Vernon, D. 1985 'A hierarchically-organized robot vision system', *Proceedings of AI EUROPA*, Wiesbaden, West Germany.

Volz, R.A., Mudge, T.N. and Gal, D.A. 1983 *Using Ada as a Programming Language for Robot-Based Manufacturing Cells*, RSD-TR-15-83. Centre of Robotics and Integrated Manufacturing, Robot Systems Division, College of Engineering, University of Michigan.

Yachida, M., Tsuji, S. and Huang, X. 1982 'Wiresight – a computer vision system for 3-D measurement and recognition of flexible wire using cross stripe light', *Proceedings of the 6th International Conference on Pattern Recognition*, Vol. 1, pp. 220–2.

9

Introduction to image understanding

9.1 Representations and information processing: from images to object models

This book began by borrowing a phrase from David Marr and defining computer vision as the endeavour to 'say what is where in the world by looking' by the automatic processing and analysis of images by computer. We immediately distinguished between industrial machine vision and image understanding, identifying the former as the heavily engineered pragmatic application of a small sub-set of the broad spectrum of imaging techniques in quite restricted, and often two-dimensional, domains. Image understanding, on the other hand, addresses general three-dimensional environments where one lifts the restrictions on the possible organization of the visual domain. Thus, image understanding must take into consideration the considerable loss of information which arises when a three-dimensional world is imaged and represented by two-dimensional digital images. In particular, it must address the recovery of range or depth information. We noted that a considerable amount of effort is expended in accomplishing this. Again, approaches which are associated with image understanding endeavour to avoid intrusive sensing techniques, i.e. imaging systems which depend on the transmission of appropriate signals (infra-red, laser, or ultrasonic beams; or grids of structured light) to facilitate the process. Instead, a more passive approach is adopted, using whatever ambient information exists, in an *anthropomorphic* manner.

There is, however, more to image understanding than just the recovery of depth information. If we are going to be able to identify the structure of the environment, we need to do more than develop a three-dimensional range map since this is still an image and the information which we need is still *implicit*. We require an *explicit* representation of the structure of the world we are imaging. Hence, we still need the process of segmentation and object recognition that we dealt with in the preceding chapters. Our difficulty is that now the data we are dealing with is much more complex than before and the simplistic image segmentation, template

matching or feature extraction and classification paradigms are wholly inadequate. To be able to proceed from raw two-dimensional images of the world to explicit three-dimensional structural representations, we must adopt a much more sophisticated stance and we must acknowledge that no single process or representation is going to be generally adequate. Thus, there is one central theme which runs through the current, and now conventional, approach to image understanding: that we require intermediate representations to bridge the gap between raw images and the abstracted structural model.

These representations should make different kinds of knowledge explicit and should expose various kinds of constraint upon subsequent interpretations of the scene. It is the progressive integration of these representations and their mutual constraint to facilitate an unambiguous interpretation of the scene that most characterizes the branch of vision known as image understanding. It is perhaps useful to note that most of the progress that has been made in the past few years has not, in fact, been in this area of representation integration, or *data fusion* as it is commonly known, but rather in the development of formal and well-founded computational models for the generation of these representations in the first place. As we shall see, this is no accident. Nevertheless, there remains a great deal of work to be done in the area of data fusion.

In summary then, we can characterize image understanding as a sequence of processes concerned with successively extracting visual information from one representation (beginning with digital images), organizing it, and making it explicit in the representation to be used by other processes. From this perspective, vision is computationally modular and sequential. What we must now proceed to look at are the possible organizations of visual processes, the representations, and the visual processes themselves. We will begin with the organization of visual processes, emphasizing one particular approach and giving an overview of the area; we will then proceed to the other two component topics.

9.2 Organization of visual processes

At present, it is not clear how information in one representation should influence the acquisition and generation of information in another representation. Some possibilities include the following three:

1. A bottom-up flow of data in which information is made explicit without recourse to *a priori* knowledge. Thus, we form our structural representation purely on the basis of the data implicit in the original images.
2. Heterarchical constraint propagation. This is similar to the bottom-up approach but we now have the additional constraint that cues, i.e. a given information representation, at any one level of the hierarchical organization of representations can mutually interact to delimit and reduce the possible forms of interpretation at that level and, hence, the generation of the

information representations at the next level of the hierarchy. Perhaps one of the simplest examples is the integration of depth values generated by two independent processes such as binocular stereo and parallax caused by a moving camera. This approach typifies much current thinking in image understanding.
3. A top-down, *model driven*, information flow whereby early vision is guided by firm expectations of what is to be seen. It should be noted that this model-based vision is quite different from the knowledge-based approach which was fashionable in artificial intelligence a decade ago, in which the effort was to design control processes that could utilize the appropriate knowledge at the appropriate time, with inadequate attention being paid to the representations used, the processes using them, and, indeed, the reasons for utilizing those representations.

We will restrict our attention in the remainder of this chapter to the first approach to image understanding. It should be noted that this emphasis is more by way of convenience than by way of making a statement about the relative importance of the three approaches. Model-based vision, for example, is currently a very popular paradigm and there is a wealth of interesting work being done in the area. The present enthusiasm for data fusion, too, in the computer vision community reflects the importance of the approach of heterarchical constraint propagation. Both these topics would require a volume each to do them justice. We choose the bottom-up approach here because it is often an essential component of the other two paradigms and it will serve to highlight the fundamental visual representations and processes without clouding the issue with additional considerations, however important they might be.

David Marr, at MIT, exerted a major influence on the development of this new computational approach to vision. Marr modelled the vision process as an information processing task in which the visual information undergoes different hierarchical transformations at and between levels, generating representations which successively make more three-dimensional features explicit. He indicated that there are three distinct levels at which the problem must be understood: a computational level, a representational and algorithmic level, and an implementation level. The computational level is concerned with the 'what' and 'why' of the problem: what is to be computed, why it is to be computed, and what is the best strategy for computing it? The representational and algorithmic levels address the 'how' of the problem: how is this theory to be implemented? This necessitates a representation for the input and the output. The third level, hardware implementation, addresses the problem of how to realize these representations and algorithms physically. Thus, this approach removes one from the image environment and addresses the more fundamental problems of the task, formulating a theory of computation and then putting this theory into effect by applying the available visual abilities.

The importance of a computational theory is stressed very strongly by Marr and he illustrates his case by pointing out that attempting to understand perception

solely from the neurological standpoint is as fruitless as trying to understand bird flight from the study of feathers. One must first understand aerodynamics; only then does the structure of the feathers make sense. Marr's contribution was not simply confined to championing this new, and now popular, approach; he also detailed techniques for implementing these philosophies. His idea was, in essence, to generate successive representations of information, each representation being richer in the type of information it makes explicit. We shall now have a brief look at each of these representations.

9.3 Visual representations

9.3.1 The raw primal sketch

Beginning with a grey-level image, Marr proposed the generation of a *Raw Primal Sketch*, which consists of primitives of *edges, terminations, blobs*, and *bars* at different scales. We will explain each of these four terms shortly. Each primitive has certain associated properties: orientation, width, length, position, and strength. The computation of the raw primal sketch requires both the measurement of intensity gradients of different scale and the accurate measurement of location of these changes. This causes a problem, however, since no single measuring device can be optimal simultaneously at all scales. For example, a watchmaker will use a micrometer for very fine measuring work and calipers for coarse work, both of which are useless to a surveyor measuring up a building site. Thus, we need an edge detector which can operate at different scales and is well-localized in the frequency domain, that is, the spatial frequencies (the rate of variation of image intensity with distance) to which it is sensitive are confined to given ranges. Since edges are also localized in space (i.e. in the image), this means that the measuring device must also be spatially localized.

Obviously, the requirements of spatial localization and confinement in spatial frequency are at variance: devices for measuring variation over large distances must themselves be large. Marr and Hildreth proposed a compromise. They suggested that one should use smoothing filters to select information in the grey-level intensity image at different scales, but choose a type of smoothing that optimizes these two opposing demands (of spatial frequency and positional localization). The Gaussian distribution is the only appropriate smoothing distribution with these properties. Marr suggested that the raw primal sketch should be generated using the Marr–Hildreth theory of edge detection (which we discussed in Chapter 5), i.e. by convolving the image with Laplacian-of-Gaussian functions, each Gaussian having a different standard deviation (hence the facility for analysis at different scales). Points at which the resultant image go from positive to negative, i.e. zero-crossings, are isolated; see Figures 9.1–9.4. These points correspond to instances of sharp intensity change in the original image. Note that these images have been automatically post-processed to remove spurious noisy zero-crossings. This is

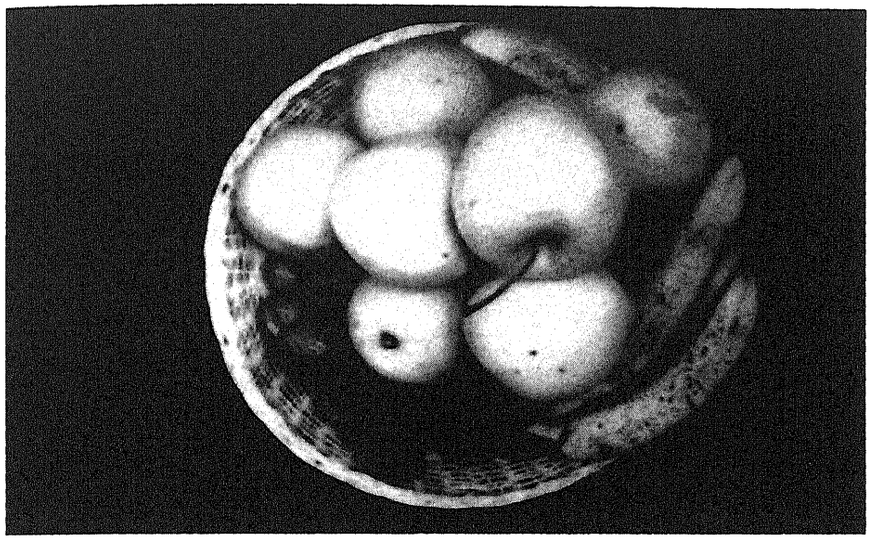


Figure 9.1 A grey-scale image.

accomplished by analysing various features of the contour points and comparing their values to statistics of these features collected from the entire image (see Vernon, 1988).

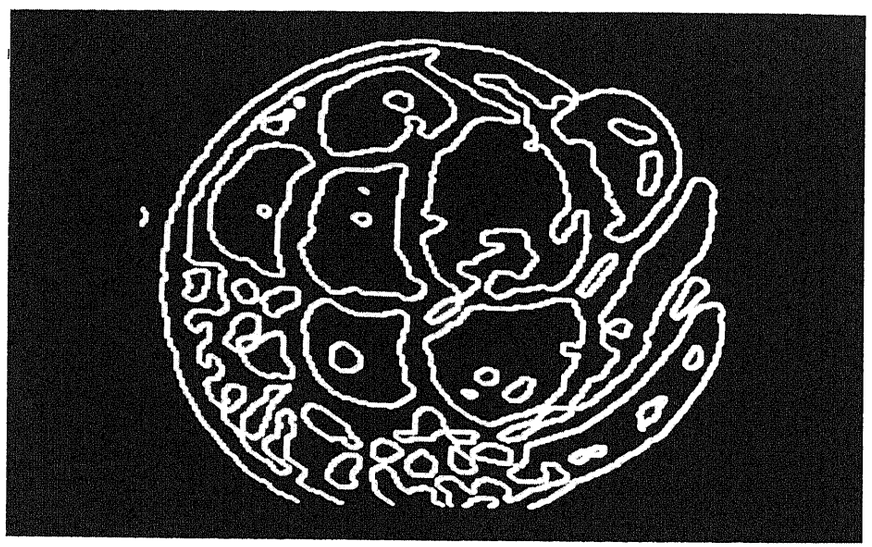
By analysing the coincidence of zero-crossings in particular positions at different scales, one can infer evidence of physical reality, i.e. of a real physical edge, and it is these spatially coincident and hence (hopefully) real physically meaningful zero-crossings that are then represented by the primitives of the raw primal sketch. We can now define each of the four primitives.

Edge primitives are, effectively, local line segment approximations of the zero-crossing contours (see Figure 9.5); curves comprise a sequence of edges, delimited at either end by the *termination* primitives (see Figure 9.6). Instances of local parallelism of these edges are represented by *bars* (see Figure 9.7), while blobs represent the zero-crossing contours which are not present, i.e. which have no spatial coincidence, in *all* of the zero-crossing images derived at the different scales, i.e. using different standard deviations for the Gaussian smoothing (see Figure 9.8).

9.3.2 The full primal sketch

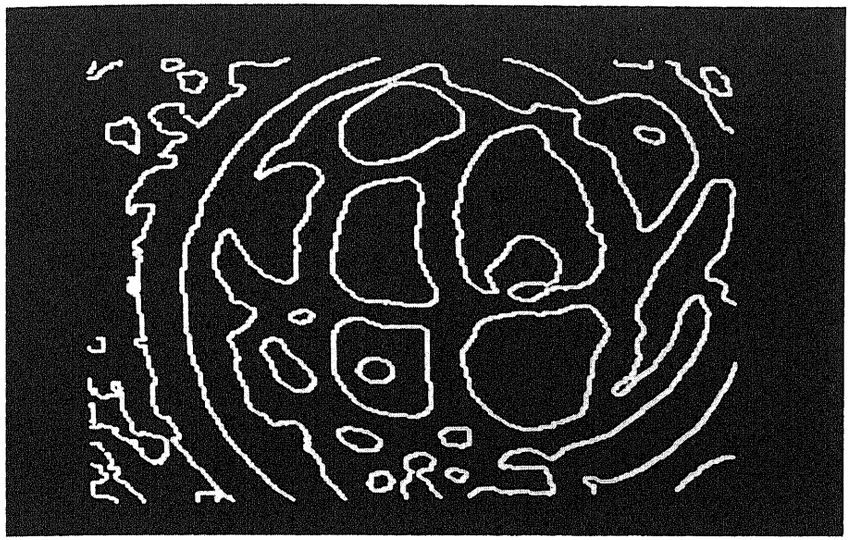
As the information made explicit in the raw primal sketch is still local and spatially restricted, i.e. it does not convey any global information about shape in an explicit manner, we now wish to group these primitives so that the groups correspond to physically meaningful objects. In this sense, the grouping process is exactly what

(a)

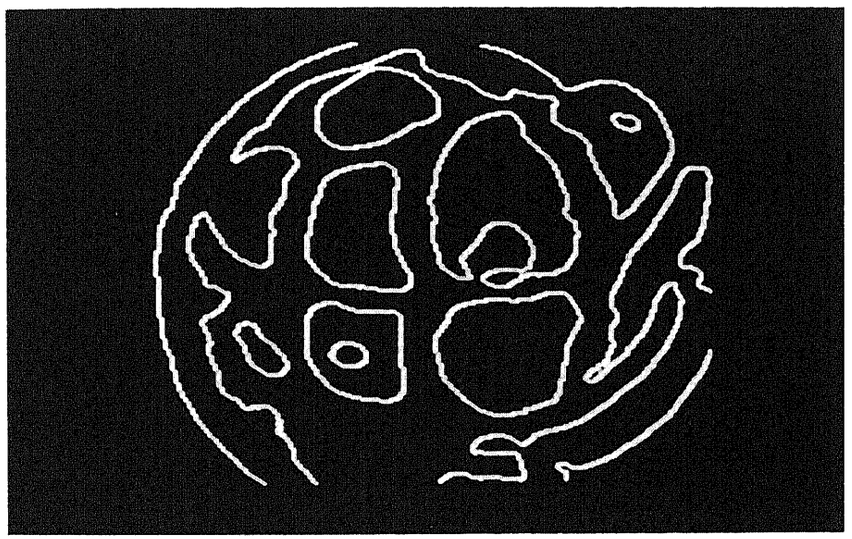


(b)

Figure 9.2 Zero-crossings derived by convolving the image with a Laplacian of Gaussian filter in which the standard deviation of the Gaussian is 3.0 pixels: (a) all zero-crossings; (b) after removal of noisy zero-crossings.

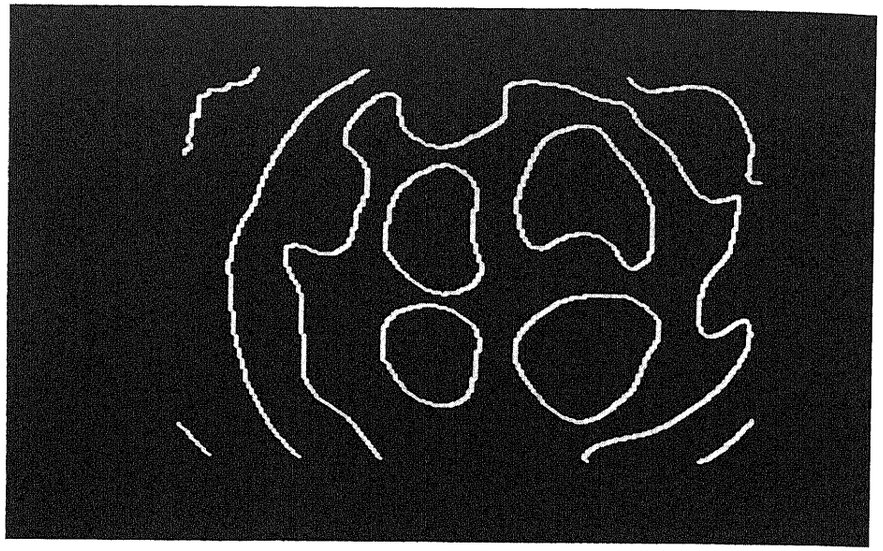


(a)

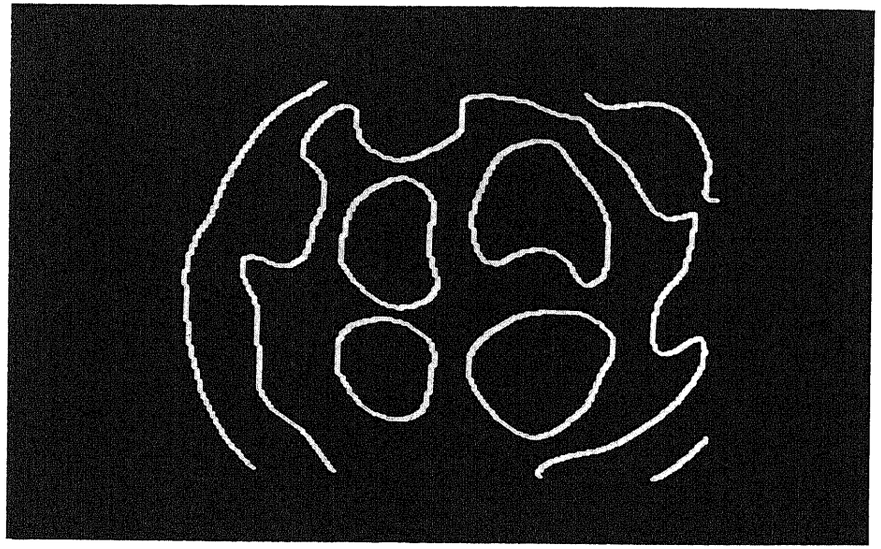


(b)

Figure 9.3 Zero-crossings derived by convolving the image with a Laplacian of Gaussian filter in which the standard deviation of the Gaussian is 6.0 pixels: (a) all zero-crossings; (b) after removal of noisy zero-crossings.



(a)



(b)

Figure 9.4 Zero-crossings derived by convolving the image with a Laplacian of Gaussian filter in which the standard deviation of the Gaussian is 9.0 pixels: (a) all zero-crossings; (b) after removal of noisy zero-crossings.

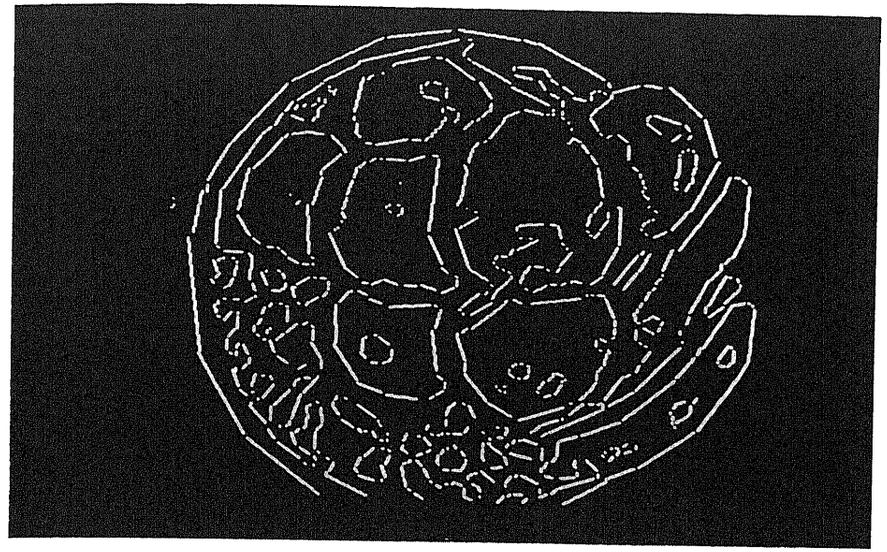


Figure 9.5 The raw primal sketch: edges.

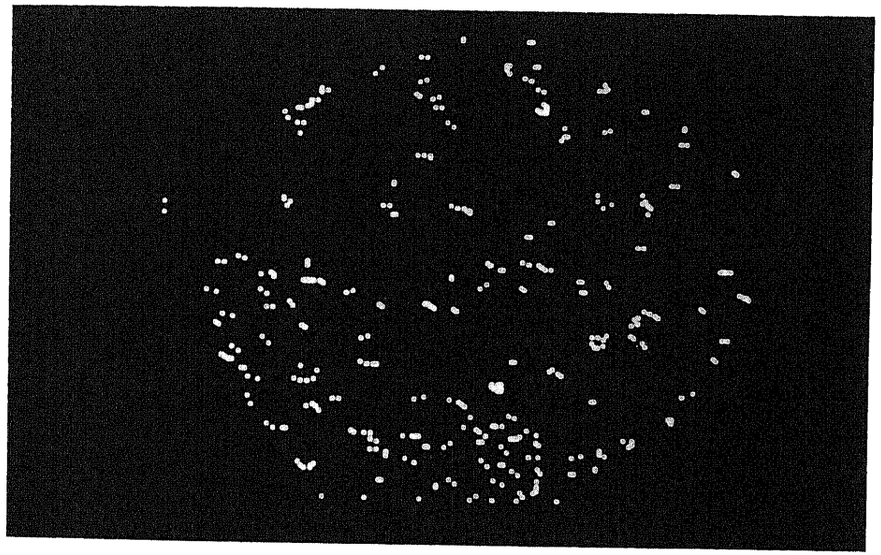


Figure 9.6 The raw primal sketch: terminations.

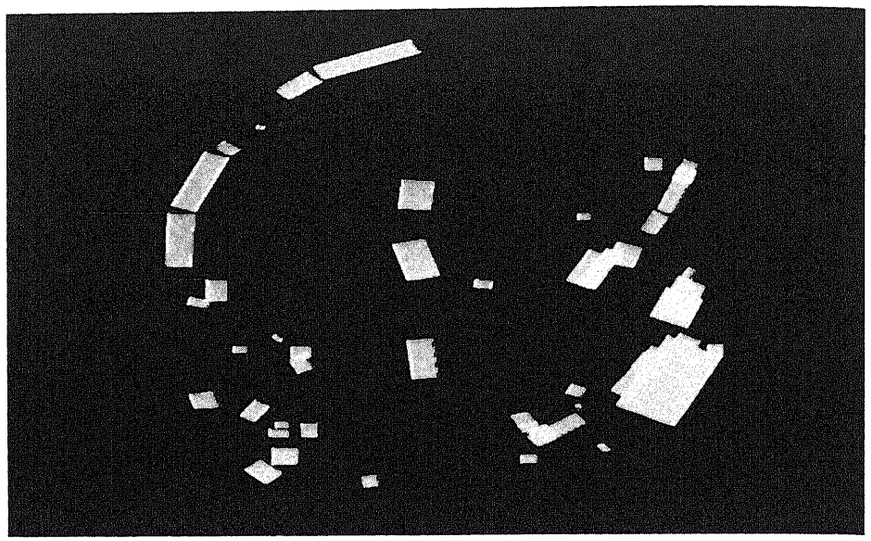


Figure 9.7 The raw primal sketch: bars.

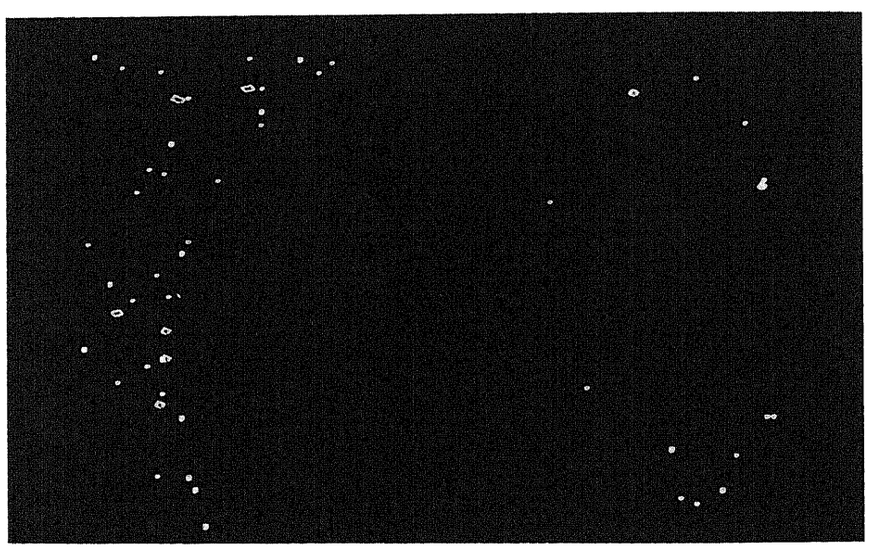


Figure 9.8 The raw primal sketch: blobs.

we mean by segmentation, which we described in Chapter 5. Since we are now dealing with more complex data, the segmentation processes must be more sophisticated. Many of them are based on Gestalt 'figural grouping principles', named after the Gestalt school of psychology formed in the early part of this century. For example, primitives can be grouped according to three criteria: *continuity, proximity*, and *similarity*. In the first case, for instance, lines bounded by terminations which are co-linear would be grouped. Primitives which are spatially proximate are also candidates for the formation of groups, while so too are primitives which belong to the same class, i.e which are similar. These criteria are not independent and the final groupings will be based on the relative weights attached to each criterion.

The contrived example shown in Figure 9.9, comprising three types of fictitious token, illustrates these three criteria. The horizontal row would be grouped on the basis of spatial proximity and continuity; the vertical columns would be grouped on the basis of similarity and, to an extent, continuity, whereas the single diagonal group is formed on the basis of continuity of absence of tokens. This may seem a little strange: grouping entities which do not exist. However, it becomes a little more sensible when we note that, apart from the criteria of grouping, there are also definite grouping principles. The first, *the principle of explicit naming*, states that grouped entities are to be treated as single entities and given an explicit name. This means that they become tokens or primitives in their own right and can be subsequently grouped. Thus, grouping to form the primal sketch is essentially a recursive process; the same grouping principles and criteria applying to all levels of the group hierarchy. Now the situation in Figure 9.9 becomes a little clearer. The groups we identified are not necessarily achieved in one pass of the grouping algorithm; typically, it would take two: one to form the groups identified in Figure 9.10 and a second pass to group the new horizontal groups into the horizontal groups originally shown in Figure 9.9. Now, however, we have new tokens comprising the termination points of these intermediate horizontal tokens; it is the grouping of these terminations, on the basis of continuity, that forms the diagonal group.

There is a second grouping principle, the *principle of least commitment*, which is pragmatic in nature. It addresses the problem that, since grouping can be applied recursively to tokens, groups can be formed which do not segment the physical objects successfully, and it may be necessary to backtrack and undo a grouping process. The principle of least commitment states that these 'mistakes' are expensive and, therefore, grouping should be applied conservatively.

In summary the outcome of such grouping, the full primal sketch, makes region boundaries, object contours and primitive shapes explicit. It is a segmented representation and it exploits processes which are knowledge-free.

9.3.3 The two-and-a-half-dimensional sketch

The next level of representation is the two-and-a-half dimensional sketch. This is

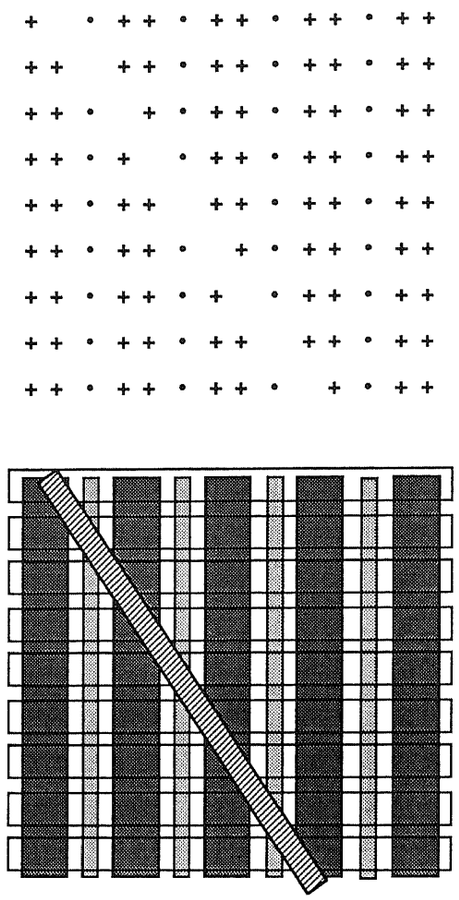


Figure 9.9 Grouping processes.

derived both from the full primal sketch and from the grey-level image by using many visual cues, including stereopsis, apparent motion, shading, shape, and texture. We will have a brief look at the first three of these visual processes in Sections 9.4.1, 9.4.2 and 9.4.3. The two-and-a-half-dimensional sketch is a rich viewer-centred representation of the scene containing not only primitives of spatial organization and surface discontinuity, but also of the local surface orientation at each point and an estimate of the distance from the viewer. The surface orientation is usually represented by a unit vector which is normal to the surface; this is commonly referred to as the 'surface normal vector' (see Figure 9.11).

Recall from Chapter 8 that there are three degrees of freedom in the general

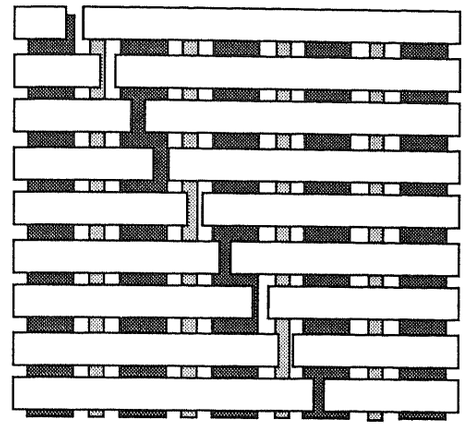


Figure 9.10 Recursive grouping.

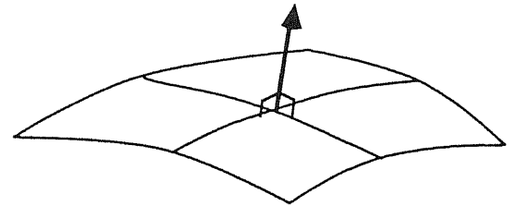


Figure 9.11 The surface normal vector.

specification of orientation of a vector. For example, we used the roll, pitch, and yaw angles to specify the orientation of the robot end effector. In this case, however, we only require two angles since the vector is symmetric and the roll angle is redundant. Thus, the two-and-a-half-dimensional sketch can be thought of as a two-dimensional array of 3-valued entities, representing the distance from the camera to the surface and the two angles specifying the surface normal vectors.

Figure 9.12 shows a schematic of the two-and-a-half-dimensional sketch for a spherical object; the surface normal vectors are projected on to the image plane so that vectors at the boundary of the sphere are the longest and those at the centre, facing the viewer, are the shortest and, in effect, have no length. The intensity with which the vectors are drawn is proportional to their distance from the camera so that the boundary vectors are the brightest and the central vectors are the darkest. It is important to realize that this array-based, or *iconic*, representation is not the complete two-and-a-half-dimensional sketch; since it is also based on the full primal sketch, it integrates the information about grouping and segmentation. Thus, it is a *viewer-centred* three-dimensional model, in that all distances are defined with

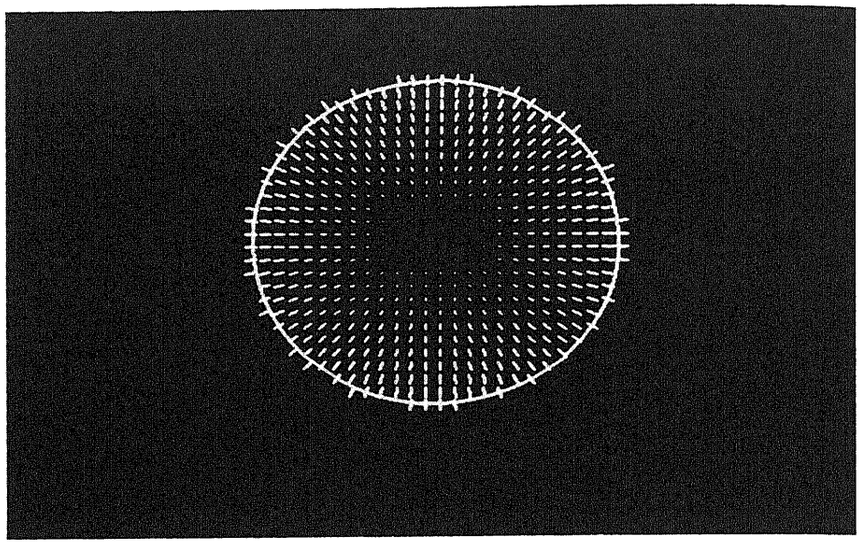


Figure 9.12 The two-and-a-half-dimensional sketch.

respect to the camera coordinate system, with integrated surface-based object representations.

Some of the visual processes which are involved in deriving local surface orientation and depth are discussed in Section 9.4.

9.3.4 Three-dimensional model

The final stage of this information processing organization of visual processes lies in the analysis of the two-and-a-half-dimensional sketch and the production of an explicit three-dimensional representation. There are two issues which must be addressed:

1. The conversion from a viewer-centred representation to an object-centred representation. This is, in effect, the transformation between a camera coordinate system and the real-world coordinate system and is exactly the process we discussed in Chapter 8 in the section on the camera model.
2. The type of three-dimensional representation we choose to model our objects. There are three types of three-dimensional representation based on volumetric, skeletal, and surface primitives.

9.3.4.1 Volumetric representations

Volumetric representations work on the basis of spatial occupancy, delineating the

segments of a three-dimensional workspace which are, or are not, occupied by an object. The simplest representation utilizes the concept of a *voxel image* (voxel derives from the phrase volumetric element) which is a three-dimensional extension of a conventional two-dimensional binary image. Thus, it is typically a uniformly sampled three-dimensional array of cells, each one belonging either to an object or to the free space surrounding the object. Because it is a three-dimensional data-structure, voxel image requires a significant amount of memory and, hence, tends to be quite a coarse representation. For example, a 1 m^3 work space comprising 1 cm^3 voxels will require approximately 1 Mbyte of memory (assuming that a voxel is represented by one byte; although eight voxels could be packed into a single byte, you do not then have simple direct access to each voxel).

The oct-tree is another volumetric representation. However, in this instance, the volumetric primitives are not uniform in size and you can represent the spatial occupancy of a work space to arbitrary resolution in a fairly efficient manner. The oct-tree is a three-dimensional extension of the quad-tree which we discussed in Chapter 5. In the same way as a quad-tree described the occupancy of a two-dimensional image by identifying large homogeneous areas in the image and representing them as single nodes in a tree, the position in the tree governing the size of the region, so too is the three-dimensional spatial occupancy represented by such a tree. Initially, the work space is represented by a single cubic volume. If the work space is completely occupied by an object (a very unlikely situation), then the work space is represented by a single root node in the oct-tree. In the more likely situation that the volume is not completely occupied, the cube is divided into eight sub-cubes of equal volume. Again, if any of these sub-cubes are completely occupied then that volume is represented by a node at the second level in the tree; alternatively, the sub-cube is further sub-divided into another eight cubes and the same test for complete spatial occupancy is applied. This process is reiterated until we reach the required resolution for spatial occupancy, i.e. the smallest required cubic volume: this is equivalent to the voxel in the preceding representation.

Note that, because the sub-division of volumes, and hence the generation of new nodes in the oct-tree, only takes place at the boundary between the object and the free-space surrounding it, this representation is extremely efficient (in terms of storage requirements) for regular solids since the interior of the object is represented by a few nodes close to the root of the tree. Furthermore, the coarseness of the object representation is easily controlled by limiting the depth to which the tree can grow, i.e. by placing a limit on the size of the smallest volumes.

9.3.4.2 Skeletal representations

The generalized cylinder, also referred to as the 'generalized cone', is among the most common skeletal three-dimensional object representations. A generalized cylinder is defined as the surface created by moving a cross-section along an axis. The cross-section may vary in size, getting larger or smaller, but the shape remains the same and the axis may trace out any arbitrary three-dimensional curvilinear

path. Thus, a single generalized cylinder can represent, e.g., a cone (circular cross-section; linear decrease in diameter; linear axis), a sphere (circular cross-section; sinusoidal variation in diameter; linear axis), or a washer (rectangular cross-section; constant area; circular axis). However, a general three-dimensional model comprises several generalized cones and is organized in a modular manner with each component subsequently comprising its own generalized cylinder-based model. Thus, the three-dimensional model is a hierarchy of generalized cylinders; see Figure 9.13.

9.3.4.3 Surface representations

Finally, we come to the third type of three-dimensional model which is based on surface representations. We immediately have a choice to make and to decide which type of surface primitives (or surface patches) we will allow: planar patches or curved patches. Although there is no universal agreement about which is the better, the planar patch approach is quite popular and yields polyhedral approximations

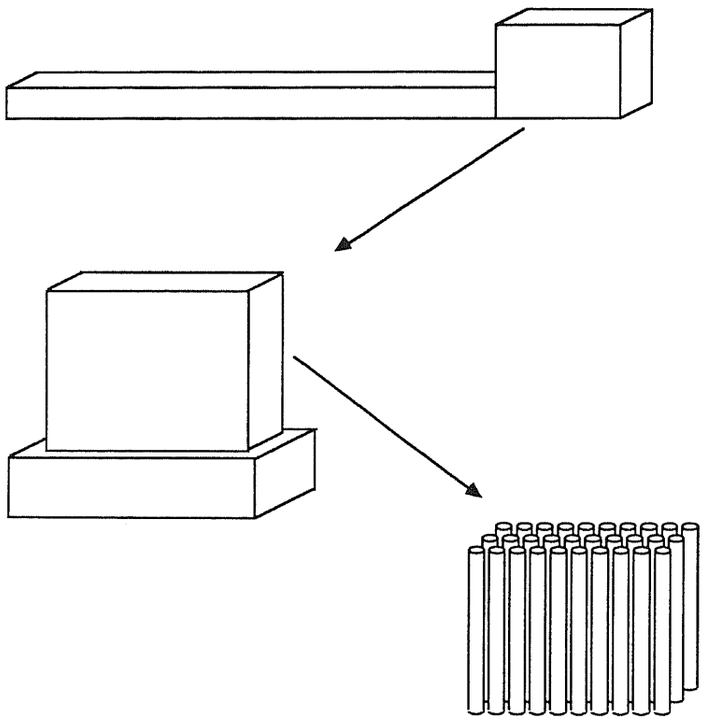


Figure 9.13 Generalized cylinder representation of a tooth-brush.

of the object being modelled. This is quite an appropriate representation for man-made objects which tend predominantly to comprise planar surfaces. It is not, however, a panacea for three-dimensional representational problems and it would appear that many of the subtleties of three-dimensional shape description cannot be addressed with simplistic first-order planar representations. Nevertheless, it does have its uses and, even for naturally curved objects, it can provide quite a good approximation to the true shape, if an appropriate patch size is used. For example, Figure 9.14 shows a sphere comprised of triangular planar patches.

To conclude this brief overview of three-dimensional object representations, it should be noted that we have not addressed the usefulness of these models for object recognition. The subject of three-dimensional shape representation and matching is a difficult one and there are at present no entirely satisfactory answers to the problems it poses. At the same time, we can make a few short comments about the representations which we have discussed. Note first, however, that these comments are made in the context of ideal object representations: as we will see in the next few sections, the three-dimensional models that we build directly from two-dimensional images tend to be incomplete, noisy and often are only approximations to the expected form.

The voxel image represents the three-dimensional shape in an implicit manner (in the same way as a two-dimensional binary image represents two-dimensional shapes) and recognition can be effected by three-dimensional template matching. However, all the problems associated with this technique (see Chapter 6) still apply and, indeed, they are exacerbated by the increase in the dimensionality of the model.

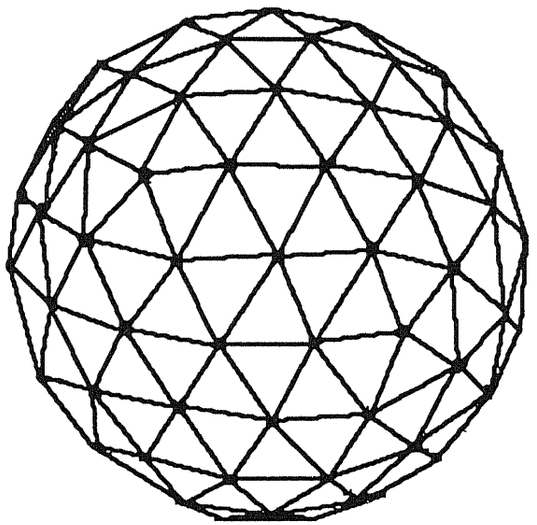


Figure 9.14 A sphere comprised of triangular planar patches.

Object recognition using oct-trees essentially requires a graph (or tree) matching algorithm. However, since the oct-tree will vary with the orientation of the object, especially at the lower levels, the matching algorithm must be able to detect structural similarities in sub-sets of the tree.

Similarly, a tree matching technique can be used to facilitate matching models based on generalized cylinders, since the representation is a hierarchical structure of nodes comprising three parametric entities: the axis, the cross-section, and the variation of cross-section.

Finally, it is difficult to accomplish object recognition using planar surface models for anything except the simplest of regular objects since the size and position of each patch, and the total number of patches, will vary considerably with the orientation of the original object and the completeness of the data. Instead, object recognition can be effected by backtracking somewhat and regenerating an iconic (array-based) version of the two-and-a-half-dimensional sketch. The recognition process is accomplished by matching this two-dimensional representation with a template. It assumes, however, that the template and object poses have first been registered; otherwise, several (indeed a very large number) of object templates will have to be used in the template matching process, each one generated by projecting the three-dimensional model on to the two-and-a-half-dimensional sketch at different orientations. It is this task of three-dimensional pose estimation to which we now turn.

9.3.5 The extended Gaussian image

The two-and-a-half-dimensional sketch provides us with the local orientation at each point on the surface of the object. These surface normals can be represented on a unit sphere called a Gauss map. If we attach a unit mass to each endpoint, we now observe a distribution of mass over the Gaussian sphere and this distribution is called the 'extended Gaussian image' (EGI) of the object. The EGI is effectively a two-dimensional histogram of surface normal orientations, distributed on a sphere, and the attitude, or pose, of an object is greatly constrained by the global distribution of EGI mass over the visible Gaussian hemisphere. Constraints on the viewer direction can be derived from the position of the EGI mass centre, and from the directions of the EGI axis of inertia. To determine object pose, we simply match partial EGIs, derived from the observed scene, with template EGIs derived from our stored prototypical model.

For example, Figure 9.15 shows a contrived, e.g. artificial, object in several poses, together with the associated EGIs. In this case, the mass at a point on the EGI (which is proportional to the surface area of the object with that attitude) is coded by grey-level. Note that the distribution of mass is not uniform; this is caused by quantization errors in the orientation of the facets of both the Gaussian sphere and the object surfaces.

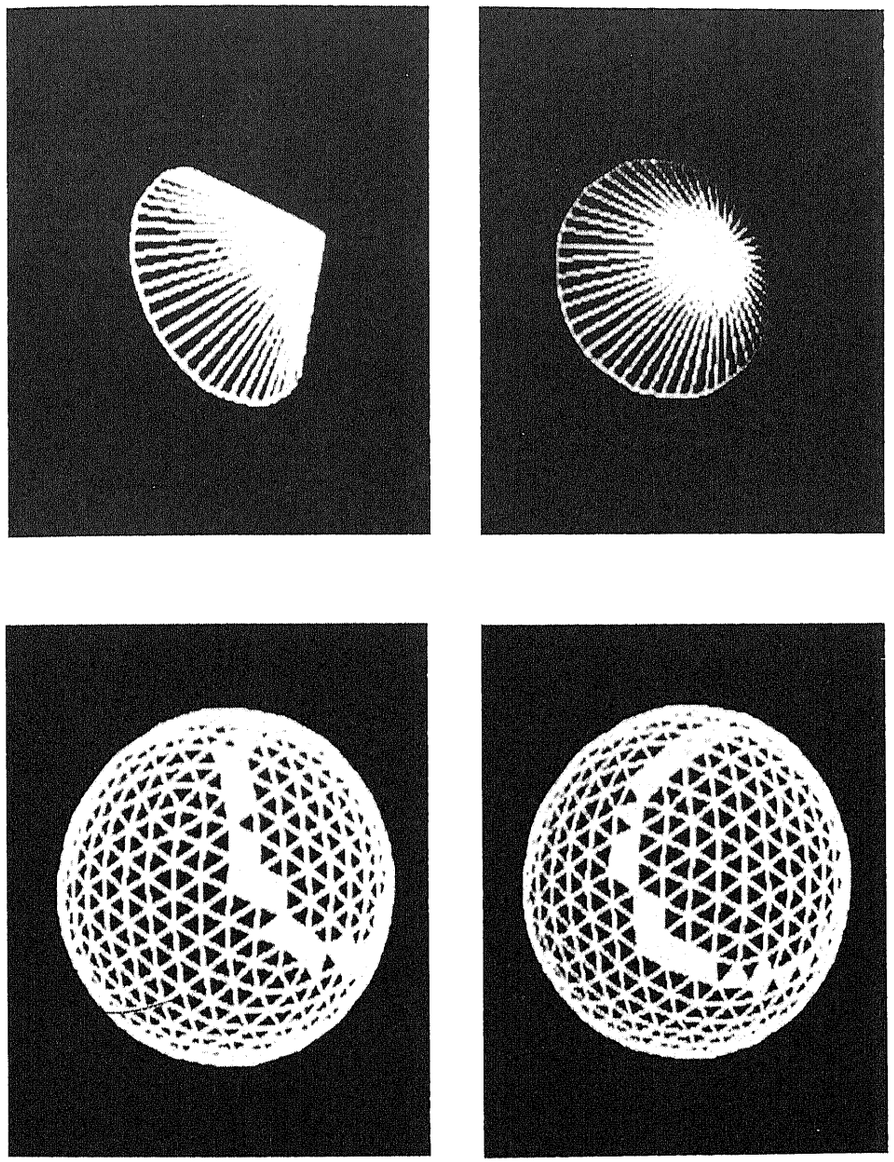


Figure 9.15 An artificial object and its extended Gaussian image (EGI).

9.4 Visual processes

We began this chapter by suggesting that there is a great deal more to image understanding than the recovery of depth information from two-dimensional images, and we then proceeded to look at the representations and organization of visual processes which comprise an image understanding system. Nevertheless, as we have stated several times, the recovery of depth information is important and so we return now to the visual processes which are involved constructing the two-and-a-half-dimensional sketch: the computation of depth information and of local surface orientation.

9.4.1 Stereopsis

Our interest here is in the use of two views of a scene to recover information about the distance of objects in the scene from the observer (the cameras). In a sense, we have already discussed stereopsis, or stereo for short, in the last chapter when we dealt with the camera model, the inverse perspective transformation, and the recovery of depth information. We discussed in detail how the inverse perspective transformation allows us to construct a line describing all the points in the three-dimensional world which could have been projected onto a given image point. If we have two images acquired at different positions in the world, i.e. a stereo pair, then for the two image coordinates *which correspond to a single point in three-dimensional space*, we can construct two lines, the intersection of which identifies the three-dimensional position of the point in question. Thus, there are two aspects to stereo imaging:

(a) the identification of corresponding points in the two stereo images;
(b) the computation of the three-dimensional coordinates of the world point which gives rise to these two corresponding image points.

Since we have already covered the second aspect in detail (and you are encouraged to review these techniques in Chapter 8), the main problem in stereo is to find the corresponding points in the left and right images and this is what we will discuss here. Before we proceed, however, it is perhaps worth while noting that quite often stereo techniques are presented without discussing the inverse perspective transformation formally; instead, stereo is discussed in the context of the computation of the stereo disparity (i.e. the relative shift in position of an imaged point in the two stereo images) and the subsequent computation of depth on the basis of knowledge of the geometry of the two camera systems, i.e. the focal length of the lens, the distance between the lens centres, the size of the sensor, and the relative attitude of the focal axes of the two cameras. We have not done this here because it neglects the calibration of the camera systems: it is difficult to measure the distance between lens centres empirically, the relative attitude of focal axes, and

the focal length quoted on camera lenses is only nominal; quite often, two 'identical' lenses will have slightly different focal lengths.

We return now to the stereo correspondence problem. In characterizing a stereo system, we must address the following:

(a) the kind of visual entities on which the stereo system works;
(b) the mechanism by which the system matches visual entities in one image with corresponding entities on the other.

Typically, the possible visual entities from which we can choose include, at an iconic level, zero-crossing points, patches (small areas) in the intensity image, and patches in filtered images; or, at a more symbolic level, line segment tokens, such as are made explicit in the raw primal sketch.

The matching mechanism which establishes the correspondence will depend on the type of visual entities which we have chosen: iconic entities will normally exploit some template matching paradigm, such as normalized cross-correlation (see Chapter 6), while token entities can be used with more heuristic search strategies. As an example, the stereo disparity image shown in Figure 9.16,* in which disparity is inversely proportional to grey-level, is derived by convolving the stereo pair (see Figure 9.17) with a Laplacian of Gaussian filter, computing the zero-crossings, and correlating patches in the convolved image, centred on the zero-crossing points at discrete intervals along the zero-crossing contour.

9.4.2 Camera motion

The analysis of object motion in sequences of digital images, or of apparent motion in the case of a moving observer, to provide us with information about the structure of the imaged scene, is an extremely topical and important aspect of current image understanding research. However, the general object motion problem is difficult, since the motion we perceive can be due to either the rotation of the object, a translation of the object, or both. We will not attempt to address this problem here and the interested reader will find references to some seminal work in the area in the bibliography at the end of the book. We confine our attention to the somewhat easier problem of camera motion and, in particular, to the study of apparent motion of objects in a scene arising from the changing vantage point of a moving camera. This restriction is not necessarily a limitation on the usefulness of the technique; on the contrary, the concept of a camera mounted on the end-effector of a robot, providing hand–eye coordination, or on a moving autonomously guided vehicle (AGV), providing navigation information, is both appealing and plausible.

*This disparity representation, which is based on the zero-crossings of Laplacian of Gaussian filtered images, is merely a short-hand way of summarizing the relative position of corresponding points in the two images. Given a zero-crossing point, we can compute its corresponding position in the second stereo image from its disparity, represented here as intensity. This assumes that we know the direction in which the corresponding point lies; for a stereo pair of cameras which have parallel focal axes, the direction is horizontal.

(a)

(b)

Figure 9.16 (a) Left stereo image. (b) Right stereo image.

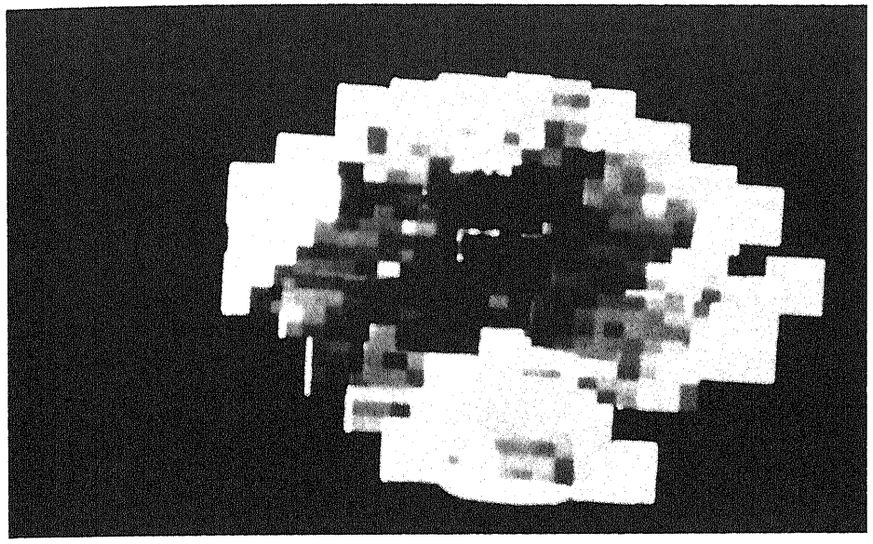


Figure 9.17 Disparity of a stereo pair of images.

From an intuitive point of view, camera motion is identical to the stereo process in that we are identifying points in the image (e.g. characteristic features on an object) and then tracking them as they appear to move due to the changing position (and, perhaps, attitude) of the camera system. At the end of the sequence of images, we then have two sets of corresponding points, connected by *optic flow vectors*, in the first and last images of the sequence. Typically, we will also have a sequence of vectors which tack the trajectory of the point throughout the sequence of images. The depth, or distance, of the point in the world can then be computed in the manner discussed in Chapter 8 and in the preceding section.

However, there are a number of differences. First, the tracking is achieved quite often, not by a correlation technique or by a token matching technique, but by differentiating the image sequence with respect to time to see how it changes from one image to the next. There is often a subsequent matching process to ensure the accuracy of the computed image change, and sometimes it is not the grey-scale image which is differentiated but, rather, a filtered version of it. Nevertheless, the information about change is derived from a derivative (or, more accurately, a first difference) of the image sequence. Second, the 'correspondence' between points is established incrementally, from image to image, over an extended sequence of images. Thus, we can often generate accurate and faithful maps of point correspondence which are made explicit by a two-dimensional array of flow vectors which describe the trajectory of a point over the image sequence.

In this book, we will confine our attention to two simple types of camera motion in order to illustrate the approach. The first describes a trajectory along the

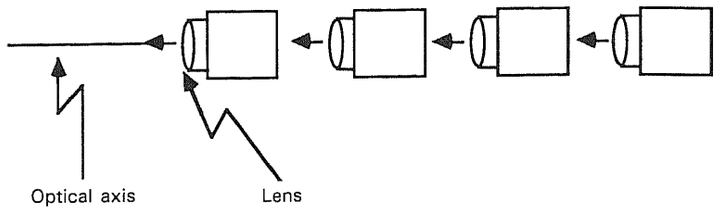


Figure 9.18 Translational motion along the optic axis.

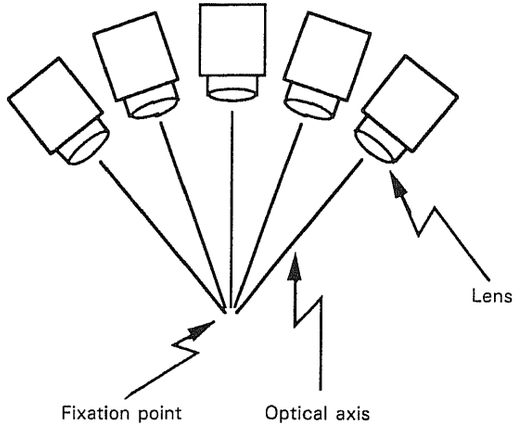


Figure 9.19 Rotational motion about a fixation point.

optical axis of the camera (see Figure 9.18), while in the second the camera is rotated about a fixation point (see Figure 9.19). The optic flow field resulting from this first type of egocentric motion is very easy to compute as all flow vectors are directed radially outward from the focus of expansion (FOE),* i.e. the centre of the image. For camera motion in rotation about a fixation point, the rotational component of optical flow can be determined directly from the known camera trajectory and the direction of the translational component is also constrained by the camera motion. Knowing the direction of the flow vector, the magnitude of the visual motion is directly derived from a time-derivative of a sequence of images acquired at successive points along the camera trajectory.

*The focus of expansion is the point which defines the direction from which all the optic flow vectors appear to emanate, i.e. all flow vectors are co-linear with a line joining the FOE and the origin of the flow vector.

There is one main difficulty when attempting to compute the true optical flow, i.e. the visual motion, of a point in the image. It is generally referred to as the *aperture problem*. Consider a contour in an image and let us say that we only have a small local window of visibility around the point of interest on the contour (see Figure 9.20). If the position of the contour changes due to the camera motion, then we cannot say with any certainty in which direction it moved, based purely on the local information available in this small local window. The only thing we can compute with certainty is the *orthogonal component of the velocity vector*, i.e. the component which is normal to the local contour orientation. This vector component is referred to as $v^{\perp}$, and the second component is referred to as the tangential component, $v^{\top}$. Thus, the true velocity vector v is given by:

$$v = v^{\perp} + v^{\top}$$

If the luminance intensity does not change with time (i.e. there are no moving light sources in the environment) the component of the orthogonal velocity vector for each image point is given by:

$$v^{\perp} = \frac{-\partial I / \partial t}{|\nabla I|}$$

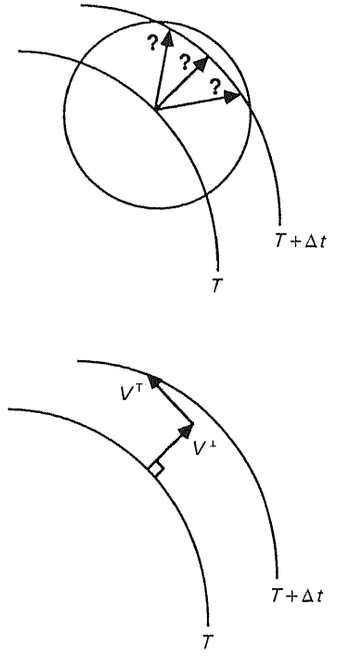


Figure 9.20 The aperture problem.

where ∂ indicates the partial derivative operator and $|\nabla I|$ is the local intensity gradient.

In the technique described here, we compute the time derivative of a $\nabla^2 G$ filtered image instead of the raw intensity image and compute the optical flow at zero-crossing contours. This means that the amount of data to be processed is limited and, furthermore, the effects of noise are less pronounced.

The computation of $v^{\perp}$ (the orthogonal component of velocity) is based on a computation of the time derivative using image subtraction, in the same way as we saw that gradient-based edge detectors can be affected by local pixel differences, according to the relationship described above.

The computation of the true velocity depends on the prior knowledge of the parameters of the camera motion: the position of the camera at time T and $T+\Delta t$ from the fixation point; θ the rotational angle of the camera around the Y-axis, and W_x and W_z the components of the translational velocity of the camera along the X-axis and the Z-axis respectively. The velocities W_x and W_z are defined with respect to the coordinate system of the camera at time T (see Figure 9.21). Using basic trigonometric relations, we find:

$$W_x = \frac{D_2 \sin\theta}{\Delta t}$$

$$W_y = 0$$

$$W_z = \frac{D_1 - D_2 \cos\theta}{\Delta t}$$

where D_1 and D_2 are the distances of the camera from the fixation point at time T and $T+\Delta t$ respectively.

These computed egomotion parameters are used to determine the true image velocity vector v. Note that v comprises two components, v_t and v_r, one due to camera translation $W=(W_x, W_y, W_z)$ and the other to camera rotation $\omega=(\omega_x, \omega_y, \omega_z)$:

$$v_t = \left(\frac{xW_z - FW_x}{Z}, \frac{yW_z - FW_y}{Z}\right)$$

$$v_r = \left(\frac{xy\omega_x - (x^2+F^2)\omega_y + y\omega_z}{F}, \frac{(y^2+F^2)\omega_x - xy\omega_y - x\omega_z}{F}\right)$$

$$v = v_t + v_r$$

where:

F is the focal length of the lens,
x and y are the coordinates of the point in the image plane at time T, and
Z is the distance from the camera to the world point corresponding to image point (x, y).

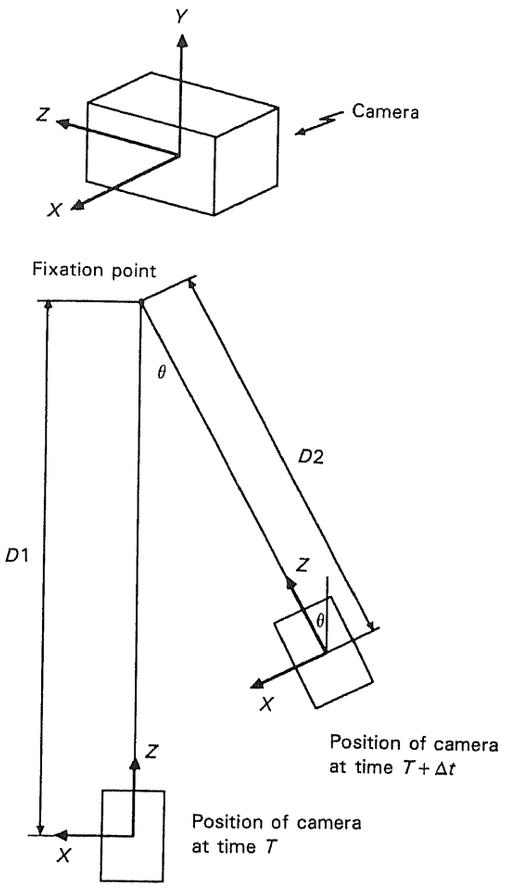


Figure 9.21 Camera coordinate system and the parameters associated with camera motion.

For the constrained camera motion shown in Figure 9.19, the camera translational velocity is given by the equations for W_x, W_y and W_z above, while the rotational velocity ω is $(0, \theta/\Delta t, 0)$.

$$v_t = \left(\frac{x(D_1 - D_2\cos\theta) - FD_2\sin\theta}{Z\,\Delta t}, \frac{y(D_1 - D_2\cos\theta)}{Z\,\Delta t}\right)$$

$$v_r = \left(\frac{-(x^2+F^2)\theta}{F\,\Delta t}, \frac{-xy\theta}{F\,\Delta t}\right)$$

In these two equations for v_t and v_r, the only unknown is Z (which is what we wish

to determine). Thus, to determine v_t and v_r, and hence Z, we exploit the value of $v^\perp$, the orthogonal component of velocity, computed at an earlier stage. This can be accomplished directly by solving the attendant system of equations or by a geometrical construction.

In the solution by geometrical construction, v is determined from the intersection of three straight lines derived from v_r (for which all terms are known), $v^\perp$ (which was computed previously), and the position of the FOE.

First, v_r defines the first line of the construction (refer to Figure 9.22).

Second, the position of the FOE defines the direction of v_t, since v_t is parallel to the line joining the FOE and the point (x, y) in question. Thus, the second line is parallel to v_t and passes through the point given by v_r(see Figure 9.22). The coordinates of the FOE are given by:

$$(x_{FOE}, y_{FOE}) = \left(\frac{FW_x}{W_z}, \frac{FW_y}{W_z}\right)$$

where W_x, W_y, and W_z are the known velocities of the camera in the x-, y-, and z-directions respectively.

Finally, we note again that v is also given by the sum of the orthogonal component and the tangential component of velocity:

$$v = v^\perp + v^\top$$

Since these two vectors are orthogonal to one another, and since $v^\perp$ is known, this relationship defines a third line through the point given by $v^\perp$ and normal to the direction of $v^\perp$. Hence, v is given by the intersection of the second and the third lines: see Figure 9.22.

In the simpler case of translatory motion along the optic axis, θ is equal to

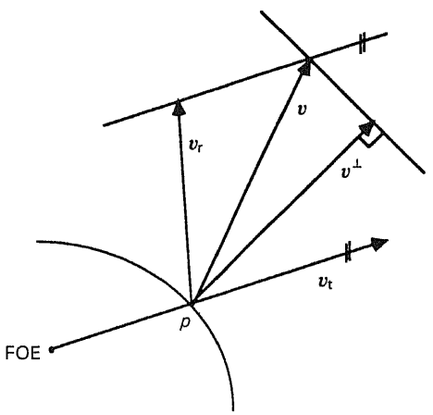


Figure 9.22 Computation of true velocity v from $v^\perp$, $v_t v_r$ at a point P on a zero-crossing contour.

zero and the translational component of velocity reduces to:

$$v_t = \left(\frac{x(D_1 - D_2)}{Z}, \frac{y(D_1 - D_2)}{Z}\right)$$

while the rotational component v_r is now zero.

Computing v in this manner and, in particular, computing $v^\perp$ using image differences, errors can still be recorded in the final flow. A significant improvement can be achieved by performing a contour-to-contour matching between successive frames, along the direction of the flow vectors, tuning the length of the flow vectors to the correct size. The tracking procedure searches in the direction of the flow vector until the next contour is found, then it searches in the direction of the new flow vector, and so forth until the whole image sequence is processed. Although a small difference between successive frames is required to guarantee the accuracy in the computation of the orthogonal component $v^\perp$, a long baseline is required for the depth measurement. For this reason, many images are normally considered and the flow field obtained for a sequence of images is used for range computation: the flow vector from the first image to the last image being employed in the computation of depth.

The algorithm for computing the optical flow can be summarized as follows:

```
Convolve the images with a Laplacian of Gaussian
operator

Extract the zero-crossings

Compute the difference between the ∇²G of successive
frames of the sequence

Compute the velocity component in the direction
perpendicular to the orientation of the contour

Compute the velocity along the contour using the known
motion parameters

Search for the zero-crossings of the second frame
projected from the first frame in the direction of the
velocity vector.
```

The depth, for each contour point, is computed as before by applying the inverse perspective transformation, derived from camera models corresponding to the initial and final camera positions, to the two points given by the origin of the optical flow vector and the end of the optical flow vector.

To illustrate this approach to inferring the depth of objects, motion sequences of two different scenes were generated, each comprising nine images. These scenes were of a white 45° cone with black stripes at regular intervals and a 'toy-town'

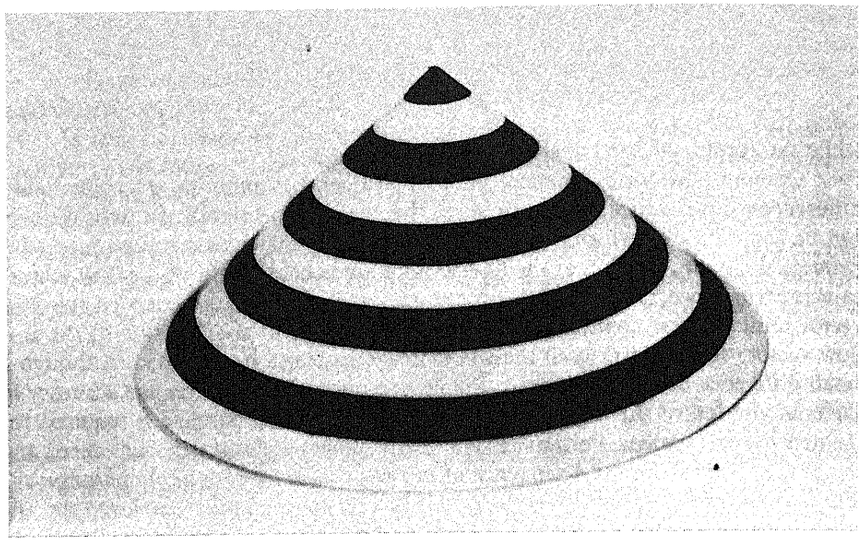


Figure 9.23 A black and white cone.

environment (see Figures 9.23, 9.24, 9.26 and 9.27). For the purposes of illustration, Figures 9.23 through 9.28 depict the results of the rotational motion only. Each of the constituent images in these image sequences were then convolved with a Laplacian of Gaussian mask (standard deviation of the Gaussian function = 4.0) and the zero-crossings contours were extracted. Since the Laplacian of Gaussian operator isolates intensity discontinuities over a wide range of edge contrasts, many of the resultant zero-crossings do not correspond to perceptually significant physical edges. As before, an adaptive thresholding technique was employed to identify these contours and to exclude them from further processing.

The zero-crossings contour images and their associated convolution images were then used to generate six time derivatives; since the time derivative utilizes a five-point operator combining the temporal difference with temporal averaging, the time derivative can only be estimated for images 3, 4, 5, 6, and 7; the associated orthogonal component of velocity was then computed, followed by the true optical flow vectors. An extended flow field was then estimated by tracking the flow vectors from image 3 through images 4, 5 to image 6 on a contour-to-contour basis, i.e. tracking a total of three images (see Figures 9.25 and 9.28). Depth images (representing the distance from the camera to each point on the zero-crossing contour) were generated for each scene (Figures 9.25 and 9.28) from the tracked velocity vectors. Finally, a range image representing the range of all visible points on the surface was generated by interpolation (again, Figures 9.25 and 9.28).

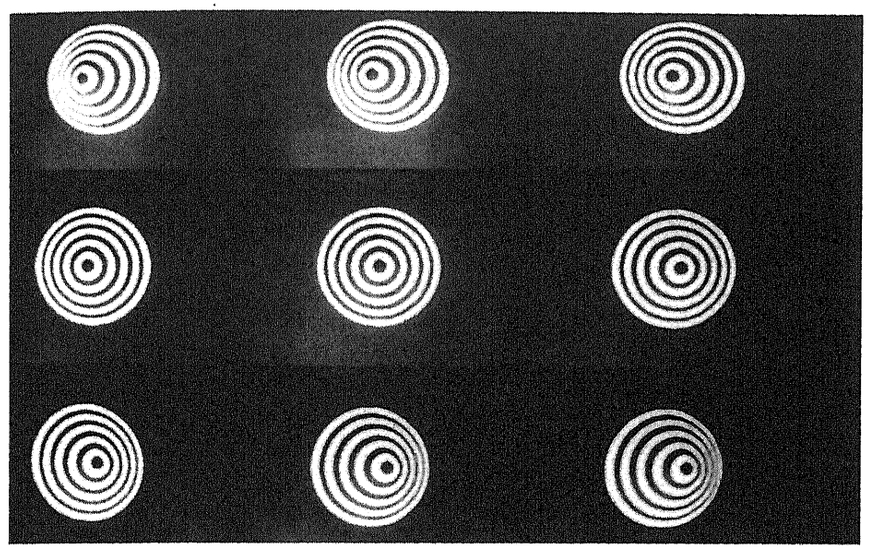


Figure 9.24 The nine views of the black and white cone.

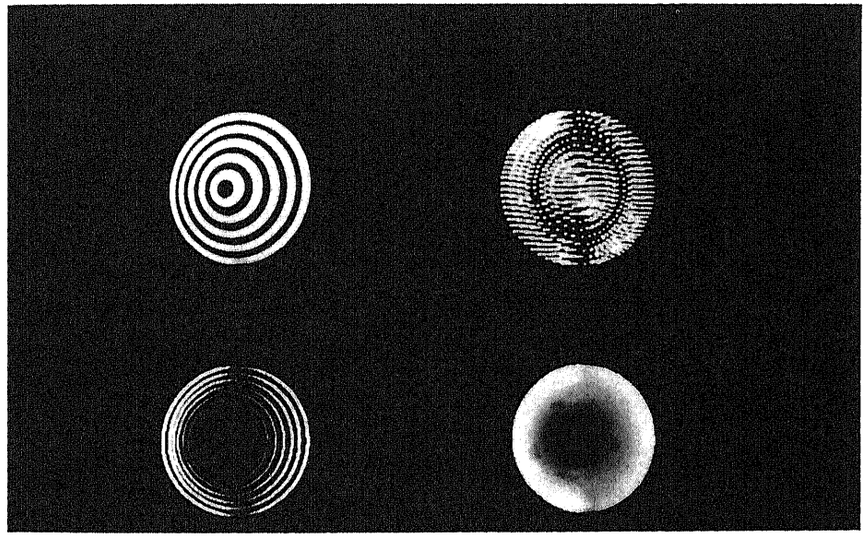


Figure 9.25 Top left: the black and white cone. Top right: the optical flow vectors. Bottom left: zero-crossings with intensity proportional to distance from camera. Bottom right: range image with intensity proportional to distance from camera.

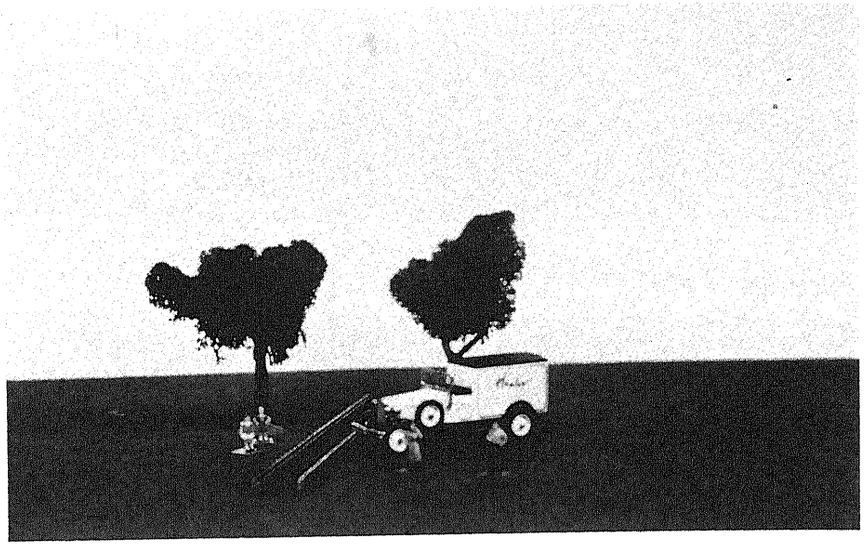


Figure 9.26 A toy-town scene.

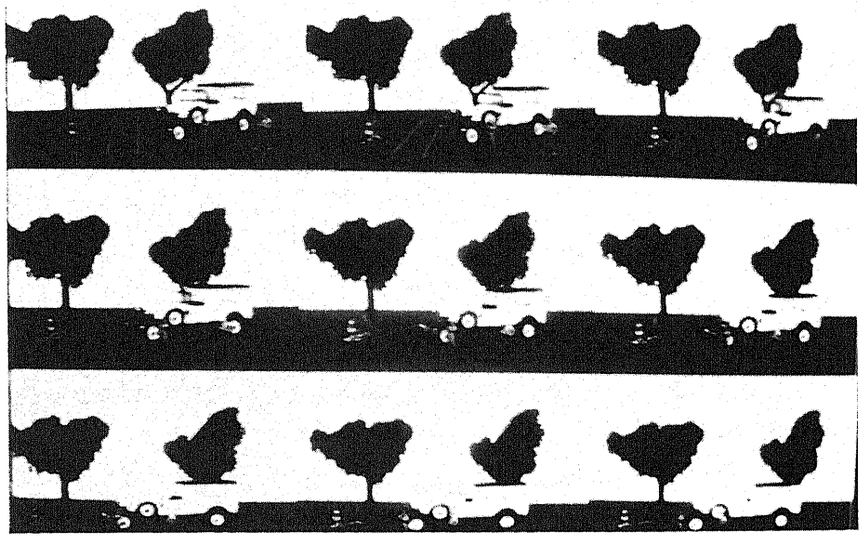


Figure 9.27 The nine views of the toy-town scene.

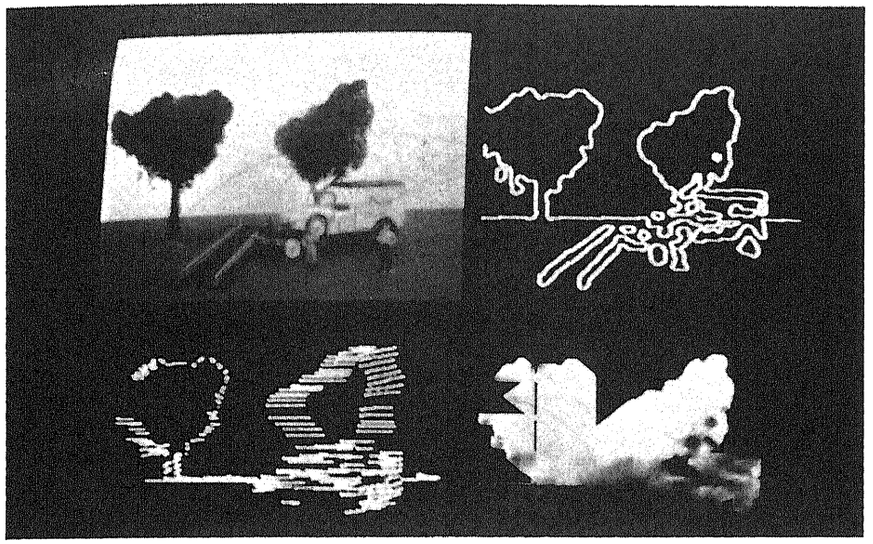


Figure 9.28 Top left: the toy-town scene. Top right: the optical flow vectors. Bottom left: zero-crossings with intensity proportional to distance from camera. Bottom right: range image with intensity proportional to distance from camera.

9.4.3 Shading

The construction of the two-and-a-half-dimensional sketch requires one further element: the computation of the local orientation of a point, i.e. the surface normal vector. The analysis of the shading of a surface, based on assumed models of the reflectivity of the surface material, is sometimes used to compute this information.

The amount of light reflected from an object depends on the following (referring to Figure 9.29):

(a) the surface material;
(b) the emergent angle, e between the surface normal and the viewer angle;
(c) the incident angle, i, between the surface normal and light source direction.

There are several models of surface reflectance, the simplest of which is the Lambertian model. A Lambertian surface is a surface that looks equally bright from all viewpoints, i.e. the brightness of a particular point does not change as the viewpoint changes. It is a perfect diffuser: the observed brightness depends only on the direction to the light source, i.e. the incident angle i.

Let E be the observed brightness, then for a Lambertian surface:

$$E = \rho \cos i$$

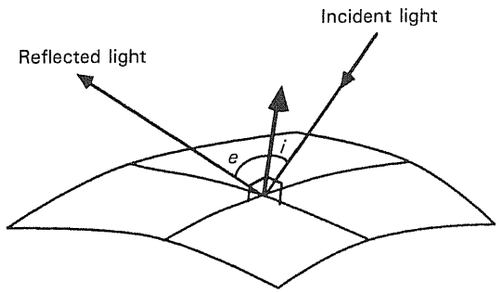


Figure 9.29 Incident and emergent angles.

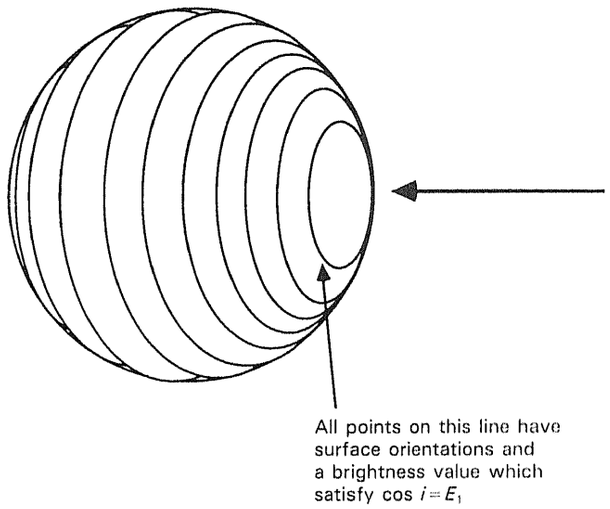


Figure 9.30 Lambertian sphere and iso-brightness lines.

where ρ is a constant called the 'surface albedo' and is peculiar to the surface material under analysis.

The reflectance properties of Lambertian surfaces are encapsulated by images of Lambertian spheres of radius 1 upon which we draw (for convenience) lines joining points of equal brightness, i.e. iso-brightness lines. There will, in general, be many points on the surface of the sphere which satisfy $E = \rho \cos i$ for a given brightness E_1; see Figure 9.30. More often, a projection of these iso-brightness values onto a plane is used. This is called a *reflectance map*. The most commonly used projection is one onto a plane which is parallel to the viewer's image plane and is a tangent to the sphere; see Figure 9.31. This is effected by drawing a line from the point on the sphere opposite the viewer, through the point to be projected.

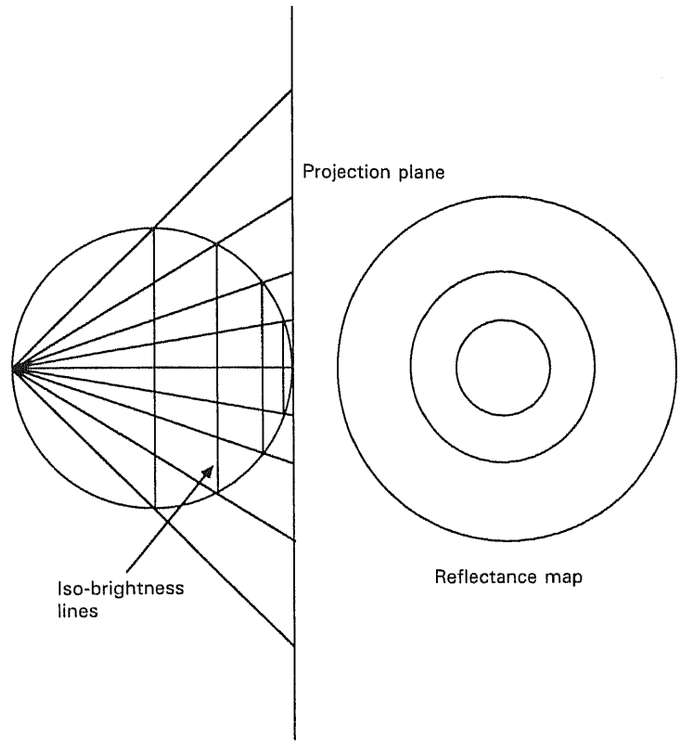


Figure 9.31 Generation of the reflectance map.

A brightness value determines only a curve on the reflectance map rather than a single point, since there will be several points on the sphere having equal brightness, i.e. the points on an iso-brightness line. Thus, an extra constraint is required to determine uniquely the orientation of the imaged point from the brightness: this is supplied by assumptions of surface smoothness (or continuity) that *the surface should not vary much from the surface direction at neighbouring parts*. Obviously, we need some coordinates on the reflectance map to anchor the process and from which we can infer the orientation of neighbouring points.

On occluding boundaries of objects without sharp edges, the surface direction is perpendicular to the viewer's line of sight. All such directions project onto a circle of radius 2 on the reflectance map and, thus, we know immediately the surface orientation of every occluding contour point and, more importantly, the correspondence between a given occluding boundary point and the point to which it maps on the reflectance function. This is achieved by observing that the required point of the reflectance map must correspond to the surface normal direction at the occluding boundary. Since the surface orientation changes smoothly, all points on

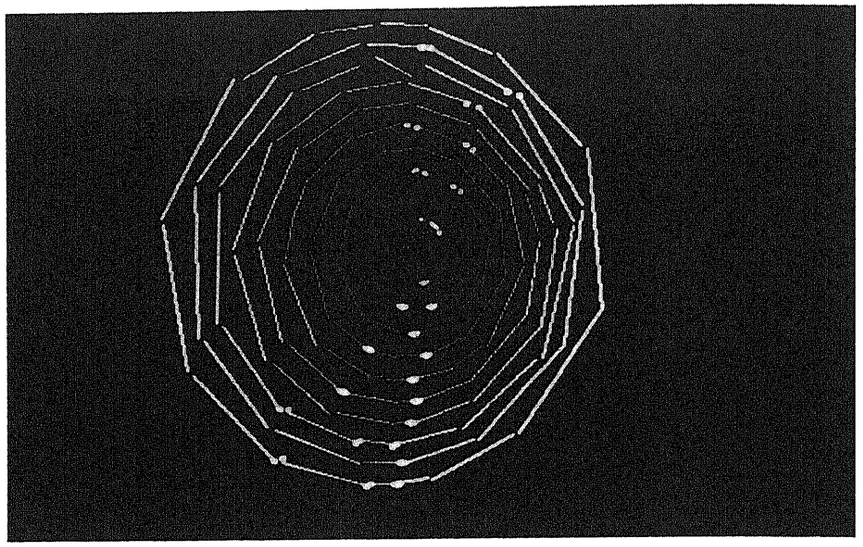


Figure 9.32 Three-dimensional raw primal sketch of a striped cone.

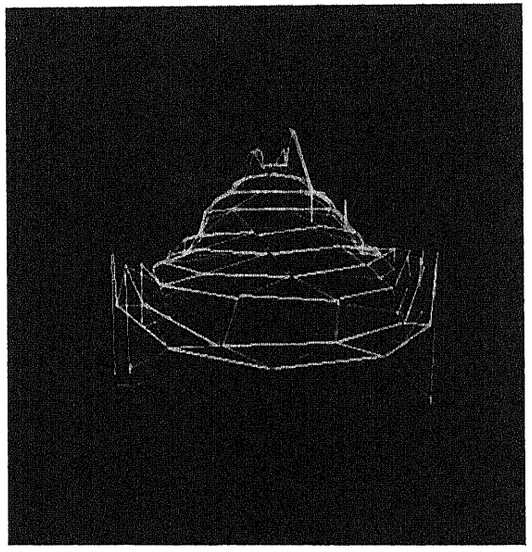


Figure 9.33 Reconstructed surface model of the striped cone.

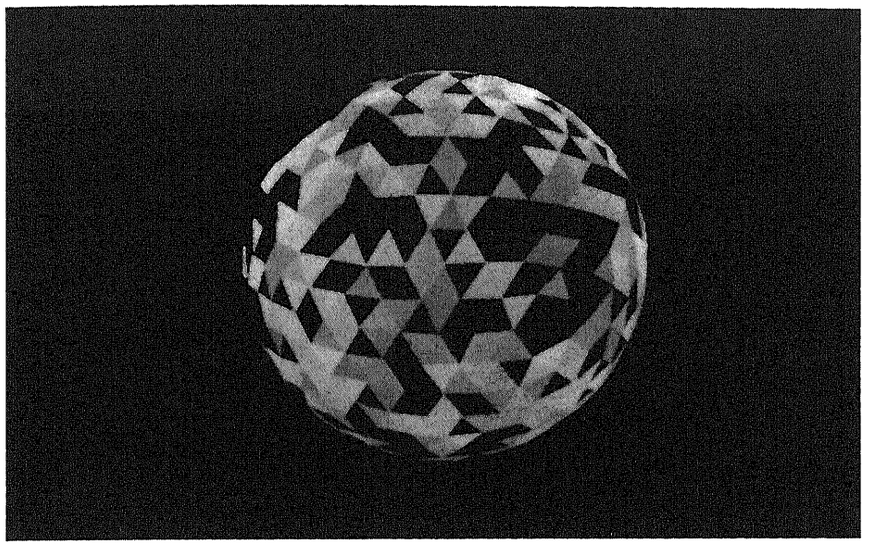


Figure 9.34 Extended Gaussian image depicting the distribution of surface normals on the polyhedral model of the cone.

the surface close to the occluding boundary must have an orientation which is not significantly different from that of the occluding boundary. The surface orientation of each point adjacent to the occluding boundary can now be computed by measuring the intensity value and reading off the corresponding orientation from the reflectance map *in a local area surrounding the point on the map which corresponds to the current occluding boundary anchor point*. This scheme of local constraint is reiterated using these newly computed orientations as constraints, until the orientation of all points on the surface have been computed.

This technique has been studied in depth in the computer vision literature and it should be emphasized that this description is intuitive and tutorial in nature; you are referred to the appropriate texts cited in the bibliography at the end of the chapter. As we have noted, however, there are a number of assumptions which must be made in order for the technique to work successfully, e.g. the surface orientation must vary smoothly and, in particular, it must do so at the occluding boundary (the boundary of the object at which the surface disappears from sight). Look around the room you are in at present. How many objects do you see which fulfil this requirement? Probably very few. Allied to this are the requirements that the reflective surface has a known albedo and that we can model its reflective properties, or, alternatively, that we can calibrate for a given reflective material, and, finally, that one knows the incident angle of light. This limits the usefulness of the techniques for *general* image understanding.

There are other ways of estimating the local surface orientation. As an example of one *coarse* approach, consider the situation where we have a three-dimensional raw primal sketch, i.e. a raw primal sketch in which we know the depth to each point on the edge segments, and if these raw primal sketch segments are *sufficiently close*, we can compute the surface normal by interpolating between the edges, generating a succession of planar patches, and effectively constructing a polyhedral model of the object (see Section 9.3.4.3). The surface normal is easily computed by forming the vector cross-product of two vectors in the plane of the patch (typically two non-parallel patch sides). For example, the three-dimensional raw primal sketch of the calibration cone which is shown in Figure 9.32 yields the polyhedral model shown in Figure 9.33, the extended Gaussian image of which is shown in Figure 9.34.

9.5 Concluding remarks

Having read this book, and this chapter in particular, you could be excused for thinking that computer vision is an end in itself, that is, that the task is complete once we arrive at our unambiguous explicit three-dimensional representation of the world. This is quite wrong. Vision is no such thing; it is merely part of a larger system which might best be characterized by a dual two-faced process of *making sense of/interacting with* the environment. Without action, perception is futile; without perception, action is futile. Both are complementary, but highly related, activities. Any intelligent action in which the system engages in the environment, i.e. anything it does, it does with an understanding of its action, and quite often it gains this by on-going visual perception.

In essence, image understanding is as concerned with cause and effect, with purpose, with action and reaction as it is with structural organization. That we have not advanced greatly in this aspect of image understanding and computer vision yet is not an indictment of the research community; in fact, given the disastrous consequences of the excessive zeal and ambition in the late 1970s, it is perhaps no bad thing that attention is currently focused on the formal and well-founded bases of visual processes: without these, the edifice we construct in image understanding would be shaky, to say the least. However, the issues we have just raised, in effect the temporal semantics of vision in contribution to and in participation with physical interactive systems, will not go away and must be addressed and understood someday. Soon.

Exercises

1. What do you understand by the term 'subjective contour'? In the context of the full primal sketch, explain how such phenomena arise

and suggest a technique to detect the occurrence of these contours. Are there any limitations to your suggestion? If so, identify them and offer plausible solutions.

2. Given that one can establish the correspondence of identical points in two or more images of the same scene, where each image is generated at a slightly different viewpoint, explain how one can recover the absolute real-world coordinates of objects, or points on objects, with suitably calibrated cameras. How can one effectively exploit the use of more than two such stereo images? How would you suggest organizing the cameras for this type of multiple camera stereo in order to minimize ambiguities?

3. Describe, in detail, one approach to the construction of the two-and-a-half-dimensional sketch and identify any assumptions exploited by the component processes.

4. Is the two-and-a-half-dimensional sketch a useful representation in its own right or is it merely an intermediate representation used in the construction of higher-level object descriptions?

5. 'The sole objective of image understanding systems is to derive unambiguous, four-dimensional (spatio-temporal) representations of the visual environment and this can be accomplished by the judicious use of early and late visual processing.' Evaluate this statement critically.

6. 'Image understanding systems are *not* intelligent; they are *not* capable of perception, and, in effect, they do *not* understand their environment.' Discuss the validity of this statement.

7. Do exercise 1 in Chapter 1.

References and further reading

Ahuja, N., Bridwell, N., Nash, C. and Huang, T.S. 1982 *Three-Dimensional Robot Vision*, Conference record of the 1982 workshop on industrial application of machine vision, Research Triangle Park, NC, USA, pp. 206–13.

Arun, K.S., Huang, T.S. and Blostein, S.D. 1987 'Least-squares fitting of two 3-D point sets', *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. PAMI-9, No. 5, pp. 698–700.

Bamieh, B. and De Figueiredo, R.J.P. 1986 'A general moment-invariants/attributed-graph method for three-dimensional object recognition from a single image', *IEEE Journal of Robotics and Automation*, Vol. RA-2, No. 1, pp. 31–41.

Barnard, S.T. and Fischler, M.A. 1982 *Computational Stereo*, SRI International, Technical Note No. 261.

Ben Rhouma, K., Peralta, L. and Osorio, A. 1983 'A "K2D" perception approach for

assembly robots', *Signal Processing II: Theory and Application*, Schurrler, H.W. (ed.), Elsevier Science Publishers B.V. (North-Holland), pp. 629–32.

Besl, P.J. and Jain, R. 1985 'Three-dimensional object recognition', *ACM Computing Surveys*, Vol. 17, No. 1, pp. 75–145.

Bhanu, B. 1984 'Representation and shape matching of 3-D objects', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-6, No. 3, pp. 340–51.

Brady, M. 1982 'Computational approaches to image understanding', *ACM Computing Surveys*, Vol. 14, No. 1, pp. 3–71.

Brooks, R.A. 1981 'Symbolic reasoning among 3-D models and 2-D images', *Artificial Intelligence*, Vol. 17, pp. 285–348.

Brooks, R.A. 1983 'Model-based three-dimensional interpretations of two-dimensional images', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No. 2, pp. 140–50.

Dawson, K. and Vernon, D. 1990 'Implicit model matching as an approach to three-dimensional object recognition', *Proceedings of the ESPRIT Basic Research Action Workshop on 'Advanced Matching in Vision and Artificial Intelligence'*, Munich, June 1990.

Fang, J.Q. and Huang, T.S. 1984 'Some experiments on estimating the 3-D motion parameters of a rigid body from two consecutive image frames', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-6, No. 5, pp. 545–54.

Fang, J.Q. and Huang, T.S. 1984 'Solving three-dimensional small rotational motion equations: uniqueness, algorithms and numerical results', *Computer Vision, Graphics and Image Processing*, No. 26, pp. 183–206.

Fischler, M.A. and Bolles, R.C. 1986 'Perceptual organisation and curve partitioning', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 1, pp. 100–5.

Frigato, C., Grosso, E., Sandini, G., Tistarelli, M. and Vernon, D. 1988 'Integration of motion and stereo', *Proceedings of the 5th Annual ESPRIT Conference, Brussels*, edited by the Commission of the European Communities, Directorate-General Telecommunications, Information Industries and Innovation, North-Holland, Amsterdam, pp. 616–27.

Guzman, A. 1968 'Computer Recognition of Three-Dimensional Objects in a Visual Scene', Ph.D. Thesis, MIT, Massachusetts.

Haralick, R.M., Watson, L.T. and Laffey, T.J. 1983 'The topographic primal sketch', *The International Journal of Robotics Research*, Vol. 2, No. 1, pp. 50–72.

Hall, E.L. and McPherson, C.A. 1983 'Three dimensional perception for robot vision', *Proceedings of SPIE*, Vol. 442, pp. 117–42.

Healy, P. and Vernon, D. 1988 'Very coarse granularity parallelism: implementing 3-D vision with transputers', *Proceedings Image Processing '88*, Blenheim Online Ltd, London, pp. 229–45.

Henderson, T.C. 1983 'Efficient 3-D object representations for industrial vision systems', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No. 6, pp. 609–18.

Hildreth, E.C. 1983 *The Measurement of Visual Motion*, MIT Press, Cambridge, USA.

Horaud, P., and Bolles, R.C. 1984 '3DPO's strategy for matching 3-D objects in range data', *International Conference on Robotics, Atlanta, GA, USA*, pp. 78–85.

Horn, B.K.P. and Schunck, B.G. 1981 'Determining optical flow', *Artificial Intelligence*, **17**, Nos 1–3 pp. 185–204.

Horn, B.K.P. and Ikeuchi, K. 1983 *Picking Parts out of a Bin*, AI Memo No. 746, MIT AI Lab.

Huang, T.S. and Fang, J.Q. 1983 'Estimating 3-D motion parameters: some experimental results', *Proceedings of SPIE*, Vol. 449, Part 2, pp. 435–7.

Ikeuchi, K. 1983 *Determining Attitude of Object From Needle Map Using Extended Gaussian Image*, MIT AI Memo No. 714.

Ikeuchi, K., Nishihara, H.K., Horn, B.K., Sobalvarro, P. and Nagata, S. 1986 'Determining grasp configurations using photometric stereo and the PRISM binocular stereo system', *The International Journal of Robotics Research*, Vol. 5, No. 1, pp. 46–65.

Jain, R.C. 1984 'Segmentation of frame sequences obtained by a moving observer', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-6, No. 5, pp. 624–9.

Kanade, T. 1981 'Recovery of the three-dimensional shape of an object from a single view', *Artificial Intelligence*, Vol. 17, pp. 409–60.

Kanade, T. 1983 'Geometrical aspects of interpreting images as a 3-D scene', *Proceedings of the IEEE*, Vol. 71, No. 7, pp. 789–802.

Kashyap, R.L. and Oomen, B.J. 1983 'Scale preserving smoothing of polygons', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No. 6, pp. 667–71.

Kim, Y.C. and Aggarwal, J.K. 1987 'Positioning three-dimensional objects using stereo images', *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 4, pp. 361–73.

Kuan, D.T. 1983 'Three-dimensional vision system for object recognition', *Proceedings of SPIE*, Vol. 449, pp. 366–72.

Lawton, D.T. 1983 'Processing translational motion sequences', *CVGIP*, **22**, pp. 116–44.

Lowe, D.G. and Binford, T.O. 1985 'The recovery of three-dimensional structure from image curves', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7, No. 3, pp. 320–6.

Marr, D. 1976 'Early processing of visual information', *Philosophical Transactions of the Royal Society of London*, **B275**, pp. 483–524.

Marr, D. and Poggio, T. 1979 'A computational theory of human stereo vision', *Proceedings of the Royal Society of London*, **B204**, pp. 301–28.

Marr, D. 1982 *Vision*, W.H. Freeman and Co., San Francisco.

Martin, W.N. and Aggarwal, J.K. 1983 'Volumetric descriptions of objects from multiple views', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No. 2, pp. 150–8.

McFarland, W.D. and McLaren, R.W. 1983 'Problem in three dimensional imaging', *Proceedings of SPIE*, Vol. 449, pp. 148–57.

McPherson, C.A., Tio, J.B.K., Sadjadi, F.A. and Hall, E.L. 1982 'Curved surface representation for image recognition', *Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, Las Vegas, NV, USA*, pp. 363–9.

McPherson, C.A. 1983 'Three-dimensional robot vision', *Proceedings of SPIE*, Vol. 449, part 4, pp. 116–26.

Nishihara, H.K. 1983 'PRISM: a practical realtime imaging stereo matcher', *Proceedings of SPIE*, Vol. 449, pp. 134–42.

Pentland, A. 1982 The Visual Inference of Shape: Computation from Local Features, Ph.D. Thesis, Massachusetts Institute of Technology.

Poggio, T. 1981 *Marr's Approach to Vision*, MIT AI Lab., AI Memo No. 645.

Pradzy, K. 1980 'Egomotion and relative depth map from optical flow', *Biol. Cybernetics*, **36**, pp. 87–102.

Ray, R., Birk, J. and Kelley, R.B. 1983 'Error analysis of surface normals determined by radiometry', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No. 6, pp. 631–71.

Roberts, L.G. 1965 'Machine perception of three-dimensional solids' in *Optical and Electro-Optical Information Processing*, J.T. Tippett *et al.* (eds), MIT Press, Cambridge, Massachusetts, pp. 159–97.

Safranek, R.J. and Kak, A.C. 1983 'Stereoscopic depth perception for robot vision: algorithms and architectures', *Proceedings of IEEE International Conference on Computer Design: VLSI in Computers (ICCD '83), Port Chester, NY, USA*, pp. 76–9.

Sandini, G. and Tistarelli, M. 1985 'Analysis of image sequences', *Proceedings of the IFAC Symposium on Robot Control.*

Sandini, G. and Tistarelli, M. 1986 *Recovery of Depth Information: Camera Motion Integration Stereo*, Internal Report, DIST, University of Genoa, Italy.

Sandini, G. and Tistarelli, M. 1986 'Analysis of camera motion through image sequences', in *Advances in Image Processing and Pattern Recognition*, V. Cappellini and R. Marconi (eds), Elsevier Science Publishers B.V. (North-Holland), pp. 100–6.

Sandini, G. and Vernon, D. 1987 'Tools for integration of perceptual data', in *ESPRIT '86: Results and Achievements*, Directorate General XIII (eds), Elsevier Science Publishers B.V. (North-Holland), pp. 855–65.

Sandini, G., Tistarelli, M. and Vernon, D. 1988 'A pyramid based environment for the development of computer vision applications', *IEEE International Workshop on Intelligent Robots and Systems, Tokyo.*

Sandini, G. and Tistarelli, M. 1990 'Active tracking strategy for monocular depth inference from multiple frames', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 1, pp. 13–27.

Schenker, P.S. 1981 'Towards the robot eye: isomorphic representation for machine vision', *SPIE*, Vol. 283, '3-D machine reception', pp. 30–47.

Shafer, S.A. 1984 *Optical Phenomena In Computer Vision*, Technical Report TR 135, Computer Science Department, University of Rochester, Rochester, NY, USA.

Vernon, D. and Tistarelli, M. 1987 'Range estimation of parts in bins using camera motion', *Proceedings of SPIE's 31st Annual International Symposium on Optical and Optoelectronic Applied Science and Engineering, San Diego, California, USA*, 9 pages.

Vernon, D. 1988 *Isolation of Perceptually-Relevant Zero-Crossing Contours in the Laplacian of Gaussian-filtered Images*, Department of Computer Science, Trinity College, Technical Report No. CSC-88-03 (17 pages).

Vernon, D. and Sandini, G. 1988 'VIS: A virtual image system for image understanding', *Software Practice and Experience*, Vol. 18, No. 5, pp. 395–414.

Vernon, D. and Tistarelli, M. 1991 'Using camera motion to estimate range for robotic parts manipulation', accepted for publication in the *IEEE Transactions on Robotics and Automation.*

Wertheimer, M. 1958 'Principles of perceptual organisation', in D.C. Beardslee and M. Wertheimer (eds), *Readings in Perception*, Princeton, Van Nostrand.

Wu, C.K., Wang, D.Q. and Bajcsy, R.K. 1984 'Acquiring 3-D spatial data of a real object', *Computer Vision, Graphics, and Image Processing*, Vol. 28, pp. 126–33.

Appendix: Separability of the Laplacian of Gaussian operator

The *Laplacian of Gaussian* operator is defined:

$$\nabla^2\{I(x, y) * G(x, y)\} = \nabla^2 G(x, y) * I(x, y)$$

where $I(x, y)$ is an image function and $G(x, y)$ is the two-dimensional Gaussian function defined as follows:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp[-(x^2 + y^2)/2\sigma^2]$$

The Laplacian is the sum of the second-order unmixed partial derivatives:

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

This two-dimensional convolution is separable into four one-dimensional convolutions:

$$\nabla^2\{I(x, y) * G(x, y)\} = G(x) * \left\{I(x, y) * \frac{\partial^2}{\partial y^2} G(y)\right\} + G(y) * \left\{I(x, y) * \frac{\partial^2}{\partial x^2} G(x)\right\}$$

This can be shown as follows:

$$\nabla^2\{I(x, y) * G(x, y)\} = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)\left(I(x, y) * \frac{1}{2\pi\sigma^2} \exp[-(x^2 + y^2)/2\sigma^2]\right)$$

$$= \frac{\partial^2}{\partial x^2}\left(I(x, y) * \frac{1}{2\pi\sigma^2} \exp[-(x^2 + y^2)/2\sigma^2]\right)$$

$$+\frac{\partial^2}{\partial y^2}\left(I(x,y) * \frac{1}{2\pi\sigma^2}\exp[-(x^2+y^2)/2\sigma^2]\right)$$

$$=\frac{\partial^2}{\partial x^2}\left(I(x,y) * \frac{1}{2\pi\sigma^2}\exp(-x^2/2\sigma^2)\exp(-y^2/2\sigma^2)\right)$$

$$+\frac{\partial^2}{\partial y^2}\left(I(x,y) * \frac{1}{2\pi\sigma^2}\exp(-x^2/2\sigma^2)\exp(-y^2/2\sigma^2)\right)$$

$$=\left(\frac{1}{\sqrt{2\pi}\sigma}\exp(-y^2/2\sigma^2)\left(\frac{\partial^2}{\partial x^2}\frac{1}{\sqrt{2\pi}\sigma}\exp(-x^2/2\sigma^2)\right)\right) * I(x,y)$$

$$+\left(\frac{1}{\sqrt{2\pi}\sigma}\exp(-x^2/2\sigma^2)\left(\frac{\partial^2}{\partial y^2}\frac{1}{\sqrt{2\pi}\sigma}\exp(-y^2/2\sigma^2)\right)\right) * I(x,y)$$

$$=\left\{G(x)\frac{\partial^2}{\partial y^2}G(y)\right\} * I(x,y) + \left\{G(y)\frac{\partial^2}{\partial x^2}G(x)\right\} * I(x,y)$$

Let $(\partial^2/\partial x^2)G(x)$ be $A(x)$ and let $(\partial^2/\partial y^2)G(y)$ be $A(y)$, then we can rewrite the above as:

$$=\{G(x)\ A(y)\} * I(x,y) + \{G(y)\ A(x)\} * I(x,y)$$

Noting the definition of the convolution integral:

$$f(x,y) * h(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x-m, y-n)\ h(m,n)\,\mathrm{d}m\,\mathrm{d}n$$

we can expand the above:

$$=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(x-m)\ A(y-n)\ I(m,n)\,\mathrm{d}m\,\mathrm{d}n$$

$$+\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(y-n)\ A(x-m)\ I(m,n)\,\mathrm{d}m\,\mathrm{d}n$$

$$=\int_{-\infty}^{\infty} G(x-m)\int_{-\infty}^{\infty} A(y-n)\ I(m,n)\,\mathrm{d}n\,\mathrm{d}m$$

$$+\int_{-\infty}^{\infty} G(y-n)\int_{-\infty}^{\infty} A(x-m)\ I(m,n)\,\mathrm{d}m\,\mathrm{d}n$$

$$=G(x) * \left\{I(x,y) * \frac{\partial^2}{\partial y^2}G(y)\right\} + G(y) * \left\{I(x,y) * \frac{\partial^2}{\partial x^2}G(x)\right\}$$

Index

MACHINE VISION

Automated Visual Inspection and Robot Vision

David Vernon

Machine vision, an evolving and fascinating topic, is a multi-disciplinary subject, utilising techniques drawn from optics, electronics, mechanical engineering, computer science and artificial intelligence. This book provides an in-depth introduction to machine vision allowing the reader to quickly assimilate and comprehend all the necessary issues in industrial vision: sensors, image acquisition, processing, analysis, and integration with robot systems. Practical aspects are treated equally with theoretical issues, equipping the reader with the understanding to implement a vision system.

Special features of the book include:

- Complete, self-contained treatment of all topics essential to the implementation of industrial vision systems, from cameras to robots
- Detailed case-study (chapter 8) introducing robot manipulation
- State-of-the-art developments in 3D robot vision

The author, Dr David Vernon, a lecturer at Trinity College, Dublin, Ireland, has lectured undergraduate and postgraduate courses on computer vision since 1983. He has also designed and presented several courses on machine vision to industry and is active in several international research projects in machine vision.

ISBN 0-13-543398-3

9 780135 433980

Prentice Hall