八分木と2段階FFTによる3次元衝突領域の高速算出*

城 道 介** 東 淳 也***

Quick Computation of Three Dimensional Collision Domain by Means of Octree and Double-stage Fast Fourier Transform

Michisuke Jo and Junya HIGASHI

To avoid collisions of two objects such as a workpiece and a tool, computation of collision domain in which the tip of the tool cannot locate is crucial. Collision domain can well be computed by Inverted Template Method using envelope operation. In this paper, a high-speed algorithm is proposed for three dimensional envelope operation by means of convolution operation. From two input domains α and β , an envelope domain γ is quickly computed by three dimensional Fast Fourier Transform (3D-FFT). Because it needs a huge memory capacity to convolute a whole cubic space precisely at one time, octree model is adopted to represent each of whole domains α , β and γ and Double-stage FFT method is established. First, the upper parts of input octree α and β are convoluted to get the rough sketch of the envelope domain γ . Second, pairs of lower subcubes of α and β are convoluted to compute precise 3D bitmaps. Among all the pairs, only the corresponding to boundary leaf cells of the upper γ octree are extracted. Each set of bitmaps for a leaf cell is converted into a suboctree through OR operation and added under the upper tree. Comparison of computing time with a precedent algorithm shows the superiority of Double-stage FFT method.

Key words: three dimensional collision domain, inverted template method, envelope operation, octree, 3D-FFT, convolution operation, double-stage FFT method, bitmap

1. はじめに

マシニングセンタで3次元加工を行う場合、工作物と工具との不用意な衝突を回避しなければならない。このような動的衝突問題の解決には衝突領域の算出が重要になる。衝突領域とは、工具の刃先、つまり移動物体の基準点が侵入してはいけない領域のことである。前報¹⁾において逆転演算と包絡演算を組み合わせた逆テンプレート法²⁾を使用して、衝突領域を算出する3種類のアルゴリズムを考案した。それらの方式の中では、重み付き縦優先方式が最も効率的であることを確認した。本報では、フーリエ変換を用いて包絡演算ができることを示し、そのアルゴリズムを提案する。

なお、本アルゴリズムは、姿勢一定の移動物体と静止物体の任意の対に適用されるものであるが、以下の記述では二つの物体の適切な例として“工具”、“工作物”を用いる。

2. 逆テンプレート法

本稿では、空間内の領域をギリシア小文字で表し、点もしくは位置ベクトルを英大文字で表す。

二つの物体の領域を α, β で表すとき、衝突領域 γ ²⁾ の内包的定義は次式で与えられる。

$$\gamma \equiv \{P | P+B=A, A \in \alpha, B \in \beta\} \quad (1)$$

衝突領域を求めるには、包絡演算²⁾と逆転演算²⁾が有効である。二つの物体領域 α と σ の包絡領域 δ は、包絡演算を記号“ $\oplus$ ”で表せば、

$$\begin{aligned} \delta &\equiv \alpha \oplus \sigma \\ &\equiv \{P | P=A+S, A \in \alpha, S \in \sigma\} \end{aligned} \quad (2)$$

領域 β の逆転領域 β^T は、

$$\beta^T \equiv \{B | -B \in \beta\} \quad (3)$$

包絡演算と逆転演算の処理ルーチンが存在すれば、衝突領域は次式に従って容易に求められる。

$$\begin{aligned} \gamma' &\equiv \alpha \oplus \beta^T \\ &\equiv \{P | P=A+(-B), A \in \alpha, B \in \beta\} \\ &\equiv \{P | P+B=A, A \in \alpha, B \in \beta\} \\ &\equiv \gamma \end{aligned} \quad (4)$$

すなわち、衝突領域は β の逆転領域と α の領域との包絡演算により算出される。この手法を逆テンプレート法(図1)と呼ぶ。本報では、 β^T と α を与えたときの包絡領域を高速に求めるアルゴリズムを提案する。

3. 八分木によるメモリの節約

3次元形状の記述形式として、一様空間格子がある。これは、一辺の分割数を増やしていくと、膨大なメモリが必要に

Fig. 1 Inverted Template Method

* 原稿受付 平成5年1月28日

** 正会員 茨城大学工学部 (日立市中成沢町4-12-1)

*** 学生会員 茨城大学大学院

Fig. 2 Octree for 3D domain representation

Fig. 3 Data structure of octree

なる。そこで、本研究では前報¹⁾と同様、3次元物体全体の記述に八分木を用いる(図2)。

物体領域に関し、八分木の各セルを外部、内部、境界および未判定に分類する。各セルの値を次のように定める。

- 0: 未判定セル
- 1: 外部セル
- 2: 内部セル
- 3: 境界セル

8等分する前のセルを親セル、8等分したそれぞれのセルを子セルと呼ぶことにする。八分木のデータ構造を図3に示す。木構造で、分割の最大レベルを I_x とすると、最下位セルの一辺の寸法は最上位セルの一辺の長さの $1/2^{I_x-1}$ となる。すべてのセルには、上で定義したセル値を与える。

一つの物体の八分木を構成するセルの個数は、物体の表面積にほぼ比例すると考えられる。たとえば、図5(a)の例では一辺の分割数の2.19乗に、球では1.8乗に比例する。

4. フーリエ変換による包絡演算

4.1 フーリエ変換³⁾

周期 T の関数 $s(t)$ の基本区間 $[0, T]$ におけるフーリエ変換は、次式で示される。

$$\tilde{s}(f) = \int_0^T s(t) e^{-j2\pi ft} dt \quad (5)$$

ここで、 t は時間、 f は周波数($1/T$)である。フーリエ変換は、時間関数の周波数領域における表現である。また、基本区間を $[0, f]$ とした逆フーリエ変換の式は次のように定義される。

$$s(t) = \int_0^f s(f) e^{j2\pi ft} df \quad (6)$$

N 個の要素を有する配列 $s(n)$ の離散フーリエ変換、離散逆フーリエ変換は次式のようになる。

$$\tilde{s}(k) = \sum_{n=0}^{N-1} s(n) e^{-j2\pi kn/N} \quad (k=0, \dots, N-1) \quad (7)$$

$$s(n) = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{s}(k) e^{j2\pi kn/N} \quad (n=0, \dots, N-1) \quad (8)$$

離散フーリエ変換の演算速度を高速化したものを、高速フーリエ変換(FFT: Fast Fourier Transform)という。FFTの計算量は、データ数 N が2のべき乗のとき、 $N \log_2 N$ に比例する。データ数 N が増大するに従って、高速化の効果が大きくなる。

4.2 たたみ込みと包絡領域

衝突領域、包絡領域は、式(1)、式(2)の内包的定義のほかに、相関もしくはたたみ込み演算によっても定義できる。二つの1次元配列 α, β の離散相関 g 、離散たたみ込み h は、次式により与えられる。

$$g(n) = \sum_{m=0}^{N-1} \alpha(m) \beta(n+m) \quad (n=0, \dots, N-1) \quad (9)$$

$$h(n) = \sum_{m=0}^{N-1} \alpha(m) \beta(n-m) \quad (n=0, \dots, N-1) \quad (10)$$

肥田ら⁴⁾は、工具と工作物各々の内部を1、外部を0とした配列に式(9)の相互相関演算を適用し、得られた値を各位置に工具の基準点を置いたときの工作物との干渉量と解釈している。これに対し、我々は、衝突領域算出において、干渉の程度を問題にせず、その値が0より大きいセルの集合を衝突領域と定義する。

式(10)のたたみ込み演算は、 α の内部セルすべて($m=0, \dots, N-1$)に工具基準点を置いたとき、観測点 n が工具領域内部に含まれる場合の総数を求めている。算出値が1以上のセルの集合は、前述した包絡領域となる。

また、要素数 N の1次元配列における逆転領域は、次式で示される。

$$\beta^T(n) = \beta(N-1-n) \quad (n=0, \dots, N-1) \quad (11)$$

逆テンプレート法は、式(10)と(11)を用いて次式で定義される。次式で与えられる領域を γ とする。

$$\gamma(n') = \sum_{m=0}^{N-1} \alpha(m) \beta^T(n'-m) \quad (n'=0, \dots, N-1) \quad (12)$$

β の逆転領域と α のたたみ込みは、式(9)の相互相関と等価であり、どちらも α と β の衝突領域を与える。

[証明]

$$\begin{aligned} \gamma(n') &= \sum_{m=0}^{N-1} \alpha(m) \beta^T(n'-m) \\ &= \sum_{m=0}^{N-1} \alpha(m) \beta(N-1-(n'-m)) \\ &= \sum_{m=0}^{N-1} \alpha(m) \beta(N-1-n'+m) \end{aligned}$$

ここで、 $n=N-1-n'$ とおくと、

$$\begin{aligned} \gamma(N-1-n) &= \sum_{m=0}^{N-1} \alpha(m) \beta(n+m) \\ &= g(n) \quad (n=0, \dots, N-1) \end{aligned} \quad (13)$$

[証明終]

式(9)、(12)のどちらでも衝突領域が求められるが、我々は前報¹⁾と整合性を保つため、式(12)を採用する。

要素数 N の配列のたたみ込みの離散フーリエ変換は、二つの入力配列の離散フーリエ変換の積として求めることができる³⁾。 $\alpha(n)$ 、 $\beta(n)$ をフーリエ変換したものをそれぞれ $\tilde{\alpha}(k)$ 、 $\tilde{\beta}(k)$ とすると、たたみ込みのフーリエ変換 $\tilde{\gamma}(k)$ は以下のよ

Fig. 4 Envelope operation using FFT and convolution

うに書ける。

$$\begin{aligned}\tilde{\gamma}(k) &= \sum_{n=0}^{N-1} \left(\sum_{m=0}^{N-1} \alpha(m) \beta(n-m) \right) e^{-j2\pi kn/N} \\ &= \tilde{\alpha}(k) \tilde{\beta}(k) \quad (k=0, \dots, N-1)\end{aligned}\quad (14)$$

$\tilde{\gamma}(k)$ を離散逆フーリエ変換すると、たたみ込み $\gamma(n)$ が得られる。

$$\gamma(n) = \frac{1}{N} \sum_{k=0}^{N-1} (\tilde{\gamma}(k)) e^{j2\pi kn/N} \quad (n=0, \dots, N-1)\quad (15)$$

要素数 8 の配列の場合を例にとり、1次元FFTとたたみ込みを用いた包絡演算の処理手順を図4に示す。二つの入力領域および一つの出力領域は、一辺の長さが同一の最上位立方体で

記述される。フーリエ変換のデータには周期性があり、領域の重複を避けるために、立方体の寸法は出力領域 γ がはみ出さないように十分大きく取る。つまり、入力領域 α, β の内部領域を完全に内含する最小直方体を考えたとき、二つの直方体の各軸方向の寸法和は、最上位立方体の寸法以下でなければならない。物体形状を表す配列を領域配列、フーリエ変換後の配列を周波数配列と呼ぶことにする。

- [1] 内部が1、外部が0の整数領域配列 α と β の値を複素領域配列 α', β' の実数部に入れる。
 - [2] α' と β' をフーリエ変換し、周波数配列 $\tilde{\alpha}, \tilde{\beta}$ を求める。
 - [3] $\tilde{\alpha}, \tilde{\beta}$ のそれぞれの複素数要素どうしの積として、たたみ込み周波数配列 $\tilde{\gamma}$ を算出する。
 - [4] $\tilde{\gamma}$ を逆フーリエ変換して、たたみ込み複素領域配列 γ'' を求める。図4の γ'' は実数部の値を示している。
 - [5] γ'' で実数部の値が0.0より大きい要素を1、それ以外の要素を0とし、整数領域配列 γ' に入れる。
 - [6] γ' の内部要素の各軸方向プラス側に隣接する外部要素の値を2（内部）に置き換えて、包絡領域配列 γ を生成する。この処理により、たとえば寸法1mmの立方体 α, β の包絡領域 γ は寸法2mmの立方体となる。
- 3次元離散たたみ込み γ は、次式のようにになる。

$$\gamma(x, y, z) = \sum_{l=0}^{N-1} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} \alpha(l, m, n) \beta(x-l, y-m, z-n) \quad (16)$$

この式と3次元FFTを用いて、3次元包絡領域が求められる。

3次元配列において、 N^3 個のセル空間の3次元高速フーリエ変換は、計算時間が $N^3 \log_2 N$ に比例する。メモリーは複素数配列を使うので、 $8 \times N^3$ byte 必要になる。

4.3 2段階FFT方式

3次元フーリエ変換は、一様空間格子に対応した3次元配列を用いるため、大きな作業領域が必要となる。たとえば、一辺の分割数が32のときは0.2 Mbyte、64のときは2 Mbyte、128のときは16 Mbyteのメモリーが占有される。メインメモリー640 Kbyteのパソコンは、分割数が32のたたみ込みしかできない。また、8 Mbyteのメインメモリーを持つEWSにおいても分割数が64のたたみ込みをするのが限界である。そこで、メモリー節約のために、全体を一度にたたみ込むをやめ、 α, β から部分立方体を取り出してたたみ込み、結果を重ね合わせていくという方法をとることにする。入力領域、出力領域は八分木で記述する。入力領域の八分木を上位と下位の2段階に処理を分ける。上位処理では、分割を粗くして大まかな包絡領域を算出し、上位八分木に変換する。 γ の上位八分木の一つの境界セルには、 α と β の内部セルもしくは境界セルのペアが複数個対応している。各ペアの片方は境界セル、もう一方は内部または境界セルである。この上位セルペアに対応する α, β の部分立方体を下位区分し、最下位セルに1bitずつ割り当てた3次元ビットマップを作成する。両方のビットマップから、たたみ込み演算により、 γ の3次元ビットマップを算出する。これをビット演算で重ね合わせて、 γ の部分立方体のビットマップを求める。確定したビットマップは、八分木に変換して、 γ の上位八分木に接続する。FFTによる下位のたたみ込みでは、前節で述べた寸法和の原則を満たすために、分割数64の領域配列を用いる。最上位セルの一辺を128等分した場合（レベル8）を例にとり、この方式の手順を説明する。

上位処理

- [1] α, β の八分木からレベル3までを取り出し, $4 \times 4 \times 4$ の領域配列を作成し, 包絡領域 γ を求める. 演算は2回実行する. 入力領域の内部セルのみを1, 他を0とした場合の包絡領域を γ_1 , 内部および境界セルを1とした場合を γ_2 とする.
- [2] γ_1, γ_2 の同一位置に対応する二つのセルについて, 出力値が両方とも1ならば内部セル, γ_2 のみが1ならば境界セル, 両方とも0ならば外部セルとして結果を上位八分木に変換する.

境界セルは, γ_2 の値1のセルのうち, 外部セル(値0)に接するものとしても判定可能であるが, 上記のように γ_1 との比較による判定の方がより効率的である.

下位処理

- [3] レベル3において, 外部セルと接する内部セルおよび境界セルの位置ベクトルを列挙したセルリストを α, β それぞれについて作成する.
- [4] α のセルリストから, セルを一つ取り出す.
- [5] β のセルリストから, セルを一つ取り出す.
- [6] 対応する上位八分木が, 外部もしくは内部セルならば, [11]へ飛ぶ.
- [7] 取り出したセルを根にして, 八分木のレベル3からレベル8までの領域配列を作成する. セル値は, 内部を1, 外部を0とする.
- [8] 包絡領域配列 ($64 \times 64 \times 64$) を算出する.
- [9] 包絡領域配列を8等分し, それぞれの領域配列の3次元ビットマップ ($32 \times 32 \times 32$) を作成する.
- [10] 各ビットマップは, 上位八分木の境界セルに対応している. 算出された新ビットマップを対応境界セルの既存ビットマップにOR演算で重ね合わせる.
- [11] β のセルリストに未処理セルがあれば, [5]へ戻る.
- [12] α のセルリストに未処理セルがあれば, [4]へ戻る.
- [13] 下位ビットマップすべてを八分木に変換して, 上位八分木に接続する.

以上の手順を2段階FFT方式と呼ぶことにする.

5. つき合わせ方式との比較

前報¹⁾では, α と β の各セルのつき合わせにより衝突領域を求めるアルゴリズムとして, 横優先方式, 縦優先方式および重み付き縦優先方式を考案した. 3種類のつき合わせ方式の中では, 重み付き縦優先方式による包絡演算が最も高速であるということが前報の結論である. そこで, この方式と今回の2段階FFT方式の計算時間の比較を行う.

以下に重み付き縦優先方式の概略について示す.

この方式は, α, β のセルをレベル1から縦に走査してつき合わせを行う. α と β の二つのセルをつき合わせて, γ セルが境界セルとなったときに, α, β とも一つ下位のレベルに降りる. 8個の子セルの中で重いセルを優先してつき合わせを行う. その中でまた γ セルが境界セルになるときは, さらに一つ下位レベルに降りていく. γ セルが内部セルと確定したときには一つ上位のレベルに戻る. この方式では, あるレベルで γ セルが内部と確定した場合, それ以下のレベルのつき合わせは行わない.

次に, 計算時間の比較について述べる.

まず, 2段階FFT方式において, 上位分割数を変えて計算

Fig. 5 Input domain for envelope operation

Fig. 6 Envelope domain, γ

Table 1 Comparison of Ctime and K to Dividing number of upper cube in Double-stage FFT method

	Dividing number		Ctime	K
	up	down		
64	16	8	1 414	2 928
	8	16	979	240
	4	32	462	11
	64	0	424	1

Ctime; Computing time (s)
 K ; Count of convolution

時間を測定した. 入力領域として与えた工作物領域 α , 工具の逆転領域 β^T を図5に示す. 図6が出力された包絡領域 γ である. 時間測定に使用した機種は, 主メモリー16 MbyteのEWS 2050 G/EXである. 上位分割数の違いによる計算時間とたたみ込みの回数を表1に示す. 上位分割数が少ないほど計算時間が短くなっている. また, 入力領域 α, β にそれぞれ2 Mbyteずつ使用して上位分割数を64, 1回で包絡演算をすると, さらに速くなっていることがわかる. 次に前報の重み付き縦優先方式と2段階FFT方式との計算時間の比較を図5の例について行った. 2段階FFT方式では, 上位分割数を4とした. 計算時間を測定した結果を, 横軸を一辺の分割数, 縦軸を計算時間としたグラフが図7である. 128等分で計算時間は重み付き縦優先方式では26 060秒, 2段階FFT方式では4 041秒であった.

Fig. 7 Comparison of two algorithms with computing time

ここで、二つの方式の計算時間を大まかに考察してみる。最上位立方体の一边を $m=2^p$ 等分したとき、物体 a, β の八分木のセル数 N_A, N_B は、3章で述べたように、ほぼその表面積に比例する。 a, b を比例定数として、

$$\left. \begin{aligned} N_A &= am^2 = a(2^p)^2 = a2^{2p} \\ N_B &= bm^2 = b(2^p)^2 = b2^{2p} \end{aligned} \right\} \quad (17)$$

よって、つき合わせの回数は、

$$Tm = N_A N_B = ab2^{4p} \quad (18)$$

他方、2段階FFT方式において、上位処理の計算時間は、上位分割数を $m_u=2^{p_u}$ 、 r を比例定数として、

$$\begin{aligned} Tfu &= rm_u^3 \log_2 m_u \\ &= r(\log_2 2^{p_u})(2^{p_u})^3 = rP_u 2^{3p_u} \end{aligned} \quad (19)$$

下位処理における分割数は、 $m_d=2^{p-p_u+1}$ となる。下位処理では、上位処理の出力領域の境界セルについて下位たたみ込みを行う。境界セルの数 N_c は、 c を比例定数として、

$$N_c = c(2^{p_u})^2 = c2^{2p_u} \quad (20)$$

下位処理の計算時間は、

$$\begin{aligned} Tfd &= N_c r m_d^3 \log_2 m_d \\ &= c2^{2p_u} r (\log_2 2^{p-p_u+1})(2^{p-p_u+1})^3 \\ &= cr2^{2p_u}(P-P_u+1)2^{3P-3P_u+3} \\ &= cr(P-P_u+1)2^{3P-P_u+3} \end{aligned} \quad (21)$$

したがって、2段階FFT方式に要する時間は、

$$\begin{aligned} Tf &= Tfu + Tfd \\ &= rP_u 2^{3P_u} + cr(P-P_u+1)2^{3P-P_u+3} \\ &= r(P_u 2^{3P_u} + c(P-P_u+1)2^{3P-P_u+3}) \end{aligned} \quad (22)$$

つき合わせ方式と2段階FFT方式の比は、

$$\frac{Tm}{Tf} = \frac{ab2^{4p}}{r(P_u 2^{3P_u} + c(P-P_u+1)2^{3P-P_u+3})} \quad (23)$$

この値は、分割レベル P つまり一边の分割数を増やしていくと、急激に増大する。以上の考察と計算時間の測定例から、以下のことが導かれる。

- (1) 主メモリの許す限り、できるだけ細かい分割の3次元FFT方式を用いる方がよい。
- (2) 全体の分割数固定の場合、2段階FFT方式では、下位分割を細かくするほど、計算時間が短縮される。
- (3) 2段階FFT方式とつき合わせ方式の出力領域は、どちらも八分木構造で表現する。16 Mbyteのメモリーがあれば、どちらの方式でも分割数が1024までの包絡領域の算出が可能である。16 Mbyteのメモリーでは、一边の分割数が64のFFTたたみ込みが限界である。2段階FFT方式において、分割数1024は上位分割数32、下位分割数64となる。全体の分割数の限界は、どちらの方式でも同じであるが、計算時間は式(23)により2段階FFT方式の方が約400倍高速となる。

2段階FFT方式のプログラムをスーパーコンピュータ(HITACS-820)に移植し実行させた。分割数が128では、計算時間は280秒であった。

6. おわりに

本研究の成果を以下にまとめる。

- (1) 逆転演算とFFTを用いたたたみ込み演算によって、衝突領域が求められることを示した。
- (2) 3次元衝突領域を算出するアルゴリズム、2段階FFT方式を考案した。この方式は、上位と下位に分けてたたみ込み演算を行い、衝突領域の八分木を出力する。
- (3) 2段階FFT方式において、メモリの許す限り、上位分割を粗く、下位分割を細かくすることにより計算時間が高速になることを確認した。
- (4) つき合わせ方式と2段階FFT方式の計算時間を比較した結果、一边の分割数が多くなるほど、2段階FFT方式が効率的であり、FFTとたたみ込みの有効性を確認した。

本報で提案したアルゴリズムによる処理例の分割は、まだ粗いが、たとえば工具の刃先以外の部分の衝突チェックには十分使用できる。MPP(超並列)コンピュータが進展すれば、十分実用に耐える手法となろう。

参考文献

- 1) 城 道介：3次元動的衝突領域の高速算出，精密工学会誌，58，9（1992）1593。
- 2) 城 道介，鳥居塚 卓，大島俊哉：旋削工程における動的衝突問題の一般解法，精密工学会誌，55，5（1989）883。
- 3) E. O. Bringham（宮川 洋，今井秀樹訳）：高速フーリエ変換，科学技術出版社，（1987）。
- 4) 肥田安雄，中村 隆，船橋鉦一：相互相関，畳込みによる形状加工の研究，1990年度精密工学会秋季大会学術講演会講演論文集，（1990）519。