PERIODICA POLYTECHNICA SER. EL. ENG. VOL. 38, NO. 2, PP. 175-190 (1994)

AN EDGE FOLLOWING ALGORITHM AND ITS APPLICATION

Tibor KOVÁCS

Department of Automation
Technical University of Budapest
H-1521 Budapest, Hungary
e-mail: kovi@bme-at.aut.bme.hu

Received: Sept. 10, 1994

Abstract

The aim of this paper is to present an algorithm to following edges, curves on pixel based pictures. The application for which the method was developed is a 3D modeller system. The main target was the accuracy of the scanner (so the accuracy of the algorithm), the speed was a secondary factor. An active triangular scanner configuration was implemented (described later) with a matrix camera, so the input image of the algorithm is a bitmap of the connected frame grabber. However, this method is usable in pattern recognition, in vectorisation of pixel based images, in geography or in other 3D scanning methods.

The advantage of the algorithm (and the ground of the accuracy) is that it takes into consideration the noise of the image system and the profile of a curve. The binarisation of the image (applying a threshold) is a waste of information, so this edge following algorithm tries to use the valuable information encoded in grey levels.

Keywords: edge following, 3D model building, 3D scanner.

Introduction

What is the visualisation? It's a technique that helps us imagine a model world, a world that describes objects and actions of daily life, like buildings and cars, electrical or mechanical accessories, mathematical structures. This world can be an abstract scene with physically non-existing objects, entities, for instance strange creatures or special effects.

We can see our world from unusual viewpoints with unusual scale, like the interior of a cylinder of an engine, or we can sit on a bullet coming from a gun.

Let us take an example from the electrical engineering. An animation was made about the installation of a 10 kV cable end in a transformer station. It was necessary to emphasise the essence of the technology, so we don't need to see the workman. It was enough to sign symbolically longer

workphases or well-known tools. The full technology (one hour long in real life) was presented in a 50-second animation.

Creating the Object Space

The result of a process was described, but how can we create these worlds?

First we should generate a mathematical model of surfaces. In a simple case it is a wireframe model, but in more complex applications it can be, for example, a spline based model. The wireframe consists of three-dimensional points (vertices), edges and faces, flat polygons.

After this we have to mark real surface elements and holes, because an area bordered by edges may be a face or a hole.

In this level our model has a not-so-realistic view, it has only flat faces. We can smooth surfaces, or at least we should have the appearance of smoothness. We can reach this aim varying surface normals between two 'edge normals'. If we use the mathematical normal of a face to shade it, it will have a flat appearance, but if we interpolate normals between two mathematical ones, and we use this temporary vector to shading, we will have a smooth face. It can be seen in *Fig. 1.*

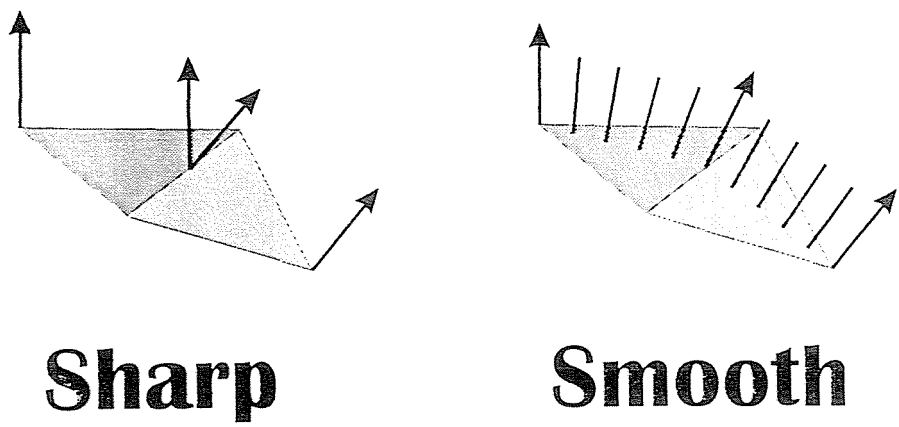


Fig. 1. If we want to see a faceted object with sharp edges, we can use a simple flat shading model with mathematical normalvectors. If smooth surface is desired, mathematical normals have to be interpolated.

It is a practical solution, because most optical surface features are functions of surface normals. These features are, for example, reflection, refraction, surface microstructure, opacity, etc. These features are applicable by using this virtual interpolated surface normal.

Based on the surface model mentioned we can construct a wide variety of materials, surface qualities, defining mentioned functions, surface textures, rovghness and colours. Now our object is ready to put to the scene.

Note that there may be more phases in creating a complete animation, like defining movements and video processing, but the objective of the work was only the modelling phase.

Construction of Wireframe

How can we generate a wireframe model? There is a summary of mentioned methods in *Fig. 2.* Theoretically all shapes and volumes can be built up from individual vertices. We can approximate all real objects by creating a dense vertex system. We have to connect these vertices in a well-defined way, and the object is ready.

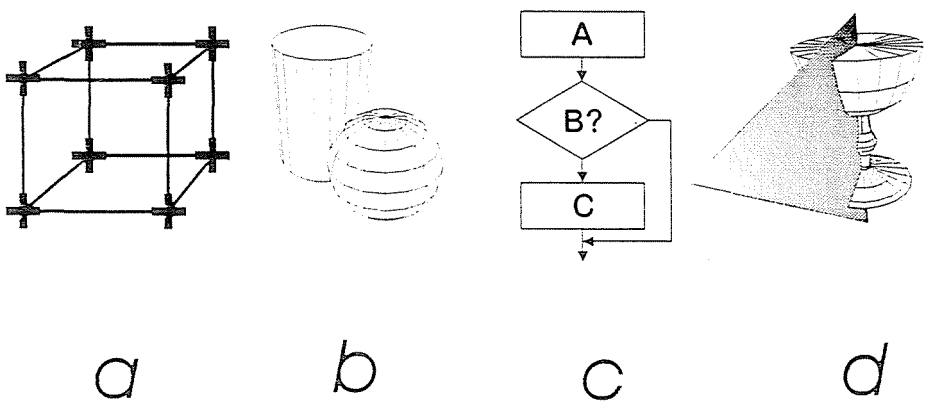


Fig. 2. Examples to generate models: *a*) manual creation of vertices and faces *b*) creation of elementary objects *c*) procedural wireframe generation *d* 3D scanning

But this process is similar to painting photos pixel by pixel. It is possible, but not effective. There are four methods we can choose from:

- If the object is simple enough, like a tetrahedron or a cubo-octahedron, we can build it up vertex by vertex. This way of model building is not so frequent; it is used to create mathematical solid bodies, well defined small objects, or to modify more complex objects by some vertices. The latter is the only possibility to correct the shape in a lot of cases.
- In the real world, mainly in the human environment (in engineering or household), objects are built up of elementary cells, like spheres, cylinders, cubes. We can use other predefined elements in modelling

as well. All modelling systems offer a set of elements, in some cases we can add user defined bodies to this set. In manufacturing it is a strict requirement that work-pieces must be processable, so this shape family is described with planes, cylindrical or spherical surfaces. A set of procedures can help us to generate or modify basic objects.

- In contrast in the objects of the natural environment are more complex. Fortunately, we are able to describe most of these objects with creating rules. These rules can be based on, for instance, fractals or they can reflect the inner structure of the object. The rules don't explain the exact structure of the appropriate object, but only their behaviour or global features. So we can create trees, clouds or hills in this procedural way. These procedures can be parameterised to slightly vary the shape, extent or the complexity.
- Many surfaces are very difficult or impossible to generate by the three ways mentioned. Today feasible rules don't exist to create human faces or complex industrial shapes. We aren't able to build them out of cones and spheres. But we have a fourth way:

The 3D Scanning

The 3D scanner is a device that scans a solid object, makes a 3D model of it for CAD or other modelling applications or gives a description of the workspace to a robot. There are a wide variety of solutions of this task. In this part a method is described to generate models.

The moiré interferometry (presented in *Fig. 3*) is based on a well-known phenomenon in press technology. If two similar patterns are projected onto each other, and one of them has a slight distortion (rotation, long wavelength disturbance), it results in a so-called moiré pattern. It characterises the distorted pattern (if the other pattern is ideal, non distorted).

A moiré-based range imaging system has the following components:

- a pattern projector that, in a simple case, projects a parallel line structure to the scene (onto the object)
- the target object that distorts this pattern
- the sensor that has the ideal (non distorted) pattern. This is often a matrix camera and a mask identical to the projector pattern.

In fact, the moiré interferometry is a special triangulation. The disadvantage of the method is that we are able to detect only relative changes of the surface, but not the steps of it. If we need absolute surface geometry, we have to refer to an absolute point in the scene.

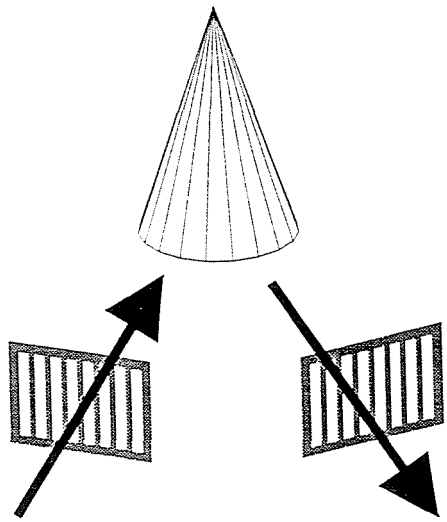


Fig. 3. The principle of the moiré topography

There are some varieties of the method with different grid configuration and patterns.

A well-known phenomenon is used by the imaging radar (shown in *Fig. 4*). The signal flights to a point of the object and back. The appliance measures the transit time of the signal from the radar transmitter to the reflecting object and back to the radar receiver. The speed of the signal propagation is known, so we can calculate the roundtrip distance, the distance of the current point.

The signal may be ultrasonic wave or electromagnetic wave (in radio or light domain, typically laser source) depending on the dimensions of the scene. It can be based on time delay, phase difference or frequency deviation. The accuracy is typically high (0.1 nm) and the depth of field is in the 100 km magnitude of order. The procedure may be inexpensive or extremely expensive as well.

The focusing technique described in *Fig. 5* is based on the Gauss thin lens law. If the sensor measures the distance of the focused point from the lens centre, the distance of the source point can be calculated through the focal length.

Unfortunately, an optical system has many errors and nonlinearities, so this method is not too accurate (1 mm), but it is a reasonable solution for inexpensive systems.

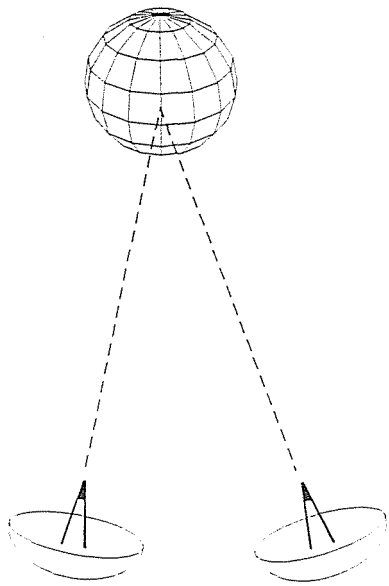


Fig. 4. The imaging radar is a very accurate, but potentially expensive process

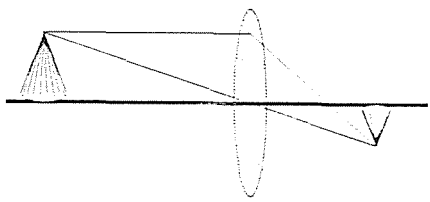


Fig. 5. The focusing is typically not an exact method due to the nonlinearities of optics and the difficulties of making sharp image for measuring (the 'focusing')

From the viewpoint of the topic the most important system is the active triangulation.

The Active Triangulation

The basic principle is known from the geographical or the astronomical triangulation. We try to determine the 3D position of a point determining two view directions of this point. The two views are created from the

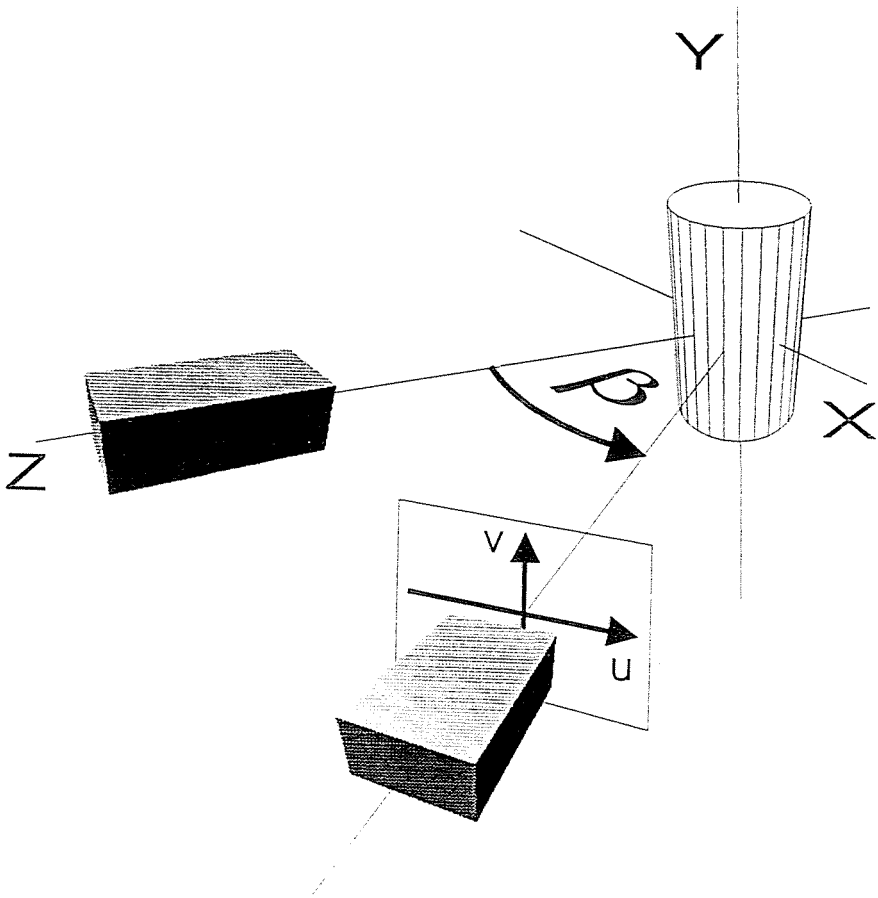


Fig. 6. This is the configuration of the test environment. There is a line structured laser source at the Z axis, the camera is at a base angle to the light source. The object is in the origin.

two endpoints of a baseline. In the active triangulation one of the two viewpoints is an 'active sensor', a marker, typically a light source. It marks current point(s), and the 'passive sensor', the detector measures the view direction(s) of it (them). Later the active light marks other points, so the surface is scannable in this way.

It's possible to apply many kinds of active light structures. It may be a point source, horizontal or vertical line, point system (for instance point matrix), line system, circles, elliptical or other special patterns, colour

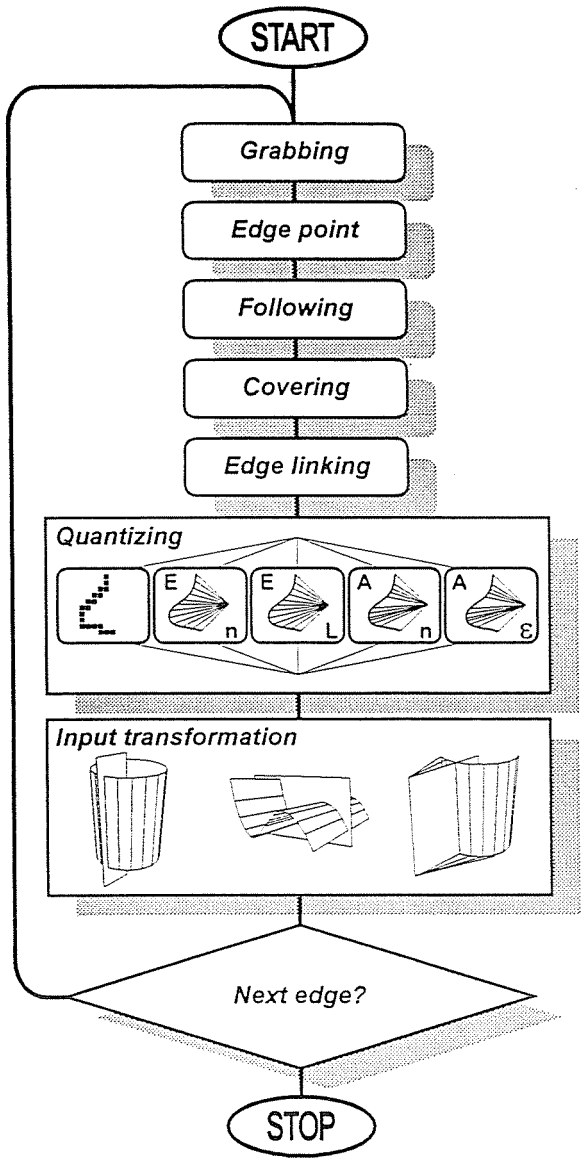


Fig. 7. The algorithmical layout of the scanning. The quantization is a process, that places vertices onto the physical edge or fits a mathematical curve on it. The input transformation is the relation between the movement of the object and the active light.

encoded patterns. The difference is the speed of preparing, the accuracy, the necessary computational capacity.

In our case a vertical line was chosen to scan the scene. It is implemented with a He-Ne laser source. The configuration can be seen in *Fig. 6.* The light line is created with a cylindrical lens. The image of the light curve is grabbed by a frame buffer under a resolution 512 × 512 and the number of greys is 32.

The edge following algorithm must follow the curve distorted by the object. This curve has a number of potential errors. The source image is more or less noisy, the image of the really continuous line may be broken, there may be pseudo-edges (line segments) on the image because of reflections or other environmental lights. A configuration was constructed that tries to decrease the effect of these disturbances, but there are a number of steps in the edge following algorithm that are intended to ignore these errors as well. The main steps of the edge creation are shown in *Fig. 7.*

The edge following

We can better understand the final method, if we follow the steps of algorithmical modifications. The algorithm the more accurate, the less effective. Note again that the goal was the accuracy, not the high speed.

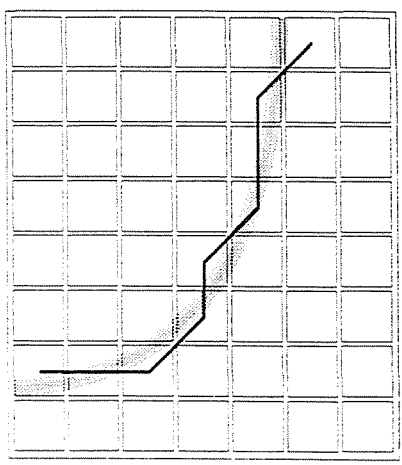


Fig. 8. The resulted edge will have many breakpoints, if only integer coordinate space is used.

The picture elements (pixels) are traditionally addressed with integer co-ordinates in a frame buffer. So we can achieve high speed, if we work in this integer domain. However, if we suppose eight-neighbourhood, the resulting edge will be rough, as *Fig. 8* shows. If we suppose ideal sampling, accurate following is possible along the eight main directions. So we have to change to real addressing (real frame buffer co-ordinates).

In the real system we use the physical integer addressed frame buffer, but we fit another logical buffer, where we can address in between two pixels. We should modify the algorithm to be able to give us directions in between the eight main directions (practically it should return a real angle value).

The new step angle (interpreted by *Fig. 9*) can be calculated by weighting angles with appropriate intensities:

$$\delta = \frac{\sum I_i \ \varphi_i}{\sum I_i} \ . \tag{1}$$

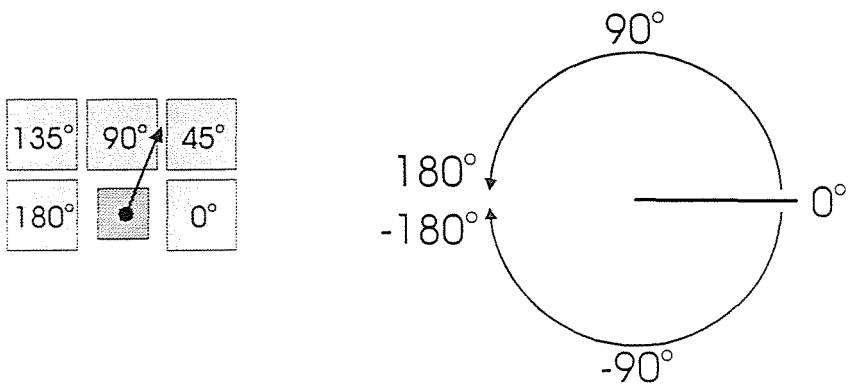


Fig. 9. The interpretation of step directions, if the following moves upward

In an ideal, noiseless case this method can result in a reasonable edge. But in the real environment the noise deforms the result of the determination of the next step direction. So the edge will be wavy, far from the requirement 'accuracy'.

To treat this problem, let's look at the following solution including lowpass filtering. The smoothing is accomplished by taking into consideration farther pixels. Since the edge is linear along a short distance (3..6 pixels), most intensive pixels statistically will stand in a well defined direction. So the looking at a longer distance will straighten, filter the local noise in the curve.

After this idea it is natural that the weighting over a bigger area should be implemented that the size of this weighting shape is a parameter. So around the current point will form shells (presented in *Fig. 10*), we can control the filtering with the shell size. This is an important step, because we will be able to set a model, for what is the relation between the necessary shell size, the noisiness of the source image and the characteristics of the frame buffer.

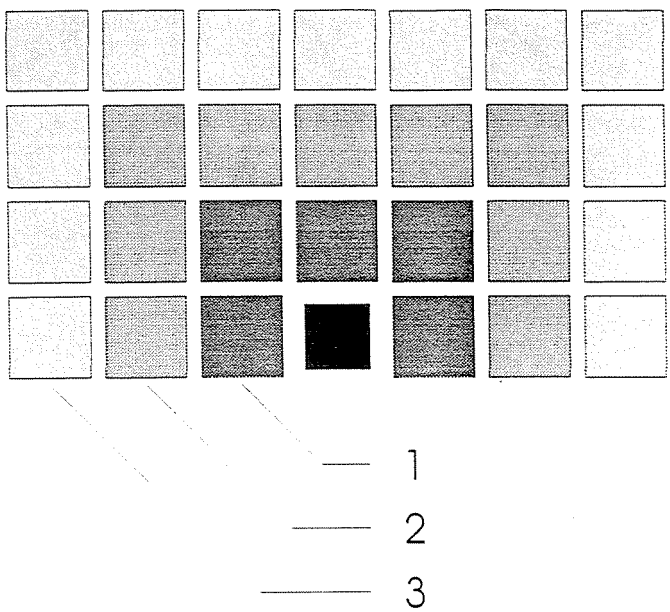


Fig. 10. We can imagine shells as pixel layers around the current pixel

In the application the light source was a He-Ne gaslaser. It is well-known that the intensity profile of a TEM_{00} gaslaser is a gauss profile. So the cross-section of the line projected to the object is a gauss curve. In ideal circumstances the intensity image of the full edge is gauss hill. The task is to go along the ridge of this hill.

Let us see the *Fig. 11.* Let us sign the width of the screen (in pixels) with W_s, the number of grey levels with D_s. At an average setting of the imager system we can suppose that the thickness of the edge is a given percentage of the width of the screen (frame buffer). This experimental value is P. The measure of the noise is N (if gauss noise is supposed, N is the variance).

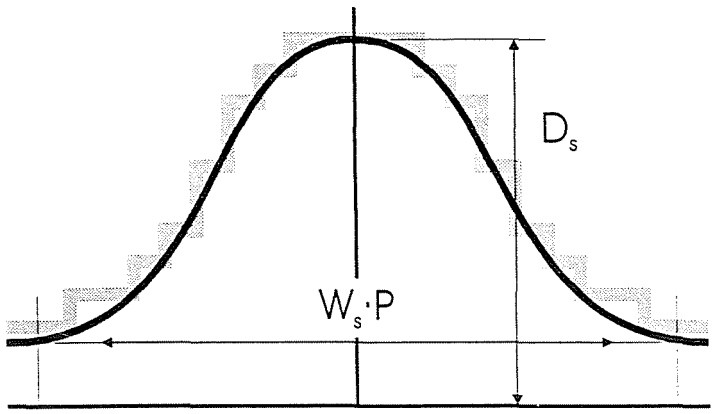


Fig. 11. The ideal and a real edge profile

At a correct setting the intensity profile is the following (assuming that the curve is symmetric to the origin):

$$I(x) = (D_s - 1) \exp \frac{-x^2}{(W_s P)^2} \,. \tag{2}$$

Since the frame buffer works with finite grey levels, a maximal intensity $(D_s - 1)$ ramp forms around the origin. If the shell size is smaller than the ramp size, we can't conclude the local direction of the ridge. The task is to calculate the minimal distance, in which we can see pixels with lower intensity then $D_s - 1$. We have to take into consideration the noise, so we should find the $D_s - N - 2$ intensity point of the mentioned profile:

$$I(s) = D_s - N - 2 \,, \tag{3}$$

$$D_s - N - 2 = (D_s - 1) \exp \frac{-s^2}{(W_s P)^2} \,, \tag{4}$$

$$-\ln \frac{D_s - N - 2}{D_s - 1} = \frac{s^2}{(W_s P)^2} \,, \tag{5}$$

where s is the minimal necessary shell size. So

$$s = W_s P \sqrt{-\ln \frac{D_s - N - 2}{D_s - 1}} \,. \tag{6}$$

The quality of edge following has become better, but there is another problem. The curve is excellent in the vertical direction (90°), but near to the

horizontal direction stepvectors keep up (supposed that the edge following moves upward, appropriate principles are true downward). The reason is that we calculate the direction using only the $0 - 180°$ range. This range reasonably covers the vertical directions, but horizontal ones are asymmetrically covered.

The insufficiency can be partially corrected, if we use rotated covering shapes shown in *Fig. 12.* We choose covering shape depending on the last step direction. The advantage of this method is that it can be relatively simply implemented. At the same time we can think about a more perfect covering process.

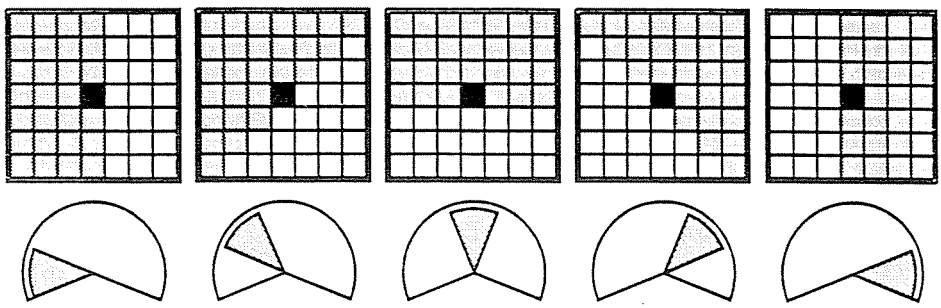


Fig. 12. The covering shape tries to following the edge direction by rotation

We can see that there is a slight anomaly in the extracted edge at the change of the covering shape because of the described asymmetry. This anomaly is presented in *Fig. 13.* We can't avoid the continuous changing of the shape. The continuous variety of the (1) expression is the following:

$$\delta = \frac{\int_\Phi I(x,y)\varphi(x,y)d\Phi}{\int_\Phi I(x,y)d\Phi} = \frac{\sum_\Phi I(x,y)A(x,y)\varphi(x,y)}{\sum_\Phi I(x,y)A(x,y)} \,. \tag{7}$$

Since a pixel has homogeneous intensity; the integral expression can be reduced to a finite sum expression. The A elementary pixel surface is the intersection of the covering shape and the given pixel ($A = 1$, when a pixel is completely covered, 0 without any section). In the case of partial covering we need to think about the following (these cases are shown in *Fig. 14*, two trivial cases are not necessary to show).

The covering shape is intersected by a line going through the origin. We calculate only with the area over the line. The equation of this line:

$$f = ax + by \,. \tag{8}$$

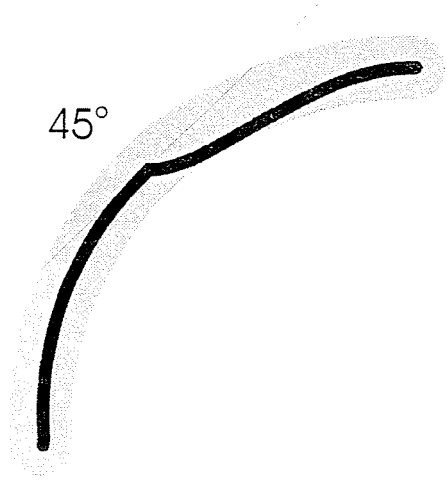
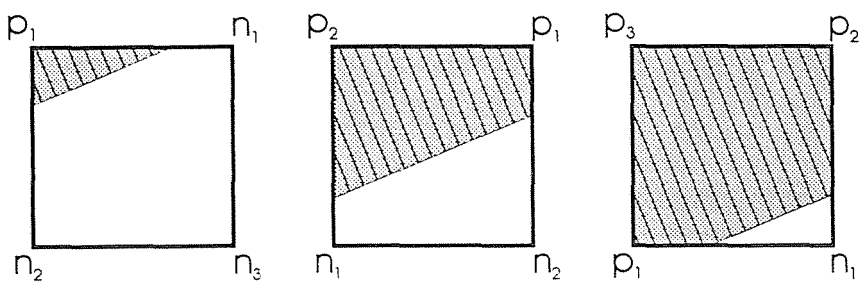


Fig. 13. The edge following by discrete covering shape rotation has an uncontinuity problem. During the changing of the covering shape the behaviour of the following is seems to be instable.



$$w = \frac{1}{2C}\,\frac{p_1^2}{p_1 + n_1} \qquad w = \frac{p_1 + p_2}{2C} \qquad w = 1 - \frac{1}{2C}\,\frac{n_1^2}{p_1 + n_1}$$

Fig. 14. The calculation of the pixel weight. There are first and the last is trivial (the pixel weight is 1 and 0).

The previous step direction is known, so

$$a = -\cos\delta\ , \tag{9}$$

$$b = \sin\delta\ . \tag{10}$$

The f expression is positive under the covering shape, and negative outside.

We calculate f in the four corners of every pixel. In fact, it doesen't need to calculate four f per pixel, because one f value besides to four pixels, so we need one calculation per pixels. We can distinguish five cases:

If every f_i are positive, the weight of the pixel is 1.

If there are three positive (let us mark them in ascending order with p_1, p_2, p_3) and a negative value (we use the absolute value of the negative figures later, let us mark them with n_1, n_2 and n_3). The following expression returns the weight of the appropriate pixel:

$$w = 1 - \frac{1}{2C} \frac{n_1^2}{p_1 + n_1} . \tag{11}$$

The C value is the result of a decision. Depending on the previous step direction we have to substitute one of the four trigonometrical functions. Since f values represent the distance from the line (the border line of the covering shape was a normal equation), these cases can be described as the calculation of the surface of triangles and trapezoids. So in the case of two positiv- two negative f values, the weight function is

$$w = \frac{p_1 + p_2}{2C} , \tag{12}$$

and in case of one positive f the expression is

$$w = \frac{1}{2C} \frac{p_1^2}{p_1 + n_1} . \tag{13}$$

If all of f values are negative, the weight of the pixel is 0.

Note that these expressions don't result in division by zero, because the value of C can't be zero (using the mentioned substitution), p_1 and n_1 can't be zero at the same time, because the line must go across the origin.

So we can say the weight of every pixel (the effective surface of them), so the described integral can be exactly calculated. The demand of computational power has increased but the edge following is ideal (supposed that the described model is good).

We can enhance the result in extremely noisy environments, if we use the later calculated direction in a previously given level. It can be considered an integrating term, the noisy reference signal is the source edge (the input image), the manipulated variable is the calculated, smoothed edge (direction). If the amplification in the feedback is decreased, the next calculated direction will have higher weight. If it is increased, the following will be less sensitive to the noisiness of the input edge. In later case the algorithm will be less accurate and won't be able to follow sharp corners of edge (like a low pass filtering).

Conclusion

A reasonable quality 3D scanner is typically expensive investment. The implemented system includes relatively cheap components, so the task was to develop an algorithm that had a high level of accuracy (this accuracy is not available today). The main problems were noisiness of the frame buffer and the relatively low resolution. The software (under developing) can be connected to any kind of hardware that implements the described model: edge images of intersections of real object can be generated (through matrix camera or linear camera, linear structured light or synchronous scanner with point light structure, any kind of described configuration, input transformation can be used). So it was an advantage to work with this simple system, because it was possible to study all the disturbances. The system will be tried later on a professional scanner hardware.

References

IRIS Universe, the Magazine of Visual Computing, number eighteen, Silicon Graphics Inc.

SANZ, J. L. C. (editor) (1989): Advances in Machine Vision, Springer-Verlag, 1989, pp. 1–53.

YOSHIAKI, S. (1987): Three Dimensional Computer Vision, Springer-Verlag, 1987, pp. 32–65, 164–188.

WOHLERS, T. T. (1992): 3D Digitizers, *Computer Graphics World*, July 1992, pp. 73–77.

VANNIER, M. W. et al. (1991): Facial Surface Scanner, *IEEE Computer Graphics and Applications*, November 1991, pp. 72–80.

FRANCOIS, B. – MARC, R. – JACQUES, D. (1991): Optical Range Image Acquisition for the Navigation of a Mobile Robot, *IEEE Computer Society* Press Reprint, April 1991.

MARC, R. – JACQUES, D. et al. (1989): Range Imaging Sensor Development at NRC Laboratories, *IEEE Computer Society* Press Reprint, November 1989.