INTERFERENCE DETECTION AND COLLISION AVOIDANCE AMONG THREE DIMENSIONAL OBJECTS*

N. Ahuja, R. T. Chien, R. Yen, and N. Bridwell

Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

ABSTRACT

Two methods for detecting intersections among three dimensional objects are described. The first method involves detecting overlap among the projections of the objects on a given set of planes. The second method uses a three dimensional octree representation of the objects. Intersections are detected by traversing the trees for the obstacles and the moving objects. Application of the methods to collision avoidance is discussed.

I INTRODUCTION

An important problem in the robotics manipulation of its environment is that of following collision free trajectories when objects have to be moved. A representation or model of the spatial configuration of objects is necessary to obtain safe solutions, i.e. those which avoid collisions. In addition, a capability to plan an efficient trajectory, say, one following the shortest available path between the starting and the goal configurations, is desirable. In either case, a procedure is required to decide if an arbitrary configuration involves occupancy of a given region of space by multiple objects.

The problem of interference detection usually involves static objects whereas collision detection refers to a situation where at least one object is in motion. Collision detection may be viewed as a sequence of intersection checks among appropriately defined static objects, and at appropriate time intervals. Thus the basic problem appears to be that of detecting intersection among a set of objects.

Given a configuration of objects, it is not hard to develop an algorithm to check any intersections among the objects. However, in a reasonably complex environment and for reasonable speeds of object manipulation, e.g. translation, rotation, etc., the availability of a limited computational power may demand efficient procedures. At the heart of the design of such procedures lies the need for a good representation of the objects. Algorithms must be developed that take advantage of the properties of chosen representation to efficiently track the dynamic environment.

In the past, most of the related work has used polyhedral approximation of objects [1,2,4, 5,11]. This facilitates a treatment of a complex solid in terms of a set of relatively simpler, planar patches used to approximate its surface. For the case of convex objects, Comba [2] obtains a pseudo-characteristic function in terms of expressions for the planar patches. The function assumes nonpositive values in regions that approximate the parts of the space where all the objects intersect. Maruyama [5] compares the minimal boxes containing the objects. Boyse [1] considers three different types of objects: solids, surfaces and containers. Intersections are detected by checking if an edge of one object intersects a face of another. Collisions are detected by checking interference among the obstacles and the surfaces traced by the edges and faces of the moving objects.

Udupa [11] uses two different representations for the manipulator and the environment. The obstacles are modelled by polyhedra (not necessarily convex) in cartesian space. The manipulator links of the Scheinman arm are approximated by the minimum bounding cylinders. Abstractions are introduced by replacing the links by their axes and enlarging the obstacles proportionately, by retaining only the major link, etc. Lozano-Perez and Wesley [4] also grow the obstacles and shrink the moving parts such that collisions in the modified representation occur if and only if they occur in the original space.

It is clear that the representation of the objects plays a major role in determining the feasibility and performance of any intersection or collision detection method using that representation. This paper discusses two methods of representation with the aim of making the resulting interference checking operations more efficient.

Section II examines a method based upon a set of two dimensional projections of the objects. It uses a conservative criterion to detect the occurrence of interference. Section III discusses a representation of the objects by regions of a fixed shape, but varying sizes determined by a recursive decomposition of the space into successively smaller regions. Section IV outlines the use of the representations described in Sections II and III to collision avoidance. Section V presents some concluding remarks.

II PLANAR PROJECTIONS

A planar projection of a three dimensional configuration of objects will always show an overlap between the projections of any objects that intersect. However the reverse is not necessarily true, i.e., overlapping projections do not necessarily correspond to intersecting objects. Such false alarms may sometimes be decreased by considering projections on a collection of planes. A configuration then is adjudged to be safe if each pair of objects has nonoverlapping projections on at lease one plane. However, for any given number and choice of planes, spatial arrangements of noninterfering objects may be devised whose projections overlap. An increase in the number of planes may only decrease, but not eliminate, the probability of erroneous decisions. The error may also be lower for objects with simpler shapes, e.g., convex objects.

A. Basic Method

We will make the usual and reasonable assumption that objects may be represented by polyhedra. Each polyhedron is uniquely described by the coordinates of, and the adjacency relationships among its vertices. The projection of a polyhedron on any plane is determined by the projections of the vertices, using the original adjacency relationships.

Let O denote the origin of a three dimensional cartesian space. Let $\overline{OV_i}$ denote the vector location of the vertex V_i of a given polyhedron. A pair of vertices V_i and V_j are said to be adjacent if the straight line joining them defines an edge of one of the faces of the polyhedron. Now consider projecting the polyhedron on a plane. The projection will be the same for all planes parallel to the given plane. We will consider projecting the polyhedron on the plane P passing through the origin O. Let $\hat{n}$ denote the unit surface normal vector of P. Then the projection of $\overline{OV_i}$ along $\hat{n}$ is given by $\overline{OV_i}\cdot\hat{n}$. Therefore, the projection V_i' of V_i on P is given by the vector $\overline{OV_i'}$ where

$$\overline{OV_i'} = \overline{OV_i} - \overline{OV_i} \cdot \hat{n}$$

The set of vectors $\overline{OV_i'}$ for various i defines the vertices of the projection. To determine the shape of the polygonal projection, we must determine its border edges. This is done by obtaining the convex hull of the projections of the vertices. For convex polyhedra, the edges of the convex hull will be a subset of the set of straight lines $V_i'\ V_j'$, where V_i and V_j are adjacent vertices. On the other hand, nonconvex polyhedra may give rise to nonconvex polygonal projections. The convex hull of the corresponding projected vertices will include edges $V_i'\ V_j'$ where V_i and V_j are not adjacent. The actual border of the projection can be found by replacing all such edges $V_i'\ V_j'$ by a sequence of edges $V_{j_1}'\ V_{j_2}'$, $V_{j_2}'\ V_{j_3}'$, ... , $V_{j_{k-1}}'\ V_{j_k}'$, where $k > 2$, $j_1 = i$, $j_k = j$, and V_{j_r} and $V_{j_{r+1}}$ are adjacent, $1 \le r \le k-1$ (fig. 1). Obtaining the convex hull of N projected vertices takes O(N log N) time [10]. This also determines the overall complexity of obtaining a polygonal projection, since the computations required to obtain the vertices V_i' are O(N). Thus, assuming that the fraction of edges of a polygonal projection that form concave cavities is low, the entire set of operations takes O(N log N) time.

Given a set of M planes, projections of all the objects are obtained on each of the planes. The polygons corresponding to a given pair of objects to be examined for intersection are then checked for overlap in the different planes. If a plane is found in which the two objects do not overlap, noninterference between them is guaranteed. Otherwise the objects may or may not intersect.

Shamos [10] gives an O(m+n) algorithm to find the intersection of a convex m-gon with a convex n-gon. Shamos [10] also describes a simpler algorithm suggested by Hoey. We use the latter. We are not interested in obtaining the exact region of intersection, but only in knowing if two given polygons intersect. For an m-gon and an n-gon, this needs only O(m+n) computations compared to the O(m log m) and O(n log n) computations required to obtain the individual polygons.

When the projection of an object is nonconvex, we must extract a description in terms of convex polygons in order to make use of the above algorithm. One obvious way is to use the convex hull of each nonconvex m-gon. However this introduces an additional source of false alarm since the cavities in an object, appearing as concavities in its projection, are treated as solids.

An alternative way is to decompose a nonconvex polygon into a set of convex polygons (fig. 2). Each of these convex polygons must then be considered in each of the intersection tests where

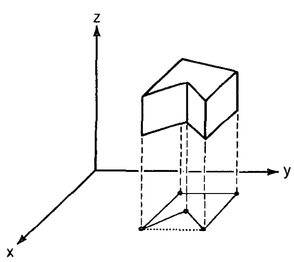

Figure 1

A nonconvex polyhedron and its nonconvex polygonal projection. To obtain the actual projection from its convex hull, the dotted line must be replaced by the remaining two sides of the triangle.

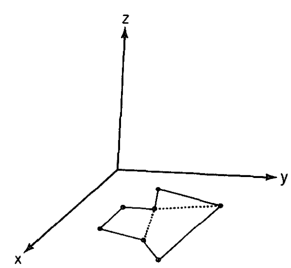

Figure 2

Decomposition of a nonconvex polygonal projection into convex polygons.

the parent polygon is involved. Pavlidis [7], and Feng and Pavlidis [3] give algorithms for convex decomposition. Schachter [9] constructs a partial Delaunay triangulation of an m-gon to obtain an O(rm) algorithm, where r is the number of concave vertices.

B. Improving Efficiency

The complexity of intersection detection increases linearly with the number of projection planes used, since each plane must be examined for overlap until a zero overlap is found, or the planes are exhausted. To avoid applying the intersection algorithm to pairs of polygons that are far apart, coarse tests of intersection on the envelopes of the polygons may be used. For example, rectangular envelopes may be obtained by identifying the points having minimum and maximum x or y coordinate values. Similarly, a circular envelope may be obtained in terms of the diameter of the polygon. An overlap between two envelopes can be detected in constant time. However, obtaining either of the two envelopes itself takes O(m) time for an m-gon, and hence such a coarse test may be of marginal utility since the intersection algorithm for polygons is already linear in the total number of vertices.

Suppose we are considering a pair of objects that do not intersect. Also suppose that their projections on at least one of the planes do not overlap. Then we would like to discover such a plane at the earliest. It may be useful to order the planes for examination according to some measure of the likelihood that a given plane demonstrates the desired relationships. An example of such a measure is the total black (polygonal) area. It takes O(m) time to compute the area of an m-gon. Thus the planes may be ordered for examination by performing a computation on each plane which is linear in the number of vertices in the plane.

The choice of the appropriate planes depends upon the given configuration. In general, a minimum of three planes would appear to be desirable to work with three dimensional objects. The computation of projections is trivial when the three orthogonal planes are used. The three projections are obtained by successively replacing one of its three coordinates by zero. Other planes with convenient orientations may also be used.

III THREE DIMENSIONAL REPRESENTATIONS

Extensions of the methods for representing two dimensional images [8] may be used for the representation of three dimensional objects. Thus MAT (medial axis transform), generalized cylinders and recursive subdivision of space may all be used among others. In this paper, we will be concerned with the third of these methods.

A. Octrees

Just as the plane is recursively divided into squares in the quadtree representation of the images [12, 13], the three dimensional space may be subdivided into octants [14]. We start with the entire space as one block. If a block is completely contained within the object whose representation is sought, it is left alone. Otherwise, it is divided into eight octants (fig. 3a) each of which is treated similarly. The splitting continues until all the blocks are either completely within or completely outside the object (fig. 3a), or a block of minimum allowed size is reached, reflecting the limits on resulution. A method of subdivision of space was also employed by Maruyama [6]. However, he used rectanguloids. A rectangular block was divided into two, along one of the three axes as determined by the region of intersection of the block with the object.

The recursive subdivision allows a tree description of the occupancy of the space (fig. 3b). Each block corresponds to a node in the tree. Let us label a node black or white if it corresponds to a block which is completely contained within the (black) object or the (white) free space, respectively. Otherwise the node is labelled gray. The gray nodes have children unless they are of the minimum allowed size, in which case they are relabelled black. The free space covered by such nodes is thus treated as part of the object in order to be conservative in detecting interference. The final tree has only black or white leaves.

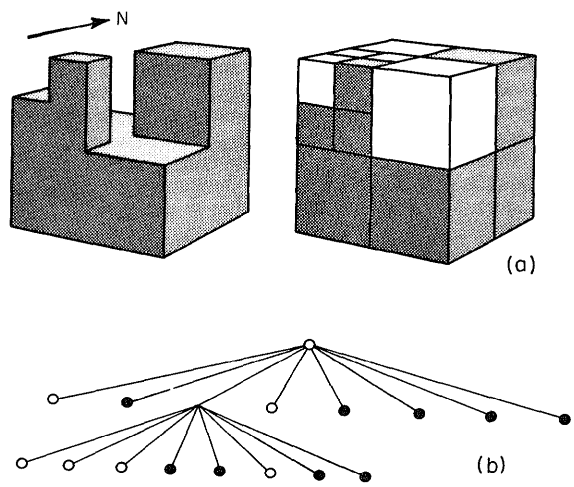

Figure 3

(a) An object and its representation by recursive subdivision of space into octants.

(b) Octree for the object in (a). The north-west, north-east, south-west and the south-east octants in the upper layer correspond to the children having labels 1, 2, 3 and 4, respectively. The nodes corresponding to the octants in the lower layer are labelled 5, 6, 7 and 8. Dark (white) circles indicate black(white) leaves. The children nodes are arranged in increasing order of label values from left to right.

B. Interference Detection

Suppose we are given the octree representations of a pair of objects. Clearly, the objects intersect if there exists at least one pair of corresponding nodes in the two trees such that one of them is black, and the other is black or gray. Thus the existence of interference can be determined by traversing the two trees in parallel. Let A and B denote a pair of corresponding nodes at any time during the traversal. If either A or B is white, we do not traverse their children, and continue the traversal of the remaining nodes. The depth of the traversal along each path down from the root is determined by the shallower of the two paths terminating in a white leaf. The time required is proportional to the number of nodes in the subset of the nodes traversed.

To detect interference among n objects, $n > 2$, n trees must be traversed. The traversal below a certain node in a tree stops only if either the node is white, or the corresponding nodes in the remaining n-1 trees are all white. The time required depends upon the actual number of nodes visited.

IV COLLISION AVOIDANCE

The use of interference detection in robotics manipulators lies in planning trajectories of moving objects in a given environment. The occupancy of any part of the space by more than one object can be used as indicative of a collision. To avoid a collision its imminence must be foreseen. Therefore, intersections should be detected in a modified space. Suppose that it requires at least a distance d to brake the speed of a moving object or to change its course to avoid a collision. The value of d may depend upon different factors including the speed. Then we must detect any obstacles within a distance d along the path of a moving object. For the obstacles off the path of the object, the required safe distance can be easily computed in terms of the relative location of the obstacles. Similarly, for two moving objects the required safe distance is 2d if they are approaching each other head on, and larger otherwise, given by the sizes of the objects, their speeds and directions of motion.

To use intersection detection for collision avoidance, the moving objects are grown [1,11] by a thickness d. Any intersection among the modified set of objects produces a timely warning of a possible collision.

The representations of the (static) obstacles are computed only once. Thus, we have a fixed set of projections of the obstacles on the planes being used, and a single octree representing the space occupied by the obstacles. Each moving object is represented separately. Every time an interference check has to be made, the representations of the moving objects are obtained. Thus, in case of the planar projections, the current state of each of the moving objects is projected on each of the planes, which already have the appropriate projections of the obstacles. The usual procedure to check intersection is then carried out. In case of the octree representation, a new tree is generated for each of the moving objects. A parallel traversal of these trees and the obstacle tree detects any interference present.

The above checks for intersection are applied to a snapshot of the potentially continuously varying spatial configuration of objects. Hence care must be taken to ensure that no collisions will occur between any two successive executions of the algorithm. This requires that not only should we detect any existing intersections, but also note if some objects are too close, and might collide before the instant when the algorithm is applied again. The safe distance D between any two objects is proportional to the speeds and relative locations of the objects involved. The interval T between two successive applications of the algorithm is controlled such that the objects do not come closer than D during the time interval T. This is accomplished by growing the objects further by a distance D. Then the problem of collision avoidance only involves periodic intersection detection among the modified objects.

V CONCLUDING REMARKS

We have discussed two approaches to detecting interference and collision among three dimensional objects. The first method involves detecting overlaps among projections of the objects on a given set of planes. It may thus be viewed as a two and a half dimensional approach. The criterion used for detecting interference is a conservative one, since for any given set of planes there exist many spatial configurations of noninterfering objects whose projections overlap on each of the planes. The second method uses a three dimensional octree representation of the objects, and does not suffer an information loss from the reduction in dimensionality like the first method. Here, the interference is detected by a parallel traversal of the octrees for the obstacles and for each of the moving objects.

Computer controlled manipulation often involves the use of a manipulator such as the Scheinman arm. Its degrees of freedom are given by the boom length and the joint angles. These parameters thus define a natural representation of the arm. In particular, the updating of the representation as the arm moves becomes trivial. However, for the other objects in the environment, the above representation is not very useful. In addition, the origin of the cylindrical coordinate system above moves with the manipulator. Therefore, the representation of the whole environment must be modified for each new position of the manipulator as noted by Udupa [11]. The use of the cartesian space for the representation of all objects requires regeneration of the octree for the moving objects only.

Use of hardware such as proximity sensors may significantly improve the efficiency of the procedures. For example, a "cluttered zone" signal from a certain sensor may channel the available computational power to a detailed investigation of the desired region of space, leaving the relatively safe areas unexamined. This essentially amounts to being a hardware implementation of comparing coarse envelopes of objects in parallel.

The computation of projections of the polyhedra in Section II may also be replaced by cameras in the appropriate planes. This will provide real time projection generation. The projections are then treated as a collection of two dimensional binary images. The projections of the moving objects are tracked and their positions checked against the projections of the obstacles.

We have not addressed the problem of finding good trajectories for moving an object from a given source position to a given goal position. The representations discussed here, in conjunction with the planning strategies discussed in [4,11], may be used to develop the desired trajectories. Experiments employing the methods described in this paper, and two Scheinman arms are being carried out currently.

ACKNOWLEDGEMENTS

This work was supported in part by the United States Department of Transportation and Federal Aviation Administration under Contract DOT FA79-WA-4360 and the Joint Services Electronics Program (U. S. Army, U. S. Navy and U. S. Air Force) under Contract N00014-79-C-0424.

REFERENCES

[1] J. W. Boyse, "Interference Detection among Solids and Surfaces," Comm. ACM 22, January 1979, pp. 3-9.

[2] P. G. Comba, "A Procedure for Detecting Intersections of Three-Dimensional Objects," Jnl. ACM, 15, July 1968, pp. 354-366.

[3] H. Feng and T. Pavlidis, "Decomposition of Polygons into Simpler Components," IEEE Trans. Comp. C-14, June 1975, pp. 636-650.

[4] T. Lozano-Perez and M. A. Wesley, "An Algorithm for Planning Collision-free Paths among Polyhedral Obstacles," Comm. ACM 22, October 1979, pp. 560-570.

[5] K. Maruyama, "A Procedure to Determine Intersections between Polyhedral Objects," Int. Jnl. Comp. Inf. Sci. 1, 3, 1972, pp. 255-266.

[6] K. Maruyama, "A Procedure for Detecting Intersections and its Application," University of Illinois Computer Science Technical Report No. 449, May 1971.

[7] T. Pavlidis, "Analysis of Set Patterns," Pattern Recognition 1, 1968, pp. 165-178.

[8] A. Rosenfeld and A. C. Kak, Digital Picture Processing, Academic Press, New York, 1976.

[9] B. J. Schachter, "Decomposition of Polygons into Convex Sets," IEEE Trans. Comp. C-27, November 1978, pp. 1078-1082.

[10] M. I. Shamos, Computational Geometry, Springer-Verlag, New York, 1977.

[11] S. Udupa, "Collision Detection and Avoidance in Computer Controlled Manipulators," Proc. 5th Int. Joint Conf. Art. Intel., Cambridge, Massachusetts, 1977, pp. 737-748.

[12] G. M. Hunter and K. Steiglitz, "Operations on Images Using Quadtrees," IEEE Trans. Pattern Analysis Mach. Int. 1, April 1979, pp. 145-153.

[13] A. Klinger and C. R. Dyer, "Experiments in Picture Representation using Regular Decomposition," Computer Graphics and Image Processing 5, 1975, pp. 68-105.

[14] C. L. Jackins and S. L. Tanimoto, "Oct-trees and their Use in Representing Three-dimensional Objects," University of Washington Computer Science Technical Report, January 1979.