ELSEVIER European Journal of Operational Research 114 (1999) 437–445

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

Fast collision detection in four-dimensional space

M.D.S. Aliyu [*], K.S. Al-Sultan [1]

Department of Systems Engineering, King Fahd University of Petroleum and Minerals, Box 572, Dhahran – 31261, Saudi Arabia

Received 18 December 1996; accepted 30 November 1997

Abstract

In this paper, we consider the collision detection problem for general objects. A four-dimensional approach is proposed for this problem which detects exactly and in one-step when and where the earliest collison will occur between the objects. This is done by using four-dimensional sets to represent the objects in both space and time. The problem is then posed as a nonlinear programming problem. The algorithm can handle the case of a rigid body moving on a general path in $\mathbb{R}^2$ or $\mathbb{R}^3$ with simultaneous translation and rotation. Simulation results on some example problems are given, and show that the algorithm is superior to those available in the literature. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Collision detection; Four-dimensional space; Nonlinear programming

1. Introduction

The collision detection problem has been extensively studied in the robotics literature and various classes of algorithms are available [1–3,5,8–13,15–26]. For a survey of these various algorithms, see Refs. [1,2]. In [1], two algorithms that use the multiple interference methodology [10] are presented. The algorithms apply to objects that can be represented as polyhedral sets defined as intersections of finite halfspaces or as convex hulls of a finite number of vertices. The algorithms can give in principle exactly the collision points (in both time and space) for objects moving on a general path in $\mathbb{R}^2$ or $\mathbb{R}^3$ with simultaneous translation and rotation. However, since the algorithms essentially reduce the dynamic interference detection problem (collision detection) into a static interference checking by discretizing the space-time frame into finite grids at which intersection is tested, the decisive choice of the time step length is very crucial to the reliability of the algorithms. Furthermore, polytopic representation of objects may be very poor in certain applications especially when smooth objects are involved.

In this paper, we remove some of the limitations of the algorithms in [1], and perform collision detection in four-dimensional space directly. A formal notion of four-dimensional intersection testing was introduced by Cameron [10,11] using an "extrusion" operation. However, constructive solid

[*] Corresponding author. E-mail: dikko@ccse.kfupm.edu.sa
[1] E-mail: alsultan@ccse.kfupm.edu.sa

geometry was used in the implementation of the algorithm, and various techniques for null set detection were employed. Therefore in this paper, we follow up the development in [10,11] and present analytical methods for four-dimensional intersection testing which are more amenable to computations, and make the algorithm computationally efficient. The algorithm also finds the collision point in one-step without discretization, and can be applied to both polytopic and smooth objects (that may be nonconvex in nature) moving on a general path in $\mathbb{R}^3$ (not necessarily linear) with simultaneous translation and rotation.

In Section 2, we review methods of representation of objects in four-dimensional space as a basis for the new algorithm. Then in Section 3, we formally introduce the problem and propose the new algorithm. We also discuss some of the ways that the algorithm can be extended to nonconvex objects in Section 4. This is followed by computational results from simulations obtained using the algorithm in Section 5. Finally, in Section 6, we give conclusions and suggestions for future work.

2. Four-dimensional representation of moving objects

Cameron [10,11] used the concepts of *sweeping* and *extrusion* to represent the four-dimensional sets generated by moving objects. While the former refers to the volume swept by the moving object over time, the latter refers to the set of all points occupied by the object at a particular time t. For an object A represented by a compact set K_A, with a location function $\Lambda_A(t)$ which describes its position at a given time t, its swept volume over a finite time interval is represented by the set

$$\mathrm{Sw}(K_A, \Lambda_A(t)) = \{x \mid x \in \Lambda_A(t)(K_A)\ \forall\, t\}, \tag{1}$$

where "Sw" is the sweeping operator. To get a better understanding of the above operation, consider a unit cube defined by the set

$$C(x,y,z) = \{(x,y,z)|0 \leqslant x \leqslant 1,\ 0 \leqslant y \leqslant 1,\ 0 \leqslant z \leqslant 1\}. \tag{2}$$

Assume that the center of the cube is to move along a straight line path to the point (10, 10, 10) at a speed of 1 unit/s. The location function of the cube is then defined by

$$L_c(t) = \{x(t) = t,\ y(t) = t,\ z(t) = t;\ 0 \leqslant t \leqslant 10\}. \tag{3}$$

To determine the sweep of such an object, we need to project all its corners parallel to the given path from its initial position to the final position. It can however be seen that, it will need a large number of inequalities (halfspaces) to describe this swept volume. The best we can do is to represent it as a union of an infinite number of cubes. Thus,

$$\mathrm{Sw}(C, L) = \lim_{\Delta t \to 0, N \to \infty} \bigcup_{i=1}^{N} C_i(x,y,z,t), \tag{4}$$

where

$$C_i(x,y,z,t) = \{(x,y,z)|i\,\Delta t \leqslant x \leqslant 1 + i\,\Delta t,\ i\,\Delta t \leqslant y \leqslant 1 + i\,\Delta t, i\,\Delta t \leqslant z \leqslant 1 + i\,\Delta t\}.$$

Furthermore, if any two or more objects interfere, their sweeps must intersect. This is however not a sufficient condition. Also, based on the above example of the cube, it is difficult to come up with analytical expressions for the interference region between the objects.

On the other hand, in extruding the above cube, we represent it by the following set:

$$C(x,y,z,t) = \{(x,y,z,t) \mid t \leqslant x \leqslant 1+t, t \leqslant y \leqslant 1+t, t \leqslant z \leqslant 1+t, 0 \leqslant t \leqslant 10\}. \tag{5}$$

As another example, a sphere of radius 4 and centre (5, 5, 5) at time 0 and moving with velocity (1, 1, 1) units/s is represented by the four-dimensional set

$$S(x,y,z,t) = \{(x,y,z,t)|(x-5-t)^2 + (y-5-t)^2 + (z-5-t)^2 \leqslant 16\}. \tag{6}$$

In general, given an object A with location function $\Lambda_A(t)$ as above, its extrusion is given by the set

$$\mathrm{Ex}(\Lambda_A, K_A) = \{(x,t) \mid x \in \Lambda_A(t)(K_A)\}, \tag{7}$$

where "Ex" is the extrusion operator. Further, if any two or more objects interfere, then it is necessary and sufficient that their extrusions intersect or have a common point.

Suppose that the objects under consideration can be represented in their initial position and orientation by compact convex sets of the form

$$G_0 = \{x\colon\ g_0(x) \leqslant 0,\ g_0 : \mathbb{R}^n \to \mathbb{R}^m,\ n = 1,2,3, g_0 \in \mathscr{C}^2\}. \tag{8}$$

With this representation, both smooth and nonsmooth objects can be treated. Furthermore, in [1] the authors suggested a polytopic representation of the objects where $g_0(x)$ is a linear function of the form

$$g_0(x) = \Lambda x - b \leqslant 0, \tag{9}$$

where $\Lambda \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. The authors have also shown that for rigid objects represented in the form (8) and (9), any rigid-body transformation (or sequence of transformations i.e., translations and rotations) results in the new instantaneous geometry given by the following set:

$$\bar{G}_0 = \{(x,t)|\ \bar{g}_0(x,t) = \Lambda R^{-1}(t)(x - p(t)) - b \leqslant 0\}, \tag{10}$$

where $R(t) \in \mathbb{R}^{3\times 3}$ and $p(t) \in \mathbb{R}^3$ are the rotation matrix and translation vector [14] parameterized in time t respectively.

In this paper, we extend the representation (10) to cover a more general class of objects (smooth convex objects) that can be represented by the following set

$$G = \{x\colon\ g(x) = x^{\mathrm{T}}Qx + c^{\mathrm{T}}x + d \leqslant 0;\ x \in \mathbb{R}^n,\ Q \in \mathbb{R}^{n\times n},\ c \in \mathbb{R}^{n\times 1},\ d \in \mathbb{R}\}, \tag{11}$$

where Q is a positive semidefinite matrix.

This can be used to represent spheres, ellipsoids, cylinders etc. Moreover, even polytopes can be represented in this form. The following proposition gives a unique representation of an object in four-dimensional space under rigid-body transformation.

Proposition 2.1. *Consider a solid object represented by the set G defined by Eq. (11). Then the resulting geometry of the object as a function of time under a rigid-body transformation is represented by the following set:*

$$\begin{aligned}\bar{G} = \{(x,t)|g(x,t) &= (x - p(t))^{\mathrm{T}}R(t)QR(t)^{\mathrm{T}}(x - p(t))\\ &\quad + c^{\mathrm{T}}R(t)^{\mathrm{T}}(x - p(t)) + d \leqslant 0\},\end{aligned} \tag{12}$$

where $R(t), p(t)$ are the appropriate rotation matrix and translation vector parameterized in time respectively.

Proof. Let x be any point on the object desribed by the set (11). Suppose, and without loss of genarality, that the object undergoes a rigid-body transformation comprising of a rotation and a translation, then the point x is transformed to the point $\bar{x} \in \bar{G}$ at time t, given by

$$\bar{x} = R(t)x + p(t). \tag{13}$$

Substituting for x in Eq. (11) and using the fact that $R^{-1}(.) = R^{\mathrm{T}}(.)$, we get

$$\begin{aligned}\bar{g}(x,t) &= (\bar{x} - p(t))^{\mathrm{T}}R(t)QR^{\mathrm{T}}(t)(\bar{x} - p(t))\\ &\quad + c^{\mathrm{T}}R^{\mathrm{T}}(t)(\bar{x} - p(t)) + d \leqslant 0.\end{aligned} \tag{14}$$

Since x is arbitrary, we can drop the bar on the x and regard the above expression as the transformed object or in the terminology of Cameron [11], the extrusion of the original object. □

Furthermore, there has been some progress in research in the approximation of nonsmooth objects by smooth ones. By appropriately shaping a superquadric function [4], many other complex solids can be brought into the above fold. A superquadric function can be defined by

$$f(x,y,z) = \left[\left(\frac{x}{f_1(x,y,z)}\right)^{2r} + \left(\frac{y}{f_2(x,y,z)}\right)^{2r}\right]^{2q/2r} + \left(\frac{z}{f_3(x,y,z)}\right)^{2q} - 1 = 0, \tag{15}$$

where $r,\ q$ are parameters and $f_1,\ f_2,\ f_3$ are scaling functions.

3. Collision detection in four-dimensional space

A formal definition of the collision detection problem is as follows:

Definition 3.1. Given representations of $N+1$ convex objects $A, B_1, B_2, \dots, B_N$, whose locations in space at any time t are represented by the sets $G_A(t)$, $G_{B_1}(t)$, $G_{B_2}(t), \dots, G_{B_N}(t)$, respectively, over a time interval $[t_s, t_f]$, determine whether any pair of the objects occupy some common space at the same time during this interval.

Without any loss of generality, we can assume A in the above definition to be a robot (or robot link) moving in a workspace $W \subset \mathbb{R}^n$ where there are obstacles $B_i, i = 1, 2, \dots, N$. Hence we are particularly interested in checking whether

$$\exists\, t \in [t_s, t_f] \ni G_A(t) \cap G_{B_i}(t) \neq \emptyset, \quad i = 1, 2, \dots, N, \tag{16}$$

where $G_A(t)$, $G_{B_i}(t)$, $i = 1, 2, \dots, N$, are of the form (10) or (12). With the above compact representation of the objects by Eqs. (10) and (12), we now present the problem as an optimization problem.

Let the motion of each object be represented by the pair of rotation matrix and translation vector $[R_A(t), p_A(t)]$, $[R_i(t), p_i(t)]$, $i = 1, \dots, N$, whose elements are parameterized in time t. To determine whether any pair of the objects say A and B_i, represented by the sets $G_A(t)$ and $G_{B_i}(t)$ respectively, collide, we seek a feasible solution to the following system of inequalities:

$$\begin{aligned} &g_A(x, y, z, t) \leqslant 0, \\ &g_{B_i}(x, y, z, t) \leqslant 0, \\ &t_s \leqslant t \leqslant t_f, \end{aligned} \tag{17}$$

where $g_A(.)$, $g_{B_i}(.)$ are of the form (10) or (12). There are algorithms available for finding feasible solutions to the above system [27]. Alternatively, the problem can be posed as an optimization problem with zero (or any) objective function. If the above system has no solution, then the objects A and B_i will not collide in the time interval $[t_s, t_f]$. However, if the above system has a solution, then any solution represents a collision point in terms of location and time. To get all collision points, one has to enumerate all solutions of the above system. If however, one is interested in finding the earliest collision in $[t_s, t_f]$, then one can solve the following optimization problem:

$$\begin{aligned} \min \quad & t \\ \text{s.t.} \quad & g_A(x, y, z, t) \leqslant 0, \\ & g_{B_i}(x, y, z, t) \leqslant 0, \\ & t_s \leqslant t \leqslant t_f. \end{aligned} \tag{18}$$

For every pair $(A, B_i), i = 1, 2, \dots, N$, one can solve the above problem and get the earliest collision in $[t_s, t_f]$.

The above problem is a nonlinear programming problem [7], and efficient algorithms for solving such a problem exist [27]. Thus, by solving the above problem, we get an exact solution to the collision detection problem; and if the problem has no solution, it means that there is no possible collision between the objects for all the time. Furthermore, the solution is obtained in one step without having to compute the distance between the objects at every step [5,16–18,21,23,24] or combinatorially checking for intersection between various features of the objects [8,9,13,20,22]. Notice also that the above algorithm is a generalization of the algorithm in [1] with the linear system replaced by a nonlinear system and the linear program [6] replaced by a nonlinear program. Moreover, we eliminate the discretization of the time-dimension which could cause errors in detecting collisions. The algorithm can also be extended to deal with multiple moving/stationary objects, and by changing the bounds on the variables, it can be used to detect all possible collisions between the objects.

4. Extension to nonconvex objects

In the previous sections, we have presented an approach for representing complex objects in three-dimensional space using convex sets, and developed an algorithm for detecting possible collisions between them when their motion is parameterized in time. In this section, we seek to extend the representation in Section 2 to cover nonconvex objects. Such objects abound everywhere and a complete solution of the collision detection problem should certainly encompass nonconvex objects. It has always been a problem to represent nonconvex objects directly using algebraic or analytical methods; usually, they are approximated by their convex hulls [7] or as union of convex objects. However, in this section, we show that a host of nonconvex objects can be represented as intersections of finite number of convex and nonconvex sets, and hence our algorithm can be applied to them directly.

For example, consider the intersection of two discs in two-dimensions defined by the set

$$D = D_1 \cap D_2, \tag{19}$$

where

$$D_1 = \{x | s_1(x) \leqslant r_1,\ x \in \mathbb{R}^2, r_1 \in \mathbb{R}\} \tag{20}$$

and D_2 is a hollow disc defined by the set

$$D_2 = \{x |\ s_2(x) \geqslant r_2\, x \in \mathbb{R}^2, r_2 \in \mathbb{R}\} \tag{21}$$

as shown in Fig. 1 below. Where $s_1(x) = r_1$ and $s_2(x) = r_2$ are the equations of the bounding circles, respectively. Clearly D is a nonconvex disc. Similarly, many other nonconvex sets can be represented as in the above, and our algorithm can be applied to them.

In general, using the representation (11), the intersection of a convex and a nonconvex object can be represented by the set

$$\tilde{G} = \{x |\ \tilde{g}(x) \leqslant 0, x \in \mathbb{R}^n\}, \tag{22}$$

where

$$\tilde{g}(x) = x^{\mathrm{T}} \begin{bmatrix} Q_1 \\ -Q_2 \end{bmatrix} x + \begin{bmatrix} c_1^{\mathrm{T}} \\ -c_2^{\mathrm{T}} \end{bmatrix} x + \begin{bmatrix} d_1 \\ -d_2 \end{bmatrix} \leqslant 0 \tag{23}$$

and Q_1, c_1, d_1 and Q_2, c_2, d_2 represent the corresponding objects.

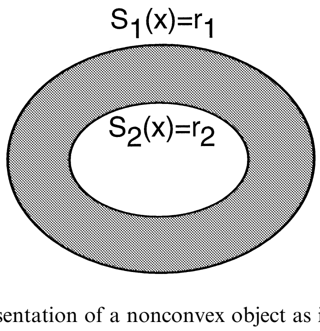

Fig. 1. Representation of a nonconvex object as intersection of two discs.

Remark. In fact, if either Q_1 or Q_2 is nonpositive semidefinite, then the corresponding sets do not define a convex object, while the limiting case when $Q_i = 0$, $i = 1, 2$, will always define a convex object; literally a polyhedron.

5. Simulation results

In this section we give results of simulation with the algorithm on various example problems to show its efficiency. We use a standard subroutine *"constr"* for solving constrained optimization problems from the MATLAB Optimization Toolbox [27] to implement the algorithm.

The first example we consider is from [10].

Example 5.1. Fig. 2 shows a sphere of radius 4 centered at (5, 5, 5) at time 0 moving with velocity (1, 1, 1) unit/s and a cube of sides 4 units centered at (44, 54, 5) at time 0 moving with velocity (0, 0, 1). It is required to find if there is any possible collision between them over the time span 0–64.

Let the two objects be represented by the four-dimensional sets

$$g_1(x, y, z, t){:}\ (x - 5 - t)^2 + (y - 5 - t)^2 + (z - 5 - t)^2 \leqslant 16 \tag{24}$$

$$g_2(x, y, z, t) : \begin{cases} 42 \leqslant x \leqslant 46, \\ 52 \leqslant y \leqslant 56, \\ 3 \leqslant z - t \leqslant 7. \end{cases} \tag{25}$$

This problem was solved on an IBM/DX 486 machine. A collision was detected at (46, 52, 47, 44) within 11.25 s. However, such a conclusion could not be arrived at in [11].

Example 5.2. Consider the objects shown in Fig. 3, an ellipsoidal object and a spherical object described by the following systems:

$$g_1(x, y, z){:}\ \frac{x^2}{2^2} + \frac{y^2}{1} + \frac{z^2}{1} \leqslant 1, \tag{26}$$

$$g_2(x, y, z){:}\ (x - 10)^2 + (y - 10)^2 + (z - 5)^2 \leqslant 4.$$

The sphere is moving on a path

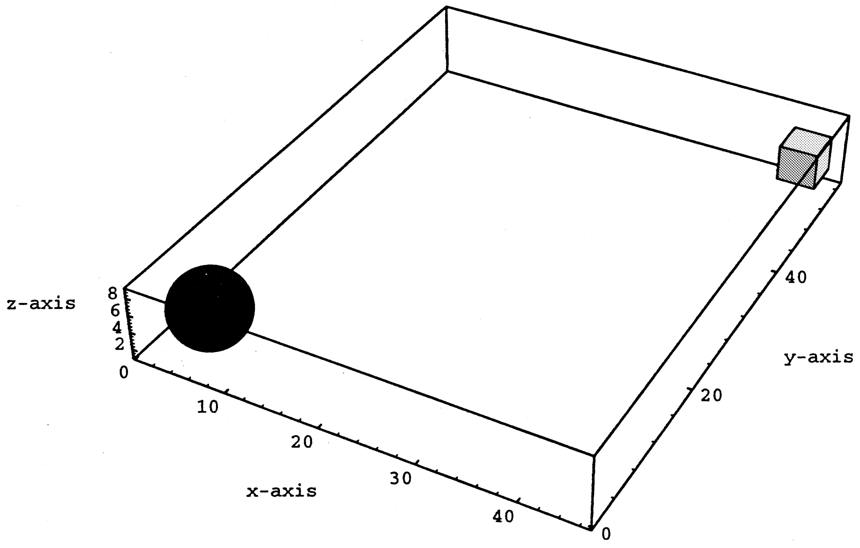

Fig. 2. Example 1 from [10].

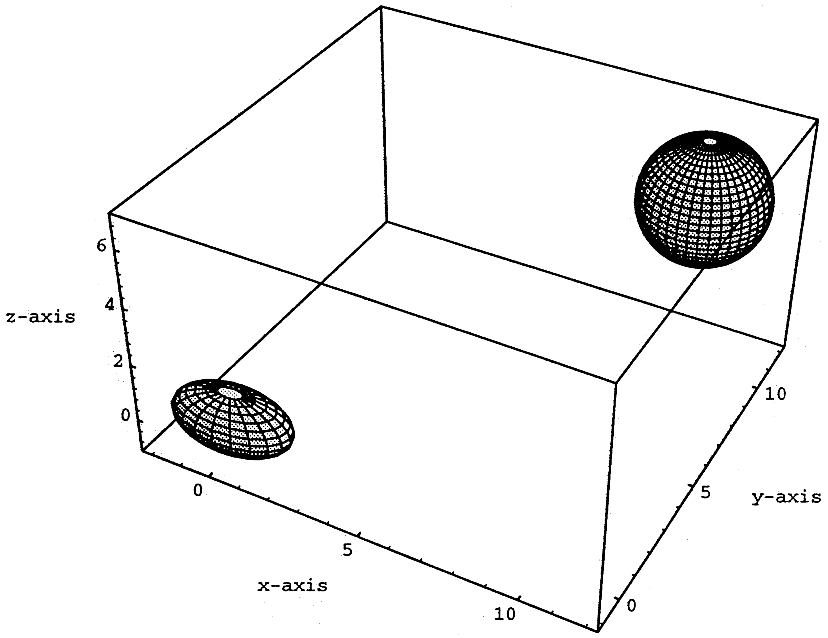

Fig. 3. Example 2.

$$p_x(t) = 10 - t, \quad p_y(t) = 10 - t^2, \quad p_z(t) = 5 - t \tag{27}$$

while the ellipsoidal object is rotating around the z-axis at a constant angular rate of 1 unit/s. It is required to determine whether a collision can occur between the two objects in $0 \leqslant t \leqslant 20$.

Notice that the ellipsoid can be described in the form (11) as

$$(x \quad y \quad z)\begin{pmatrix}1 & 0 & 0\\ 0 & 4 & 0\\ 0 & 0 & 4\end{pmatrix}\begin{pmatrix}x\\ y\\ z\end{pmatrix} \leqslant 1. \tag{28}$$

The rotation matrix around the z-axis [14] at constant angular velocity $\theta = t$ is given by

$$R(t) = \begin{pmatrix}\cos(t) & -\sin(t) & 0\\ \sin(t) & \cos(t) & 0\\ 0 & 0 & 1\end{pmatrix}. \tag{29}$$

Introducing this in Eq. (28) according to Eq. (12), we get the following representation of the ellipsoid at any time t:

$$\begin{aligned} &x^2\cos^2(t) + \frac{1}{2}xy\sin(2t) + 4x^2\sin^2(t)\\ &\quad - 2xy\sin(2t) + \frac{1}{2}xy\sin(2t) + y^2\sin^2(t)\\ &\quad - 2xy\sin(2t) + 4y^2\cos^2(t) + 4z^2 \leqslant 1. \end{aligned} \tag{30}$$

Observe also that at $t = 2\pi$, we immediately get back our original ellipsoid.

For the sphere, we need not put it in the form (12), we represent it simply as

$$(x - 10 + t)^2 + (y - 10 + t^2)^2 + (z - 5 + t)^2 \leqslant 4. \tag{31}$$

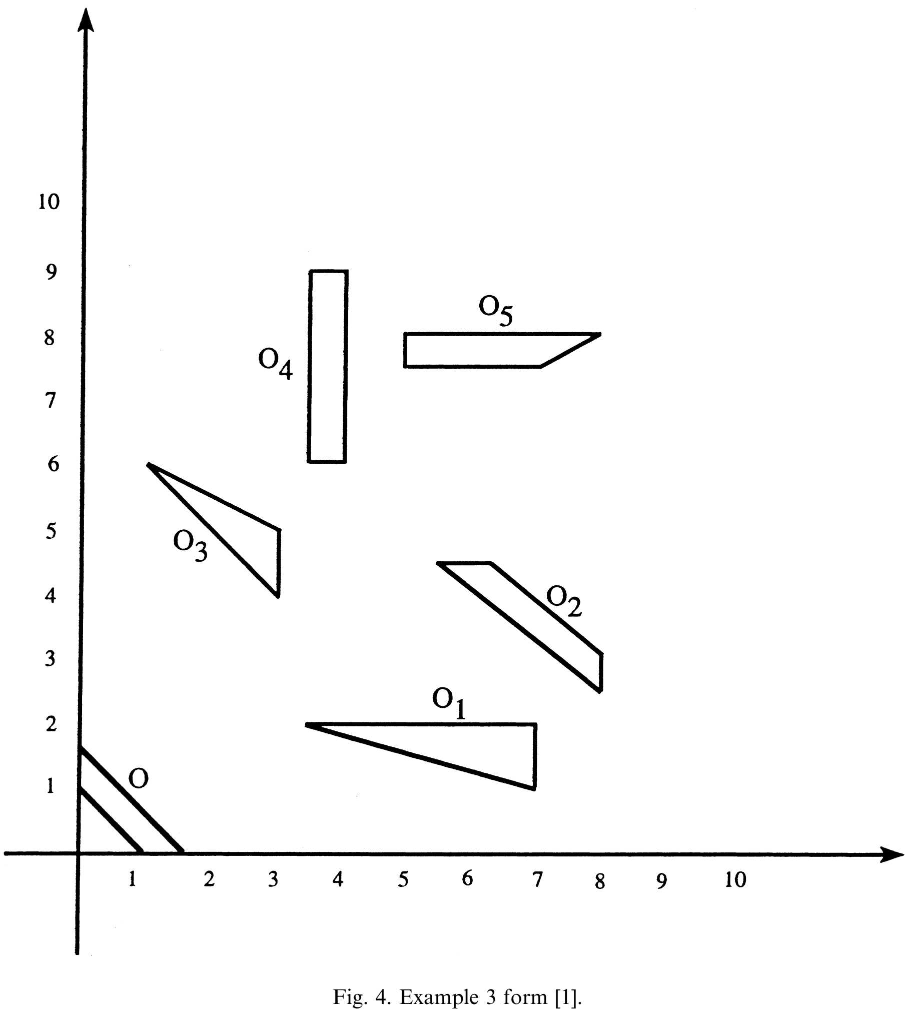

Fig. 4. Example 3 form [1].

Table 1

No. of objects	*XX*	*X*
1	0.44	2.20
2	0.77	6.20
3	1.16	9.11
4	1.48	15.77
5	1.82	21.97

X: Time in seconds for Algorithm 1 in [1] with $\Delta t = 0.5$. *XX*: Time in seconds for the new algorithm.

This problem was solved using the IBM/DX 486 machine, and no collision was detected.

Example 5.3. Our last example is a two-dimensional problem from [1] shown in Fig. 4 which we used to test the reliability of the algorithm. It was shown there that when $\Delta t \geqslant 0.6$ the object O moving on a linear path misses a collision with objects O_1, O_2, O_3. We have resolved this problem with the new algorithm. The collisions at (3.5, 2) with O_1, at (5.9, 4.5) with O_2 and at (3, 4) with O_3 were all detected within 0.2 s. Moreover, the computational time for the two algorithms is also tabulated on Table 1.

6. Conclusion

We have presented a collision detection algorithm for detecting possible collisions between moving objects in three-dimensional space. The algorithm can deal with both polyhedral and smooth objects that can be represented by systems of linear or nonlinear inequalities. Moreover, it can handle the case of an object moving on a general path (parameterized in time) in three-dimensional space with simultaneous translation and rotation. To the best of our knowledge, this is the first of its kind that can handle both polyhedral and smooth objects (without gross approximation), and with simultaneous translation and rotation. Furthermore, it detects exactly (in both space and time) the earliest possible collision between the objects without having to monitor the distance between the objects as in most of the algorithms. It does not also use the combinatorial approach applied in other algorithms. It utilizes a unique representation of objects in a four-dimensional space.

The algorithm is also efficient, because it can utilize efficient nonlinear programming routines which are available. Comparing it with some of the best algorithms on the subject, Canny [13], Gilbert et al. [16], Kawabe et al. [20], Cameron [10,11], it is seen that it supersedes all of them in terms of simplicity, efficiency and wider applicability.

Finally, it is desired to explore the possibility of using quaternions to represent orientation rather than matrices to compare the performance of the two approaches.

Acknowledgements

The authors wish to acknowledge the assistance of Mr. M. Atif Memun in preparing some of the figures, and king Fahd University of Petroleum and Minerals Dhahran, Saudi Arabia for supporting this work.

References

[1] M.D.S. Aliyu, K.S. Al-Sultan, LP-based algorithms for detecting the collision of moving objects, Journal Operational Research Society 46 (1995) 854–866.

[2] M.D.S. Aliyu, K.S. Al-Sultan, Collision detection algorithms: A survey, Technical Report CCSE-029, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, 1994.

[3] N. Ahuja, R.T. Chien, R. Yen, N. Bridwell, Interference Detection and Collision Avoidance Among Three-Dimensional Objects, First Annual National Conference on Artifical Intelligence, Stanford, 1980, pp. 44–48.

[4] A.H. Barr, Superquadrics and angle-preserving transformations, IEEE Computer Graphics and Applications (1981) 11–23.

[5] R.A. Basta, R. Mehrotra, M.R. Varanasi, Collision Detection for Planning Collision-free Motion of Two Robot Arms, Proceedings of the IEEE International Conference on Robotics and Automation, vol. 1, 1988.

[6] M.S. Bazaraa, J.J. Jarvis, H.D. Sherali, Linear Programming and Network Flows, Wiley, New York, 1990.

[7] M.S. Bazaraa, H.D. Sherali, C.M. Shetty, Nonlinear Programming: Theory and Algorithms, Wiley, New York, 1993.

[8] S. Bonner, R.B. Kelley, A representation scheme for rapid 3-D collision detection, Proceedings of the IEEE International Symposium on Intelligent Control, 1988, pp. 320–325.

[9] W.J. Boyse, Interference detection among solids and surfaces, Communication of the ACM. 22 (1979) 3–9.

[10] S.A. Cameron, A study of the clash detection problem in robotics, Proceedings of the IEEE International Conference on Robotics and Automation, 1985, pp. 488–493.

[11] S.A. Cameron, Collision detection by four-dimensional intersection testing, IEEE Transactions Robotics and Automation 6 (1990) 291–302.

[12] R.K. Culley, K.G. Kempf, A collision detection algorithm based on velocity and distance bound, Proceedings of the IEEE International Conference on Robotics and Automation, 1986, pp. 1064–1069.

[13] J. Canny, Collision detection for moving polyhedra, IEEE Trans. Pattern Anal. Machine Intell. 8 (1986) 200–209.

[14] K.S. Fu, R.C. Gonzalez, C.S.G. Lee, Robotics: Control, Sensing, Vision and Intelligence, Mcgraw-Hill, New York, 1987.

[15] R. Gallerini, On using LP to collision detection between a manipulator arm and surrounding obstacles, European Journal of Operational Research 63 (1993) 343–350.

[16] E.G. Gilbert, S.M. Hong, New algorithm for detecting the collision of moving objects, Proceedings of the IEEE International Conference on Robotics and Automation, 1989, pp. 8–14.

[17] E.G. Gilbert, D.W. Johnson, S.S. Keerthi, A fast procedure for computing the distance between complex objects in three-dimensional space, IEEE Transactions on Robotics and Automation RA- 4 (1988) 193–203.

[18] E.G. Gilbert, C.P. Foo, Computing the distance between general convex objects in three-dimensional space, IEEE Transactions Robotics and Automation 6 (1990) 53–61.

[19] V. Hayward, Fast collision detection scheme by recursive decomposition of a manipulator workspace, Proceedings of the IEEE International Conference on Robotics and Automation 1–3 (1986) 1044–1049.

[20] S. Kawabe, A. Okano, K. Shimada, Collision detection among moving objects in simulation, Proceedings of the Fourth International Symposium on Robotics Research, 1988, pp. 489–496.

[21] K.J. Kyriakopoulos, G.N. Saridis, Distance estimation and collision prediction for on-line robotic motion planning, Automatica 28 (1992) 389–394.

[22] T. Lozano-Perez, A simple motion-Planning algorithm for general robot manipulators, IEEE Transactions on Robotics and Automation RA-3 (1987) 224–237.

[23] W. Meyer, Distances between boxes: Application to collision detection and clipping, Proceedings of the IEEE International Conference on Robotics and Automation 1–3 (1986) 597–602.

[24] J.T. Schwartz, Finding the minimum distance between two convex polygons, Information Processing Letters 13 (1981) 168–170.

[25] A. Schweikard, Polynomial Time collision detection for manipulator paths specified by joint motions, IEEE Transactions on Robotics and Automation 7 (1991) 865–869.

[26] Y. Shigematsu, Y. Kakazu, N. Okino, Interference detection algorithm by simplex method, Journal of Japanese Society of Precis. Eng (Japan) 49 (11) (1983) 1561–1566.

[27] MATLAB Optimization Toolbox manual, 1992, MATHWORKS, MA.