Pengembangan Modul Pemesinan Awal (*Roughing*) 3-Axis dengan Metode *Modified Copy Finishing* pada Sistem CAM (*Computer Aided Manufacturing*) berbasis Model Faset 3D dengan Simulasi Pergerakan Pahat

Gandjar Kiswanto, Abdurrasyid Mujahid
Departemen Teknik Mesin Universitas Indonesia
Kampus Baru UI – Depok 16424
gandjar_kiswanto@eng.ui.ac.id

Abstrak

Salah satu tahap dalam proses pemesinan adalah 'roughing' atau pemesinan awal. Tahap ini bertujuan mengilangkan material lebih dari benda kerja secepat mungkin mendekati bentuk akhir yang dikehendaki. Laboratorium Teknologi Manufaktur dan Otomasi Departemen Teknik Mesin – Universitas Indonesia mengembangkan sistem-CAM berbasis model faset 3D. Dalam penelitian yang telah dilakukan dikembangkan teknik baru (novel method) dalam pembuatan lintasan pahat roughing yang disebut teknik modified copying dan diimplementasikan pada sistem-CAM tersebut. Teknik ini secara bertahap memotong material lebih dalam setiap layer (lapis) dengan interval vertikal tertentu, dengan lintasan pahatnya mengikuti lintasan pahat hasil finishing. Lintasan pahat finishing disalin ke arah vertikal pada masing-masing layer sehingga terbentuk beberapa lapis lintasan pahat untuk roughing. Untuk memastikan titik-titik potong (modified-copying points) yang dihasilkan, maka perlu visualisasi titik-titik tersebut yang menjadi lintasan pahat roughing, serta simulasi pergerakan pahatnya di atas model produk.

Kata kunci: *sistem CAM, lintasan pahat, roughing, faset 3D, Java, VTK.*

1. PENDAHULUAN

Tahap akhir dalam proses pemesinan adalah tahap *finishing*. Tahap ini bertujuan membentuk material mentah menjadi produk jadi yang rapi, akurat, dan tanpa cacat. Parameter utama dalam proses *finishing* adalah akurasi.

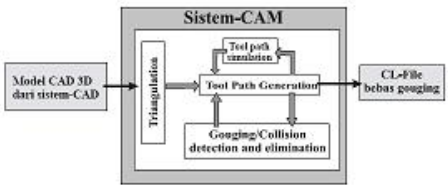
Berbeda halnya dengan tahap *roughing*. Tahap ini dikerjakan sebelum masuk tahap *finishing*. Bagian *roughing* bertujuan membuang material mentah secepat mungkin sedemikian sehingga menjadi bentuk yang mirip dengan model produk yang diharapkan. Paramter utama dalam *roughing* adalah kecepatan dan luas pahat dalam membuang material mentah.

Namun, meskipun *roughing* sebenarnya dikerjakan sebelum tahap *finishing*, teknik *modified copying* yang dikembangkan sebagai salah satu teknik *roughing* dalam tulisan ini membutuhkan lintasan pahat hasil proses *finishing*. Dengan kata lain , prosesnya adalah *roughing*, namun lintasannya diambil dari *finishing*. Maksudnya lintasan akhir *modified-copy finishing* merupakan modifikasi (salinan) dari lintasan proses *finishing*.

Makalah ini akan menjelaskan tahapan pembuatan lintasan pahat dengan metoda *modified copy-finishing.* Bagian pertama dalam paper ini diawali dengan pengenalan arsitektur sistem CAM yang sedang dikembangkan dimana modul *roughing* diintegrasikan. Kemudian dilanjutkan dengan penjelasan algoritma dalam teknik *modified copying*. Langkah-langkah teknis pengerjaan teknik ini kemudian dijelaskan secara singkat dalam impelementasi program menggunakan bahasa Java (J2SDK) beserta visualisasi dan simulasinya. Bagian terakhir ditutup dengan kesimpulan dan rencana kerja berikutnya (*further works*).

2. ARSITEKTUR SISTEM

Dalam [1] dijelaskan bahwa sistem CAM yang dikembangkan membaca model faset yang telah jadi dalam bentuk format STL. Kemudian dilakukan proses pembuatan lintasan pahat diikuti oleh simulasi pergerakan pahat, identifikasi interferensi model pahat dan model faset. Setelah lintasan pahat dinyatakan bebas interferensi (*gouging*) maka *cutter-location file* (CL-file) dapat dibuat. Alur kerja ini dapat secara singkat dapat dilihat pada gambar 1 di bawah ini.



Gambar 1 Alur proses pembuatan lintasan pahat dari sistem-CAD ke sistem-CAM berbasis model faset 3D yang dikembangkan

Untuk itu, modul-modul yang dikembangkan terdiri dari empat buah modul, ditambah kernel (*database*) dan sistem interaksi mesin-manusia atau *graphical user interface* (GUI). Keempat modul tersebut adalah modul simulasi (*simulation module*), modul perencana lintasan pahat (*tool path planning module*), modul pembuat lintasan pahat dan pendeteksi dan penghilang gouging (*tool path generation module*), serta modul pembacaan triangulasi.

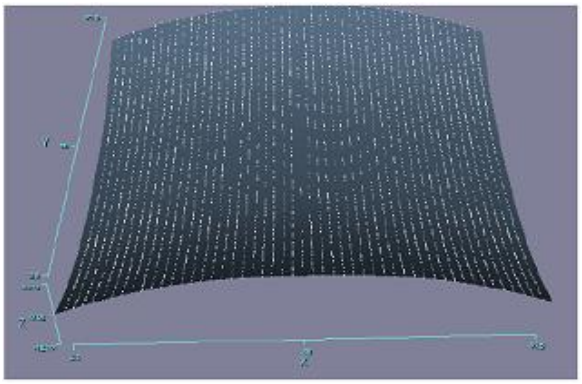
Implementasi *roughing* sendiri dalam tulisan ini termasuk ke dalam modul perencana dan pembuat lintasan pahat dan modul simulasi. Untuk itu, keluaran yang diharapkan adalah titik-titik koordinat yang menjadi lintasan pahat menggunakan teknik *modified copying*, beserta visualisasi titik-titik tersebut dan pergerakan pahat di atas model produk untuk mensimulasikan pekerjaan *roughing* pada mesin NC.

3. ANALISIS TEKNIK *MODIFIED COPYING*

Teknik ini disebut dengan *modified copying* (penyalinan yang dimodifikasi) karena pada dasarnya teknik ini adalah memodifikasi hasil proses *finishing* dengan menyalin (*copy*) titik-titiknya ke dalam beberapa lapis (*slices*). Interval antar lapis (*slices*) disebut dengan *slicing interval*, sekaligus menjadi variabel dalam proses *roughing* termasuk *modified-coopying* ini.

Proses finishing menghendaki hasil akhir berupa material yang sudah dipahat bersih, rapi, tepat, dan tanpa cacat, sesuai modelnya. Proses ini menghasilkan *cutter-contact points* atau sering disingkat dengan *CC-points*. Jadi, CC-points merupakan lintasan pahat tahap terakhir sehingga menghasilkan produk jadi.

Perhatikan contoh CC-points pada gambar di bawah.



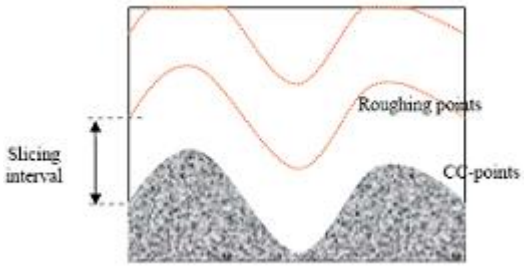
Gambar 2 Contoh visualisasi CC-points hasil finishing

Teknik *modified copying* kemudian menyalin semua titik *x,y,z* dari CC-points sehingga diperoleh sekumpulan titik x,y,z_i dimana i = 1, 2, ..., k. Nilai k bergantung kepada nilai *slicing interval* antara z_i dengan z_{i+1}.

Hal penting yang harus diperhatikan di sini adalah untuk tujuan efisiensi kerja sistem CAM, CC-points yang disalin tidak harus CC-points tahap akhir. CC-points tahap akhir biasanya dibuat setelah melewati *gouging elimination* atau penghilangan/ penghindaran interferensi. CC-points yang disalin bisa dari CC-points yang masih ada kemungkinan interferensi, namun supaya lebih aman lintasan pahat hasil salinan disesuaikan (misalnya diangkat dengan tinggi tertentu) sehingga dipastikan tidak akan menabrak material produk atau terjadi interferensi.

Langkah-langkah untuk menghasilkan lintasan pahat *modified copy finishing* adalah sebagai berikut:

1) Tentukan nilai *step over* ***SO*** dan *slicing interval* ***SI***. *Step over* adalah kerapatan jarak makan pahat. Pada gambar 2 di atas, jarak antar garis putih ditentukan dari nilai *step over* ini.
2) Tentukan arah awal pemotongan, misalnya dari y_{min} ke y_{max} dan dari x_{min} ke x_{max}. Arah pemotongan berikutnya harus bisa berubah secara otomatis, sehingga membentuk zig-zag.
3) Pada setiap z_i, salin (*copy*) CC-points dengan mengubah setiap titik *x,y,z* CCpoints menjadi *x,y,z'* kemudian simpan dalam sebuah struktur data *Vector*. Nilai *z'* diperoleh dengan menambahkan *z* dengan nilai *slicing interval*.
4) Yang perlu diperhatikan adalah jika *z'* melebihi koordinat z_{max} maka titik tersebut tidak perlu disimpan. Perhatikan contohnya pada gambar berikut.



Gambar 3 Ilustasi konsep roughing dengan teknik modifed copying

Titik-titik yang dihasilkan ini selanjutnya akan digunakan untuk visualisasi dan simulasi proses *roughing* dengan teknik tersebut.

4. IMPLEMENTASI PEMROGRAMAN

Teknik ini diimplementasikan dalam bahasa pemrograman *Java* menggunakan J2SDK versi 1.5, sesuai dengan rancangan keseluruhan pengembangan sistem CAM berbasis model faset 3D ini.

Teknik *modified copying* ini diimplementasikan dalam sebuah *method* yang diberi nama `modifiedCopying()`. Di dalam method ini, ada beberapa variabel lokal yang terlebih dahulu harus ditentukan nilainya, antara lain:

- `step_over` : interval antara y1 dan y2
- `slicing_interval` : interval antar *layer*
- `maxCoord` : sebuah array yang menyimpan koordinat *x*, *y*, dan *z* (saling lepas) tertinggi dari titik-titik dalam model.
- `minCoord` : sama halnya dengan `maxCoord` untuk koordinat *x*,*y*,*z* terendah.
- `min_to_max_y` : variabel *boolean* untuk menentukan arah lintasan pahat apakah dari Y_{max} ke Y_{min} atau sebaliknya.
- `min_to_max_x` : variabel *boolean* untuk menentukan arah lintasan pahat apakah dari X_{max} ke X_{min} atau sebaliknya.
- `Vector` : sebuah instance dari strutktur data jenis *Vector* untuk menyimpan titik-titik hasil *roughing*.

Selanjutnya implementasi *method* ini dibuat ringkasannya sebagai berikut.

```
public synchronized void modifiedCopying(){
    /* inisialisasi variable */
    // ...
    /* mengambil cc-points dari finishing */
    Vector vertexCP= machining.getVertexCP();
    /* looping dari z max hingga z min dengan interval sebesar slicing interval */
    for(float z= maxCord[2]- Configuration.SLICING_INTERVAL;
            z> minCord[2]+0.5f;
            z-= Configuration.SLICING_INTERVAL)
    {
        /* lintasan pahat searah sumbu y (inner path).
           Arah step over searah sumbu x (outer path) */
        if(min_to_max_x){   /* outer path: path dari x min ke x max; */
            for(int i=0; i<vertexCP.size(); i++){
                // ...
                if(min_to_max_y){   /* inner path: path dari y min ke y max; */
                    while(iterator.hasNext()){
                        //...
                        if(v.getZ()+add_z < maxCord[2])
                            /* jika koordinat z ditambah slicing interval kurang
                               dari z max, masukkan ke dalam vector */
                    }
                } else{ /* path dari y max ke y min; */
                    // ...
                    while(iterator.hasNext()){
                        // ...
                        if(v.getZ()+add_z < maxCord[2])
                        // ...
                    }
                }
                min_to_max_y= !min_to_max_y;    /* ubah arah path; */
            }
        } else{     /* path dari x max ke x min; */
            for(int i=vertexCP.size()-1; i>=0; i--){
                // ...

                /* same as above; */

                if(min_to_max_y){   /* inner path: path dari y min ke y max; */
                    while(iterator.hasNext()){
                        // ...
                    }
                } else{ /* path dari y max ke y min; */
                    // ...
                }
                min_to_max_y= !min_to_max_y;    /* ubah arah path; */
            }
        }
        min_to_max_x= !min_to_max_x;
    }
}
```

Gambar 4 Implementasi method modified copying.

5. VISUALISASI DAN SIMULASI

Visualization Toolkit (VTK)

Implementasi proses *roughing* beserta visualisasi dan simulasi bertujuan untuk memberikan tampak visual bahwa titik-titik yang dihasilkan adalah benar. Untuk visualisasi dan simulasi ini, digunakan *class-class* dari *library* VTK (*Visualization Toolkit*). VTK sangat banyak digunakan untuk keperluan *image processing*, visualisasi dan simulasi proses, *rendering* objek, dan sejenisnya.

VTK dikembangkan pertama kali untuk bahasa pemrograman *C++*. Namun, karena semakin tingginya kebutuhan untuk dapat menggunakan VTK pada bahasa pemrograman yang lain, maka VTK melakukan *porting* ke dalam bahasa pemrograman yang lain. Bahasa pemrograman tersebut antara lain: *Java*, *Python*, dan *Tool Command Language*(*TCL*).

Tidak hanya dapat berjalan pada bahasa pemrograman lain, *vtk* juga telah tersedia untuk komputer dengan *Operating System (OS)* yang berbeda, bahkan telah tesedia untuk komputer dengan arsitektur yang berbeda dengan arsitektur komputer *PC*. VTK telah dapat berjalan dengan baik pada *OS Windows*, dan berjalan cukup baik untuk komputer dengan *OS* turunan *UNIX*. VTK juga dapat berjalan dengan cukup baik, meskipun masih terdapat banyak kekurangan pada komputer dengan arsitektur *Power PC* dengan *Mac Tiger OSX*.

Meskipun *library* VTK memiliki lisensi *open source* yang identik dengan *OS Linux*, namun dukungan secara penuh masih dipegang oleh bahasa pemrograman *C++* dengan *OS Windows*. Hal tersebut dapat dilihat karena semua program *vtk* yang ditulis dalam bahasa apapun akan kembali di*porting* ke dalam bahasa pemrograman *C++*. Untuk bahasa pemrograman Java, *libarary* untuk mengakses kelas-kelas VTK terdapat dalam *archieve vtk.jar*.

Visualisasi dan simulasi

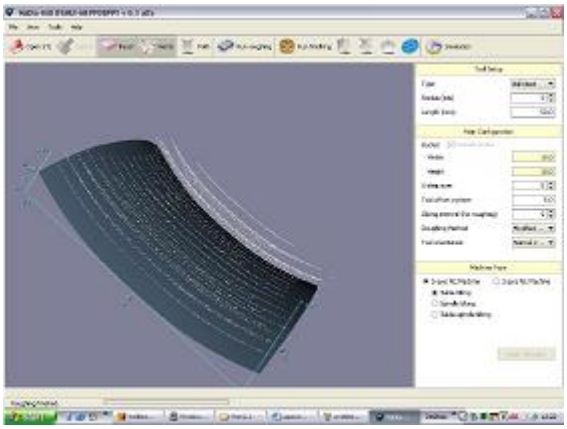
Untuk melakukan visualisasi, beberapa class dalam VTK yang diperlukan antara lain: `vtkPoints`, `vtkPolyVertex`, `vtkUnstructuredGrid`, `vtkDataSetMapper`, `vtkActor`, `vtkRenderer`, dan `vtkRenderWindow`.

Titik-titik *modified-copying roughing* yang sudah dihasilkan disimpan dalam struktur data `vtkPolyVertex` dalam bentuk (tipe data) `vtkPoints`. Selanjutnya untuk menampilkannya secara grafis di layar, digunakan class `vtkActor` dan `vtkRenderer` untuk merender objek-objek dari titik tersebut. `vtkRenderWindow` sendiri berperan ibarat sebagai *stage* yang menjadi tempat objek tadi divisualisasikan.

Adapun untuk simulasi, class-class yang digunakan sama dengan simulasi, ditambah dengan `vtkLineSource`, `vtkTubeFilter` dan `vtkSphereSource`. Ketiga class tambahan ini digunakan untuk memodelkan bentuk pahat yang digambarkan dengan model silinder ditambah setengah bola di ujungnya.

Simulasi pergerakan pahat berarti pergerakan objek silinder tersebut di atas model produk. Untuk menghasilkan efek silinder yang bergerak, maka dilakukan *looping* perubahan titik pusat silinder sesuai dengan titik-titik potong yang dilaluinya. Dengan demikian akan tampak pergerakan pahat di atas titik-titik *modified-copying roughing*.

Di bawah ini adalah *screenshot* implementasi visualisasi *modified-copying roughing* dan simulasi pergerakan pahatnya.

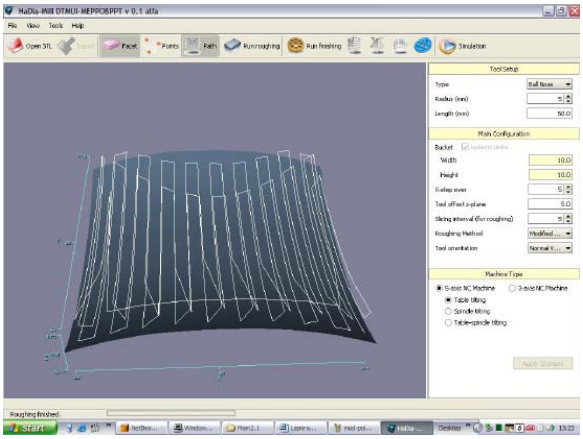


Gambar 5 Visualisasi titik-titik modified-copying

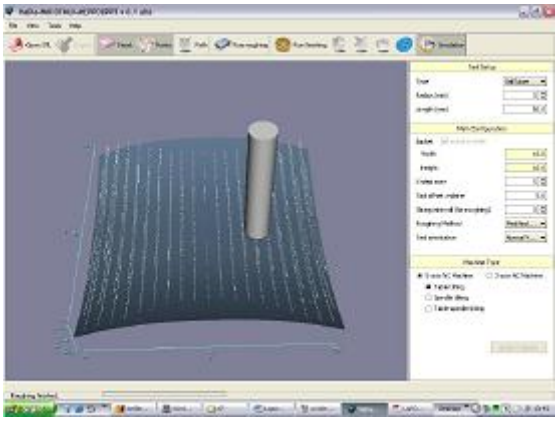
Titik-titik putih pada gambar di atas adalah titik-titik hasil *modified-copying*. Titik-titik ini terlihat memiliki kemiringan sama dengan kontur permukaan model, sebab titik-titik tersebut memang hasil salinan titik-titik pada permukaan model.

Pada gambar 6 di bawah, titik-titik *modified copying* divisualisasikan dengan garis yang menjadi lintasan pahat. Pada gambar tersebut tampak lintasan pahat berjalan seperti alur zig-zag, ke arah X_{max} (menjauhi pengamat) dan ke arah X_{min} (mendekati pengamat) secara bergantian.

Adapun pada gambar 7 di bawah adalah hasil pengambilan gambar (*screenshot*) pada saat simulasi pergerakan pahat berlangsung. Pahat dimodelkan dengan bangun silinder berwarna abu-abu, dengan bangun setengah bola pada ujungnya (model pahat *end-ball*).



Gambar 6 Visualisasi lintasan pahat modified-copying roughing



Gambar 8 Simulasi pergerakan pahat pada modified-copying roughing

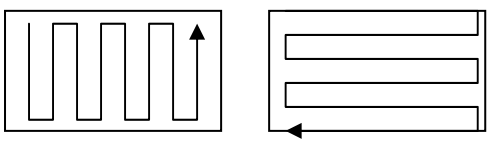
Ketiga gambar di atas adalah hasil implementasi pemrograman menggunakan bahasa Java J2SDK dipadukan dengan *clas-class* dari *library* VTK untuk keperluan visualisasi dan simulasi.

6. KESIMPULAN DAN PENGEMBANGAN LANJUTAN

Dari penjelasan-penjelasan di atas, *modified-copying roughing* adalah salah satu teknik yang digunakan untuk memakan material dalam waktu cepat dan dengan cakupan wilayah ‘makan’ yang cukup luas.

Arah pemotongan zig-zag mengikuti kontur permukaan model produk dapat dengan cepat membuang material, karena pahat bergerak secara parallel ke arah titik-titik yang paling dekat. Ukuran diameter pahat bisa disesuaikan dengan kebutuhan untuk mempercepat waktu membuang material.

Saran untuk pengembangan berikutnya adalah adaptasi arah pemotongan sesuai dengan bentuk model produk. Artinya, jika pada paper ini arah pemotongan berjalan searah sumbu koordinat Y, maka bisa dicoba arah lain yang searah sumbu koordinat X. Perhatikan gambarnya berikut ini.



Gambar 9 Variasi arah pemotongan (a) vertikal (b) horizontal

Ada beberapa kondisi di mana arah pemotongan vertikal bekerja lebih efisien dibandingkah arah horizontal. Demikian pula sebaliknya. Untuk itu, pemilihan arah yang tepat dapat memaksimalkan hasil proses *modified-copying roughing*.

DAFTAR ACUAN

[1] Kiswanto, Gandjar. (2005). **Pengembangan dan Pembuatan Sistem CAM (Computer-Aided Manufacturing) yang Handal berbasis Model Faset 3D untuk Pemesinan Multi-axis dengan Optimasi Orientasi Pahat dan Segmentasi Area dan Arah Pemesinan**, Laporan Kemajuan RUT XII Tahap II Tahun 2005, Kementrian Riset dan Teknologi dan LIPI.

[2] www.vtk.org