

TM-1078

曲面同士の干渉判定アルゴリズム

沢田 浩之

August, 1991

© 1991, ICOT

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03)3456-3191 ~ 5
Telex ICOT J32964

Institute for New Generation Computer Technology

曲面同士の干渉判定アルゴリズム

沢田浩之

新世代コンピュータ技術開発機構

機械設計やロボットの動作計画問題において必要とされる空間における曲面同士の干渉を判定するアルゴリズムについて述べ、実際にプログラムとして計算機上で実行した実験結果を示す。本アルゴリズムでは曲面を代数方程式で表現し、この問題を連立方程式の実根の有無の判定問題に帰着させている。その特徴は2つの方程式のうち一方の方程式をもう片方の方程式に対する制約条件とみなすことにあり、最終的にはその方程式の制約条件下における実根の有無を判定する問題としている。そのための手法としてLagrangeの未定乗数法を用いている。本アルゴリズムは実際の問題に対しての適用が容易であり、実験結果を見ても十分実用に耐え得るものと考えられる。

The Algorithm for Checking the Intersection between Surfaces

Hiroyuki Sawada

The Institute for New Generation Computer Technology

1-4-28 MITA, MINATO-KU, TOKYO 108, JAPAN

We discuss the algorithm for checking the intersection between surfaces, which is necessary for mechanical design and robot motion planning. We also show experimental results of the algorithm for some simple problems. Surfaces are expressed by polynomial equations and the intersection problem results in the problem of checking whether these equations have a real root. In this algorithm, one equation is regarded as a constraint for the other, and by Lagrange multiplier method we check whether the equation has a real root under that constraint. This algorithm is easy to implement, and from the experimental result it is thought to be practical.

1. はじめに

機械設計において重要なものの1つに、各部品同士の干渉の判定が挙げられる。また、ロボットの動作計画問題においては障害物の回避がやはり重要な問題となる。これらはいずれも空間における立体同士の干渉の判定問題であり、さらには平面および曲面の干渉判定問題と考えられる。従来の研究では、立体を多面体に限定したもの[1]が多く、また、一般の曲面を取り扱ったものにしても理論的には美しいものの実際にその方法を用いるのはかなり困難である([2],[3])。

著者はこの干渉判定問題を連立方程式の実根の有無の判定問題と考え、Lagrangeの未定乗数法を応用した解法を考案した。

本アルゴリズムの特徴は曲面を表す2つの方程式のうち一方を方程式、もう片方をそれに対する制約条件式と考えるところにある。すなわち最終的にはこの問題を、与えられた方程式が与えられた制約条件の下で実根を持つかどうかを判定するという問題に帰着させている。Lagrangeの未定乗数法はすでに最適設計等の分野では広く用いられている手法であり、この干渉判定方法は実際に用いるのも容易であると考えられる。

本アルゴリズムでは途中、多変数連立方程式を解く必要がある。本稿では曲面を表す方程式を代数方程式に限定し、その連立方程式の解法においてはRittの三角化手法を用いている。この連立方程式の解法は本アルゴリズムにおいてもっとも重要な部分であり、本稿の大部分をその考察に費やしている。

2. 問題の設定

今、2つの曲面の方程式をそれぞれ

$$f(x)=0, \quad g(x)=0$$

とし、対象とする領域を

$$R(x)<0$$

とすると、この2曲面の干渉判定問題は連立方程式の方程式の実根の有無の判定問題に帰着させる

ことができ、次のように定義される。

• 連立方程式

$$\begin{cases} f(x)=0 \\ g(x)=0 \end{cases} \\ x=(x_1, x_2, \dots, x_n)$$

が領域

$$R(x)<0$$

において実根を持つかどうかを判定する。

ところでこの定義は次の定義と等価である。

• 方程式 $f(x)=0$ が制約条件 $g(x)=0$ および $R(x)<0$ のもとで実根を持つかどうかを判定する。

本稿では後者の定義を用いて議論を進める。ただし、領域 $R(x)<0$ において関数 $f(x)$ および $g(x)$ はともに連続、微分可能であるものとする。

3. 基本アルゴリズム

方程式 $f(x)=0$ の実根の有無は関数 $f(x)$ の極値を求めることによって判定できる。すなわち、関数 $f(x)$ の与えられた条件の下での極値をすべて求めればそれらの符号により実数解の有無を判定できる。

極値を求めるために本稿ではLagrangeの未定乗数法を用いる。

まずはじめに2つの関数 $F(x)$ および $G(x)$ を次のように定義する。

$$\begin{aligned} F(x) &= f(x) + \lambda \cdot g(x) \quad [R(x)<0] \\ G(x) &= f(x) + \lambda \cdot g(x) + \mu \cdot R(x) \end{aligned}$$

ここで λ, μ はLagrangeの未定乗数である。

$F(x)$ および $G(x)$ の極値は、それぞれ与えられた領域内および境界上における $f(x)$ の極値となるので(制約条件: $g(x)=0$)、これら2つの関数の極値を求めれば良いことになる。すなわち、2組の連立方程式

$$\frac{\partial F}{\partial x} = 0, \quad \frac{\partial F}{\partial \lambda} = 0$$

および

$$\frac{\partial G}{\partial x} = 0, \quad \frac{\partial G}{\partial \lambda} = 0, \quad \frac{\partial G}{\partial \mu} = 0$$

を解いて極値点を求め、極値を求めればよい。関数はすべて代数多項式に限定しているので微分は容易にできる。

なお、領域Rが複数の不等式によって表される場合は次のような手続きによって干渉の判定を行う。

連立方程式: $f(x) = g(x) = 0$

領域R: $R_1(x), R_2(x), \dots, R_m(x) < 0$

(1) 関数 $F(x)$ 、 $G_i(x)$ 、 $H_{ij}(x)$ を定義する。

$$F(x) = f(x) + \lambda \cdot g(x)$$

$$G_i(x) = f(x) + \lambda \cdot g(x) + \mu_i \cdot R_i(x) \\ (i = 1, 2, \dots, m)$$

$$H_{ij}(x) = f(x) + \lambda \cdot g(x) \\ \mu_i \cdot R_i(x) + \mu_j \cdot R_j(x) \\ (i \neq j)$$

(2) 関数 $F(x)$ 、 $G_i(x)$ 、 $H_{ij}(x)$ の極値点を求め、領域外のものを除いて $f(x)$ の極値を求める。

(3) 領域の端点、すなわち

$$g(x) = R_1(x) = \dots = R_{n-1}(x) = 0$$

の場合について $f(x)$ の値を求める。

(4) 以上により得られた $f(x)$ の値の符号から実根の有無を判定する。

4. 連立代数方程式の解法

本節では連立代数方程式の解法について述べる。

本稿で用いている方法は、まず、与えられた連立方程式に三角化を施し、そのあとで順次方程式を解いてその代数解を代入していくというものである。

4.1 ではRittの三角化手法を示し、4.2 では1変数方程式の解法を示す。

4.1 Rittの三角化手法 ([5], [6])

与えられた連立方程式を

$$A_1(x) = A_2(x) = \dots = A_m(x) = 0,$$

$$x = (x_1, x_2, \dots, x_n)$$

とする。各方程式を x_n についてまとめると、次のように表すことができる。

$$A_i(x) = \sum_{j=0}^{N_i} a_{ij} x_n^j = 0 \\ (i = 1, 2, \dots, m)$$

$$a_{ij} = a_{ij}(x_1, x_2, \dots, x_{n-1})$$

ここで、まず、2つの方程式から x_n の次数を下げる方法について考える。2つの方程式をそれぞれ $A_k(x) = 0$ および $A_l(x) = 0$ 、 $N_k \leq N_l$ とする。 $a_{kN_k} \neq 0$ という条件の下で次の (I) および (II) は等価である。

$$(I) A_k(x) = A_l(x) = 0$$

$$(II) A_k(x) = 0, \\ B_l(x) \\ = a_{kN_k} A_l(x) - a_{lN_l} x_n^{N_l - N_k} A_k(x) \\ = 0$$

ところで、(II) の第2式を実際に計算してみると

$$a_{kN_k} A_l(x) - a_{lN_l} x_n^{N_l - N_k} A_k(x) \\ = a_{kN_k} \sum_{j=0}^{N_l} a_{lj} x_n^j \\ - a_{lN_l} x_n^{N_l - N_k} \sum_{j=0}^{N_k} a_{kj} x_n^j \\ = \sum_{j=0}^{N_l - 1} b_{lj} x_n^j$$

となって $B_l(x)$ では $A_l(x)$ に比べて x_n の次数が1つ下がっていることがわかる。ただし、

$$b_{lj} = \begin{cases} a_{kN_k} a_{lj} & (0 \leq j \leq N_l - N_k - 1) \\ a_{kN_k} a_{lj} - a_{lN_l} a_{kj - N_l + N_k} & (N_l - N_k \leq j \leq N_l - 1) \end{cases}$$

である。

この繰り返しにより、 $A_k(x)$ を用いて x_n の

次数を最終的に $N_k - 1$ 次まで下げることができる。
この作業をpseudo divisionと呼び、新たに得られた式をpseudo remainderという。

pseudo divisionの繰り返しによって結局 x_n に関する1次方程式と x_n を含まない方程式が得られる。

このようにして未知変数を1つずつ消去して三角化を行う。

なお、 $a_{kN_k} = 0$ となる場合については別に場合分けを行う。

【例】次のような連立方程式の三角化を行う。

$$x^2 + y + z - 6 = 0 \quad ①$$

$$xy - 2 = 0 \quad ②$$

$$y^2 + z - 2 = 0 \quad ③$$

(1) ①を用いて③から z を消去する。

$$x^2 + y + z - 6 = 0 \quad ①$$

$$xy - 2 = 0 \quad ②$$

$$x^2 - y^2 + y - 4 = 0 \quad ④ (=① - ③)$$

(2) ②を用いて④から y を消去する。

a. $x = 0$ の場合

②より $-2 = 0$ となり矛盾。

b. $x \neq 0$ の場合

$$② \times y + ④ \times x$$

$$x^3 + (x - 2)y - 4x = 0 \quad ⑤$$

$$② \times (x - 2) - ⑤ \times x$$

$$x^4 - 4x^2 + 2x - 4 = 0 \quad ⑥$$

(3) 三角化された方程式を出力する。

$$x^2 + y + z - 6 = 0 \quad ①$$

$$xy - 2 = 0 \quad ②$$

$$x^4 - 4x^2 + 2x - 4 = 0 \quad ⑥$$

4.2 1変数代数方程式の解法

与えられた方程式を $h(x) = 0$ 、 $h(x)$ の導関数を $h'(x)$ とする。 $h'(x) = 0$ の解および $\pm\infty$ によって解を挟み込むことで二分法で求める。

$h'(x) = 0$ の解はこのアルゴリズムを再帰的に用いれば求められる。方程式が1次式または2次式にたどり着いたらそこで解を与える。

【例】 $x^4 - 4x^2 + 2x - 4 = 0$ の解を求める。

(1) 微分を行う。

$$4x^3 - 8x + 2 = 0 \quad ⑦$$

(2) 微分を行う。

$$12x^2 - 8 = 0 \quad ⑧$$

(3) ⑧の解を求める。

$$x = \pm\alpha \quad (3\alpha^2 - 2 = 0, \alpha > 0)$$

(4) ⑦の解を求める。

$$x = \beta_1, \beta_2, \beta_3$$

$$-\infty < \beta_1 < -\alpha < \beta_2 < \alpha < \beta_3 < +\infty$$

(5) 元の方程式 $x^4 - 4x^2 + 2x - 4 = 0$ の解を求める。(この例の場合、実数解は $x = -2.359304\cdots, 2$ である。)

5. 実用上の留意点

本節では以上に示したアルゴリズムを実際に用いて問題を解決する場合の留意点について述べる。ここで述べる留意点はいずれも連立方程式の解法に関する部分である。

5.1 方程式系の分割

与えられた連立方程式

$$A_1(x) = A_2(x) = \cdots = A_n(x) = 0$$

のうち、 $A_i(x)$ が

$$A_i(x) = A_{i1}(x) \cdot A_{i2}(x) \cdots A_{im}(x)$$

のように因数分解が可能であるとする。このとき元の方程式系は

$$A_1(x) = \cdots = A_{i1}(x) = \cdots = A_n(x) = 0,$$

$$A_1(x) = \cdots = A_{i2}(x) = \cdots = A_n(x) = 0,$$

⋮

$$A_1(x) = \cdots = A_{im}(x) = \cdots = A_n(x) = 0$$

のような m 個の系に分割することができ、全体の解はこれらの解の和集合となる。

一般に $A_i(x)$ が m_i 個の因数の積で表されるとき、全体の系は $m = m_1 \times m_2 \times \cdots \times m_n$ 個に分割される。

あらかじめこのように分割しておくことによって取り扱う多項式の次数を下げることが効率

上望ましい。

5.2 Pseudo Division

2つの多項式

$$f(x) = a_n x^n + \cdots + a_0$$

$$g(x) = b_m x^m + \cdots + b_0 \quad (n \geq m)$$

に対してpseudo remainder $r(x)$ はpseudo divisionによって

$$f_1(x) = b_m f(x) - a_n x^{n-m} g(x)$$

$$f_2(x) = b_m f_1(x) - a_{n-1} x^{n-m-1} g(x)$$

⋮

$$f_{n-m+1}(x) = b_m f_{n-m}(x) - a_m x^{n-m} g(x) \\ = r(x)$$

のようにして求められる。ただし、

$$f_1(x) = a_{n-1} x^{n-1} + \cdots + a_0$$

とする。

今、 a_{n-1} と b_m の最小公倍数を L_1 とし、

$$L_1 = b_m' \cdot a_{n-1} = a_{n-1}' \cdot b_m$$

であるものとする。 $f_1(x)$ の最高次項は

$$b_m' f_1(x) - a_{n-1}' x^{n-m-1} g(x)$$

によって消去できる。

【例】次のような f 、 g について三角化を行う。

$$f = (y^2 - y)x + y^3 - y + 2$$

$$g = (y - 1)x + y^2 - y + 2$$

(1) f 、 g の最高次項の係数の最小公倍数 L を求める。

$$L = y^2 - y$$

$$= 1 \cdot (y^2 - y) = y \cdot (y - 1)$$

(2) pseudo divisionにより三角化を行う。

$$1 \cdot f - y \cdot g$$

$$= y^2 - 3y + 2$$

(3) 三角化した多項式を出力する。

$$(y - 1)x + y^2 - y + 2$$

$$y^2 - 3y + 2$$

6. 例題

本節ではいくつかの例をあげ、実際に干渉の判定を行う。

6.1 基本図形

基本図形として以下の6つの図形を考えることにする (Figure. 1)。

$$\text{球1: } x^2 + y^2 + z^2 = 1$$

$$\text{球2: } (x - 1)^2 + (y - 1)^2 + (z - 1)^2 = 1$$

$$\text{円柱1: } x^2 + y^2 = 1/2$$

$$\text{円柱2: } 1/2 \cdot (x + z)^2 + (y - 1)^2 = 1/2$$

$$[\text{軸: } x + z = y - 1 = 0,$$

$$\text{半径} R: R^2 = 1/2]$$

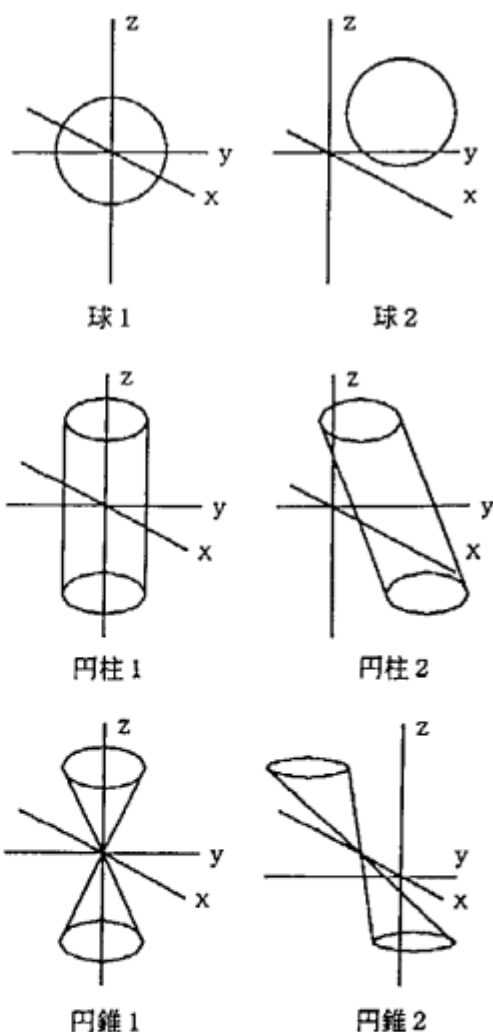


Figure. 1 基本図形

$$\text{円錐1: } x^2 + y^2 - 1/3 \cdot z^2 = 0$$

$$\text{円錐2: } (x+1)^2 + 1/2 \cdot (y+z)^2 - 1/6 \cdot (y-z)^2 = 0$$

$$\text{[軸: } x+1=y+z=0,$$

$$\text{頂点: } (-1, 0, 0), \text{頂角 } \pi/6]$$

これらの基本図形の幾何学的な干渉判定方法はAppendixに譲る。

6.2 干渉判定

基本図形同士の干渉判定の結果と、その実行時間をTable. 1に示す。領域はいずれも

$$x^2 + y^2 + z^2 < 100$$

である。なお、言語はKL1、計算機はPSIを用いた。

基本図形	干渉	実行時間(秒)
球1-球2	有	1.0
円柱1-円柱2	有	1.8
円錐1-円錐2	有	6.6
球2-円柱2	有	1.5
球2-円錐2	無	13.9
円柱2-円錐2	有	4.4

Table. 1 干渉判定の実行例

6.3 干渉範囲の特定

基本図形同士の干渉範囲の特定の例として、円錐1-円錐2の場合を取り上げる。この2つの図形はともにx軸に関して対称であるので $z \geq 0$ の領域を対称とする。

この領域をまず第1象限から第4象限まで分け、さらに $0 \leq z \leq 1$ 、 $1 \leq z \leq 3/2$ 、 $3/2 \leq z \leq 10$ の3つの場合に分ける。すなわち、領域を全部で12

の部分に分け、各部分領域について干渉判定を行う。その結果をTable. 2に示す。なお、いずれの場合も $x^2 + y^2 - 100 \leq 0$ を満たす範囲しか考えていない。

さらに細かく分割して行けばこれら2つの図形の交線の軌跡をたどることも可能であるが、計算量を抑えるためある程度その軌跡について見当をつけておくことが必要である。

蛇足だが、この交線の軌跡は次のように表現される。

$$x = r \cos \theta, y = r \sin \theta, z = \sqrt{3}r$$

$$2(\sin^2 \theta - 2\sqrt{3} \sin \theta - 3)r^2 - 6r \cos \theta - 3 = 0$$

象限	zの範囲	干渉	実行時間(秒)
1	$0 \leq z \leq 10$	無	12.7
2	$0 \leq z \leq 10$	無	12.7
3	$0 \leq z \leq 1$	無	12.4
	$1 \leq z \leq 3/2$	有	13.0
	$3/2 \leq z \leq 10$	有	2.2
4	$0 \leq z \leq 1$	無	12.4
	$1 \leq z \leq 3/2$	無	13.1
	$3/2 \leq z \leq 10$	有	2.2

Table. 2 干渉範囲の特定

6.4 考察

Table. 1において、球2と円錐2の干渉判定は他のものに比べて実行時間が多くかかっていることがわかる。これは判定結果が「干渉なし」であることと関係がある。すなわち、「干渉なし」と

いう判定を下すためにはすべての極値を調べればならないのに対し、「干渉がある」という判定を下すには符号が反転する極値を1つ見つければ済むためである。したがって「干渉あり」の場合であっても調べる極値の順番によっては多くの時間を費やすことが考えられる。

また、実行時間の多くは連立方程式を解くことに費やされていると考えられるので、曲面を表す方程式が複雑になればなるほど多くの時間がかかることが予想される。

7. 結論および今後の課題

機械設計やロボットの動作計画問題で必要となる曲面同士の干渉判定アルゴリズムとその計算機実装上の留意点について述べ、実験結果を示した。実験はいずれも2次元で表現される基本的な図形に限ったが、実行時間に関する限りでは実用に耐えうるものと考えられる。

さて問題点であるが、まず第1に、数値として浮動小数を使用しているため計算精度が低いということがある。これについては浮動小数ではなく有理数を用いることで向上が図れるが、その場合実行速度はかなり遅くなることが予想される。そしてもう1つは、第5節で述べた方法を実現するには多項式の因数分解機能が必要となるが、現在のところ因数分解を行うことのできる多項式が限られているため、それほど大きな効果を上げることが期待できないということである。

今後、これらの問題点を解決するとともに、連立方程式の解法としてBuchberger Algorithmなどの別の方法〔7〕,〔8〕,〔9〕や干渉範囲の特定方法についても考えていく必要がある。

謝辞

連立方程式の解法に関し、多くの助言を頂いたICOT第4研究室の佐藤洋祐主任研究員に深く感謝いたします。

参考文献

〔1〕Canny, J.F., "The Complexity of Robot

Motion Planning", The MIT Press, 1988.

〔2〕Schwartz, J.T and Sharir, M., "On the Piano Movers' Problem: II. General Techniques for Computing Topological Properties of Real Algebraic Manifolds", "Planning, Geometry and Complexity of Robot Motion", Ablex Publishing Corp., pp.51-96, 1987.

〔3〕Coste, M., "Effective Semialgebraic Geometry", Lecture Notes in Computer Science, pp.1-27, 1988.

〔4〕日本機械学会(編), 構造・材料の最適設計, 技報堂出版, 1989.

〔5〕Chou, S-C and Gao, X-S, "Techniques for Ritt-Wu's Decomposition Algorithm", The University of Texas, TR-90-02, 1990.

〔6〕Chou, S-C, "Mechanical Geometry Theorem Proving", D.Reidel Publishing Company, 1988.

〔7〕小林英恒, "多変数連立代数方程式の解法", 情報処理, vol.27, No.4, pp.414-421, 1986.

〔8〕佐武一郎, 線型代数学, 裳華房, 1958.

〔9〕ファン・デル・ヴェルデン(銀林浩訳), 現代代数学, 東京図書, 1959.

Appendix. 基本図形同士の干渉判定法

基本図形同士の幾何学的な干渉判定方法について述べる。ここで述べる方法は基本図形がいずれも回転体であるということを利用したものである。

2つの図形をそれぞれAおよびaとし、それぞれの中心軸上の点をPおよびp、線分PpとAおよびaの交点をそれぞれQおよびqとする。

$$\overline{Pp} \leq \overline{PQ} + \overline{pq}$$

となる2点P、pが存在すればAとaは干渉し、そうでない場合は干渉しない。