Solid Texturing of Soft Objects

by

Geoff Wyvill
University of Otago
New Zealand

Brian Wyvill, Andrew Pearce and Craig McPheeters
University of Calgary
Alberta, Canada

Abstract

Since the shape of a Soft object changes in response to its surroundings, it is difficult to give a single position in space as the location of the object. Indeed objects can and do break into sub-objects dynamically. This means that you cannot map a solid texture onto such an object simply using a function of the space co-ordinates.

We have taken a different approach. Our soft objects are modelled as the volume enclosed by an iso-surface of a field calculated from a set of key points. We ascribe to each key point a set of values which represent a *position* in an abstract *texture space*. Any point on the surface of an object has a field value due to each key point and this value is used as a weight in finding a weighted vector sum of these *positions*. This vector sum is used to select a surface specification from the *texture space*.

These textures retain their consistency during distortion and metamorphoses of objects. A great variety of animation effects can be achieved with this process.

Keywords: Soft Objects, Texture Mapping, Solid texturing, Animation, In-betweening.

Introduction

The last five years has seen great progress in representing natural objects in computer-generated pictures. Two techniques are of particular significance: texture mapping and soft objects. Texture mapping (Blinn 1976, Blinn 1978 and Peachey 1985) has enabled us to represent rich surface details with otherwise simple models. Soft object modelling (Wyvill 1986b, Blinn 1982, Nishimura 1985) has provided a simple way to represent objects whose shape changes with time.

We want to represent objects which are both soft and textured. A good example is concrete being poured where small stones or grains of sand are visible on the surface of a mass which is changing shape. This task introduces some special problems and seemingly self-contradictory requirements.

To explain these problems we give a brief description of texture mapping and soft object modelling. This is followed by a description of our technique, some examples and a short discussion.

Texture mapping

The fundamental notion behind texture mapping is that we can separate the description of an object into two parts. The first describes the major aspects of the shape and the second describes a refinement in terms of surface detail.

The reasons for the success of this separation are variously physical, perceptual and technical. The physical justification is that many objects are literally constructed in this way. A painted object has its surface appearance uniformly modified by the layer of paint; an otherwise flat water surface is disturbed by surface ripples and so on. The perceptual aspect is connected with the way in which we deduce surface detail from the visual appearance. At a distance, a slightly wavy or bumpy surface is visible only because of the perceived colour or shade differences produced by the bumps. The technical reason is that we already have efficient ways to model and render objects of smooth, simple shapes and texture mapping allows us to extend this range of objects at low computational cost.

The description of surface detail is called a texture map and it can be represented by mathematical functions or tables. In the process of texture mapping we must relate each visible point on the surface of an object to a point in the texture map. A "point" for this purpose, will correspond to a pixel in the final image. The general method of doing this requires that we set up a co-ordinate system on the object's surface. The method used to construct such a co-ordinate system depends on the way the object is defined. If parametric patches have been used then each surface element is already defined as a function of some s, t and the same s, t make convenient co-ordinates for texture mapping. If the surface is defined by polygons, the simplest approach is to divide the polygons into rectangles or triangles and then treat these as a special case of patches. Continuity at patch borders is achieved by careful definition of the texture map, using complementary co-ordinates (1-s, 1-t) in adjacent patches and other methods.

Peachey (1985) introduced the idea of using 3D co-ordinates of surface points for texture mapping. This is equivalent to carving the object out of a non-uniform substance represented by the 3D texture map. This is an excellent approach for modelling materials like wood and marble where the surface is indeed the result of carving from a material patterned in 3D. Where this approach can be used, there are advantages over the 2D method. Consistency of texture is achieved regardless of the topology of the object. For most rendering techniques, the 3D coordinate values which correspond to a particular screen pixel have to be calculated anyway, so the co-ordinates for texture mapping are available at no extra computational cost.

There are, however, cases where 3D texture mapping is unsuitable: surface marks on machined objects, patterns on textiles and applied patterns like paintwork. In these examples the texture of the object we are modelling is not a surface manifestation of a three dimensional structure so we should not expect to be able to model it that way. Soft objects do not lend themselves to solid texturing for different reasons, explained below.

An excellent recent survey of texture mapping can be found in Heckbert (1986). That paper also deals with related issues like antialiasing which are not considered here.

Soft objects

Our soft objects are represented by surfaces of constant value in a scalar field. Details of this representation and methods of rendering have been described elsewhere (Blinn 1982, Wyvill 1986b, Nishimura 1985). In this paper we give only sufficient background to explain the rather special problems associated with adding texture.

A soft object is defined by a number of key points in space and these key points form a kind of skeleton around which the soft object is drawn. What we actually do is to regard each key point as a source of energy, a hot spot around which the temperature drops off as a function of distance. Every point in space is then considered to possess an energy or *field* value. The field value due to two or more key points is the sum of the values due to the individual key points. The surface of the soft object is a surface whose field value is constant and equal to a special value which we call *magic*.

The field value of an isolated key point at the location of the key point is greater that *magic*. So the soft object due to a single key point is a sphere. A collection of key points produces a shape which is a kind of blending of these spheres. This shape can be very complicated. During animation, the key points can move relative to each other causing a smooth change of shape. Indeed, if the key points move far enough apart, the objects actually break up. A single closed surface can change its topology as holes appear and disappear. This kind of effect is discussed in Wyvill (1986a).

To achieve these smooth changes, the field produced by each key point has to conform to certain rules. In particular, the function, f, which describes the field must vanish to zero at some distance from the key point. For a point at distance r from a key point, the derivative:

$$\frac{df}{dr}$$

must also vanish at this special distance. We call this the 'radius of influence' of a key point: the distance at which the field contribution falls to zero.

This modelling technique is still fairly new and there are many further developments expected. We have already been experimenting with a new formulation in which the key points are replaced by a more complex key with three associated axes (Wyvill 1986c). Some of the figures have been generated using the new technique but the texturing technique is the same and easier to understand in the context of the original method.

Texturing soft objects

It should be clear from the last two sections that soft objects present a special problem for texture mapping. If we use a conventional, two-dimensional map, we have to provide some sort of co-ordinate system on the surface of our object. But this surface not only changes shape, it can change topologically. A torus can change (nearly) smoothly into a sphere and as the hole closes up, we may not want an obvious discontinuity in the texture.

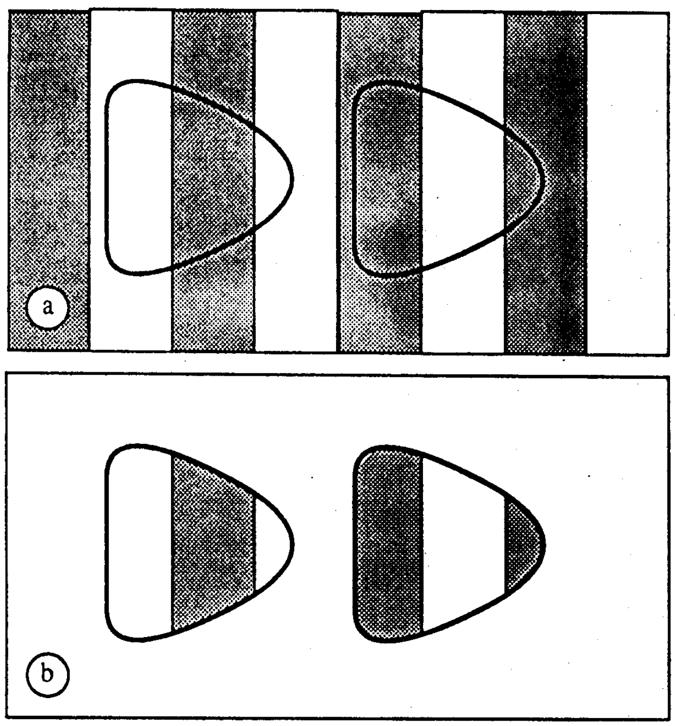

Figure 1: a) The same object is shown in two different positions relative to a solid texture space.
b) The result is that the object changes its appearance as it moves.

It would seem that our only choice is to use a three-dimensional (solid) texture, but this, too, is unsatisfactory for the following reason. Suppose we use a three-dimensional texture map which is fixed in world space, then the texture of an object moving through space will change as it moves, Figure 1. One way to avoid this, is to tie the texture space to the object. That is, we use a co-ordinate system in which the object is defined to be stationary and define our texture map there. During the rendering process, we must translate the co-ordinates of each surface component back into the object's system before we look up the texture. This is a satisfactory solution for rigid objects, but it fails in many cases for soft objects.

Figure 2 shows such a case. A large droplet splits into two smaller ones. If we regard the two droplets as a single object, then the origin of the texture space will be at the centre of gravity of the large droplet. In the final frame, it will be mid-way between two separate objects travelling in opposite directions. Each of these objects will be travelling through a texture space just as if it were tied to the world co-ordinates.

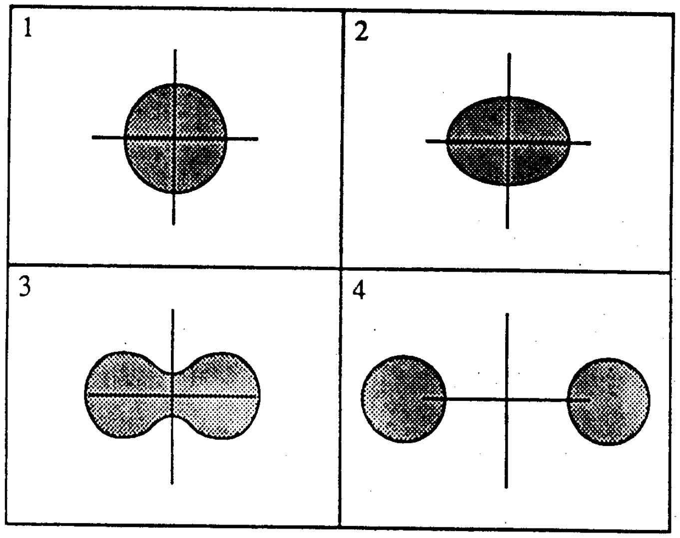

Figure 2: Four stages of an animation. The texture space co-ordinates are tied to the centre of gravity of the object. By frame 4, we have two objects moving through texture space.

Similar problems arise when we consider the rotation of soft objects. If an object rotates without distortion, then the textures on its surface should not change. This implies that the solid texture space must rotate with the object. But it is an essential feature of the soft object modelling technique that we do not refer to objects as such. We control the position of key points and allow the surface topology to change as it will. It seems that whatever method we choose, we can find an example where we do not get the effect we are seeking.

We need a method of setting up a texture map which refers only to key points and never to objects. Each key point, therefore, is embedded in its own **abstract texture space.** This space is used only as a device for assigning a value to every part of the surface. It need have no geometrical significance. There are many ways in which we can set up this abstract texture space, so let us first demonstrate it by means of a simple example. Suppose a key point is considered to be at the origin of an ordinary co-ordinate system. Then <x, y, z> is the name of a point in space defined with respect to this key point. In creating the coordinate system, we have also given the key point a new property. It is no longer symmetrical. We can assign properties to the space surrounding it and we have given meaning to the idea that the point itself can rotate: Rotation of a key point implies rotation of these local co-ordinates. We define the abstract texture space as follows:

$$f = F(x, y, z)$$
$$h = H(x, y, z)$$
$$c = C(x, y, z)$$

<f, h, c> is the name of a point in abstract texture space and the functions F, H, C can

be chosen at will, depending on the application. In most of what follows, we simply set $f = x$, $h = y$, $c = z$ so that <f, h, c> can be regarded as a point in space in a system whose origin is a particular key point.

For our purpose, we need to assign an <f, h, c> triplet to every point P, on the surface of a soft object. We do this by means of a simple, weighted sum. Each key point i, contributes an amount Q_i to the field at P. If $<f_i, h_i, c_i>$ are the values of the abstract co-ordinates of P in the system of i, then:

$$f = \Sigma f_i Q_i / \Sigma Q_i$$
$$h = \Sigma h_i Q_i / \Sigma Q_i$$
$$c = \Sigma c_i Q_i / \Sigma Q_i$$

Of course, ΣQ_i is equal to *magic,* the field value at the surface.

Once we have defined <f, h, c> for a point on the surface, we can look up values in tables, or apply functions to describe the texture and colour at that point. Our textures become functions of the three dimensional f-h-c space and we can perform solid texturing just as we would for a conventional object in world space.

Properties of f-h-c space

Figure 3 shows a classical soft object picture with texturing. In this case, identical F, H, C functions are used for each of two key points, so when they are sufficiently separated, they appear as two identical spheres. As the key points approach each other, parts of the surface acquire <f, h, c> values interpolated by the weighted sum and the texture changes accordingly. Notice that at no stage does the texture exhibit a sudden break. Even at the moment the two objects become one, the texture retains its continuity. This is because the values of f, h and c change continuously both as functions of time and as functions of position on the surface.

There is, of course, distortion. In Figure 3, the texture is defined as a pattern of coloured slices in f-h-c space. Because we are using $F(x) = x$, $H(y) = y$, $C(z) = z$, every <f, h, c> in the region of an isolated key point is equivalent to a simple position in space and the isolated sphere appears to have been carved from a solid mass with this texture. In the region where the droplets begin to merge, <f, h, c> is interpolated between two values. One is characteristic of the top of the lower sphere and the other is characteristic of the bottom of the upper sphere. Thus in a small region of physical space there will be a rapid, continuous change in f-h-c space. This causes the texture to appear compressed.

Such distortion is inevitable when we try to map a texture onto a surface which can stretch and even change its topology, but if we know, in advance, what kind of relative motion of key points to expect, we can take steps to minimise this. For example, we can define:

$$F(x) = x, H(y) = y, C(z) = z$$

as before, for the top key point and:

$$F(x) = x, H(y) = y - R, C(z) = z$$

for the lower point, where R is the radius of influence associated with these key points. The effect of this is that the two spheres appear to have been carved out of different parts of f-h-c space, but when they merge, the parts of the surface most violently affected by the interpolation are not too far apart in f-h-c space. This is illustrated in Figure 4.

Solid textures as functions

Any solid texture can be regarded as a function in that it maps position in space into values which describe surface properties. Similarly, we express any surface property as a function of f-h-c space. Figures 5 to 7 show the droplets textured with a variety of simple functions. In Figure 6, we are using the function to describe perturbations of the surface normal rather than changes in colour. This produces a bump map and, as before, continuity is maintained during the motion. Figure 7 shows both bumps and colour changes.

Smoothing

It is not the purpose of this paper to describe the rendering of textured soft objects. But it is worth noting that the use of an abstract texture space applies equally well to rendering by scan-line techniques or ray tracing. Ray tracing of soft objects has been described (Nishimura et al 1985), but the photographs in this paper were all produced using a modified A-buffer algorithm (Carpenter 1984, Abram 1985). The soft objects are first converted into a fine polygon mesh and colour values are assigned to the polygon vertices as for Gouraud shading (Gouraud 1971). At this stage, each vertex is also associated with an <f, h, c> value and a surface normal direction which is an average of the normals for the polygons which meet at that vertex.

As each polygon is scanned, the colour, surface normal and <f, h, c> values are interpolated independently and stored in a pixel buffer. Different colour and textures can be applied, after rendering, by post processing in this buffer. This technique follows Perlin (1985).

In-betweening

One of the most pleasing features of soft objects is that in-betweening of three dimensional shapes becomes much easier than with other modelling techniques. The surfaces of soft objects are guaranteed to be closed and this enables us to produce consistent in-betweening even when the starting and finishing shapes are topologically different. This is discussed in more detail in Wyvill (1986). When we add texture, we also add the requirement that sudden breaks should not occur in the texture and this is handled very neatly by the use of the abstract texture space. A further problem occurs if the starting and finishing shapes are required to have different textures. Figure 8 shows a sphere with texture in-betweened into a torus in four stages. The texture change is effected by separately interpolating the features of the two textures. For example, the normal at each point of the surface is perturbed in two directions at angles α, β where:

$$\alpha = A(f, h, c)$$
$$\beta = B(f, h, c)$$

and A and B are texture functions. If the in-betweening takes place over n frames of animation, then for frame i we set:

$$\alpha = ((n-i) * A_1(f, h, c) + i * A_2(f, h, c)) / n$$
$$\beta = ((n-i) * B_1(f, h, c) + i * B_2(f, h, c)) / n$$

Discussion

There would appear to be no simple, consistent way to create surface co-ordinates for texture mapping on soft objects, nor can they be treated satisfactorily by the technique of solid texturing. By creating an abstract texture space related to each control point, however, we can successfully map textures onto soft objects which are moving, deforming and even changing topology.

The way the texture behaves in animation, depends on what we suppose the motion means. In Figure 9, a striped ellipsoid is rotating but the abstract space associated with each key point is not rotating. If we want to regard the motion of a collection of key points as a rotation, then we must rotate the <f, h, c> space. Otherwise the texture behaves as if the object has merely transformed to the new shape by distortion. There is nothing *wrong* with Figure 9. The animation represents a blob of soft material whose surface is changing (perhaps because of some internal flow of material). Only if we want to regard the motion as a rotation without distortion need we deliberately rotate the texture space.

We can, indeed, represent objects which both rotate and distort. In doing so, we make a conscious choice about how to handle our abstract space. We are made to specify just how much of the motion of key points is attributable to the rotation. What appeared to be an impasse, turns out to be a need for more detailed specification. The idea of an abstract texture space gives us a simple tool with which to express that specification.

References

Blinn J (1982) "A Generalization of Algebraic Surface Drawing" ACM Transactions on Graphics Vol 1 pp 235

Blinn J and Newell M (1976) "Texture and Reflection in Computer Generated Images" CACM Vol 19 No 10 pp 542-547

Blinn J (1978) "Simulation of Wrinkled Surfaces" *Computer Graphics (Proc. SIGGRAPH '78)* Vol 12 No 3 pp 286-292

Carpenter LC (1985) "The A-buffer, an Antialiased Hidden Surface Method" *Computer Graphics (Proc. SIGGRAPH '84)* Vol 18 No 3 pp ••• check this

Abram G, Westover L and Whitted T (1985) "Efficient Alias-free Rendering using Bit-masks and Look-up Tables" *Computer Graphics (Proc. SIGGRAPH '85)* Vol 19 No 3 pp 53-59

Gouraud H (1971) "Continuous Shading of Curved Surfaces" *IEEE Transactions on Computers* C-20 No 6 pp 623-628

Heckbert P (1986) "Survey of Texture Mapping" *IEEE Computer Graphics & Applications*, Vol 6 No 11 pp 56-67

Nishimura H, Hirai M, Kawai T, Kawata T, Shirakawa I and Omura K (1985) "Object Modeling by Distribution Function and a Method of Image Generation" Journal of papers given at the Electronics Communication Conference '85 Vol. J68-D No 4 (in Japanese)

Peachey, D (1985) "Solid Texturing of Complex Surfaces" *Computer Graphics (Proc. SIGGRAPH '85)* Vol 19 No 3 pp 279-286

Perlin K (1985) "An Image Synthesizer" SIGGRAPH 85 Computer Graphics Vol 19 No 3 pp 287-296

Wyvill BLM, McPheeters C and Wyvill G (1986a) "Animating Soft Objects" *The Visual Computer* Vol 2 No 4 pp 235-242

Wyvill G, McPheeters C and Wyvill BLM (1986b) "Data Structure for Soft Objects" *The Visual Computer* Vol 2 No 4 pp 227-234

Wyvill BLM and Wyvill G (1986c) "Using Soft Objects in Computer Generated Animation" •••fill in details ••• Springer Verlag (in press)

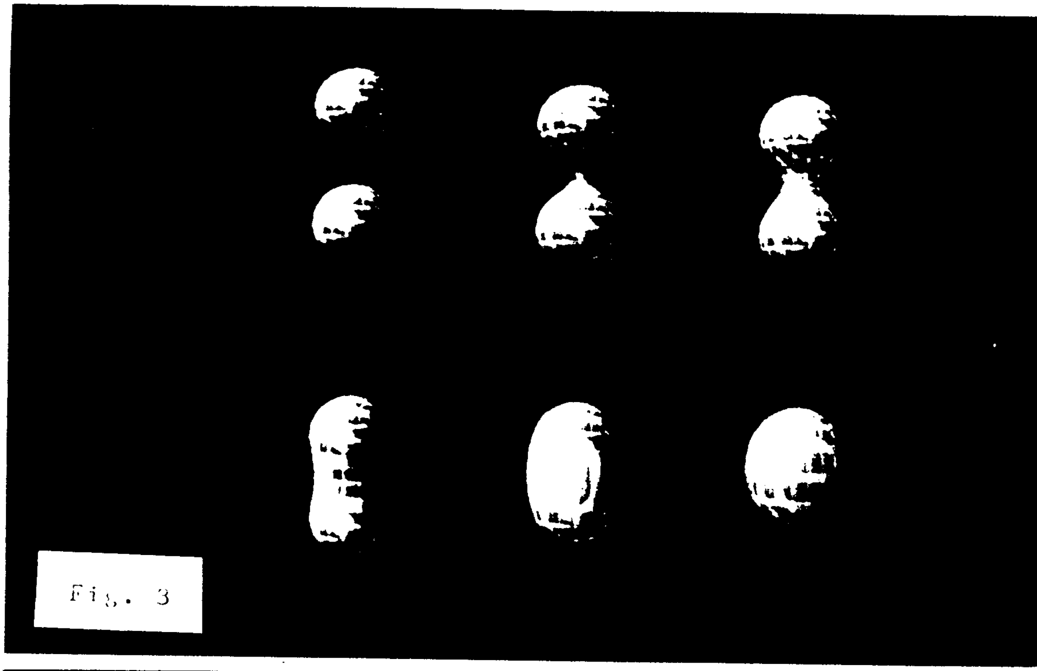

Fig. 3

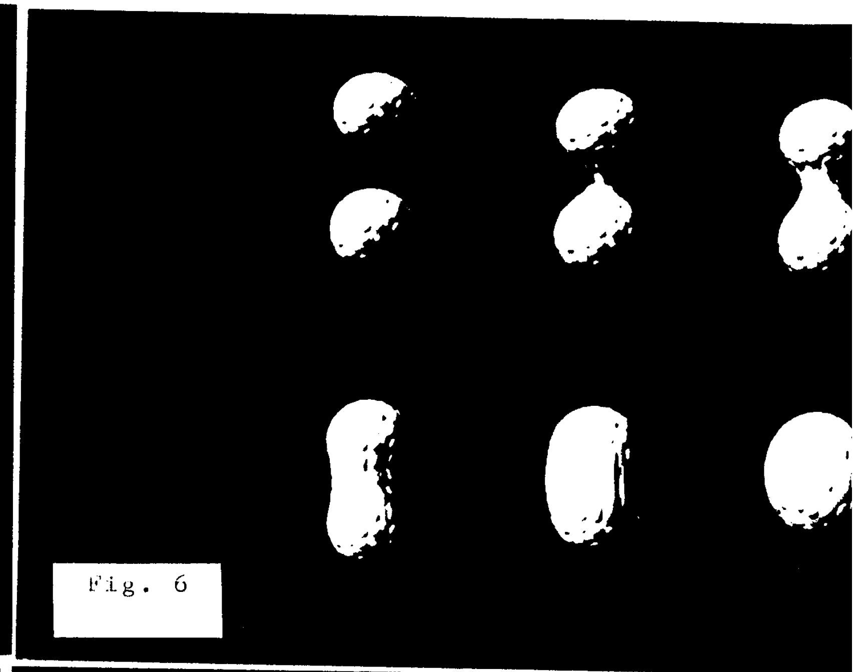

Fig. 6

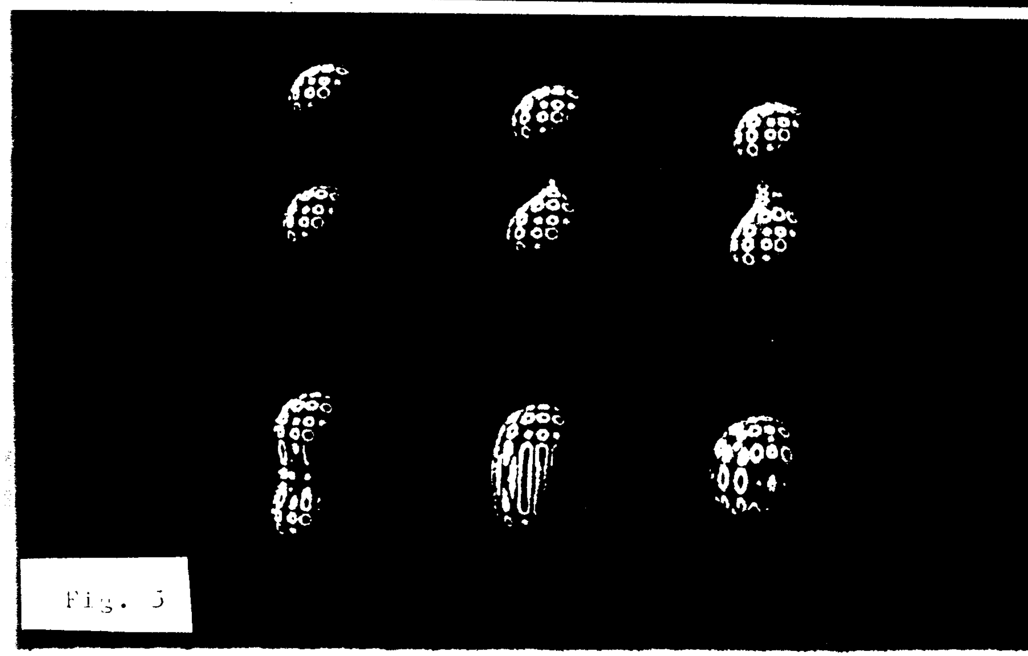

Fig. 5

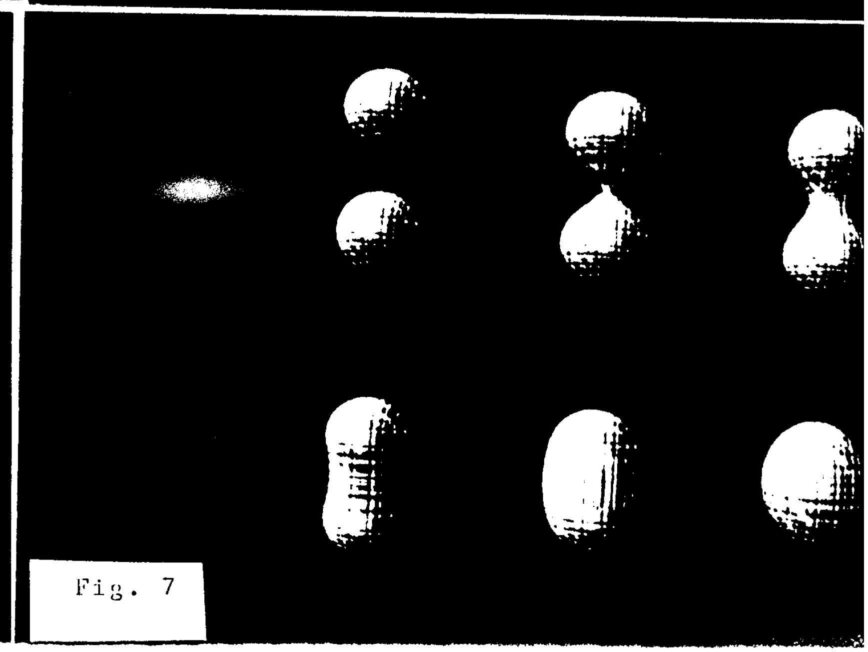

Fig. 7