Sādhanā, Vol. 18, Part 2, June 1993, pp. 325–336. © Printed in India.

Volume intersection for shape from silhouettes

ARUN K PUJARI

Artificial Intelligence Laboratory, School of Mathematics and Computer/ Information Sciences, University of Hyderabad, Hyderabad 500 134, India

Abstract. Volume intersection is one of the simplest paradigms for recovering shape from 2D images. The underlying principle in this method is reconstructing an object by intersecting the volumes obtained by back-projecting its silhouettes in their corresponding directions. The set of silhouettes could be predetermined or it could be decided dynamically for optimum reconstruction. The former approach is passive while the latter comes under the purview of active vision. In this paper, the author attempts to integrate his research results in obtaining a linear octree description of an object from its silhouettes making use of a data structure called *range*$[i,j,0/1]$. The problem of shape from silhouette is formalized and it is shown here that the conventional approach of volume intersection for this problem need not be always efficient. The advantage of active vision technique is also discussed in the present context.

Keywords. Volume intersection; shape recovery; silhouettes.

1. Introduction

Shape recovery is one of the prime areas of interest in computer vision at present, finding applications in CAD/CAM, recognition of a robot's workspace etc. Several cues like texture, shading, stereo, multiple images etc. aid in the process of recovering shape. Volume intersection is a technique of reconstruction of an otherwise unknown object/scene from its multiple 2D images. The 2D images are obtained by projecting the 3D object/scene on to the 2D plane. A tentative reconstruction is obtained by back-projecting the images in their viewing directions. As a result, the projection (in obtaining the input images) and the direction (of projection) have a direct bearing on the reconstruction algorithms. Most researchers (Chien & Aggarwal 1986; Kim & Aggarwal 1986; Ahuja & Veenstra 1989) have earlier concentrated on orthographic projection of silhouettes for recovery of 3D shape as it turns out to be algorithmically simple. But all these approaches cannot be extended to accommodate perspective images. Moreover, any approach of shape recovery from silhouettes is heavily dependent on the set of viewing directions (i.e. the images). If the set is determined *a priori*, then the reconstruction, though acceptable, may not be optimum or satisfactory. On the other hand, if the set of directions is determined dynamically, an optimum reconstruction could be achieved with a minimum number of images. This active vision approach is weighted down by its complexity. Very few attempts are versatile enough to handle a static or dynamic set of orthographic/perspective images in reconstruction.

In this paper, we review a structure *range*$[i,j,0/1]$, first introduced by Lavakusha *et al* (1989) and explain how this data structure is capable of handling perspective as well as orthographic projections; active and passive vision. The results reported here are essentially the integration of several results highlighting the capability of range[] structure for volume intersection methods.

2. Volume intersection

Volume intersection aims at generating a volumetric description of a 3D object/scene from its multiple binary 2D images called silhouettes. Each silhouette is back-projected along its viewing direction and the volumes so obtained are intersected to obtain an approximation to the object. A cube of size N, called a *universe cube*, is assumed to contain the object under consideration. The object can be unambiguously described by a 3D array of $N \times N \times N$ voxels. Deciding if the individual voxels are white (0) or black (1) is the ultimate of object reconstruction. For compactness, a set of contiguous voxels may also be considered as a volume element. The set of silhouettes, I_k, $1 < k \leqslant p$ is obtained by projecting the object on to the 2D plane in direction d_k, $1 < k \leqslant p$. The (i,j)th element of I_k is denoted by x_{ij}. Let v be a volume element and *project*(v,k) be defined as the set of pixels in I_k such that the pixel x_{ij} when extended along d_k intersects v. Clearly, project(v,k) gives the occluding contour of the object as seen from d_k. The inverse mapping, sweep(x,k), is the set of volume elements v' such that project(v',k) contains x.

The volume element v is considered black (or, 1) if all x_{ij}'s in project(v,k) are black for each k and it is labelled white(or, 0) if there exists at least one d_k such that project(v,k) has all pixels as 0. If none of these two cases holds and if v is not a single voxel, it is split into smaller blocks. If it is a single voxel, it is deemed white (figure 1).

There are fundamentally two approaches for labelling the volume elements v in sweep(x,k). A volume element could be labelled by considering all the I_k's before another volume element is taken up, or all the volume elements could be labelled with reference to one I_k and then with another I_k and so on. In the former approach, the intersection of the volumes is performed simultaneously while in the latter it is done incrementally. The process of labelling depends on the set of d_k's considered and the method of projection involved in getting the I_k's. Further, the 2D and 3D representation schemes for the input images and the resulting objects have to be decided upon. All the results published so far have made use of various combinations of these four parameters (Martin & Aggarwal 1983; Srivastava & Ahuja 1987; Nobborio *et al* 1988).

2.1. *Choice of viewing directions*

The number and orientation of the images play a role in the accuracy of reconstruction. This is because only those characteristics of the object that manifest themselves in the images in these directions can be reconstructed. One could imagine that the bigger the set of images the better the approximation would be. But this is not necessarily true. Moreover, a big set would increase the complexity of reconstruction. The set

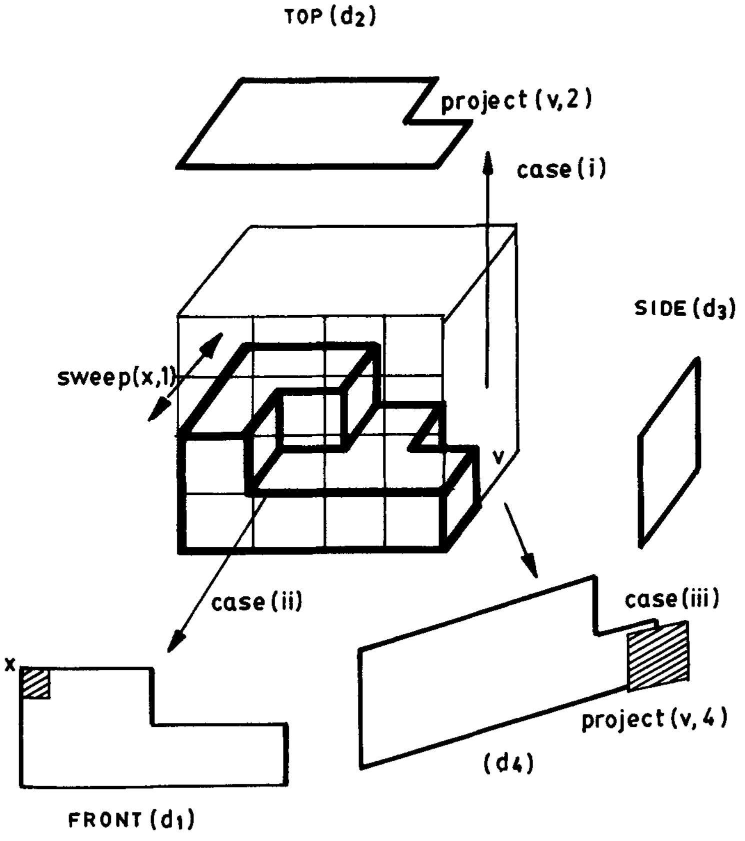

Figure 1. The volume intersection paradigm.

of viewing directions should be so chosen as to bring down the complexity while giving a satisfactory approximation. A prespecified set comes under the purview of passive vision whereas a dynamic choice of d_k to incrementally carry out the intersection for approximating the object is studied in active vision. In the following subsections, a brief discussion on these two approaches relevant to volume intersection is given.

2.1a *Passive vision*: Passive vision is concerned with the shape recovery process which is independent of the shape sensing process. Thus, in this context, it is to fix the d_k's, $1 < k \leq p$ and p before activating the volume intersection algorithm. Since the object is enclosed in a universe cube, generally the set of directions that are chosen *a priori* are the set of three mutually perpendicular directions along the front, side and top of the cube. The pixel to voxel mapping is very simple in such cases. Hence this set of directions is generally taken to test any new algorithm.

The main disadvantage in deciding the set in advance is that the set becomes independent of the object. Hence the reconstruction may not be optimum. Moreover, the same object may be reconstructed differently if another set of viewing directions is considered. But passive vision is still relevant if the object and the sensor are stationary.

2.1b *Active vision*: Interest in this area with regard to volume intersection is still very recent. In this paradigm, the object reconstruction process controls the image acquisition process. This is relevant only if either the object or the camera can be moved about. The reconstruction process attempts to determine incrementally the next viewing direction based on the geometric properties of the object that have been acquired so far. This process, though not simple, would yield a better approximation as it is target-specific. The volume intersection methods proposed earlier cannot be extended to accommodate active vision as they all carry the intersections simultaneously. But range$[i, j, 0/1]$, with its inherent scope for incremental intersection is very suitable for active vision.

2.2 *Type of projection: Orthographic/perspective*

The projection involved in obtaining the 2D silhouette is crucial as volume intersection is carried out by intersecting the volumes obtained by back-projecting the silhouettes. The simplest way is to consider a set of parallel projectors, giving rise to orthographic silhouettes. If the directions of projection are parallel to the principal axes of the universe cube, the front, the side and the top views of the object can be obtained. For such a case, identification of *sweep*(x, k) is just pixel to voxel mapping which is straightforward.

But orthographic images are not natural in the sense that the eye or the camera can form only perspective images. Orthographic images can be obtained only by keeping the object size small and the distance between the object and the camera large.

For *perspective images*, the projectors emanate from the viewpoint. Hence the camera position is important. Sweeping a perspective silhouette gives a cone with the apex at the viewpoint and the silhouette as its cross-section. This gives the impression of the silhouette growing in size as it is back-projected. Here *sweep*(x, k) would not be parallel to the axes of the cube. When perspective silhouettes are used, 'natural' images can be used but with an overhead of complexity.

3. Proposed approach

Incremental intersection of back-projected volumes highlights this approach. A minimum of three views is necessary. One of the views I_r is used as reference. Two other views are back-projected in terms of their rows or columns. The common area is represented by *range*$[i, j, 0/1]$ with reference to every x_{ij} of I_r. The *range*$[i, j, 0/1]$ gives the tentative reconstruction. It is updated by considering other views and a volumetric description of the object obtained. Assuming that the set of viewing directions is specified *a priori*, volume intersection is carried out in three clear steps –

- computation of range[],
- updating of range[],
- generation of a linear octree representation.

But if the set is not specified, an additional step of finding the next viewing direction has to be incorporated. This structure was first proposed in Lavakusha *et al* (1989a). They used this structure to reconstruct 3D objects from three orthographic silhouettes. Pai *et al* (1990) improved upon this by making use of an intermediate data structure, *an inverted segment tree* to store range$[i, j, 0/1]$. They obtained a linear octree description of the object. Nitya *et al* (1992) applied this concept to perspective silhouettes and reconstructed 3D objects successfully. Though these approaches made use of front, side and top views of the object, they can be easily extended to handle other views too. In the following subsections, we explain the concept of range$[i,j,0/1]$ and integrate the research done in passive and active vision using orthographic and perspective silhouettes. In Pai *et al* (1990), obtaining a linear octree description from a segment tree is explained.

Coordinate system: The top left-hand corner of the universe cube is taken as the origin. An octant of the cube is defined by the top-left-hand corner coordinate. For the silhouette, the top-left-hand corner coordinate is the origin and the bottom-right-hand corner coordinate is $(N-1, N-1)$.

3.1 *Range*$[i,j,0/1]$

As mentioned earlier, *project*(v,k) is the set of black pixels in I_k giving the occluding contour of the object as seen from a particular point or direction d_k. A pixel x in *project*(v,k) corresponds to one or more black voxels in the universe cube. A voxel is considered black if it is in the *sweep*(x,k) for all k. Starting with views I_1 and I_2 the intersection of *sweep*$(x,1)$ and *sweep*$(x,2)$ would give a tentative reconstruction. All the voxels in this intersection are candidates to be labelled black finally. Such candidate voxels form the range for the corresponding pixel in the reference. The correspondence is the pixel-voxel mapping perpendicular to the image and parallel to one of the axes of the universe cube. Hence, every pixel in I_r is associated with range given by *range*$[i,j,0/1]$,

range$[i,j,0]$ – for pixel(i,j) of I_r, the start of range. range$[i,j,1]$ – for pixel(i,j) of I_r, the end of range.

Hereafter, we refer to a set of candidate voxels corresponding to a pixel in I_r as *range*$[\,]$ and the limits of range for pixel (i,j) of I_r as *range*$[i,j,0/1]$.

We make use of an intermediate data structure called the *segment tree* to maintain the *range*$[\,]$. The *segment tree* $T(l,r)$ is built recursively as follows: The tree consists of a root V; and if $r-l>1$, of a left subtree $T(l,(l+r)/2)$ and a right subtree $T[((l+r)/2)+1,r]$. The range$[\,]$ is stored in a segment tree $T(0,N-1)$ where N is the size of the silhouette. The *range*$[\,]$, given by $[r0,r1]$ is decomposed into a set of standard intervals, each corresponding to some node of the tree. A secondary list is appended to each node to keep track of the intervals allocated to it. If the universe cube is visualised as a set of planes parallel to I_r, then the nodes of the segment tree would correspond to the planes and the secondary lists attached to the nodes to black pixels in those planes.

Certain issues give rise to differences in the computation of *range*$[\,]$ for the two types of projection, namely perspective and orthographic. The volumes generated by back-projecting orthographic and perspective silhouettes are different. Orthographic silhouettes yield cylindrical volumes whereas perspective ones give cones.

- Every pixel in I_r is associated with range$[\,]$. Range$[\,]$ is updated by back-projecting the reference silhouette. For the orthographic, this step can be implicit by computing range for only the back pixels of I_r.
- The correspondence of a pixel in I_r and the row and column in I_1 and I_2 is not simple for the set of perspective silhouettes. This is due to the fact that sweep$(x_i,1)$ and sweep$(x_j,2)$ for row i and column j of I_1 and I_2 respectively would not necessarily intersect along the line passing through x_{ij} in I_r and normal to it.

3.2 *Three orthographic views*

3.2a *Computation of range*[]: When the directions of projection are parallel to the principal axes of the cube, the front, side and top views of the object can be obtained. Hereafter, these silhouettes will be referred to as FRONT, SIDE and TOP. The union of *sweep*(x_{ij}, k), for all i,j, gives the swept volume of a silhouette. In the case of orthographic silhouette, the swept volume would be a cylinder with the silhouette as its cross-section. This implies that

$$sweep(x_{ip}, SIDE) \cap sweep(x_{kj}, TOP)$$

would be either null or voxel(i,j,k) only. This voxel would definitely belong to *range*$[i,j,0/1]$. Hence, for a particular row i of a silhouette, the swept volume of the row i, henceforth referred to as sweep(x_i, k) is the union of the *sweep*(x,k) of the individual black pixels in that row. The intersection of such unions (rows) of one silhouette with other such unions (columns) of another silhouette would give the range for the pixels in the reference. If FRONT is considered as reference, then for a pixel x_{ij} of FRONT, the ith row of SIDE and the jth column of TOP are to be considered to compute *range*$[i,j,0/1]$ (figure 2).

For a pixel i,j of FRONT, range[] is given by an interval $[r0, r1]$, where

$$r0 = \max(\text{smin}, t\text{min}),\ t\text{min} = N - 1 - k2,$$

$$r1 = \min(\text{smax}, t\text{max}),\ t\text{max} = N - 1 - k1.$$

Here, smin and smax give the start and end of the run of 1*s* in row i of SIDE and *t*min and *t*max give the same for the column j of TOP. In reality, range[] is not one interval $[r0, r1]$ but a set of such intervals. Range[] for the complete reference image is stored in a segment tree.

3.2b *Updating of range*[]: This step is trivial for orthographic views. The fact that the union of *sweep*(x, k) for a silhouette would be a cylinder can be used to implicitly incorporate the intersection in range by limiting the computation of range[] of only the black pixels in I_r. Range would now contain the intersection of the swept volumes

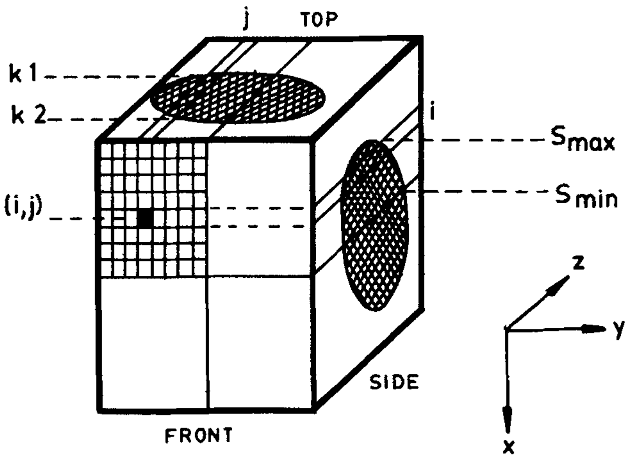

Figure 2. Range[] for orthographic silhouettes.

of the three silhouettes, I_r, I_1, I_2. The detailed algorithm is discussed in Shanmukh & Pujari (1991). From the range, a linear octree representation of the object can be generated.

3.3 *Three perspective views*

3.3a *Computation of range*[]: As the swept volumes would be expanding cones with the apex at the viewpoint and the silhouette as the cross-section, the camera position is important here. The front, side and top views of the object are obtained by placing the camera appropriately. Let the three viewpoints be given by (fx, fy, fz), (sx, sy, sz) and (tx, ty, tz). For their experiments, Nitya *et al* (1992) discuss the camera location on the lines parallel to the axes of the cube and passing through its centre. Unlike the orthographic case, the *sweep*(x_{ij}, k) is not parallel to the axis of the cube but lies along the projector through x_{ij}. If [$s$min, $s$max] is the run of 1's for a row i in I_k, then the sweep(x_i, k) is given by [$s$min*, $s$max*], where $s$min* and $s$max* lie on the projectors through $s$min and $s$max, respectively. Moreover, the non-null intersection of sweep$(x_i, 1)$ and sweep$(x_j, 2)$ is not limited to one voxel only. This implies that a voxel v in sweep$(x_{ik}, 1) \cap$ sweep$(x_{kj}, 2)$ could be in the range of a pixel in I_r other than x_{ij}. Hence computation of *range*[] for x_{ij} in I_r involves

- identifying the appropriate row i^* and column j^* in SIDE and TOP respectively so that sweep$(x_i^*$, SIDE) and sweep$(x_j^*$, TOP) belong to range[] of x_{ij}.
- finding the limits of sweep$(x_i^*$, SIDE) and sweep$(x_j^*$, TOP) along the line normal to the reference and through x_{ij}. Let[$s$min*, $s$max*] and [$t$min*, $t$max*] give sweep$(x_i^*$, SIDE) and sweep$(x_j^*$, TOP) on this line (figure 3), *range*[] is then given by the interval [$r0, r1$] where

$$r0 = \min(s\text{min}^*, t\text{min}^*),$$

$$r1 = \max(s\text{max}^*, t\text{max}^*).$$

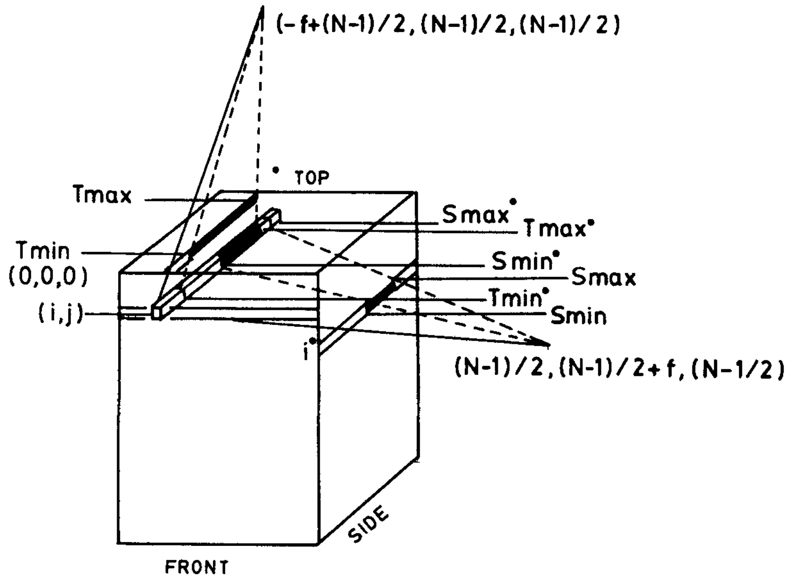

Figure 3. Range[] for perspective silhouettes.

These two steps involve the intersection of lines or the intersection of a line and a plane. These can be accomplished incrementally avoiding divisions.

3.3b *Updating of range*[]: The fact that sweep(x, FRONT) is not parallel to the z-axis but lies along the projector makes an explicit back-projection step of FRONT necessary. If the cube is visualised as a set of planes all parallel to FRONT, then the voxel(i, j, k) is given by the pixels(i, j) in plane k. The sweep(x, FRONT) would be given by the black pixels in these planes along the projector through x. The intersection of sweep(x, FRONT) for all x in FRONT with the range[] can be accomplished by the comparisons of black pixels in these planes with the corresponding nodes in the segment tree. For efficient implementation, FRONT could be represented in rectangular code bringing down drastically the number of comparisons. Moreover, a rectangular representation of FRONT would also facilitate its back-projection to these planes as one would need to back-project only the corner points of the rectangles to identify the project(v, FRONT) in these planes.

This approach of computation and updating of range[] makes it possible to use different views with any orientation. This is generalized whereas the one given for orthographic works only for the FRONT, SIDE and TOP views. Such a method could be adopted for orthographic views with arbitrary directions.

3.4 *Active vision*

An emerging theme in current vision research is that of active vision, in which the vision process is dynamic rather than static. Instead of applying image analysis to a single *snapshot* environment, the active vision process, in its most general implementation, applies image analysis operations in a purposive and integrative manner. In the present context, we address the active vision process to dynamically determine the sensor location for the best possible object reconstruction.

The volume intersection method cannot accurately recover the 3D shape due to the many-one mapping inherent to *sweep* and *project*. Even though a pixel is seen to be black in pure mathematical sense, the volume intersection (VI) process generates a set of black voxels given by

$$\bigcap_{x,k} \{\text{sweep}(x,k), k \in I_k\}, 1 < k \leqslant p.$$

Let us call this set the *VI hull* of the object. Clearly *VI hull* contains the object irrespective of the set of viewing directions. The accurate reconstruction is obtained when the volume intersection hull is the object itself. Thus any active vision process aims at determining the viewing direction incrementally in order to construct the VI hull as close to the object as possible. In this context, we review two approaches.

3.4a *Voxel fixing method*: A voxel is said to be *fixed* if there exists a direction d_k such that

$$\text{sweep}(x,k) = \{v\}, \text{ for some pixel } x \in I_k.$$

If a voxel is not fixed, it is called a free voxel. The aim of the active vision process is to determine viewing directions so as to fix as many free voxels as possible. At any particular stage, a new direction is chosen which can expose the maximum number of free voxels for the purpose of fixing. It can be noted that if for any I_k and for $x \in I_k$, sweep(x, k) contains at least one fixed voxel, then no other voxel can be fixed using I_k. This is because the fixed voxel occludes all the remaining voxels. Lavakusha *et al* (1989b) propose a method for fixing the voxel in which an optimization strategy is followed to incrementally fix voxels till a satisfactory accuracy is attained. The accuracy measure is defined as follows:

$$\sigma = (bf + w)/N^3,$$

where

bf = total number of black voxels that are fixed,
w = total number of white voxels, and
N = size of the arrays, hence N^3 is the total number of voxels.

When σ is very close to 1, the algorithm is terminated. Otherwise, if the accuracy is not satisfactory, a new direction is determined after unfixing the voxels at the previous process.

3.4b *Minimum width method*: This method is based on a principle which is dual to the voxel fixing method. The degree of ambiguity, for a particular direction k, is dependent on the sizes of sweep(x, k), $x \in I_k$. Hence the measure of ambiguity of a voxel v can be defined as

$$\lambda_v = \text{Min}_k\{|\text{sweep}(x, k)| : v \in \text{sweep}(x, k)\}.$$

Clearly, a voxel is fixed if $\lambda_v = 1$. However, it is not possible to fix all the voxels of the reconstructed object. Moreover, it could be possible that the direction which fixes certain voxels may cause a higher degree of ambiguity in the remaining voxels (figure 4).

The present approach aims at minimising this overall degree of ambiguity. The proposed algorithm uses the following accuracy measure and works towards minimizing this ambiguity. This can be accomplished using the *minimum width method* where, starting from one silhouette, two orthogonal directions are computed based on the minimum width of the silhouette. Silhouettes in these two additional directions are taken, which ensures increase in the prescribed accuracy measure.

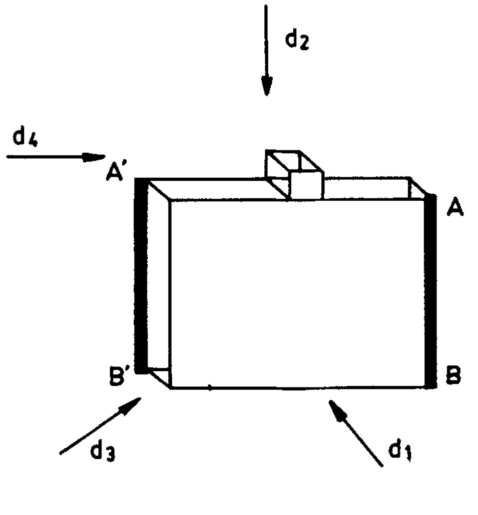

Figure 4. Voxels along AB and A'B' are fixed due to d_3, which generates a high degree of ambiguity for other voxels.

The accuracy is measured as follows:

$$\text{accuracy} = \frac{\text{volume of the object}}{\text{volume of intersection of silhouettes}}.$$

Experimental study confirms the following advantage of active vision over the conventional passive vision.

- This method provides a better reconstruction than the passive vision method of a prespecified set of views.
- It requires a lesser number of silhouettes than the passive case.

4. Drawbacks of the present approach

We have proposed an efficient method to recover 3D shape from multiple silhouettes using the volume intersection method. It has two major drawbacks.

(i) *Silhouette formation*: The silhouettes are obtained by a bilevel thresholding process from the grey-level image that is acquired with any imaging system. As a result, there is loss of information which cannot be recovered subsequently. For example, the edge information, which is prominent in a grey-level image is ignored in this process.

(ii) *Intersection process*: Due to the underlying principle of volume intersection, we get a unique 3D reconstruction; whereas, in reality, there may be many possibilities. The holes and cavities that do not render themselves to the occluding contours are suppressed in this process. Hence, this method may be inefficient for defect identification.

In Nagaraju (1991), a general framework for shape from silhouettes is proposed using first-order logic as the underlying tool. It is shown that objects that can never be reconstructed by the volume intersection method can be accurately reconstructed by using such a framework. This is due to the reasoning tools available in the theory of logic.

5. Conclusions

The structure range[], giving scope for incremental intersection, is versatile enough to be used with orthographic or perspective silhouettes in passive or active vision. For the set of three orthographic silhouettes, FRONT, SIDE and TOP, the computation and updating of range[] becomes very simple, giving the least complexity $O(N \log N)^2$ known so far for object reconstruction. For the three perspective views, the complexity of $O(N^4 \log N)$ is still acceptable as any approach using perspective silhouettes is expected to be computationally quite expensive in comparison. Experiments were carried out to reconstruct 3D objects from three orthographic and perspective views successfully (figure 5). For active vision, the reconstruction was impressive with 90% accuracy in comparison to the 70% accuracy for passive vision.

For clarity, range[] was discussed as if consisting of one interval $[r0, r1]$. But if rows of I_1 and columns of I_2 have multiple runlengths, then for a pixel in I_r, multiple

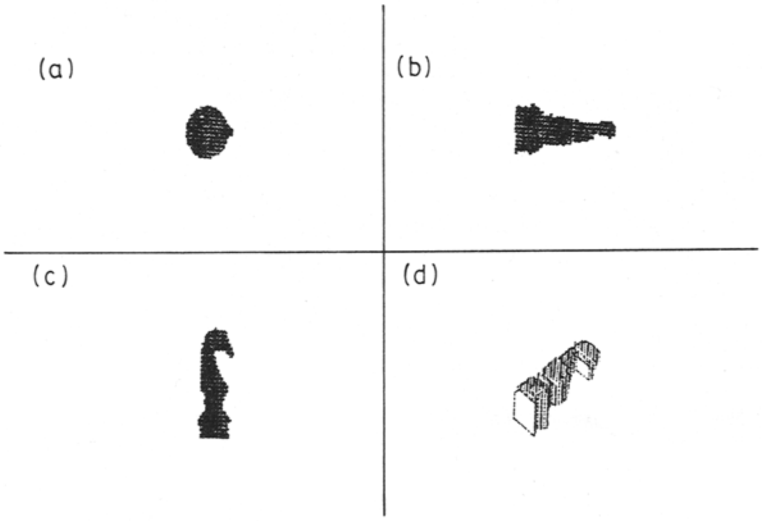

Figure 5. A chess piece: (**a**) front view; (**b**) side view; (**c**) top view; (**d**) angle in degrees (x, y): 45, 45.

range[] is a possibility. The computational process shown above can be trivially extended to handle such cases.

Volume intersection was discussed in detail for three orthographic and three perspective views in passive vision. Volume intersection, as discussed for orthographic views (FRONT, SIDE and TOP), is specific to the orientation of the silhouettes. But the method used in the computation and updating of range[] for perspective is independent of orientation and projection to a certain extent. Hence, such a method can be used to handle orthographic or perspective views of any orientation for object reconstruction.

For active vision, the most important step is the detection of the next viewing direction. Hence, only the detection step is explained. The computation and updating of range[] is the same as that given in passive vision.

The author gratefully acknowledges the contribution of other colleagues towards the research on volume intersection; Lavakusha, G Geetha, E Uma, Ramadevi, Anita Pai, H Usha, K Shanmukh, Nitya, Sridevi, Padmaja and Nagaraju.

References

Ahuja N, Veenstra J 1989 Generating octrees from object silhouettes in orthographic views. *IEEE Trans.* PAMI-11: 137–149

Chien C H, Aggarwal J K 1986 Identification of 3D objects from multiple silhouettes using quadtrees, octrees. *Comput. Vision, Graph., Image Process.* 36: 256–273

Kim Y C, Aggarwal J K 1986 Rectangular parallelopiped coding: A volumetric representation of 3D objects. *IEEE Trans. Robotics Autom.* 2: 127–134

Lavakusha, Pujari A K, Reddy P G 1989a Linear octrees by volume intersection. *Comput. Vision, Graph. Image Process.* 45: 371–379

Lavakusha, Pujari A K, Reddy P G 1989b Volume intersection algorithm with changing direction of views. *Proc. Int. Workshop on MIV-89*, Tokyo, pp. 309–314

Martin W N, Aggarwal J K 1983 Volumetric description of the objects from multiple views. *IEEE Trans. Pattern. Anal. Mach. Intell.* PAMI-5: 150–158

Nagaraju, Padmaja, Pavan Kumar, Pujari A K 1991 A logical framework for recovering shape from silhouettes. *Proc. 2nd Indian Computing Congress*, Hyderabad

Nitya B, Sridevi N, Pujari A K 1992 Linear octree by volume intersection using perspective silhouettes. *Pattern Recogn. Lett.* (communicated)

Nobborio H, Fukuda S, Arimoto S 1988 Construction of octrees approximating three-dimensional objects by using multiple views. *IEEE Trans. Pattern Anal. Mach. Intell.* PAMI 10: 769–781

Pai A G, Usha H, Pujari A K 1990 Linear octree of a 3D object from 2D silhouettes using segment tree. *Pattern Recogn. Lett.* 11: 619–623

Shanmukh K, Pujari A K 1991 Volume intersection with optimal set of directions. *Pattern Recogn. Lett.*

Srivastava S, Ahuja N 1987 An algorithm for generating octrees from object silhouettes in perspective views. *Proc. IEEE Workshop on Computer Vision* pp. 363–365