Proceedings of the 2006 IEEE International Conference on Robotics and Automation
Orlando, Florida - May 2006

Collision Detection of Cylindrical Rigid Bodies for Motion Planning

John Ketchel
Department of Mechanical and Aerospace Engineering
Florida Institute of Technology
Melbourne, Florida 32901–6975
Email: jketchel@fit.edu

Pierre Larochelle
Department of Mechanical and Aerospace Engineering
Florida Institute of Technology
Melbourne, Florida 32901–6975
Email: pierrel@fit.edu

***Abstract*— This paper presents a novel methodology for detecting collisions of cylindrically shaped rigid bodies moving in three dimensions. This algorithm uses line geometry and dual number algebra to exploit the geometry of right circular cylindrical objects to facilitate the detection of collisions. First, the rigid bodies are modelled with infinite cylinders and an efficient necessary condition for collision is evaluated. If the necessary condition is not satisfied then the two bodies do not collide. If the necessary condition is satisfied then a collision between the bodies may occur and we proceed to the next stage of the algorithm. In the second stage the bodies are modelled with finite cylinders and a definitive necessary and sufficient collision detection algorithm is employed. The result is a straightforward and efficient means of detecting collisions of cylindrically shaped bodies moving in three dimensions. This methodology has applications in robot motion planning, spatial mechanism design, and workspace analysis of parallel kinematic machines such as Stewart-Gough platforms. A case study of motion planning for an industrial robot is included.**

I. Introduction

We present a two stage algorithm for determining if two bodies moving in collide. In the first stage, infinite length cylinders are used to model the objects, then line geometry is used to determine if the cylinders intersect. If these infinite cylinders do not intersect then the two bodies do not collide and no further testing is required. If the two infinite cylinders do intersect then further testing is necessary. We proceed to the second stage where cylinders of finite length are used to model the objects and they are tested to determine quantitatively if they collide.

The methodology presented here is general and can be used to detect collisions between rigid bodies moving in three dimensions provided that the bodies are predominantly cylindrical in shape. We focus upon such bodies because this shape is commonly found in industrial robots and parallel kinematic machines (e.g. Stewart-Gough platforms).

The paper proceeds as follows. First, an efficient necessary condition for collisions between infinite cylinders is presented. Next, a similar necessary condition for the collision of finite length cylinders is shown. Then a detailed algorithm is presented via flowcharts for detecting collisions. Finally, a case study illustrating the application of this collision detection algorithm to the motion planning of an industrial robot is presented.

II. Related Works

Collision detection is vital for real world implementation of robotic mechanical systems. Collision detection assists in motion planning, real-time control, digital prototyping and motion simulation of these systems. Zsombor-Murray [5] presents the visualization of the shortest distance between two lines in space. His constructive geometry and algebraic solutions to the problem motivated the work proposed here. [4] correctly states that failure to detect a collision is less acceptable than false positives, which can be further checked and that for the sake of speed exact or accurate collision detection is often sacrificed.

A great amount of prior work has been done on the collision detection problem that has resulted in the development of many software packages such as: VEGAS[9], V-Clip[7], RAPID[7], SOLID[7], I-Collide[6], V-Collide[8] and PQP[7]. They vary in their modelling and in mathematical method for determining if a collision has occurred.

III. COLLISION DETECTION

A. Infinite Cylinder Testing

Infinite and finite length cylinders are used to model rigid bodies in three dimensions. Initially, each object is modelled by a cylinder of infinite length and finite radius. Infinite cylinders are simple models to check for collisions since they can be represented as a line in space with a radius. The shortest distance between two lines in space is along their common normal line N. An advantage of using cylinders is that their common normal line has a finite line segment between the two cylinders and by subtracting the two cylinders radii from the segment, possible collisions can be detected. If the distance between the two cylinders is less than the sum of their two radii then the infinite cylinders have collided. Hence, if the actual finite cylindrical objects have collided it is *necessary* that the minimum distance between their associated infinite cylinders be less than the sum of their radii.

We use Plücker coordinates and dual vectors to represent the major axis of an infinite cylinder [3]. For example line S_1 can be defined by points $\vec{c}$ and $\vec{f}$ or point $\vec{c}$ and direction vector $\vec{s}$ (see fig. 1 and eqs. 1 & 2).

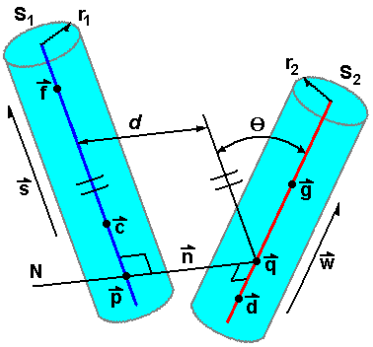

Fig. 1. Infinite Cylinders

$$S_1 = \left(\frac{\vec{f}-\vec{c}}{\|\vec{f}-\vec{c}\|}, \vec{c}\times\frac{\vec{f}-\vec{c}}{\|\vec{f}-\vec{c}\|}\right) = (\vec{s}, \vec{c}\times\vec{s}) \quad (1)$$

$$S_2 = \left(\frac{\vec{g}-\vec{d}}{\|\vec{g}-\vec{d}\|}, \vec{d}\times\frac{\vec{g}-\vec{d}}{\|\vec{g}-\vec{d}\|}\right) = (\vec{w}, \vec{d}\times\vec{w}) \quad (2)$$

We use the dual vector representation of the lines and dual vector algebra ([1] and [3]) where ϵ^2=0.

$$\hat{S}_1 = (\vec{s}, \vec{c}\times\vec{s}) = a + \epsilon a^0 \quad (3)$$

$$\hat{S}_2 = (\vec{w}, \vec{d}\times\vec{w}) = b + \epsilon \vec{b}^0 \quad (4)$$

Line dot product:

$$\hat{S}_1 \cdot \hat{S}_2 = (a, a^0)\cdot(b, b^0) = \cos\theta - \epsilon d\sin\theta = \cos\hat{\theta} \quad (5)$$

Line cross product:

$$\hat{S}_1 \times \hat{S}_2 = (a, a^0)\times(b, b^0) = (\sin\theta + \epsilon d\cos\theta)\hat{N} = \sin\hat{\theta}\hat{N} \quad (6)$$

where $\hat{N}$ is the common normal line to $\hat{S}_1$ and $\hat{S}_2$. The above operations are useful for calculating the distance d and the angle θ between two lines. The resultant dual number of the dot product of two dual vectors yields the angle and distance between the two lines as long as they are not parallel to each other (see eq. 5). If $d\sin\theta \neq 0$ then the lines do not intersect ($d \neq 0$) and are not parallel ($\sin\theta \neq 0$). If $d\sin\theta = 0$ and $\cos\theta \neq 1$ then the lines intersect ($d = 0$) and are not parallel. If the $\cos\theta$ term of the dot product is equal to 1 then the lines are parallel and the resultant dual vector of the cross product will have a 0 real component. The cross product's dual component ($d\cos\theta$) will be 0 when the lines are identical. If $d\cos\theta \neq 0$ then the distance d can be calculated. Figure 2 shows a detailed flow chart of the necessary condition for a collision. If the necessary condition is satisfied then additional testing is required.

This is an efficient method of determining the distance between the two lines and if a possible collision has occurred.

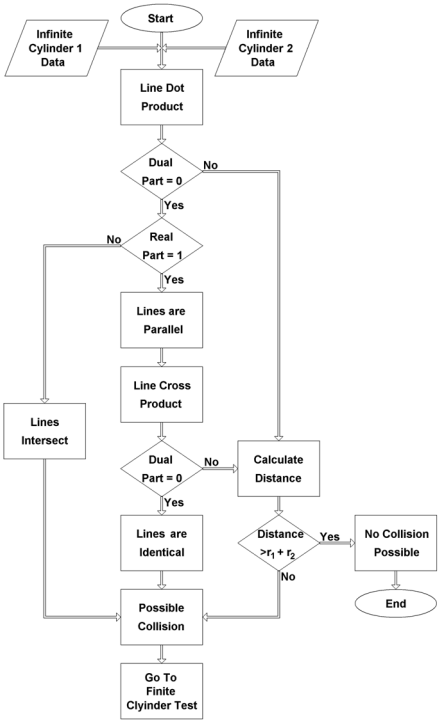

Fig. 2. Infinite Cylinder Testing

If the resulting distance is greater than the sum of the two radii then no collision is possible regardless of the length of the finite cylinders. If the result is not greater than the sum of the two radii then a collision *may have* occurred. A finite cylinder model is used in the next stage of the collision detection algorithm to determine if indeed a collision *has* occurred.

B. Finite Cylinder Testing

If a possible collision has been detected by the infinite cylinder test then further testing is required to determine if an actual collision has occurred. The model is modified from cylinders of infinite length to cylinders of finite length. This changes the approach from testing lines to testing line segments. However, the same idea applies that the shortest distance between the cylinders is along their common normal but the point where the common normal intersects the cylinder's axis now becomes important. Figure 3 shows a detailed flow chart of the initial testing of the finite cylinders.

1) Parallel Testing: From the initial testing, the angle θ and distance d between the infinite cylinders is known. The distance between the cylinders was calculated during the infinite cylinder testing and found to be less than the sum of their radii. If the finite cylinders are parallel then there are two general cases: their associated line segments overlap in some manner or there is no overlap. It is necessary for the finite model to determine if they overlap. To determine if the segments overlap the plane that is orthogonal to the lines and passes through one endpoint is determined. The intersection point of this plane with the other line is computed. This point is then tested to see if it is on the line segment or not. This

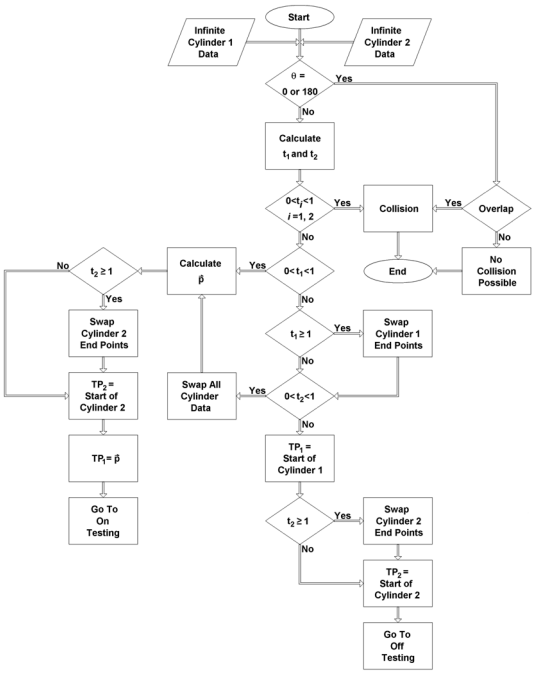

Fig. 3. Finite Cylinder Testing

is repeated for up to two other endpoints (a fourth endpoint being redundant) to test for overlapping. When no overlapping occurs, no collision is possible.

2) Non-Parallel Testing: To facilitate the testing of the non-parallel finite cylinders we require that the point of intersection of their common normal line within or before both cylinders. If either cylinder's point does not lie on or before it, then the endpoints of the finite cylinder are interchanged. The points of intersection of the common normal line with the cylinder axis determines the shortest distance between the two lines in space.

We begin by determining where the common normal line intersects the axis of each cylinder. The axes are described by their endpoints ($\vec{c}$ and $\vec{d}$) and *non*-unit direction vectors ($\vec{s}$ and $\vec{w}$), where $\|\vec{s}\|$ and $\|\vec{w}\|$ are equal to the length of their corresponding cylinder (see fig. 4). The common normal line N intersects the lines S_1 and S_2 at points $\vec{p}$ and $\vec{q}$ respectively. Parametric equations for points $\vec{p}$ and $\vec{q}$ of lines S_1 and S_2 are:

$$\begin{aligned} \vec{p} &= \vec{c} + t_1\vec{s} \\ \vec{q} &= \vec{d} + t_2\vec{w} \end{aligned} \tag{7}$$

where

$$\begin{aligned} t_1 &= \frac{[(\vec{d}-\vec{c})\times\vec{w}]\cdot\vec{n}}{\vec{n}\cdot\vec{n}} \\ t_2 &= \frac{[(\vec{d}-\vec{c})\times\vec{s}]\cdot\vec{n}}{\vec{n}\cdot\vec{n}} \\ \vec{n} &= \vec{s}\times\vec{w} \end{aligned}$$

Next for each cylinder we determine the test point along it's axis, on or within the finite cylinder, that is closest to the common normal line. These test points are referred to as TP_1 and TP_2. Then we determine if common normal points, $\vec{p}$ and $\vec{q}$, are on the segments, before the segments or after the segments. If $t_1 \leq 0$ then $\vec{p}$ lies at the start of the segment or earlier, so the start point of the cylinder is used as TP_1. If $t_1 \geq 1$ then $\vec{p}$ lies at the end of the segment or further, so the end point is used as TP_1. If $0 < t_1 < 1$ then $\vec{p}$ lies on the line segment so $\vec{p}$ can be used as TP_1. The above procedure is repeated for cylinder 2 to determine TP_2.

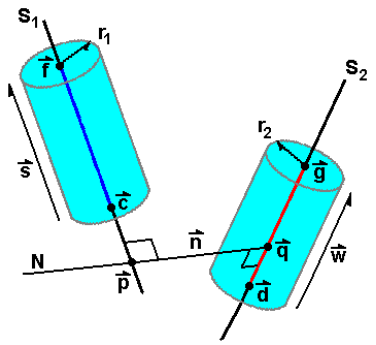

Fig. 4. Finite Cylinders

From the determination of whether points $\vec{p}$ and $\vec{q}$ lie on or off the cylinders there are three possible cases to consider. If $0 \leq t_i \leq 1$, where $i = 1, 2$, both $\vec{p}$ and $\vec{q}$ lie on the segments and On-On testing is necessary. If either but not both $\vec{p}$ and $\vec{q}$ lie on the segments, On testing is necessary. Additionally, if only one cylinder is On, the testing requires that it is cylinder 1. If it is cylinder 2 that is On, all cylinder 1 and 2 data are swapped. If neither $\vec{p}$ nor $\vec{q}$ lie on the segments, Off testing is necessary. These cases are discussed below.

3) Case 1 - On-On Testing: If both points $\vec{p}$ and $\vec{q}$ lie on the finite cylinders then a collision has occurred and no further testing is required. There is no need to test the distance between the points since on the infinite level of testing this distance was already checked.

4) Case 2 - On Testing: Figure 5 shows a detailed flow chart of the finite cylinder On test procedure. We begin by finding the closest point along cylinder 1's axis to TP_2 (see fig. 6). This point, $\vec{p}_1$, is the intersection of lines S_1 and N_1 (N_1 is orthogonal to S_1 and passes through TP_2). Calculating $\vec{p}_1$ (see eqs. 8) yields t_3 which is used to determine if $\vec{p}_1$ lies on or off the cylinder. If the distance from TP_2 to $\vec{p}_1$ is greater than the sum of the radii then no collision is possible and no further testing is required. Otherwise, the point TP_2', which is the closest point of cylinder 2 to cylinder 1's axis. TP_2' is on the circular end of cylinder 2 and is found through an direct search approach discussed at the end of this section. The iteration returns t_3', TP_2' and $\vec{p}_1'$ where $\vec{p}_1'$ is the closest point along cylinder 1's axis to TP_2'. Calculating $\vec{p}_1'$ (see eqs. 10) yields t_3' which is used to determine if $\vec{p}_1'$ lies on or off the cylinder. If $0 < t_3' < 1$ then $\vec{p}_1'$ lies in cylinder 1 and if the distance between TP_2' and $\vec{p}_1'$ is less than the radius of cylinder 1 then a collision has occurred. If the distance is greater than the radius, then no collision can occur and further testing is not required. If t_3' is outside the range, the test point of cylinder 1 has slid off the end of the cylinder and end testing is required

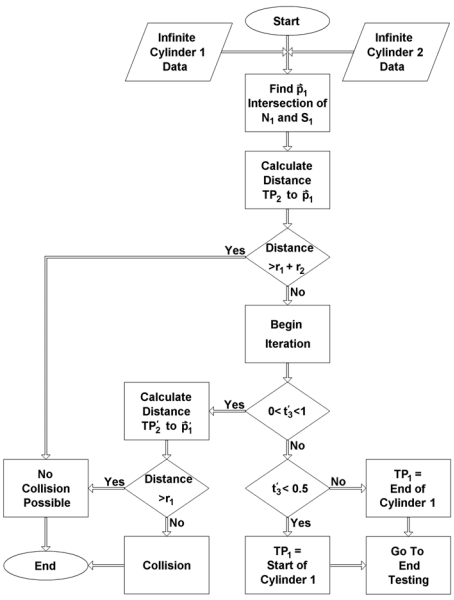

Fig. 5. On Testing

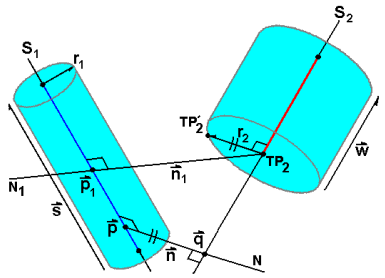

Fig. 6. On Testing Labels

(see fig. 7).

$$\vec{p}_1 = \vec{c} + t_3\vec{s} \quad (8)$$

$$\vec{s} \cdot \vec{p}_1 = \vec{s} \cdot TP_2$$

$$TP_2' = TP_2 + [rot] * r_2 * [cos\theta, sin\theta, 0] \quad (9)$$

$$[rot] = [\vec{x}, \vec{y}, \vec{z}]^T$$

$$\vec{p}_1' = \vec{c} + t_3'\vec{s} \quad (10)$$

$$\vec{s} \cdot \vec{p}_1' = \vec{s} \cdot TP_2'$$

Direct Search: Direct search is used to determine the point TP_2' along the starting edge of cylinder 2 that is closest to the axis of cylinder 1. A start point on the edge of the cylinder is found for the two cases, wether the axis are perpendicular to each other or not. If the cylinder's axes are perpendicular then the start point is found by moving the distance of the radius of cylinder 2 from TP_2 along $\vec{n}$. If the axes are not perpendicular the direction of the translation is found by calculating the point of intersection of the plane that is perpendicular to $\vec{s}$ through TP_2 with S_1. The start point for the search is used to define a local set of coordinate axes that will be used for calculating subsequent TP_2's. The X-axis is defined as the unit vector from

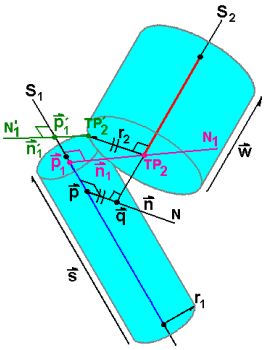

Fig. 7. Projecting Off

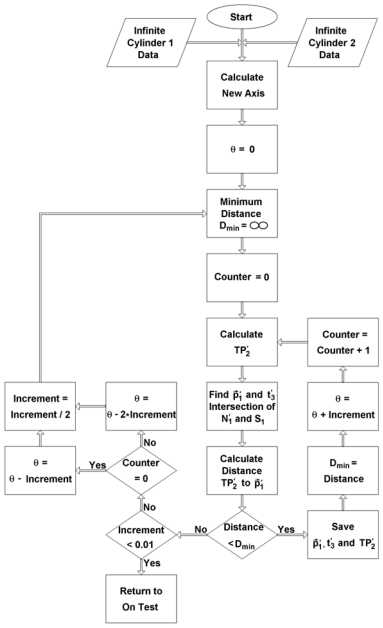

Fig. 8. TP_2' Search

TP_2 to TP_2'. The Z-axis is the unit vector along the axis of cylinder 2. To complete the right hand frame, the Y-axis is the cross product of the Z and X axes, see fig. **??**. The new coordinate system is used to calculate the TP_2's, see equation 9, that will be tested during the search. A direct search with respect to θ is performed to determine the point TP_2' nearest p_1'.

5) Case 3 - Off Testing: The distance from TP_2 to S_1 is found. If the distance is greater than the sum of the radii then no collision is possible since TP_2 is the closest point on cylinder 2 to cylinder 1. If the distance is not greater than the sum of the radii then On testing must be performed. Additionally, the other end of cylinder 2 must be similarly tested. The cylinder data is swapped and the Off testing is repeated with the new data.

6) End Testing: End cylinder testing is necessary when the circular ends of the cylinders may intersect [2]. The first step is to find the equations of the planes, π_1 and π_2, that are orthogonal to each cylinder's axis and pass through their test

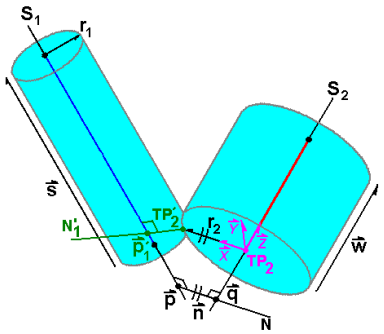

Fig. 9. TP_2' Local Coordinate Frame

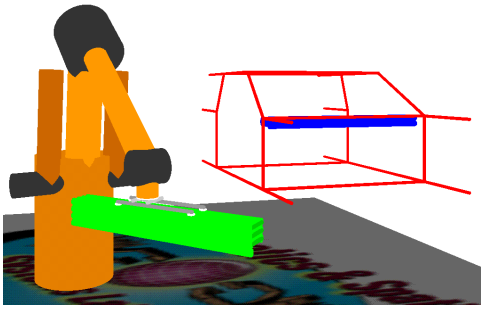

Fig. 10. Work Cell Model

points. Then the parametric equation for the line of intersection N' of π_1 and π_2 is found. Next the distance from TP_1 to N' is set to r_1 to obtain a quadratic in t_4:

$$\begin{aligned} 0 \quad = \quad & t_4^2(n_x^2+n_y^2+n_z^2)+t_4(2n_x(x-TP_{1x}) \\ & +2n_y(y-TP_{1y})+2n_z(z-TP_{1z})) \\ & +((x-TP_{1x})^2+(y-TP_{1y})^2+(z-TP_{1z})^2-r_1^2) \end{aligned} \tag{11}$$

Similarly, t_5 can be found by setting the distance from TP_2 to N' as r_2 (see eqs. 12).

$$\begin{aligned} 0 \quad = \quad & t_5^2(n_x^2+n_y^2+n_z^2)+t_5(2n_x(x-TP_{2x}) \\ & +2n_y(y-TP_{2y})+2n_z(z-TP_{2z})) \\ & +((x-TP_{2x})^2+(y-TP_{2y})^2+(z-TP_{2z})^2-r_2^2) \end{aligned} \tag{12}$$

If the quadratic yields complex roots then the cylinder end circle and the line N' do not intersect and no collision is possible. Repeated roots from the quadratic formula mean that the cylinder end circle is tangent to the line N' and no collision has occurred. If both roots are real then a range is found for t_4 and t_5 and if these ranges overlap then a collision has occurred.

IV. Case Study

We demonstrate the methodology presented to a robotic work cell (fig. 10) for a pick and place application. The robot picks-up a dashboard and has to maneuver it around, then into, an auto body and place it in its mounting location. The robot then retraces its motion out of the auto back to the pickup location to begin the next cycle.

The model of the work cell includes a robot, a tool, a workpiece (dashboard) and the drop location (automobile frame). The robot selected for the case study is the ABB6000 (fig. 11) and it is modelled by 9 cylinders shown in shades of orange and grey of varying radii. The tool is also modelled by 9 cylinders shown in shades of light grey of varying radii. The dashboard to be installed is modelled by 15 green cylinders of varying radii. The drop-off location is modelled by 25 red and blue cylinders.

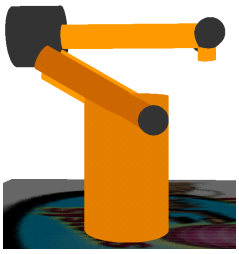

Fig. 11. ABB 6000: Photo [10] & Model

It is necessary that at each increment of motion planning that every cylinder is tested for a possible collision with every other cylinder. Since speed is important for the calculations, we look to reduce the number of tests that must be run for each incremental step. First, cylinders can not collide with themselves since they are rigid links. Second, if cylinder 5 has already been tested for a collision with cylinder 1 then testing cylinder 1 for a possible collision with cylinder 5 is redundant. The number of tests required here for each step is 1032 (in this model the tool and the dashboard can not collide).

At each incremental step all 1032 tests are preformed using infinite cylinders to quickly check for possible collisions. Any infinite cylinders that are identified as possibly colliding are sent to finite cylinder testing to definitively determine if an actual collision has occurred.

A. Collision Case

The above work cell model was analyzed and a robot motion was planned for the task. A set of via points for the tool center point to follow and the number of intermediate steps were determined (tbl. I). At every incremental step all of the 1032 possible collisions were tested. Several collisions occurred. The first collision, (data shown in tbl. II) occurs between the 2^{nd} and 3^{rd} set of via points. Figure 12 shows where the dashboard side (segment 56) collides with the car frame right windshield support (segment 16). Hence this motion is unsuitable and additional motion planning is necessary.

B. Non-Collision Case

This case study uses the same robot and work cell however the planned motion (tbl. I) is different. The 2^{nd} and 3^{rd} via points are altered as follows: X was decreased to 800, Z decreased to 950, and the pitch was increased to -120 to have the tool center point closer to the robot base and the dash rotated slightly more vertically. These alterations provide the dashboard with needed clearance from the car frame. The final

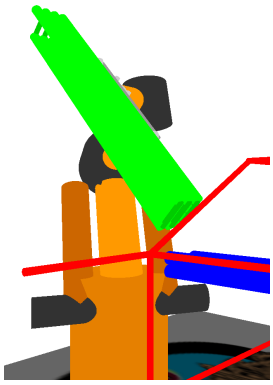

Fig. 12. Collision of the Dashboard and Car Frame

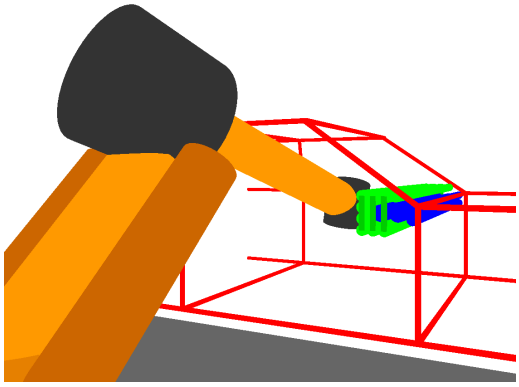

Fig. 13. Successful Dashboard Placement

TABLE I
COLLISION CASE STUDY - PLANNED MOTION

X mm	Y mm	Z mm	Yaw deg	Pitch deg	Roll deg	Increments
500	-1400	0	0	-180	-45	10
900	-1200	1200	90	-135	0	20
900	-375	1200	90	-135	0	20
1300	-375	521.5	90	-135	0	20
2150	-375	521.5	90	-180	0	40
2150	-725	521.5	90	-180	0	40

position of the dashboard places it within the car frame with $\frac{1}{2}$ (inch) clearance (fig. 13).

V. CONCLUSION

We have presented a novel methodology for detecting collisions of cylindrically shaped rigid bodies and the application of this methodology to motion planning for robotic mechanical systems. This algorithm uses line geometry and dual number algebra to exploit the geometry of cylindrical objects to facilitate the detection of collisions. The result is a straight-forward and efficient means of detecting collisions of cylindrically shaped bodies applicable to robot motion planning.

ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation under grant #0422705. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. This material is based on the preliminary work reported in [2].

TABLE II
COLLISION CASE STUDY - OUTPUT

Collision Type		Seg. No.		Seg. No.	
Two Hit		16		56	
X	Y	Z	Yaw	Pitch	Roll
900	-750	1200	90	-135	0
mm	mm	mm	deg	deg	deg
Axis 1	Axis 2	Axis 3	Axis 4	Axis 5	Axis 6
-28.08	-30.81	20.46	-84.53	62.41	-56.69
deg	deg	deg	deg	deg	deg

REFERENCES

[1] Fischer, Ian, (1999), Dual-number methods in kinematics, statics and dynamics, CRC Press LLC.
[2] Ketchel, J. and Larochelle, P. (2005), Collision detection of cylindrical rigid bodies using line geometry, *Proceedings of the ASME 2005 Design Engineering Technical Conferences and Computers and Information Conference, DETC05/MECH-84699.*
[3] McCarthy, J.M., (2000), Geometric design of linkages, Springer-Verlag New York.
[4] Xavier, P. (2000), Implict convex-hull distance of finite-screw-swept volumes, *Proceedings of the 2002 IEEE International conference on robotics & automation.*
[5] Zsombor-Murray, P.J. (1992), Spatial visualization and the shortest distance between two lines problem, *Proceedings of the 5th ASEE International Conference ECGDG.* Melbourne, Australia.
[6] Cohen, J., Lin, M., Manocha, D., and Ponamgi, M. (1995), I-collide: An interactive and exact collision detection system for large-scale environments, *Symposium on Interactive 3D Graphics*, 1995.
[7] Caselli, S., Reggiani, M., and Mazzoli, M. (2002), An experimental evaluation of collision detection packages for robot motion planning, *IEEE International Conference on Intelligent Robots and Systems.* Lausanna, Switzerland.
[8] Hudson, T., Lin, M., Cohen, J., Gottschalk, S., and Manocha, D., (1997), V-collide: Accelerated collision detection for vrml, *Proceedings of the 2nd Symposium on VRML*, ACM Press, 1997
[9] Johnson, T. and Vance, J. (2001), The use of voxmap pointshell method of collision detection in virtual assembly methods planning, *Proceedings of the ASME 2001 Design Engineering Technical Conferences and Computers and Information Conference, DET2001/DAC-21137.*
[10] www.SurplusRecord.com, 2005.