２　金型高精度加工システムの開発（第５報）

加工誤差予測シミュレーションの高速化

菊田敬一，西川隆敏，大田耕平，古本浩章，小玉　龍，佐野　誠，門藤至宏，筒本隆博

Development of the system for high accuracy machining of die and mold (5th Report)

Development of the acceleration method for end milling simulation software

KIKUTA Keiichi, NISHIKAWA Takatoshi, OTA Kohei, FURUMOTO Hiroaki, KODAMA Ryu,
SANO Makoto, MONDOU Munehiro and TSUTSUMOTO Takahiro

End mill is generally low rigidity. Tool deflection caused by cutting force, therefore, becomes major factor for machining error. Prediction and compensation of this machining error are effective for machining accuracy improvement. We developed a system which can predict cutting force, dynamic deflection of tools, and machining error. This system is designed for 3-axis NC machining and uses rendering functions of GPU. The acceleration of this end milling simulation becomes important especially for applying this software to the machining of large die. In this study, we developed methods to speed up the collision detection between tool and work-piece, and improved the software to enable parallel distributed processing. Computation time was reduced to 1/18 with four PCs in the large die example.

キーワード：エンドミル，シミュレーション，計算速度，加工誤差，工具たわみ，切削力，GPU

１　緒　　言

金型加工等に用いるエンドミル工具は，一般に径方向剛性が低いため，切削力によって生じる工具のたわみが加工誤差の大きな要因となる。よって，この加工誤差を事前に予測し補償することが，加工精度の向上に有効である。そこで，NC データによる３軸の切削加工を対象として，工具に加わる切削力，動的たわみ，それに伴う加工誤差を予測するシミュレーションソフトウェア [1) 2) 3)] の開発を進めている。このソフトウェアは，予測結果に基づき NC データの送り速度 [4)] や移動座標の指令値を修正する機能がある。

実際の金型加工に使用される NC データは，大規模なものでは工具移動指令（パス）が数百万行に及び，計算に１日以上かかるものもある。企業から要望を聞くと，終業時に計算を開始し，翌朝始業時には終了していることが望ましい，という回答が多かった。このため，ソフトウェアを生産工程に組み入れるためには，計算時間の短縮すなわち計算速度の高速化が重要になる。

そこで本研究では，切削力や加工誤差の予測計算速度を高速化するためのさまざまな機能を開発し，ソフトウェアに実装した。本報では，工具移動軌跡と被削材の干渉判定の高速化に関する手法を中心に，その概要と適用例について述べる。

２　開発手法について

2.1　ソフトウェアの概要

図１に本ソフトウェアのフローチャートを示す。計算には NC データや被削材の初期形状データ，使用する工具のデータ等が必要で，これらを最初に入力する。次に加工誤差の計算ルーチンに入る。まず切削領域の有無を，**図２**に示すように GPU を使って画像で判定する。ここでは，視点を被削材の下から上方へ向け，工具の半球面と被削材，工具移動軌跡の干渉する部分を作図する。工具部分が表示されていれば，これを切削領域として切削力を計算し，以降工具たわみ，加工誤差を計算していく。

加工誤差の推定には，切削面の中で最終的に残る部分，すなわち完成面領域の推定が必要となる。切れ刃がこの完成面領域を通過するときの工具たわみから加工誤差を計算するのである。完成面領域は，対象とする切削後の加工で切削除去される部分，すなわち対象切削より後の工具移動軌跡（以後掃引体と記す）を切削領域から差し引いて決める。金型の形状や大きさによっては，掃引体作成で対象となる NC データが膨大になり，計算に時間がかかることがある。

2.2　高速化の手法

計算時間短縮のため，被削材や NC データの形態に応じて干渉判定を高速化する，次の２つの手法を開発した。

①XY 平面上の移動量が大きい加工に対して，被削材を複数の領域に分割し，領域毎に処理を行う。
②XY 平面上の狭い領域内に Z 軸方向の加工が密集する深堀加工に対して，過去の工具移動軌跡を Z-map 化する。
これにより，干渉判定に必要な描画時間を短縮した。

また，処理する NC データを複数のパソコンに分散して計算時間を短縮した。なお今回適用例として計算に使用した加工誤差予測モデルは，工具を剛体と仮定した剛体切削力・動たわみモデル[5]である。

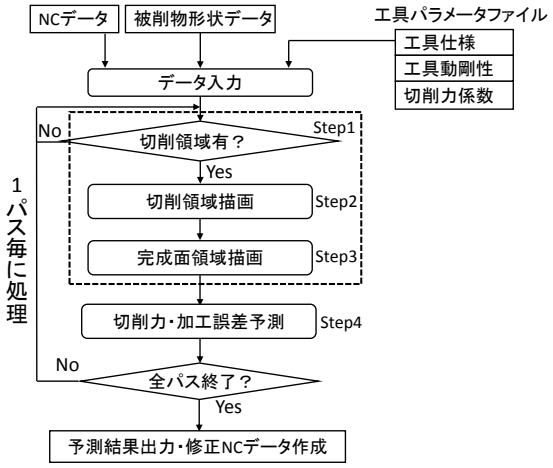

図１　ソフトウェアのフローチャート

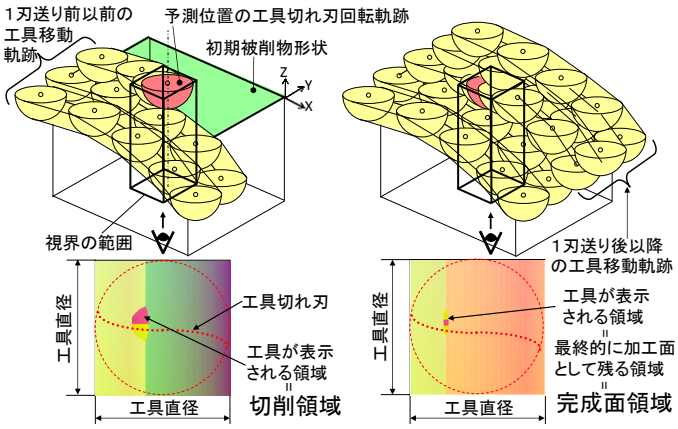

図２　干渉領域描画方法

３　干渉判定の高速化

3.1　手法①：領域分割による高速化

3.1.1　概要

すべての掃引体から切削領域を判定すると，描画に要する時間が非常に長くなる。そこで**図２**の視界内に含まれる可能性のある掃引体を事前に判別し，必要な掃引体のみを描画している。しかし XY 平面上の移動量が大きい NC データの場合，現在の予測位置から遠く離れた掃引体についても，視界に対する内外判定を行う必要がある。そこで，被削材を複数の領域に分割し，その領域内に含まれる掃引体のみを判定対象とした。

領域の幅及び高さは，計算する NC データの加工に使用される工具の中の最大の工具直径に，ユーザが設定する倍率を掛けた長さ以下になるように決定する。次に，この領域に含まれる掃引体の情報のみをソフトウェア上で配列としてシステムメモリに記憶させる。描画の判定は，この配列に記憶された掃引体に対してのみ行うため，領域外掃引体の判定が省略され，処理を高速化できる。領域内の計算対象パスの計算が終了すると，領域の幅または高さの半分をオーバーラップさせた隣接領域の計算へ移行する。被削材全領域の計算が完了するまでこの作業を繰り返す。またこのソフトウェアでは，OpenGL のディスプレイリストや頂点バッファオブジェクトといった，メモリリソースを使って描画を高速化する機能を利用しているが，領域内の掃引体のみを描画対象とすることで，メモリ消費量を削減することができる。なお本手法は，NC データの計算順序が領域の設定に依存し，加工順序とは異なるため，工具の摩耗を考慮する機能[6]は使うことができない。

3.1.2　適用例と効果

表１に示す３種類の薄板プレス用金型加工に使用される NC データ群のうち，直径 30mm から 4mm までの工具を使用する仕上げ工程を対象に，領域分割した場合としなかった場合の計算時間を比較した。分割領域の一辺の長さは，計算する NC データの加工に使用する工具の最大径 30mm の 10 倍以下に設定した。加工 A の場合，被削材のサイズが 620mm×320mm なので，X 軸方向３分割，Y 軸方向２分割となった。**図３**に示すように，これに重なり部分の領域（例えば[1]－②，[2]－①等）を含め，X 軸方向５領域，Y 軸方向３領域の計 15 領域に分割した。同様の設定により，加工 B では 7×9＝63 領域，加工 C では 15×5＝75 領域に分割した。分割領域数が 100 以下となるように設定しているため，加工 C では Y 軸方向の分割数が少なくなっている。使用したパソコンの仕様は，CPU が Corei7 950 3.07GHz，GPU が NVIDIA GeForce GTX680 である。**表１**に示す計算時間から，領域分割有の計算速度は領域分割無に対して，加工 A では約 1.1 倍，加工 B では約 2.1 倍，加工 C では約 4.2 倍速くなり，被削材サイズが大きくパス数が多いほど効果が大きいことがわかる。また計算開始時のソフトウェアのメモリ使用量は，加工 C で分割無の場合 8,400MB，分割有の場合 2,500MB となり，約 70%削減することができた。

３つの加工の，１パスの予測に要した計算時間（計算時間÷計算パス数）を，**図４**のグラフに示す。領域分割無の場合，被削材サイズが大きくパス数が多くなるほど，

1パスの計算時間が長くなる。一方，領域分割有の場合，1パスの計算時間はNCデータの規模が大きくなってもほとんど変化がなかった。これは，領域分割した場合，小規模な被削材の加工を複数個分実行するのと同じ形態となるためと考えられる。

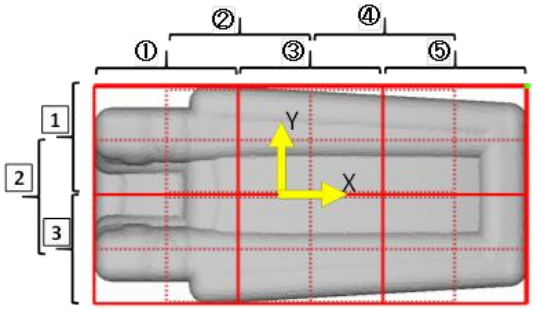

図３　領域分割例

表１　計算した加工情報

	全パス数 (NCファイル数)	計算パス数 (NCファイル数)	被削材サイズ mm	領域分割	計算時間
加工A	477,050 (15)	306,705 (7)	620×320	無	2.10時間
				有	1.84時間
加工B	1,556,382 (43)	1,113,945 (15)	1,150×1,260	無	12.83時間
				有	6.10時間
加工C	3,811,188 (31)	3,396,578 (16)	2,170×1,000	無	92.78時間
				有	22.13時間

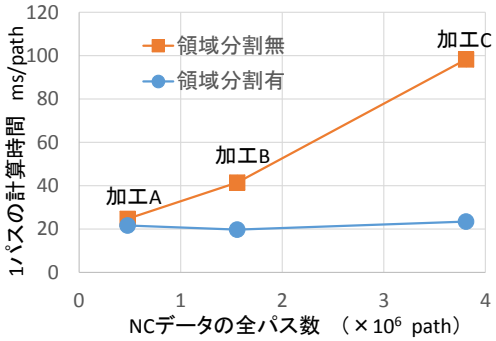

図４　１パスの計算時間の比較

3.2　手法②：掃引体の Z-map 化による高速化

3.2.1　概要

XY 平面状の狭い領域で Z 軸方向の加工が密集する深く掘り下げていくタイプの加工では，視界内に多数の掃引体を描画する必要がある。このために描画速度が低下しやすい。そこで，予測位置以前のすでに加工が済んだ掃引体と被削材初期形状を用いて，加工途中の被削材形状の Z-map を作成し，工具との干渉を判定した。これにより，描画する掃引体数を削減させ，処理速度を高速化した。Z-map とは，XY 平面を格子状に分割し，各格子点に高さ情報を持たせた形状表現方法である。本ソフトウェアでは，画素毎の高さ情報を取得する OpenGL コマンドを利用して Z-map を作成している。

次に Z-map 作成要領を**図５**に基づき説明する。設定画面の[Work 画素数]コンボボックスの入力値 1,024 は，Z-map の XY 平面各方向の格子数となる。格子数を大きくすると高精度な Z-map を作成できるが，計算速度は低下する。 [CalcGroup]テキストボックスには，Z-map を更新するパス数を入力する。Z-map①を作成する場合，作成時点のパスから入力値 3,000 パス後までに工具が移動する範囲を包含する，被削材上の XY 平面内の矩形領域を Z-map①の作成領域とする。次に Z-map①作成時のパスに対して[OverLap]テキストボックス入力値 1,000 パス前までの過去の掃引体形状（Z-map①作成用掃引体）と被削材初期形状から，Z-map①を作成する。[OverLap]パス数を設定できるようにした理由は，予測位置に近いパスを，Z-map より形状精度の高い掃引体として描画するためである。**図５**の予測パスを計算する場合， Z-map①作成用掃引体以降の掃引体を描画し，これと Z-map①を用いて工具干渉状態の判定を行う。Z-map①作成時から 3,000 パスの予測が終了した時点で，Z-map①のデータを破棄し，新しい Z-map②を作成し計算を続行する。この機能を使用する場合，Z-map 作成領域が広くなると，精度が悪化するので，[CalcGroup]の設定値を決める際には注意が必要である。

3.2.2　適用例と効果

被削材のサイズが 370mm×290mm で，Z 軸方向に最大 86mm 掘り込む金型の加工に使用される NC データ群を対象に，本手法の適用有無による計算時間の比較を行った。NC データのパス数は 1,962,673 パスで，すべてのパスの計算を行った。**表２**に示す結果から，本手法適用有の計算速度は，適用無に対して約 5.6 倍となった。また，本手法に加え前章の領域分割（重なり部分を含めて 15 領域）を適用した場合，計算速度は約 0.9 倍とやや遅くなった。このことから，領域分割は被削材サイズ(XY 平面)の大きい場合に対し効果的であることを確認した。

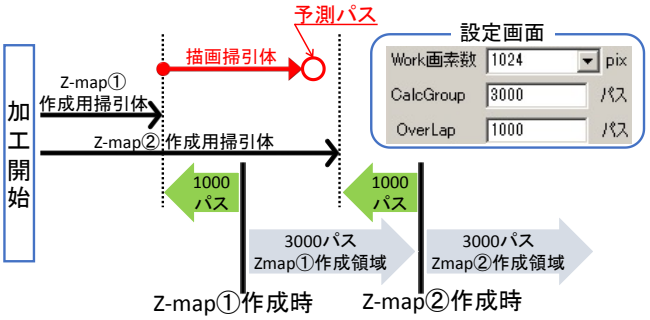

図５　掃引体による Z-map 作成要領

表２　計算時間の比較

高速化手法適用		計算時間
領域分割	Z-map化	
無	無	28.26時間
無	有	5.06時間
有	有	5.53時間

4　複数台パソコン利用による高速化

4.1　概要

大容量の NC データを処理する場合，パソコン１台の処理能力では，計算時間短縮には限界がある。そこで，複数台のパソコンに，なるべく均等になるよう NC データの処理を振り分けて，分散して処理を行うソフトウェアの開発を行った。**図６**に，複数台パソコンによる分散並列処理を行う場合の構成イメージを示す。サーバとなるパソコンには，クライアントパソコンとの通信用ソフトウェアと，設定・結果表示・ユーザインターフェイス等を省略した計算速度重視の専用ソフトウェアをインストールし，通信用ソフトウェアを起動しておく。

クライアントパソコンでは，加工誤差予測ソフトウェアを起動し，全パソコンからアクセス可能なフォルダ内にある，計算に必要な各種データファイルを読み込む。続いて，サーバパソコンに割り当てられた IP アドレスを入力し，各サーバとの通信を確立する。クライアントの加工誤差予測ソフトウェアでは，設定されたサーバの台数に応じて計算負荷がなるべく均一になるよう，どのサーバがどの NC データを処理するか割り振りを行い，それを指示する設定ファイルを生成する。続いて計算を開始すると，各サーバ上で計算専用ソフトウェアが順次自動起動し，割り当てられた設定ファイルを入力して分散処理を開始する。計算終了後に，クライアントパソコンですべての計算結果ファイルを読込後，計算結果の表示や修正パス出力を行う。

4.2　適用例と効果

図７に１台のパソコンで，①加工誤差予測ソフトウェアと，②計算専用ソフトウェアで処理した場合と，４台のパソコンで分散並列処理した場合の計算時間の比較を示す。適用例は**表１**－加工 C のデータを領域分割で予測するモデルとした。４台中最も時間のかかったパソコンを基準として，計算速度は①に対して約 4.5 倍，②に対して約 3.5 倍速くなった。②に対してパソコン台数分の倍率で高速化することができなかった原因として，各 PC の処理データ数の不均衡やパソコン性能の不均衡などが考えられる。

5　結　　言

本研究では，加工誤差予測ソフトウェアの処理速度向上に供する機能の開発・実装と，その他周辺ソフトウェアを製作した。その効果について下記に示す。

（１）領域分割手法を XY 平面上の工具移動量が大きい加工予測に適用した結果，計算速度を最大約 4.2 倍高速化できた。

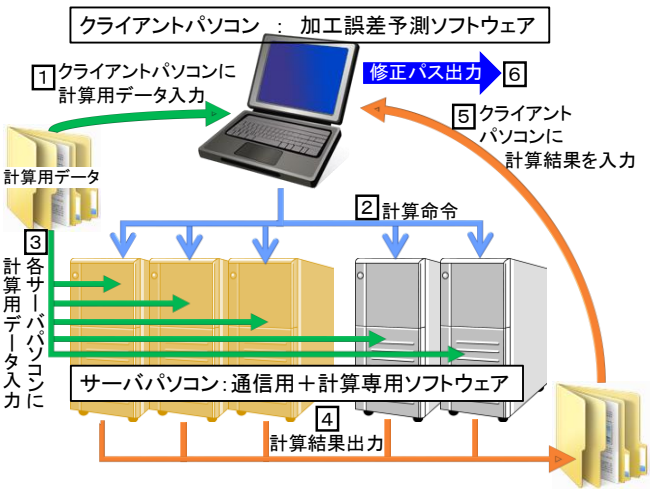

図６　分散並列処理イメージ

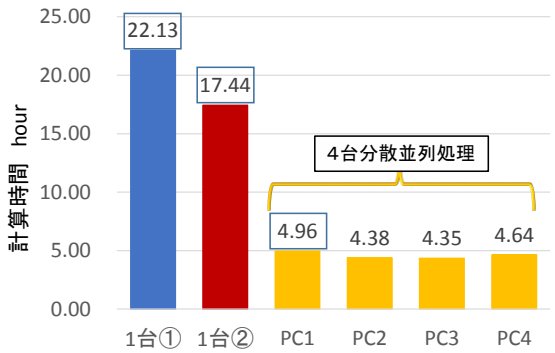

図７　計算時間の比較

（２）掃引体 Z-map 化手法を Z 軸方向に深く掘り下げるタイプの加工予測に適用した結果，計算速度を最大約 5.6 倍高速化できた。

（３）領域分割手法と分散並列処理（４台分）により，92 時間以上かかっていた加工モデルの計算時間を，約５時間に短縮し，計算速度は約 18 倍となった。

今後は，CPU や GPU 等のハードウェアの進歩に期待しつつ，ソフトウェアのアルゴリズムの改良を行い，更なる計算速度向上を図っていく予定である。

文　　献

1) 西川他：精密工学会誌, 78, 11 (2012), 975
2) 菊田他：精密工学会秋季大会講演論文集, (2011), 337
3) 金子他：精密工学会秋季大会講演論文集, (2007), 67
4) 西川他：広島県西部工技研究報告, 54 (2011), 5
5) 西川他：精密工学会秋季大会講演論文集, (2013), 457
6) 西川他：広島県西部工技研究報告, 56 (2013), 5