Collision Detection for Spacecraft Proximity Operations

Robin M. Vaughan*
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California 91109
Edward V. Bergmann†
Charles Stark Draper Laboratory, Inc., Cambridge, Massachusetts 02139
and
Bruce K. Walker‡
University of Cincinnati, Cincinnati, Ohio 45221

A new collision detection algorithm has been developed for use when two spacecraft are operating in the same vicinity. The two spacecraft are modeled as unions of convex polyhedra, where the resulting polyhedron may be either convex or nonconvex. The relative motion of the two spacecraft is assumed to be such that one vehicle is moving with constant linear and angular velocity with respect to the other. Contacts between the vertices, faces, and edges of the polyhedra representing the two spacecraft are shown to occur when the value of one or more of a set of functions is zero. The collision detection algorithm is then formulated as a search for the zeros (roots) of these functions. Special properties of the functions for the assumed relative trajectory are exploited to expedite the zero search. The new algorithm is the first algorithm that can solve the collision detection problem exactly for relative motion with constant angular velocity. This is a significant improvement over models of rotational motion used in previous collision detection algorithms.

Introduction

MANY new missions now under consideration for the space program will require an orbital base such as a space station. The long-term goals for these missions require a large number of vehicles operating in the vicinity of this base. One of the concerns resulting from this environment is collision avoidance. This is one of several related issues in rendezvous and proximity operations that have been grouped under the heading of space traffic control.[1,2]

The first step in avoiding a collision between two spacecraft is the reliable detection of potential collisions. For many proximity operations, the two spacecraft are close enough that their shapes and rotational motions must be modeled along with their translational motions to predict potential collisions. This paper presents a collision detection algorithm for the simple case where one spacecraft is assumed to be moving with constant linear and angular velocity relative to the other and where the two vehicles are modeled as polyhedra. As will be seen later in the paper, combining relative rotation with relative translation between the two spacecraft greatly increases the complexity of the problem.

The theoretical basis for the new collision detection algorithm is the *C*-function formulation of the configuration space approach recently introduced by researchers in robotics and artificial intelligence.[3-6] This approach provides a global view of the problem by working directly with the constraints on the motion of one spacecraft due to the presence of the other craft. This paper first presents the fundamental ideas of configuation space theory. Three collision conditions are formulated using this theory. Next, the application of these conditions to the case of relative motion with constant linear and angular velocities is discussed. This leads directly into a description of the design and structure of the new algorithm.

Received July 12, 1989; revision received Feb. 5, 1990.

*Member of the Technical Staff, Navigation Systems Section. Member AIAA.

†Section Chief, Flight Systems. Senior Member AIAA.

‡Associate Professor of Aerospace Engineering & Engineering Mechanics. Associate Fellow AIAA.

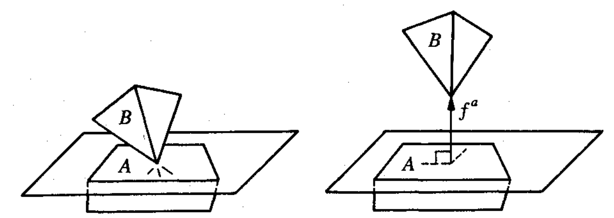
To simplify the presentation, the material in the following three sections has been restricted to the case where both spacecraft are modeled as convex polyhedra. A straightforward extension to the case where the spacecraft are modeled as nonconvex polyhedra is briefly discussed in a subsequent section. The final section of the paper gives conclusions and recommendations for possible future work.

Configuration Space Approach

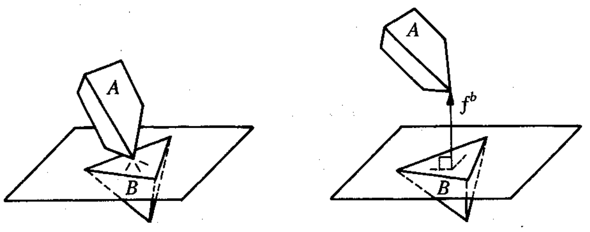
The relative motion of two spacecraft can be described by choosing a reference coordinate system such that one of the spacecraft is stationary in this system. Let the spacecraft that is stationary in the reference coordinate system be denoted B and the spacecraft that is moving in this system be denoted A. Let $\boldsymbol{x}$ represent the position of the center of rotation of A in reference coordinates and $\boldsymbol{\theta}$ represent the orientation of a body coordinate system rigidly attached to A with respect to reference coordinates. A given set of values for $\{\boldsymbol{x},\boldsymbol{\theta}\}$ is called a configuration of A. The space of all possible values of $\boldsymbol{x}$ and $\boldsymbol{\theta}$ is the six-dimensional configuration space, denoted $C_{\text{space}}(A)$. The motion of A as it translates and rotates in the original three-dimensional space is equivalent to the trajectory of a single point in $C_{\text{space}}(A)$.

The presence of B implies that there are configurations in $C_{\text{space}}(A)$ that correspond to contact of the surfaces of A and B or overlap of their interiors. The configuration space obstacle for A due to B is the set of all such configurations and is denoted $\mathcal{CO}^A_B$. The boundary of $\mathcal{CO}^A_B$ is the set of configurations where A and B have only surface points in common. The A and B are said to be just touching at these configurations because their interiors are disjoint. For configurations of A in the interior $\mathcal{CO}^A_B$, the interiors of A and B overlap. The set of configurations of A outside of $\mathcal{CO}^A_B$ is called free space.

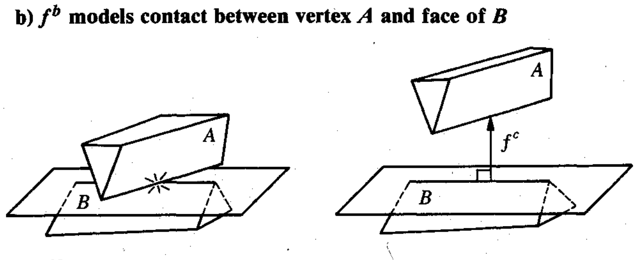
When both A and B are assumed to be convex polyhedra, $\mathcal{CO}^A_B$ can be characterized by a set of functions called *C*-functions that depend on $\boldsymbol{x}$ and $\boldsymbol{\theta}$. The *C*-functions model the three types of contact between the vertices, edges, and faces of two polyhedra illustrated in Fig. 1. A type (a) contact occurs when a face of A touches a vertex of B. The corresponding type (a) *C*-function f^a measures the distance of the vertex of B from the plane containing the face of A along its outward-pointing normal vector. Similarly, a type (b) contact occurs when a vertex of A touches a face of B. The type (b) *C*-



a) f^a models contact between face of A and vertex of B



b) f^b models contact between vertex A and face of B



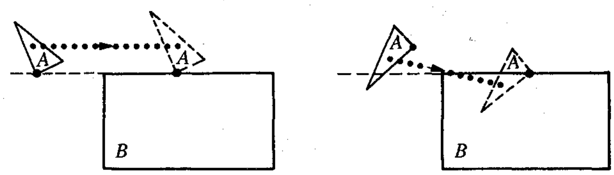
c) f^c models contact between edge of A and edge of B

Fig. 1 Three types of contact and the corresponding C-functions for polyhedra.

function f^b measures the distance of the vertex of A from the plane containing the face of B along its outward-pointing normal vector. The final type of contact, type (c), occurs when an edge of A touches an edge B. A type (c) contact is modeled by considering the plane that contains an edge of B and is parallel to an edge of A. The C-function f^c measures the distance of the edge of A from this plane along its normal vector. The magnitude of f^c is equal to the minimum distance between the edge of A and this plane, which is equivalent to the minimum distance between the lines containing the edge of A and the edge of B.

Each C-function is formulated using inner products of vectors representing the vertices, edges, and faces of A and B. Its value is zero when two of these features are in contact. Because A and B are in contact for configurations on the boundary of CO^A_B, at least one C-function must be zero at a configuration on the boundary of CO^A_B. This is the first of three collision conditions that are satisfied by configurations on the boundary of CO^A_B. This condition is necessary for a collision. However, it is not a sufficient condition for a configuration to be on the boundary of CO^A_B.

The value of a C-function can be zero at configurations that are not on the boundary of CO^A_B. One set of configurations where this occurs are those for which the interiors of A and B overlap when the two features defining the C-function are in contact. The orientation of A determines whether the interiors of A and B will overlap when two features are brought into contact. A C-function is said to be applicable for orientation θ if an arbitrary pure translation of A with fixed orientation θ can bring the feature of A and the feature of B defining this C-function into contact without overlap of the interiors of A and B. The applicability of a type (b) C-function f^b defined by a vertex of A and a face of B is illustrated in Fig. 2. The moving spacecraft labeled A is modeled as a triangular wedge, whereas the stationary spacecraft labeled B is modeled as a rectangular box. Figure 2a shows the two spacecraft at a configuration where f^b is applicable. In Fig. 2b, the orientation of A has been changed so that f^b is no longer applicable.



a) f^b is applicable b) f^b is not applicable

Fig. 2 Applicability of a type (b) C-function.

When the two spacecraft are modeled as convex polyhedra, the applicability of each C-function can be determined using a set of functions called the applicability constraint functions (ACFs). The signs of these ACFs determine whether a C-function is applicable. Consider, for example, the type (b) C-function of Fig. 2. Its ACFs are defined to be the inner products of the outward-pointing vector normal to the face of B and each of the vectors along the edges of A that meet at and point away from the given vertex. If all of these ACFs are nonnegative, then the plane containing the face of B separates A and B. Therefore, f^b is applicable for orientations θ where all of its ACFs are nonnegative. This is confirmed by the two configurations shown in Fig. 2. In Fig. 2a where f^b is applicable, all ACFs are positive or zero. However, one of the ACFs is negative for Fig. 2b, where f^b is not applicable. Similar sets of ACFs can be defined for C-functions of types (a) and (c). Convexity of A and B insures a finite set of ACFs for each C-function by restricting the number of features of A and B that must be considered.

By definition, the interiors of A and B must be disjoint at configurations on the boundary of CO^A_B. Hence, the second collision condition is that at least one C-function whose value is zero at a configuration on the boundary of CO^A_B must be applicable for that orientation of A. This condition and the first collision condition form a necessary but still not sufficient set of conditions for a configuration to be on the boundary of CO^A_B. These two conditions are not sufficient because there are zeros of a C-function that occur where it is applicable but where the two features defining it are not touching. Such a zero occurs at the configuration shown in Fig. 2a. A final condition to eliminate configurations where this type of zero occurs is developed in the following paragraphs.

When a C-function is applicable, the sign of its value can be used to determine if A and B are disjoint. This is called orienting the C-function. A C-function has positive orientation for a value of θ where it is applicable if a positive value of the C-function means that A and B cannot be in contact. All type (a) and (b) C-functions have positive orientation. Conversely, a C-function has negative orientation if a negative value of the C-function means that A and B cannot be in contact. A type (c) C-function can have either positive or negative orientation depending on the value of θ. Free space, therefore, is the set of all configurations where at least one applicable C-function with positive orientation is positive or where at least one applicable C-function with negative orientation is negative.

Although the value of one applicable C-function can determine that A and B are not in contact, the values of all applicable C-functions must be checked to determine if A and B overlap or are in contact. In other words, CO^A_B is the set of all configurations where each applicable C-function with positive orientation is nonpositive and each applicable C-function with negative orientation is nonnegative. This is the third and final collision condition. When combined with the first two collision conditions, it completes a set of necessary and sufficient conditions for a configuration of the moving spacecraft A to lie on the boundary of CO^A_B where A and B are just touching.

The collision detection problem can be viewed as the process of determining if and when a given trajectory of the configuration of A starting in free space intersects the boundary of CO^A_B for the first time. From the first collision condition, any such intersection point must lie on a five-dimensional surface in

$C_{\text{space}}(A)$ where one of the C-functions is zero. Obviously, only a part of the infinite surface where a C-function is zero can be part of the finite boundary of CO_B^A. The part of this surface forming part of the boundary of CO_B^A is determined by the second and third collision conditions. The collision detection algorithm can thus be formulated as a search along the time history of configurations of A for the first zero of a C-function where the second and third collision conditions are also satisfied.

The C-function zero search approach to the the collision detection problem is illustrated in Fig. 3. Figure 3a shows both spacecraft modeled as rectangular boxes in the original three-dimensional space of motion. The moving vehicle labeled A is shown at three different configurations along its trajectory where one of the C-functions is zero. At configuration 1, the type (b) C-function defined by the bottom face of B and the lowest vertex of A is zero. However, this C-function is not applicable because A and B lie on the same side of the plane containing the bottom face of B. This violates the second collision condition. At configuration 2, a type (a) C-function is zero and it is applicable. However, the third collision condition is violated because several other C-functions with positive orientation are positive. The first contact is a type (b) contact occurring at configuration 3 where all three collision conditions are satisfied.

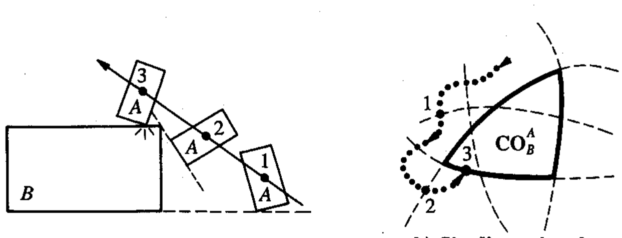
Figure 3b illustrates the corresponding situation in the six-dimensional $C_{\text{space}}(A)$. Surfaces where C-functions are zero are depicted as curved lines. The dashed portions of the lines represent configurations where either the second or third collision condition is not satisfied. The solid portions of the lines represent the boundary of CO_B^A where the second and third collision conditions are both satisfied. The trajectory of A in this space is drawn as a dotted line. Configurations 1, 2, and 3 occur at the intersections of this dotted line with one of the curved lines. Although the trajectory intersects several surfaces where a C-function is zero, the intersection at configuration 3 is the first to fall on the boundary of CO_B^A. This represents the first contact between the two spacecraft as shown in Fig. 3a.

Collision Detection for Constant Linear and Angular Velocities

The C-functions and ACFs have been presented as functions of the configuration of the moving spacecraft A. In order to formulate the collision detection alogrithm as a search for the zeros of the C-functions, the time dependence of these functions must be known. For relative motion with constant linear and angular velocities, all three types of C-functions share a common form as functions of t. This form will be referred to as the generalized C-function $f(t)$ where

$$f(t) = (m_1 t + l_1) + (m_2 t + l_2)\cos(\omega t) + (m_3 t + l_3)\sin(\omega t) \tag{1}$$

The six coefficients m_1, m_2, m_3, l_1, l_2, and l_3 are scalar constants that depend on the vectors representing particular



a) Three-dimensional space of motion **b) Six-dimensional $C_{\text{space}}(A)$**

Fig. 3 Collision detection by searching for zeros of the C-functions.

features of A and B, the direction of the angular velocity vector ω, and the linear velocity vector v. The scalar constant ω is the magnitude of the angular velocity vector.

The form of $f(t)$ is crucial to the structure of the collision detection algorithm. Because $f(t)$ is a transcendental function of t, no analytic formulas exist for the zeros of $f(t)$. Instead, numerical procedures must be developed to iterate for the zeros of individual C-functions. The use of these numerical zero search procedures will be practical only if the search can be confined to bounded intervals of time. It can be shown, however, that the intervals of time where it is mathematically possible for $f(t)$ to be zero are unbounded for most values of the six coefficients.[7] Fortunately, additional techniques for restricting the zero search intervals can be developed using the second and third collision conditions.

The second collision condition limits the zeros of a C-function that can represent valid contacts to those that occur at orientations where the C-function is applicable. For the case where A is rotating with constant angular velocity, the ACFs for each type of C-function all have the form

$$g(t) = g_1 + g_2\cos(\omega t) + g_3\sin(\omega t) \tag{2}$$

The ACFs are, therefore, periodic functions of t, and it is only necessary to examine the signs of the ACFs over a single cycle of ωt to determine the intervals where each C-function is applicable. These intervals can be found in closed form because $g(t)$ is merely a shifted sinusoid. If there are no intervals in one cycle of ωt where a particular C-function is applicable, then no zero search is required for that C-function. However, if any applicability intervals of a C-function exist, they will occur in each cycle of ωt and consequently do not yield bounded intervals for the zero search.

The third collision condition specifies the sign of an applicable C-function that may indicate contact between the two spacecraft. The relative translational motion of the two vehicles can be used to find the intervals of time, if any, where all applicable C-functions can have the appropriate signs indicating that contact is possible. Since translation with constant linear velocity is not cyclic in t, this yields a finite interval of time where a collision may occur. Outside of this bounded interval, at least one of the applicable C-functions will always have a sign that indicates no contact between A and B. Such a finite interval $[T_1, T_2]$ is found by determining all values of t where two spheres enclosing A and B overlap. For constant linear velocity, $[T_1, T_2]$ includes all values of t where a quadratic function of t is nonpositive.

Similar reasoning about the relative translational motion of the two spacecraft can be used to find intervals of time where a particular C-function can have a sign indicating possible contact between particular features of the two spacecraft. As an example, refer to the two rectangular boxes shown in Fig. 3a. For the linear trajectory shown in the figure, there will be some time t_0 prior to configuration 1 where the perpendicular distance from the centroid of A to the plane containing the bottom face of B will become less than the distance from the centroid to any of the vertices of A. For $t < t_0$, the position of the centroid of A is such that none of the vertices of A can touch the plane containing the bottom face of B no matter how A is oriented. A type (b) C-function defined by the bottom face of B and any vertex of A will always be positive in the interval $(-\infty, t_0)$. If the orientation of A at some time t in this interval is such that the C-function is also applicable, then contact between A and B is not possible at time t. A type (b) C-function defined by the bottom face of B and a vertex of A can be nonnegative, indicating possible contact between A and B only when $t \geq t_0$.

A constraint such as the one described earlier for a type (b) C-function is called a physical constraint. A physical constraint defines the values of t where the position of A is such that the feature defining the C-function could touch the feature of B defining the C-function for some orientation of A. The presence of the other features of A and B is ignored in finding this region. The physical constraint for each type of C-function is fully described in Ref. 7. The search for zeros of the C-function is restricted to the intervals of overlap, if any, between $[T_1, T_2]$, the intervals where its physical constraint is satisfied, and its cyclic applicability intervals.

Now that bounded intervals have been defined in which to search for the zeros of each C-function, some means of locating the zeros of $f(t)$ in these intervals must be developed. A reliable zero search procedure begins by creating a sequence of test points consisting of the bounds of the search interval and any values of t within the interval where f' or f'' equals zero. The value of $f(t)$ is then computed at each of the test points. If $f(t)$ changes sign between any two consecutive test points, an iteration is performed to locate the zero of $f(t)$ between the two points. Because the test points include relative maxima and minima of $f(t)$, only one zero of $f(t)$ can be located between any two consecutive test points where $f(t)$ changes sign. This guarantees that no zeros of $f(t)$ are overlooked.

The methods for finding values of t where f' and f'' equal zero depend on the values of the six coefficients of $f(t)$. When all six coefficients are nonzero, the second derivative of $f(t)$ has the form

$$f'' = (a_1 t + b_1)\cos(\omega t) + (a_2 t + b_2)\sin(\omega t) \tag{3}$$

where a_1, a_2, b_1, and b_2 are functions of m_1, m_2, m_3, l_1, l_2, and l_3. A careful analysis of this equation gives formulas for finding the bounds of intervals that must contain a single point where $f'' = 0$. The details of this analysis can be found in Ref. 7. Specialized methods for 11 other cases corresponding to one or more zero-valued coefficients of $f(t)$ are also presented in this reference.

The C-function zero search procedure is divided into three stages. The first stage finds the points in the search interval where the second derivative is zero. The results of this stage are used in the second stage to find the points in the search interval where the first derivative is zero. The third and final stage uses the results of the first two stages to find the zeros of $f(t)$ itself in the search interval. In all three stages, a standard method, such as linear interpolation or the secant method, is used to iterate for the zeros of the function or one of its first two derivatives.

Collision Detection Algorithm

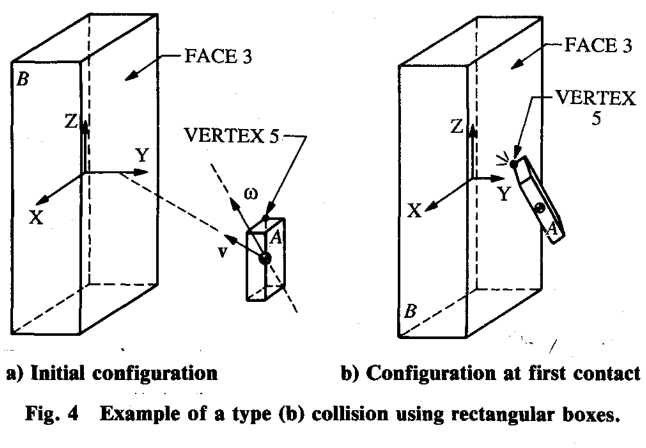
The collision detection algorithm applies the techniques discussed in the preceding section in a hierarchy that represents more complex, but more accurate, modeling of the interaction of the two spacecraft. First, the overall interval $[T_1, T_2]$ where a potential contact may occur is found by considering the relative motion of two spheres enclosing the two spacecraft. Each C-function is then examined individually in this interval. The object of this examination is to limit the zero search as much as possible because of the complexity of the numerical iteration procedures.

The examination of a single C-function $f(t)$ is a two-stage process. In the first stage, checks are made to determine if there are any subintervals of $[T_1, T_2]$ that may contain zeros of $f(t)$ that can be valid collisions between the spacecraft. First, the applicability of $f(t)$ is checked. If there are no intervals of ωt where it is applicable, then no zero search is necessary. If there are intervals of ωt where $f(t)$ is applicable, the physical constraint for $f(t)$ is checked. If there are no subintervals of $[T_1, T_2]$ where its physical constraint is satisfied and where it is applicable, then no zero search is necessary. Otherwise, the algorithm moves on to the second stage where the zeros of $f(t)$ are found. The appropriate iteration is performed in those subintervals of $[T_1, T_2]$ where $f(t)$ simultaneously satisfies its applicability criteria and its physical constraint. The first zero found in one of these subintervals where the third collision condition is also satisfied is stored as the current value for the time of first contact. The type of contact is also stored. The time and type of first contact are updated when necessary as the algorithm examines each of the remaining C-functions.

As an example, let the two spacecraft be modeled by rectangular boxes, as shown in Fig. 4. Figure 4a shows the initial configuration of the two boxes along with the linear and angular velocity vectors. The moving spacecraft A is translating toward the center of a face of the stationary spacecraft B. Because A is also rotating while it translates, the predicted collision is a type (b) contact between a vertex of A and the nearest face of B, as shown in Fig. 4b. In order to find the particular vertex of A that first touches the face of B, the collision detection algorithm examines each of the 240 C-functions defined by the vertices, edges, and faces of the two boxes. Of these 240 functions, only two remain after the zero search restrictions are imposed in the first stage of the examination. This is a substantial reduction in the number of C-functions for which the second stage numerical iterations are required. The algorithm eliminates one of the two remaining C-functions by identifying the one that has the earliest zero where the third collision condition is satisfied. As shown in Fig. 4, this occurs when vertex 5 of A touches face 3 of B.

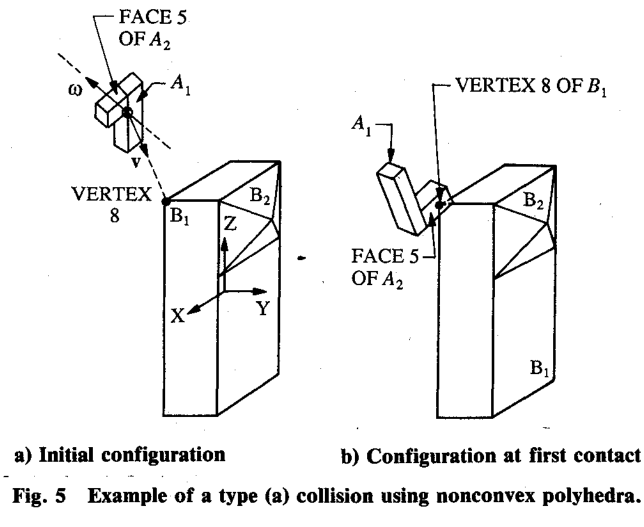
Extension to Nonconvex Polyhedra

The basic collision detection algorithm applies only to cases where A and B are both single convex polyhedra. It can be easily extended to the case where either spacecraft is a nonconvex polyhedron. In this case, each of the two vehicles is broken up into a set of convex polyhedra. The extended algorithm considers each pair of convex polyhedra, one from



a) Initial configuration b) Configuration at first contact

Fig. 4 Example of a type (b) collision using rectangular boxes.



a) Initial configuration b) Configuration at first contact

Fig. 5 Example of a type (a) collision using nonconvex polyhedra.

A and one from B, separately. The C-functions from each pair are examined to determine if and when these two convex polyhedra first come into contact. The time of first contact for A and B is the earliest time of contact between a pair of their constituent convex polyhedra.

An example where each spacecraft is modeled by a nonconvex polyhedron is illustrated in Fig. 5. The A is an L-shaped polyhedron formed by two rectangular boxes A_1 and A_2, and B consists of rectangular box B_1 with pyramid B_2 attached to one of its faces. Figure 5a shows the initial configuration of the two polyhedra and the linear and angular velocity vectors of A. The trajectory of A is such that the first contact is most likely to occur between vertex 8 of B_1 and one of the faces of A_1 or A_2. This is confirmed by the collision detection algorithm that finds the first contact to be a type (a) contact between vertex 8 of B_1 and face 5 of A_2, as shown in Fig. 5b.

The algorithm examines 812 C-functions for the two nonconvex polyhedra in this example. The two pairs of rectangular boxes, $\{A_1, B_1\}$ and $\{A_2, B_1\}$, contribute 240 C-functions each. The two pairs of a box from A and the pyramid from B, $\{A_1, B_2\}$ and $\{A_2, B_2\}$, contribute 166 C-functions each. The larger number of C-functions to be examined for this nonconvex example causes the algorithm to require more computation time than for the convex example. The increase in computation time is not directly proportional to the increase in the number of C-functions because it depends on the order in which the algorithm examines pairs of convex polyhedra and the C-functions associated with each pair.

The simple extended algorithm used for the preceding example examined all C-functions defined by each pair of convex polyhedra. Some computation time could be saved by taking into account the way in which the convex polyhedra are connected to form the overall spacecraft. For example, any C-functions defined by the square face of the pyramid B_2 need not be examined because this face cannot touch A without overlap of the interiors of the two spacecraft. This is one of many algorithmic improvements that should be considered for future applications.

Conclusions and Recommendations

The new collision detection algorithm has been designed as a search for zeros of functions that indicate contact between the vertices, faces, and edges of polyhedra representing the two spacecraft. The feasibility of this approach is determined by the model for the relative motion of the two spacecraft, which, in turn, determines the form of the functions as functions of time. This paper has presented several new techniques that allow the algorithm to handle the transcendental form of these functions resulting from relative motion with constant linear and angular velocities.

The new algorithm has several attractive features. It is the first algorithm that can predict an exact value for the time and type of first contact when one spacecraft moves with constant linear and angular velocity relative to the other. In particular, the ability to handle rotation with constant angular velocity is a significant improvement on previously available collision detection algorithms. Moreover, the new algorithm is guaranteed to detect a collision, if one will occur, to within the accuracy of the computer's calculations.

Many areas remain to be pursued in future work on collision detection and avoidance. The efficiency of the detection algorithm presented in this paper could be improved by developing additional criteria for the order in which pairs of convex polyhedra are considered and in which the vertices, edges, and faces from each pair are examined. More sophisticated algorithms could be studied that use different models for the shapes and relative trajectory of the two spacecraft. Another interesting subject for future study is the integration of the detection algorithm into an overall system for collision avoidance. Some issues that should be considered are the degree of automation of the system, the design of maneuvers to avoid predicted collisions, and the means of validating that a collision has been successfully avoided.

Acknowledgments

The research described in this paper was carried out at the Charles Stark Draper Laboratory, Inc., under NASA Contract NAS9-17560.

References

[1]Young, K., and Bell, J., "Rendezvous and Proximity Operations–Mission Operations—Flight Design Perspective," *Proceedings of the Rendezvous and Proximity Operations Workshop*, NASA Johnson Space Center, Houston, TX, Feb. 1985.

[2]"Autonomous Operation," Space Station Work Package 2 Meetings, McDonnell Douglas Aircraft Corp., NASA Johnson Space Center, Houston, TX, Dec. 1985.

[3]Lozano-Perez, T., "Spatial Planning with Polyhedral Models," Ph.D. Dissertation, Dept. of Electrical Engineering and Computer Science, Massachusetts Inst. of Technology, Cambridge, MA, 1980.

[4]Lozano-Perez, T., "Spatial Planning: A Configuration Space Approach," *IEEE Transactions on Computers*, Vol. C-32, No. 2, 1983, pp. 108–120.

[5]Donald, B. R., "Motion Planning with Six Degrees of Freedom," Massachusetts Inst. of Technology Artificial Intelligence Lab., Cambridge, MA, TR-791, 1984.

[6]Canny, J., "Collision Detection for Moving Polyhedra," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 2, 1986, pp. 200–209.

[7]Vaughan, R. M., "Collision Detection for Spacecraft Proximity Operations," Ph.D. Dissertation, Dept. of Aeronautics and Astronautics, Massachusetts Inst. of Technology, Cambridge, MA, 1987.