Kobe University Repository : Kernel

PDF issue: 2025-12-05

ボクセルモデルを用いた切削シミュレーションにおける微小時間および微小空間解析の高速処理手法

西田，勇
佐藤，隆太
白瀬，敬一

(Citation)
精密工学会誌,84(2):175-181

(Issue Date)
2018-02-05

(Resource Type)
journal article

(Version)
Version of Record

(Rights)
© 2018 精密工学会

(URL)
https://hdl.handle.net/20.500.14094/90006019

KOBE UNIVERSITY

論文

ボクセルモデルを用いた切削シミュレーションにおける微小時間および微小空間解析の高速処理手法*

西田　勇** 佐藤隆太** 白瀬敬一***

High Speed Computational Algorithm in Voxel Based Milling Process Simulation for Minute Time and Minute Space Resolution Analysis

Isamu NISHIDA, Ryuta SATO and Keiichi SHIRASE

In order to improve machining efficiency, it is required to recognize machining status and optimize cutting conditions. Cutting force is meaningful information to recognize machining status. In the instantaneous rigid force model, which is the most popular model for milling force prediction, milling force is calculated based on the geometrical intersection between cutting edge and workpiece for each feed per tooth. In this model, both of static tool deflection and tool dynamic vibration are not considered. In order to overcome this problem, a new high speed computational algorithm in our voxel based milling process simulation is proposed. The proposed algorithm permits to consider both of static tool deflection and tool dynamic vibration in our voxel based milling process simulation. In the proposed algorithm, the intersection between cutting edge and workpiece is calculated in each minute time interval or minute tool rotational angle interval. Furthermore, the proposed algorithm permits to shorten the computational time of detecting removal voxels to calculate uncut chip thickness discretely. Therefore, high precision analysis can be performed in minute space resolution. The effectiveness of the proposed algorithm is validated by experimental 3-axis milling tests. Predicted milling forces under several cutting conditions have good agreement with the measured milling forces.

Key words: cutting process simulation, voxel model, cutting force prediction, end-milling operation, NC machining

1. 緒　　言

切削加工の加工効率を向上させるためには，加工状況を把握し切削条件を適切に設定することが重要となる．加工状況を把握する目的で切削力推定の研究が行われ，これまでに多くの切削力モデルが提案されてきた．過去に研究された切削力シミュレーションの手法の 1 つに瞬間切削力モデルがある [1)~4)]．この他にもエネルギ解析法で切りくず流出角を決定して切削力を推定する方法 [5)]や，有限要素法を用いて切削現象を解析する市販の AdvantEdge FEM などがある．これらの中でも瞬間切削力モデルは工具切れ刃と被削材の干渉量から比較的容易に現実的な切削力の計算を行うことができる．しかし，この手法は工具切れ刃と被削材の干渉量から切削力の予測に必要な実切込み厚さを計算するために，加工形状の変化が複雑で工具切れ刃と被削材の接触状態が非一様になると切削力の推定が困難となる．そこで，工具切れ刃と被削材の干渉量を算出するために，これまでの研究では，3DCAD のソリッドカーネルを用いて加工対象物の形状変化を再現する手法 [6)]や，Z-Map モデルを用いて工具軸方向からの加工対象物表面形状を推定し，これと工具との干渉量を推定する手法 [7)~10)]，ボクセル形状表現に再帰的分割を導入する手法 [11)]や距離属性を追加して，高精度な推定を可能とする手法 [12)]などが提案されている．これらの中でも，ボクセル形状表現による手法では，工具の姿勢変化や加工対象物のオーバーハング，立壁等の複雑な形状に対応しやすく，多軸制御切削加工に対応した被削材形状評価では多くの適用例が従来の研究において見られる [13)~16)]．ボクセル形状表現による解析では，切削シミュレーションの精度を向上する際に必要となる計算機メモリの総量と，多数のボクセルと工具切れ刃との間での干渉判定の高速化が大きな課題となる．従来の研究では，ボクセルモデルに Octree 構造を導入してメモリの消費を抑える手法が提案されている [11)]．Octree 構造では，工具と干渉したボクセルに対して，1 辺の長さが 1/2 となるボクセルに分割する処理を繰り返すことにより，ボクセルの総数を抑制しながら微小な被削材表面形状の記述が可能となっている．一方で，干渉判定の高速化において，GPU (Graphics processing unit)を用いた大規模並列計算手法が提案されている [17)]．GPU を用いた計算では，CPU 上で処理を行う対象となるデータとプログラムを用意し，これを GPU 上に転送して処理を行い，結果を CPU に再び転送する過程の繰り返しによって実現されるため，GPU-CPU 間の情報転送のオーバーヘッドが問題となる．そこで，情報転送のオーバーヘッドを小さくするためのメモリ設定法が提案され [18)]，GPU の並列計算機能を効率よく利用可能な方式に拡張することで，Octree 構造での高精度ボクセル表現を可能にしている．

しかし，これまでの研究では工具 1 刃当たりの送り量ごとに解析を行っているため，工具軌跡の近似誤差が発生するといった問題がある．また，工具 1 刃当たりの送り量の間では切削状態は変化しないことが前提であった．さらに，解析に使用する工具を工具軸に回転した回転体形状として被削材との干渉判定を行っていた．つまり，円柱形状で表現できるスクエアエンドミルや球形状で表現できるボールエンドミルなどの工具に限定されていた．

本研究では，被削材をボクセル形状表現する従来の切削シミュレータ [11)16)]を拡張して，工具切れ刃の形状も微小間隔の点群で離散的に表現し，工具 1 刃当たりの送り量ごとの解析ではな

* 原稿受付 平成 29 年 8 月 1 日
掲載決定日 平成 29 年 9 月 20 日
** 正 会 員 神戸大学（兵庫県神戸市灘区六甲台町 1-1）
*** フェロー 神戸大学

く，工具微小回転量ごとの解析を可能とする切削シミュレータを新たに開発した．これにより微小時間および微小空間の分解能で切削現象のシミュレーションが可能となるため，加工中の工具の静変形や動変形（振動）を考慮した解析に応用が期待できる．

また，最近ではPCの性能が飛躍的に向上し，解析時に確保できるメモリ量は大幅に増加している．そこで本研究では，工具微小回転量ごとの解析においても高速な処理を実現するため，Octree構造を用いない新たな計算処理手法を提案する．最後に，提案手法の妥当性を検証するために切削加工実験を行い，提案手法により推定される切削力の結果と測定結果とを比較した．

2. ボクセルモデルを用いた微小時間および微小空間分解能での切削現象の解析

2.1 微小時間および微小空間の分解能での解析手法

切削加工プロセスのシミュレーションにおいて，従来からボクセルモデルを用いて離散的に切削現象をシミュレーションする手法が提案されてきた[11)13)]．ボクセルモデルは被削材と工具間の干渉を検出し，複雑な加工形状を簡便に表現することが可能である．従来のボクセルモデルを用いた干渉判定では，図1に示すように工具1刃当たりの送り量ごとに工具中心を移動させ，工具が新しい位置に移動するごとに工具領域内部に存在するボクセルを探索して，工具と被削材との干渉量を算出している．このとき，工具中心軸に対する各ボクセルの距離 L および各ボクセルを包含する球の半径 r と工具半径 R との大小を比較することにより工具領域内部のボクセルを判定している．従来のシミュレータでは工具1刃当たりの送り量ごとに解析を行うことで，工具軌跡は円弧で近似していた．そのため，発生するカスプ形状を正しく表現することができず，工具1刃当たりの送り量の途中で変化する工具の静変形や動変形（振動）を考慮するには不十分であった．

本研究で開発する新しいシミュレータでは，工具1刃当たりの送り量ごとの解析ではなく，工具微小回転量ごとに解析を行うことで，微小時間および微小空間分解能で切削現象のシミュレーションが可能となる．図2(a)に本研究で提案する工具微小回転量ごとの解析を示す．新しいシミュレータでは，図2(b)に示すように工具を工具軸方向に沿って微小薄板要素に分割して，微小薄板要素ごとに工具中心と工具切れ刃を結ぶ線分上に存在するボクセルを判定して，工具と被削材との干渉量を算出する．このとき，工具微小回転量は工具切れ刃の円周部の移動量が解析に使用するボクセルの1辺の長さと等しくなるように設定する．

例えば，工具回転速度 Smin^{-1}，工具半径 Rmm，解析に使用するボクセルサイズを Vmm とすると，1解析ステップ当たりの解析時間 t_{step} sec は下記のように求められる．

$$t_{step} = V/R/(S/60 \times 2 \times \pi) \quad (1)$$

新しいシミュレータでは，工具切れ刃の軌跡に忠実な解析が可能となるだけでなく，工具切れ刃の形状が特殊な場合や工具姿勢が非一様に変化する加工の解析が可能となる．図3に工具切れ刃の軌跡とカスプ形状を従来手法と本提案手法とを比較して示す．従来手法における円弧で近似した軌跡（黒色）と本提案手法におけるトロコイド曲線（青色）を示す．トロコイド曲線は

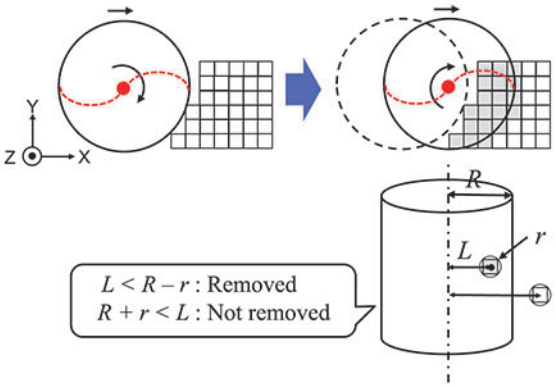
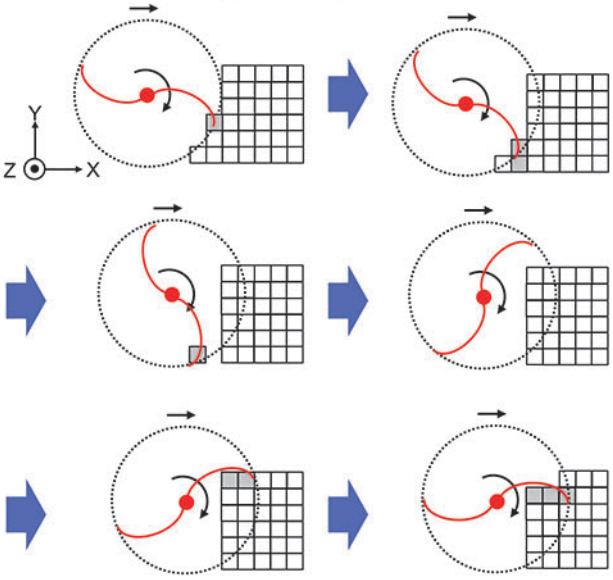
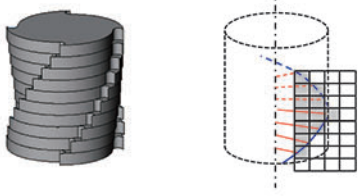


Fig.1 Extraction of removal voxels in our previous simulator for each feed per tooth analysis



(a) Extraction of removal voxels by each minute tool rotation angle



(b) Minute disk element of the cutting edge

Fig.2 Extraction of removal voxels in our new simulator for each minute tool rotational angle analysis

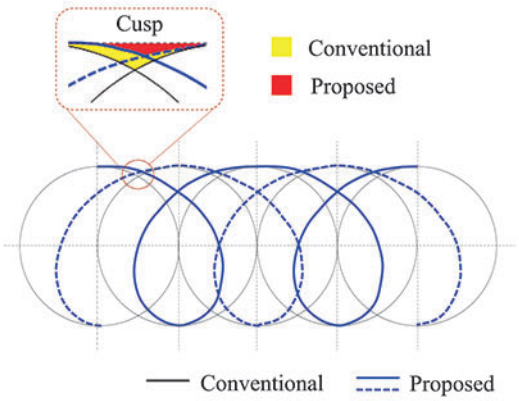


Fig.3 Comparison of conventional method and proposed method for tool cutting edge trajectory and cusp shape

2枚刃の工具を使用した場合の工具切れ刃軌跡を示している．また，図3に示したカスプ形状は，工具切れ刃軌跡を円弧で近似している従来手法では大きくなることがわかる．工具半径が小さく工具1刃当たりの送り量が大きい（工具回転数が小さく，工具送り速度が大きい）場合に，カスプ形状の誤差は大きくなることが幾何学的な関係からわかる．

2.2 工具と干渉するボクセルの高速検出手法

ボクセルモデルを用いた切削シミュレータでは，シミュレーションの精度を向上する際に必要となる計算機メモリの総量と，多数のボクセルと工具切れ刃との間での干渉判定の高速化が大きな課題となる．従来の研究では，ボクセルモデルにOctree構造を導入してメモリの消費を抑える手法が提案されている[11]．しかしながら，Octree構造では1個のボクセルを8等分に分割して小さなボクセルとし，これを多階層にすることで微細な形状を表現しているため，大きな被削材を高精度に表現し，実用的な加工形状のシミュレーションを行うには計算時間の短縮化が課題であった．また，干渉判定の高速化において，GPUを用いた大規模並列計算手法が提案されている[17][18]が，GPUを用いた計算では，CPU上で処理を行う対象となるデータとプログラムを用意し，これをGPU上に転送して処理を行い，結果をCPUに再び転送する過程の繰り返しによって実現している．このため，GPU-CPU間の情報転送のオーバーヘッドが問題となり，本研究で実現する工具微小回転量ごとの解析においては不向きである．本研究では，微小時間および微小空間の分解能で切削現象を高速で解析するため，Octree構造を用いないCPUの計算処理による新たな解析手法を考案した．

まず，本研究で用いるCPUの計算処理による解析の高速化の有効性を示す．繰り返し計算処理においては，代表的には3つの方法があり，1.CPUによる直列計算，2.CPUによる並列計算，3.GPUによる並列計算，である．並列計算では，複数の同じ処理を同時に実行でき，計算時間の短縮が期待できるが，並列化のためのオーバーヘッドが必要となるため，処理内容や繰り返し回数によっては計算時間が長くなることがある．単純処理を繰り返し行う処理において，繰り返し回数を変化させて3つの計算処理に要する時間を比較した．ここでは，bool型の変数にtrueをセットする単純な処理を繰り返し実行したときの処理時間の比較を行った．使用した計算機の主な仕様はCPU: Intel Xeon 3.5GHz（論理プロセッサ数: 8コア），GPU: NVIDIA M4000（CUDAコアプロセッサ数: 1664コア）である．CPU処理においては，.NET Framework 4.6を用いた．GPU処理においては，GPU-CPU間の情報転送処理およびメモリ管理を自動で行うAlea GPUライブラリ[19]を用いた．プログラム言語C#を使用し，プログラムを記述した．使用したプログラムの記述を**図4**に示す．また，各計算方法での10回平均の処理時間の結果を**表1**に示す．表1に示した結果から，繰り返し数が10^7回より少ない場合は，CPUによる直列計算で最も処理時間が短いことがわかる．この結果から，本研究ではボクセルと工具切れ刃との間での干渉判定の処理を単純にし，繰り返し回数を少なくすることで，CPUによる計算処理による解析手法を開発する．

従来のOctree構造によるボクセルと工具切れ刃との間での干渉判定の方法を**図5**に示し，本研究で開発する新たな干渉判定方法を**図6**に示す．

Octree構造による干渉判定では，図5(a)に示すように被削材を大きいサイズのボクセルで表現しておき，全てのボクセルから工具切れ刃と干渉しているボクセルを探索する．そして，干渉判定されたボクセルに対して，図5(b)に示すように1辺の長さが1/2となるように分割して，分割したボクセルに対して再度工具切れ刃と干渉しているボクセルを探索する．この処理を繰り返すことで，微細な形状表現が可能となり，シミュレーションの精度（分解能）を向上している．しかしながら，Octree構造による解析では，段階的にボクセルサイズを小さくするため，より微細な形状を表現する場合に繰り返し処理が多くなる点や，工具切れ刃と干渉しているボクセルを各階層で探索する必要がある点が欠点となり，処理時間の観点で大きな被削材を高精度に表現することが困難であった．

新しい提案手法では，図6(a)に示すように，被削材を初めから最小サイズのボクセルで表現して，各ボクセルにインデクスを付与して規則的に整列しておく．これにより，位置座標からボクセルのインデクスを特定することが可能となる．例えば，最小サイズのボクセルの1辺の長さがV_L μmで，x軸方向にn個の

```
for (int i = 0; i < num; i++)
{
        answer[i] = true;
}
```

(a) Serial calculation by CPU

```
ParallelOptions options = new ParallelOptions();
options.MaxDegreeOfParallelism = 6;
Parallel.For(0, num, i =>
{
        answer[i] = true;
});
```

(b) Parallel calculation by CPU

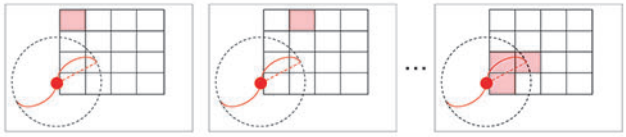
```
Gpu.Default.For(0, num, i =>
{
        answer[i] = true;
});
```

(c) Parallel calculation by GPU

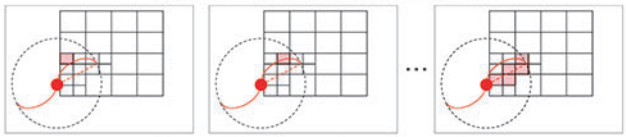
Fig.4 Source code for verifying calculation time

Table 1 Comparison of processing time

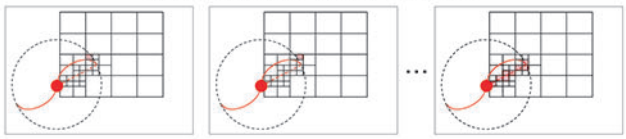
Iteration number times	10^6	10^7	10^8	10^9
CPU Serial sec	0.0023	0.017	0.19	1.60
CPU Parallel sec	0.0064	0.023	0.13	1.15
GPU Parallel sec	0.94	0.91	0.96	1.22



(a) Extraction of removal voxels in Level 1

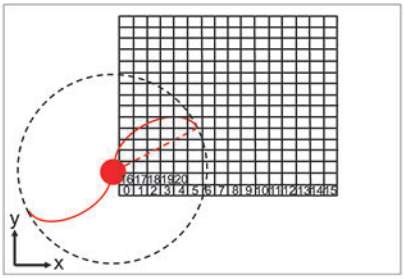


(b) Extraction of removal voxels in Level 2

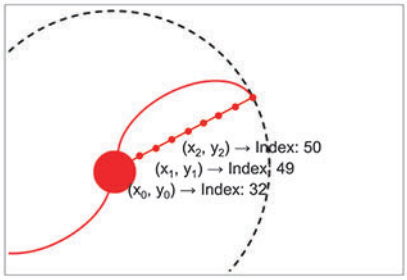


(c) Extraction of removal voxels in Level 3

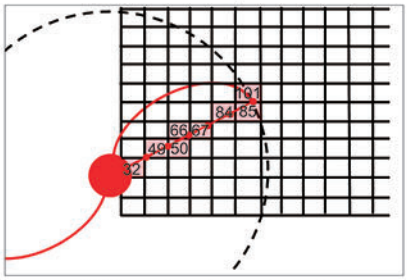
Fig.5 Extraction of removal voxel by octree method



(a) Workpiece represented by indexed minimum voxels



(b) Dividing tool cutting edge vector and specifying the index of workpiece voxel from each position



(c) Extraction of removal voxels

Fig.6 Extraction of removal voxels by proposed method

ボクセルが整列している場合，x_i μm, y_i μm の位置に相当するボクセルのインデクス I は以下のように求めることができる．

$$I = y_i/V_L \times n + x_i/V_L \qquad (2)$$

工具切れ刃と干渉しているボクセルを検出する際には，各薄板要素での工具切れ刃ベクトルを最小サイズのボクセルの 1 辺の長さで分割し，分割した要素の工具中心に近い位置の座標から式(2)より，干渉しているボクセルのインデクスを算出する．分割した各要素で求まったインデクスに相当するボクセルが除去対象のボクセルとして検出できる．これにより，被削材のボクセル全てを走査して干渉しているボクセルを探索するための繰り返し処理が不要となり，処理時間が短縮できる．

しかしながら，被削材を初めから最小サイズのボクセルで表現する場合，ボクセルに対応する変数の配列の数が膨大になるといった問題がある．例えば，1 辺が 1 m の立方体形状の被削材を 1 辺の長さが 1 μm のボクセルで表現する場合，10^{18} 個の配列が必要となる．ところが，.NET Framework では 1 つの変数で確保できる配列数は 4×10^9 個に制限されている [20]ため，被削材を表現することができない．そこで，被削材を 2 階層のボクセルサイズで表現することでこの問題に対処した．例えば，1 辺が 1 m の立方体形状の被削材を 1 辺の長さが 1 mm のボクセル(Large Voxel)で表現する場合，10^9 ($< 4\times10^9$) 個の配列で表現することができる．そして，工具切れ刃と干渉している Large Voxel を，1 辺の長さが 1 μm のボクセル(Small Voxel)で表現して解析

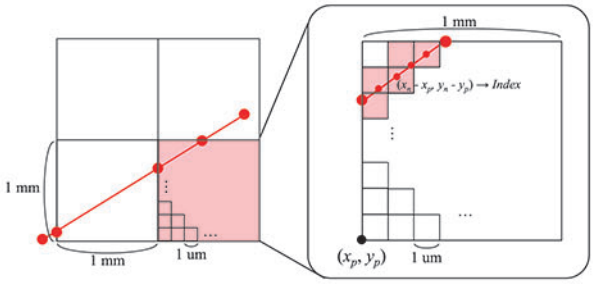


Fig.7 Extraction of removal voxels by 2 levels calculation

を行う．工具切れ刃と干渉しているボクセルの検出では，図 7 に示すように，まず各薄板要素での工具中心と工具切れ刃を結んだベクトル（以下，工具切れ刃ベクトルと呼ぶ）と干渉している Large Voxel を検出する．次に，検出された Large Voxel と工具切れ刃ベクトルとの交点を算出する．算出した交点を結ぶ線分を Small Voxel の 1 辺の長さで分割し，図 6 および式(2)に示した方法で除去対象のボクセルを検出する．その際，Large Voxel の位置だけオフセットされているため，工具切れ刃ベクトルを分割した各要素の位置座標から Large Voxel の原点の座標を引くことで，式(2)と同様に除去対象のボクセルのインデクスを算出することができる．ここで，Large Voxel と Small Voxel の 1 辺の長さは任意に設定することが可能であり，被削材の大きさとシミュレーションの精度（分解能）に適した値を設定することができる．

2.3 ケーススタディ

本研究で提案した工具切れ刃と被削材の干渉量の算出手法の有効性を確認するために，実加工用の工具経路を用いて，ケーススタディを行った．ケーススタディでは，Octree 構造による干渉判定手法 [11]での検証に用いている加工形状と同様のものを使用した．加工形状はドーム形(ドームの高さ：10 mm)とし，3 軸制御加工でのシミュレーションを行った．工具は直径 4 mm のスクエアエンドミル(荒加工)およびボールエンドミル(仕上げ加工)，加工前の被削材形状は 50×50×30 mm の直方体である．

Large Voxel の 1 辺の長さを 5 mm とし，Small Voxel の 1 辺の長さを変化させて，CPU の直列計算と CPU の並列計算との処理時間を比較した．使用した計算機の主な仕様は CPU: Intel Xeon 3.5 GHz (論理プロセッサ数: 8 コア)である．図 8 に Small Voxel の 1 辺の長さを 100 μm としたときに得られた加工形状を示す．表 2 に Small Voxel の 1 辺の長さを変化させた際の解析ステップの総数，処理時間および 1 解析ステップ当たりの処理時間を示す．処理時間は，Small Voxel の 1 辺の長さによらず，CPU の直列計算より CPU の並列計算の方が短くなっていることが分かる．この結果から，本研究で提案したボクセル表現による解析では，工具切れ刃と干渉するボクセルを検出する際の繰り返し処理の 1 回当たりの処理が CPU の並列化のためのオーバーヘッドよりも長くなるため，CPU の並列計算の場合に処理時間が短くなることが分かる．

また，Small Voxel の 1 辺の長さが 1/2 になると，解析ステップの総数が 2 倍になっていることが分かる．これは，工具微小回転量を工具外周部での切れ刃移動量が Small Voxel の 1 辺の長さと等しくなるように決定しているためである．さらに，Small Voxel の 1 辺の長さが 1/2 になると，1 解析ステップ当たりの解析時間が 2 倍になっていることが分かる．これは，工具切れ刃が被削材と干渉しているボクセルを検出する際に，工具の微小

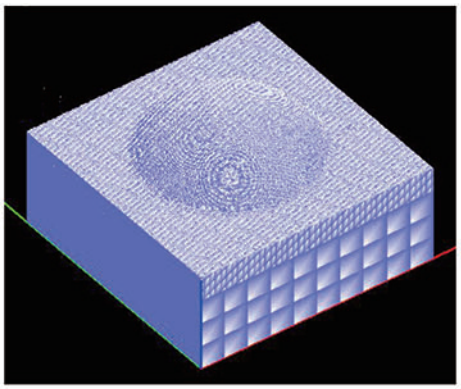


Fig.8 Cutting shape simulation (Small voxel size 100 μm)

Table 2 Comparison of computational time in different small voxel size

Small voxel size μm	Number of steps	Computational time sec		Time per step sec	
		Serial	Parallel	Serial	Parallel
200	271,081	97.3	68.5	0.00035	0.00025
100	535,484	405.0	254.3	0.00076	0.00048
50	1,064,552	1949.8	1108.5	0.00180	0.00100

薄板要素における工具切れ刃ベクトルを Small Voxel の 1 辺の長さで分割して解析を行うため，Small Voxel の 1 辺の長さが 1/2 になると，工具切れ刃ベクトルの分割数が 2 倍になるためである．つまり，Small Voxel の 1 辺の長さが 1/2 になると，解析ステップ数が 2 倍になり，1 解析ステップ当たりの解析時間が 2 倍となるため，合計の処理時間は 4 倍となる．

ケーススタディの結果から，提案手法でのシミュレーションの精度（分解能）と計算時間の関係性が示され，シミュレーションの要求精度に応じた計算時間の見積もりが可能である．

3. 切削力モデル

本研究では，工具切れ刃と被削材の干渉量の解析手法を新しく更新しているが，干渉量（実切込み厚さ）から切削力を予測する計算は瞬間切削力モデルを踏襲している．瞬間切削力モデルでは，**図 9** に示すように，工具を工具軸に沿って微小薄板要素に分割して，個々の要素ごとに微小切削力を計算する．この微小切削力を力の方向を考慮しながら足し合わせて，工具に作用する切削力を求める．微小切削力は各薄板要素の切れ刃先端に作用すると仮定し，切れ刃に垂直な面内での加工を二次元切削状態で近似している．それぞれの薄板要素に作用する切削力の接線方向成分 dF_t, 半径方向成分 dF_r, 軸方向成分 dF_a は以下の式で表される[21]．

$$dF_t = [K_{te} + K_{tc} h(\theta, z)]dz \quad (3)$$

$$dF_r = [K_{re} + K_{rc} h(\theta, z)]dz \quad (4)$$

$$dF_a = [K_{ae} + K_{ac} h(\theta, z)]dz \quad (5)$$

ここで，K_{te}, K_{re}, K_{ae}, K_{tc}, K_{rc}, K_{ac} は予備実験から得られる切削係数であり，$h(\theta,z)$ は工具半径方向の実切込み厚さ，dz は工具軸方向に分割した微小薄板要素の厚さである．つまり，工具半径方向の実切込み厚さ $h(\theta,z)$ が求まれば，切削力を算出することができる．

各微小薄板要素における工具切れ刃ベクトルでの除去対象ボクセルの個数は，第2章で示した解析手法で検出されるが，各軸方向の干渉量（実切込み厚さの各軸成分）は，除去対象ボクセルの個数とボクセルの1辺の長さの積から計算

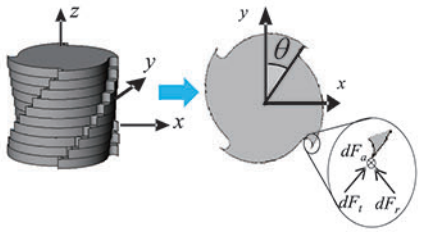


Fig.9 Cutting force model

できる．実切込み厚さ $h(\theta,z)$ は，x 方向成分 h_x と y 方向成分 h_y, z 方向成分 h_z を用いて次式で表すことができる．

$$h(\theta,z) = \sqrt{h_x(\theta,z)^2 + h_y(\theta,z)^2 + h_z(\theta,z)^2} \quad (6)$$

4. 実加工によるシミュレーションの検証

提案手法の有効性を検証するためにスクエアエンドミルを用いた切削加工実験を行い，切削力の測定結果と推定結果を比較した．切削条件および切削力シミュレーション条件を**表3**に示す．加工実験では立て形マシニングセンタを用い，切削加工を行い，加工中の切削力を水晶圧電式動力計（KISLER 9257B）で測定した．

切削力の予測に必要な切削係数は竹内ら[22]が示した方法を用いて，予備実験の結果から算出した．予備実験で決定した切削係数を**表 4** に示す．**図 10** に新しいボクセルモデルを用いた切削力シミュレーションの実行画面と切削力の推定結果を示す．図 10 の A, B, C, D は加工途中の被削材(図中の紫色)と各薄板要素での工具中心と工具切れ刃を結んだベクトル（図中の黄色）の関係をスナップショットで示している．図 10 に示した被削材を表すボクセルモデルでは，工具切れ刃が一度も干渉していない領域は Large Voxel の状態のままとなっているが，工具切れ刃が干渉した領域は Large Voxel が Small Voxel に分割され，工具中心と工具切れ刃を結んだベクトルが通過した部分の Small Voxel が除去されている様子がわかる．また，工具 1 刃当りの送り量ごとの解析ではなく，工具微小回転量ごとに解析を行っていることを示している．このように工具微小回転量ごとの解析が可能となることで，本稿ではまだ考慮していないが，加工中の工具の静変形や動変形（振動）を考慮した解析に応用が期待できる．

図 11 に切削力の測定結果と推定結果を比較して示す．それぞれの結果において，切削開始付近と切削終了付近に差異が見られるものの，測定結果と推定結果の波形はよく一致していることがわかる．切削開始付近と切削終了付近の差異の原因として，以下の 2 点が考えられる．1 点目は，シミュレーションでのボクセルの分解能（サイズ）によるものが考えられる．検証ではボクセルの最小サイズが 20 μm であったため，それ以下の精度で実切り取り厚さが計算できず切削力に誤差が生じたと考えられる．2 点目は，本稿では工具の変形を考慮していないことが考えられる．切削開始付近や切削終了付近の動的な工具の変形を考慮していないため，切削力に誤差が生じたと考えられる．しかしながら，測定結果と推定結果の波形が概ね一致していることから，本研究で提案した手法を用いた切削力予測が従来の手法と同等に正しく行われていることが確認できた．

Table 3 Cutting condition and cutting force simulation condition

Machine tool		NMV1500DCG
Workpiece		A5052
Cutting tool	Tool type	Square end mill
	Helix angle	30 °
	Number of flutes	2
	Diameter	6.0 mm
Cutting conditions	Cutting direction	Up cut
	Axial depth of cut	3.0 mm
	Radial depth of cut	3.0 mm
	Spindle speed	2000 min^{-1}
	Feed rate	200, 400 mm/min
Disk element thickness		0.020 mm
Minimum voxel size		0.020 mm

Table 4 Determined cutting coefficients

Cutting coefficients	K_{te}	0.5 N/mm
	K_{tc}	1323.7 N/mm
	K_{re}	0.4 N/mm
	K_{rc}	792.2 N/mm^2
	K_{ae}	3.1N/mm^2
	K_{ac}	81.6 N/mm^2

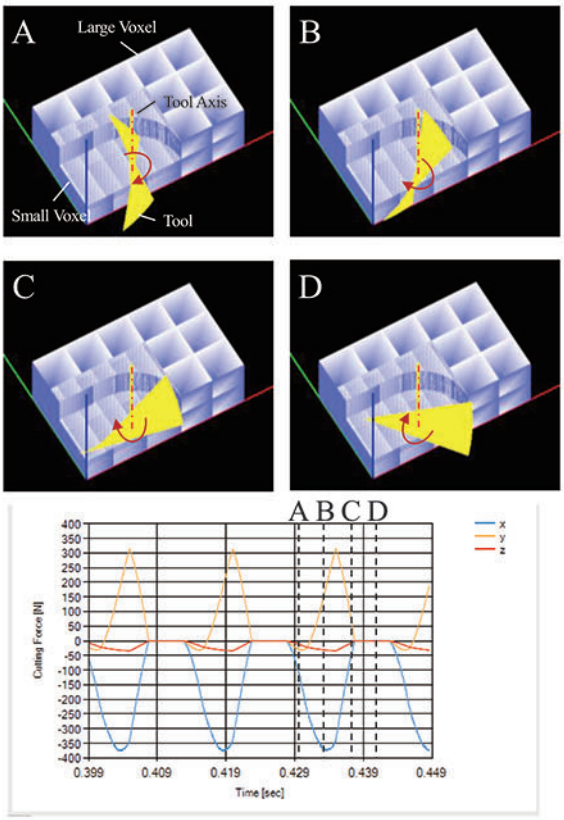
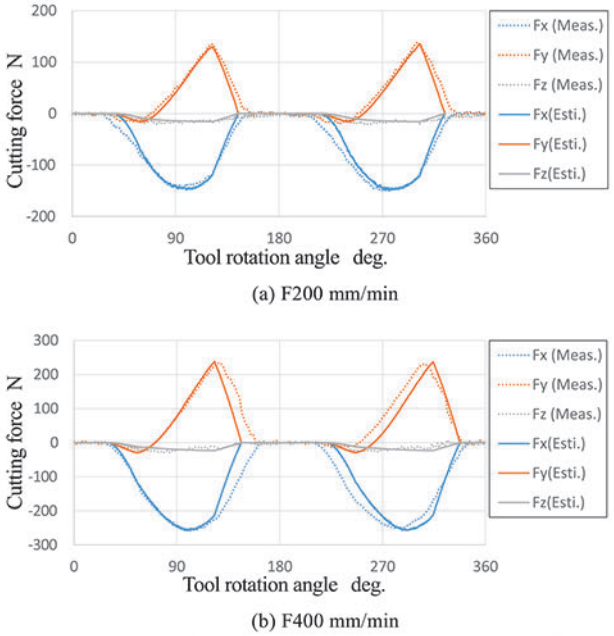


Fig.10 Cutting force simulation developed by this study

5. 結　　言

本研究では，被削材をボクセル形状表現する従来の切削シミュレータの機能を拡張して，工具 1 刃当たりの送り量ごとの解析ではなく，工具微小回転量ごとの解析を可能とする切削シミュレータを新たに開発して，その妥当性を検証した．

(1) 工具を工具軸方向に沿って微小薄板要素に分割して，微小薄板要素ごとに工具切れ刃を離散的に表現するとともに，工具微小回転量ごとの解析を実現した．これにより加工中の工具の静変形や動変形（振動）を考慮した解析に応用が期待できる．



(a) F200 mm/min

(b) F400 mm/min

Fig. 11 Measured and estimated cutting force using square end mill

(2) 工具切れ刃と被削材の干渉判定において，Octree 構造を用いず，除去対象のボクセルを高速に検索する解析手法を開発して，解析の高速化を実現した．これにより，微小時間および微小空間分解能での切削シミュレーションが可能となった．

(3) 切削実験を行って切削力の推定結果と測定結果とを比較して，新しい切削シミュレータの妥当性を検証した．検証の結果，本研究で提案した手法を用いた切削力予測が正しく行われていることが確認できた．

今後は本手法を加工中の工具の静変形や動変形（振動）を考慮した解析に応用してその効果を検証する予定である．

謝　　辞

本研究の一部は科研費(17H03158)の助成を受けたものです．切削加工実験に使用した 5 軸立て形マシニングセンタ（DMG 森精機 NWV1500 DCG）は MTTRF より貸与されました．ここに謝意を表します．

参 考 文 献

1) J.Tlusty et al.: Dynamics of Cutting Forces in End Milling, CIRP Annals, **24**, 1 (1975) 21.
2) D.Mongomery et al.: Mechanism of cutting force and surface generation in dynamic milling, J. of Eng. for industry, **113**, 2 (1991) 21.
3) Y.Altintas et al.: A General Mechanics and Dynamics Model for Helical End Mills, CIRP Annals, **45**, 1 (1996) 59.
4) K.Shirase et al.: Cutting force and dimensional surface error generation in peripheral milling with variable pitch helical end mills, Int. J of Machine Tools and Manufacture, **36**, 5 (1996) 567.
5) 松村 隆　他: 曲線切れ刃形状エンドミル加工における切削力解析（第 1 報），日本機械学会論文集(C), **69**, 688 (2003) 3396.
6) 岩部 洋育　他: 三次元 CAD を活用したボールエンドミルによる切削機構の解析(傾斜面加工における切削面積と評価値による切削特性)，日本機械学会論文集(C), **72**, 713 (2006) 247.
7) 西川 隆敏 他: エンドミル加工の誤差補償システム(第 1 報)，精密工学会誌, **78**, 11 (2012) 975.
8) Takeuchi Y. et al.: Development of a personal CAD/CAM system for mold manufacture based on solid modeling techniques, Ann. CIRP, **38**, 1 (1989) 429.
9) 乾正知: 3 軸数値制御工作機械による曲面加工の高速なシミュレーション，情報処理学会論文誌, **40**, 4 (1999) 1808.
10) 郝 明暉 他: 拡張 Z-map モデルによる CAM システムの開発-オフセ

ット面生成法の一提案-, 精密工学会誌, **60**, 2 (1994) 275.
11) 中本 圭一 他: ボクセルモデルを用いたヴァーチャルマシニングシミュレータの開発, 精密工学会誌, **74**, 12 (2008) 1308.
12) S. Alan et al.: High Accuracy NC Milling Simulation Using Composite Adaptively Sampled Distance Fields, J. of Computer-Aided Design, **44**, (2012) 522.
13) 岸波建史 他: Voxel 表現法の機械加工シミュレータへの応用, 精密工学会誌, **55**, 1 (1989) 105.
14) Balasuabramaniam M. et al.: Generating 5-axis NC roughing paths directly from a tessellated representation, Computer-Aided Design, **32**, 4 (2000) 261.
15) Hauth S. et al.: Extended linked voxel structure for point-to-mesh distance computation and its application to NC collision detection, Computer-Aided Design, **41**, 12 (2009) 896.
16) 長谷川 輝人 他: 被削材のボクセルモデルを用いたエンドミル加工の切削力シミュレーションと切削力の予測結果に基づく適応制御, 精密工学会誌, **82**, 5 (2016) 467.
17) 乾正知 他: 3 方向デクセルモデルによる同時 5 軸制御加工の幾何シミュレーション, 精密工学会誌, **76**, 3 (2010) 361.
18) 土棚善貴 他: Voxel 表現に基づく多軸制御加工切削シミュレーションの大規模並列処理手法, **79**, 5 (2013) 467.
19) Alea GPU, http://www.quantalea.com/ accessed 2017.07.19
20) MSDN Microsoft, https://msdn.microsoft.com/ja-jp/library/system.array.aspx accessed 2017.07.19
21) 成田浩久 他: ヴァーチャルマシニングシミュレータを用いた NC プログラムの評価と修正, 日本機械学会論文集(C 編), **66**, 648 (2000) 2871.
22) 竹内芳美 他: Excel で学ぶ生産加工ソフトウェアの基礎, 日刊工業新聞社, (2011).