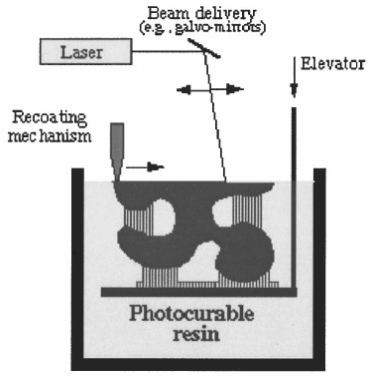
BAB II
RAPID PROTOTYPING

2.1 Klasifikasi *Rapid Prototyping*

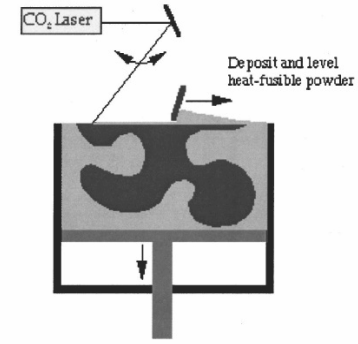
Rapid Prototyping (RP) atau *Layered Manufacturing* (LM) merupakan proses fabrikasi suatu produk dengan *layer by layer*. Prosesnya melibatkan penambahan *raw material* berturut-turut pada *layer*, sampai terbentuk produk yang sesuai. Ada empat klasifikasi utama pada RP atau LM, yaitu [1]:

- *Photopolymer-based*, penambahan material secara photopolimerisasi, seperti: *Streolytography* (SLA), *Solid Ground Curing* (SGC).
- *Deposition-based*, secara fisik material diendapkan, seperti: *Extruded Deposition-based* (ED), *Inkjet Deposition-based* (ID), *Shape Deposition Manufacturing* (SDM).
- *Powder-based*, metode pengikatan serbuk, seperti: *Selective Laser Sintering* (SLS), *Selective Inkjet Binding* (SIB).
- *Lamination-based*, metode lapisan, seperti: *Laminated Object Manufacturing* (LOM).

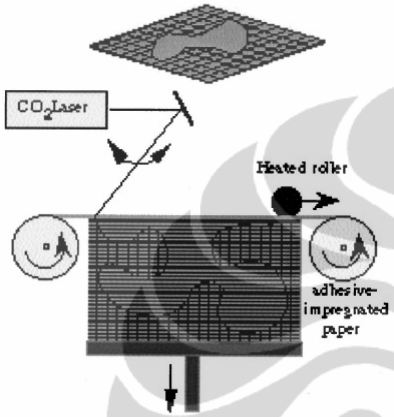
Pada *photopolymer-based*, proses pembuatan produk menggunakan sinar untuk solidifikasi resin secara selekif. Material resin dalam bentuk cair akan berikatan dan membeku setelah terkena sinar laser (gambar II.1a). Proses pembuatan produk tipe *powder-based* melalui pengikatan (*sintering*) serbuk halus (gambar II.1b). Sinar laser-CO_2 memancar ke permukaan layer yang dipilih hingga serbuk berikatan membentuk lapisan satu demi satu sampai permukaan layer bagian atas. Penambahan material dengan pengisian dari filamen thermoplastik yang terlebih dahulu dipanaskan merupakan gambaran tipe prototyping *deposition-based* (gambar II.1d, 1e). Prosesnya merupakan representasi printer 3D yang mengendapkan filamen ke layer secara kontinyu hingga permukaan atas layer. Sedangkan pada *laminated-based* (gambar II.1c), material filamen berbentuk lembaran yang jika terkena sinar laser-CO_2 akan terendapkan dan mengisi bagian lapisan produk secara bertahap.



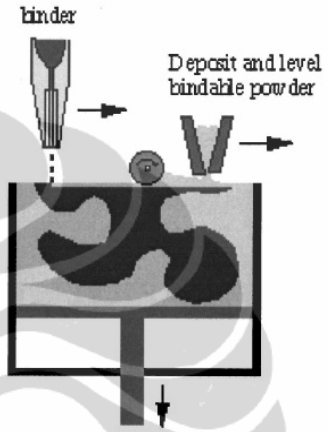
a. Streolytography (SLA)



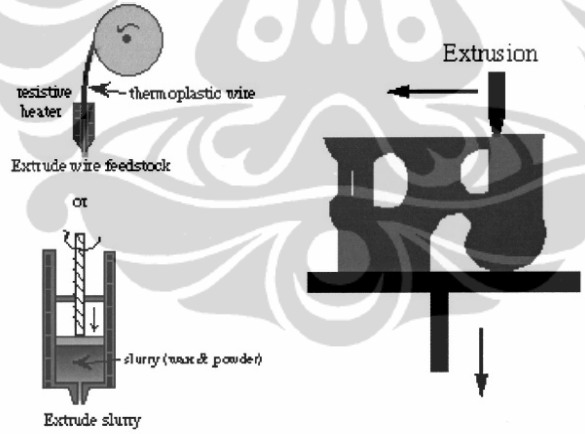
b. Selective Laser Sintering (SLS)



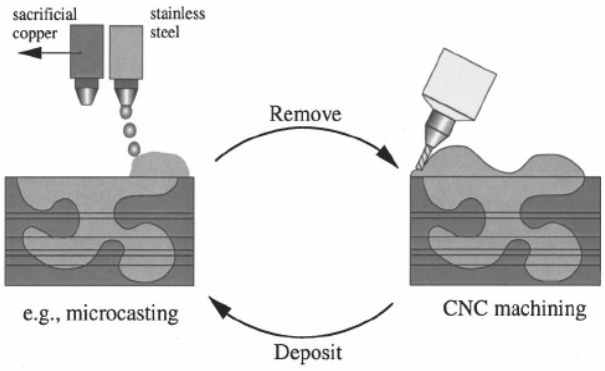
c. Laminated Object Manufacturing (LOM)



d. Inkjet Deposition-based (ID)



e. Extruded Deposition-based (ED)



f. Shape Deposition Manufacturing (SDM)

Gambar II.1 : Macam-macam proses RP [14]

Beberapa mesin RP yang ada menunjukkan bahwa ketebalan layer minimum bervariasi tergantung keakuratan mesin. Tabel II.1 menunjukkan karakteristik dari beberapa proses *rapid prototyping*.

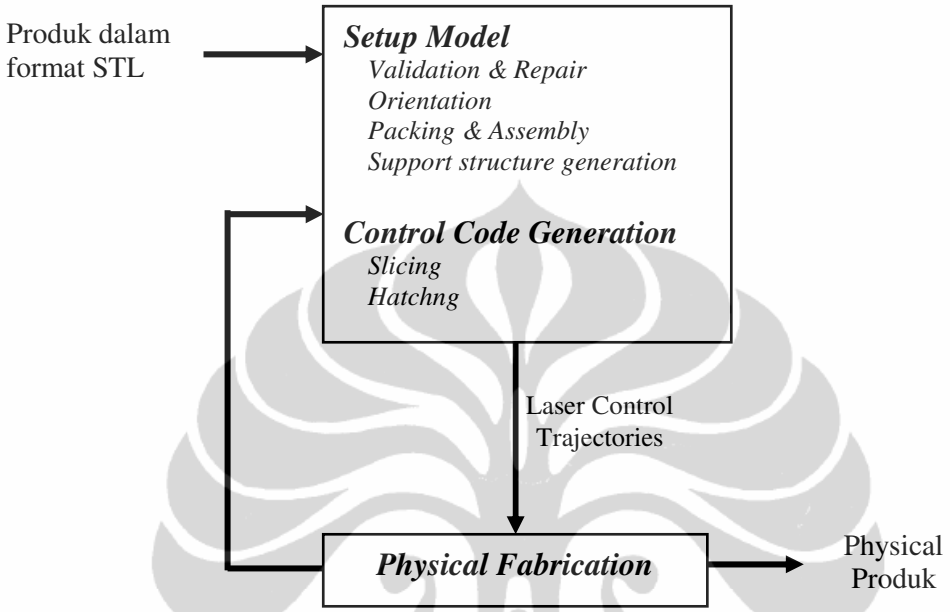
Tabel II.1 : Karakteristik proses rapid prototyping [14]

Proses	Ukuran Pembuatan	Akurasi	Material	Objek Heterogen
Lithography (SLA)	Sampai dengan 500x500x580 mm	±0.05 mm + 0.0015 mm/mm	Photo-polymers	Secara theori mampu, tetapi hanya pada topik riset
Laser Fusion (SLS)	380x330x460 mm	±0.1 mm	Plastic (PC,nylon, Polyamide) and Steel	Dalam pengembangan
Laminated Object Manufacturing (LOM)	Sampai dengan 810x560x500 mm	±0.5 mm	Paper, plastic sheet, beberapa jenis keramik	Tidak
Extrusion (FDM)	Sampai dengan 600x500x600 mm	±0.13 mm + 0.0015 mm/mm	ABS, Elastomer, Wax	Tidak
Inj-Jet Printing	Z Corporation 200x250x200 mm	±0.5 mm	Starch, Plaster, berbagai infiltrants	Dalam pengembangan
	Extrude Hone 305x305x250 mm	±0.13 mm	Stainless Steels dengan Bronze	Tidak
Shape Deposition Manufacturing (SDM)		Material dan ukuran tak-bebas	Berbagai polymer, keramik, dan logam	Ya
Direct Metal Deposition	900x450x450 mm	500 mikron	Semua logam	Ya

2.2 Proses *Rapid Prototyping*

Proses *rapid prototyping* diawali dengan validasi model *CAD* tiga dimensi suatu produk, langkah ini dilakukan untuk memastikan bentuknya *solid*. Model yang sudah *valid* kemudian diorientasikan terhadap ruang pembuatan (*parts orientation*), dengan mempertimbangkan waktu pembuatan dan kualitas permukaan. Beberapa model dapat digabung menjadi satu bangunan *asembly* untuk efisiensi penggunaan mesin dan material. Berdasarkan pada persyaratan prosesnya, dukungan struktur dapat ditambahkan ke model jika diperlukan. Setelah validasi, kemudian model dipotong dengan bidang horisontal. Tiap bidang horisontal menghasilkan bidang potong sebagai penentu *laser trajectory* untuk mengontrol proses *sintering* atau solidifikasi.

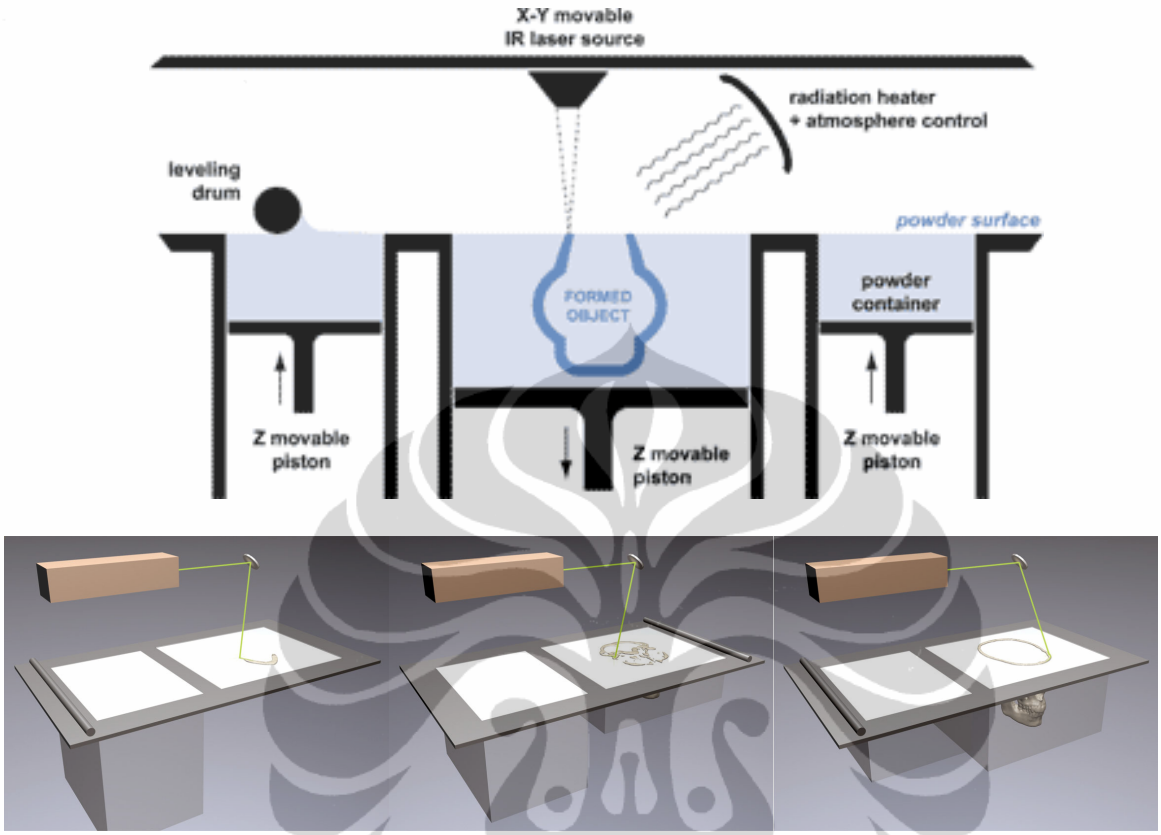
Langkah utama untuk proses planning termasuk orientasi, *generate* struktur pendukung jika diperlukan, *slicing* dan pemilihan parameter proses.



Gambar II.2 : Diagram proses rapid prototyping [6].

Perencanaan proses dilakukan untuk memilih parameter proses dan pembuatan instruksi kontrol untuk fabrikasi produk. Umumnya desainer menyelesaikan perencanaan proses dengan mempelajari produk dan persyaratan kualitas, yang tentunya sangat memakan waktu.

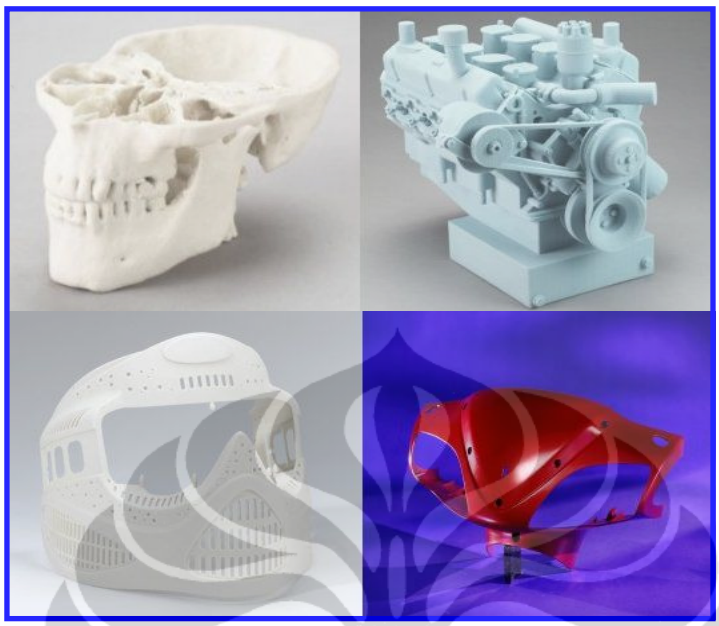
Oleh karena itu, *rapid prototyping* membutuhkan otomasi proses. Ini dapat dicapai dengan manghubungkan pemahaman desainer dan membuat keputusan dengan proses fisik untuk membuat produk dengan kualitas yang diinginkan. Otomasi perencanaan proses juga salah satu tujuan dasar RP [3]. Otomasi ini bertujuan untuk membuat bentuk tiga dimensi yang kompleks, untuk menggunakan mesin fabrikasi *generic* yang tidak membutuhkan *fixture* khusus atau *tooling*, untuk membuat perencanaan proses secara otomatis didasarkan pada model CAD, dan untuk meminimalkan kesalahan manusia.



Laser bekerja sesuai lintasan dari tiap layer. Disetiap layer sinar laser akan melakukan proses pengikatan (sintering) serbuk-serbuk halus yang sudah diratakan oleh roller. Serbuk yang tidak terkena sinar akan tetap dan tidak berikatan. Proses sintering ini dilakukan untuk setiap layer. Meja kerja bergerak turun setelah satu kontur tiap layer telah dilalui dan disinari laser. Proses akan berulang untuk kontur layer selanjutnya.

Gambar II.3 : Proses Selective Laser Sintering (SLS)[16]

RP memfasilitasi pemenuhan dari dua tujuan utama diatas. Bagaimanapun, hal ini membutuhkan jumlah yang signifikan dari intervensi manusia untuk memproduki produk yang optimal. Optimasi tergantung pada persyaratan fungsional, termasuk akurasi, waktu pembuatan, kekuatan, dan efisiensi. Persyaratan kualitas, bagaimanapun bervariasi dari arahan visual ke master pola untuk proses kedua. Karenanya, perlu kontrol kualitas dari seorang ahli untuk memproduksi produk dengan kualitas yang konsisten.



Gambar II.4 : Contoh produk prototyping[16]

2.3 Parameter Proses *Rapid Prototyping*

Membuat kode pengontrolan otomatis untuk persyaratan yang diinginkan adalah suatu aspek yang dimunculkan pada RP. Diane et al. [4] mengklasifikasikan parameter proses RP, yaitu:

- Parameter gangguan (*nuisance*), seperti: umur laser, keakuratan posisi *beam*, kelembaban dan temperatur yang tidak terkontrol pada analisa eksperimental tetapi memberikan beberapa pengaruh terhadap produk.
- Parameter konstan, seperti: diameter *laser beam*, fokus laser, dan sifat-sifat material, dan
- Parameter control, parameter ini mempengaruhi keluaran proses dan pengaturan ketika dijalankan seperti: *layer thickness*, *hatch space*, orientasi produk, penyusutan material dan kompensasi lebar batang laser.

Diane et al. [4] menyatakan bahwa *layer thickness*, *hatch space,* orientasi produk dan kedalaman penanganan adalah bagian yang vital pada parameter kontrol. Mereka melakukan eksperimental dengan *hatch space*, orientasi produk,

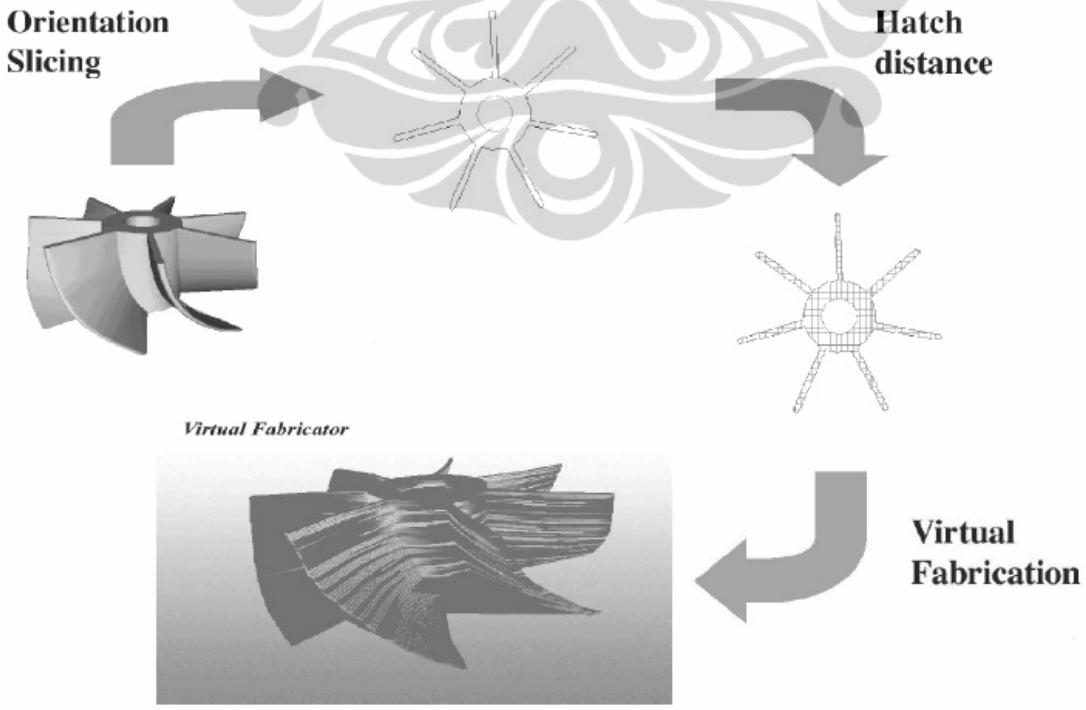
layer thickness serta kedalaman penanganan yang berlebih dan menetapkan pengaruhnya pada kualitas produk *streolithography* (SLA) adalah cukup signifikan.

Zhou dan Hersovici [5] menjelaskan masalah keakuratan pada proses SLA, dan menemukan bahwa ketebalan layer, *hatch space*, gap, penanganan berlebih, dan posisi pada bidang pembuatan dari proses SLA adalah fokus pengontrolan keakuratan. Mereka menggunakan metode *taguchi* untuk mencari hubungan fungsional antara kombinasi yang berbeda dari faktor kontrol dan kualitas produk untuk bentuk permukaan yang standar. Bagaimanapun, ekstrapolasi hasil ini untuk permukaan produk *rapid prototyping* yang komplek adalah sangat sulit.

Waktu pembuatan, keakuratan permukaan dan persyaratan efisiensi dari proses *rapid protoyping* sangat besar ditentukan oleh orientasi, *layer thickness* dan parameter *hatch space*. Orientasi produk RP mempengaruhi keakuratan dan waktu pembuatan. Orientasi produk dalam arah yang optimal akan memberikan sudut yang relatif kecil antara *facet* dan arah pembuatan, yang menghasilkan keakuratan permukaan yang tinggi. Waktu pembuatan produk adalah sebanding dengan ketinggian-z dalam arah pembuatan. Orientasi produk dengan tinggi-z minimum akan menghasilkan *slice* yang sedikit, dan karenanya, mengurangi waktu pembuatan.

Ketebalan *layer* mempengaruhi keakuratan dan waktu pembuatan. Keakuratan permukaan akan diperbaiki ketika produk dibuat dengan dengan ketebalan sangat kecil, tetapi waktu pembuatan akan naik secara kebalikannya. Dengan kata lain, produk akan dibuat lebih cepat dengan ketebalan yang besar yang menghasilkan penurunan keakuratan, terutama sekali pada daerah kurvatur yang tinggi. *Hatch space* merupakan jarak antara vector pararel yang digunakan untuk solidifikasi permukaan *layer*. *Hatch space* yang besar mengurangi waktu pembuatan. Bagaimanapun, jika *Hatch space* terlalu besar, material produk dalam layer tidak dapat disinter. Oleh karenanya, penting untuk menetapkan *hatch space* supaya waktu pembuatan menjadi minimum dan layer penyinteran berjalan dengan baik.

Choi S.H. [6] menjelaskan bahwa pengaruh parameter control pada persyaratan adalah berubah-ubah dari satu proses ke proses lainnya. Sebagai langkah awal, digunakan untuk memulai dengan model matematik yang relatif simpel, yang hanya menyertakan parameter proses *independent*, dan sesudah itu ditingkatkan dengan karakteristik proses individual. Ada perbedaan pendekatan untuk menghitung persyaratan dengan memperhatikan terhadap parameter control seperti keakuratan permukaan. Keakuratan permukaan dapat dijelaskan sebagai deviasi geometri dari model CAD sebelumnya terhadap produk yang menyebabkan kerugian keakuratan. Kerugian keakuratan ini pertama tergantung pada proses awal karena pertukaran data pada sistim CAD. Kerugian kedua pada tahap proses perencanaan dimana pada bagain produk berkontur akan terbentuk effek tangga bertingkat (*stair-step*) yang tampak jelas akibat layer thickness yang besar. Dan kerugian ketiga pada saat proses, kerugian pada saat proses terutama pada mekanisme penggerakan (*deliver*) laser dan sudut dorongan (*induced angle*) dengan permukaan produk dan kerugian akibat proses solidifikasi.



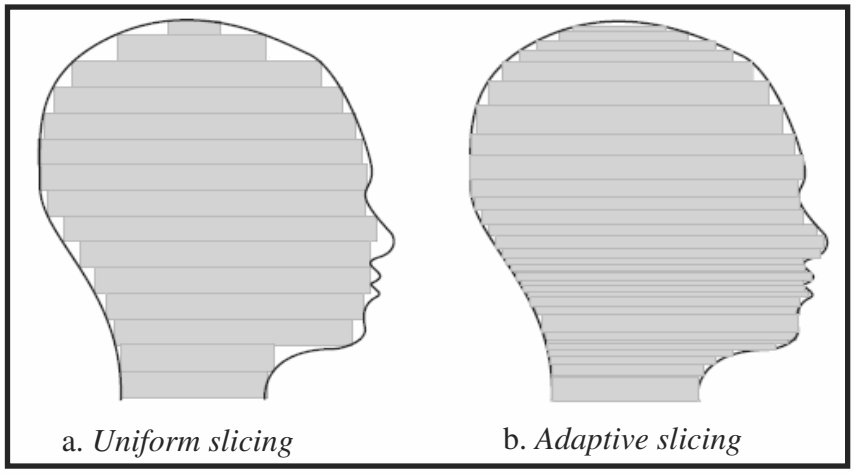
Gambar II.5 : Simulasi Virtual proses rapid prototyping [15].

2.4 Reviuw *Slicing* dan Pembuatan Lintasan

Setiap teknologi *rapid prototyping* memiliki spesifikasi sendiri-sendiri [13]. Ada yang berbentuk formasi dinding terluar, metode pengisian, dan pemisahan produk dari material pelingkup yang menentukan pola lintasan mesin. Formasi dinding terluar merupakan pola yang saat ini banyak dipakai pada mesin *prototyping*, formasi ini menghasilkan produk yang cepat dari segi proses pembuatannya dan sedikit membutuhkan material. *Prototype* yang dihasilkan sangat lemah dari segi kekuatan sehingga tidak cocok untuk diberi pembebanan. Formasi dengan metode pengisian seluruhnya akan menghasilkan produk yang sesungguhnya, formasi ini prosesnya akan semakin lama bila dibandingkan dengan formasi dinding terluar. Bila tujuannya untuk membuat *protoype* yang kuat seperti produk yang diinginkan formasi ini sangat diperlukan.

Pola lintasan mesin rapid *prototyping* dapat digerakkan secara robotik pada bidang-XY untuk Mesin FDM, pola laser untuk solidifikasi dan *sintering* material pada Mesin SLA dan SLS, atau pola pisau laser untuk Mesin LOM. Proses-proses ini membutuhkan strategi pembuatan lintasan mesin (*machine path*) yang berbeda. Beberapa pendekatan pembuatan proses *slicing* dan *NC-path* sudah diusulkan dan diimplementasikan ke karakteristik khusus dan kebutuhan-kebutuhan berbagai proses RP.

Pendekatan-pendekatan proses *slicing* dikategorikan kedalam empat kelompok [9], yaitu:



Gambar II.6 : Metode slicing pada RP [7].

a. Metode *slicing* model file STL dengan ketebalan layer seragam (*uniform*).
b. Metode *slicing* model file STL dengan ketebalan layer *adaptive*.
c. Metode *slicing* model CAD dengan ketebalan layer *adaptive*.
d. Metode *slicing* dengan perhitungan kontur yang tepat.

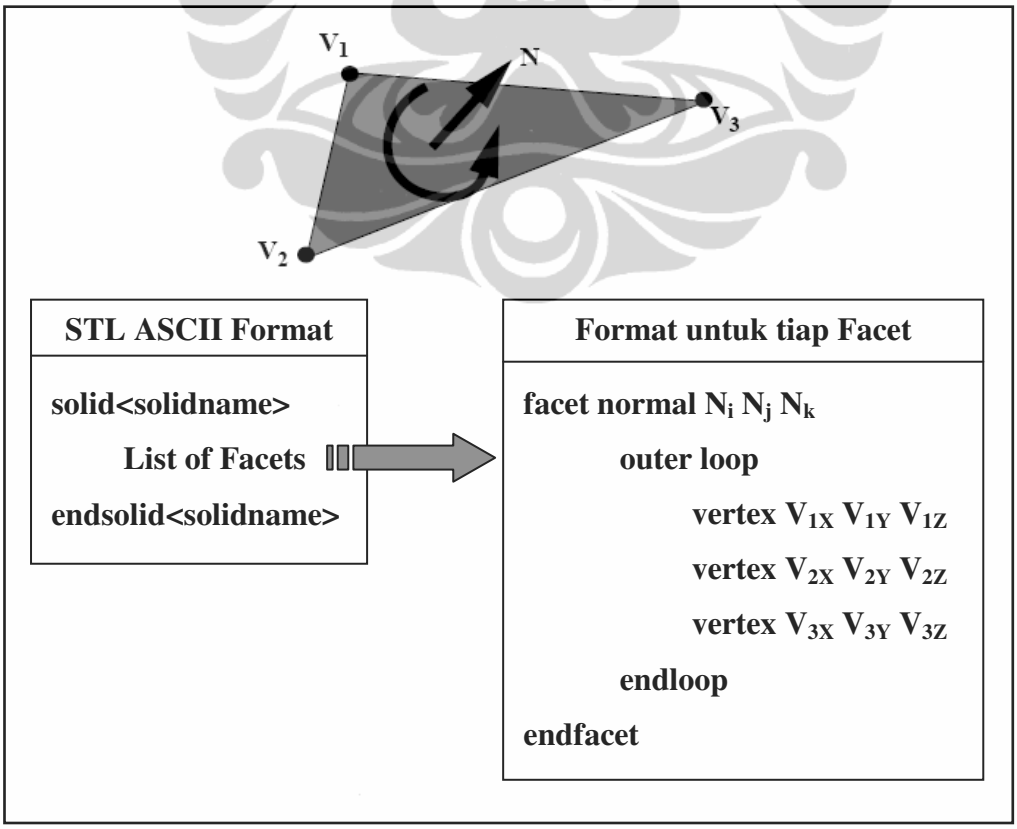
Metode *slicing* ketebalan layer seragam, semua layer memiliki ketebalan yang sama, sedangkan metode *slicing* ketebalan layer *adaptive*, ketebalan layer akan bervariasi menurut kompleksitas geometri. Ada empat konfigurasi yang mungkin ketika melakukan *slicing* terhadap model telah dikembangkan oleh Kulkarni dan Dutta [11] untuk prosedur *slicing* yang lebih akurat pada daerah *convex dan concave curvature*. Pendekatan bertujuan menentukan ketebalan layer yang dihitung berdasarkan kurva geometri, sehingga pendekatan ini kurang cocok dilakukan untuk bentuk STL karena perhitungan didasarkan pada radius kurvatur.

Proses pembuatan lintasan pahat (*tool path*) dapat mempengaruhi kualitas permukaan, kekuatan, kekakuan, dan waktu pembuatan produk dalam proses *rapid prototyping*. Perencanaan lintasan termasuk perencaan lintasan bagian dalam dan lintasan bagian luar [11]. Lintasan bagian dalam merupakan proses pengisian material pada bagian dalam layer. Sedangkan untuk bagian luar hanya dilakukan oleh mesin LOM, karena lintasan luar dilakukan untuk memotong lembaran *raw material*. Kock B [7] menjelaskan mengenai analisa geometri proses RP dengan formasi dinding terluar menggunakan *offset object* model STL secara tidak seragam. Metodenya menghasilkan *boundary layer* yang *smooth* menggunakan *biarch fitting*. Untuk membuat kontur yang lebih efisien dan kompleks, Choi dan Kwok [12] mengembangkan algoritma secara hirarki pada kontur dengan *slicing* yang toleran. Hasilnya dapat digunakan untuk model komplek dan besar. Sedangkan Asiabanpur [13] mengembangkan proses pembuatan lintasan mesin untuk proses *rapid prototyping* baru berbasis serbuk, *Selective Inhibition Sintering* (SIS). Hasilnya berupa algoritma *slicing* dan *hatching* hanya untuk proses tersebut. Karena proses SIS berbeda dengan RP berbasis laser, algoritma yang dikembangkan belum bisa digunakan untuk laser trajectory.

2.5 Model *Facet* 3D

Sistim *Rapid Prototyping* menerima input sebuah objek produk 3D berupa sebuah file dengan format STL, sehingga model CAD perlu disimpan dalam bentuk file STL. Hasil dari proses *exporting* adalah sebuah pendekatan diskritisasi dari objek tersebut. Pendekatan diskritisasi akan merepresentasikan secara tepat sebuah model produk ke dalam kumpulan segitiga-segitiga. Proses *exporting* ini merupakan fitur standar yang dimiliki oleh sistem CAD.

File STL merupakan kependekan dari *stereolithography*. File yang berekstensi STL terdiri dari dua jenis format. Format yang pertama adalah *binary*. Pada format *binary*, model surface yang tersusun atas segitiga-segitiga disimpan dalam bentuk biner yang tidak terbaca dalam *text editor*. Format lainnya adalah ASCII (*American Standard Code for Information Interchange*). Format ini adalah yang paling umum digunakan karena lebih mudah dibaca dan dimengerti, serta dapat dibuka di *text editor*. Format ASCII dapat dilihat pada gambar berikut.



Gambar II.7 : STL ASCII Format

File STL ini menyimpan informasi objek produk dalam bentuk model *facet* 3D. Model *facet* sendiri adalah suatu model atau bentuk permukaan luar bidang yang tersusun dari satu atau lebih segitiga. Asal mula segitiga-segitiga ini adalah titik-titik (*vertex*) yang menyusun model, dan dihubungkan dengan garis-garis yang akan menjadi sisi (*edge*) segitiga, sehingga terbentuklah segitiga-segitiga yang saling berhubungan dan membentuk sebuah permukaan bidang yang lebih dikenal sebagai model *facet*. Ada dua fungsi utama dari pembentukan segitiga ini. Yang pertama adalah sebagai penghubung antar vertex untuk membuat sebuah permukaan, dalam hal ini yang menjadi permukaan adalah bidang segitiga (*face*). Fungsi yang kedua adalah untuk menentukan vektor normal bidang pada wilayah tertentu. Vektor normal tersebut merupakan vektor normal segitiga, yang didapat melalui *cross product* antara 2 vektor pembentuk sisi segitiga. Arah dari vektor normal bergantung pada arah putaran vektor dari ketiga vertex yang digunakan. Arah vektor normal terhadap putaran vektor pembentuk segitiga dapat ditentukan mengikuti kaidah tangan kanan. Jika putaran searah dengan jarum jam (*clockwise*), maka vektor normal akan menuju bidang. Sebaliknya, jika putaran berlawanan arah jarum jam (*counter clockwise*), maka vektor normal keluar dari bidang.

```
solid
  facet normal +0.0000000E+00 +9.8480775E-01 +1.7364817E-01
    outer loop
      vertex   +1.5289809E+01 +1.6289809E+01 +0.0000000E+00
      vertex   -1.5289809E+01 +1.6289809E+01 +0.0000000E+00
      vertex   -1.0000000E+01 +1.1000000E+01 +3.0000000E+01
    endloop
  endfacet
  facet normal +0.0000000E+00 +9.8480775E-01 +1.7364817E-01
    outer loop
      vertex   -1.0000000E+01 +1.1000000E+01 +3.0000000E+01
      vertex   +1.0000000E+01 +1.1000000E+01 +3.0000000E+01
      vertex   +1.5289809E+01 +1.6289809E+01 +0.0000000E+00
    endloop
  endfacet

  facet normal +0.0000000E+00 +0.0000000E+00 -1.0000000E+00
    outer loop
      vertex   -1.5289809E+01 +1.6289809E+01 +0.0000000E+00
      vertex   +1.5289809E+01 +1.6289809E+01 +0.0000000E+00
      vertex   +1.5289809E+01 -1.4289809E+01 +0.0000000E+00
    endloop
  endfacet
endsolid
```

Gambar II.8 : Contoh isi file STL

File yang berformat *.stl* merepresentasikan sebuah model *facet* dengan menyimpan informasi sesuai dengan standar tertentu. Informasi yang disimpan dalam file tersebut adalah segitiga-segitiga yang memiliki beberapa properti, berupa posisi ketiga buah vertex dalam bidang 3D, dan vektor normal dari segitiga yang bersangkutan.

Pada gambar II.8, dapat dilihat bahwa file berformat *.stl* menyimpan objek produk 3D dalam bentuk segitiga yang tersusun tiga buah vertex yang berada pada lokasi tertentu dalam sistem koordinat 3D. Berikut adalah penjelasan mengenai isi dari file stl di atas :

1. Kata *solid* menandakan dimulainya penggambaran atau penyimpanan model *facet* hingga ditutup dengan kata *endsolid.*
2. Kata *facet normal* menandakan bahwa akan dibangun sebuah permukaan yang berbentuk segitiga dengan nilai vektor normal berada pada kata setelah kata *facet normal*, hingga bertemu dengan kata *endfacet* yang berarti sebuah permukaan segitiga telah terbentuk, beserta informasi urutan dan letak vertex, serta vektor normal dari segitiga tersebut.
3. Kata *outerloop* menandakan dimulainya *loop* dari koordinat vertex-vertex yang membangun segitiga hingga bertemu dengan kata *endloop*.
4. Kata *vertex* merupakan vertex penyusun sebuah segitiga yang sebelumnya telah didefinisikan dengan *outerloop*. Informasi yang berada setelah kata *vertex* adalah posisi vertex pada sistem koordinat 3D.

Dalam sistem RP yang sedang dikembangkan, informasi yang diberikan oleh file STL disimpan dalam dua buah vektor,yaitu vektor segitiga dan vektor vertex. Setiap objek segitiga menyimpan informasi berupa nilai vektor normal segitiga tersebut, dan index-index vertex penyusunnya. Dan setiap objek vertex menyimpan posisi vertex tersebut, serta vektor normal vertex tersebut (jika ada). Penggunaan 2 buah vektor yang menyimpan objek segitiga dan vertex, dilakukan untuk menghindari redundansi, mengingat sebuah vertex dapat dimiliki lebih dari

satu segitiga. Dengan digunakannya dua buah vektor yang berbeda untuk menyimpan objek segitiga dan vertex, maka beberapa segitiga bisa memiliki vertex yang sama. Berikut adalah tabel yang menggambarkan struktur data dalam penyimpanan objek vertex dan segitiga,

Tabel II.2 : Tabel struktur index segitiga

Index segitiga	Index vertex 1	Index vertex 2	Index vertex 3
1	65	46	87
2	776	456	543
3	345	523	99
4	2343	765	5678
5	8293	7921	87
…	…	…	…

Tabel II.3 : Tabel struktur index vertex

Index vertex	Koordinat x	Koordinat y	Koordinat z
1	+1.63413E-01	+9.90319E+01	-1.22082E+00
2	+0.00000E+00	+9.99999E+01	-1.00000E+00
3	+7.11646E-02	+9.84506E+01	-1.29780E+00
4	+9.98365E+01	+9.68020E-01	-1.85569E+01
5	+9.99999E+01	+9.01001E-08	-1.87777E+01
…	…	…	…

2.6 Metode Penentuan CC-Point

Gandjar K. [8] menjelaskan mengenai algoritma cepat dalam penentuan cc-point. Hal yang pertama yang harus dilakukan pada pembuatan *laser trajectory* adalah mendapat nilai *cc-point*. Secara sederhana dapat dikatakan bahwa untuk mendapatkan cc-point dilakukan dengan mengiriskan bidang $z = c$ dengan model *facet* dari prismatic atau berkontur produk. Irisan ini akan menghasilkan titik-titik perpotongan antara bidang-z dengan model. Karena modelnya berbasis *facet* segitiga, maka perpotongan terjadi antara bidang-z dengan sisi-sisi segitiga.

Bidang-bidang yang digunakan untuk mencari perpotongan ini tentunya bukan satu bidang saja melainkan bidang-bidang $z = c_i$ untuk $i = 1, 2, \ldots, k$, dimana nilai c_i berkisar dari koordinat z paling rendah sampai koordinat z paling tinggi, atau $z_{min} \leq c_i \leq z_{max}$. Untuk mendapatkan nilai-nilai c_i cukup diberikan berapa interval nilai antara c_i dan c_{i+1} yang diinginkan. Jadi, masukan (*input*) pada proses menentukan cc-point ini adalah interval antar bidang potong.

Ada dua algoritma yang bisa digunakan untuk mendapatkan cc-point.

1. Metode *brute searching*,

   metode ini dilakukan dengan mengecek setiap segitiga yang ada apakah berpotongan dengan bidang-z atau tidak. Jika berpotongan, indeks segitiga tersebut disimpan dalam sebuah struktur data Vektor. Pengecekan ini dilakukan untuk setiap bidang-z yang akan diiriskan dengan model *facet*.

2. Metode *adjacent serching*,

   metode ini diawali dengan mencari secara acak sebuah segitiga yang berpotongan dengan bidang-z (sebut saja segitiga u), begitu didapatkan satu segitiga tersebut, maka algoritma berjalan sebagai berikut:

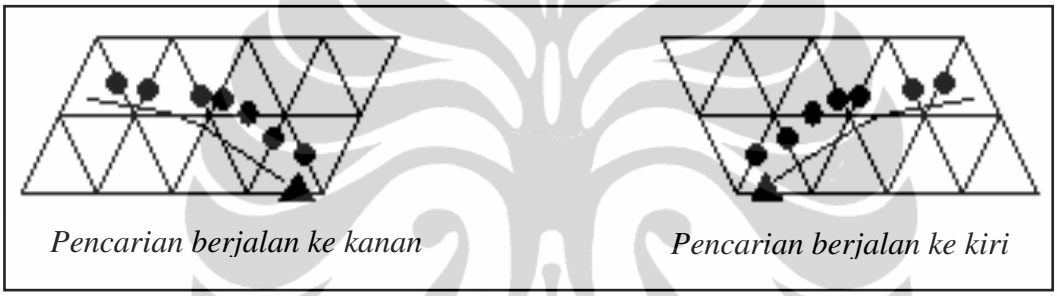
   1. ambil satu sisi segitiga, missal sisi s yang berpotongan dengan bidang-z, kemudian tentukan titik perpotongannya. Simpan titik ini pada sebuah struktur data vector.
   2. cari segitiga yang memiliki sisi s sebagai salah satu dari tiga sisi-sisinya selain segitiga yang sudah disebutkan pada no.1 di atas.
   3. lakukan kembali langkah no.1 di atas.

   Algoritma akan berhenti jika menemui salah satu dari kondisi-kondisi berikut:

   1. Segitiga terakhir yang ditemukan memiliki indeks yang sama dengan segitiga yang pertama kali ditemukan secara acak, yaitu segitiga u. Kondisi ini dapat terjadi manakala titik-titik perpotongan membentuk kurva tertutup.

2. Sisi s hanya dimiliki oleh satu segitiga saja, artinya tidak ada segitiga lain yang bertetangga dengan segitiga tersebut. Ini dapat terjadi pada titik-titik perpotongan yang membentuk kurva terbuka dan sisi s adalah sisi tepi dari model *facet*.
3. Tidak ada lagi yang berpotongan dengan bidang-z.

Sama halnya dengan metode brute searching, metode ini dilakukan untuk setiap kali pengirisan bidang-z dengan model *facet*. Perhatikan gambar berikut sebagaimana ilustrasinya.



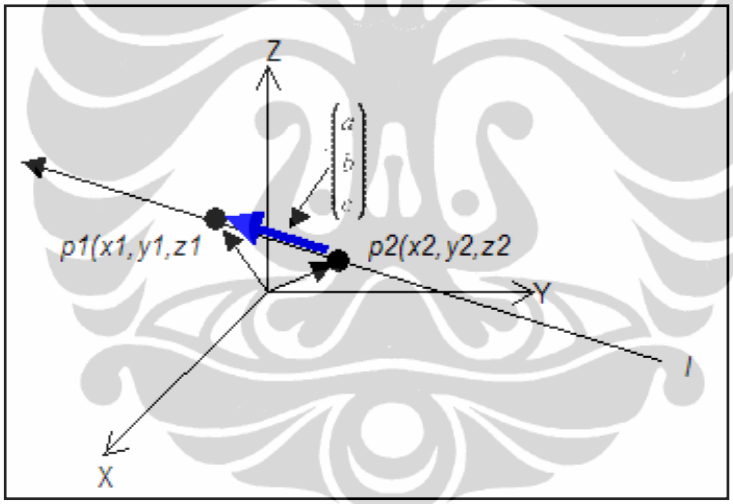
Gambar II.9 : Arah pencarian perpotongan antara bidang potong dengan model facet [8]

Ada satu kelebihan utama metode *adjacent searching* dibandingkan metode *brute searching* dalam hal keterurutan. Jika menggunakan *brute*, setelah pencarian titik-titik potong selesai dilakukan, sulit untuk melakukan tracking (perjalanan) dari titik satu ke titik lainnya. Sebab, pencarian segitiga dimulai dari indeks terkecil hingga indeks terakhir dimana belum bias dipastikan terurut. Namun, berbeda jika metode *adjacent searching* yang digunakan. Sejak awal pencarian hingga akhir, metode ini justru mempertahankan keurutan ini. Manfaat dalam implementasi riil proses *rapid prototyping*, *laser trajectory* akan bekerja jauh lebih efisien jika titik-titik yang akan dipotong berada dalam posisi terurut berdasarkan kedekatanya. Oleh karena itu, dalam tesis ini akan digunakan metode *adjacent searching* dalam menentukan cc-point.

2.7 Metode Untuk Menentukan Perpotongan Sisi Segitiga Dengan Bidang-Z

Gandjar K. [8] menjelaskan mengenai algoritma untuk menentukan perpotongan sisi segitiga dengan bidang z. Algoritma ini ini maksudnya menetukan formulasi untuk mencari titik potong bidang $z = c$ dengan sisi dari sebuah segitiga. Sisi segitiga dibentuk oleh dua buah titik $p_1(x_1,y_1,z_1)$ dan $p_2(x_2,y_2,z_2)$. Langkah pertama adalah mengecek apakah c berada di dalam rentang z_1 dan z_2. Jika iya, maka berarti sisi segitiga tersebut berpotongan dengan bidang $z = c$. Langkah selanjutnya adalah mencari titik dimana perpotongan terjadi.

Secara matematis, persamaan garis dalam ruang R3 dirumuskan sebagai berikut.



Gambar II.10 Garis pada ruang R3[8]

Berdasarkan gambar di atas, persamaan garis-l yang dibentuk oleh 2 titik adalah [8],

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix} t + \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} \qquad \text{(II.1)}$$

$$\text{Karena } p1 = p_2 + \begin{pmatrix} a \\ b \\ c \end{pmatrix}, \text{ maka } \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} - \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} \qquad \text{(II.2)}$$

Dengan demikian persamaan (II.1) dapat disubstitusi menjadi

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \left(\begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} - \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} \right) t + \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} \qquad \text{(II.3)}$$

Karena $z = c$, maka nilai t dapat dihitung, yaitu $t = \dfrac{z - z_2}{z_1 - z_2}$

Maka dari persamaan (II.3) dapat diperoleh titik (x,y,z) sebagai titik perpotongan antara sisi segitiga yang dibentuk oleh titik $p_1(x_1,y_1,z_1)$ dan $p_2(x_2,y_2,z_2)$ dengan bidang z = c.

Metode ini hanya menunjukkan bahwa jika bidang potong terletak di sisi-sisi segitiga, sehingga ada kemungkinan jika bidang potong bertemu di titik vertex segitiga, dipastikan algoritma tidak akan berjalan. Sehingga metode ini perlu ditambahkan untuk kondisi kemungkinan jatuhnya titik potong pada titik vertex.