COMPUTER ANALYSIS OF VISUAL PROPERTIES OF CURVED OBJECTS

Lawrence J. Krakauer

May 1971

PROJECT MAC

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Cambridge Massachusetts 02139

COMPUTER ANALYSIS OF VISUAL PROPERTIES OF CURVED OBJECTS*

Abstract

A method is presented for the visual analysis of objects by computer. It is particularly well suited for opaque objects with smoothly curved surfaces. The method extracts information about the object's surface properties, including measures of its specularity, texture, and regularity. It also aids in determining the object's shape.

The application of this method to a simple recognition task -- the recognition of fruit -- is discussed. The results on a more complex smoothly curved object, a human face, are also considered.

---

*This report reproduces a thesis of the same title submitted to the Department of Electrical Engineering, Massachusetts Institute of Technology, in partial fulfillment of the requirements for the degree of Doctor of Philosophy, June 1970.

Acknowledgements

Professors Marvin L. Minsky and Seymour A. Papert were most helpful during the course of this research and the production of this report. Professor Papert suggested the problem originally.

Russell A. Kirsch was very generous with his time and ideas.

A number of methods were developed during brainstorming sessions with Gerald J. Sussman. He and Thomas L. Jones were particularly encouraging by their enthusiasm for this research.

My wife-to-be, Margret Berman, contributed several ideas for specific experiments, as well as general encouragement and support. The project might never have been completed without her aid.

Work reported herein was supported in part by Project MAC, an M.I.T. research project sponsored by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research Contract Nonr-4102(02).

Table of Contents

Chapter I The Problem

Consider the problem of programming a computer to recognize objects with smoothly curved surfaces, such as the object in the photograph of figure I.1. Images such as these can be digitized by an image-dissector camera, so that the picture is represented by a raster of intensities at closely spaced sample points, represented numerically in figure I.2. We will consider a method of processing such input with the ultimate goal of recognizing the object in the image.

There are numerous more or less adequate known techniques for classifying an image once significant features have been extracted from it, but the problem of extracting such features from the basic optical data is less well understood. The methods which will be discussed here are "low-level", in that they manipulate actual picture points and try to extract salient features, rather than working with high-level descriptions and attempting to produce an identification.

It must be recognized, however, that the so-called high- and low-level aspects of vision cannot really be cleanly separated. There is no foolproof

Figure 1.1: A Simple Smoothly Curved Object

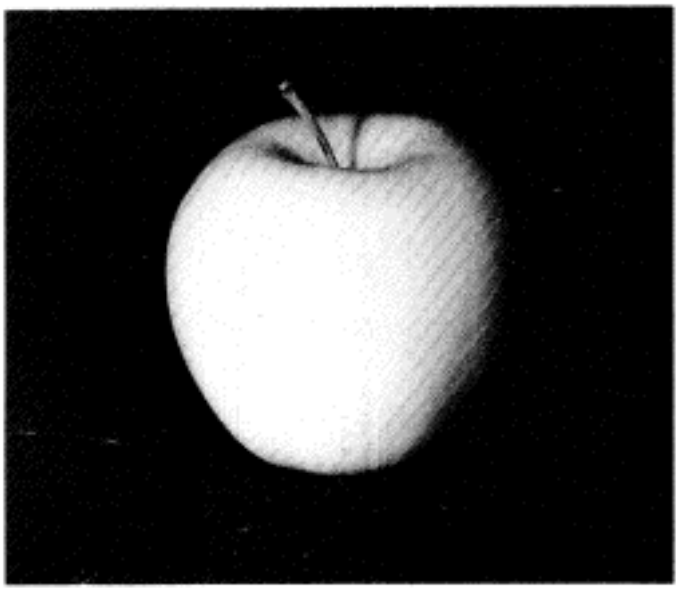


Figure 1.2: Sampled Light Intensities from the Apple of figure 1.1

The intensities in this array have been scaled to be between 0 and 99

5	6	5	7	9	7	8	10	14	14	14	14	14	14	14	13	13	13	12	11	10	9	7	5	4	4	5	4	4
6	7	6	6	8	9	10	12	14	16	16	17	16	15	15	14	13	12	12	10	9	8	6	5	6	4	3	5
6	6	7	7	9	11	12	14	15	17	17	20	20	18	19	16	16	14	13	12	10	9	7	6	5	6	4	4
7	8	9	9	8	11	14	16	18	18	20	20	21	56	20	20	18	15	16	14	11	8	8	7	6	6	4	5
8	8	8	9	10	11	14	17	19	19	22	36	58	58	54	55	49	50	29	14	11	9	9	7	7	4	4	4
7	7	10	9	9	12	15	18	20	24	60	58	50	64	50	67	73	73	65	54	19	10	8	8	5	5	5	5
8	8	10	10	11	13	15	19	25	72	74	75	69	63	59	60	69	63	54	57	49	15	0	0	0	0	2	3
8	7	9	9	13	14	16	21	75	80	82	81	83	79	77	72	70	67	65	62	56	39	1	0	0	0	0	0
8	10	9	11	13	15	17	51	79	84	84	84	85	84	81	79	77	75	72	66	62	47	17	0	0	0	0	0
7	10	10	12	13	14	18	72	79	86	87	88	87	91	85	83	80	80	74	69	61	48	24	0	0	0	0	0
7	9	11	10	13	14	20	72	83	86	87	89	93	99	85	83	81	79	74	69	61	50	24	0	0	0	0	0
9	9	10	10	14	14	20	74	84	85	84	88	89	92	84	82	79	80	74	66	61	47	18	0	0	0	0	0
10	8	11	12	13	15	18	72	81	84	87	87	86	87	86	83	81	79	75	66	56	45	14	0	0	0	0	0
8	9	11	11	12	14	17	67	81	83	84	85	87	84	83	82	81	77	72	65	45	43	9	12	0	0	0	0
7	7	8	12	14	14	14	22	78	81	83	84	85	84	83	80	78	74	68	58	51	33	10	0	0	0	0	0
7	9	9	11	10	10	14	19	75	79	81	81	84	83	80	79	74	73	65	52	41	17	0	0	0	0	0	0
7	9	10	7	10	12	15	16	37	76	79	79	80	80	76	76	74	68	57	46	29	10	0	0	0	0	0	3
7	8	8	13	8	11	21	14	21	70	77	77	78	76	73	72	69	63	49	31	14	0	0	0	0	2	11	7
9	6	9	8	12	9	11	15	14	17	68	59	74	72	69	67	62	53	33	14	0	0	0	3	8	8	6	6
7	6	9	7	14	10	13	12	14	12	18	39	60	64	61	59	50	28	8	0	0	11	5	9	8	8	9	8
7	6	7	5	8	11	11	11	11	13	15	15	18	15	14	15	12	13	12	8	11	9	12	9	8	8	9	7
8	14	7	9	9	8	10	11	14	15	20	15	19	14	15	13	13	10	13	7	7	9	6	9	7	10	7	8
7	9	7	7	10	8	16	10	14	11	13	12	14	15	9	10	10	9	10	12	13	9	7	6	5	9	8	5
9	7	9	9	7	11	11	10	14	11	10	10	13	14	12	12	11	12	9	12	10	7	5	9	8	6	7	6

completely local way to find features, as there will always be ambiguities which can only be resolved through the use of context. For example, one must know the light intensity (at least roughly) in order to determine whether an object is white or black, as a white object in very dim light can easily reflect less light than a black object in sunlight. A plum cannot be easily distinguished from an isolated grape, unless the size is known. Highlights on a smooth surface cannot be understood unless the form of the illumination is known.

The context can of course be determined partly from the scene itself. For example, a real scene will generally contain surfaces with a wide range of reflectivities. This establishes a light intensity frame of reference in which the lighter objects will appear white and the darker ones black. One cannot tell the size of a white sphere alone in a photograph, but if it is shown next to a tennis ball, its size is known by comparison. (It is possible, but unlikely, that the tennis ball is actually a scaled-up model three feet in diameter. This usually happens only on movie sets.) In a similar manner, the highlight on a known object gives information about the lighting which can be used to interpret the highlights on other objects in the image.

So far, the use of context has been considered only on the level of object identification. Actually, context is even more necessary at the level of finding visual parts of objects, such as edges. A line-finding program can be saved an enormous amount of work if it is told approximately where to look. If a program thinks it is seeing an apple, it can know that a good way to verify this hypothesis is to look on top for a stem.

A program can only make use of these cues, however, if it can pass information resulting from a partial identification back to the low-level feature-finding routines. This sort of system shall be referred to as "vertical", in the sense that control passes frequently between high- and low-level routines. The term "horizontal" refers to a system which works in stages, each of which produces a more abstract representation of the scene. Much of the previous work in vision has been of this sort. A typical sequence might be to remove noise, enhance features, extract features, group them, and then identify objects. Since no provision is made in a horizontal system for passing information back down this chain, the system cannot make use of context information obtained from the image itself.

The methods which will be presented here are intended to fit into a vertical system in two ways. First, they can be used to start off a vertical system with information good enough to get it going. Second, they extract features which are useful for object identification. These features will be extracted in such a manner as to allow easy advantage to be derived from context information.

This work is intended to be a step towards making computers see. This goal is interesting for a number of reasons. Computers with vision would be useful for applications in automation, and would be able to interact better with humans. Computer vision may well provide instructive models for the understanding of human vision. The problem is also very interesting in its own right, as an aspect of the study of Artificial Intelligence.

Chapter 2 Previous Work

Techniques have been investigated which could be applied to smoothly curved objects as a step towards recognition.

2.1 Shape from Shading

It is possible to find a great deal about the shape of a smoothly curved object from a single monocular image, given a knowledge of its surface reflection properties and the position and nature of the light sources. Horn [10] generates curves lying on the surface of the object by an iterative solution of a set of differential equations relating shape to the intensity of image points. Similar methods have been applied to the analysis of lunar topography from Lunar Orbiter photographs [14,5].

This method requires a uniform object surface. Its reflectance must be a smooth function of the angle the surface makes with the incident and exit rays. Any marks on the surface will disrupt the solutions to the differential equations, although very small marks can be

ignored by appropriate averaging techniques.

Since his results depend so heavily upon the object surface properties, Horn [10] has undertaken an investigation of how typical surfaces behave. To within a reasonable approximation, the reflectance of a real object can be considered to be a simple linear superposition of matte and specular components. The specular component refers to the light reflected such that the angle of incidence equals the angle of reflection, as if the surface were mirrored. The matte component is light which is scattered by the surface. The simplest model of a matte surface is a "lambertian" surface, which emits light uniformly in all directions, regardless of the incident angle. The physical effects which produce these two components are different, as shown by the fact that they frequently have different spectral properties. The specular component has the spectral distribution of the incident light, while the matte component, which penetrates the surface more deeply before being scattered, is spectrally distributed as the product of the incident light spectrum and the spectrum of the object pigment.

2.2 Detection of Optical Edges

Much research has gone into the detection and tracing of contrast edges in an image. These edges can be emphasized by differentiation preprocessing operations, such as the gradient or Laplacian.

2.2.1 Plane-surfaced Objects

Edge detection is particularly attractive for plane surfaced objects. Since the edges are straight lines (the intersection of two planes), a determination of the position of the edges completely specifies the position of the plane surface which they enclose, and an edge itself can be located in terms of just a few of its points.

A program by L. G. Roberts recognizes white plane surfaced objects on a dark background [15]. He considers objects which can be put together out of a set of given sub-shapes, such as rectangular parallelopipeds and wedges. The image is first differentiated. Lines are then found in the resulting picture by a multiple-step procedure, first fitting short lines to local areas, eliminating tiny loops, then fitting longer and longer

lines to the shorter ones, and finally generating a least-mean-square line which is taken to represent the original edge.

The next phase is recognition of polygons in the line drawing, followed by the matching of sets of polygons against the possible models. The matching is first done on a straight topographical basis. The two-dimensional projection of a brick, for instance, generally contains three quadrilaterals with one corner point in common. No such point exists on a wedge. Assuming, then, that this point corresponds to the corner of a brick, the program can match the other lines and points in the quadrilaterals to what must then be the corresponding lines and points of the model. A least-mean-square error matrix procedure is then used to find the best brick (in 3-space) which generates the given two-dimensional line drawing. If the least-mean-square error is small enough, the fit is accepted as correct.

When a set of lines are matched by a model, the model can then be projected back onto the line drawing, but now with all of the hidden lines present. The model is now "removed" from the line drawing, which may entail the deletion of some lines, but also may entail the addition of some others. The procedure is now iterated

until all of the lines of the input figure have been accounted for. Thus objects are recognized as being compounded of a number of the basic building blocks.

Roberts depends on a high degree of precision of measurement of the position of the edges, since he uses perspective in an essential way. Unfortunately, his procedure is useless for objects lacking straight line edges. One particularly interesting aspect of Roberts' work is his use of a powerful internal model of the potential object in the image. A similar approach might be useful for scenes consisting of regular smoothly curved objects such as spheres and cylinders, but it is difficult to envision successful results using more amorphous forms.

A program by R. W. Gosper visually locates white rectangular parallelepipeds on a black table. Due to the high reflectance difference between the objects and the background, the outer edges are very clearly defined. (The program also finds interior edges of the object where the contrast between adjacent faces is high enough.) The edges are found by an algorithm which scans in a line perpendicular to the edge, and moves this line along the edge from one end to the other. From the position of the edges in the image, and the knowledge

that the object is rectangular and rests on a table whose position is known, the exact three-space position of the object can be calculated from its hexagonal outline.

Griffith [8] has made a study of the edges of geometrical objects, and has developed a theory of their optical detection in the presence of noise. His system is designed to be easily integrated into a vertical vision system, and includes a high-level line proposer and verifier.

Guzman [9] "parses" a straight-line drawing into its component plane-surfaced objects. His work is notable here in that it depends little upon accuracy of measurement, and is concerned with dissecting a complex scene into individual objects prior to determining their exact position or shape. In these respects it has goals similar to those of this work with respect to smoothly curved objects. Guzman's techniques could in fact be extended to smoothly curved objects, and used to complement the methods discussed in the next chapter; his methods do not depend so much upon the edges being straight, as do Roberts and Gosper.

Different recognition methods call for different degrees of precision in the final determination of line position. Roberts requires high precision, because of

his use of second-order perspective effects. The use of stereo distance determination would also require such high precision. Gosper requires only medium precision. His goal is to actually pick up the block, which only requires locating it to within a centimeter or so. No perspective, stereo, or other second-order effects are used, so the calculated position is not as sensitive to small errors in the line position. The programs of Guzman and Griffith requlre only low precision, except in a few parts which make use of the parallelism of two lines.

2.2.2 Curved Edges

There has been much study of recognition of alphanumeric characters. Black characters on a white background provide high-contrast edges, and some character-recognition programs work by tracing around the character's edge. There has been little edge-oriented research on images derived from three-dimensional objects, and the results of the two-dimensional work has little relevance to this problem.

It is considerably easier to find a straight edge than a curved one, since only two points determine a

straight line, and additional points can then be verified by very sensitive tests. If many tests are positive along a straight line, the existence of the edge can then be asserted with a high statistical confidence, as by Griffith's programs. These techniques can be used only over a short interval for a curved edge.

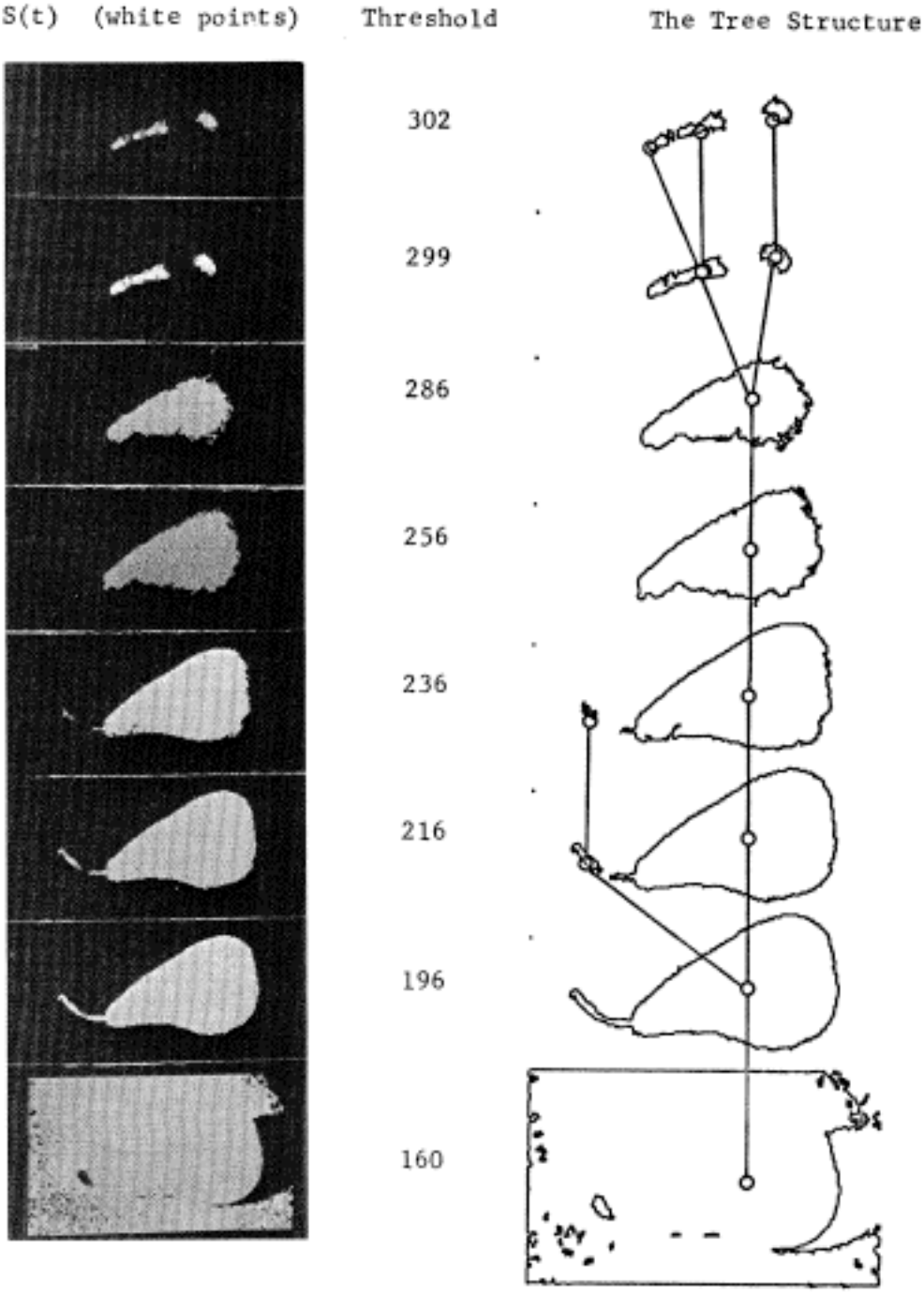
2.3 The "Regions" Approach

Instead of looking for high-contrast edges, some pattern recognition methods look for homogeneous areas of low contrast. Analysis then proceeds from the shape and interelations between these "regions". There are a number of techniques for characterizing the shape of a region, such as various moments [2], or more complicated shape descriptors [3]. Kirsch [11] analyzes photomicrographs of cells by building a tree structure of image regions with various levels of homogeneity. His methods are the closest in the literature to those which are developed in this thesis.

2.4 Textural Information

The optical behavior of an object depends very much on the texture of its surface. The word "texture" may refer to either markings or departures from a smooth surface, but in either case they must be small compared with the size of the object in order to be considered texture. Texure analysis may be done by a wide variety of methods, such as Fourier analysis or cross-correlation. Texture has been used to advantage in a range of studies, in such areas as recognition of terrain types [16] or cell images [13]. Different types of texture will be discussed further in section 3.9.

Figure 3.1: The Intensity-region Tree



Chapter 3 Representing an Image as an Intensity-region Tree

3.1 The Basic Method Used

Consider an image, I, defined on a rectangular raster of points, so that $I(p)$ is the light intensity at the point p. For any given light intensity threshold t, define a set of points $S(t) = \{p \mid I(p) \geq t\}$, the set of points of intensity t or greater. Each of the eight pictures in figure 3.1 (previous page) shows such a set of points, for some threshold. For any t, the set $S(t)$ can be partitioned into disjoint connected subsets $R_i(t)$, which will henceforth be called "regions". Thus:

$$S(t) = R_1(t) \bigcup R_2(t) \bigcup - - - \bigcup R_h(t),$$

where $R_i \bigcap R_j = \underline{\Phi}$ if $i \neq j$, and each R_i is a connected set of points. Note that $S(t_2) \underline{\subseteq} S(t_1)$ if $t_2 > t_1$, so each region at threshold t_2 must be a subset of some region at t_1. The regions thus fall naturally into a tree structure based on this subset relation, as shown in figure 3.1.

Another particularly graphic way of looking at the tree is to visualize the intensity function plotted

in the form z=f(x,y). Slicing this function with a horizontal plane at several threshold levels, the tree can be pictured as in figure 3.2. An intensity contour map of the pear is shown in figure 3.3 in order to show how the regions are actually nested.

3.2 Quantization

Choosing a set of threshold levels $\{t_i\}$ is equivalent to quantizing the light intensities in the image, in terms of the information retained in the tree. The more threshold levels in the set, the greater the depth of the tree generated using these levels. We will generally consider threshold sets which are evenly spaced in the log of the light intensity, although a tree could be generated from any arbitrary set of levels. Using the log of the light intensity generates a tree whose structure remains basically the same if the illumination is scaled up or down by a constant factor.

3.3 Geometry of the Tree

In the limit of a continuous tree (in which the spacing between threshold levels approaches zero), the

Figure 3.2: The Region Planes Shown as Slices of the Intensity Function

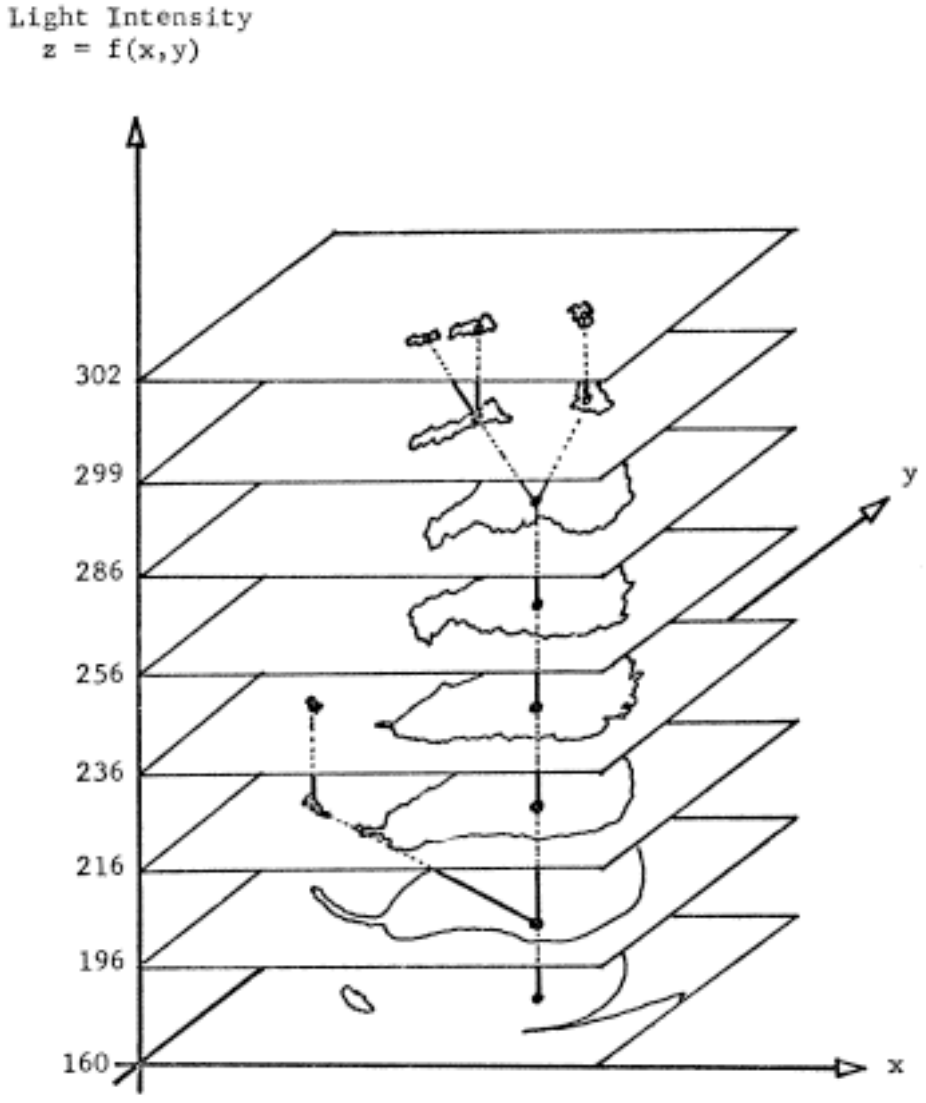
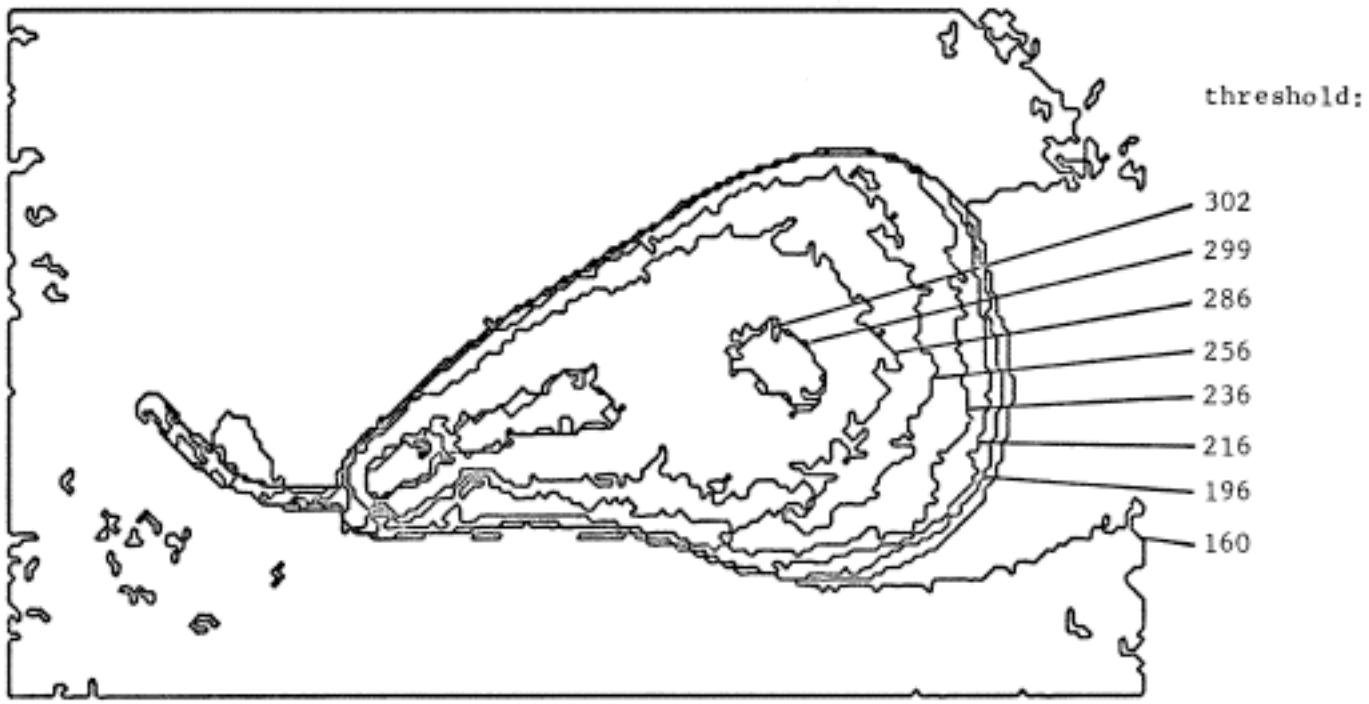


Figure 3.3: A Contour Map of the Pear

The same threshold levels are used as in figure 3.1



tips of the branches represent local maxima in the image. Beginning at a branch tip and moving along it in the direction of lower intensity, the region expands from the maximum point to include other nearby points, assuming the intensity function is continuous in that area. Each tree branch can thus be thought of as a growing region. A fork in the tree occurs whenever two or more of these regions combine, forming one new larger region. In this case, the branch associated with the sub-region of largest area shall be considered the "main branch", and the other branches shall be called "sub-branches". If the original image is slightly noisy, then as a region "expands" (moving along a tree branch from high to low intensity), it will engulf large numbers of smaller regions which appear ahead of its advancing edge, resulting in many short sub-branches on the tree. When two regions of substantial area are combined, it is not really important which is considered the sub-branch.

The highest region on the tree represents the brightest point in the image. If the threshold is lowered far enough, all of the regions will eventually merge into one region containing all of the image points. This shall be referred to as the "root" of the tree.

3.4 Trees with Incomplete Region Information

In the preceeding discussion, the regions themselves have been considered to be the elements of the tree. Let us now consider an abstract tree structure in which the elements of the tree are not the regions themselves, but nodes containing information about these regions. Such a tree shall be called an "Image Tree". If each node contains a complete description of the region to which it corresponds (that is, if $R_i(t_j)$ is given for all i and t_j), then the tree contains enough data to be able to re-construct the image exactly, to within the limits imposed by the quantization.

If each node contains only statistics of the corresponding region, rather than a complete description of the region, then the tree contains less information than the original image. These are the interesting trees, despite the fact that the image cannot be reconstructed from them. The problem of pattern recognition can be viewed as one of throwing away information in a selective way. To go from a picture of an apple to the word "apple" represents an enormous reduction in information ("a picture is worth a thousand words").

In general, the nodes may contain any arbitrary set of functions of the corresponding region. In particular, the ones which will be used are the position of the region's center of mass (x_c, y_c), the area A of the region (i.e. the number of points in it), and a measure of the second moment about the center of mass, called the eccentricity e.

The eccentricity is defined by

$$e = \frac{2\pi}{A^2} \sum_{\substack{\text{all pts p} \\ \text{in region}}} (x_p - x_c)^2 + (y_p - y_c)^2$$

e is 1.0 for a perfectly circular region, and is larger for a more elongated region.

The eccentricity is a dimensionless quantity, which remains the same if the region size is scaled up or down. It represents a normalized moment of inertia about a line thru the region center of mass perpendicular to the region plane. It can be shown that no region can have an eccentricity less than 1.0, and that any shape other than a circle has a higher eccentricity. This is because a circle has the smallest moment of inertia for a given area.

For a l by f rectangle, the eccentricity is

$$e = \frac{\pi}{6}\left(f + \frac{1}{f}\right) ,$$

which is 1.047 for a square, 1.31 for a 2 by 1 rectangle, and 2.23 for a 4 by 1 rectangle. For a high elongation f, $e \cong \pi f/6$.

Note that this definition of eccentricity is not the standard eccentricity of second order curves. The eccentricity of an elliptical region of semi-axes a and b is

$$e = \frac{1}{2}\left(\frac{a}{b} + \frac{b}{a}\right) ,$$

which ranges from 1 to ∞. The normal definition of the eccentricity of an ellipse is

$$\sqrt{1 - \left(\frac{b}{a}\right)^2} , \quad (b \leq a),$$

which ranges from 0 to 1.

More complex region statistics could be stored on the tree. If the x and y second moments are stored separately, then the "dominent axis" thru the region center of mass can be easily computed. This is a line in the plane of the region points through which the region has minimum moment of inertia. Higher moments could also be computed, although their interpretation in terms of high-level shape descriptors is less clear. More complete shape descriptors, such as the results of a Medial Axis Transform [41] could also be used.

The choice of more complex shape descriptors depends on the particular recognition tasks being performed. The simple statistics of area, center of mass, and eccentricity can yield much useful information, however, and attention will be focused on them. It will be seen that they are quite useful for the analysis of surface properties and simple shapes.

3.5 Sub-programs of the Image Tree System

Programs have been written to obtain the image tree of a given scene. Measurements from a laboratory scene are read into an array by an image-dissector camera, and a list-structure tree is generated. The tree can be printed out, showing the parameters associated with each node. Programs also can graph against the threshold any region statistic stored on the nodes, along some path on the tree from a branch tip to the root. The original image can be displayed, and any arbitrary region can be shown superimposed upon it. For more detail about these programs, see the appendix.

3.6 The Tree of a Matte Sphere

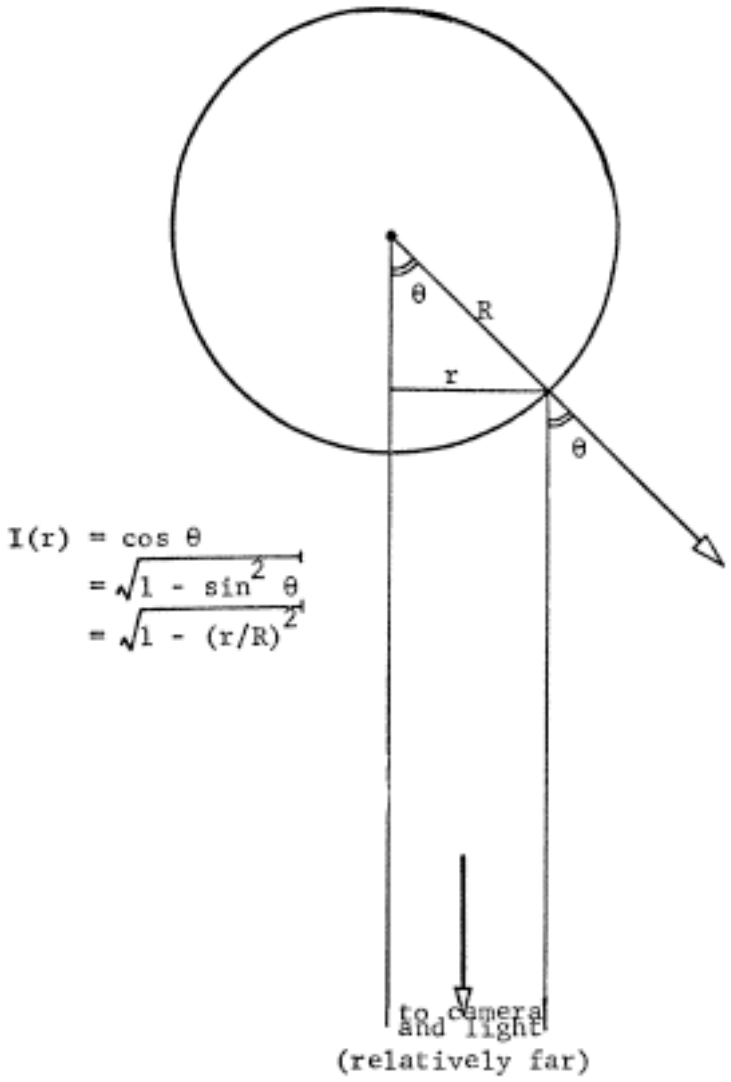
Let us consider the tree resulting from an image of a sphere with a matte surface. A matte surface exhibits a reflectance which is fairly uniform in all directions regardless of the angle of the incident light. The image of a sphere is a circle. If we assume the reflectance to be completely uniform, and consider a sphere lit from the camera position, then the intensity as a function of radius r over this circle is

$$I(r) = [1 - (r/R)^2]^{1/2},$$

where R is the radius of the projected circle, and the intensity is normalized to 1 at the central point. This formula simply expresses the fact that the projection of a surface seen by a viewer is proportional to the cosine of the angle of the viewer from the normal to the surface (see figure 3.4). Thus, assuming uniform scattering, the intensity of the light is proportional to the cosine of the incident (and viewing) angle. The intensity value actually read from the vidisector is $t = C + 32\mathrm{Log}(I)$, where C is the reading at the central point, I is the intensity, and the Log is base 2. Solving for the region area as a function of the threshold t, we get

Figure 3.4: Formula for the Reflectance of a Sphere

The sphere is lit from the camera position.



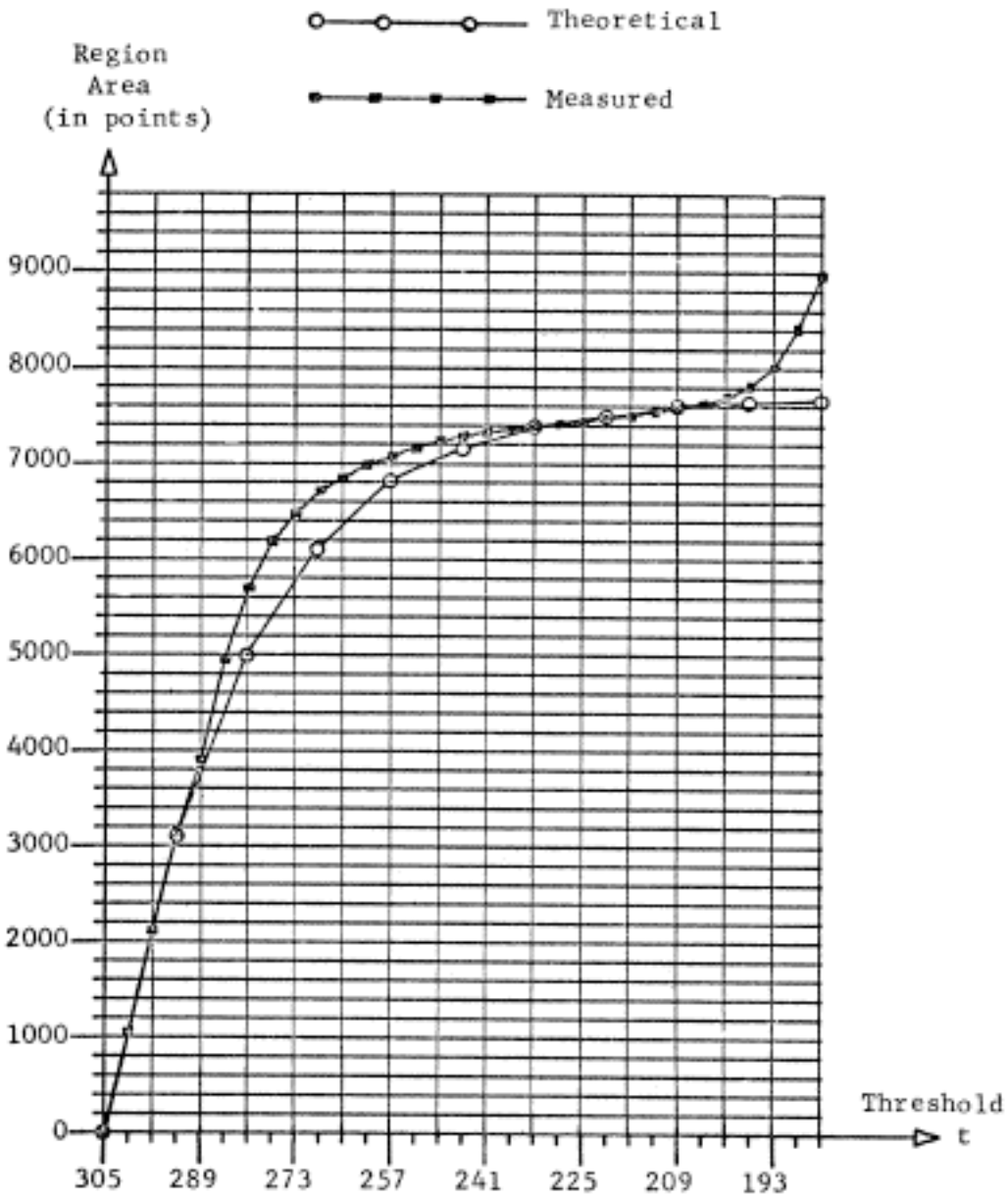
$$A = B\left[1 - 2^{(t-C)/16}\right],$$

where B is the area of the full circle. Its image tree should have only a single straight branch, whose tip corresponds to the central point. Each of the nodes on this branch represents a circular region centered about this point.

A picture of a white sphere on a black background was actually read into the computer from the vidisector, and a tree was generated by the procedure previously described. The tree had essentially one main branch, although there were a few very short sub-branches representing regions of very small area, which were neglected. The measured region area and the theoretical curve are plotted together in figure 3.5.

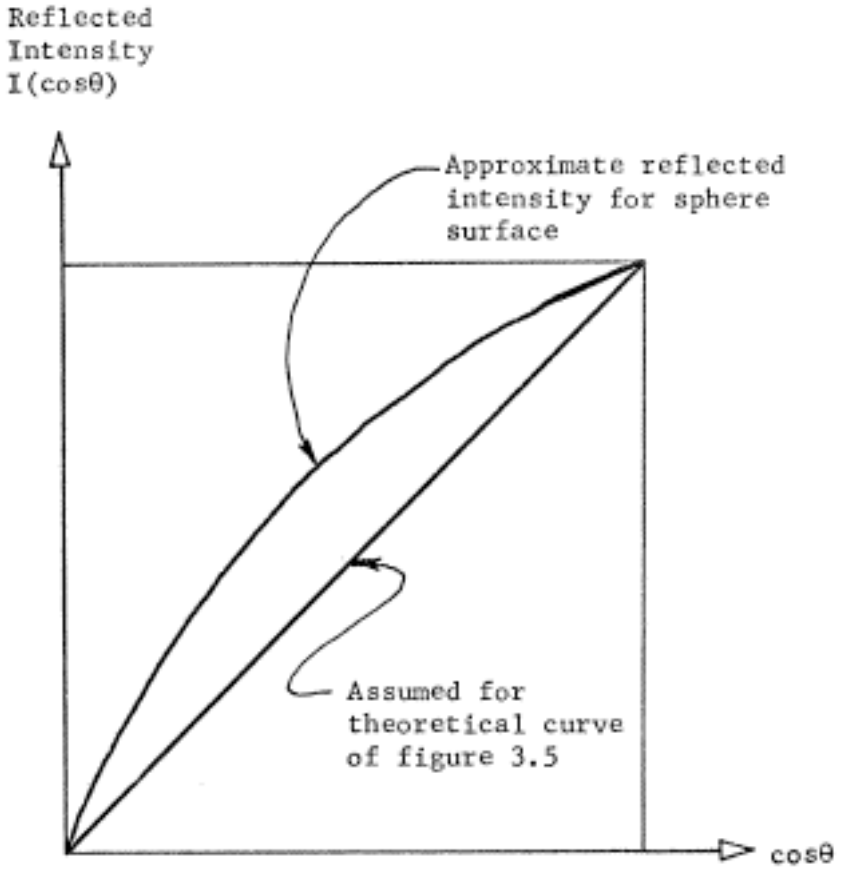
Note that the measured curve rises considerably above the theoretical curve in the central region. This implies that the intensity is not linear in cosine of the incident angle, but is somewhat convex, as in figure 3.6. The sphere used for these studies had an extremely matte surface, and hence a negligible highlight. The sudden rise at the end of the curve is due to the threshold lowering to below the intensity of points in the black background.

Figure 3.5: Region Growth for a Sphere



Theoretical curve: $A = 7720\left[1 - (10^{0.0188(305-t)})^{-1}\right]$

Figure 3.6: Actual and Assumed Surface Reflectance



3.7 Effect of the Specular Component

As was discussed in chapter 2, the reflectance of a surface can be considered to be a superposition of a specular and a matte component. A mirrored sphere would give rise to a pure specular reflection, which would clearly be an image of the light source, plus a reflection of anything else in the room. If the surface is not highly mirrored, this specular component will be greatly attenuated, so that it can be neglected, except for the image of the bright light source, which will be significant despite the attenuation. This reflection of the light source is called a "highlight", and will generally be considerably brighter than the surrounding points. The magnitude of this highlight relative to the matte component is a measure of the specularity of the surface.

Consider the effect of this highlight on the image tree, assuming the light to come from a small (nearly point) source. This will produce a small, bright spot on top of the local maximum in the matte component. As a result, a long section of the tip of the tree will represent a small region of fairly constant area. This is a result of the "spike" in the light intensity

function resulting from the small, bright highlight.

Consider the set of spheres shown in figure 3.7. They were all painted with a matte white paint, and then coated with zero through seven coats of clear enamel, giving them varying degrees of specularity. A graph of the region area vs. threshold (figure 3.8) shows the small flat section of the curve representing the highlight, for one of the spheres. Figure 3.9 gives this highlight depth h as a function of the number of coats of laquer, illustrating how the surface specularity can be measured in a simple manner. The irregularities in this curve are probably due to the difficulty in applying the coats of laquer uniformly.

3.8 The Surface Convolution

Locally, consider a curved surface to be a part of a sphere of the same radius of curvature. According to classical optics, a spherical mirror has a focal length of one half its radius R, and will form a virtual image of the light source as shown in figure 3.10. If a light of diameter d and distance L from the object is not too far off the camera-object axis, then the diameter of its image is about

Figure 3.7: Specularity Test Spheres

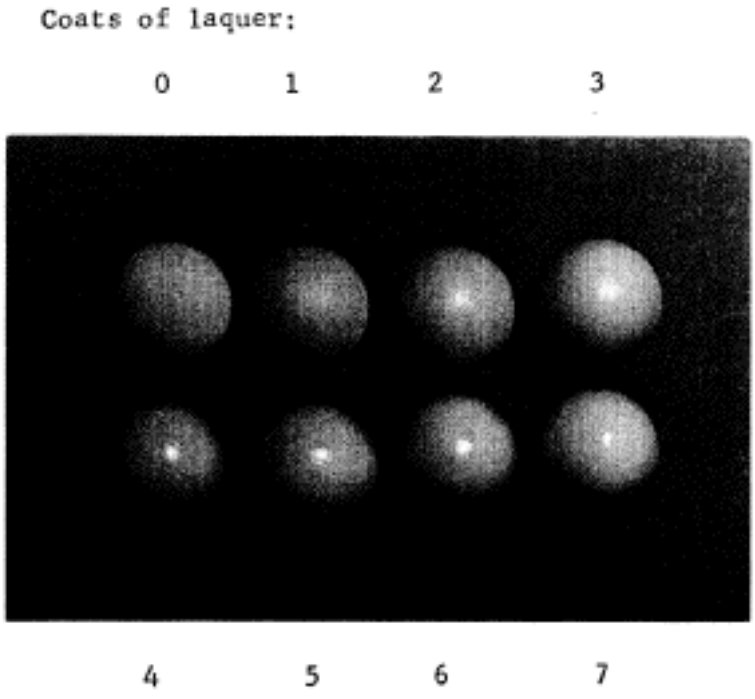


Figure 3.8: Illustration of Highlight Depth

Graph is for sphere 7 of figure 3.7

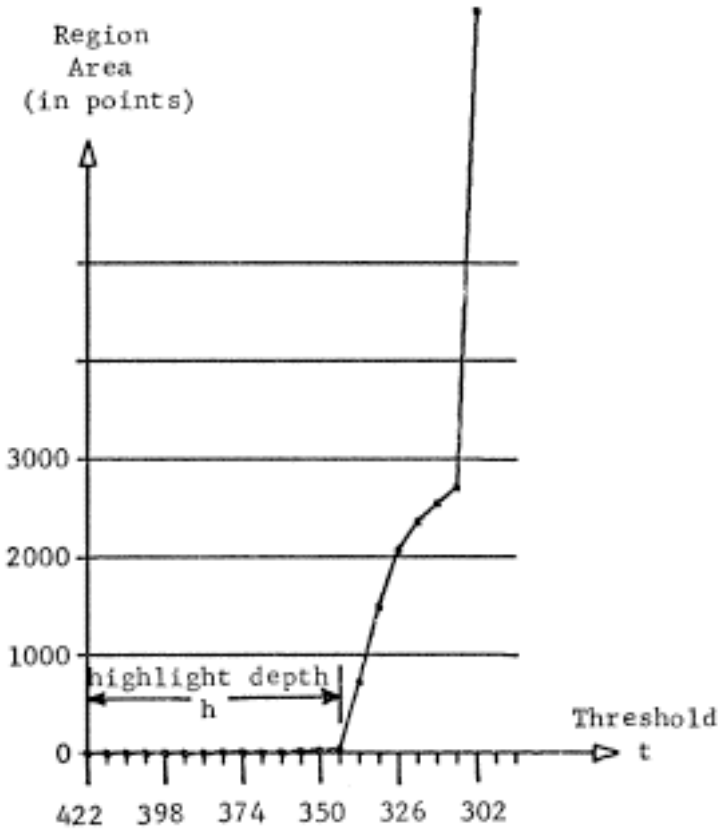
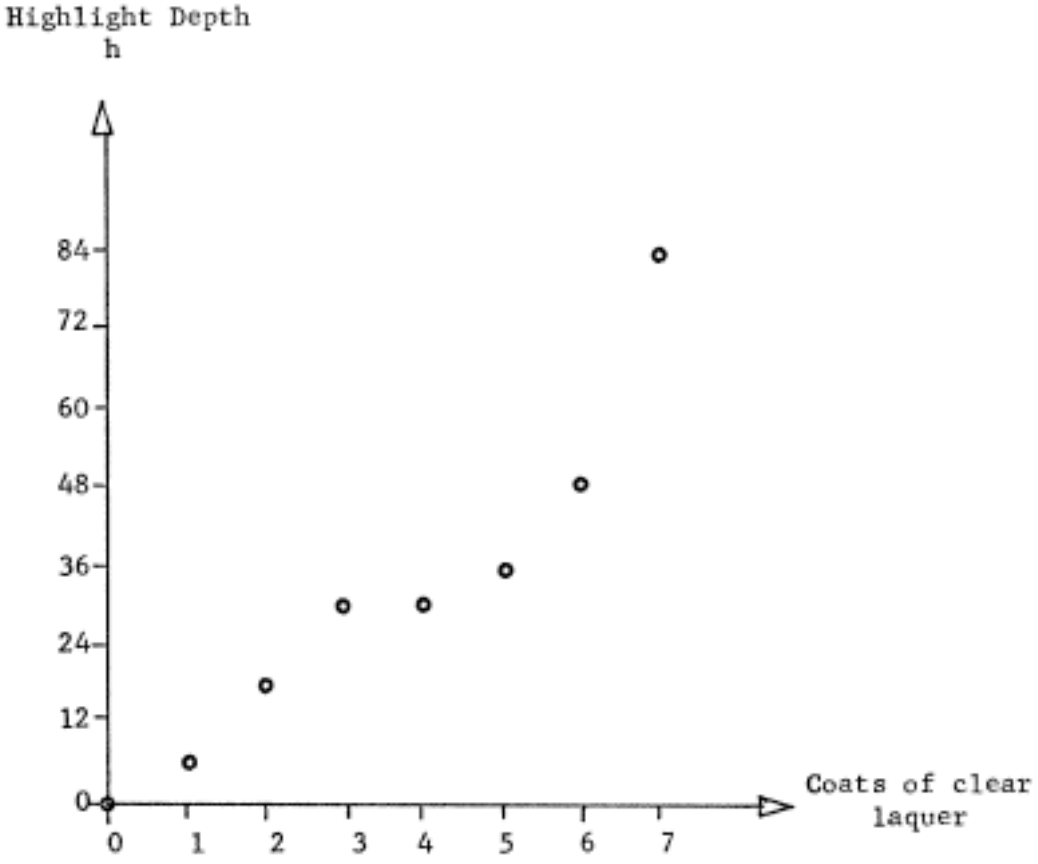


Figure 3.9: Highlight Depth vs. Number of Laquer Coats



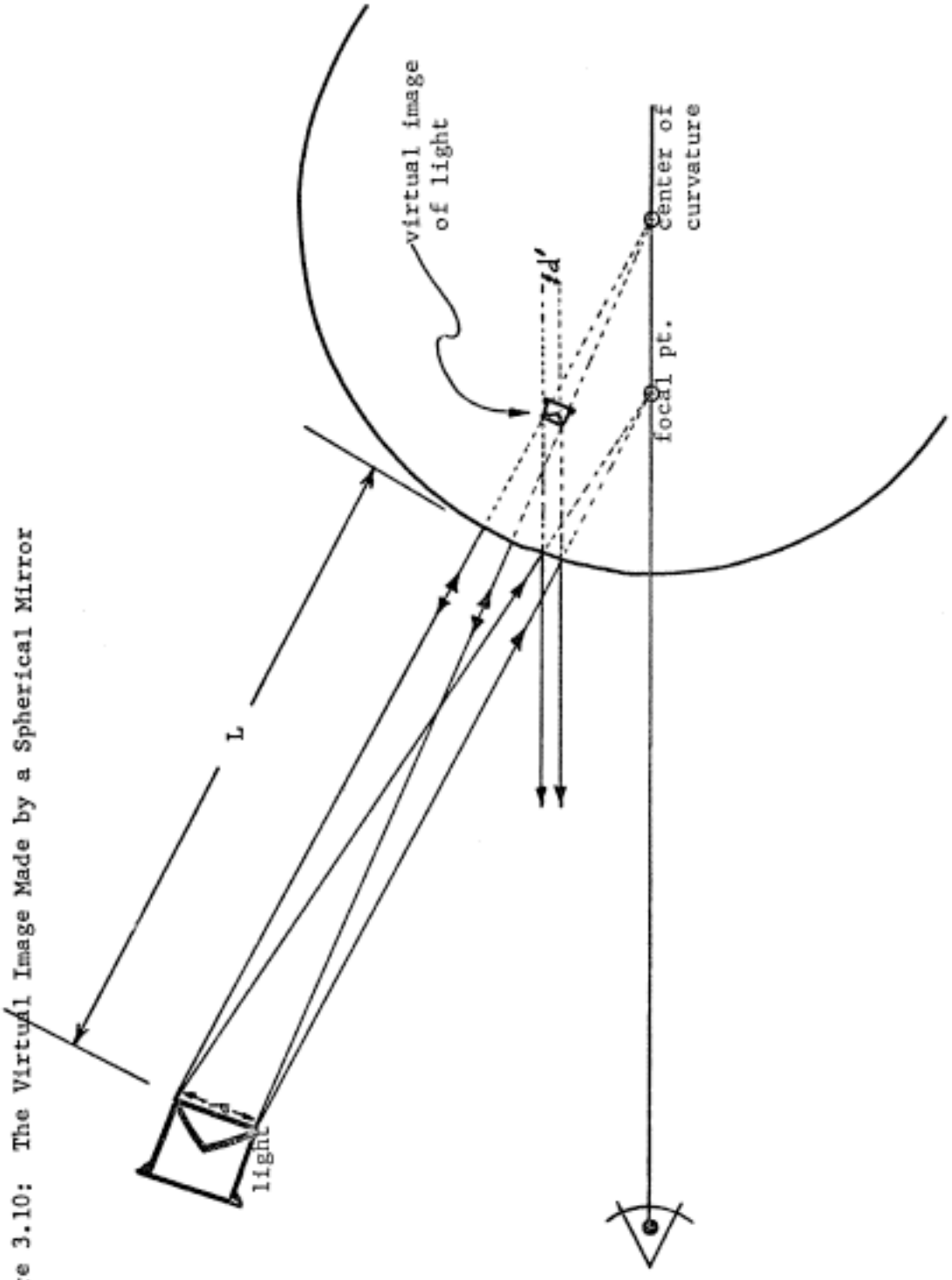


Figure 3.10: The Virtual Image Made by a Spherical Mirror

$$d' = d\left(\frac{1}{1 + 2L/R}\right)$$

(As $R \rightarrow \infty$, $d' \rightarrow d$, as is indeed the case for a flat mirror.) Thus if the size of the light source and the approximate distance of the object from the camera are known, the curvature of the surface can be determined near a highlight. Even if the size of the light source is not known, this method gives the relative curvatures if there are several different highlights in the scene. A good way to determine the size of the source is to take advantage of verticality by knowing the approximate curvature of some object in the image.

Many surfaces will "smear out" the image of the light, resulting in a broader highlight than would be gotten from a mirrored surface of equivalent curvature. The highlight seen can be considered to be the convolution of the image of the light source and the "impulse response" of the surface reflectance. If the light source is a sufficiently small point, then its image can be considered to be an impulse, and the surface "smear" function can be read directly from the region area vs. threshold curve.

3.9 Texture

"Texture" refers to variations in the light intensity which are very small in size compared to the objects being recognized. It has two basic causes. "Visual texture" is due to variations in the reflectance of the surface, and "tactile texture" is due to minute protrusions or depressions superimposed upon a basically smooth surface (the sort of texture one can feel with a finger). If the size of the texture is smaller than the resolution with which the image has been sampled, the intensity variations will average out, and the texture will have little effect on the tree, aside from affecting the surface "smear" function. If the texture is large enough to be discernable, however, it will produce a distinctive effect on the tree.

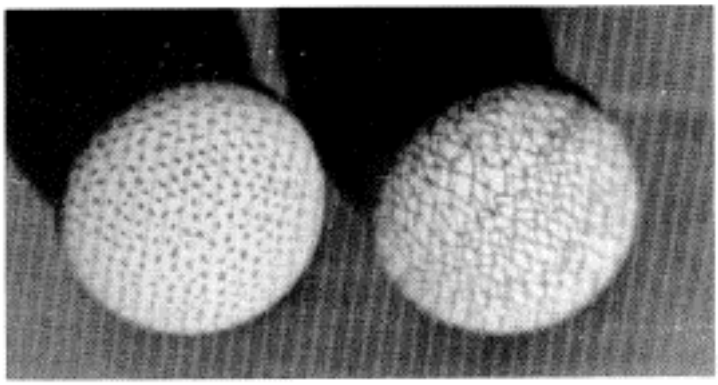
Texture is a multi-dimensional feature, and there are a correspondinly large number of textural properties which could be measured. We are not concerned here with producing a complete description of texture, but rather with detecting features which might be useful in making an object identification. Although such features can help discriminate between objects, they do not give enough information to re-construct the texture exactly.

3.9.1 Visual Texture

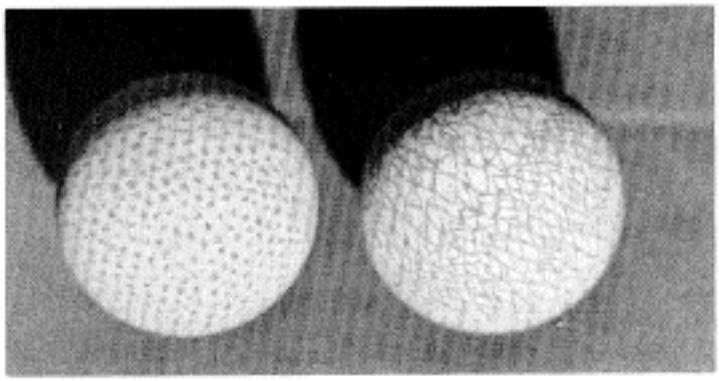
Consider the two spheres shown in figure 3.11. The spheres were painted with a matte white paint, then marked with red ink to produce visual texture. The same two spheres are shown in red, white, and green light. Since the red ink is highly reflective in the red, and very absorptive in the green, these lighting conditions produce light, medium, and heavy texture contrast respectively, with all other factors being held constant.

There are two kinds of texture, with respect to effect on the image tree. The right sphere shows small disconnected light patches on a connected dark background, and the left sphere shows disconnected dark speckles on a connected light background. A light spot, being a local maximum in the light intensity, will produce a tree branch. The nodes on this branch will represent regions the size of the spot, and so will have very small area. The length of the branch will depend on the relative brightness of the spot compared to its neighbors, since when the threshold reaches the intensity of the neighbors, the region corresponding to the spot will be swallowed up by the larger region surrounding it.

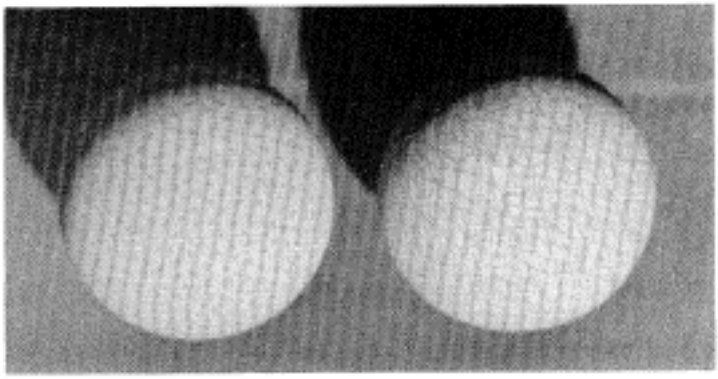
Figure 3.11: Texture Test Spheres



High contrast
(green light)



Medium contrast
(white light)



Low contrast
(red light)

Light speckles will thus produce a large number of sub-branches whose length represents the intensity of the speckle, and whose "size" (the size of the corresponding regions) represents the size of the speckles. The tree corresponding to the light-speckled sphere photographed in the green light (deepest texture) is shown in figure 3.12. Note the many branches produced by the speckles.

The number and length of the sub-branches provides a measure of the degree of contrast of the texture. These quantities are shown in figure 3.13 for the light-speckled sphere under the three lighting conditions. Note how these quantities thus provide an index of texture contrast, just as the highlight depth and surface smear function provide an index of specularity. Information about the details of the texture can also be obtained, up to the limits imposed by the particular shape descriptors used on the nodes of the tree. Round speckles will produce regions of low eccentricity, whereas streaks will produce regions of very high eccentricity. If the direction of the dominent axis of the region were recorded (corresponding to recording the second moments in the x and y directions separately), the dominent axis of the streaked texture

Figure 3.12: Tree of the Light-speckled Texture Test Sphere (green light)

All sub-branch nodes represent regions of small area.

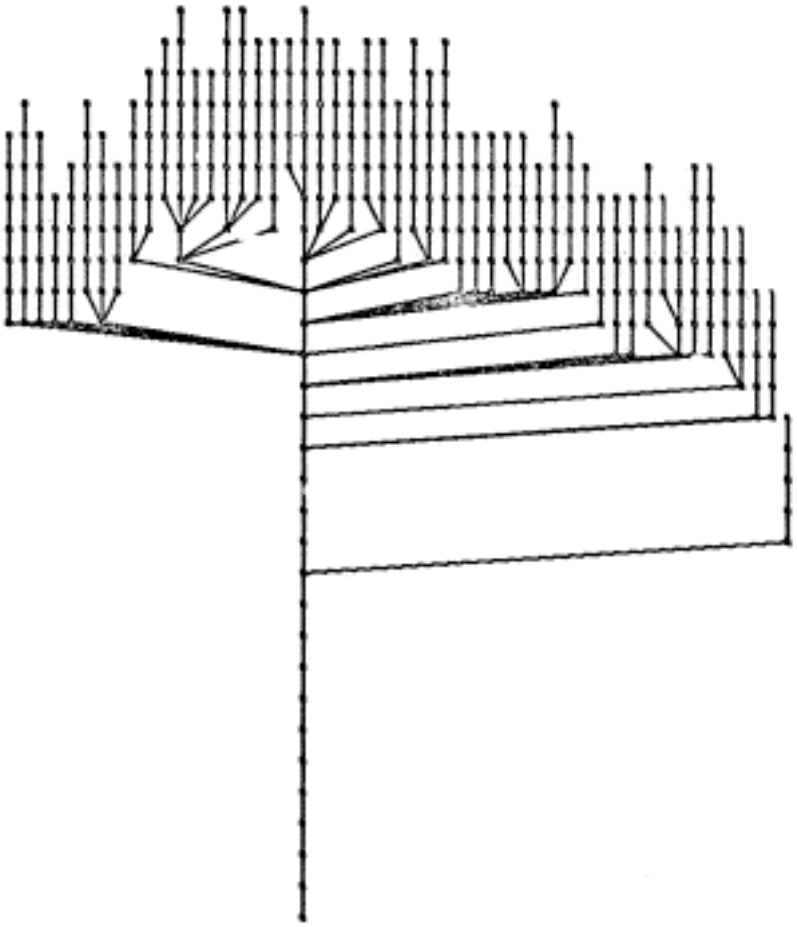
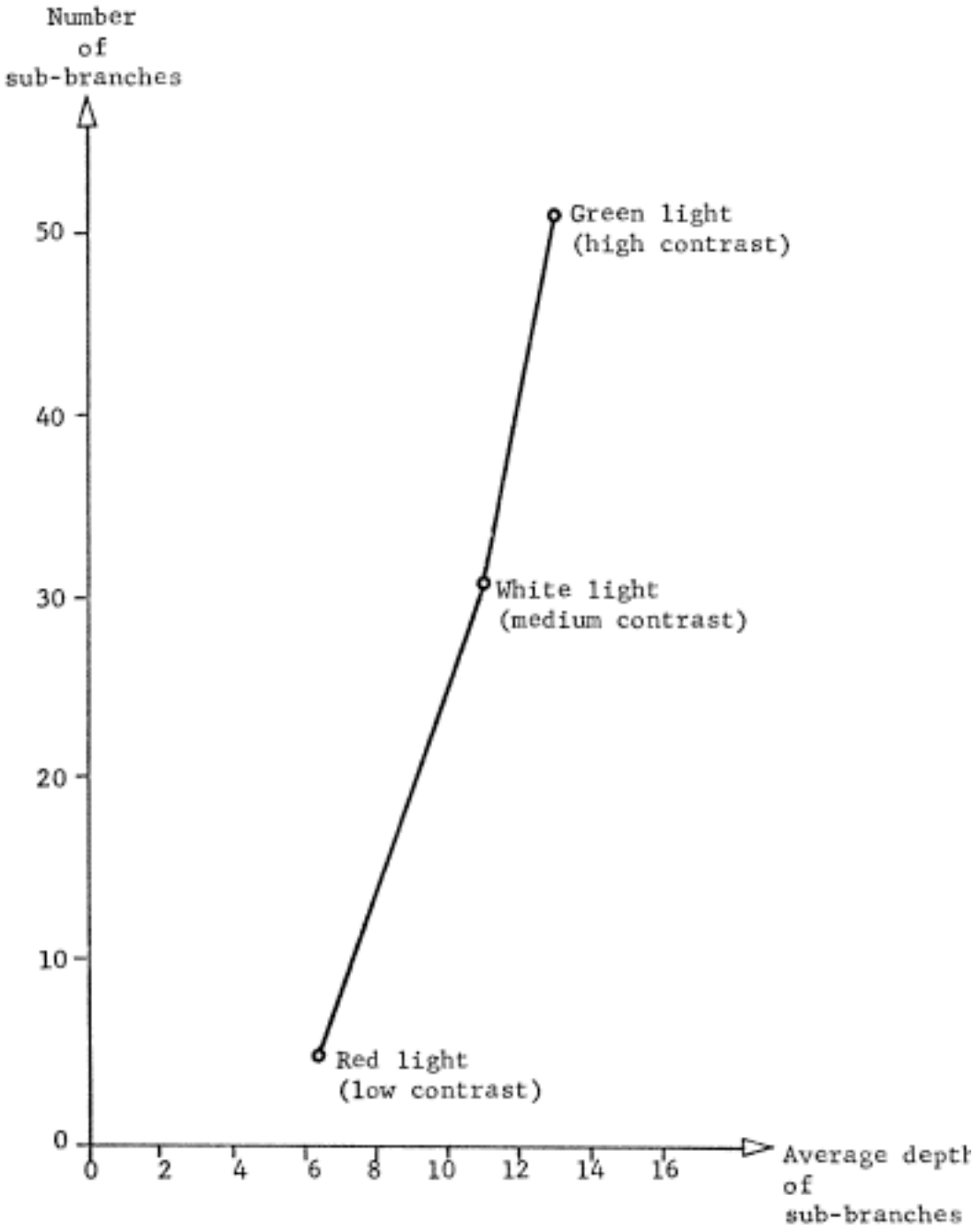


Figure 3.13: Number and Average Depth of Sub-branches for the Light-speckled Texture Test Spheres



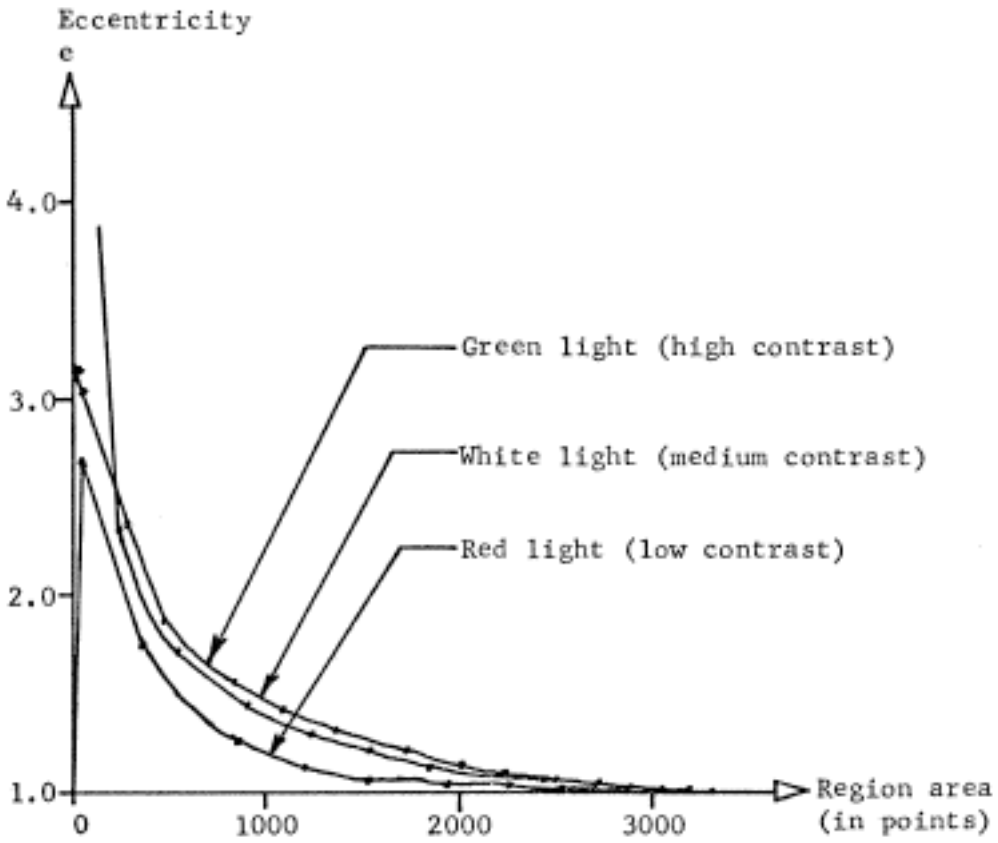
could be determined as well.

Dark speckles will have a different effect, however. Since they are local minima in the intensity function, rather than local maxima, they will not produce branches on the tree, but rather will produce holes in regions. This is shown by the tree of the dark-speckled sphere, shown in figure 3.14. The only effect of these small holes is to raise the eccentricity of the growing region, as shown in figure 3.15, which shows the main branch eccentricity vs. region area for the dark-speckled sphere in the three different colored lights. Since the eccentricity change is so small, these three curves can be compared in this way only because all factors except the degree of texture were held absolutely constant - the same sphere was viewed from exactly the same camera position and with exactly the same light source. Nothing was moved; only the filter over the light was changed.

The difference between the trees for the dark speckled and the light speckled spheres (figures 3.12 and 3.14) exposes a basic asymmetry in the image tree with respect to light and dark. This asymmetry is not just confined to texture, of course. Locally bright areas will always produce regions and hence tree nodes, while locally dark areas will always produce holes in regions,

Figure 3.14: Tree of the Dark-speckled Texture Test Sphere (green light)

Figure 3.15: Eccentricity vs. Region Area for the Dark-speckled Texture Test Spheres



altering the statistics of nodes that would otherwise exist anyway.

The tree could easily be extended to find dark speckles by generating an "inverted" tree for the area inside each region. An inverted tree is a tree in which the regions represent image areas less than threshold, instead of greater than or equal to. This will be further discussed in section 5.4.2.

3.9.2 Tactile Texture

Small bumps on the surface of an object essentially produce many tiny "micro-objects" with the same surface properties. If the size of these is below the resolution of the image sampling, the effect will be only on the surface smear function. If the texture is larger than that, and the surface is fairly specular, the result will be many tiny highlights, producing the equivalent of a light-speckled visual texture.

3.10 Shape

The image tree carries shape information in two ways: In its form, and in the behavior of the region

statistics stored along its branches. The interpretation in terms of object shape of the simple region statistics discussed so far depends upon the object being simply shaped, since the eccentricity does not give enough information to distinguish between different complex-shaped regions. Nevertheless, much useful shape information can be obtained even with very simple statistics, particularly in a recognition-oriented application in which there can be restrictions on the shapes considered.

3.10.1 The Main Branch

Consider the object shown in figure 3.16. Its tree is a single main branch, just as in the case of a sphere (a crude contour map is shown in figure 3.17). The simplest indicator of its shape is the eccentricity of the entire object, which is about 1.4, clearly indicating it to be quite elongated. The entire curve of eccentricity vs. threshold is shown in figure 3.18. The flatness of this curve indicates that the region probably doesn't change its shape very much as it grows, and that it has a smooth surface with no significant irregularities. This is not a unique interpretation of

Figure 3.16: A Matte-white Painted Squash

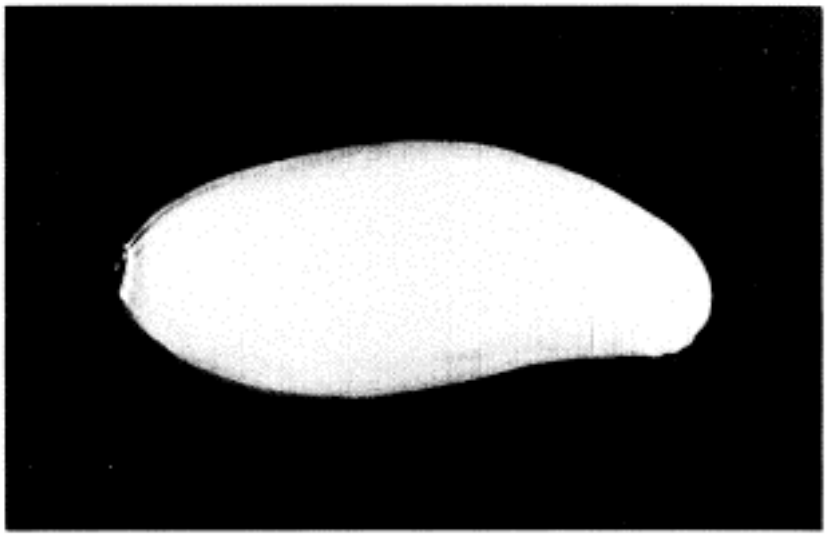


Figure 3.17: Contour Map of the Squash

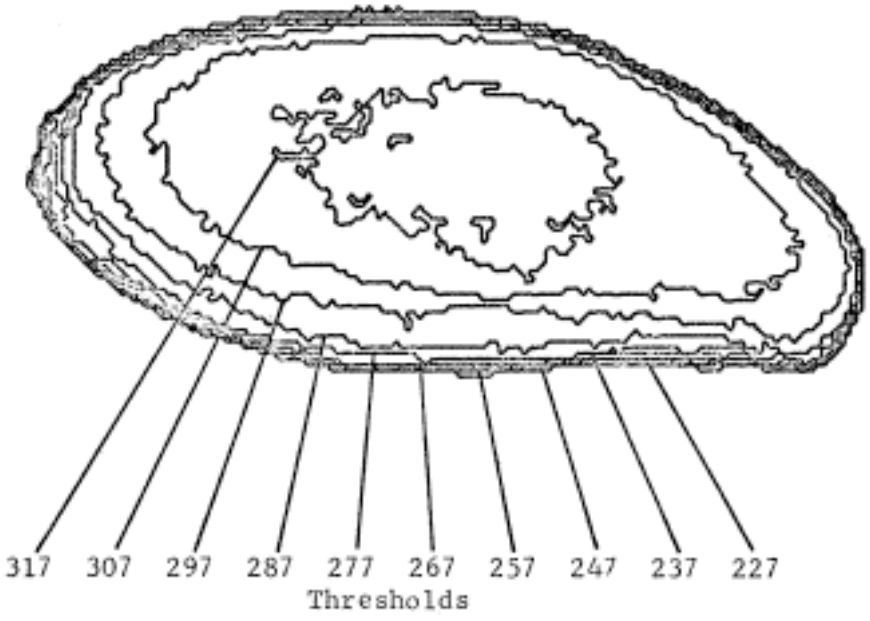
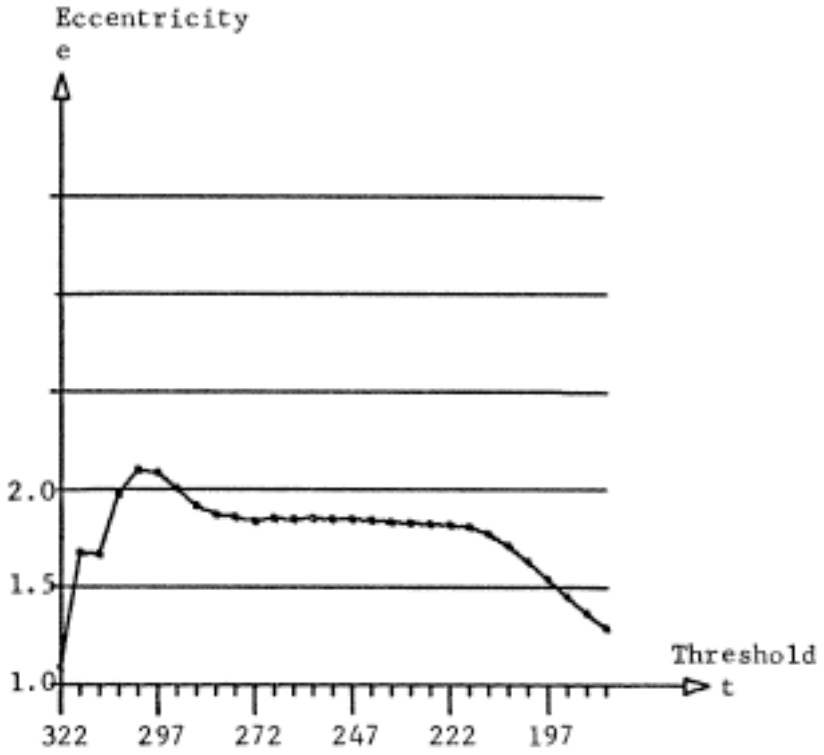


Figure 3.18: Eccentricity Curve of the Squash



the curve, but is a reasonable inference given the assumption that the object is not highly irregular. The bump in the eccentricity curve at the bright end is typical of a small newly developing region. Since the slope of the light intensity function is very small near a local maximum, a small region about that point will tend to have jagged edges, and hence a high eccentricity. As the region expands, the intensity gradient at the edge increases, so the edge becomes straighter, and the eccentricity is reduced.

Consider the plot of added region area, shown in figure 3.19. This quantity shows the excess area added to a region above the sum of the areas of its sub-regions. Since the intensity measured is a monotonic function of the angle of the surface to the camera, the added region area is the projected area of that part of the surface on the object with a particular slope. A bump in this curve represents a large area of relatively low curvature. The only one in this case is near the highlight.

Figure 3.20 shows what the area added to a region looks like - it is the area of a region minus the area of all its sub-regions. Note that the statistics used are such that from the statistics of a region A and those of

Figure 3.19: Added Region Area Curve of the Squash

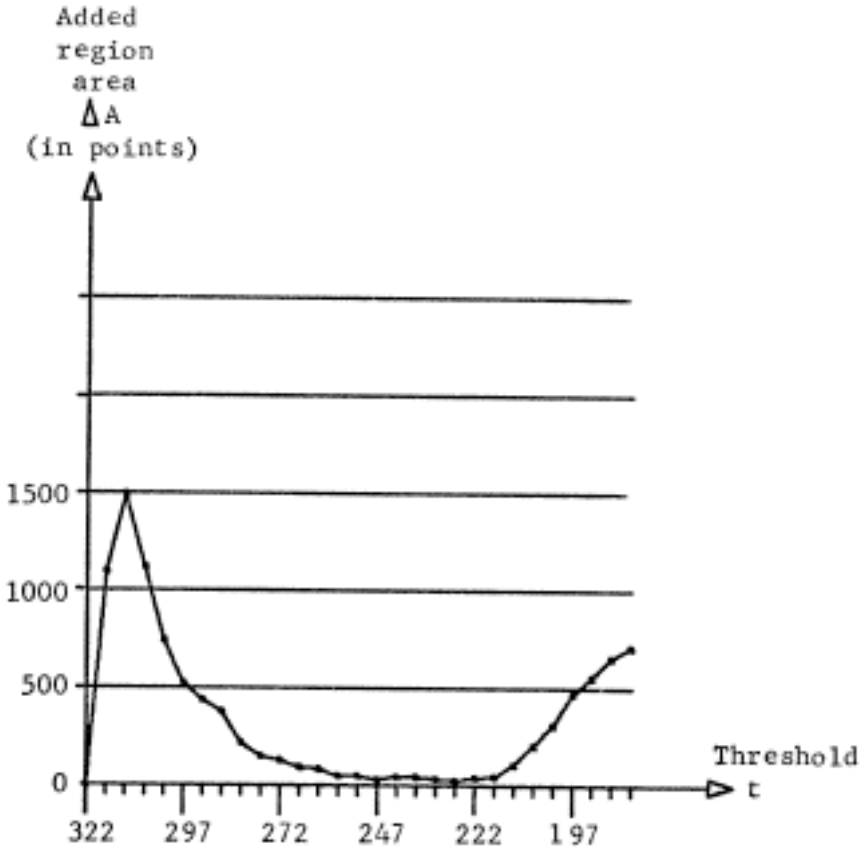
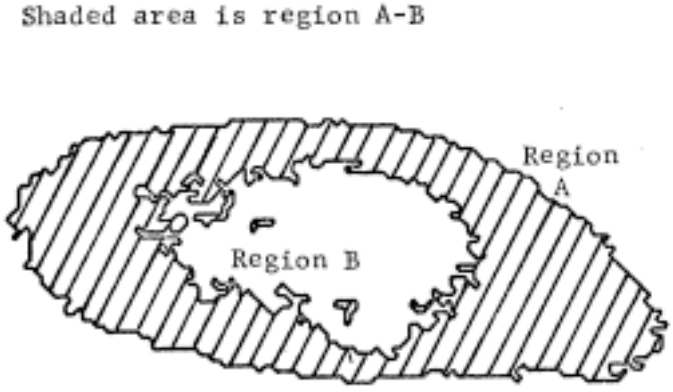


Figure 3.20: Illustration of an Added Area Region



a sub-region B, the statistics of the difference A-B can be computed. (To compute the eccentricity of region A-B, the eccentricity, region area, and center of mass position of regions A and B must all be known.) Computing information about the shape of such a difference region gives information about bulges developing in a region, direction of motion of the center of mass, and other properties of all those points on the surface within some given range of inclination to the camera.

The added area curve would have two peaks for the hypothetical object shown in figure 3.21, due to the low curvature of the annular region indicated. In this case the eccentricity would be constant at 1.0 and the center of mass position would be stationary, since the regions would all be concentric circles due to the rotational symmetry. For the pear-like object in figure 3.22, the protrusion would also increase the added area curve, but in this case, the eccentricity would increase as well, and the center of mass would shift.

Figure 3.21: A Symmetrical Object with Two Added Area Peaks

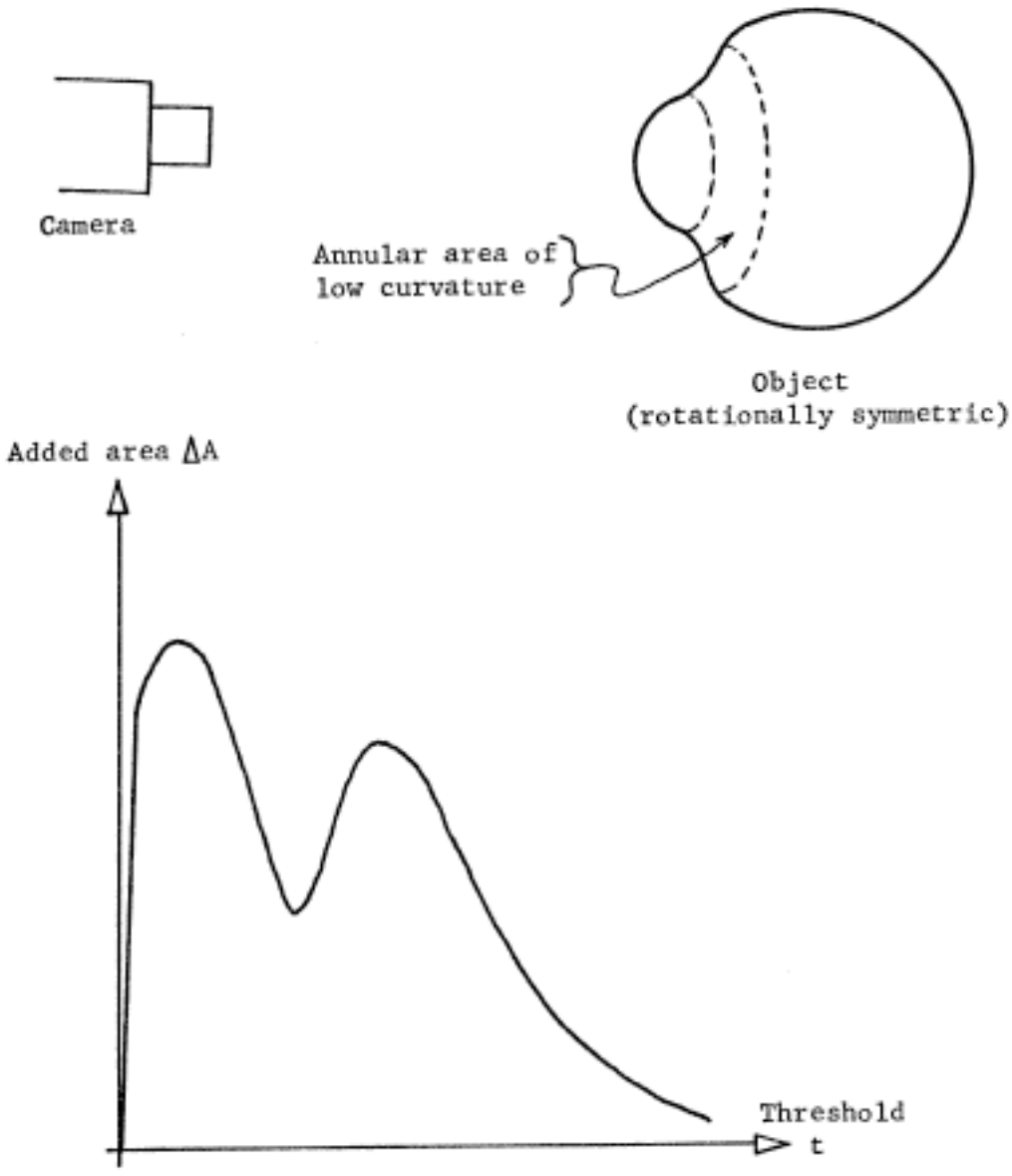
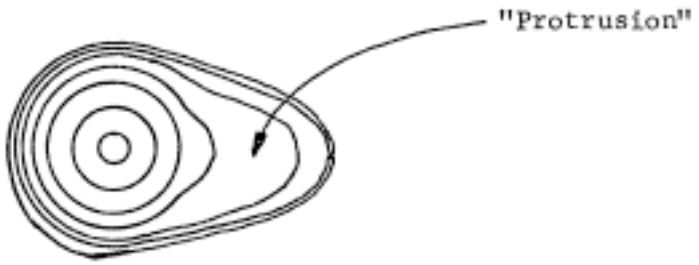


Figure 3.22: A Contour Map of a Hypothetical Object with a Protrusion



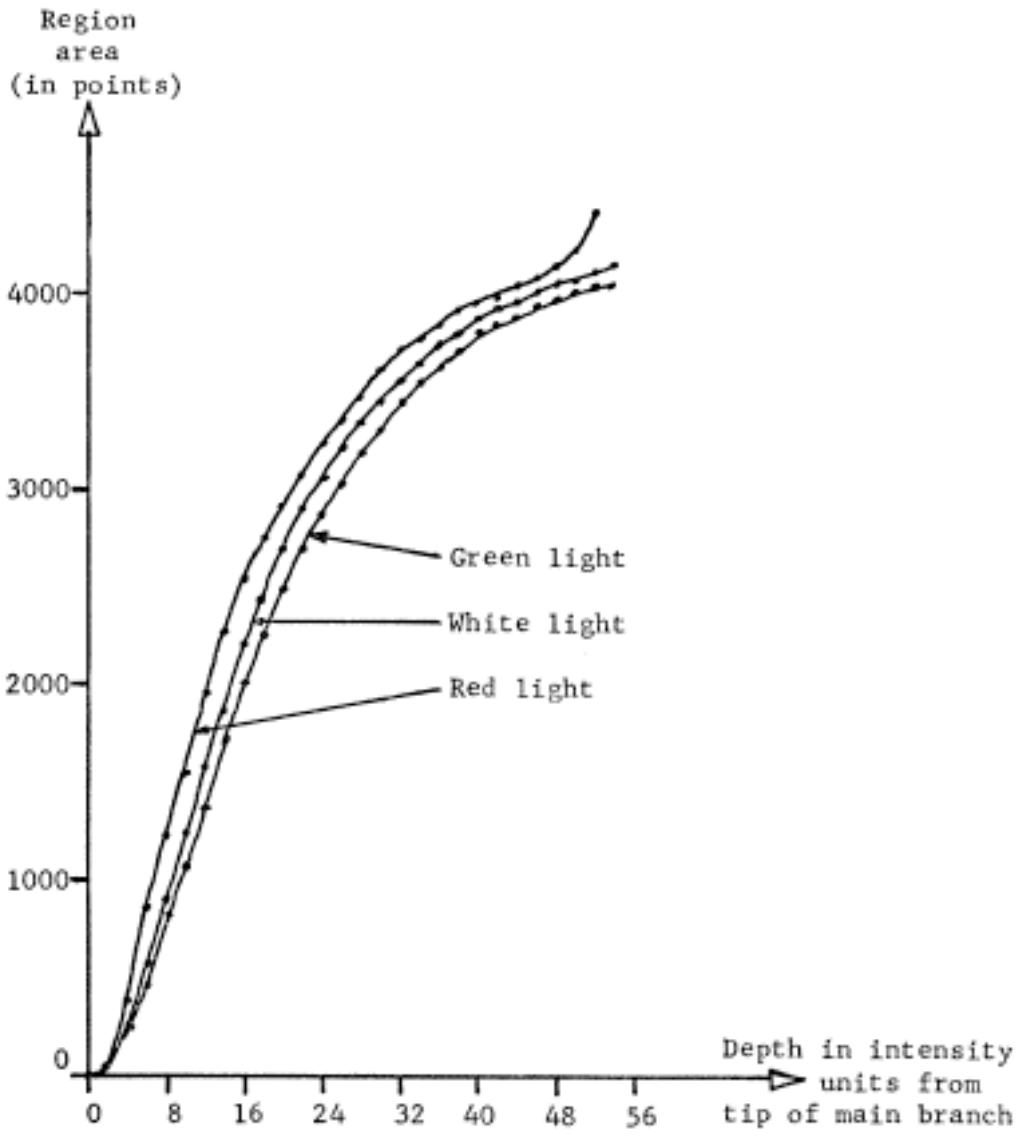
3.10.2 Sub-branches

Protrusions of the sort illustrated in figure 3.22 will often produce significant sub-branches on the tree. The meaning of a sub-branch must be interpreted in conjunction with the information stored on it, and on the main branch to which it attaches. The attachment of a protrusion region, for example, will generally produce a rise in the eccentricity of the main region, and a shift in its center of mass. The possible interpretations of a sub-branch depend very heavily on the particular identification for which the tree is being used. A discussion of the interpretation of shape information for a particular set of test objects will be given in section 4.2.

3.10.3 Non-interference of Texture with Shape

Figure 3.23 shows graphs of the region area for the speckled spheres of figure 3.11, normalized to the light intensity. These graphs illustrate that the basic shape-describing parameters are not affected by object texture in a significant way. This is basically due to the averaging nature of the region descriptors used.

Figure 3.23: Region Area Curves for the Light-speckled Texture Test Spheres



This insensitivity to textural interference is a great improvement over most previous methods used on curved objects, such as Horn's analytical method, which is completely useless in the presence of texture. Edge-finding methods are also confused by sharp texture. This advantage is very important in the recognition of real objects, as will be seen in the next chapter.

Chapter 4 Use on Real Objects

4.1 Pruning

Regions generated by smooth objects with smooth surfaces should in theory always have smooth boundaries. In an actual image, however, minute surface fluctuations and noise will cause the edge of the region to be highly irregular. If the irregularities are great enough, small sections of the region will be detached; that is, they will actually form separate small regions. Since the area separating these small regions from the edge of the nearby large region is only slightly dimmer than the region points, these small regions will join the main region at a threshold only slightly lower than that at which they started. They will thus produce very short branches on the image tree, whose regions are of small area. These regions are essentially artifacts of the particular levels at which the threshold is placed, and thus have no particular significance. In order to avoid the waste of space and time needed to store and analyze these branches, they can be "pruned" away as the tree is generated. This is done simply by removing branches

which are shorter than a fixed length in intensity units and which also represent regions smaller in area than a fixed size.

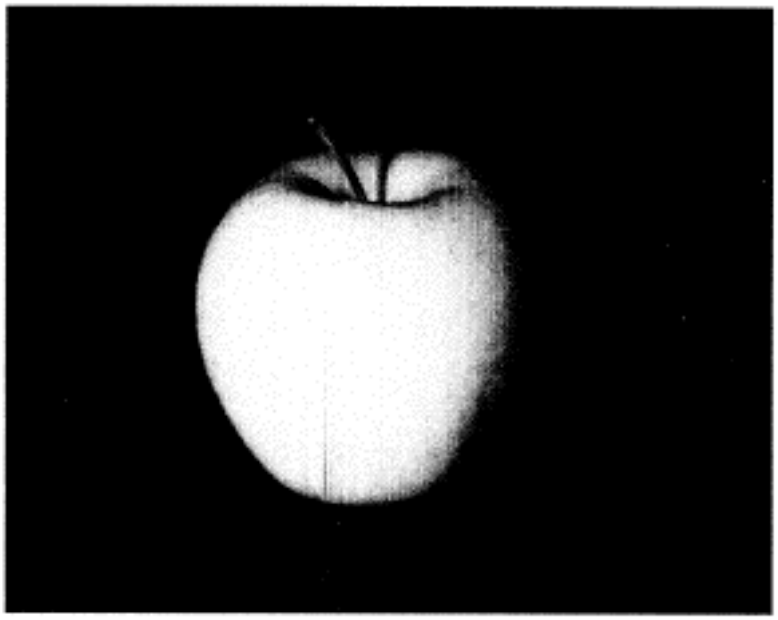
Texture also produces short, small branches, so these thresholds must be adjusted so as not to throw away too much. The goal of pruning is to eliminate all the really tiny threshold-artifact branches without eliminating branches which represent texture. Since the artifact branches will be only one or two threshold levels deep, this can be accomplished by adjusting the spacing of the levels so that one level represents a smaller increment of intensity than the minimal intensity texture to be considered.

4.2 Some Sample Trees

Various types of fruit will be used as examples of fairly simple real objects with smoothly curved surfaces, in order to get a feel for how the image tree behaves on interesting images. Two sample images will now be discussed to give a general idea of what the trees are like.

Consider the apple in figure 4.1. Starting at the brightest node and plotting the region area down the

Figure 4.1: Apple



tree to the root gives figure 4.2. The small flat section of the curve from 384 to 356 is caused by the bright highlight. The region area then begins to grow in a manner similar to the sphere previously discussed. The growth tapers off as the region expands to fill the object. Then, at about intensity 216, the region breaks out of the object into the background.

The region center of mass is shown in figure 4.3. It stays at the highlight for a while, then shifts to the left towards the lit side of the apple. It then slowly moves to the right towards the center of the apple, as the region grows out to the apple's edges. The eccentricity (figure 4.4) starts off fairly high, but rapidly reduces towards circularity, nearly reaching 1.0 when the apple has filled out.

A portion of the tree, displaying only the region area parameter, is shown in figure 4.5. Although the tree is fairly simple, a significant region is added on the the main branch at level 286. The center of mass shows this area to be above the main region, and it in fact is the bright reflection from the back of the stem hollow on the top of the apple, which can be seen in the photograph in figure 4.1. It is very common to have such a branch on the trees of fruits with stems, and higher

Figure 4.2: Region Area of the Apple

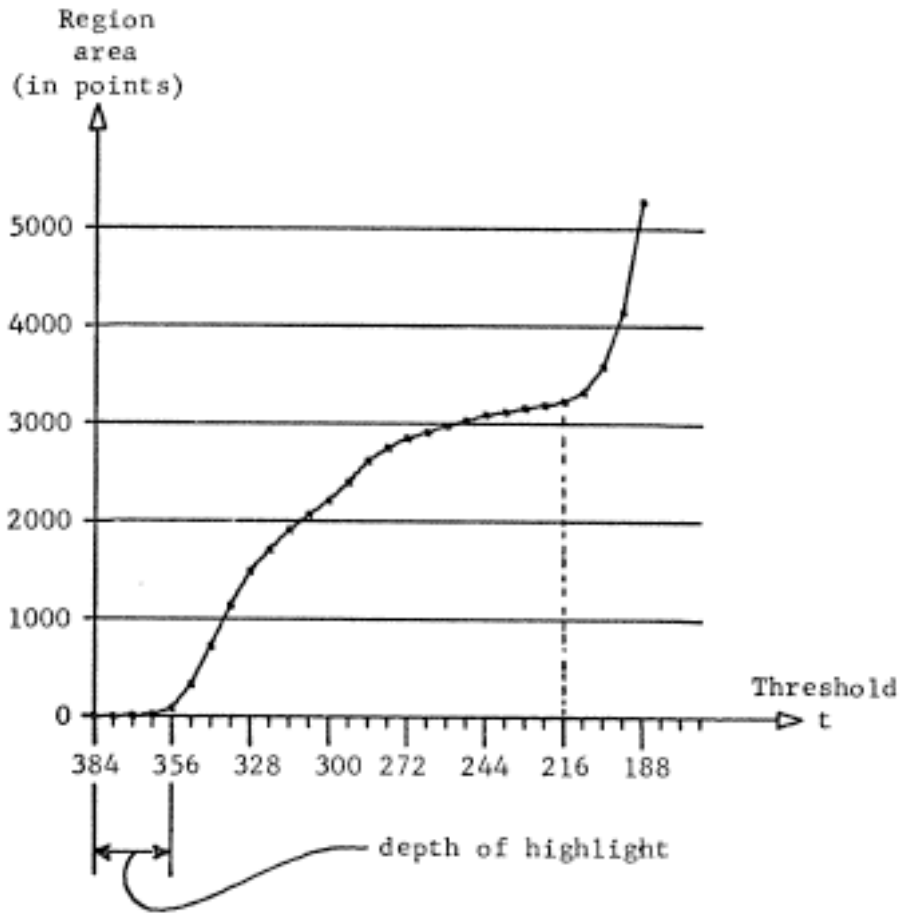


Figure 4.3: Region Center of Mass of the Apple

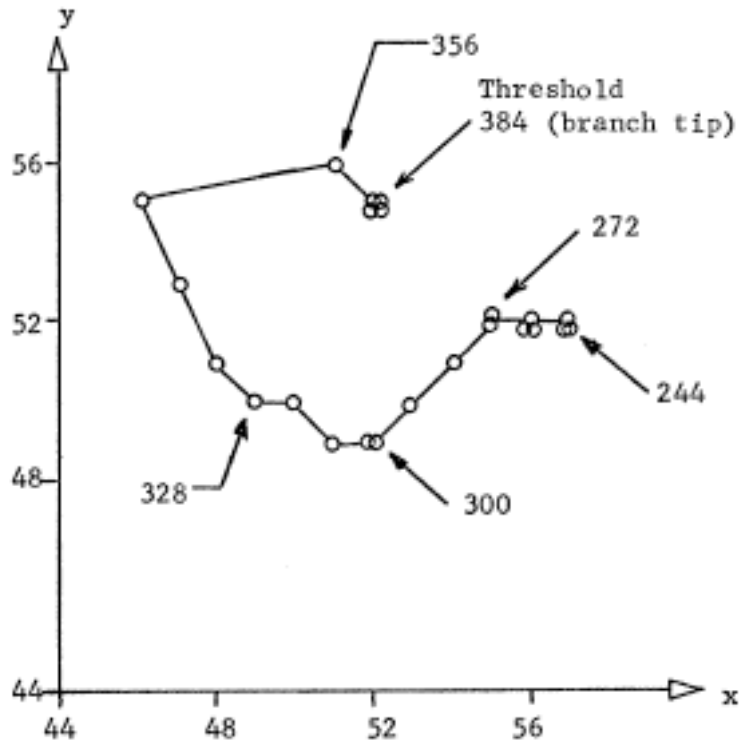


Figure 4.4: Eccentricity of the Apple

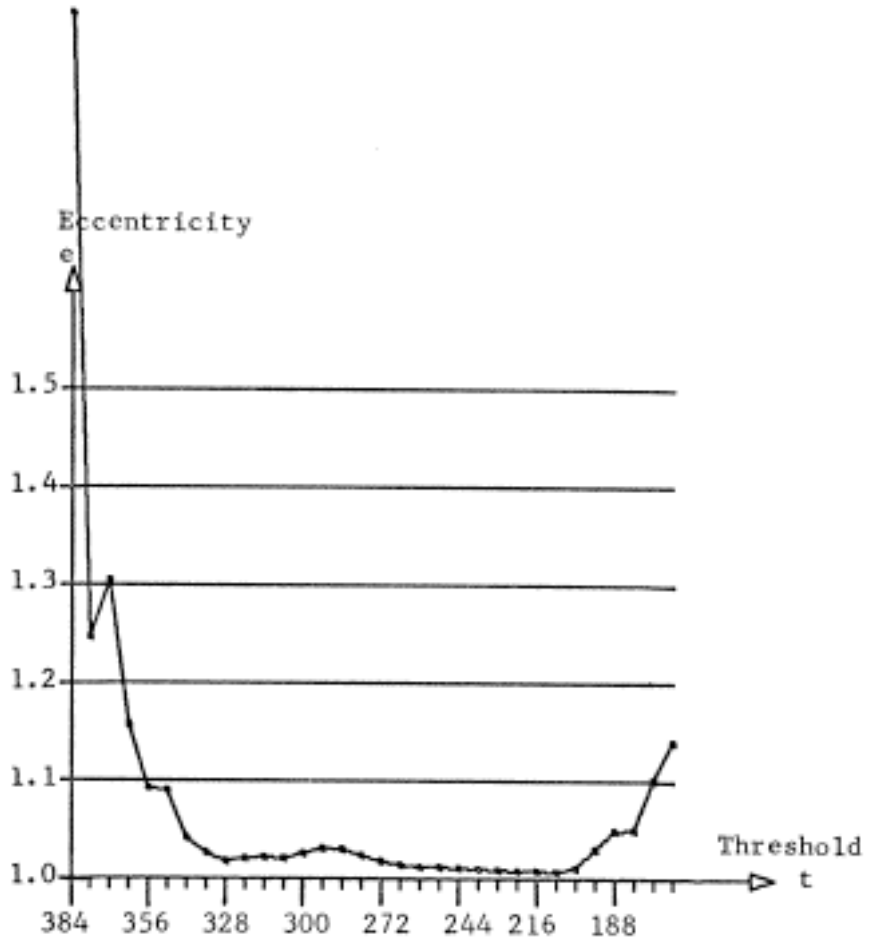
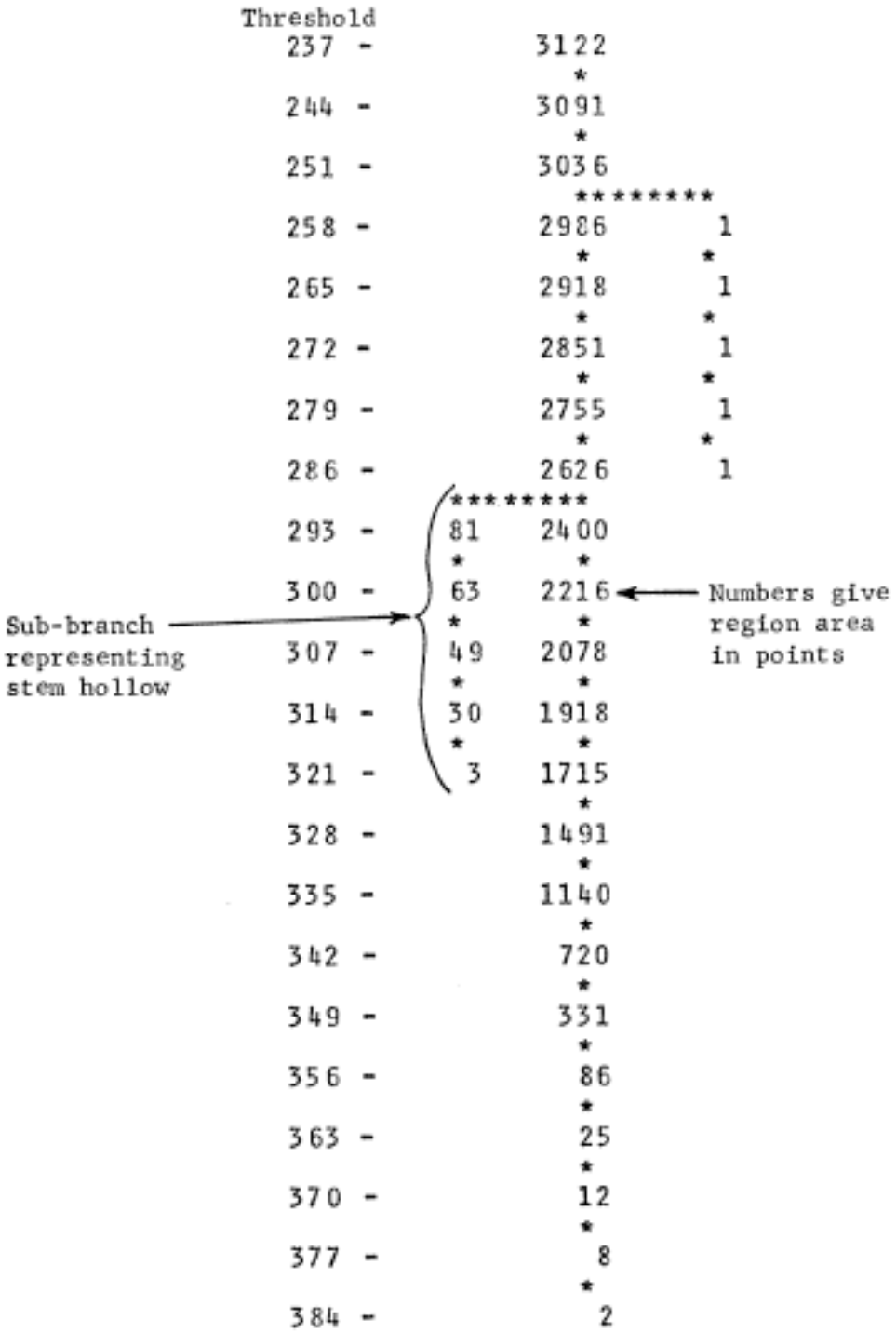


Figure 4.5: Tree of the Apple



level recognition routines could take advantage of this fact to help find stem areas.

Now consider the pear shown in figure 4.6. Its tree, shown in figure 4.7, is topologically similar to the tree of the apple, including a small sub-branch with significant area. The graphs of the various parameters, however, shown in figures 4.8, 4.9, and 4.10, reveal that this sub-branch has a different interpretation than in the case of the apple. First, its center of mass shows it to be positioned to the left of the main region, rather than directly above it. Second, at the point at which the two branches join, there is a rise in the eccentricity in the case of the pear, whereas there is not in the case of the apple. Finally, the eccentricity of the apple just before breakthrough into the background was near 1.0, whereas the eccentricity of the pear is about 1.2, which is significantly higher. Information is also available concerning the surface properties of the pear. The pear's highlight shows a wider "impulse response", which indicates that its surface, although somewhat shiny, is not as highly specular as the apple.

Figure 4.6: Pear

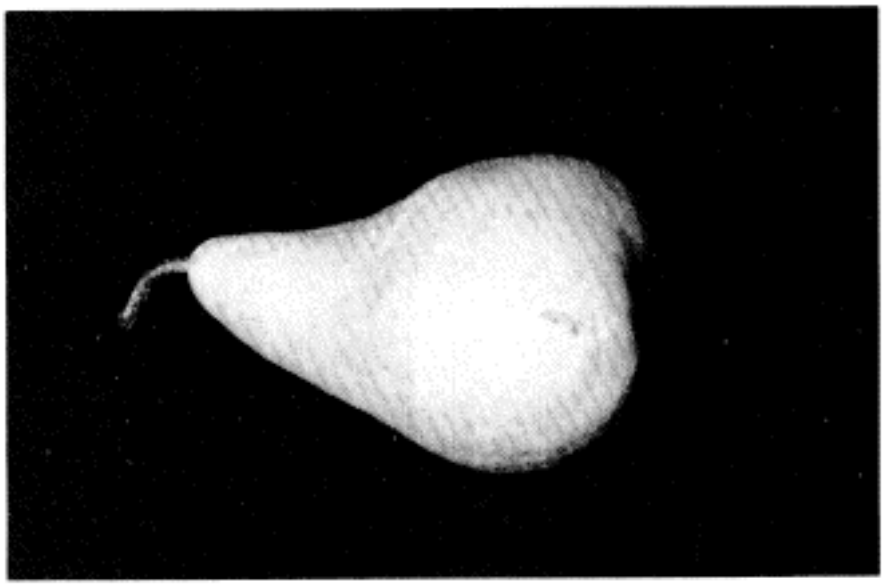


Figure 4.7: Tree of Pear

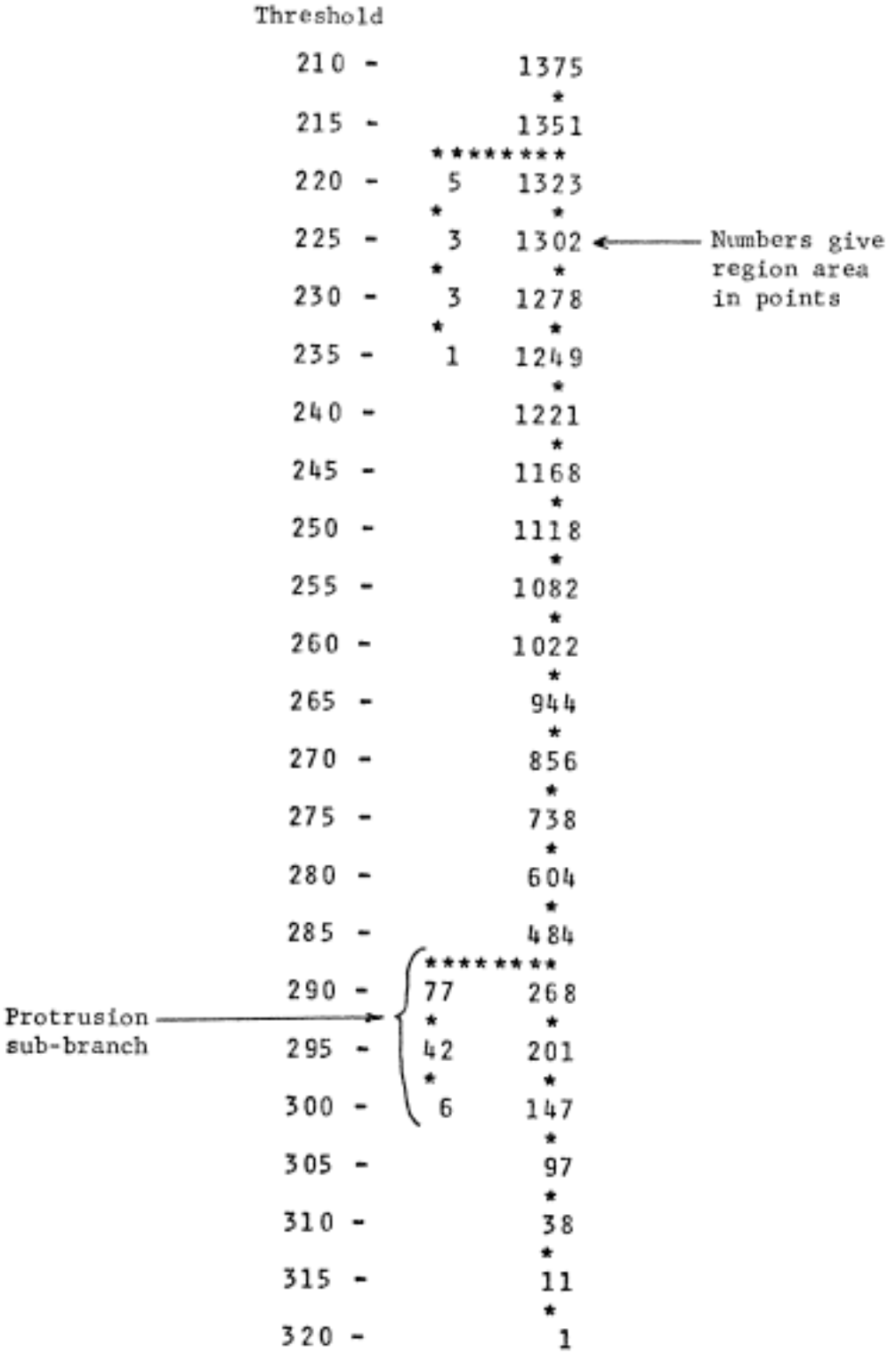


Figure 4.8: Center of Mass of the Pear

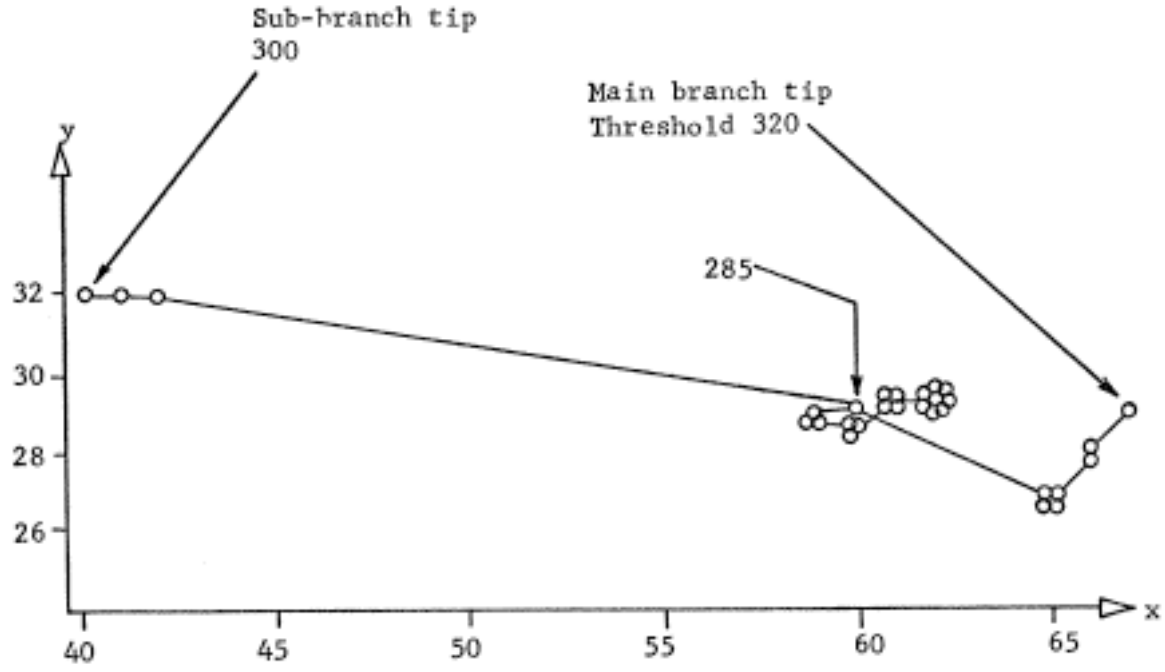


Figure 4.9: Eccentricity of the Pear

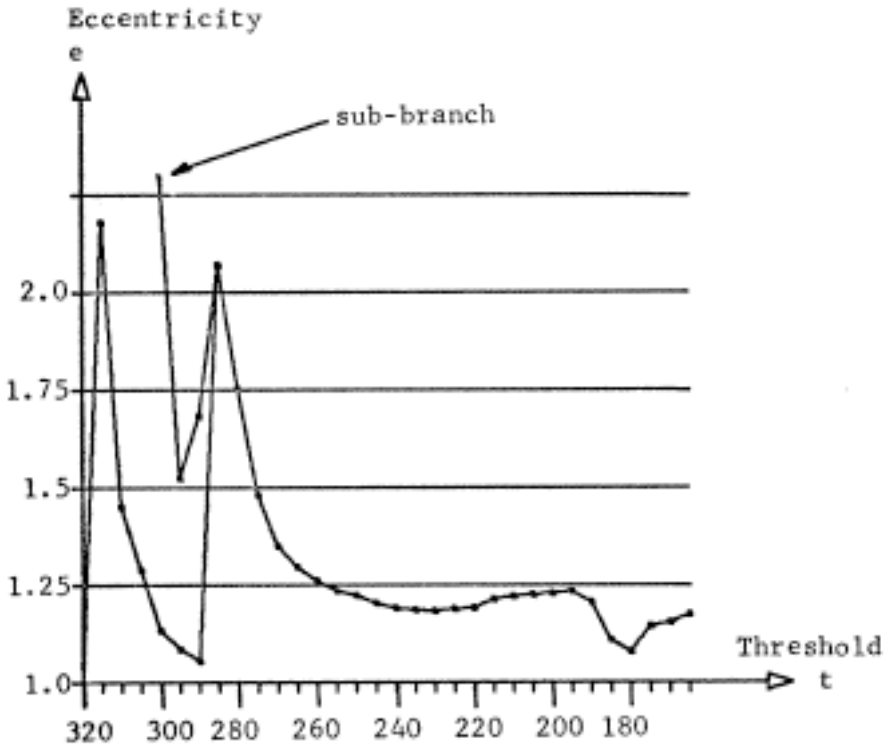
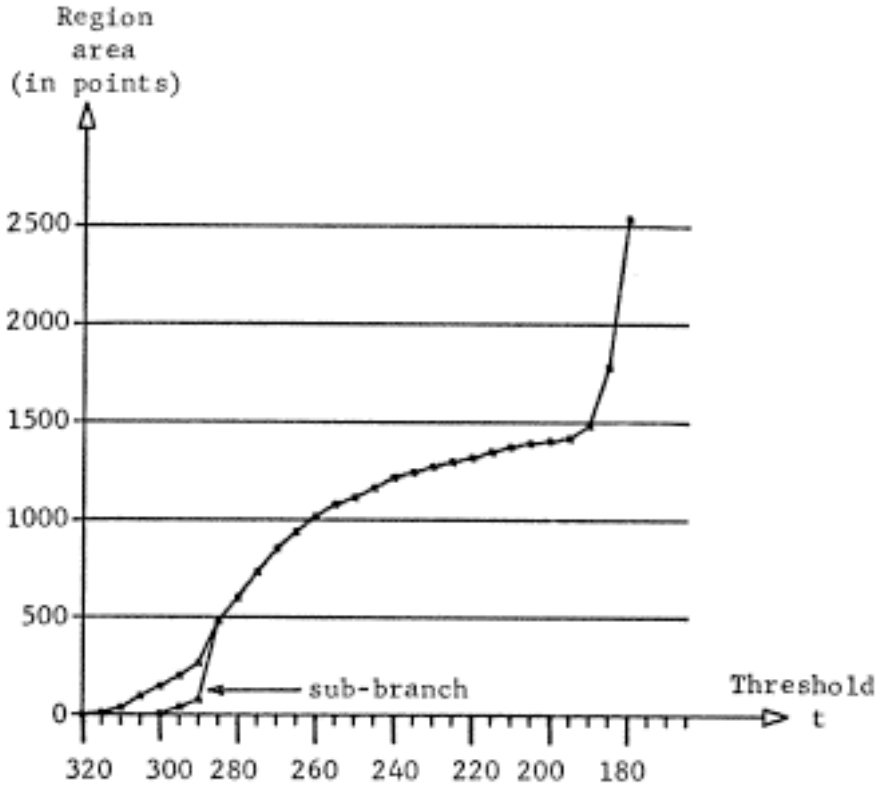


Figure 4.10: Region Area of the Pear



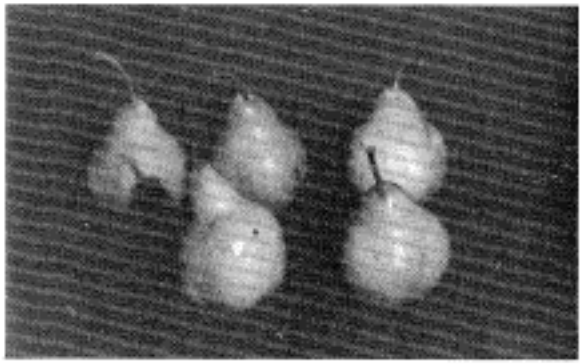
4.3 Useful Features for Fruit Recognition

We will now attempt to list some features which can be easily extracted from the image tree, so that the classification of fruit may be systematized. This list is not intended to be exhaustive. In fact, quite to the contrary; it is intended to show that recognition of fruit is possible with only a few very simple features.

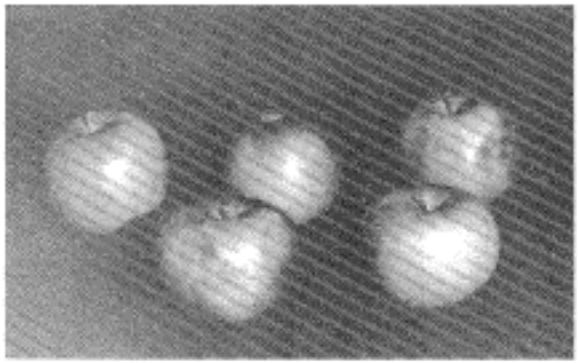
4.3.1 A Sample Set of Fruit

In the course of studying the image tree method, a large number of fruit were processed to study the effects on real images. In addition, a large number of fruit were given identical processing under identical conditions one day in order to gather some statistics on the various features which can be extracted. Photographs of the fruit in this sample set are shown in figure 4.11. The fruit used were Bartlett pears, Macintosh apples, sweet pears, and oranges. The test images include five views each of the Bartlett pears for a total of 25, two views each of the apples (total 10), three of the sweet pears (total 15), and one each of the oranges. Three taped images of peaches are also included in the sample

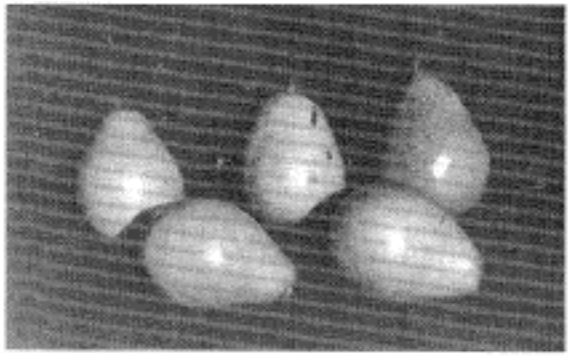
Figure 4.11: The Fruit in the Sample Set



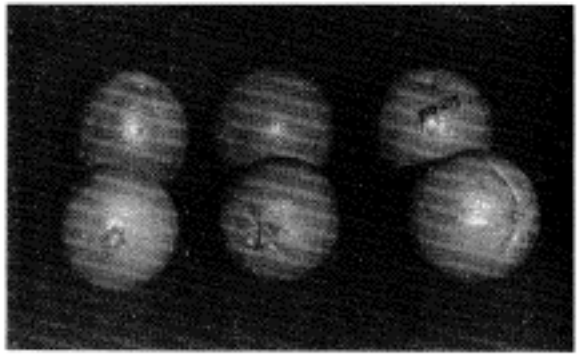
The Bartlett pears



The Macintosh apples



The sweet pears



The oranges

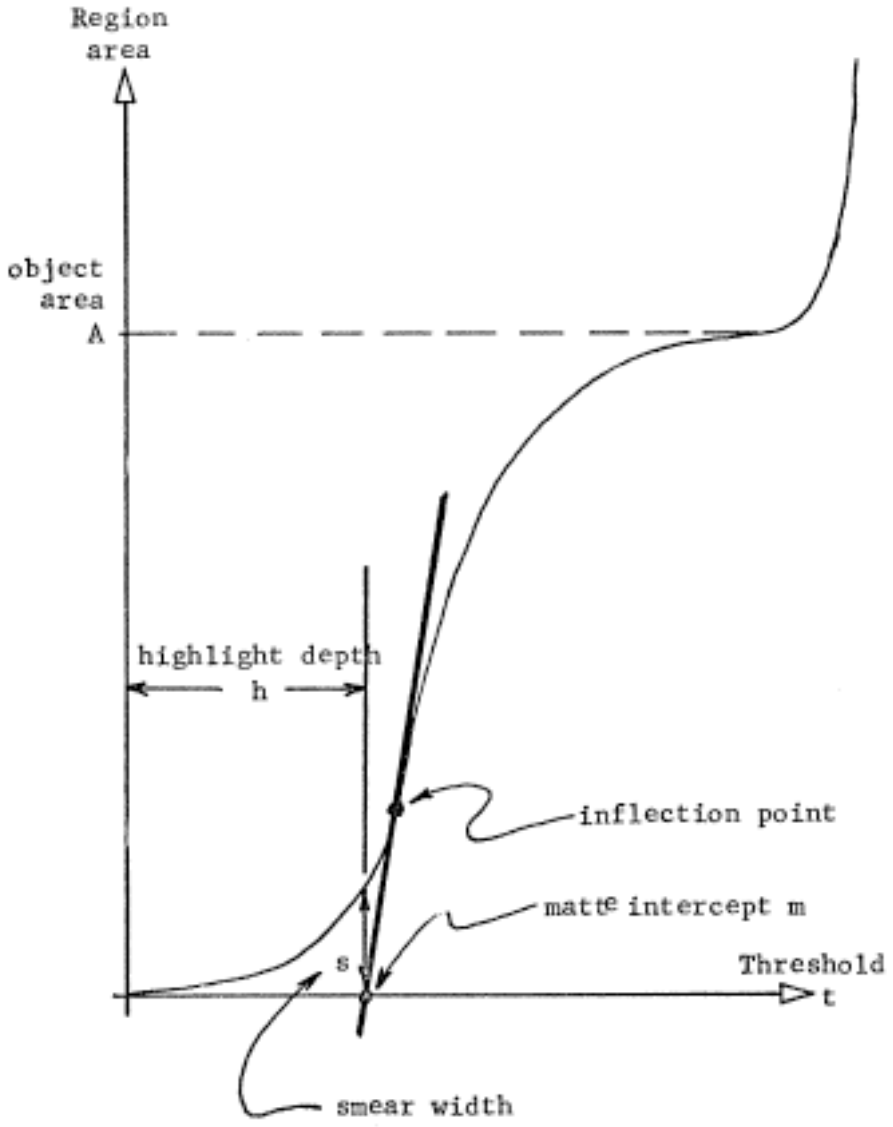
set, although they were recorded under different circumstances. Peaches were unavailable at the time the sample set was run.

4.3.2 Specularity

As was discussed in Section 3.7, the "impulse response" of the surface can be approximately obtained from the region area vs. threshold curve at a branch tip. We would like to characterize this curve in order to extract some significant features that are useful for recognition purposes. One way to do this is shown in figure 4.12. At the branch tip, the second derivative of the region area curve is positive due to the specular component, but negative due to the matte component. A straight line fitted to the curve at the inflection point is shown, extended to intersect the axis. The intersection point is called the "matte intercept". The value of the curve above this intercept is used as a measure of the width of the surface function, as shown on the figure. It is called s, for the highlight "smear" width.

Another measure of the surface function is the amplitude of the highlight, also marked in the figure.

Figure 4.12: Characterizing the Region Area Curve



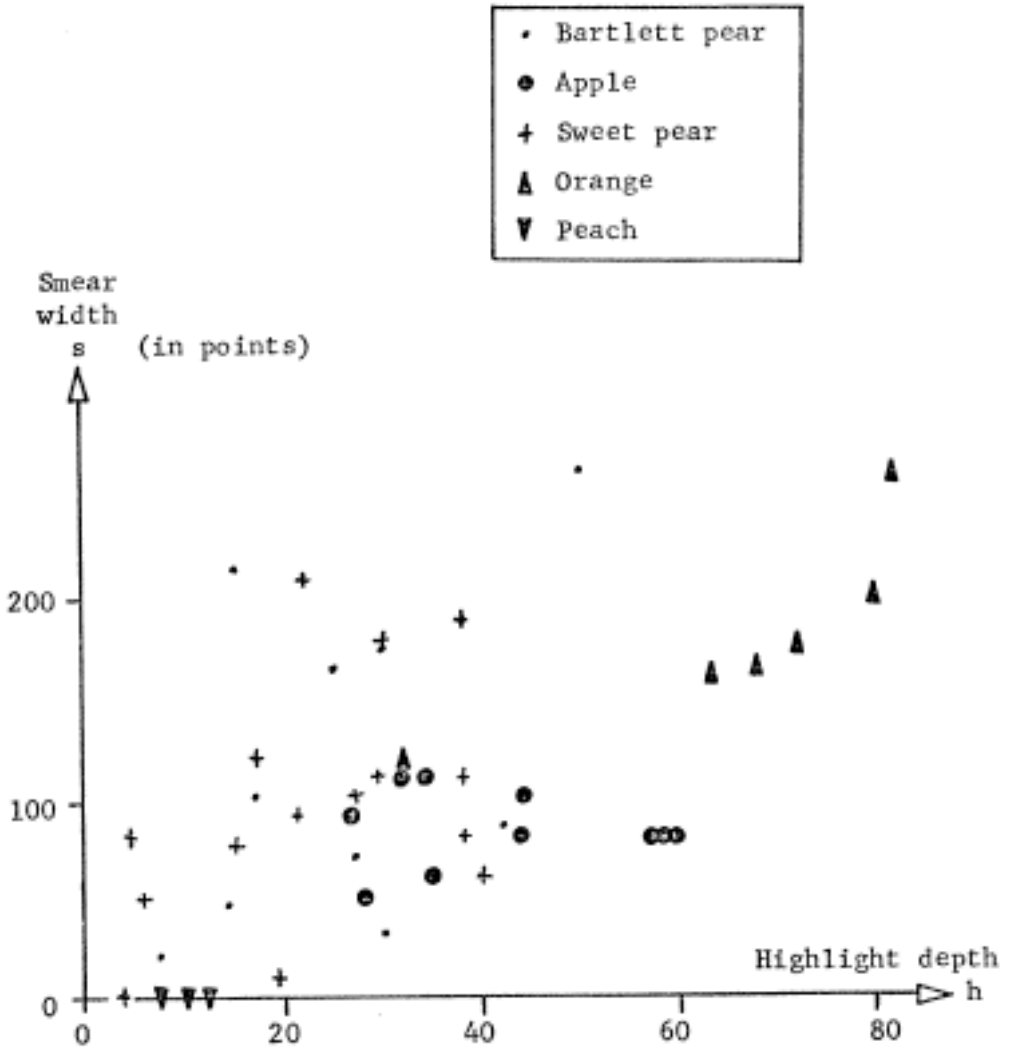
This can be measured in various ways, but is here measured as the amplitude of the highlight above the matte intercept.

A scatter diagram of the smear width s vs. the highlight amplitude h is shown in figure 4.13. Note that the peaches, apples, and orange are separated very well by their highlight properties, but that the two types of pears not only have similar properties, but also show a very high degree of variation in these parameters. This is partly because their surfaces are rather lumpy and uneven, which disrupts the highlight region. As will be seen later, this unevenness can be used to help identify them.

4.3.3 Simple Global Properties

Two very simple properties of a fruit are its brightness and its size. These are both properties which are useful only relative to some additional information not contained in the image alone; specifically, the light intensity and the object's distance from the camera. If this information is available, these two features can contribute recognition information. These quantities can be obtained, in many cases, from other known objects in

Figure 4.13: Smear Width vs. Highlight Amplitude for the Sample Set



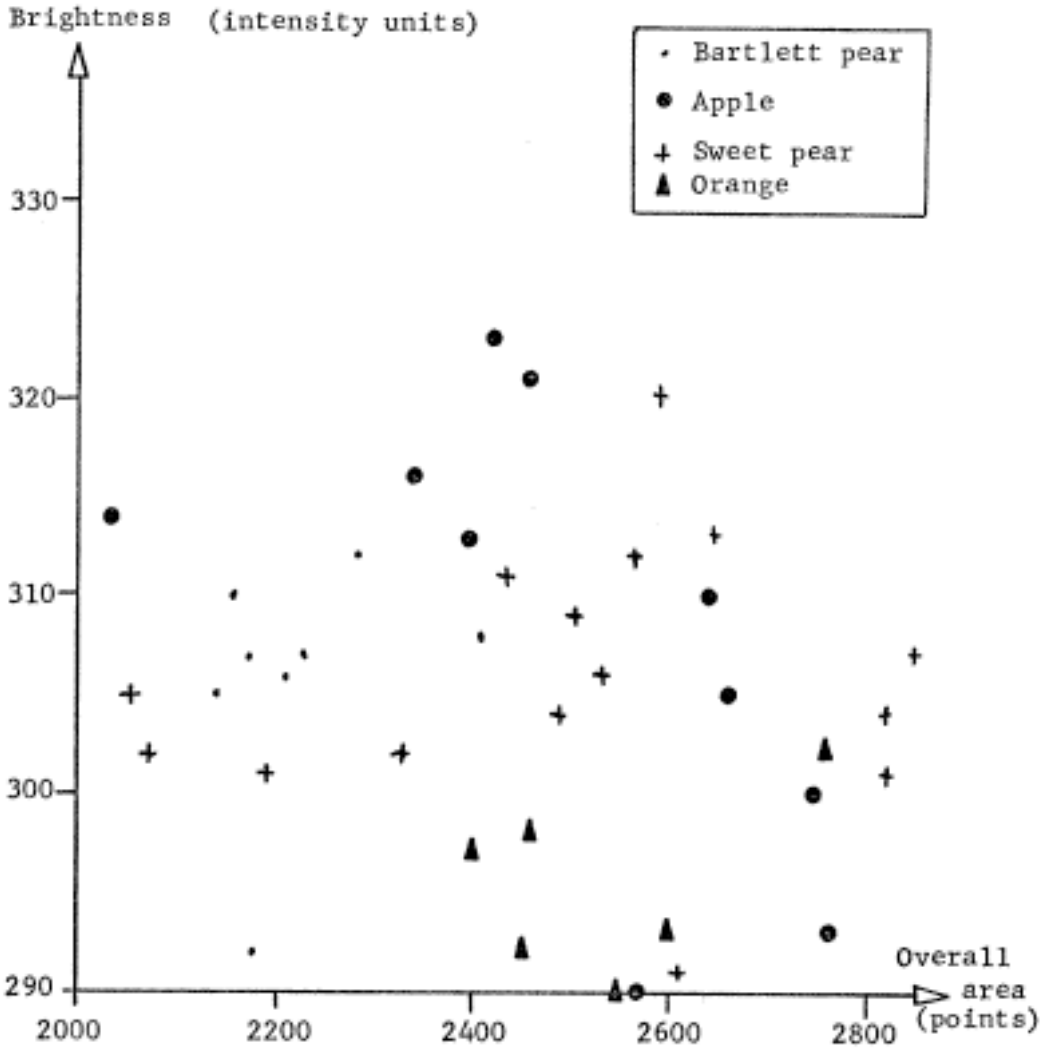
the image. In the experiment described next, the sample fruit were all viewed with the same light intensity and at the same distance from the camera, so that their intensity and size are comparable.

The brightness of an object is taken to be the intercept of the straight line approximation to the matte component with the line of zero region area, thus estimating the brightness of the surface if there were no highlight. The overall area is estimated by scanning up from the root of the tree until the first local minimum in the slope of the region area curve is found. The region area of this node is taken as the object's projected area (see figure 4.12).

A scatter diagram of these two quantities is shown in figure 4.14 for the sample fruit. They are clearly not very useful for distinguishing between the fruit in the sample set. They would be very helpful if very large objects such as watermelons were included, however.

Another optical feature which could be used is color, which would be very powerful for fruit. This feature was not studied in our experiments, because the processing of different color images of the same object would have added complexities and delays without much

Figure 4.14: Brightness vs. Overall Area for the Sample Set



added understanding of the image tree.

4.3.4 Overall Shape

Our simplest shape descriptor is just the eccentricity of the entire fruit outline region, which is shown plotted with the highlight depth in figure 4.15. This parameter alone will identify a banana, which has not been included in the sample set. Note that oranges and apples are extremely round.

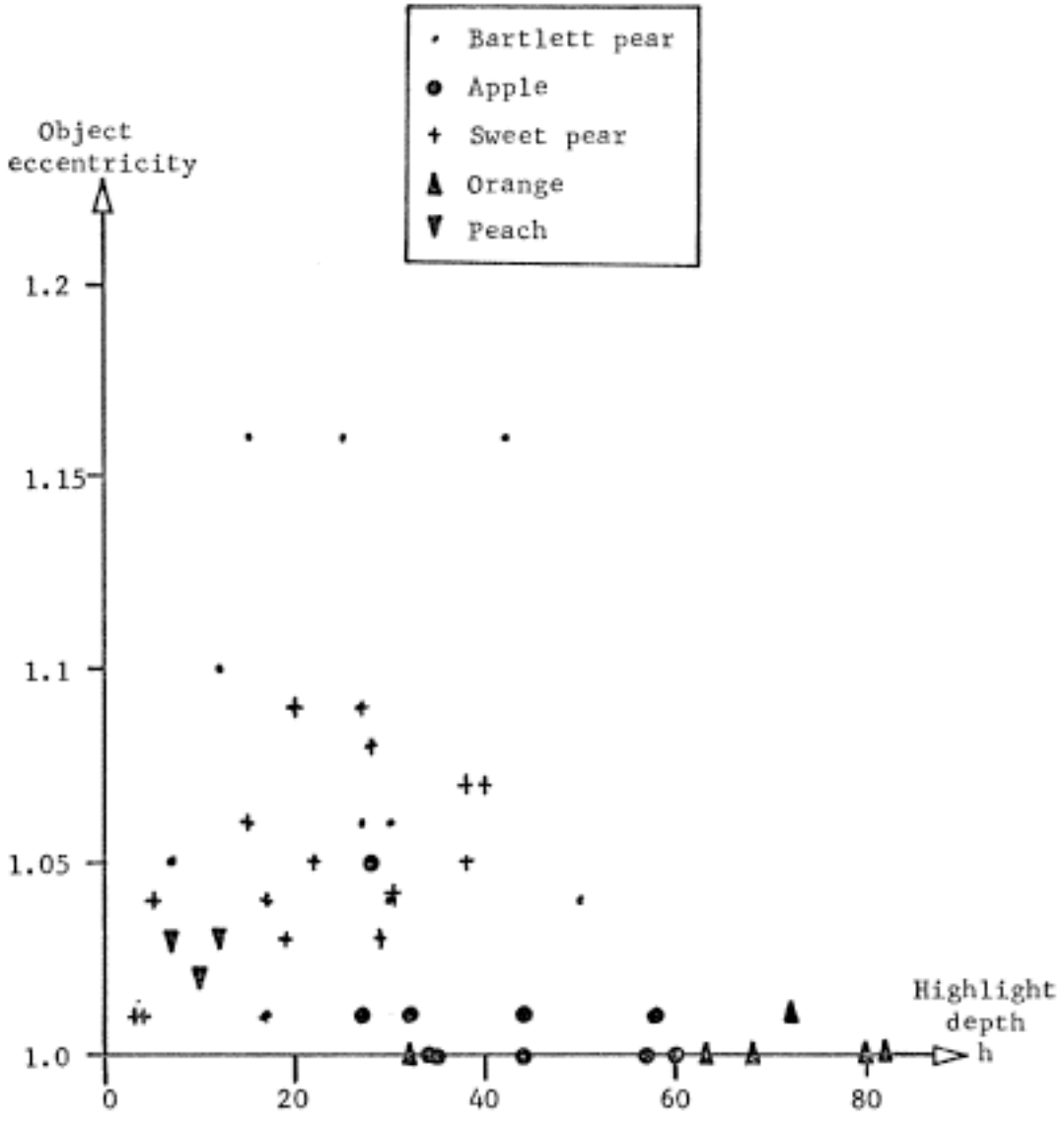
4.3.5 Sub-branch Types

So far, we have used only information extracted from the main branch. Many properties of an object produce sub-branches. In understanding an image we must figure out what these sub-branches represent. Some types of sub-branches will now be discussed, and a simple sub-branch classification algorithm presented.

4.3.5.1 Tactile Texture

The oranges in the sample set supply good examples of tactile texture. A close examination shows

Figure 4.15: Object Eccentricity vs. Highlight Depth for the Sample Set



their surface to be covered by small bumps and valleys. Since the surface is also highly specular, this graininess produces myriad small highlights, as discussed in section 3.9.2. These produce small short branches on the tree. Textural branches represent regions of small area, and are near the tip end of the tree. The number of sub-branches on a tree identified as textural by the classification algorithm shall be denoted by the variable T.

4.3.5.2 Stems

The Bartlett pears show large, long, light-colored stems. The branches produced by these stems are easily identified by their small size and large eccentricity. The number of stem branches is denoted by S.

4.3.5.3 Protrusions

A pear is basically a spherical shape with a protruding bump. These protrusions will frequently produce a major sub-branch on the tree, as in the case of the pear discussed in section 4.2. Such protrusions

generally have a large area, and usually produce a significant jump in the eccentricity of the main branch at the point where they join it. The number of protrusions will be denoted by the letter P (usually 0 or 1).

4.3.5.4 Stem Hollows

An apple has a somewhat conical depression on top in the spot the stem is attached. The stem itself is smaller and darker than in the case of the pear. This stem hollow will often produce a separate branch on the tree, as the light reflected from the back of the hollow is surrounded by darker points on the rim of the hollow. Furthermore, the dark stem will often bisect this region, producing two sub-branches. Thus a significant sub-branch which causes a drop in the main branch eccentricity when it joins is likely to be a stem hollow, and this is reinforced if there is another similar region nearby. The number of stem hollow regions is denoted by the letter H (usually 0, 1 or 2).

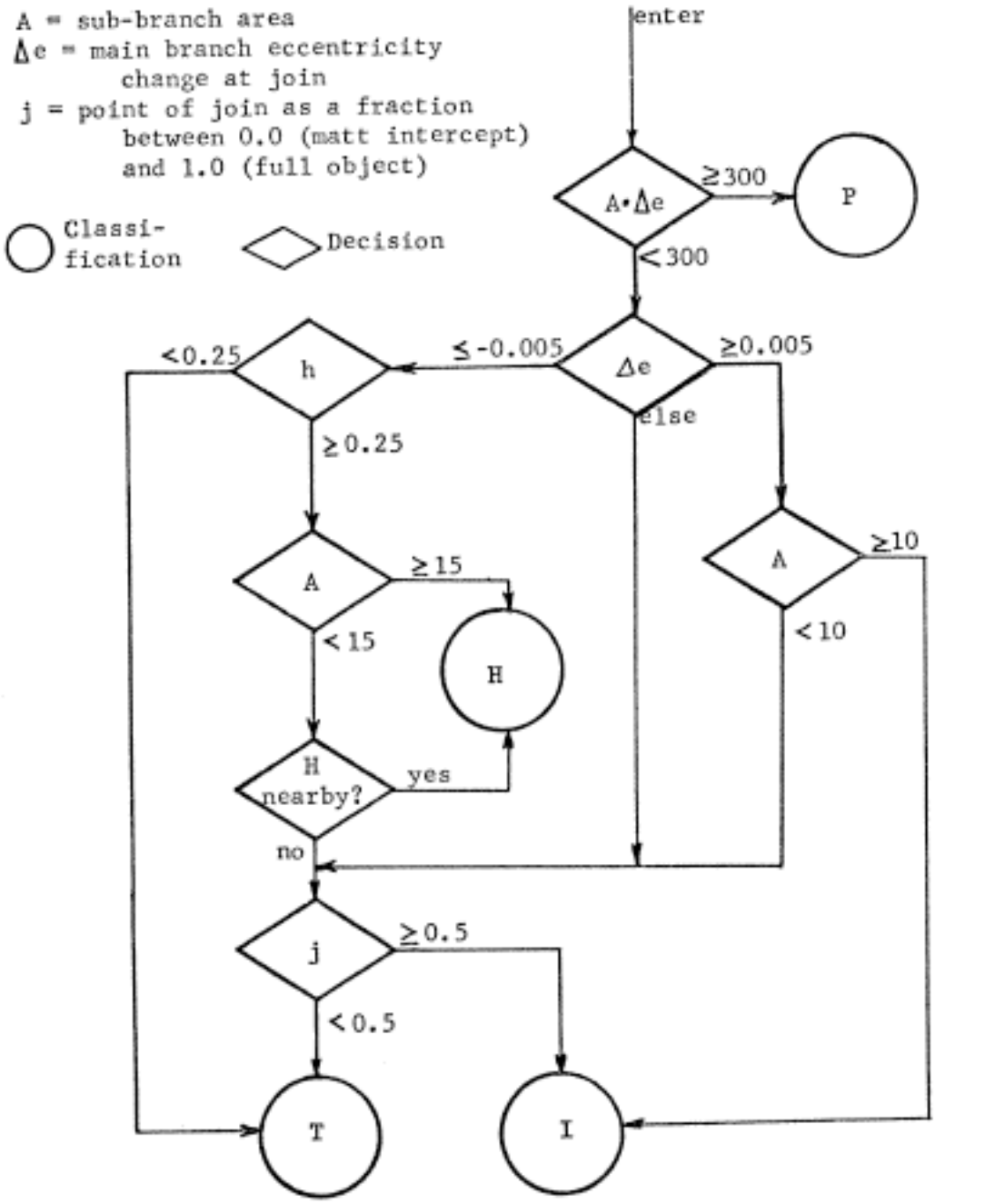
4.3.5.5 Surface Irregularities

There are frequently a number of branches which do not fall into any of the above catagories. These often are due to irregularities in the surface of the object. These irregularities are larger than what is called tactile texture, but smaller than those large enough to be called protrusions. The number of such branches shall be denoted by the letter I.

4.3.6 Sub-branch Classification

A very simple algorithm was written to classify sub-branches. It is shown in flow chart form in figure 4.16. The parameter A represents the area of the sub-branch just before it joins the main branch. The parameter Δe is the change in the eccentricity of the main branch at the point where the sub-branch joins. Δe is positive if the sub-branch produces an increase in the eccentricity, and negative if it produces a decrease. The parameter j tells where on the main branch the sub-branch is attached, on a scale from 0.0 (matte intercept) to 1.0 (full object). If the sub-branch joins the main branch in the highlight region (above the matte

Figure 4.16: Sub-branch Classification Algorithm



P = protrusion H = stem hollow T = texture I = irregularity

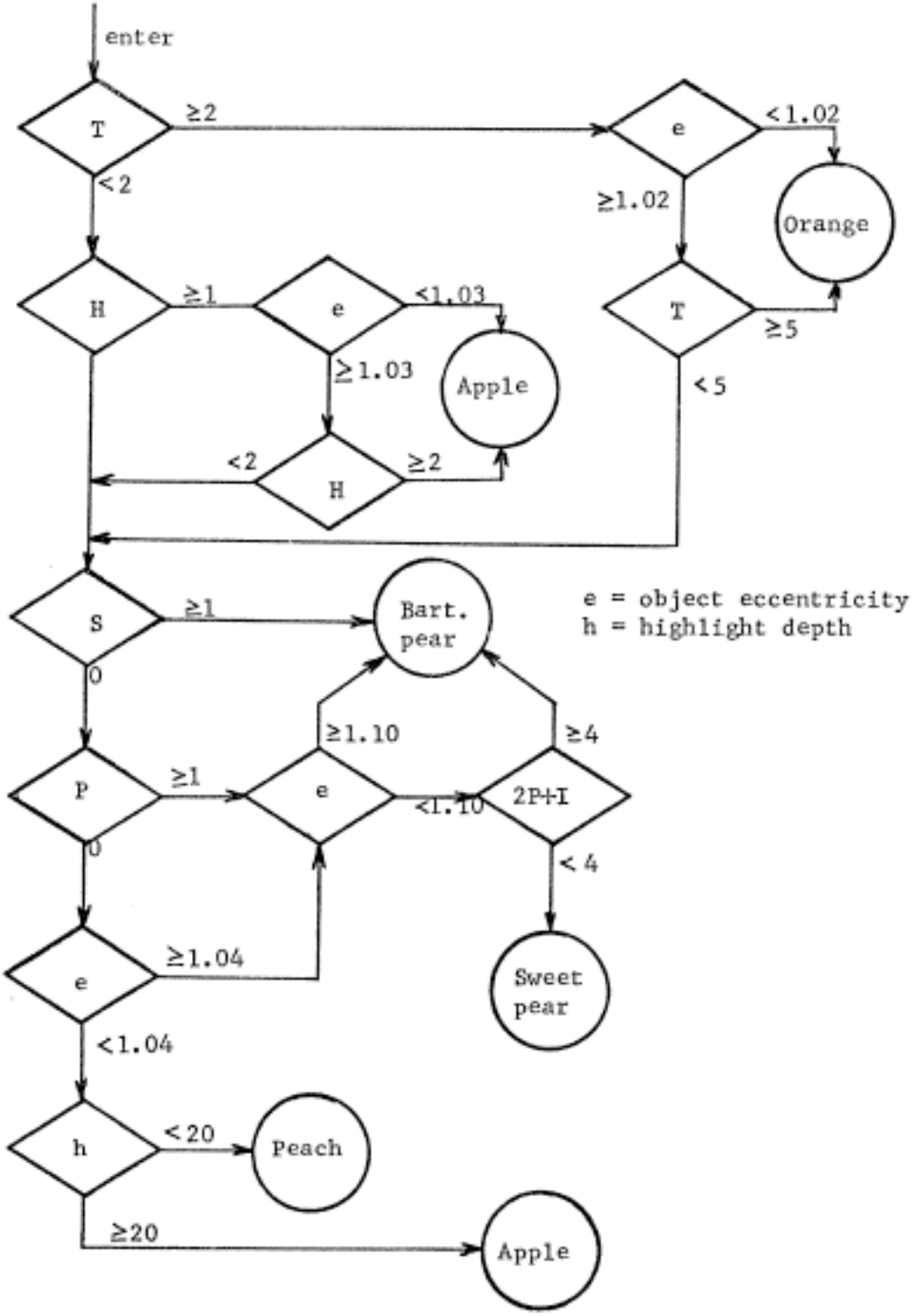
interface), we set j to 0.

This algorithm was hand-computed for each of the fruit in the sample. The flow chart is recommended over the following very sketchy description of the algorithm: If the biggest region represented by the sub-branch is large and increases the eccentricity of the main branch significantly, call it a protrusion region P. Otherwise, if it increases the eccentricity only slightly, and is not too small, it is an irregularity I. If there is no change in the main branch eccentricity, or if a small positive change is produced by a small region, than the sub-branch is either texture T or an irregularity I, depending on whether it is towards the tip or the root of the tree. Finally, if the region is large enough and produces a drop in the main branch eccentricity, and is not at the tip of the tree, it is a stem hollow H. A small region will be accepted as a stem hollow if there is another stem hollow nearby.

4.3.7 Object Recognition

The flow chart of figure 4.17 produces an object identification once the sub-branches have been characterized. In essence, oranges are identified by

Figure 4.17: Object Identification Algorithm



having lots of texture branches and being very round. Apples show stem hollows and are very round. A stem area identifies a Bartlett pear immediately. The two types of pears are sorted out on the basis of their eccentricity, the number of protrusion branches, and the number of irregularities. Round objects with essentially no highlights are peaches.

The flow-chart shown correctly identified all of the fruit with the exception of one Bartlett pear (BP11) which was identified as a sweet pear. The pertinant data for each of the sample fruit are shown in figure 4.18.

Our conclusion is that recognition of images of single fruits is relatively easy, using the image tree. The image tree allows the easy extraction of enough information about surface properties, shape irregularities, and general shape, as well as helping to spot specific characteristics such as stem hollows and stems, and the procedures which extract this information are reasonably simple. More complex routines which take the trouble to look more closely at the tree's statistics should be even more reliable.

The recognition procedures described would be disrupted (as would many others) by occlusions, shadows, missing stems, and object positions which hide

Figure 4.18: Eccentricity and Sub-branch Types for the Sample Set

BP11 incorrectly identified

Obj.	e	S	H	P	I	T	Obj.	e	S	H	P	I	T	Obj.	e	S	H	P	I	T
BP01	1.06	1		1			BP21	1.09	1	-	-	-	-	S1	1.00					7
BP02	1.04	1		1		2	BP22	1.14	1	-	-	-	-	S2	1.00				1	5
BP03	1.16	1			2		BP23	1.11	1	-	-	-	-	S3	1.00		1			9
BP04	1.05	1		1			BP24	1.18			1	4	1	S4	1.01					3
BP05	1.01	1		1	2		BP25	1.15	1	-	-	-	-	S5	1.00					9
BP06	1.16			1	2		SP01	1.04				1		S6	1.00					6
BP07	1.10						SP02	1.08			1	1		M01	1.00					
BP08	1.06	1		1		1	SP03	1.04						M02	1.01		2			
BP09	1.04	1			2	1	SP04	1.05						M03	1.00		1			
BP10	1.16			1			SP05	1.09				1		M04	1.00		2			
BP11	1.09				2		SP06	1.07						M05	1.00		2			
BP12	1.13	1		1			SP07	1.07						M06	1.01				1	
BP13	1.10	2	-	-	-	-	SP08	1.05				1		M07	1.05		2			
BP14	1.04			3		1	SP09	1.06						M08	1.01		1			
BP15	1.13	1	-	-	-	-	SP10	1.09				2		M09	1.00		1	2		
BP16	1.17	1	-	-	-	-	SP11	1.04						M10	1.00		2			
BP17	1.07			1	2		SP12	1.03						P1	1.03					
BP18	1.06			2			SP13	1.01				1		P2	1.02					
BP19	1.06			1			SP14	1.03				1		P3	1.03					1
BP20	1.11			1			SP15	1.01			1									

BP: Bartlett Pear
S: Sunkist Orange
Sp: Sweet Pear
M: MacIntosh Apple
P: Peach

S: Stem
H: Stem Hollow
P: Protrusion
I: Irregularity
T: Texture

significant features. Many of these problems could be eased by a suitable vertical system, which could use other knowledge to explain and correct changes in the image tree. Other problems can be solved without higher-level aid, simply by making the recognition routines more clever. For example, occlusions can generally be detected by the way in which two regions connect. Once an object is known to be partially occluded, corrections can be made to its region statistics which give an idea of its form, under the assumption that the visible and the hidden parts are similar.

Even in the presence of severe occlusion problems, the tree still gives valuable local information about highlights and texture. Although the stems gave significant aid in identifying Bartlett pears, the stems were not seen in ten of the test cases, yet nine of these were correctly identified.

4.4 Faces

This section illustrates the behavior of the image tree produced from a more complex smoothly curved object: a human face. It is included to show another example of a real recognition task for which the image

tree is potentially useful. A tree was generated from an image of a face, seen full face and lit from the front. This tree is shown in figure 4.19. Branches of the tree have been labeled with the local maxima on the face to which they correspond, and the shapes and positions of these regions is shown in figure 4.20. These regions might be useful for face recognition, at least for the simple angle of view and lighting considered here.

Contour maps at a single level of the tree are shown in figure 4.21, for each of two levels (marked in figure 4.19). At level 313, most of the major regions seen in the photo appear, with the exception of the lower lip highlight, which is considerably dimmer. The contour map at level 268 is rather interesting. Consider not the region included within the contour, but the area excluded. This includes most of the mouth, the eyebrows, the eyelids (the eyes are closed), the nostrils, and a shadow area on either side of the nose. These are locally dark areas in the image. These could be isolated by making an inverted tree - that is, by making a tree with the image negated. These locally dark areas are probably better places to begin face location, since there are fewer of them than there are locally bright areas, and they are more prominent. Indeed, there are

Figure 4.19: Tree of a Face

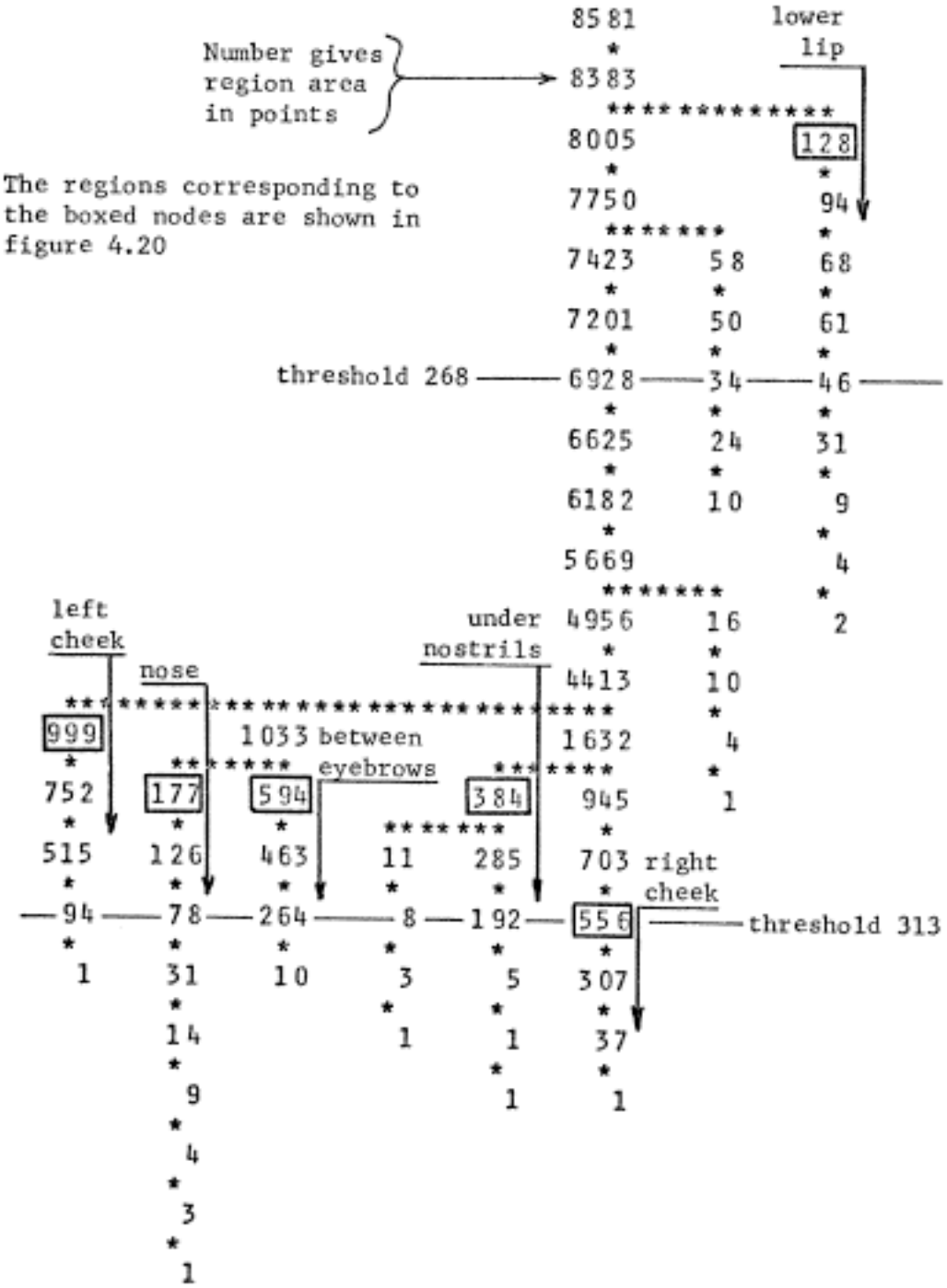


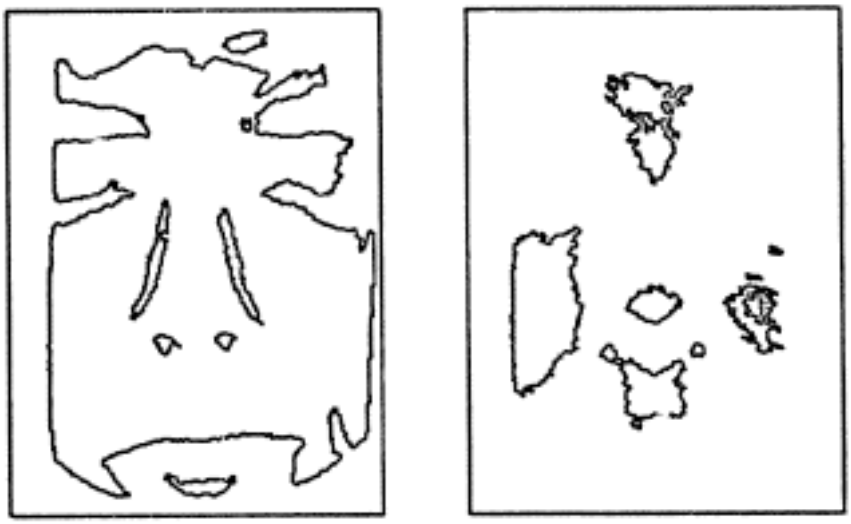
Figure 4.20: Some of the Regions of the Face Tree

Regions corresponding to boxed nodes

Face alone

Regions superimposed on face

Figure 4.21: Slices of the Tree at Two Thresholds



Threshold 268 Threshold 313

experiments which indicate that as babies learn to see faces, they first fixate on the pair of eyes [1,6]. Once a face is roughly located, higher level routines can make sense of the locally bright areas with less difficulty. Figure 4.22 shows a contour map with both levels superimposed, with the dark regions shaded.

Note that the image tree can easily be used to isolate facial features and determine their approximate position. In order to better characterize their shapes, more complex shape descriptors would probably be needed than those which have been used so far. The image tree can be used to characterize the shapes of objects, such as noses, which have no "hard edge" boundary. This will be further discussed in section 5.2.4.

Figure 4.22: The Two Contour Maps Superimposed

Shaded area below threshold 268

Chapter 5 Discussion

5.1 Comparison with Previous Work

The image tree can now be situated among the pattern recognition methods discussed in chapter 2. It is a "regions" method, rather than an edge detection scheme, and does no differentiation or other pre-processing of the image. It extracts information about both the surface properties of an object and about its shape. It does not require any high degree of precision of measurement with regard to the exact location of specific points in the image, and does not make any essential use of perspective information. It does not attack problems of the "parsing" of an image into its component parts directly, although it may aid this process by the way it organizes the image information.

5.2 Advantages

The image tree has a number of advantages for pattern recognition over many previously used methods.

5.2.1 No Edge Problems

By virtue of being a "regions" method, the image tree is immune to many of the problems which beset edge detectors. These include false edges and gaps in real edges.

5.2.2 Noise Immunity

The use of moments and averages as parameters on the tree results in a system which does not require great precision of measurement, and which is insensitive to noise and distortion in the image.

5.2.3 Obtains and Separates Surface and Shape Information

The tree also carries information about both surface properties and shape. The two forms of information are well separated on the tree, with the surface information being carried at branch tips, and the shape information appearing further down the branch.

Indeed, the tree can be thought of in a general way as carrying detail information near the tips of branches, and lower resolution information towards the root. For an image containing many objects, the root

will represent the entire scene, and various sub-branches will represent sub-parts, and then sub-parts of the sub-parts. The tree can thus be thought of as providing a range of measurements of differing degrees of acuity. These notions of pattern recognition as a sort of "measuring" problem are due to Kirsch.

5.2.4 Objects Without Boundaries

The image tree is easy to apply to the recognition of objects without real edges or well-defined boundaries, such as a nose, or an object lit so that one side fades off gradually into shadow. Assuming the object produces a separate tree branch, it can be analyzed from the data at the tip of the branch, working down towards the base until the parameters indicate that the region is taking in too much extraneous area to be useful. Thus some information about a nose can be extracted even though it has no well-defined upper boundary, because it has well-defined lower and side boundaries. This simple task can be rather complicated for edge-oriented procedures, or for programs which are regions oriented but which do not make a series of related measurements at different levels, as in the tree.

By the same arguments, the tree will contain information about a smoothly curved object even if it is partially obscured, provided it contains a local brightness maximum. The procedures which analyze the tree must be able to detect the occlusion and to try to compensate for it.

5.3 Problems

The separation of coarse and fine information is not always maintained by the tree, unfortunately. When branches representing two different objects merge, information about those parts of the object not yet filled out by the region may be lost. If a small highlight area is swallowed up by a larger region before achieving much depth in its own right, the information that would have been obtained about the local surface properties of that area are swamped out. When a region representing some object in a scene joins with a larger region representing the background, the information about the smaller object is lost. One case in which this can occur is when a dark object is on a light background, or near a lighter object. Or, alternatively, a region may extend beyond the boundaries of an object on one side

before reaching the boundary on the other side, possibly due to an overall gradient in the light intensity. This shall be referred to as a "breakthrough". Although it can usually be easily detected by its effect on the region parameters (sharp rise in the region area and eccentricity, and sudden shift in the center of mass), it still means a loss of information about the side of the object which the region has not "filled".

5.4 Further Considerations

5.4.1 Other Statistics

So far region shape has been characterized by the region area, eccentricity, and center of mass position. There are many other region statistics which could be used to characterize the regions, depending upon the particular recognition task at hand.

One very simple addition which could be made would be to compute the x and y second moments separately, so that the major axis of the region could be found. This is the axis about which the region has a minimal moment of inertia. This would allow the tree

parsing programs to check highly eccentric regions for proper orientation, given some hypothesis about what the region represents. Other higher moments could also be used, although their interpretation would be somewhat more difficult.

It would be very useful to know if a region has any major holes in it. It is very easy to tell if it has any holes at all by calculating its euler number while the connectedness of the region is being verified. This number can be calculated on a local basis, using procedures reported by Gray [7]. Since the euler number gives the number of objects minus the number of holes, and since the region is known to be one connected object, the number of holes in it is one minus the euler number. Unfortunarely, this is the topological number of holes, rather than the number of holes of large area. Of course, there can be no large holes if there are no holes at all, so the euler number can be used to see if a test for significant holes is needed. Unfortunately, a large percentage of regions generated by real images will have small holes in them, especially near the edge, so this test will not reject very many. If a region is known to have no major holes, and has a high eccentricity, then it must be elongated. In the absence of a "major hole"

predicate, there is always the possibility that a high eccentricity may be due to a perfectly round region, but with a large hole in the middle.

In general, any sort of shape-descriptor algorithm can be applied to the regions, such as the Blum algorithm (Medial Axis Transform) [3]. I believe, however, that one of the strengths of the image tree as a method is to allow easy recognition with relatively simple region shape descriptors. Using very complicated descriptors not only will consume a great deal of computer time, but will also complicate the analysis required of the higher-level programs. A more detailed shape analysis should probably be reserved for cases in which problems arise in the simpler procedures.

5.4.2 Region-hole Duality

The tree procedures are not symmetric with respect to light and dark, as has been pointed out earlier. Thus a black spot on a light object is not perceived as an object, but as a hole in a region. Furthermore, these holes are not detected by the programs, and insufficient information is stored on the tree to tell that they are there. Thus the effect of a

hole is to decrease the region area, and increase the region eccentricity, but it is not detected as a hole per se. In the detection of texture, black speckles have a completely different effect than white speckles. An object is harder to recognize on a white background than on a dark one.

This is not a desireable situation. An object should be easy to recognize on any highly contrasting background, regardless of whether it is darker or lighter than the object. A possible solution would be to make two trees, one with the image negated. Thus one would be the tree already discussed in detail, and the other would be a tree of dark regions on lighter backgrounds, in which the tips of the branches would represent locally dark areas, rather than locally light ones. For the face considered in section 4.4, these dark branches would represent significant locally dark areas, such as the eye sockets, the nostrils, and the dark areas along the side of the nose. The eye sockets and the nostrils, in particular, are probably very important in orienting visually with respect to a face.

There is no reason why this procedure should not be carried to more than one level. Whenever a region is isolated, the contiguity scan routines could be called

again, but scanning only inside the region, and with their sense inverted, so that they would find holes. Small holes could then be eliminated, but if there were any large ones, they would be noted on the tree. Furthermore, the sense could then be inverted once more, and the contiguity scan tried once again to find additional light regions inside the dark holes.

This procedure would succeed in finding a dark apple on a light background. The apple could be isolated by an inverted run of the tree procedures, and then the normal procedure could be carried out on the region thus isolated.

5.4.3 Complex Lighting

In the above discussion, it was assumed that the illumination was coming from a single point source. Changing the source of the illumination will change the properties of the highlight region, but will not alter the basic properties of the tree. If the illumination is from a diffuse source, specularity information is lost. Light from several point sources will produce multiple highlights. If the high level parsing routines know about the light source, they can compensate for these

effects. By making hypotheses about the objects in the image, these routines could equally well find out about the lighting from the image.

5.4.4 Isolations of Regions

A by-product of the image tree is the isolation of regions which can be used as data for other feature extraction programs. One might, for example, take a fairly large region around the highlight, subtract out the small region containing the highlight itself, and hand this difference region to a textural analysis program. This program could use this region to extract texture information in various ways, such as performing a Fourier transform, autoconvolution, or similar processing, obtaining information about surface speckles not available directly from the tree. Using a region generated from one of the tree nodes helps assure that the portion of the image upon which the analysis is performed is a suitable one.

5.5 Summary and Conclusions

A procedure has been outlined for processing images of three-dimensional objects with smoothly curved surfaces. The method is able to extract some information about the surface properties of the objects, such as the texture, specularity, and surface irregularity. Information about shape is also extracted. The procedures are insensitive to noise and distortion, and can be used to perform real recognition tasks. It is hoped that this work will provide a stepping-stone in the challenging study of computer vision.

Appendix: Description of Algorithms

This appendix contains an outline of the algorithms used in the tree generating program.

The image tree is generated one threshold level at a time, starting at the highest level (branch tips). At each level, the image is scanned, and the points above the threshold are marked in a scratch array. This scratch array is then scanned for marked points. When one is found, a contiguity routine is called, which visits all marked points which can be reached from the start via a connected path. The marks are erased by this routine as it goes, and statistics are kept on the region thus generated, such as the sums of the x and y coordinates of the points, and the sum of the squares of the x and y coordinates (used to compute the center of mass and the eccentricity). A tree node is then made up for the region, and the scan for marked points continues. A special mark is left in the scratch array for each region. When this mark is encountered during the scan at the next level, it is looked up on an association list. This establishes the link between a region and the regions which are a subset of it at the previous

level - i.e. between a node and its sub-nodes.

The contiguity scan is the most complex program. It works by leaving directional pointers in the scratch array. These are three-bit codes denoting one of the eight possible neighboring points. The contiguity scan is always started at a point which is on the bottom edge of the region. It traces along this edge to the right by moving from one marked point to the next, but always keeping an un-marked point to the right side. As it goes, it erases the marks, so that for a region with smooth boundaries, it will follow a spiral path to the center, "eating up" the marks as it goes, like a lathe with the tool continually advancing into the work.

As the contiguity routine scans, it lays down back pointers in the scratch array which enable it to retrace its path back to the start. If a dead end is reached (no more marked neighbors), it traces back along this path, looking for marked points to the right. There can be no marked points on the left side while backtracking, since this was the right side on the way out, and the outgoing scan stayed as far to the right as possible. If a marked point is found on the backtrace, it is replaced with a pointer to the adjacent path already traced out, and then a new path is traced as if

this were a new starting point. When the backtrace reaches the original starting point, the contiguity scan is completed. The effect of this algorithm is to construct a tree of pointers in the scratch array, with the starting point at the root. All points which can be reached via a connected path from the starting point will be a part of this tree, an example of which is shown in figure A1.

An algorithm developed by S. Bryan [4] could speed the contiguity scan considerably. It entails coding the scratch array line by line as strips, as in figure A2. Each strip is specified by its y coordinate, and the x coordinates of its left and right end. The contiguity of these strips is then checked, rather than operating on the individual points. This algorithm not only avoids scanning the entire scratch array, most of which is blank, but also requires fewer operations to find all of the contiguous points, since they are gathered into groups. It thus takes advantage of the fact that regions produced by real images, as opposed to random noise, will tend to have the points clustered into bunches.

A number of other programs were written in the course of this research. In order to make it convenient

Figure A1: The Tree of Pointers Layed Down by the Contiguity Scan Algorithm

(Shown for an arbitrary region)

• = marked point, included in region

\ = pointer in direction of root
(arrowhead not shown due to small size)

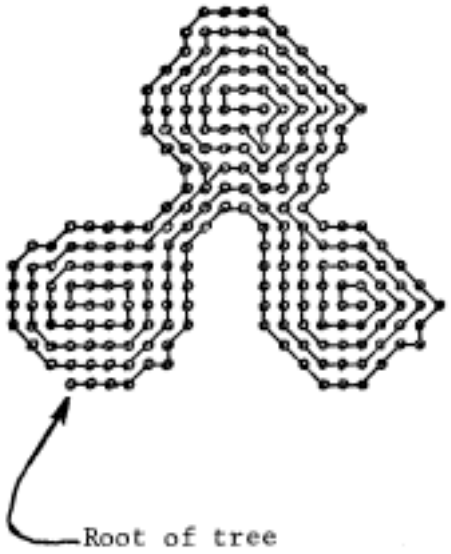


Figure A2: A Region Coded as Strips

The same region is used as in figure A1

to study a large number of trees, programs were written to print out the trees on the line-printer, with the significant parameters associated with each node. Furthermore, a program was produced to plot any parameter vs. threshold along any set of branches of the tree. This program was used to produce the graphs in this paper.

Programs were also written to display an intensity modulated picture of the image, using the seven intensity levels of a DEC 340 display. Since our 340 has no fast raster mode, a display compiler was written which generates a display list in increment mode, allowing fairly large images to be shown virtually flicker-free. Other routines enable any arbitrary region in the image to be shown superimposed on this picture. The pointer method used in the contiguity scan was actually written for these display routines, which were developed first. The existence of this program made the writing of the contiguity scan very simple, which is one reason why faster algorithms such as the Bryan algorithm were not sought.

A large amount of code was required to back up the programs mentioned above. This includes a dynamic storage allocator for manipulating a large number of

arrays of changing size, display and plotter routines and other I/O routines, routines for manipulating list structure, and routines which map arbitrary local procedures over an array. The programs comprise over 5200 words of PDP-10 MIDAS assembly language code, not including about 1700 words of fixed buffer and tables, and not including the dynamically allocated array and list structure area, which can grow to an arbitrary size.

Also used was the CNTOUR program [12], which draws intensity contour maps of an image, and which was written early in the course of this research, before the exact area of study had been decided upon.

Bibliography

1. Ahrens, R., "Beitrӓge zur Entwicklung des Physiognomie - und Mimikerkennis", Z. f. Exp. u. angew. Psychol., 1954, 2,412-454,599-633.

2. Alt, F. L., "Digital Pattern Recognition by Moments, Journal of the Association for Computing Machinery 9,2 (April 1962), (pages 240-258).

3. Blum, H., "A Transformation for Extracting New Descriptors of Shape", Models for the Perception of Speech and Visual Form, W. Wathen-Dunn (Ed.), MIT Press, Cambbidge, Mass., 1967, (pages 362-380).

4. Bryan, Sam, The Strip Blobber: A Scene Analysis Algorithm, National Bureau of Standards, Washington, D.C.

5. van Diggelen, J., "A Photometric Investigation of the Slopes and the Heights of the Ranges of Hills in the Maria of the Moon", Bulletin of the Astronomical Institutes of the Netherlands, Vol. II, 423 (July 26, 1951).

6. Gibson, Eleanor J., Principles of Perceptual Learning and Development, Appleton-Century Crofts, 1969, New York.

7. Gray, S. B., Local Properties of Binary Images in Two and Three Dimensions, Information International, Report, I/21/70.

8. Griffith, Arnold K., Computer Recognition of Prismatic Solids, M.I.T. Ph.D. Thesis (EE), June 1970.

9. Guzman-Arena, Adolfo, Computer Recognition of Three-Dimensional Objects in a Visual Scene, MAC-TR-59 (Thesis), Project MAC, December 1968.

10. Horn, Berthold Klaus Paul, Shape from Shading: A Method for the Dedermination of the Shape of a Smooth Opaque Object from One View, M. I. T. Ph.D. Thesis, Electrical Engineering, June 1970

11. Kirsch, Russell A., Computer Determination of the Constituent Structure of Biological Images, National Bureau of Standards Report 10173

12. Krakauer, Lawrence J., CNTOUR, A.I. Memo 113A, January 1968.

13. Prewitt, J. M. S., and Mendelsohn, M. L., The Analysis of Cell Images, Annals of the New York Academy of Science 128 (January 1966), (pages 1035-1053).

14. Rindfleisch, T., Photometric Method for Lunar Topography, Jet Propulsion Laboratory Technical Report 32-786, September 15, 1965.

15. Roberts, L. G., "Machine Perception of Three-dimensional Solids", Optical and Electro-optical Information Processing, James J. Tippett et al (ed.), The MIT Press, Cambridge, Mass. (1965), (page 159).

16. Rosenfeld, A., "Automatic Recognition of Basic Terrain Types from Aerial Photographs", Photogram. Eng. 28 (March 1962), (pages 115-132).