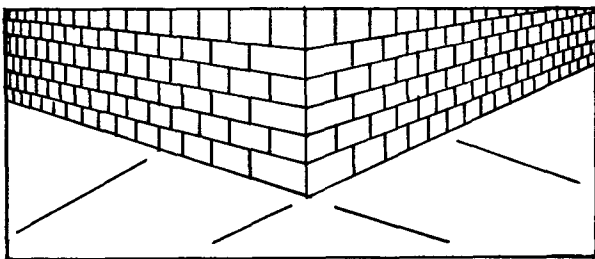


Figure 2-1: Simple texture

2.2. Conventional Algorithms

In an ever increasing struggle for realism, computer graphics researchers have developed different approaches to modeling surface texture.

Catmull suggested one of the earliest techniques for mapping a defined texture onto subdivided patches [4]. His rendering algorithm subdivided parametric patches down to the pixel level for display. As a patch was subdivided, the parametric boundaries of each subpatch were carried along down to the pixel. Once at the pixel level, the (u,v) coordinates of the pixel corners specified an area in a texture definition array defined in (u,v) space. The color of the pixel was determined by averaging the values in this texture definition area.

Blinn and Newell took Catmull's technique a step further by introducing a more sophisticated filtering method [1]. The quadrilateral formed in texture definition space by the (u,v) corners of the pixel was used as the base for a square pyramid to weight the texture values. Their paper also discussed different techniques for creating texture patterns such as (u,v) functions, hand drawings, and scanned photographs.

Blinn also suggested a more novel approach to creating actual bumps in the surface via normal perturbation [2]. In this method, a perturbation function was mapped onto the surface to modify the normal vectors giving the illusion of wrinkles. This technique produced extremely realistic images, but at about twice the time of standard texture mapping.

Because of the amount of detail involved, textured images are extremely vulnerable to aliasing. More recent approaches have concentrated on the filtering aspect of the texture mapping [5] [10].

In each approach to creating texture the same two general functions must be performed: some sort of mapping is required from texture space to image space, and filtering is necessary to create acceptable images. Both of these increase the computation time of image rendering. Additionally, the mapping procedure is extremely data texture dependent, and is not always easy to define. For example, if one had a collection of polygons representing a surface and wanted to simulate bricks, it is not obvious how to map each polygon's coordinates to the brick texture definition. Although both of these functions are necessary to create realistic texturing, a simpler approach can be used to obtain the perceptual benefits of texture: artificial texturing.

3. Artificial Texturing

Artificial texturing is a means of applying a pattern to a surface in order to obtain the perceptual benefits of texture gradients. The goal of this approach is not to produce realistic textured images, but rather to aid the visualization of rendered surfaces. No knowledge of the type of object space data is required, but it is assumed that depth and normal information is available at each pixel as is required for z -buffer hidden surface rendering with normal interpolation for shading. Each of the perceptual characteristics of texture discussed earlier will be presented with a look at the information required to generate each.

3.1. Texture Size

Psychological experiments have shown that the texture pattern most effective in aiding perception is one of equally sized elements positioned on the surface at a regular density [3]. An example would be a set of equally spaced grid lines. As mentioned earlier, the problem with projecting such lines on an arbitrarily warped surface is that some mapping knowledge is required to know how such lines continue from surface element to surface element. However, a type of pattern which avoids this problem is one composed of individual texture elements, or "texels", such as a pattern of disjoint squares on a piece of cloth. The same texture gradients apply to individual texels: their changing size, shape, and density.

Along with the advantage of not requiring mapping knowledge, the use of individual texels also lends itself to a simplifying assumption: that each texel is only a function of the center of the individual element. Although this assumption does not imitate realism, it allows for a significant decrease in computation time. This simplification will be used in the calculation of all three texture gradients.

The size of each individual texel is strictly a function of the depth of the element from the projecting plane. The idea of perspective projections is well understood in the field of computer graphics and has been addressed by many authors [6]. Using the same simplifying assumption as previously stated, the size of the texel can be determined as a function of the depth of the center of the element using a standard perspective transformation.

3.2. Texture Shape

When equally shaped texels are used, the shape gradient is a strong cue as to the surface orientation [11]. The shape will be a function of the surface orientation over the texel as well as the orientation of the texel itself. For example, square elements will not necessarily be aligned with all edges parallel, and would appear unnatural if so forced. To avoid this problem, symmetric texels are used in this algorithm, specifically circles, or polka dots.

The shape as a function of surface orientation can be simplified using the same assumption as before, that it is strictly based on the orientation of the element's center. This will not result in exact realism since individual texels will not follow the surface curvature over the area of the element. However, as a visualization aid, this assumption allows for a simple texture generating method which produces perceptually helpful gradients. The shape of individual polka dots on the surface can be determined by projecting a circle perpendicular to the normal at the circle's center onto the display screen. Figure 3-1 shows a graphic description of this projection. The projection of the circle will be an ellipse whose eccentricity depends on the direction of the normal.

3.3. Texture Density

Another texture gradient providing perceptual information is the density gradient. When the texture pattern has regular density, the changes in the

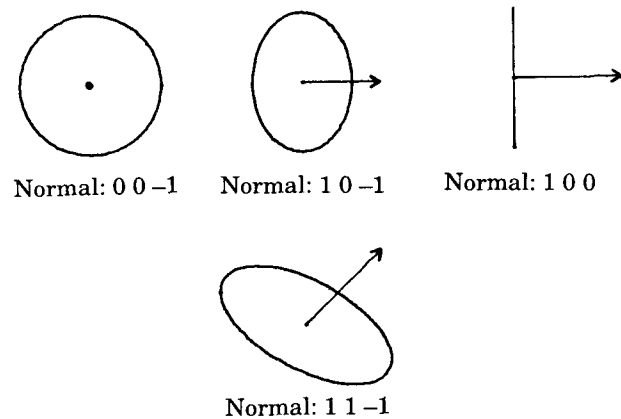


Figure 3-1: Circle projections for different normals

distance between elements, or density gradient, on the textured surface give clues to the depth and orientation of points on the surface. Researchers have distinguished two types of density gradients, compression and convergence [3]. Compression is a decrease in distance between texels due to the perspective transformation of elements at a greater depth from the viewer. Convergence is a result of elements being projected closer together when the surface is at a slant to the viewer. Thus, compression is a function of depth while convergence is based on normal information. As in the other two gradients, the approximation of using only the element's center simplifies the texture density calculations at any given point on the surface.

Algorithmically, the density information is required to determine where to put each texel. If only a density change is desired in either the horizontal or vertical direction, a simple solution is possible. The normal information available at each pixel during rendering can be used to estimate the surface distance between pixel centers. For a given density, the rendering process can determine where the next element goes by accumulating the distances across a scan line. Similarly, images could be scanned in a vertical fashion to create a vertical density change. Although either of these density techniques may provide perceptive information, the images they produce appear awkward and off balance.

To produce density information in both directions requires more detailed computation than simple horizontal or vertical distances. One solution is to approximate areas covered by each pixel's boundaries rather than distances between pixel centers. For a given normal at the center of each pixel, an approximation can be derived to the area bounded by the projection of the pixel boundaries onto the surface. This approximation can be modified by a perspective factor based on the depth of the surface at the pixel center. For a specified texture density, the placement of texels can be determined by summing surrounding pixel areas until enough area is found to place one element. Because the area calculation encompasses both depth and normal information, both compression and convergence density results. Figure 3-2 illustrates the difference in images using the variable density calculation.

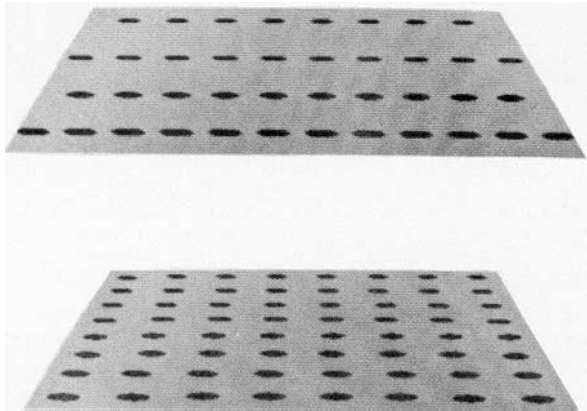


Figure 3-2: Static versus variable density

4. Implementation

The implemented version of artificial texturing uses a modified z-buffer image file as input. Rather than the standard r, g, b components at each pixel, the input z-buffer contains normal and intensity information. This data is readily available to the rendering program and can be encoded into the same size fields for compatibility. For this implementation, input was obtained from a standard rendering program which contained a modified shading routine to return spherical normal coordinates in the r and g values, and an intensity in the b. It requires no additional computation time to generate such input files beyond normal rendering time.

4.1. Texel Projection

The shape of each texel is determined by projecting pixels surrounding the texel center onto the plane defined by the center normal. The distance from the projected point to the texel center is compared to the element radius to decide which pixels lie within the projected texel. Given a normal (N_x, N_y, N_z) and point (P_x, P_y) relative to the texel center, the equation for the z-component of the projected point, P_z , and distance to the texel center, D , are:

$$P_z = (P_x N_x + P_y N_y) / N_z$$

$$D = (P_x^2 + P_y^2 + P_z^2)^{1/2}$$

The texel radius is specified by the user and modified by the z depth of the texel center to reflect changes in the element's size due to perspective as described earlier.

A simple filtering mechanism is incorporated during texel projection. For each surrounding pixel, the corners of the pixel are projected and compared rather than the pixel centers. Thus, each surrounding pixel will have an associated count of 0 - 4 corners inside of the projected texel. This count can be used to

weight the color of the texel with the surface to provide a simple filter. The weighted color is then modified by the intensity at each pixel. This is important to ensure that dots appearing in darker shaded portions of a surface are similarly darkened. Figure 4-1 shows the different shapes and sizes of projected dots caused from a variety of normals at different depths.

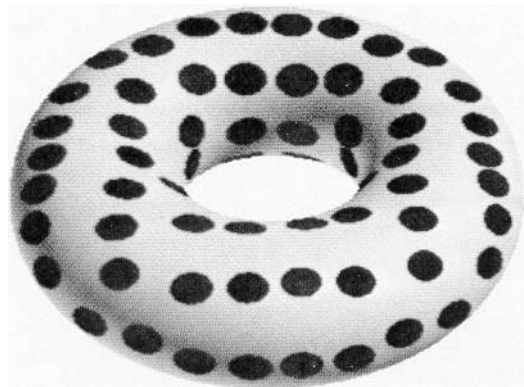


Figure 4-1: A textured torus

4.2. Texel Density

The algorithm as described could be implemented using a simple density function for texel positions such as placing them at equal x, y screen intervals. Although this provides size and shape information, it tends to appear awkward as in Figure 3-2, because the density is not changing as our intuition tells us it should. To provide for the density gradient the input file is preprocessed to determine positions for texel centers. As described earlier, this involves computing areas under each pixel in the image. To approximate these values the area cosine principle is used to determine the area on the surface bounded by the projected pixel. This approximation assumes that the surface bounded by the projected pixel is a plane defined by the normal at the pixel. The equation for this area is:

$$\text{Surface Area} = \text{Pixel Area} / \cos(\text{pa})$$

where pa is the angle between the pixel plane and surface plane. The cosine is determined by taking the dot product of the surface normal with the screen normal and is simply the z-component of the surface normal, N_z . Thus,

$$\text{Surface Area} = 1 / N_z$$

To account for perspective, this area is further modified by the z depth of the sampled point similar to the texel radius.

Once individual pixel areas are found, it is necessary to group them into cohesive sections of equal area. This implementation uses a simple approach to this grouping by building an "area binary tree" from the individual areas. At the top level is the bounding box for the entire image. This image is divided into two subareas by a horizontal or vertical line with approximately equal areas on either side of the line. Each of these subareas are further subdivided into two equal segments, and so forth down to some specified level. At each level n , 2^n rectangular sections are defined of approximately equal areas. The direction of division for each area is determined by the longest side of the bounding box in an attempt to keep the subareas relatively square. The area center of each rectangle can be determined by weighting the pixel locations within the rectangle by their individual areas. This area center is stored as a texel location for the density defined by this level. Figure 5-2 shows a level of bounding boxes and texel centers for the surface in figure 5-1. Using these texel centers, figure 5-3 shows the resulting textured image.

The z-buffer input file is preprocessed to build an "area binary tree" of several levels (usually around 6). The resulting tree consists solely of texel locations for each level and becomes an input to the artificial texture rendering program. The user specifies a desired density level, and the appropriate level of the tree is loaded which specifies texel locations to be projected. Figure 5-4 shows the difference between static and variable density at different levels of the area binary tree. The upper two cylinders have a static density (texel centers equally spaced in x and y directions in screen space.) The lower cylinders were textured using different levels of the area binary tree, and do not demonstrate the "gaps" along the edges apparent in the upper cylinders.

5. Conclusion

Artificial texturing provides simple visualization enhancement at a low computation cost. Figures 5-1 and 5-3 are an example of the information available using artificial texturing. The lower left portion of the shaded image in figure 5-1 contains very little shading information to help interpret the shape of the surface in that area. It appears from the shading to be relatively flat. The artificially textured image, figure 5-3, gives a visual impression of the curvature of the surface.

The information provided can be compared to the "hedgehog" technique (displaying normal vectors attached to the surface) [9]. In fact, projected polka dots can be viewed as sort of "flattened hedgehogs". However, they provide a much more natural means of visualizing shape information since textures are interpreted in our everyday visual experiences. Also, additional information, such as depth, is more readily interpreted from texture gradients. One enhancement to artificial texturing which was found to be effective is to combine hedgehogs and texture. Projected polka dots are ambiguous in that they can represent one of two possible normals. By overlaying hedgehogs, the ambiguity is reconciled. Figures 5-5 through 5-8 show a warped surface with hedgehogs, texture, and both. Although the combined picture may not appear natural, it is easier to interpret than strictly

hedgehogs and resolves the ambiguity of the texture elements.

Because of the number of assumptions and simplifications made, textures are sometimes generated which do not appear realistic. For example, on a surface with a sharp corner, if a texel is centered close to the edge of the corner, it will not "bend" around the corner as would seem natural. Another example is that because of the simple means of generating regions of equal surface area, densities at times will not look aligned exactly as they should. However, as stated at the outset of this paper, the goal is not extreme realism, but to simulate the texture gradients which provide strong perceptive information. To this goal, artificial texturing provides a fast, inexpensive approach to enhance surface visualization.

References

1. Blinn, James F. and Newell, Martin E. "Texture and Reflection in Computer Generated Images", *Communications of the ACM*, Vol. 19, No. 10, October 1976, pp. 542-547.
2. Blinn, J. F. "Simulation of Wrinkled Surfaces", *Computer Graphics*, Vol. 12, No. 3, August 1978, pp. 286-292.
3. Braunstein, M. L. *Depth Perception through Motion*, Academic Press, New York, 1976.
4. Catmull, E. "A Subdivision Algorithm for Computer Display of Curved Surfaces", Tech. Report No. UTEC-CSc-74-133, University of Utah, December 1974.
5. Feibush, E. A., Levoy, M., and Cook, R. L. "Synthetic Texturing using Digital Filters", *Computer Graphics*, Vol. 14, No. 3, July 1980, pp. 294-301.
6. Foley, J. D. and Van Dam, A. *Fundamentals of Interactive Computer Graphics*, Addison-Wesley, Reading, Mass., 1982.
7. Gibson, J. J. . *The Perception of the Visual World*, The Riverside Press, Cambridge, Mass, 1950.
8. Haber, Ralph N. and Henderson, Maurice. *The Psychology of Visual Perception*, 2nd ed., Holt, Rhinehart and Winston, New York, 1980.
9. Newman, William F. and Sproull, Robert F. *Principles of Interactive Computer Programming*, 2nd ed., McGraw-Hill, New York, 1979.
10. Norton, A., Rockwood, A. P., and Skolmoski, P. T. "Clamping: A Method of Antialiasing Textured Surfaces by Bandwidth Limiting in Object Space", *Computer Graphics*, Vol. 16, No. 3, July 1982, pp. 1-8.
11. Phillips, R. J. "Stationary Visual Texture and the Estimation of Slant Angle", *Quarterly Journal of Experimental Psychology*, Vol. , No. 22, 1970, pp. 389-397.

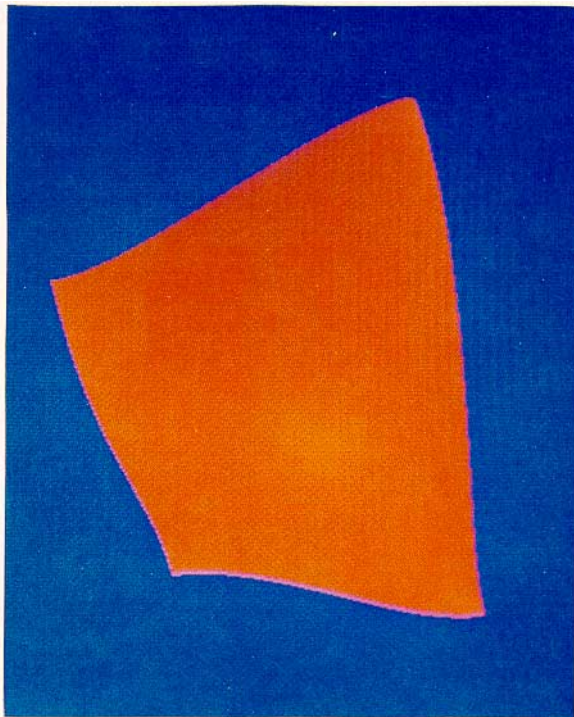


Figure 5-1: Arbitrarily warped surface

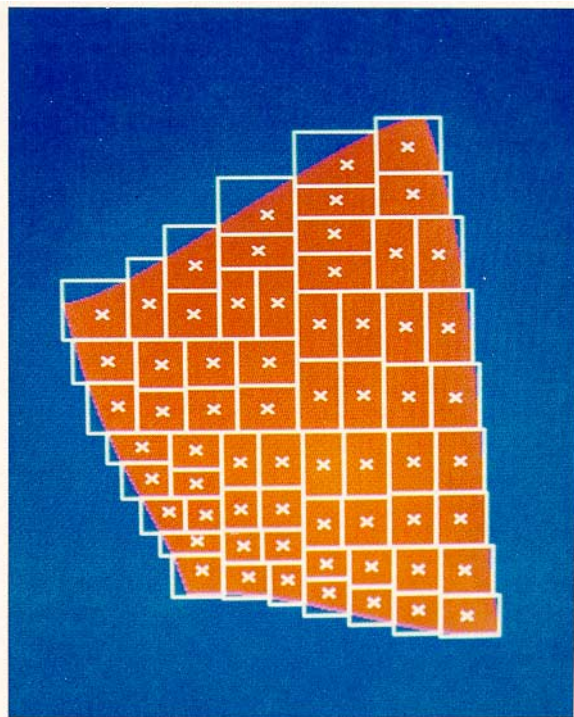


Figure 5-2: Equal area decomposition

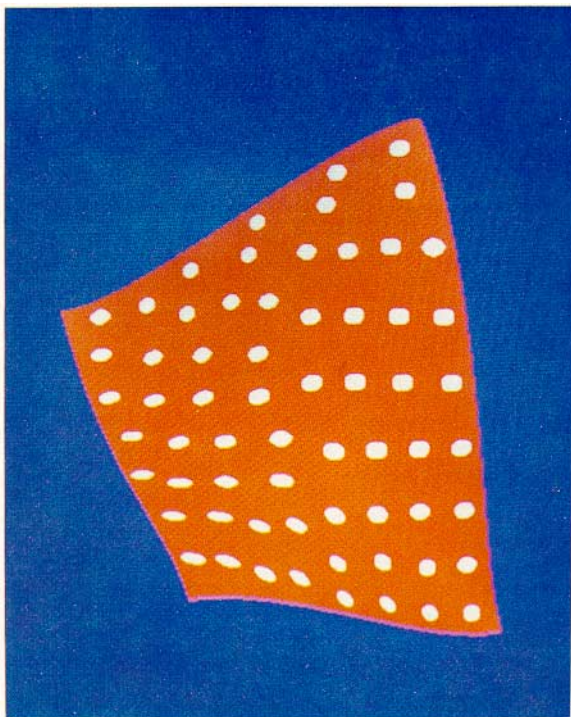


Figure 5-3: Surface with artificial texture

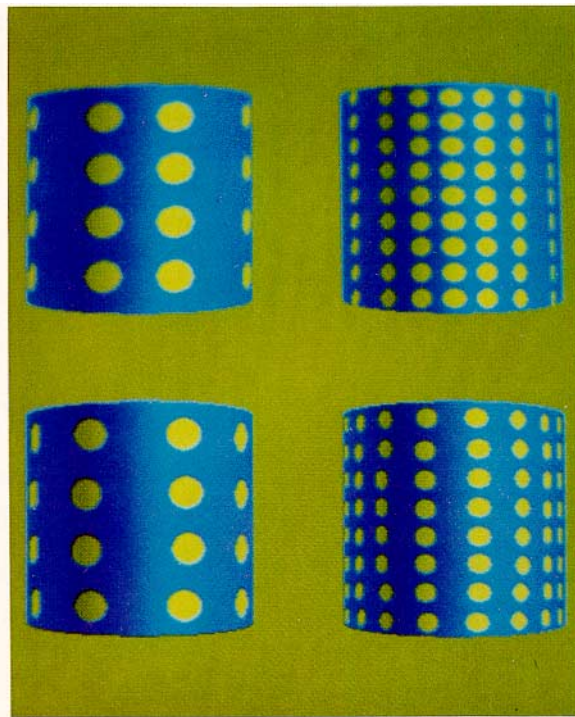


Figure 5-4: Different static and variable densities

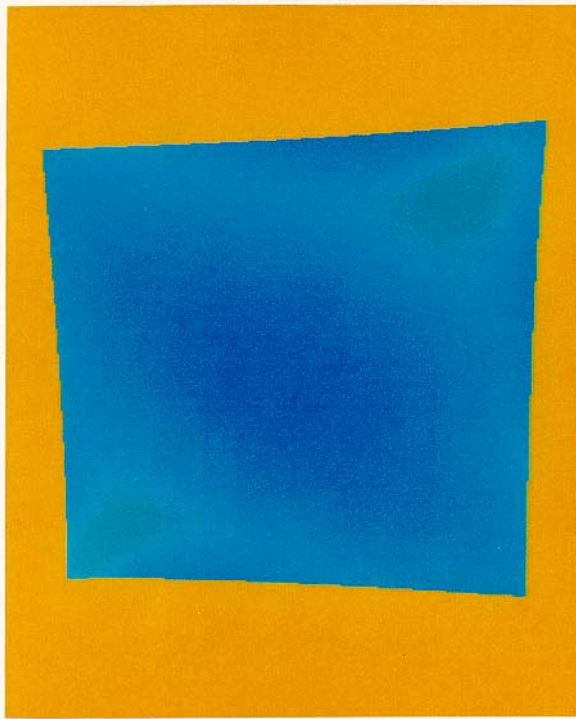


Figure 5-5: Shaded surface



Figure 5-6: Surface with hedgehogs

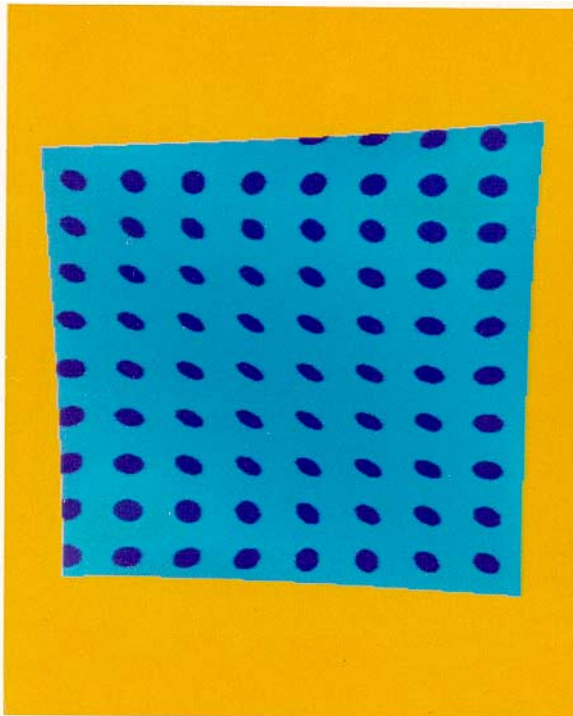


Figure 5-7: Surface with artificial texture

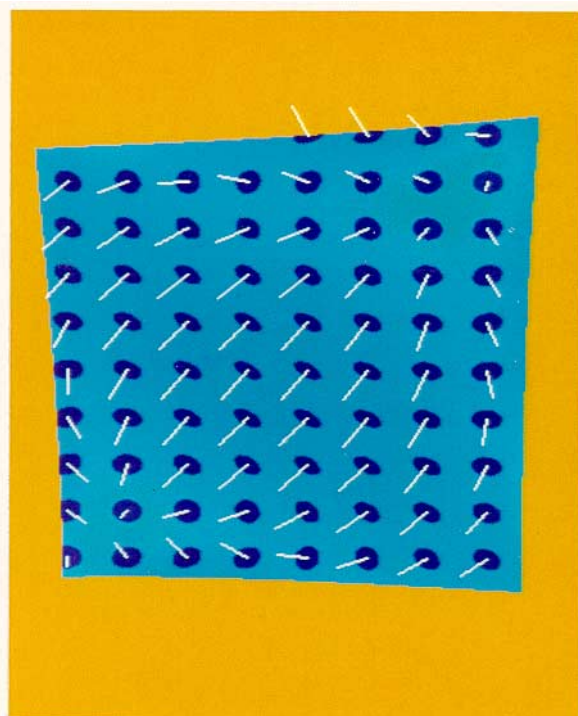


Figure 5-8: Combined hedgehogs and texture