Universidade Federal da Bahia

Escola Politécnica / Instituto de Matemática

Programa de Pós-Graduação em Mecatrônica

SILVIO ROBERTO MONTENEGRO MARTINS

UM SISTEMA CAM PARA PROTOTIPAGEM RÁPIDA POR ADIÇÃO DE CAMADAS

DISSERTAÇÃO DE MESTRADO

Salvador

2011

Universidade Federal da Bahia

Escola Politécnica / Instituto de Matemática

Programa de Pós-Graduação em Mecatrônica

SILVIO ROBERTO MONTENEGRO MARTINS

UM SISTEMA CAM PARA PROTOTIPAGEM RÁPIDA POR ADIÇÃO DE CAMADAS

Trabalho apresentado ao Programa de Pós-Graduação em Mecatrônica da Escola Politécnica e do Instituto de Matemática da Universidade Federal da Bahia como requisito parcial para obtenção do grau de Mestre.

Orientador: Prof. Dr. Herman Lepikson

Salvador

2011

RESUMO

Com a evolução das técnicas de manufatura, novos produtos são lançados no mercado com uma velocidade cada vez maior. Uma das formas de viabilizar a redução do tempo gasto no ciclo de desenvolvimento de produtos é através da junção de etapas do processo, que vão desde o projeto aos testes e fabricação. A prototipagem rápida está inserida neste contexto, pois permite a construção de objetos físicos a partir de modelos CAD (*Computer-aided design*). Essa técnica de construção assistida por computador oferece uma alternativa rápida e, no todo, mais barata e efetiva para produção de protótipos funcionais, se comparada aos processos convencionais de criação, demorados e custosos. Entretanto, a maioria das soluções disponíveis no mercado são proprietárias, fato que resulta em tecnologias fechadas e em custos ainda elevados. Este trabalho tem como foco o desenvolvimento de um sistema CAM (*Computer-aided manufacturing*) capaz de atuar em um sistema de prototipagem rápida de baixo custo, visando tornar esta tecnologia acessível a pequenas e médias empresas (PMEs). O sistema CAM desenvolvido será responsável pelo tratamento dos arquivos tridimensionais obtidos a partir do CAD, apresentando soluções para o fatiamento e planejamento das rotas de deposição de material na máquina de prototipagem rápida, bem como por comandar os eixos da máquina por CNC (Controle Numérico Computadorizado).

ABSTRACT

The evolution of the manufacture techniques forces the companies to launch their products in the market with an increasing velocity. One way to reduce time in the product development cycle is merging steps of the process such as design, tests and manufacturing. Rapid prototyping can help in this context, allowing fabrication of physical objects from CAD (Computer-aided design) models. This technique of construction provides a fast alternative and, on the whole, cheaper and more effective for producing functional prototypes when compared with traditional models of fabrication. However, the majority market solutions are proprietary. This fact results in high costs and not accessible technologies. The main goal of this work is to develop a new CAM (Computer-aided manufacturing) system that can be used in low cost rapid prototyping machines, making this technology accessible to small and medium-sized enterprises. The developed system will be able to handle tridimensional files from CAD model, providing solutions for slicing procedures and path planning of material in rapid prototyping machine and controlling axis of the machine with CNC (Computer Numerical Control).

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 Introdução

A integração do mercado mundial aumentou a concorrência entre as empresas. Para manter a competitividade, é preciso investimentos constantes em pesquisa e tecnologia, visando ao desenvolvimento de novos produtos e processos. Com a prototipagem rápida, é possível a redução de tempo e custo no ciclo de produção.

O princípio utilizado na fabricação permite dividir a prototipagem rápida em duas classes. Na primeira classe, o objeto é produzido através da subtração de material, ou seja, o sólido é esculpido a partir de um bloco maciço. Nesta categoria, destacam-se as máquinas de controle numérico (CNC – Controle Numérico Computadorizado). A segunda classe (manufatura aditiva) trabalha com a fabricação através da adição de material. Neste caso, o sólido é construído camada por camada.

Apesar das tecnologias possuírem o mesmo objetivo final, o princípio utilizado na fabricação traz características diferentes para o sólido construído. Segue um comparativo entre estas tecnologias (WOHLERS e GRIMM, 2003).

1. **Material utilizado:** enquanto na manufatura aditiva há certa limitação neste aspecto, nas máquinas CNC é possível utilizar-se de uma grande diversidade de materiais, desde que sejam facilmente usináveis;
2. **Complexidade do sólido:** nas máquinas de controle numérico, quanto maior a complexidade, maior o tempo e o custo para produzir o protótipo. Na manufatura aditiva, qualquer sólido possível de ser modelado em um software CAD (*Computer-Aided Design*) pode ser produzido;
3. **Repetitividade:** a manufatura aditiva é fortemente influenciada por fatores como temperatura, umidade, orientação do sólido. Em máquinas CNC, estes fatores influenciam com menor intensidade, o que permite a esta tecnologia uma repetitividade melhor;
4. **Volume de trabalho:** máquinas CNC não possuem limitações neste aspecto e podem produzir peças de diversos tamanhos. As máquinas de manufatura aditiva estão atualmente limitadas a um volume máximo de cerca de 600 x 900 x 500 mm;
5. **Treinamento de pessoal:** a necessidade de treinamento para utilização de uma máquina de controle numérico é muito maior comparando-se a uma máquina de manufatura aditiva;

Estes aspectos analisados diferenciam as tecnologias no que se refere à fabricação. Porém, estas tecnologias estão inseridas em um processo mais amplo que envolve outras atividades.

A Figura 1 mostra as quatro etapas que compreendem o processo de prototipagem rápida (KAMRANI e NASR, 2006).

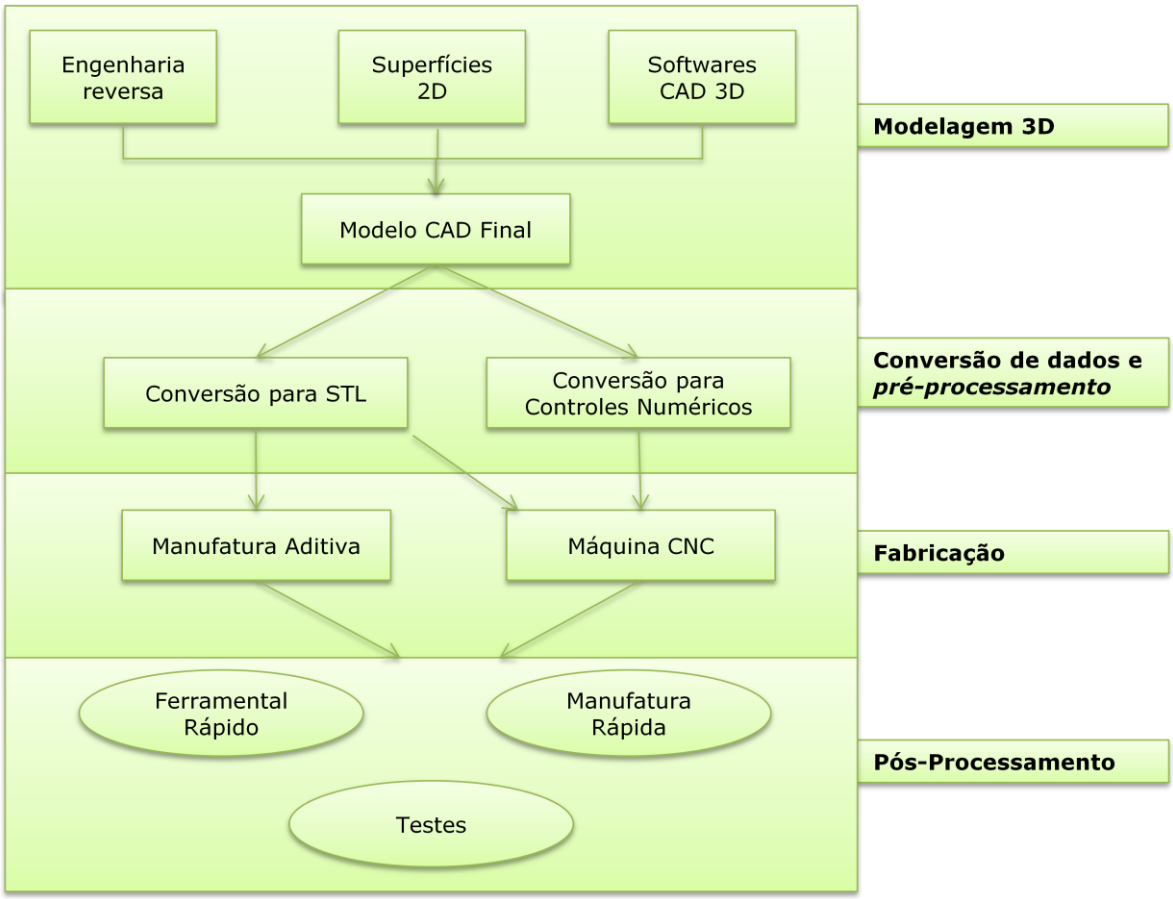


Figura 1 - Fluxo do processo de prototipagem rápida, tradução de (KAMRANI e NASR, 2006).

Modelagem 3D: compreende a fase de concepção do sólido a ser fabricado. O projeto é feito em um software CAD utilizando técnicas de modelagem convencionais ou através de outras possibilidades, como engenharia reversa em objetos físicos já existentes. Existem diversos softwares capazes de atuar nesta modelagem. Alguns exemplos são: SolidWorks, SolidEdge, Catia, Pro-Engineer e Google Sketchup. Ao final deste processo, será gerado um arquivo capaz de representar o sólido tridimensional no computador. Cada software possui uma extensão própria, o que significa que os dados estão organizados de acordo com o padrão definido pelo próprio software.

Conversão de dados e pré-processamento: esta etapa inicia-se com a necessidade de converter o modelo da etapa anterior em um padrão que seja compreendido nas próximas fases. Na prototipagem por adição de material, o formato de exportação mais utilizado é o STL. Este formato atualmente é considerado como padrão de arquivo tridimensional para a manufatura aditiva.

Após esta conversão é necessário realizar o planejamento dos caminhos de ferramenta. Esta atividade consiste em definir a melhor forma de gerar o sólido a partir das informações do arquivo. Na manufatura aditiva, há a conversão de informação tridimensional para bidimensional através do fatiamento do objeto. Sendo assim, o planejamento é feito para cada camada. Nas máquinas de controle numérico, há a possibilidade de planejar rotas em mais de duas dimensões, com máquinas capazes de realizar cortes em três ou mais dimensões.

Ao final do planejamento, os comandos devem ser repassados para as máquinas para que estas iniciem a fabricação do objeto. As máquinas de controle numérico possuem uma padronização que define uma série de comandos que possibilitam a comunicação entre o *software*, responsável pelo planejamento, e o *hardware*, responsável pela fabricação. Estes comandos são chamados de código G.

Fabricação: nesta etapa, os comandos gerados na fase de planejamento são repassados para a máquina. Esta é responsável por receber estes comandos e controlar seus sensores e atuadores, de forma a atender às solicitações. Considerando que o processo ocorreu sem a presença de falhas, o protótipo físico será construído e terá dimensões compatíveis com as especificações de projeto.

Pós-processamento: assim que o sólido final é obtido, iniciam-se as atividades que visam garantir que o protótipo atenda aos requisitos estabelecidos na fase de concepção. Nesta etapa, estão envolvidas diversas técnicas que podem ser aplicadas aos protótipos. A prototipagem rápida pode envolver reações químicas aplicadas ao protótipo para diversos fins, como aumentar a rigidez do protótipo ou garantir uma maior colagem entre as diversas camadas. Para as máquinas CNC, é comum realizar polimento, tratar termicamente ou retirar pequenos excedentes.

Técnicas de ferramental rápido podem ser utilizadas para auxiliar na fabricação de pré-séries dos protótipos. Neste caso, a máquina de prototipagem rápida seria utilizada para criar apenas os moldes do produto final.

1.1 Sobre este trabalho

O processo de prototipagem rápida é bastante complexo; cada um dos tópicos citados pode ser muito mais aprofundado. Este é um tema de estudo altamente relevante, pois esta tecnologia pode ser aplicada nas diversas engenharias, áreas de saúde e arquitetura. Esta visão geral do processo produtivo demonstra a necessidade da integração de componentes mecânicos, eletrônicos e computacionais para se conseguir resultados efetivos.

O presente trabalho está focado na manufatura aditiva. Nesta tecnologia, existem diversas técnicas de prototipagem. Entre os processos mais populares, pode-se citar: estereolitografia (SLA - *Stereolithography Apparatus*), modelagem por fusão e deposição (FDM - *Fused Deposition Modeling*), sinterização seletiva a laser (SLS - *Selective Laser Sintering*), manufatura de objetos em lâminas (LOM - *Laminated Object Manufacturing*) e impressão tridimensional (3DP - *Three Dimensional Printing*).

Apesar das diferenças nos princípios da tecnologia de manufatura aditiva, existem atividades que são comuns a todas estas tecnologias. No que tange ao aspecto computacional, quatro tarefas são comuns ao processo de planejamento para manufatura em camadas: orientação, fatiamento, geração de suporte e planejamento da rota (KULKARNI et al., 2000).

A maioria das soluções para manufatura aditiva disponíveis no mercado são proprietárias, fato que resulta em tecnologias fechadas e em custos elevados. O presente trabalho pretende contribuir para o tema com um sistema CAM (*Computer-Aided Manufacturing*) capaz de realizar as principais etapas do processo de planejamento e execução dos protótipos rápidos.

Esta dissertação é resultante de um grupo de pesquisa da UFBA voltado para a prototipagem rápida por adição de material, tendo por escopo os aspectos computacionais do problema. A validação deste trabalho será feita em um protótipo de uma máquina de baixo custo construída por este grupo de pesquisa.

1.2 Objetivo

O foco deste trabalho é o desenvolvimento de um sistema CAM capaz de atuar em uma máquina de manufatura aditiva de baixo custo. Tal sistema será responsável pelo tratamento dos arquivos tridimensionais, apresentando soluções para o fatiamento e planejamento da rota de deposição da máquina de prototipagem.

Os objetivos secundários traçados para este trabalho são:

- Auxiliar o produto mecatrônico a atender requisitos de protótipos, como tolerância dimensional adequada, baixa rugosidade superficial e resistência mecânica;
- Disponibilizar um método simplificado de operação do software;
- Prover uma interface amigável para fins de parametrização e configuração;
- Validar o sistema desenvolvido num protótipo funcional, tendo como base o método FDM;
- Permitir a integração deste sistema com outras soluções disponíveis;

1.3 Justificativa

A materialização de sólidos a partir de arquivos no computador atrai a atenção de variados setores da economia. Entretanto, esta tecnologia está restrita apenas a alguns segmentos por envolver um alto custo para aquisição de equipamentos, manutenção e produção dos protótipos. A principal motivação deste trabalho está no desenvolvimento de um sistema de baixo custo que permita a difusão da tecnologia.

1.4 Estrutura desta dissertação

Este trabalho está estruturado em seis capítulos, sendo este o primeiro. No segundo capítulo, será apresentado o estado da arte das tecnologias envolvidas, visando a contextualizar o escopo do trabalho. O terceiro capítulo apresentará o modelo conceitual pesquisado, com detalhamento dos conceitos envolvidos e a respectiva justificativa para adoção de cada um, bem como a delimitação de escopo seguida para implementação do modelo. No quarto capítulo, será apresentado o resultado obtido após a conversão do modelo conceitual em um sistema efetivo para prototipagem rápida por adição de material. O quinto capítulo apresentará a integração entre o sistema proposto e o *hardware*. O sexto capítulo, por sua vez, trará os resultados dos testes e validações realizados, assim como as limitações deste trabalho que merecem atenção. No sétimo capítulo apresentar-se-ão as conclusões com as contribuições trazidas pelo trabalho e indicações para trabalhos futuros, visando a sua continuidade.

2 Manufatura Aditiva

A primeira máquina de manufatura aditiva comercial foi lançada pela 3D Systems em 1988 e foi chamada de "Stereolithography Apparatus (SLA)". Esta máquina é responsável pela criação de quarenta patentes nos Estados Unidos e outras vinte internacionais (CHUA, 2003).

É possível encontrar outras nomenclaturas associadas a este tema, como LM - l*ayered manufacturing* (manufatura em camadas) ou SFF – *solid freeform fabrication* (fabricação de sólidos de forma livre). Apesar de esta tecnologia existir há alguns anos, os principais conceitos relacionados à manufatura aditiva ainda estão em processo de formação (ASTM, 2009).

Este fato não retira a importância da tecnologia para a ciência e indústria, apenas demonstra que há um grande interesse de diversos grupos de pesquisa, que neste momento discutem a padronização dos conceitos gerados ao longo destes anos, a exemplo da ASTM, a qual divulgou recentemente norma visando definir um padrão de nomenclaturas e termos a serem empregados (ASTM, 2009). Contudo, mesmo esta norma ainda é um trabalho em andamento. Esta primeira versão muda, inclusive, o nome da tecnologia de *Rapid Prototyping* para *Additive Manufacturing*, porém, não há nenhuma garantia de que estas definições serão aceitas por toda a comunidade, nem que se afirmarão ao longo do tempo.

Sells (2009) busca diferenciar os conceitos de impressão 3D (*3D Printing*), prototipagem rápida (*Rapid Prototyping*) e fabricação de sólidos de forma livre (SFF), classificação útil para orientar as diferentes linhas de pesquisa:

Fabricação de sólidos de forma livre (SFF): série de técnicas para manufatura de objetos a partir da entrega de energia e/ou material em determinados pontos do espaço;

Prototipagem rápida: é um sistema que carrega as informações a partir de um arquivo STL e o converte em um modelo de fatias. Esta informação é utilizada para guiar o processo SFF para gerar as camadas fisicamente. Vale ressaltar que o termo "prototipagem rápida" vem sendo utilizado também pelos processos que geram protótipos por remoção de material;

Impressão 3D: pertence ao processo de prototipagem rápida que implementa as tecnologias de SFF mais simples. O objetivo é a impressão de modelos mais rapidamente, porém com uma menor qualidade. É deste conceito que surgiu o termo "Impressora 3D", que tem sido bastante divulgado na mídia.

Como visto, não há um consenso na literatura em relação às nomenclaturas utilizadas para definir conceitos relacionados à tecnologia. Como não é o foco deste trabalho, o termo manufatura aditiva será utilizado para referenciar a tecnologia de forma mais ampla,

relacionando-a ao processo de manufatura baseada em camadas e à capacidade de produzir objetos complexos a partir de arquivos CAD no computador.

Segundo Volpato (2007), inúmeras aplicações vêm surgindo para esta tecnologia; um dos principais motivos é o fato das pessoas assimilarem de forma mais rápida a informação do projeto quando esta é transmitida através de um modelo físico. Dos diversos aspectos em que a prototipagem pode atuar, alguns são apresentados a seguir:

- Aprendizagem: em todas as etapas do projeto são necessárias decisões da equipe. A representação física auxiliará na visualização de necessidades, requisitos e restrições, possibilitando uma decisão mais criteriosa e fundamentada;
- Comunicação: nos projetos multidisciplinares é necessária uma referência para acelerar o processo de troca de informações entre as equipes. Documentos ou desenhos bidimensionais podem ser bastante úteis para uma determinada equipe, porém podem não garantir o entendimento de equipes de outras áreas. Além disso, para pessoas leigas, é fundamental a avaliação física direta;
- Integração: existem etapas do projeto que têm natureza integrativa. Na montagem, por exemplo, que exige a interligação física de várias peças, a representação física pode ser bastante útil para antecipar problemas e soluções;
- Identificação de erros: problemas de projetos podem ser detectados e antecipados. Possíveis falhas verificadas prematuramente ajudam a diminuir substancialmente o custo de correção;
- *Redesign*: o redesenho de produtos exige a comparação direta entre as possíveis modificações. Pode-se realizar esta verificação diretamente em modelos concretos;
- Publicidade e pesquisas de mercado: as empresas, antes de investir na produção real de um produto, podem realizar uma pesquisa de mercado para avaliar a viabilidade do investimento ou até mesmo captar novos recursos através de contratos ou encomendas, veiculando na mídia um produto ainda em fase de construção;
- Estudos ergonômicos: em algumas situações, os projetistas não conseguirão prever condições inadequadas de uso do produto. A simulação de atividades reais é possível tanto para avaliação dos projetistas quanto para testes de usabilidade com possíveis usuários;

2.1 Principais tecnologias

As características físicas do protótipo final estão associadas diretamente à tecnologia de prototipagem rápida utilizada. A seguir, será apresentada uma descrição do processo de funcionamento das principais técnicas existentes atualmente.

Estereolitografia (SLA)

A patente desta tecnologia foi gerada por Hull (1986). Este processo é baseado em foto-polimerização. Uma plataforma do tipo elevador é submergida em um líquido foto-curável. Inicialmente, esta é deixada próxima da superfície, sendo baixada na medida em que as camadas são geradas. Para a realização da cura do líquido, um laser ultravioleta de baixa potência é direcionado para os pontos em que o líquido deve se solidificar. A Figura 2 ilustra o funcionamento desta técnica (SELLS, 2009).

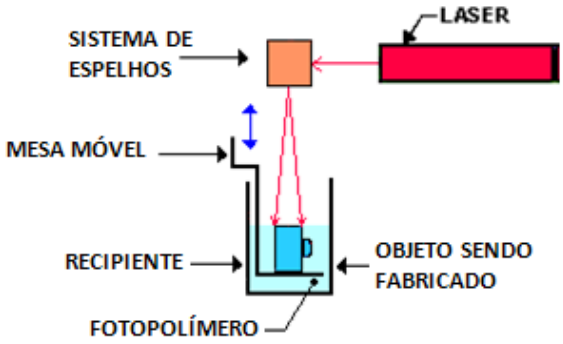


Figura 2 - Processo de prototipagem SLA (SELLS, 2009).

Depois que todo este processo é realizado, o sólido está curado por volta de noventa e cinco por cento. É necessária uma pós-cura para solidificar completamente o objeto e dar uma maior resistência mecânica ao mesmo. Este procedimento é feito utilizando luz ultravioleta (YAN e GU, 1996).

Como os sólidos são criados a partir de um meio líquido, regiões que não têm material de sustentação abaixo de si podem flutuar ou o peso das áreas onde há material pode gerar uma instabilidade na peça durante o processo de prototipagem. Para resolver este problema, são geradas colunas verticais, formando uma espécie de malha, desde a base da plataforma até a parte que necessita de suporte (YAN e GU, 1996; RAMASWANI, 1997).

A rota gerada para o laser é feita linha por linha. Existe também a necessidade de calcular as partes que necessitam de suporte (RAMASWANI, 1997).

Sinterização Seletiva a Laser (SLS)

Neste processo de prototipagem, a máquina contém duas plataformas: a primeira possui a matéria prima e a segunda é o local onde o sólido será prototipado. Para cada camada, a plataforma com matéria prima sobe e um rolo transporta o material até a plataforma em que o protótipo será produzido. Em seguida, um laser de CO2 é aplicado nos pontos em que se deseja solidificar. A solidificação ocorre por conta do calor gerado, que provoca a aglutinação das partículas da camada atual com a anterior. O restante da matéria prima que não foi aglutinada servirá como estrutura de suporte, como pode ser observado na Figura 3 (SELLS, 2009).

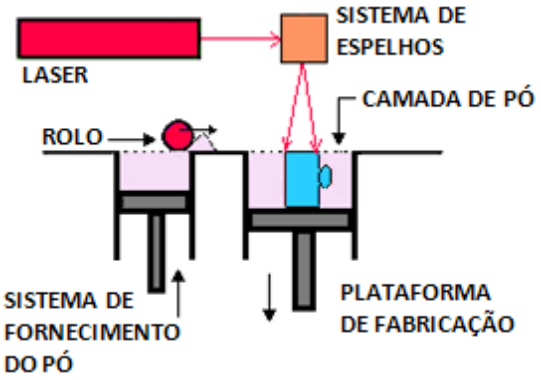


Figura 3- Processo de prototipagem SLS (SELLS, 2009).

A patente desta tecnologia foi gerada por Deckard (1986).

Manufatura de objetos em lâminas (LOM)

A matéria prima desta tecnologia é um rolo de papel que contém cola na parte inferior. Para cada camada, o papel é disposto sobre a plataforma. Em seguida, um laser corta o papel apenas na área correspondente ao contorno do objeto. Esta tarefa é realizada para todas as camadas até que o protótipo esteja completo. Ao final, é necessário remover o excesso de papel para obter o protótipo final. Para facilitar a remoção do material excedente, as partes que não fazem parte do sólido final são cortadas em pequenos cubos. A Figura 4 ilustra o funcionamento desta tecnologia (HOPKINSON et al., 2006).

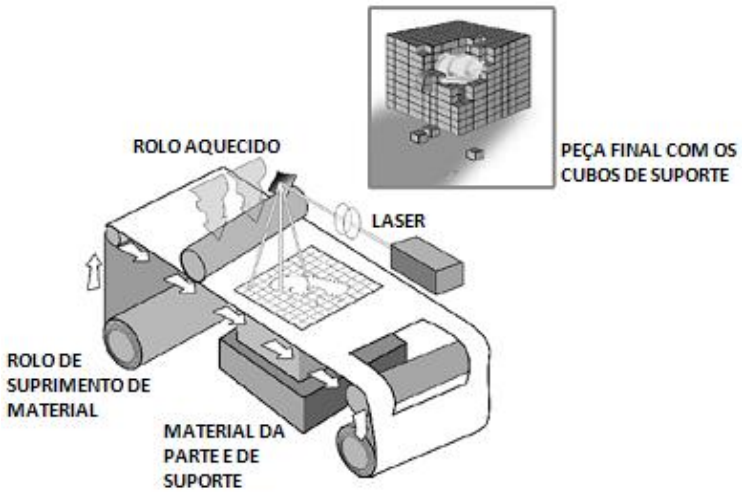


Figura 4 - Processo de prototipagem LOM (HOPKINSON et al., 2006).

O processo de planejamento envolve a geração dos contornos do objeto que deve ser seguido pelo laser. O papel serve também como estrutura de suporte (RAMASWANI, 1997).

A patente desta tecnologia foi gerada por Feygin e Ashland (1988).

Impressão 3D (3DP)

Esta tecnologia utiliza duas plataformas e pó como matéria prima. À medida que as camadas são produzidas, a primeira plataforma contendo o pó sobe e a segunda, em que o objeto está sendo produzido, desce. Um rolo é utilizado para transferir a matéria prima de um compartimento para o outro. A Figura 5 demonstra o processo de funcionamento (SELLS, 2009).

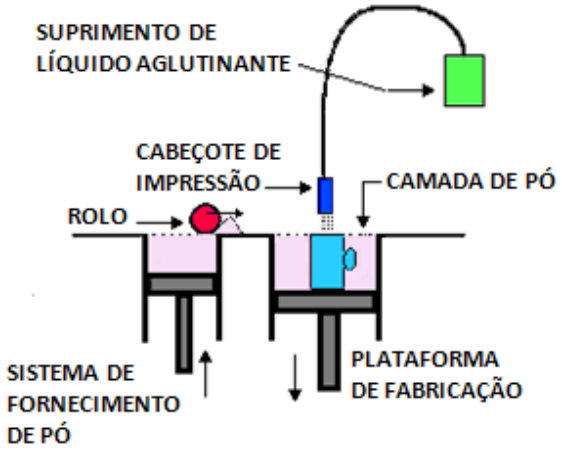


Figura 5 - Processo de prototipagem 3DP (SELLS, 2009).

Para solidificar o objeto, é utilizado um cabeçote de impressora jato de tinta contendo uma substância aglutinante. Este aglutinante é expelido em gotículas nos pontos que devem ser

solidificados. Não são necessárias estruturas de suporte, pois o pó das camadas anteriores pode ser utilizado para tal (LIRA, 2008).

A patente desta tecnologia foi gerada por Sachs et al. (1993).

Modelagem por fusão e deposição (FDM)

Nesta tecnologia a matéria prima está armazenada em uma bobina em forma de filamento. Para deposição do material (polímeros termoplásticos), existe um extrusor responsável por tracionar o material da bobina e depositá-lo através do seu bico. Como a temperatura do bico extrusor fica próxima do ponto de fusão da matéria prima, o material, depositado fundido, rapidamente se solidifica. Este processo é ilustrado na Figura 6 (SELLS, 2009).

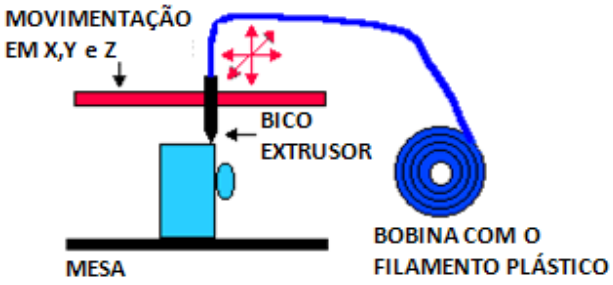


Figura 6 - Processo de prototipagem FDM (SELLS, 2009).

Devido ao curto tempo de solidificação, as partes mais simples não necessitam de suporte. Porém, podem existir pontos específicos em que, caso o suporte não seja utilizado, haja uma distorção no objeto final. Nestes casos, recomenda-se o uso de materiais auxiliares para suporte (RAMASWANI, 1997).

A patente desta tecnologia foi solicitada por Crum (1989).

Existem abordagens e patentes que utilizam métodos diferentes ou são variantes das tecnologias citadas anteriormente. Dentre os autores que aprofundam este tema, pode-se citar: Chua (2003), Gibson (2010), Hopkinson (2006).

2.2 CAD para a manufatura aditiva

A criação de um protótipo para a manufatura aditiva inicia-se sempre a partir de um software CAD (Computer-Aided Design – projeto auxiliado por computador). Estes softwares permitem aos projetistas modelar e definir atributos inerentes ao modelo idealizado. A qualidade do resultado obtido nesta fase é de extrema importância para o resultado final.

O modelo CAD precisa representar com coerência o sólido a ser produzido. Neste contexto, dois elementos ganham destaque: primeiramente, a questão estética, pois é necessário que modelo computacional esteja de acordo com os aspectos visuais pretendidos para o produto

planejado; o segundo item corresponde à precisão matemática, aspecto este que irá interferir diretamente na fidelidade dimensional das amostras produzidas (MCDONALD et al., 2001).
Há algumas técnicas que permitem a criação de modelos sólidos em um sistema CAD. McDonald et al. (2001) define como principais métodos de criação a modelagem por superfície (*Surface Modeling)* e modelagem por sólido (*Solid Modeling)*:

Modelagem por superfície

Neste caso, a superfície pode ser criada com a definição de uma série de curvas ou através da combinação de seções transversais. Com isso, tem-se uma superfície que contém uma série de pontos interligados. Movendo-se os pontos, individualmente ou em grupo, provocar-se-á uma mudança de forma no sólido. Através deste método é possível criar qualquer forma imaginável.

Modelagem por sólido

Nesta técnica, o sólido pode ser criado a partir da replicação de uma superfície bidimensional (que pode ser movida ou rotacionada enquanto é replicada), ou através do uso de objetos 3D primitivos (cubos, esferas, cilindros, etc). Para criar formas mais complexas, os sólidos podem ser combinados através de operações booleanas de subtração, adição e interseção.
Um conjunto de linhas ou arcos pode ser definido para ter o mesmo comprimento e fórmulas matemáticas podem ser utilizadas para definir as relações entre estes elementos. Desta forma, alterando-se um determinado parâmetro, é possível simular o impacto causado na forma final do objeto ou até mesmo validar se esta alteração seria fisicamente viável (MCDONALD et al., 2001).

Exportação do sólido produzido

Um ponto importante a ser observado é que atualmente existe uma infinidade de softwares CAD disponíveis. Os fornecedores destes sistemas buscam disponibilizar um ambiente integrado de modelagem em que é possível ter equipes de projetistas trabalhando sobre um mesmo produto. Cada um destes sistemas possui uma forma proprietária de armazenar suas informações em arquivo. Analisando tal situação, seria muito custoso para um sistema de manufatura aditiva conseguir interpretar cada um destes formatos de arquivo diferentes (GIBSON et al., 2010).
A solução para o problema foi criar um formato comum capaz de expressar as informações do sólido tridimensional. O STL foi feito com este objetivo e atualmente é um padrão de domínio

público que vem sendo adotado pelos fabricantes de máquinas de manufatura aditiva (GIBSON et al., 2010).

2.3 O formato STL

Como dito, a entrada de um sistema de prototipagem rápida é um arquivo tridimensional contendo as informações necessárias para gerar o modelo físico a partir das informações que constam no arquivo.

O formato mais utilizado atualmente é o STL. Lira (2008) realizou um levantamento com várias máquinas de prototipagem comerciais e os formatos de entrada utilizados. Das vinte e quatro máquinas pesquisadas, apenas cinco não interpretavam este formato.

O STL pode ser representado em ASCII (*American Standard Code for Information Interchange*) ou em formato binário. O formato ASCII apresenta um tamanho maior, porém é mais fácil de ser interpretado por um técnico. A informação que o arquivo contém é uma lista de faces triangulares descritas por pontos X, Y e Z. Para cada uma dessas faces, há a indicação do vetor normal que informa a orientação da face (CHUA, 2003).

A Figura 7 mostra um arquivo STL no padrão ASCII. A primeira informação presente no arquivo é o nome do sólido, a qual deve suceder a *tag* "solid". No exemplo, abaixo o nome do sólido é Teste. Após a informação do nome, vem a lista de faces que unidas irão gerar um objeto tridimensional.

```
solid Teste
        facet normal 0.00000e+00 1.00000e+00 0.00000e+00
                outer loop
                        vertex 0.00000e+00 0.00000e+00 2.00000e+01
                        vertex 0.00000e+00 0.00000e+00 0.00000e+00
                        vertex 1.00000e+01 0.00000e+00 2.00000e+01
                endloop
        endfacet
```

Figura 7 - Arquivo STL descrito no formato ASCII

Neste exemplo, existe apenas uma face descrita. O valor do vetor normal para X, Y e Z é [0,1,0] e os três vértices da face a ser formada estão respectivamente em: [0,0,2] , [0,0,0] e [1,0,2]. A Figura 8 mostra a face que seria formada através da descrição do exemplo anterior.

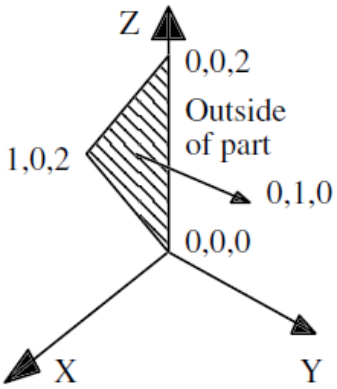


Figura 8 – Representação de uma face, adaptado de (CHUA , 2003)

O STL é um arquivo de exportação, ou seja, quando o usuário está modelando seu objeto em algum software CAD, este utiliza um padrão próprio para guardar as informações originais do desenho. Este fato faz com que, no momento da conversão para o formato STL, sejam geradas aproximações, pois os softwares CAD nem sempre são robustos o suficiente para gerar todas as faces corretamente (CHUA, 2003).

Uma desvantagem atribuída a este formato é que a redundância de vértices aumenta desnecessariamente o tamanho do arquivo. A Figura 9 mostra uma das possibilidades de se criar um quadrado através do STL. Neste caso, há duplicidade de informação em quatro vértices. É importante ressaltar que há um exagero no exemplo, pois para gerar este quadrado seriam necessários apenas dois triângulos, o que resultaria na redundância de apenas um vértice.

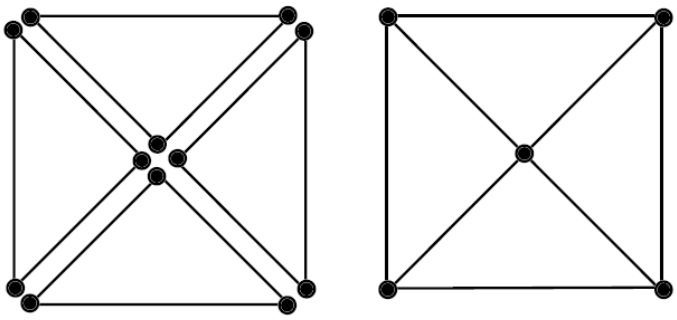


Figura 9 - Faces geradas para representar um quadrado. (CHUA, 2003)

Outra consideração importante é que no formato STL não há definição de escala. O software que vai interpretar as faces não saberá, por exemplo, se o valor “2” presente no arquivo está representando dois metros, dois centímetros ou dois milímetros. Conseqüentemente, apenas com as informações contidas no arquivo não há como saber as dimensões do sólido.

Apesar dessas desvantagens, algumas características importantes pesam a favor do STL. O fato de possuir um método bastante simplificado para representação de arquivos

tridimensionais gerou sua aceitação por praticamente todos os sistemas de prototipagem rápida. Além disso, o arquivo apresentará um tamanho bastante reduzido para representar algumas formas mais simples. (CHUA, 2003).

2.4 CAM para manufatura aditiva

Apesar de haver diferenças nos princípios físicos das tecnologias, as etapas do processo de planejamento para a prototipagem possuem muito em comum. De modo geral, o processo pode ser dividido em etapas. A Figura 10 mostra o fluxo das atividades, descrito a seguir (KOC, 2001):

- **Projetar em um software CAD o sólido a ser prototipado:** assim que o modelo estiver concluído, este deve ser exportado para um formato de arquivo específico (como comentado, atualmente o STL é o formato de arquivo tridimensional mais utilizado em máquinas de prototipagem);
- **Definir a orientação do sólido:** a definição da orientação pode afetar diversos aspectos, pois a posição de prototipagem irá influenciar, por exemplo, a quantidade de suporte necessária, o total de fatias, etc.;
- **Calcular a necessidade de suporte:** o suporte é necessário sempre que a ação da gravidade possa afetar alguma face do sólido, provocando a queda ou desestabilização da parte. No caso da Figura 10, um exemplo de partes que podem ser afetadas seriam o nariz e o queixo.
- **Fatiamento:** nesta etapa, são geradas todas as camadas do sólido. As fatias obtidas serão produzidas de forma sequencial na máquina de prototipagem.
- **Planejamento de rotas:** consiste na tradução das informações de cada camada no modelo real. O planejamento envolve definição de rotas para preenchimento de bordas, deposição de material e deposição de suporte.

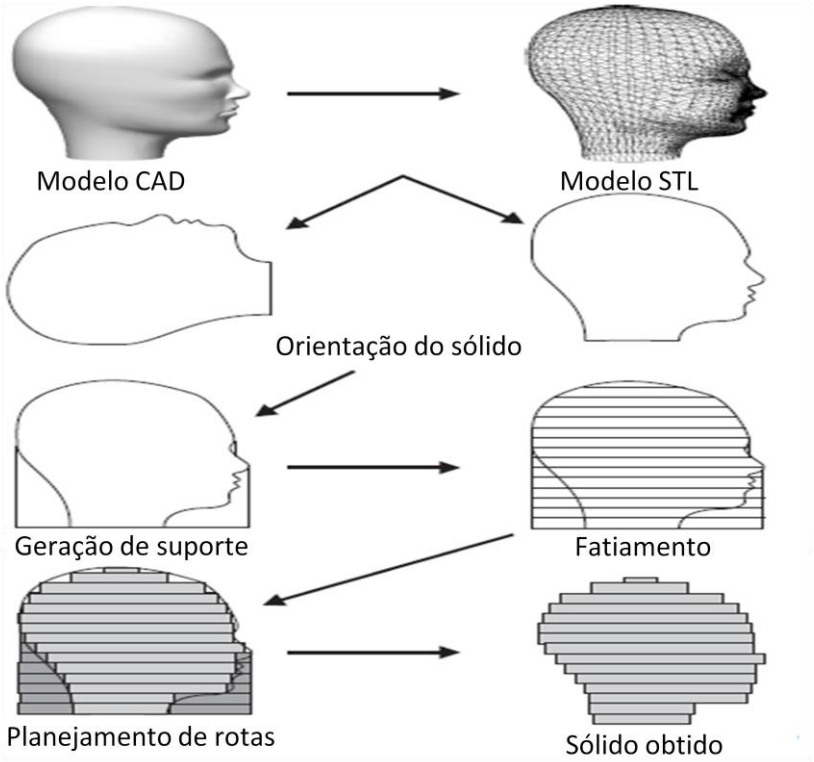


Figura 10 - Etapas do processo de prototipagem (KOC, 2001).

A partir dessas informações, pode-se definir quatro tarefas básicas como pertencentes ao processo de planejamento: fatiamento, orientação, suporte e planejamento de rota (KULKARNI et al., 2000).

Como pode ser observado na Figura 11, estas etapas podem ser subdivididas em dois domínios. O domínio do modelo envolve o processamento tridimensional e irá definir questões como orientação e suporte. O domínio da camada tratará da definição do planejamento da rota da ferramenta. A tarefa responsável por realizar a conversão das informações tridimensionais para o domínio das camadas (bidimensional) é o fatiamento.

O fato do processo de prototipagem ser realizado em camadas reduz a complexidade geométrica do objeto modelado. Este fator permite que os sistemas de prototipagem rápida funcionem com pouca intervenção humana (KULKARNI et al., 2000).

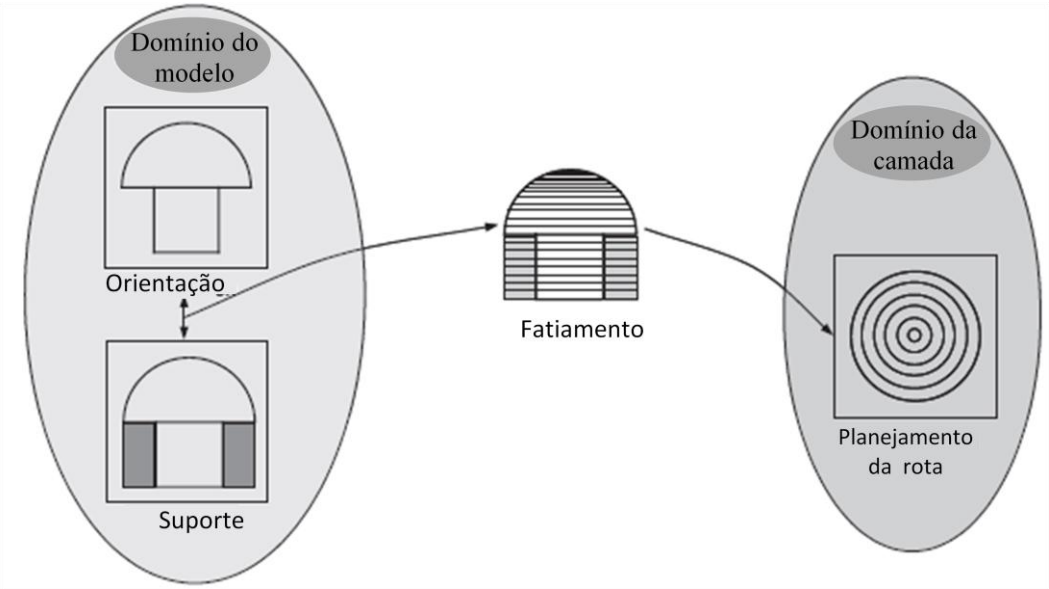


Figura 11 - Mapeamento dos domínios da informação (KULKARNI et al., 2000).

A cada atividade citada podem ser empregadas diversas estratégias, as quais serão detalhadas a seguir:

Orientação

A escolha da orientação vai interferir em diversos fatores que devem ser considerados, dentre eles estão:

- **Tempo de prototipagem:** é um fator importante, pois a altura está diretamente associada com o tempo total de prototipagem. Quanto maior for o número de camadas, maior o tempo para produzir o sólido. Dependendo da orientação escolhida, a altura do sólido também mudará;
- **Qualidade das superfícies:** em alguns processos, o contato com a plataforma de prototipagem reduz a qualidade da superfície. Nas técnicas que utilizam suporte, os pontos que estão em contato com esta estrutura apresentam um aumento na rugosidade superficial. Uma superfície de má qualidade aumentará substancialmente o custo de pós-processamento do sólido;
- **Base da peça:** a área da base afetará na estabilidade do sólido.

Como a orientação interfere em diversos fatores, a sua escolha deve ser feita considerando-se um esquema de análise de opções em que todos sejam examinados de forma concorrente. A Figura 12 demonstra cinco orientações diferentes para o objeto; cada escolha traz uma nova característica para a orientação: maior área da base, área mínima de contato com suporte, menor altura do objeto, menor rugosidade das faces, menor volume de suporte.

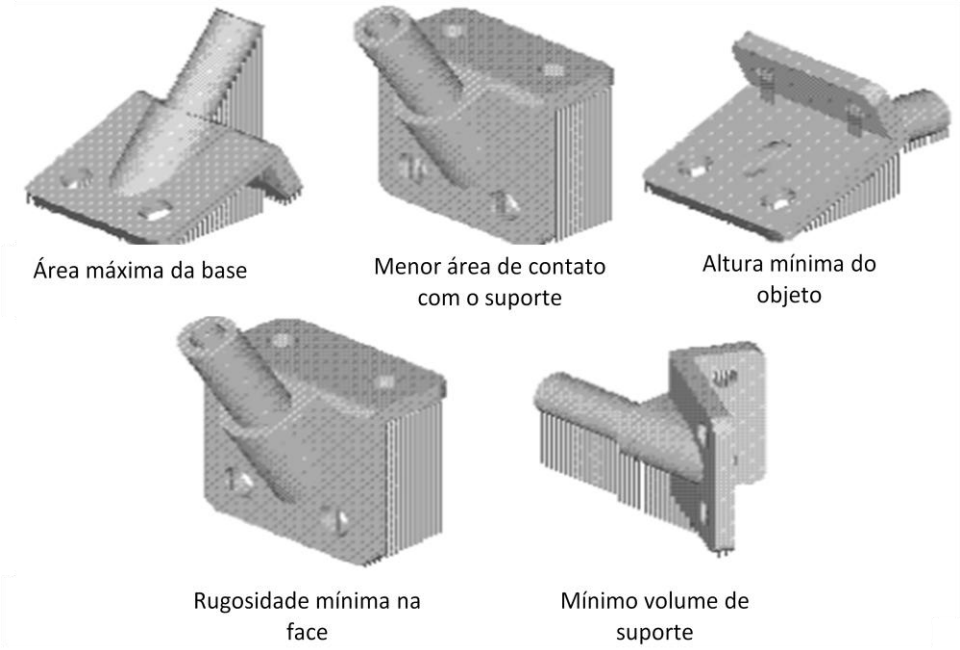


Figura 12 - Diferentes orientação possíveis. Tradução de (KULKARNI et al., 2000).

Suporte

A estrutura de suporte é necessária para manter a estabilidade do modelo tridimensional, apoiar saliências, segurar paredes inclinadas e sustentar faces que estão no topo. Sempre que a ação da gravidade possa afetar alguma face do sólido, provocando a queda ou desestabilização da mesma, é indicado que o suporte seja utilizado. A definição do suporte depende da orientação da parte (KOC, 2001).

Segundo GIBSON et al. (2010), as estruturas de suporte podem ser classificadas em duas formas gerais:

- **Material de suporte similar**: o mesmo material utilizado para criar o sólido é utilizado na estrutura do suporte. Neste caso, as partes do sólido e o suporte precisam ser planejados de modo a possibilitar uma separação posterior. Uma das formas de realizar tal procedimento é utilizando uma temperatura diferente durante a deposição do suporte, o que vai dificultar a adesão do material de suporte às partes da peça. Adicionalmente, podem ser utilizadas distâncias diferentes de separação entre a camada atual e a próxima quando houver uma mudança de material da parte para o suporte ou vice-versa. Esta ação irá gerar uma menor resistência mecânica nestes pontos, facilitando um posterior rompimento.
- **Material de suporte secundário:** para utilizar tal estratégia, o sistema de manufatura aditiva deve possuir um segundo sistema de extrusão. Desta forma, permite-se explorar a diferença de propriedades entre o material da peça e o do suporte. As

diferenças podem ser mecânicas (utilizando material mais fraco para o suporte) ou químicas (material de suporte que se dissolva ao entrar em contato com um solvente). As máquinas que utilizam este extrusor auxiliar podem realizar a deposição do material de suporte em paralelo à deposição do material do sólido.

Comparando-se as duas abordagens, ao utilizar o mesmo material como estrutura de suporte, há um menor custo de projeto da máquina. Em contrapartida, serão afetados a qualidade do protótipo e o tempo de prototipagem. Com a abordagem de dois tipos de materiais, há um custo maior de equipamento e de aquisição de matéria-prima, porém, ganha-se na qualidade e no tempo de prototipagem.

Fatiamento

O fatiamento é responsável pela conversão do modelo tridimensional para o domínio das camadas. As fatias são geradas a partir da interseção do modelo em CAD com um plano horizontal. Existem duas técnicas de fatiamento: o fatiamento uniforme e o fatiamento adaptativo (KOC, 2001).

No fatiamento uniforme, todas as fatias geradas seguem uma distância fixa, enquanto no fatiamento adaptativo, a distância varia dependendo da curvatura da superfície do modelo. Quanto maior o grau de afastamento das fatias, menor é a quantidade de fatias geradas, o grau de exatidão e o tempo de prototipagem. Por outro lado, se os planos gerados estão mais próximos uns dos outros, maior será a fidelidade ao modelo CAD; em contrapartida, o tempo de fabricação será maior (KULKARNI et al., 2000). A Figura 13 ilustra a diferença entre as duas técnicas de fatiamento.

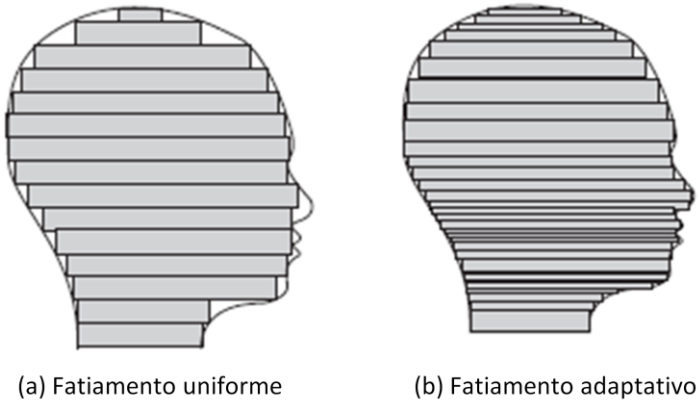


Figura 13 - Tipos de fatiamento, adaptado de (KOC, 2001).

Independente da estratégia utilizada (uniforme ou adaptativo), a maioria dos algoritmos trabalha com informações obtidas diretamente do arquivo STL (*Tesselated CAD*) (PANDEY, 2003). O processo de fatiamento consiste em gerar uma série de planos paralelos ao modelo.

A interseção destes planos com a superfície triangular será a informação do contorno de cada camada. Este fluxo é ilustrado em duas etapas, conforme a Figura 14 (HUANG, 2009).

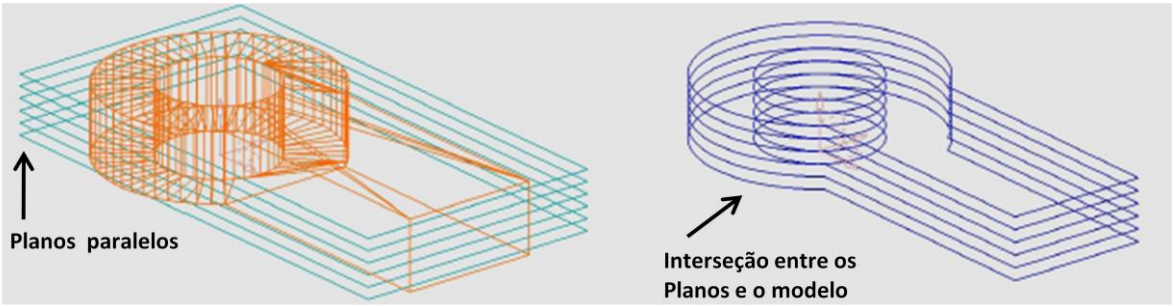


Figura 14 - Etapas do fatiamento (HUANG, 2009).

Planejamento da rota

O método de fatiamento demonstrado irá gerar uma série de pontos em uma ordem indefinida. O primeiro passo do processo de planejamento é identificar e definir a rota de deposição das bordas da fatia. Para realizar tal tarefa, ligam-se os pontos que pertencem à mesma face, voltando à informação triangular. Devem ser atribuídos números para identificar cada um destes triângulos (GIBSON, 2002).

O compartilhamento entre as arestas é outra informação que deve ser considerada. Arestas que são compartilhadas não entram na rota de deposição; elas são utilizadas para definir qual será a próxima face avaliada. (GIBSON, 2002).

A Figura 15 ilustra as etapas que compõem o processo descrito.

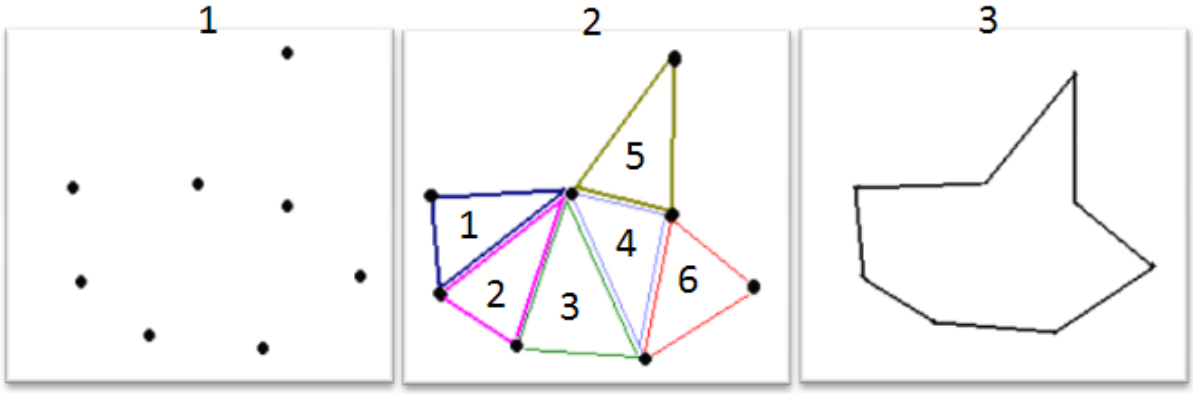


Figura 15 - Definição das bordas do sólido.

Depois que este procedimento é concluído, tem-se a rota definida para a borda da camada. Em seguida, devem ser definidas as rotas de deposição do interior da camada. Estas rotas podem ser geometricamente classificadas em espiral ou zigzag. No espiral, a deposição é gerada com uma série de contornos que são paralelos às bordas da fatia. No preenchimento em zigzag, a rota segue uma direção fixa e são traçadas retas paralelas interligadas, de forma a gerar um caminho contínuo em zigzag. A partir destas duas formas básicas de preenchimento, é

possível gerar outras estratégias mais elaboradas (RAMASWANI, 1997). A Figura 16 ilustra estes dois métodos de preenchimento.

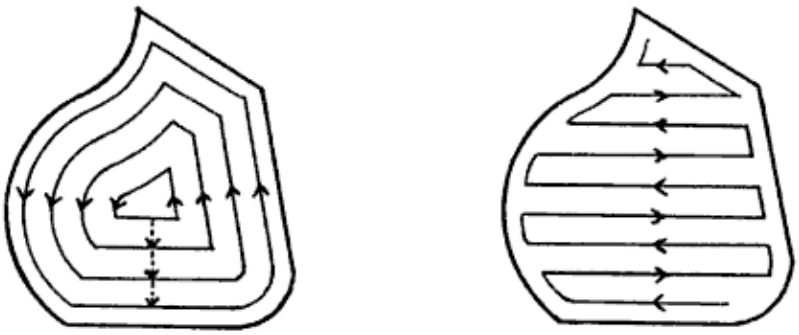


Figura 16 - Tipos de rotas de preenchimento (HELD, 1991).

Estas duas técnicas de planejamento são originárias de máquinas de controle numérico. A diferença básica entre esta técnica, quando aplicada à prototipagem rápida, é que as rotas são geradas para os locais onde deve haver deposição de material, enquanto nas máquinas de controle numérico, a rota é planejada nos locais onde deve haver a subtração do material.

As técnicas de planejamento utilizadas afetam diretamente as características do protótipo final. Por exemplo, a espessura da camada irá afetar o tempo de fabricação do protótipo e as propriedades mecânicas, como rigidez e resistência à tração. A orientação afetará o tempo de fabricação, as propriedades mecânicas e o volume de suporte necessário. As áreas que estiverem em contato direto com o suporte possuirão uma menor qualidade no acabamento superficial (KULKARNI et al., 2000). Portanto, há uma interdependência entre estas tarefas que fazem com que a qualidade final do protótipo obtido seja afetada por cada uma destas etapas. Estas atividades podem ser consideradas como elementares para o sistema de prototipagem rápida. Técnicas de planejamento mais eficazes podem ser o diferencial do processo.

2.5 Trabalhos correlatos

Nesta seção, serão analisados os principais trabalhos voltados para a prototipagem no Brasil e no mundo. Juntamente com este levantamento, serão analisados alguns *softwares* que possuem propostas similares à desta dissertação.

Dois projetos de destaque na área de prototipagem rápida são: RepRap (BOWYER, 2006) e FAB@Home (LIPSON e MALONE, 2010). Estes dois grupos buscam soluções inovadoras e de baixo custo que possibilitem a disseminação da tecnologia.

Com o crescimento destes projetos, surgiram subprojetos, tomando como base a proposta inicial, ou seja, reutilizam a infraestrutura disponibilizada e focam as pesquisas em aspectos específicos da prototipagem rápida. As equipes MakerGear (MAKERGEAR, 2010),

BitsFromBytes (BITSFROMBYTES, 2010) e MakerBot (PETTIS et al., 2009) trabalham focadas apenas no *hardware* da máquina de prototipagem e utilizam como base o software desenvolvido pela equipe RepRap. O objetivo destes grupos é apresentar novas soluções mecânicas para a prototipagem rápida. Estes três grupos disponibilizam kits de *hardware* na internet para usuários que tenham interesse em adquirir os produtos e realizar suas próprias pesquisas.

O Thingiverse (HOEKEN e PETTIS, 2010) disponibiliza um acervo de arquivos em que os projetistas CAD compartilham arquivos 3D gerados. Os usuários que tiverem interesse podem acessar e baixar livremente as criações disponíveis no site. Essa base de arquivos é importante para a comparação dos resultados obtidos por cada equipe. Os usuários podem incluir imagens com resultados obtidos e discutir com a comunidade, visando melhorias em seu processo.

No Brasil, há alguns grupos voltados para a prototipagem rápida e suas aplicações. A Tabela 1 lista os principais grupos. Estas informações foram colhidas do diretório dos grupos de pesquisa do CNPq (CNPq, 2010).

Tabela 1 - Grupos de pesquisa no Brasil envolvidos com a prototipagem rápida.

Nome do grupo	Instituição	Objetivos do grupo
CIMJECT	Universidade Federal de Santa Catarina – UFSC	Atuar nas áreas de projeto e de fabricação de moldes, auxiliadas por computador, para o processo de injeção Capacitação no uso de sistemas CAE/CAD/CAM voltados para a área de moldes de injeção para plásticos
Desenvolvimento Integrado de Produtos	SENAI - Departamento Regional da Bahia	Apoiar a indústria com pesquisa e desenvolvimento nas tecnologias de projeto de produto, simulação de processos, prototipagem rápida, engenharia reversa;
Laboratório Integrado de Design e Engenharia do Produto – LIDEP	Universidade Federal de Minas Gerais – UFMG	Auxiliar pequenas e médias empresas no projeto e engenharia de produtos; Buscar soluções em tecnologias de projeto e prototipagem por meio de sistemas CAE/CAD/CAM e metodologias voltadas para aspectos específicos de produto (como prototipagem rápida, montagem, processos de fabricação, etc);
Núcleo de Prototipagem e	Universidade Tecnológica Federal	Atuar nas linhas de Prototipagem Rápida, Ferramental Rápido, Sistemas CAD/CAM e usinagem CNC;

Ferramental – NUFER	do Paraná - UTFPR	Difundir estas tecnologias, no meio acadêmico e empresarial; Atuar no desenvolvimento de novos processos e metodologias;
Tecnologias Tridimensionais	Centro de Tecnologia da Informação Renato Archer – CTI	Fomentar o uso da prototipagem rápida nas pequenas e médias empresas; Cooperar na bioengenharia com a simulação de fenômenos e na modelagem de estruturas orgânicas; Apoiar pesquisadores nacionais, oferecendo suporte em prototipagem e manufatura rápida para aceleração de experimentos científicos; Apoio aos profissionais de saúde no planejamento cirúrgico virtual e físico com biomodelos;
Teorias e tecnologias contemporâneas aplicadas ao projeto	Universidade Estadual de Campinas – UNICAMP	Desenvolver teorias e métodos computacionais de apoio ao processo de projeto em arquitetura; Produção de maquetes e protótipos de edifícios e detalhes arquitetônicos por meio do uso de sistemas de prototipagem rápida;

Uma das etapas do processo de construção desta dissertação foi a interação com alguns destes grupos de pesquisa, visando o compartilhamento de informações e a troca de experiências. Em março de 2009, foi feita uma visita às instalações do grupo de pesquisa do SENAI – BA, em Salvador. Nesta visita, os responsáveis pelo grupo demonstraram os trabalhos desenvolvidos pela equipe. O foco deste centro é o fornecimento de serviços de prototipagem rápida para PMEs. O SENAI – BA trabalha apenas com a aplicação da prototipagem rápida e conta com máquinas que utilizam as tecnologias FDM e 3DP.

Em agosto de 2010, foi feita uma visita ao CTI, em Campinas – São Paulo. Este centro, por meio do Departamento de Tecnologias Tridimensionais (DT3D), atua em pesquisas no processamento de imagens médicas, engenharia reversa de estruturas anatômicas e aplicações de biomaterias para a prototipagem rápida. O DT3D possui máquinas com a tecnologia SLS, FDM e 3DP. Além disso, o CTI possui uma máquina Fab@Home e alguns componentes da equipe cooperam no desenvolvimento deste projeto (CTI, 2010).

Além dos grupos de pesquisa, foi feita uma visita à fabrica da ROBTEC, em Diadema – São Paulo. Esta empresa é uma importante aplicadora de tecnologias de prototipagem rápida no

Brasil; presta serviços a grandes e médias empresas usuárias de protótipos rápidos. Técnicos da empresa relataram como ocorre a interação com o cliente, desde a solicitação do serviço até a obtenção do protótipo e pós-processamento. Durante o período da visita, a fábrica estava operando normalmente e foi possível acompanhar máquinas com a tecnologia SLA, SLS e 3DP em operação (ROBTEC, 2010).

Resumindo: no Brasil, iniciou-se a disseminação do uso da prototipagem rápida entre as empresas, e prestadores de serviço especializados começam a se desenvolver. Entretanto, pesquisas na área só podem ser encontradas em poucos grupos, com destaque para o CIMJECT, NUFER, CTI e LIDEP. Observe-se, contudo, que apenas o NUFER e o CTI desenvolvem pesquisa e possuem trabalhos similares ao foco desta dissertação. Dentre as publicações de destaque temos Volpato et al. (2007a), Volpato et al (2007b) e Lixandrão (2009).

Com base nas investigações feitas acerca de pesquisas no Brasil e no mundo, foram levantadas as principais contribuições que possuem foco similar ao desta dissertação. A seguir, serão detalhados os projetos e suas contribuições em relação ao processo de interpretação de arquivos tridimensionais e planejamento de rotas. Os softwares de código aberto serão analisados com um maior nível de detalhes, pois, além de haver maior facilidade no acesso à documentação, serão mais úteis aos objetivos deste trabalho.

REPRAP

A ideia que originou o projeto RepRap surgiu em 2004, quando Adrian Bowyer (idealizador do projeto) publicou um artigo na internet com os resultados de suas pesquisas - um extrusor de liga de metal com baixo ponto fusão (Figura 17). Tal extrusor foi testado em uma máquina FDM comercial da *Stratasys* (BOWYER, 2004).

Figura 17 - Primeiro extrusor desenvolvido pela equipe RepRap (BOWYER, 2004).

Neste mesmo artigo, foi divulgada a ideia de criar uma máquina de prototipagem de baixo custo capaz de replicar suas próprias peças. Desde então, o RepRap tem evoluído e atualmente é capaz de gerar 60% de suas peças (desconsiderando parafusos e porcas). Os colaboradores do projeto RepRap divulgam diversos vídeos na internet, os quais exibem o processo de construção dos protótipos, que vão desde peças para máquinas RepRap até pequenos utilitários, como pendurador de roupa, maçaneta, copos, etc. Na Figura 18 é possível observar duas máquinas de prototipagem criadas pelo projeto.

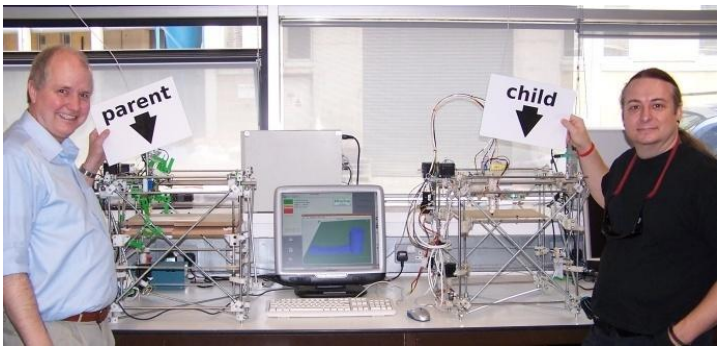


Figura 18 - Modelo atual do RepRap (BOWYER, 2006).

Visão geral do software do RepRap

O modelo básico de funcionamento do software do RepRap é constituído por duas aplicações desenvolvidas em Java e por um software embarcado executado em um microcontrolador. O primeiro programa (RepRap) é responsável por carregar os arquivos tridimensionais e realizar o planejamento para deposição. O segundo (ReplicatorG) interpreta as rotas geradas pelo RepRap, organiza estas rotas e as repassa para o módulo embarcado através de comunicação serial (RS232) ou USB. O módulo embarcado recebe os diversos comandos e deve controlar os sensores e atuadores da máquina, a fim de se obter o protótipo final. O software RepRap e o ReplicatorG são compatíveis com os sistemas operacionais Windows, Linux e Mac OS X.

A Figura 19 mostra os componentes de software do modelo RepRap.

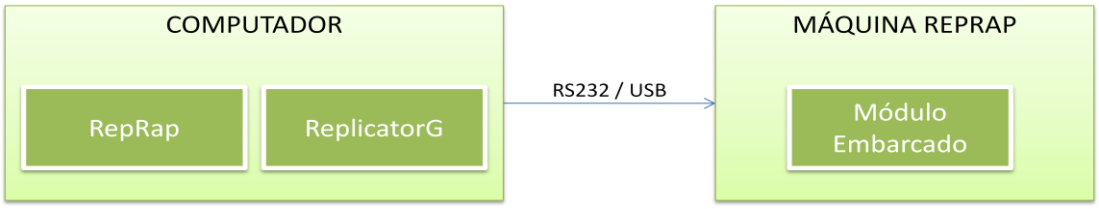


Figura 19 - Visão dos módulos de software do projeto RepRap.

Módulo RepRap

Assim que o RepRap é executado, o usuário deve carregar um arquivo no formato STL para ser prototipado. Os próximos passos são interpretar este arquivo, gerar as diversas fatias correspondentes ao modelo e planejar as rotas de deposição de cada fatia.

Uma biblioteca Java é responsável por interpretar a informação contida no arquivo STL em um objeto 3D. Esta operação realiza a conversão da lista de triângulos tridimensionais em um objeto na memória. Em seguida, o objeto é fatiado em camadas.

Fatiamento do Objeto 3D

A técnica desenvolvida para fatiar o arquivo consiste em gerar um plano em uma determinada altura Z. Posteriormente, calcula-se a interseção de todas as faces do arquivo STL com este plano. As faces do arquivo STL que interceptarem o plano serão adicionadas ao domínio da fatia. A Figura 20 ilustra como ocorre este processo (BOWYER, 2006).

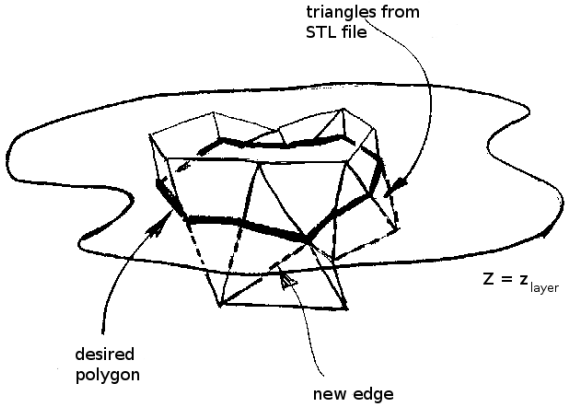


Figura 20 - Técnica de fatiamento desenvolvida pelo RepRap (BOWYER, 2006).

Ao final deste processo, é gerada uma lista de segmentos de linhas. Cada segmento é resultado do ponto de interseção do plano. Porém, há uma imprecisão que precisa ser corrigida: durante o fatiamento, as linhas geradas não se fecham perfeitamente. Os pontos iniciais e finais das linhas não coincidem, gerando sobreposição das linhas ou espaços vazios entre os segmentos de reta. A Figura 21 traz um exemplo da situação descrita (BOWYER, 2006).

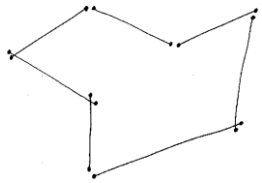


Figura 21 - Disjunções geradas pelo fatiamento (BOWYER, 2006).

Para resolver este problema e corrigir as imperfeições geradas pelo fatiamento, é aplicado um algoritmo denominado *QuadTree* (BERG et al., 2008). São considerados apenas os pontos inicial e final de cada linha. Estes são divididos recursivamente e, ao fim do processamento, os pontos que estiverem mais próximos de acordo com a ordenação do algoritmo serão unidos. Este procedimento é responsável por gerar as bordas do objeto.

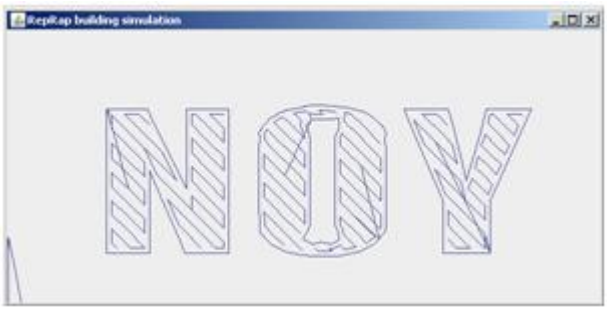
Tal processo é passível de erros, pois a forma de determinar as bordas da camada não é a ideal. Estes erros não ocorreriam caso fosse utilizado o método tradicional de fatiamento, descrito por Gibson (2002).

Preenchimento

A próxima etapa vai definir o preenchimento da fatia a ser prototipada. A aplicação RepRap disponibiliza uma opção para visualizar a rota para cada fatia. A análise das rotas é feita com base nesta simulação, pois não há no projeto documentação referente a este processo. Das diversas fatias analisadas, duas foram destacadas para exemplificar o funcionamento.

Como pode ser observado na Figura 22a e Figura 22b, o algoritmo de preenchimento utilizado é o zigzag. Analisando a imagem do retângulo, é possível perceber que o RepRap consegue gerar de forma satisfatória a rota desejada.

a - Preenchimento de um retângulo



b - Preenchimento da palavra NOY

Figura 22 - Preenchimento em zigzag do RepRap.

Uma função que poderia ser melhorada é a ligação entre a deposição da borda e o caminho de ferramenta do preenchimento. Na Figura 22a, destaca-se em vermelho que a ligação entre o fim da borda e o início do preenchimento é feita no meio da rota. Isso faz com que o preenchimento interior seja quebrado em duas partes. Caso a ligação fosse realizada com um dos pontos extremos do preenchimento, seria necessário apenas um caminho contínuo.
Outra consideração relevante é que em alguns momentos houve perda de precisão nas bordas. Na Figura 22b, percebe-se que a borda interna da letra "O" sofreu uma forte distorção nas partes inferior e superior. Estas observações foram inferidas através da utilização do software RepRap. Não há como saber a origem desta falha. Uma possível explicação para a ocorrência da mesma está no método utilizado para a ligação e definição das bordas durante o fatiamento que, como foi observado, diverge da literatura na ligação das rotas.

Integração com o hardware

A próxima etapa do processo é passar as rotas geradas para a máquina de prototipagem rápida. Na máquina, há um software embarcado responsável por interpretar e atender cada um dos comandos recebidos. Para viabilizar esta comunicação, é necessária a definição de um protocolo de comunicação que possibilite tal integração. No modelo RepRap há duas possibilidades:

1. Integração através do protocolo SNAP

O SNAP (*Scaleable Node Address Protocol*) é um protocolo de rede aberto desenvolvido pela HTH com o objetivo de ser um protocolo genérico que forneça flexibilidade e escalabilidade. Ele pode ser usado tanto em sistemas com processamento e memória limitados quanto em sistemas maiores e que exijam um maior poder computacional. O principal fator que permite tal flexibilidade é o fato do SNAP possibilitar diferentes tamanhos de pacotes, podendo ser utilizado com ou sem *flags* ou detecção de erros (HTH, 2002).
O RepRap utiliza o SNAP através de comunicação serial. O principal motivo para adoção do SNAP é a possibilidade de diminuir o esforço no futuro, caso sejam criadas máquinas RepRap com comunicação via rede. A topologia estabelece a existência de apenas dois nós na rede e segue o padrão mestre/escravo. O mestre é o computador que envia os comandos pela porta serial para o escravo (máquina RepRap).

2. Integração através de código G

Neste caso, o usuário irá gerar um arquivo seguindo o padrão de código G. Em seguida, basta carregar este arquivo e solicitar que o software ReplicatorG repasse a série de comandos para o software embarcado.

O código G (*g code*) foi criado originalmente para as máquinas de controle numérico. O objetivo era padronizar o controle de máquinas de diversos fabricantes. Um arquivo que siga este padrão contém uma série de comandos que devem ser interpretados e seguidos na ordem que se encontram no arquivo. Existem algumas normas que visam definir esta linguagem, dentre as quais é possível destacar: RS-274D e ISO 6983 (FALCK, 2008).

A Figura 23 mostra uma captura da tela do RepRap, na qual está marcada de vermelho a área do software em que o usuário pode decidir pelas opções citadas anteriormente.

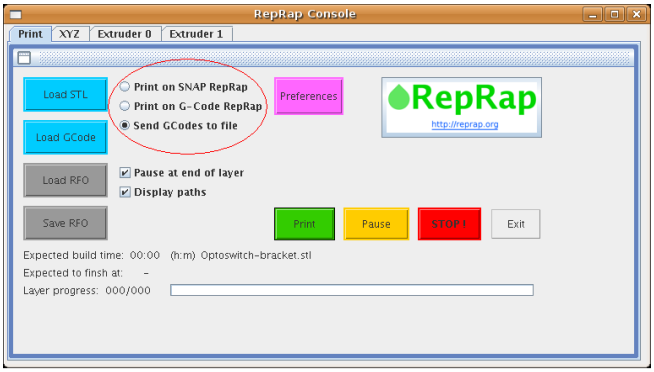


Figura 23 – Tela de decisão do protocolo de comunicação

Replicator G

O ReplicatorG é um módulo de *software* responsável por interpretar o código G gerado pelos sistemas CAM e repassá-los para a máquina de prototipagem. Originalmente foi desenvolvido para ser utilizado em máquinas RepRap, porém pode ser facilmente integrado para ser utilizado com outros projetos. O MakerBot (PETTIS et al., 2009), por exemplo, reutiliza o ReplicatorG e foca sua pesquisa apenas no desenvolvimento do *hardware*. A Figura 24 mostra como ocorre o fluxo da informação no processo de prototipagem.

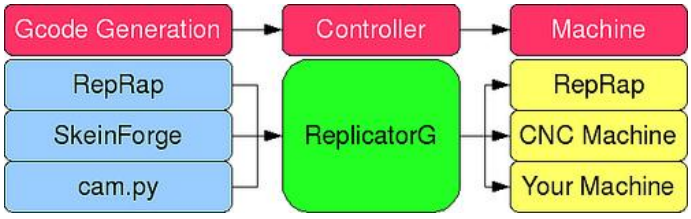


Figura 24 - Visão geral da arquitetura do ReplicatorG

O procedimento básico para execução do software consiste em carregar o arquivo G, gerado pelo módulo RepRap, e solicitar a prototipagem do arquivo. O Replicator G irá controlar a

máquina de forma a atender o caminho desejado. Uma função importante do ReplicatorG é a de simulação do código G. Assim, é possível comprovar se o código gerado pelo RepRap está adequado no que diz respeito à trajetória de preenchimento das camadas. A Figura 25 demonstra que o caminho de ferramenta das rotas de preenchimento não é ordenado.

Figura 25 - Simulação do preenchimento no ReplicatorG.

É possível perceber que, por diversas vezes, o extrusor alternou desnecessariamente entre as letras “N” “O” e “Y” que seriam prototipadas nesta camada. Esta característica do software RepRap aumenta o tempo total de prototipagem e o número de vezes que o extrusor é pausado e reiniciado novamente, com todas as implicações daí advindas.

Skeinforge

Este sistema é um dos subprodutos do RepRap. Foi criado com o objetivo de auxiliar no processo de planejamento das camadas. Dentre as soluções de *software* analisadas, o S*keinforge* se mostrou a mais estável e com maior possibilidade de adaptação. Além disso, possui compatibilidade com os sistemas operacionais Windows, Linux e Mac OS X.

Este *software* é dividido em módulos, cada um responsável por realizar uma determinada tarefa do processo de planejamento. O primeiro módulo a atuar no modelo 3D é denominado *carve*, que é responsável pelo fatiamento do sólido. Após a conclusão do processo, é gerado *gcode* para representar as camadas.

Há uma ordem pré-definida para os módulos atuarem sobre o arquivo G gerado. A entrada e saída destes módulos é o mesmo arquivo G criado no fatiamento. Cada módulo recebe o *gcode* como entrada, realiza as intervenções de acordo com seus parâmetros de configuração e, em seguida, passa para o próximo módulo o *gcode* modificado. Caso o módulo esteja desabilitado, este apenas repassa, sem qualquer modificação, o *gcode* recebido.

No *skeinforge* há diversos módulos que permitem uma infinidade de combinações de configurações. A seguir, são destacados os principais módulos e o que pode ser feito de modificação no processo de planejamento:

- *Carve*: responsável pelo fatiamento do sólido. Um parâmetro importante que este módulo possui é denominado *Layer Thickness (mm)*. Este parâmetro define, em milímetros, a espessura utilizada para as camadas;

- *Clip:* permite configurar o acúmulo de material em partes da peça. Os parâmetros deste módulo podem ser utilizados para evitar, por exemplo, a sobreposição de deposição em uma mesma camada.
- *Fill:* define a estratégia de preenchimento utilizada para a peça. Neste módulo, há três opções de preenchimento: em quadrados, linhas ou hexagonal. Parâmetros como a distância entre os diversos filamentos e a espessura dos filamentos depositados podem ser definidos.
- *Raft:* utilizado para customizar as configurações de suporte. O *Skeinforge* gera malhas que servem como base para as partes sólidas da peça. Estas malhas podem ter suas dimensões configuradas de acordo com a necessidade do usuário. A temperatura em que o material de suporte será depositado também pode ser definida.

As imagens a seguir mostram o sistema em funcionamento. A primeira imagem (Figura 26) mostra a tela inicial do software, em que é possível definir os parâmetros utilizados na manipulação do modelo 3D.

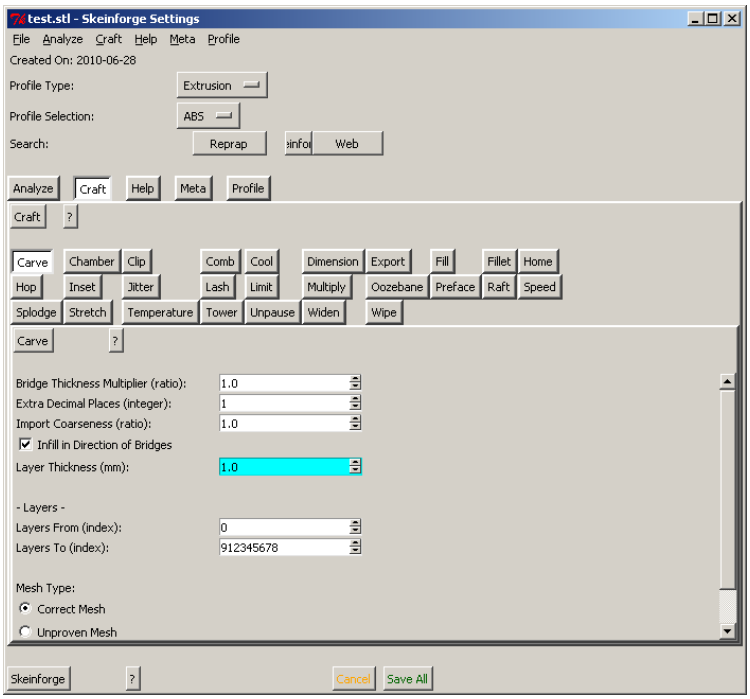


Figura 26 - Tela inicial do Skeinforge.

A próxima imagem (Figura 27) mostra uma visualização que o software disponibiliza após concluir o fatiamento e planejamento de todas as camadas. Na imagem, é possível visualizar todas as camadas geradas de forma simultânea.

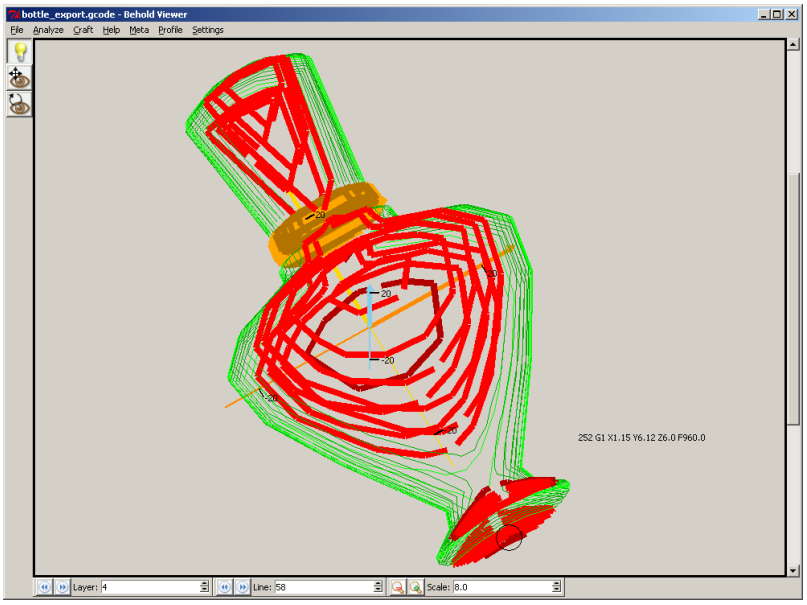


Figura 27 – Visualização das camadas geradas a partir do arquivo STL.

A imagem a seguir (Figura 28) mostra que o *software* permite também a visualização de apenas uma das camadas do sólido. É possível ver também a sequência de segmentos de reta que representam os vários comandos G salvos em arquivo.

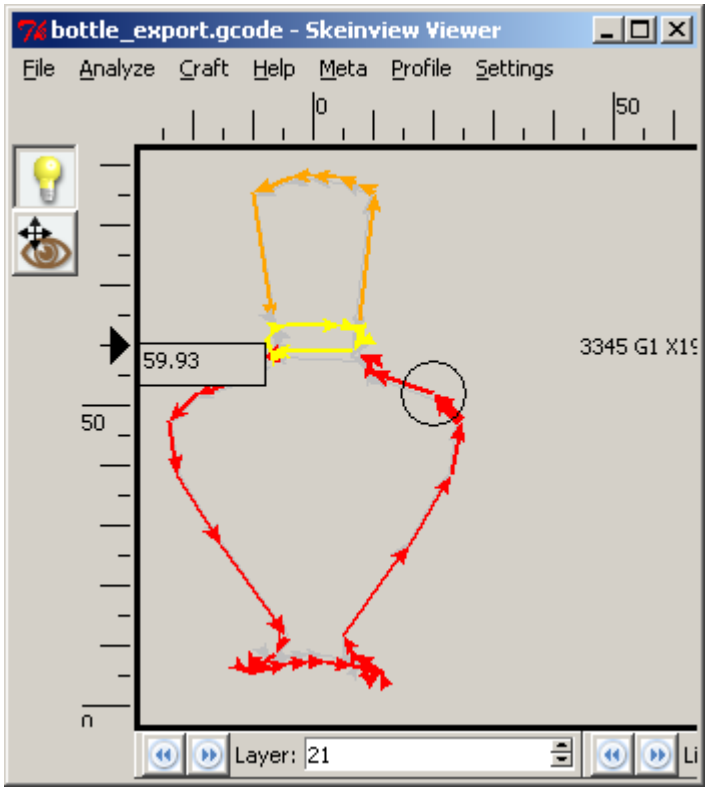


Figura 28 – Planejamento de uma das camadas.

Pode-se utilizar o código G gerado pelo sistema em máquinas compatíveis com o *ReplicatorG*. Por isso, o *Skeinforge* tem sido bastante utilizado em máquinas, como a

MakerBot (PETTIS et al., 2009) e Bits From Bytes (BITSFROMBYTES, 2010). Estes são vertentes do RepRap que direcionam suas pesquisas no desenvolvimento de novas soluções de *hardware*.

FAB@HOME

Este projeto foi iniciado em 2006, na universidade de Cornell, pelos professores Hod Lipson e Evan Malone. Da mesma forma que os outros projetos de pesquisa, a idéia era produzir uma máquina de prototipagem rápida de baixo custo (MALONE e LIPSON, 2007).

Uma das diferenças básicas entre o Fab@Home e o RepRap é o fato do Fab@Home utilizar uma seringa para fazer a injeção de material à temperatura ambiente, enquanto que no RepRap é utilizado um extrusor para aquecer a matéria-prima. Este fato faz com que o custo de uma máquina Fab@Home seja menor. Em contrapartida, a qualidade final do protótipo também é inferior, quando comparado às máquinas RepRap.

Uma observação importante é o fato de o sistema executado no computador ser compatível apenas com o sistema operacional Windows (LIPSON e MALONE, 2010). A Figura 29 mostra a tela principal do sistema.

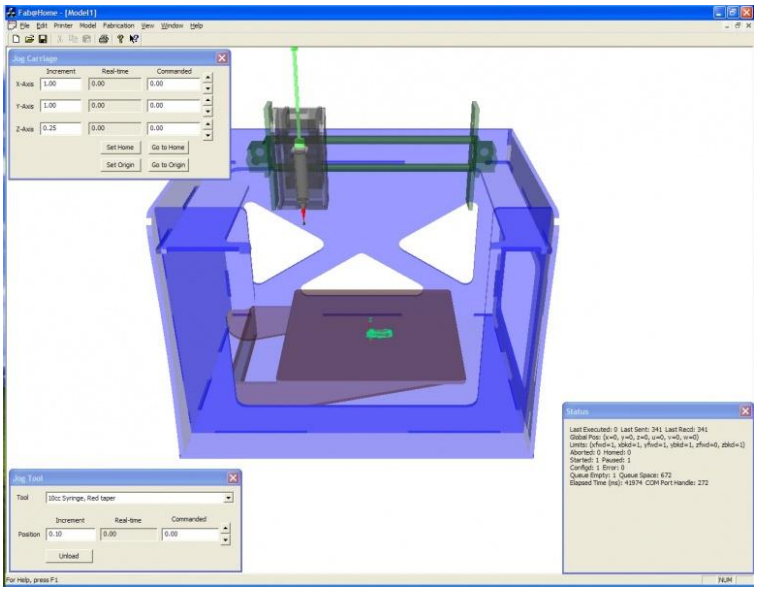


Figura 29 – Tela inicial do Fab@Home.

É possível carregar até 5 arquivos STL simultaneamente. Esta função é importante para construir objetos mais complexos. Nesse caso, o objeto deve ser decomposto em partes, prototipado e, ao final, suas partes podem ser montadas manualmente.

É permitido ao usuário manipular a geometria do objeto tridimensional carregado. As dimensões do sólido podem ser alteradas e também é possível rotacioná-lo em qualquer direção.

Após realizar o processo de planejamento, o sistema gera como saída um arquivo com a extensão “fab”. Este arquivo contém as instruções que serão interpretadas pela máquina Fab@Home para realizar a prototipagem. A utilização deste formato incompatibilizava o projeto com outras soluções, como: RepRap, Makerbot e Bits from Bytes. Para contornar tal situação, foi criado um aplicativo denominado Fab2GCode (LIPTON, 2010). A função deste software é converter um arquivo no formato “fab” para código G, tornando possível a compatibilização do sistema Fab@Home com as máquinas de outros grupos de pesquisa.

3 Modelo conceitual

A partir da análise do estado da arte da prototipagem rápida, foi definido o ambiente em que a proposta deste trabalho está inserida. O objetivo é atender às etapas compreendidas ao software no PC (*personal computer*). Porém, para validar o sistema é necessário atuar em conjunto, pois, na prototipagem rápida, o software PC é integrante de um sistema maior, que envolve também um software embarcado e um hardware que serão responsáveis por atender aos comandos do software e realizar as tarefas necessárias para gerar o protótipo final.

Como dito, esta dissertação é oriunda do trabalho de um grupo de pesquisa formado na UFBA, focado na prototipagem rápida. O objetivo inicial deste grupo é gerar um protótipo de uma máquina de manufatura aditiva. Esta meta será alcançada através da junção do presente trabalho com o de Costa (2011), que endereça os aspectos eletromecânicos do sistema desenvolvido.

A Figura 30 mostra o ciclo completo definido para a prototipagem de um objeto, e as áreas relacionadas ao software PC, que estão incluídas no escopo deste trabalho.

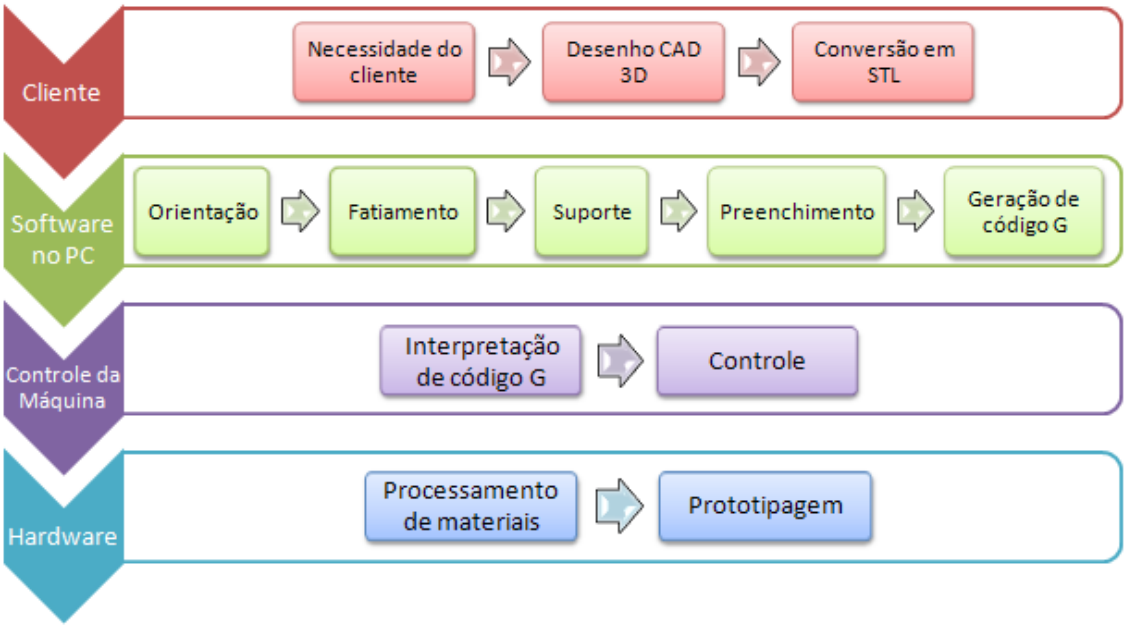


Figura 30 - Escopo deste trabalho.

A necessidade do cliente de prototipar um determinado objeto define o início de todo o processo. A idéia do objeto é, então, modelada em um software CAD e, em seguida, exportada para o formato STL. A partir deste ponto, inicia-se a interação com o sistema de prototipagem, que é o escopo deste trabalho. O formato STL é, portanto, a interface de entrada do sistema.

Assim que o arquivo STL é carregado pelo software, deve ser escolhida uma orientação para o objeto. Em seguida o fatiamento é realizado, sendo definidas também as estruturas de

suporte. Para cada camada obtida, será definida a rota de preenchimento. Ao fim deste processo, o software no PC deve gerar a sequência de comandos planejadas, seguindo o padrão de código G (ISO 6983).

O sistema responsável por controlar a máquina de prototipagem irá carregar o arquivo gerado e interpretar os comandos definidos. Cada comando será executado no hardware da máquina de prototipagem na ordem em que estão definidos no arquivo. Desta forma, ao final do processo, será obtido o sólido tridimensional. Este deve estar de acordo com o modelo vislumbrado pelo cliente no seu software CAD de modelagem 3D.

A proposta do presente trabalho é oferecer um modelo genérico, capaz de ser gerado em qualquer linguagem de programação. Porém, para fins de validação, é necessário codificar a proposta. A linguagem Java foi escolhida para desenvolvimento do software, com base em algumas vantagens oferecidas pela linguagem em relação aos requisitos do software:

- **Orientação a objetos:** este paradigma, além de trazer conceitos que ajudam na codificação, possui um importante facilitador nas fases de análise, projeto e documentação que são os diagramas UML (*Unified Modeling Language*);
- **Multi-plataforma:** como o Java trabalha com o conceito de máquina virtual, o programa desenvolvido nesta linguagem pode rodar em diversos sistemas operacionais sem necessitar ser recompilado. Isto acontece porque o código fonte é compilado para uma linguagem intermediária denominada b*ytecode*. No momento da execução, a máquina virtual é responsável por traduzir para linguagem de máquina o programa em execução.
- **Suporte à computação 3D:** A Oracle, empresa responsável pelo desenvolvimento da linguagem Java, disponibiliza uma API (A*pplication Programming Interface*) para permitir que o desenvolvedor crie aplicações que necessitam de computação gráfica tridimensional. Esta API é denominada JAVA 3D API (SUN, 2000).

A JAVA 3D API possui alguns conceitos e classes importantes. A seguir, estes aspectos serão identificados para viabilizar o entendimento da conversão do modelo conceitual em código fonte.

No Java, para gerar um ambiente 3D, é necessário criar um mundo virtual denominado *VirtualUniverse*. O *VirtualUniverse* contém uma série de elementos que podem controlar ou influenciar este mundo. A classe *Shape3D* representa qualquer objeto visível e encapsula todas as informações do objeto dentro do Universo. Ao *Shape3D* podem ser associadas outras informações que vão determinar como o objeto será mostrado na tela do computador (SELMAN, 2002).

Cada objeto existente dentro do universo virtual pode conter formas e posições diferentes no espaço. Existe uma classe denominada *SceneGraphObject*, que é responsável por manter todas estas informações. Esta classe armazena seus dados em forma de árvore hierárquica. Cada nó na árvore é armazenado através da classe *Node*. Um nó pode ser do tipo grupo ou nó folha. Um nó grupo possui filhos que trazem informações pertinentes a um determinado objeto. Enquanto o nó folha especifica os elementos utilizados pelo Java 3D durante a renderização (SUN, 2000). A Figura 31 demonstra como essa estrutura é organizada.

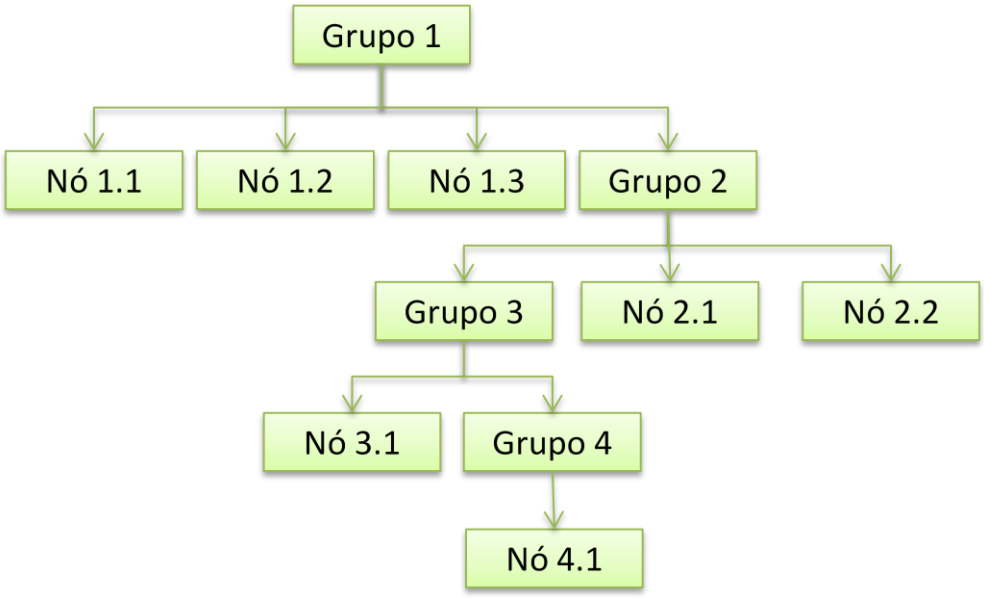


Figura 31 – Estrutura de dados mantida em um SceneGraphObject, adaptado de (SELMAN, 2000)

Além do *Shape3D*, que irá representar um objeto renderizado, há outras quatro classes importantes que podem estar associadas a este grafo de cena: *Locale, BranchGroup, TransformGroup e ViewPlatform*.

Cada universo virtual precisa possuir pelo menos um objeto *Locale* associado. Este irá agrupar todos os objetos de uma determinada região no grafo de cena. É através do *Locale* que são definidas as dimensões do objeto em um determinado universo (SELMAN, 2002).

O *TransformGroup e BranchGroup* estão associados ao *Locale.* Enquanto o *BranchGroup* vai agrupar todos os objetos pertencentes a um *Locale*, o *TransformGroup* armazena as informações de rotação, translação e escala desses objetos.

O *ViewPlatform* representa um ponto de observação do universo. A partir desta classe, é possível renderizar os diversos objetos do universo na tela do monitor. O volume de visão deste objeto será determinado pela distância entre dois parâmetros que são configuráveis: *near clipping plane* (plano de visão mais próximo) e *far clipping plane* (plano de visão mais

distante). Só serão renderizados os objetos do universo que estiverem dentro do volume de visão. A Figura 32 ilustra este processo.

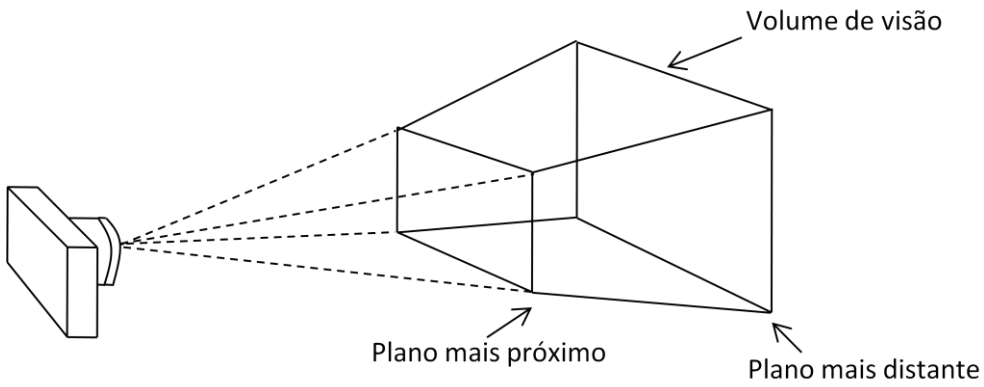


Figura 32 - Modelo de visualização das câmeras

A partir desses conceitos, a proposta deste trabalho foi montada. A seguir, será detalhado como os recursos disponibilizados pela API Java 3D foram aplicados de forma a atender as necessidades de um sistema de prototipagem rápida. Será demonstrado como o universo virtual foi criado e de que forma este é manipulado a fim de realizar as etapas do processo de planejamento da manufatura aditiva (proposta deste trabalho).

Criação do universo virtual

O universo virtual inicialmente possui apenas uma câmera. O volume de visão (*view frustum*) desta câmera é reduzido. Assim, à medida que um objeto estiver passando por dentro da câmera, esta será capaz de visualizar apenas uma fatia deste objeto. A Figura 33 demonstra o estado inicial do universo no momento de sua criação.

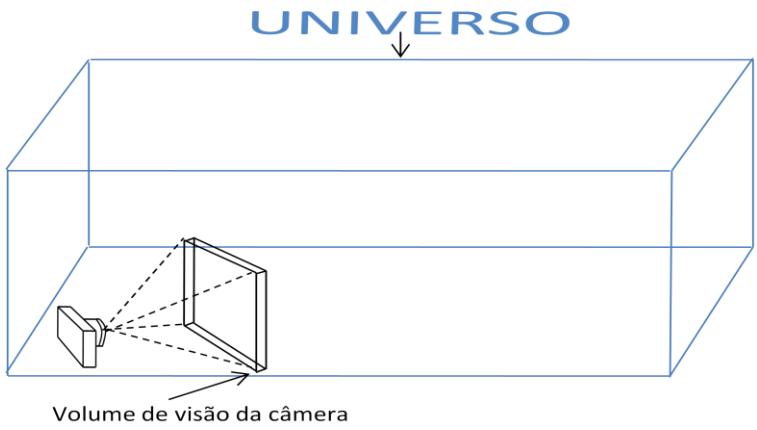


Figura 33 - Estado inicial do universo

Neste estado, o universo virtual está preparado para carregar os modelos tridimensionais a serem prototipados. Quando a importação é realizada, o objeto será carregado como um

Shape3D. Outras informações associadas ao comportamento do objeto STL são necessárias, pois é preciso que este realize algumas tarefas. Será criada uma classe denominada *ObjetoSTL*. A esta classe estarão associadas todas as transformações necessárias ao objeto dentro do universo, como: mover, girar, definição de cor, alteração de escala, etc. A Figura 34 exemplifica como ficaria o universo após carregar um cilindro.

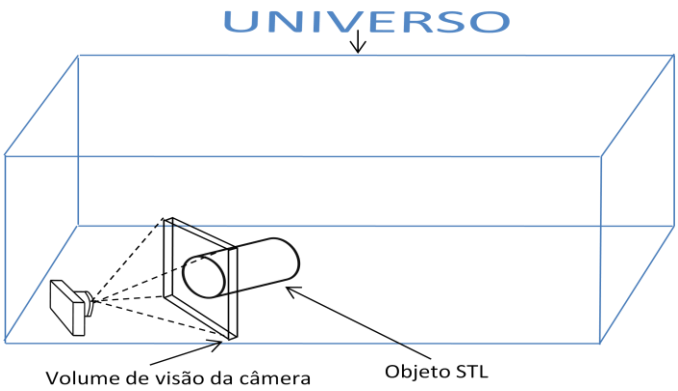


Figura 34 - Universo com um objeto STL carregado

Quando o universo se encontra com um arquivo tridimensional carregado, o seu grafo de cena fica da seguinte maneira (Figura 35):

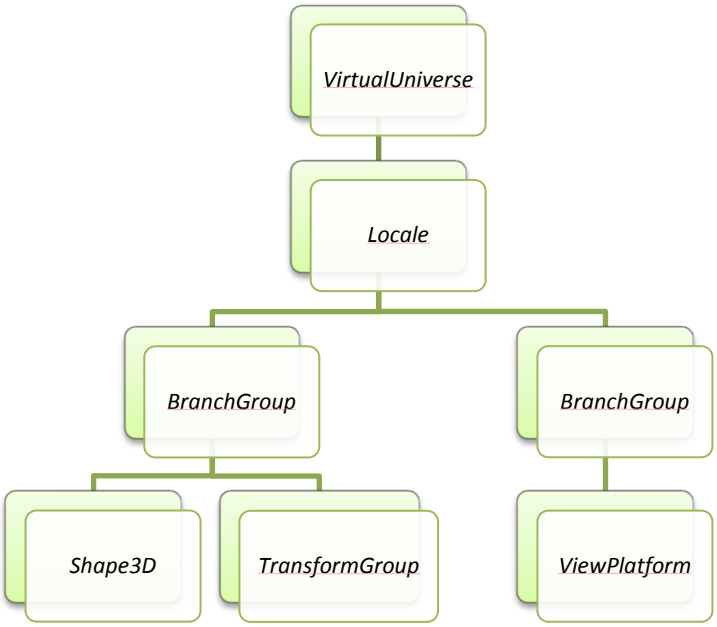


Figura 35 - Grafo de cena

É importante ressaltar que, como observado no capítulo 2, o formato STL não possui definição de escala. Esse é um problema a ser tratado pelo sistema, pois não se sabe quais as dimensões projetadas pelo usuário inicialmente em seu modelo CAD. A solução proposta para este problema é utilizar o *Locale* como referencial. Com base no *Locale* do universo, é

possível aplicar transformações ao *BranchGroup* do *Shape3D* para determinar o tamanho do objeto. Desse modo, é possível determinar quais as dimensões do objeto STL.

Neste ponto, temos o modelo projetado no CAD carregado no universo. O sistema agora precisa realizar cada uma das atividades incluídas no escopo deste projeto: orientação, fatiamento, suporte e preenchimento.

3.1 Orientação

A orientação definirá a posição em que o objeto será prototipado. No momento em que o arquivo é exportado para o formato STL, a definição das normais define a orientação inicial. Além desta orientação, o usuário pode escolher entre outras cinco opções. A Figura 36 exemplifica as orientações possíveis para um cubo. Uma das formas de validar esta funcionalidade é carregando um dado e verificando o número que ficará no topo. Para cada opção de orientação, um número diferente deve aparecer.

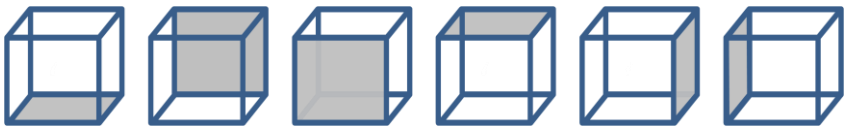


Figura 36 – Opções de orientação

Para realizar a rotação do modelo STL, o Java 3D API possui uma classe denominada *Transform3D*, que pode ser associada ao *TransformGroup*. A classe *Transform3D* disponibiliza uma abstração para o desenvolvedor, a qual facilita nas operações de rotação, definição de escala e translação. A Figura 37 mostra como fica o grafo de cena após adicionar as funcionalidades de transformações ao *Shape3D*

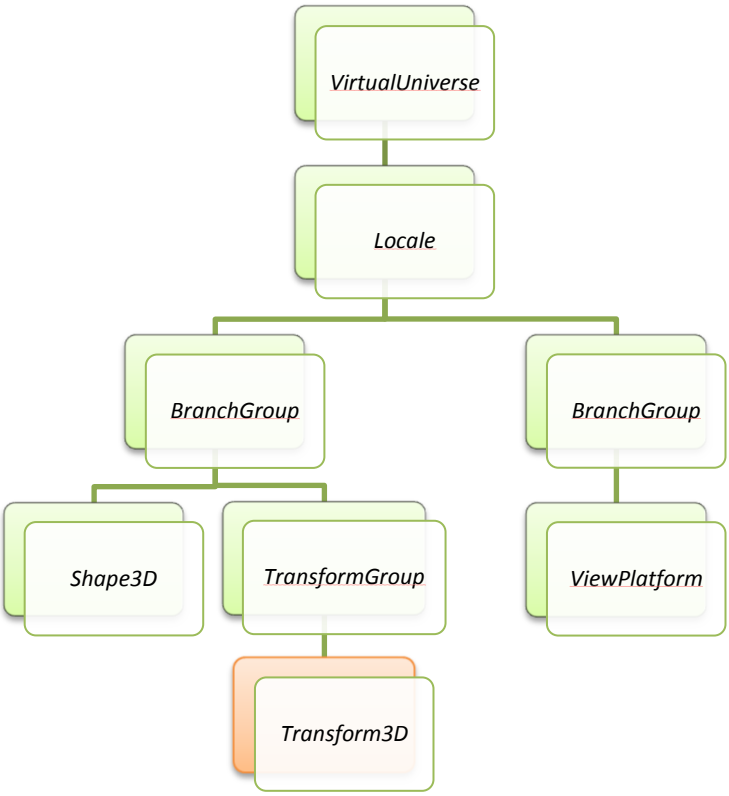


Figura 37 – Grafo de cena após adicionar o Transform 3D

A classe *Transform3D*, destacada no grafo de cena, também será utilizada em outras tarefas posteriores, pois é através dela que será definida a posição inicial e translação do objeto dentro do universo.

3.2 Fatiamento

Definida a orientação do objeto, o próximo passo é realizar o fatiamento em camadas. Para realizar esta atividade o primeiro passo consiste em calcular a posição inicial do objeto. Esta posição será calculada com base na orientação e escala do objeto. Dados estes valores, é preciso definir quais seriam os pontos X,Y e Z para o objeto no universo, de forma a posicioná-lo próximo ao volume de visão da câmera.

Através do Java 3D, é possível obter o ponto mais alto e mais baixo de um *Shape3D* em relação ao universo. Como os pontos do início do volume de visão da câmera são sabidos (definidos no momento da criação da câmera), basta transladar o objeto, de forma que o seu ponto máximo fique o mais próximo do volume de visão.

Feito isso, o objeto está preparado para ser fatiado. Nesse momento, deve-se utilizar novamente a classe *Transform3D* para fazer o *Shape3D* caminhar pelo universo. Aplicando um movimento retilíneo e constante sobre este objeto em direção à câmera, esta passará a visualizar as camadas do objeto. Desta forma, as fatias do objeto são visualizadas à medida

que o mesmo vai atravessando o volume de visão da câmera. A Figura 38 exemplifica como ocorreria o fatiamento de um cubo.

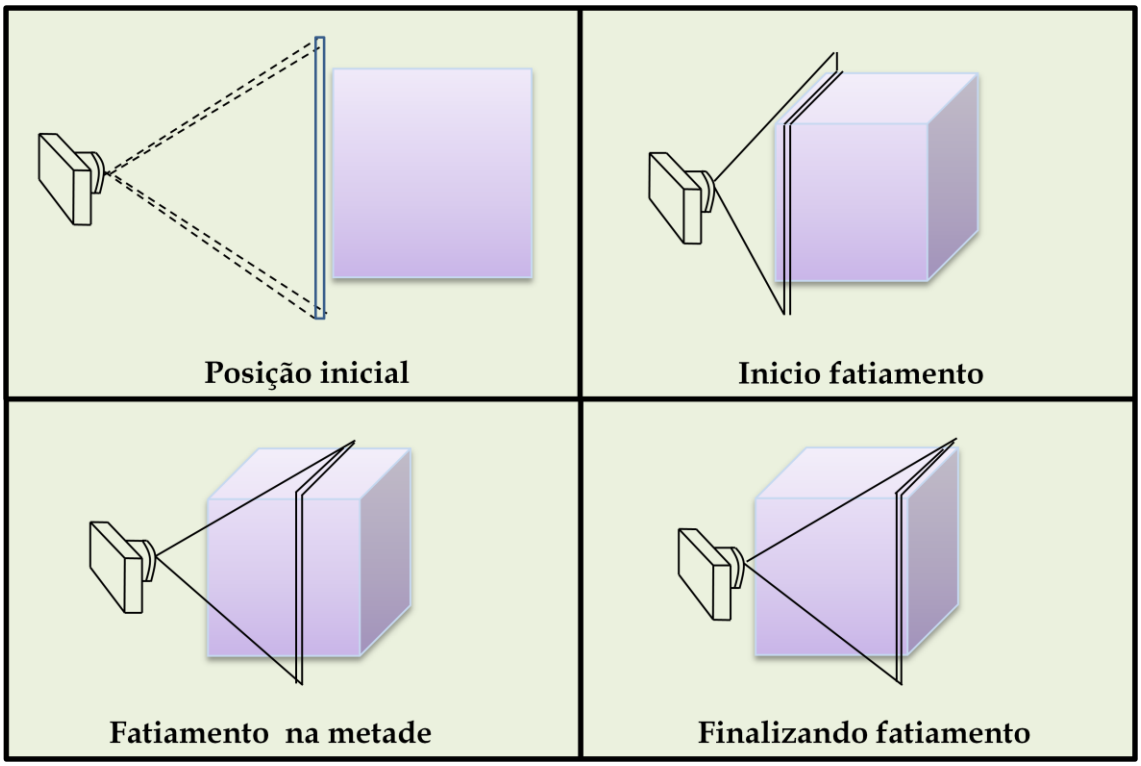


Figura 38 - Processo de fatiamento

Para gerar as diversas fatias de um determinado objeto tridimensional, basta fazer com que este atravesse a câmera. A quantidade de fatias obtidas é inversamente proporcional ao incremento de translação do objeto, ou seja, quanto maior for o deslocamento do sólido, menor a quantidade de fatias geradas e, consequentemente, menor a quantidade de detalhes obtidas deste objeto. Desta forma, a qualidade do fatiamento pode ser facilmente configurada através da definição do incremento de translação do objeto.

A cada vez que a imagem da câmera for alterada, esta deve capturar a nova imagem e salvá-la como uma fatia. Neste momento, estamos convertendo a informação tridimensional (sólido obtido a partir do arquivo STL) em bidimensional (imagem obtida a partir da renderização da câmera).

Com esta proposta, segundo a classificação de Dragomatz e Mann (1997), temos um modelo de fatiamento baseado em *pixels*. Uma vantagem obtida a partir desta conversão é a possibilidade de utilizar diversos algoritmos associados às técnicas de processamento de imagens. Será descrito nas próximas etapas como estes algoritmos foram utilizados.

3.3 Preenchimento

Ainda considerando o exemplo do cubo, com o fatiamento teríamos uma lista de camadas em que as intermediárias seriam iguais, enquanto a primeira e última camada seriam as extremidades do cubo, como demonstrado na Figura 39.

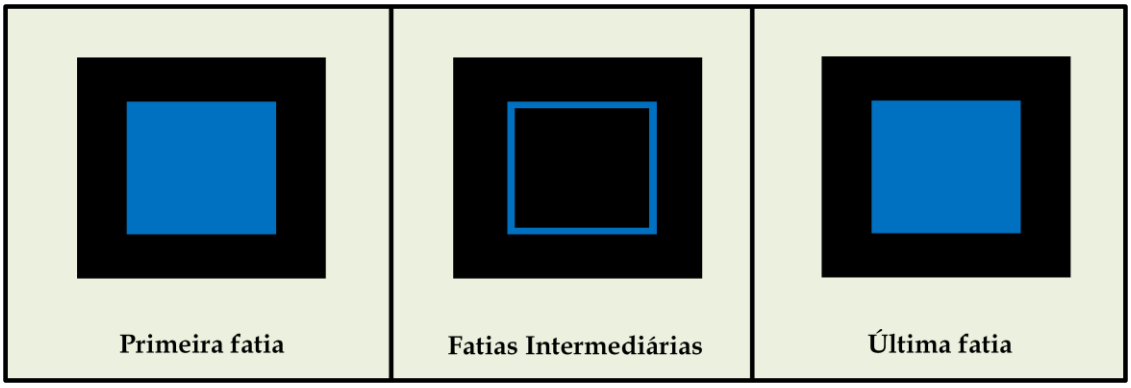


Figura 39 - Images geradas com o fatiamento do cubo

Para aumentar a qualidade superficial do sólido final, a deposição de material deve ser iniciada pelos contornos da fatia. Em seguida, o restante da fatia deve ser feito utilizando algum algoritmo de preenchimento.

Observando a Figura 39, é possível perceber que as imagens obtidas a partir do fatiamento possuem apenas duas cores. Os *pixels* em azul representam o objeto, enquanto os pretos representam o plano de fundo. Para obter as bordas de cada imagem, foi aplicado um algoritmo para extração de contornos em imagens binárias. A lógica deste algoritmo consiste em percorrer toda a imagem à procura dos pixels que tiverem a cor azul. Quando um pixel desta cor for encontrado, observa-se seus vizinhos e, caso ao menos um dos oito vizinhos tenha a cor do plano de fundo (preto), o pixel atual deverá ser marcado como borda. Esta abordagem foi baseada nos algoritmos propostos por Costa e Cesar (2001). A seguir (Figura 40), temos o pseudo-código que demonstra a implementação deste algoritmo.

```
Para cada pixel p(x,y) faça
  saida(x,y) = PRETO

 se p(x,y) = AZUL
   se p(x,y+1) = PRETO ou p(x,y-1) = PRETO ou p(x+1,y) = PRETO ou p(x-1,y)= PRETO
         saida(x,y) = BRANCO
```

Figura 40 - Algoritmo para detecção de bordas, adaptado de (COSTA e CESAR, 2001).

Aplicando esse algoritmo à Figura 39, seriam obtidos apenas os contornos de cada fatia. O resultado está representado na Figura 41

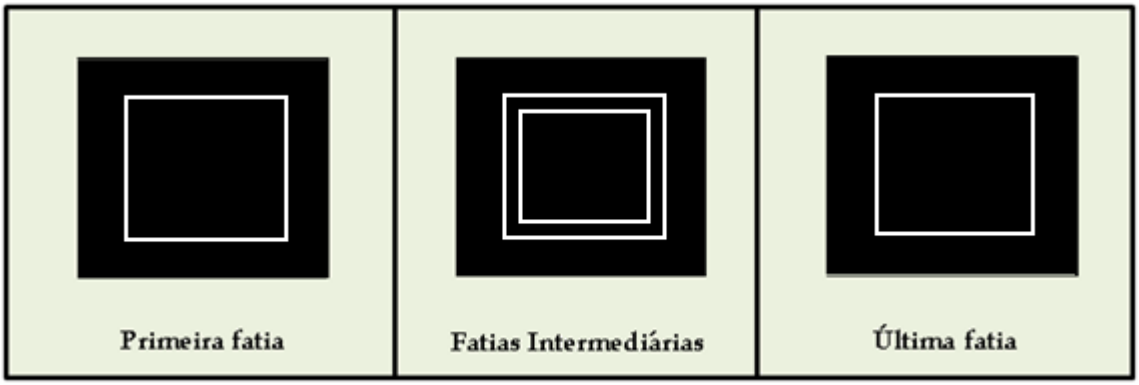


Figura 41 - Bordas das fatias do cubo.

Com as imagens obtidas na Figura 41, foi possível identificar a borda de cada uma das fatias. Em seguida, dois novos algoritmos são aplicados para organizar a deposição dessas bordas. O primeiro algoritmo visa segmentar as diversas rotas presentes na imagem. O segundo busca gerar a rota de deposição para cada segmento obtido. O algoritmo de segmentação é importante, pois permitirá ao algoritmo de geração de rotas tratar de forma independente as diversas rotas obtidas em uma mesma camada.

A técnica de segmentação de imagens utiliza o processo de rotulação (*labelling*). Neste processo, a imagem é percorrida e, no instante em que um *pixel* diferente da cor de fundo for encontrado, lhe será atribuído um rótulo *(label).* Neste momento, seus vizinhos serão observados e, a todos aqueles que forem da mesma cor, será atribuído o mesmo rótulo. Esta operação será realizada recursivamente até que mais nenhum vizinho seja rotulado. Posteriormente, a imagem volta a ser percorrida em busca de novos *pixels* (MARCHAND-MAILLET e SHARAIHA, 2000). A Figura 42 ilustra o funcionamento do algoritmo. Na imagem inicial, o "0" representa os pixels na cor branca, enquanto o "x" representa os *pixels* de preto.

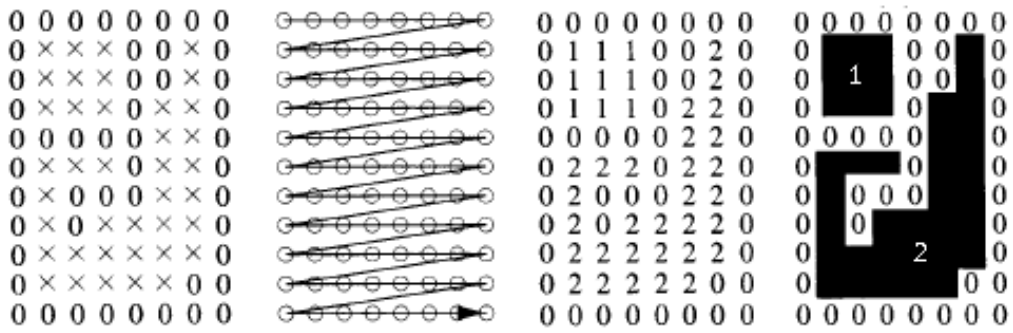


Figura 42 - Processo de segmentação da imagem, adaptado de (Marchand-Maillet e Sharaiha, 2000)

Após a execução do algoritmo de segmentação, obtêm-se as imagens de cada segmento de forma isolada. Em seguida, é preciso analisar um destes segmentos de forma a gerar uma rota de deposição contínua, evitando que o extrusor necessite ser ligado e desligado por diversas vezes, reduzindo o tempo de prototipagem.

Esta situação apresenta um problema clássico para sistemas computacionais científicos ou de engenharia: encontrar um caminho através de um grafo. Há duas abordagens que permitem resolver este tipo de problema (HART et al., 1968):

Abordagem matemática: nesta forma de tratar o problema, há uma preocupação em encontrar um caminho ótimo, porém, sem atentar-se à viabilidade computacional da solução.

Abordagem heurística: utiliza conhecimento sobre o domínio do problema a fim de encontrar uma solução, visando uma maior eficiência computacional.

Para resolver o problema desta dissertação, foi utilizado um algoritmo com uma abordagem heurística denominado AStar ou A*. De forma sucinta, o objetivo deste algoritmo é encontrar uma rota entre dois pontos especificados. A escolha deste algoritmo, explicado a seguir, justifica-se pela capacidade de resolver o problema de forma rápida e sem custos computacionais elevados (RABIN, 2002).

A Figura 43 demonstra a utilização do algoritmo AStar aplicado ao objetivo deste trabalho. O primeiro passo é escolher dois pontos na imagem. Em seguida, são identificados os quatro pontos extremos da imagem. Calcula-se a distância entre todos esses pontos e aqueles que apresentarem a maior distância são eleitos. No exemplo abaixo, os pontos escolhidos foram o ponto mais alto à esquerda e o ponto mais baixo à direita.

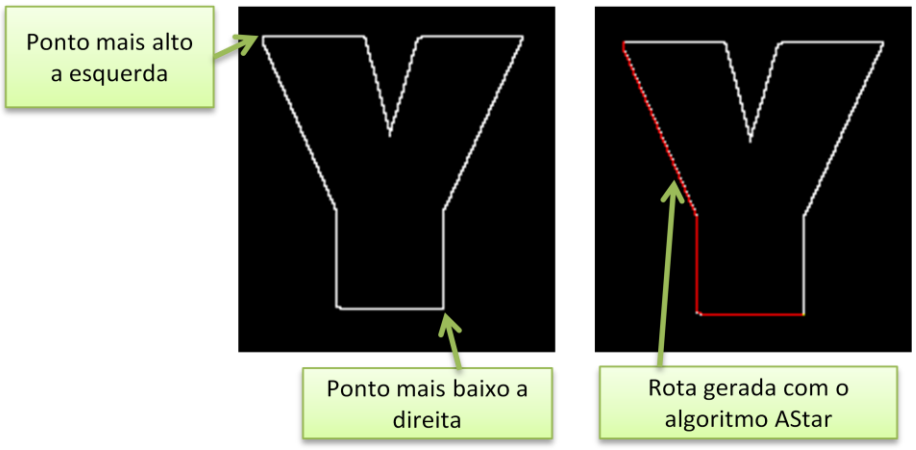


Figura 43 - Aplicação do algoritmo AStar

Após a seleção entre o ponto de origem e destino, o algoritmo AStar é executado. O resultado da execução do algoritmo é a rota que está marcada em vermelho. Em seguida, aplica-se novamente o algoritmo, invertendo os pontos de origem e destino para que este indique a rota

restante. Este processo é executado repetidas vezes até que não sejam mais encontrados pontos na cor branca na imagem. A seguir, será demonstrado o funcionamento do algoritmo AStar.

Algoritmo AStar (A*)

O objetivo do algoritmo A* é encontrar um caminho entre dois pontos em um mapa. Um ponto forte deste algoritmo é que ao invés deste explorar cegamente o seu ponto de destino, busca a melhor direção a ser explorada, algumas vezes tentando rotas alternativas. O A* se baseia em alguns conceitos que estão descritos a seguir:

- **Mapa ou grafo:** é o espaço utilizado para o algoritmo encontrar a o caminho entre os dois pontos; neste trabalho o mapa é constituído apenas dos *pixels* na cor branca;
- **Nós:** estruturas que representam posições no mapa; cada *pixel* branco é considerado um nó;
- **Distância ou heurística:** utilizado para definir a distância entre o nó explorado e o objetivo final. O termo distância pode ser utilizado em casos de busca em mapas bidimensionais;
- **Custo:** O custo é utilizado em aplicações para diferenciar as opções de rotas. Por exemplo, os nós em marrom representam terra firme e os nós em azul representam o mar. Custos diferentes podem ser atribuídos para cada uma destas opções de rota. O algoritmo vai optar sempre pela rota que implique em menor custo.

A fórmula utilizada para determinar a melhor opção entre as rotas disponíveis é: f = g + h, em que g representa o custo e h representa a heurística. A proposta das variáveis f, g, h é determinar o quão promissora é a rota. Quanto menor o valor de f, melhor será a rota.

Outra característica do A* é manter duas listas (Aberta e Fechada). A lista de abertos contém os nós que não foram explorados ainda, enquanto a lista de fechados contém os nós que já foram avaliados.

1. Escolher P como Ponto inicial;
2. Atribuir os valores f, g e h para P;
3. Adicionar P à lista de abertos. Neste momento ele é o único nó na lista de Abertos;
4. Escolher B (nó de menor valor de f na lista de abertos);
    a. Se B é o nó de destino, finalizar. A rota foi definida;
    b. Se a lista de abertos está vazia então finalize. Não existe rota possível;

5. Escolher C como um nó valido conectado a B;
    a. Atribuir os valores f, g e h para C;
    b. Checar se C está na lista de abertos ou fechados;
        i. Se estiver, checar qual rota é mais eficiente (menor valor de f)
            1. em caso afirmativo, atualizar a rota;
        ii. caso contrário, adicionar C para a lista de abertos;
    c. Repitir o passo 5 para todos os filhos válidos de B
6. Repitir a partir do passo 4;

Figura 44 - Algoritmo AStar em pseudo código, adaptado de (RABIN, 2002).

Após gerar as bordas da imagem, o próximo passo é planejar a rota de deposição do interior da camada. A proposta é que o sistema disponibilize duas opções de preenchimento: zigzag e espiral. O algoritmo zigzag foi escolhido pelo fato de estar presente em todos os softwares CAM analisados. A decisão de desenvolver o algoritmo em espiral se deu pelo fato deste estar disponível apenas em sistemas comerciais, constituindo um diferencial para este trabalho. A seguir, essas propostas serão detalhadas.

3.4 Preenchimento ZigZag

A abordagem tradicional para o algoritmo zigzag indica que deve haver uma variação fixa de 45 graus entre a rota de deposição atual e a próxima camada (HUANG, 2009). O valor em graus pode ser alterado de acordo com a necessidade do usuário.

Este trabalho implementa o modelo proposto por Ramaswani (1997). Neste caso, a direção da rota de deposição é determinada pela maior aresta do fecho retangular da camada, geralmente resultando em um número menor de retrações na rota definida. A Figura 45 ilustra um comparativo entre os resultados que seriam obtidos segundo cada modelo.

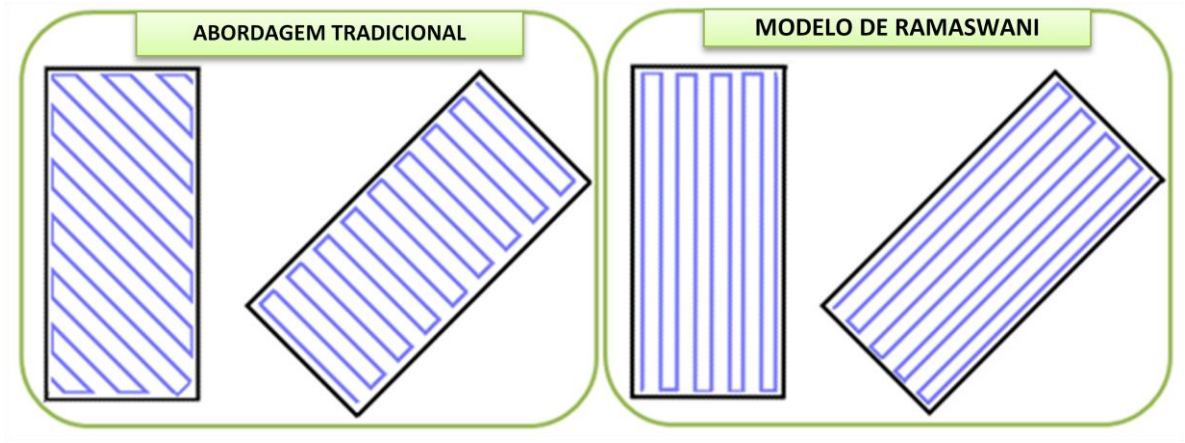


Figura 45 - Comparativo entre as opções de algoritmo em espiral.

Antes de aplicar esta proposta, é necessária uma preparação da camada a ser prototipada. A seguir, serão descritos de forma mais ampla os passos do algoritmo em zigzag propostos por esta dissertação:

1. As bordas da imagem são obtidas utilizando o modelo de Costa e Cesar (2001);
2. É gerada a rota para cada uma destas bordas utilizando a adaptação aqui descrita do algoritmo AStar;
3. A imagem original é segregada em seus vários pedaços utilizando o algoritmo de segmentação com base em Marchand-Maillet e Sharaiha (2000);
4. Para cada um dos segmentos, é aplicado o algoritmo de ZigZag proposto por Ramaswani (1997);
5. As diversas rotas geradas são ordenadas a fim de ganhar tempo durante o processo de prototipagem.

A partir desta visão do algoritmo, é necessário detalhar o passo 4. Este passo é o momento em que a proposta de Ramaswani (1997) é utilizada. A seguir, serão detalhados os passos aplicados a cada um dos segmentos da camada:

1. Obter o fecho convexo da imagem;
2. Calcular o retângulo envolvente de menor área que contenha todos os pontos do fecho convexo;
3. A partir do retângulo obtido, devem ser geradas linhas paralelas à maior aresta do retângulo; estas linhas são geradas com a condição de interceptar a imagem original;
4. O ponto final de cada segmento de linha deve ser interligado ao ponto inicial do próximo segmento.

Com isto, a rota em ZigZag será gerada. A Figura 46 ilustra tais passos.

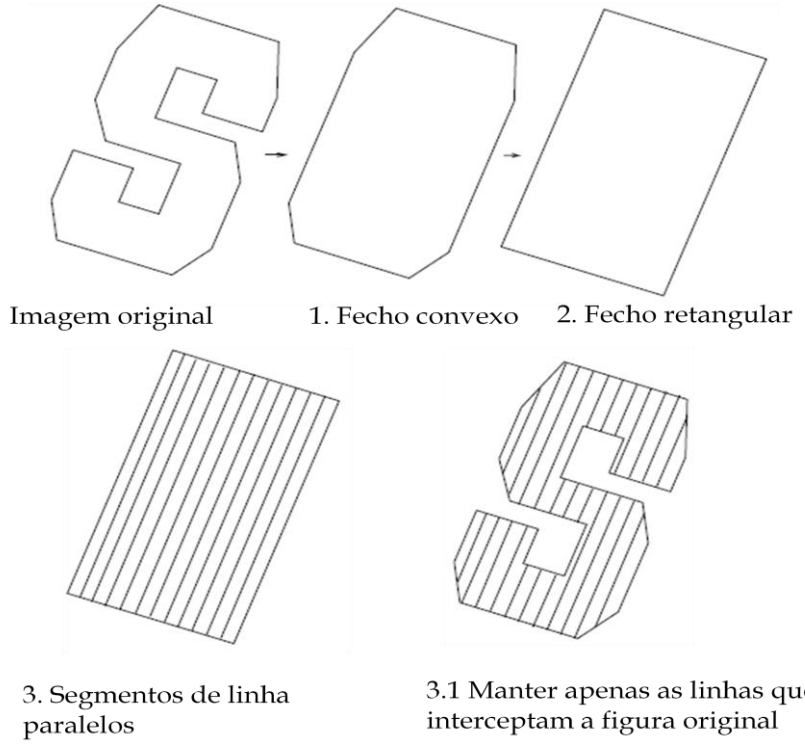


Figura 46 - Passos para obter a rota em zigzag. Tradução de (RAMASWANI, 1997)

Depois da execução dos passos descritos, o algoritmo zigzag retorna as rotas que representam o preenchimento da camada. Neste instante, foi acrescentada uma funcionalidade ao modelo a fim de ganhar tempo de prototipagem. Após interligar o ponto final e inicial das linhas anterior e subsequente, são verificadas todas as ligações possíveis entre linhas que não estão interligadas. O ponto positivo desta modificação é que será possível aumentar a continuidade de algumas rotas da camada. Em compensação, nesses mesmos pontos pode haver o acúmulo de material, provocado pelo fato de uma destas ligações passar muito próxima a uma rota pré-existente. A Figura 47 resume o fluxo de execução do algoritmo que foi definido para esta dissertação.

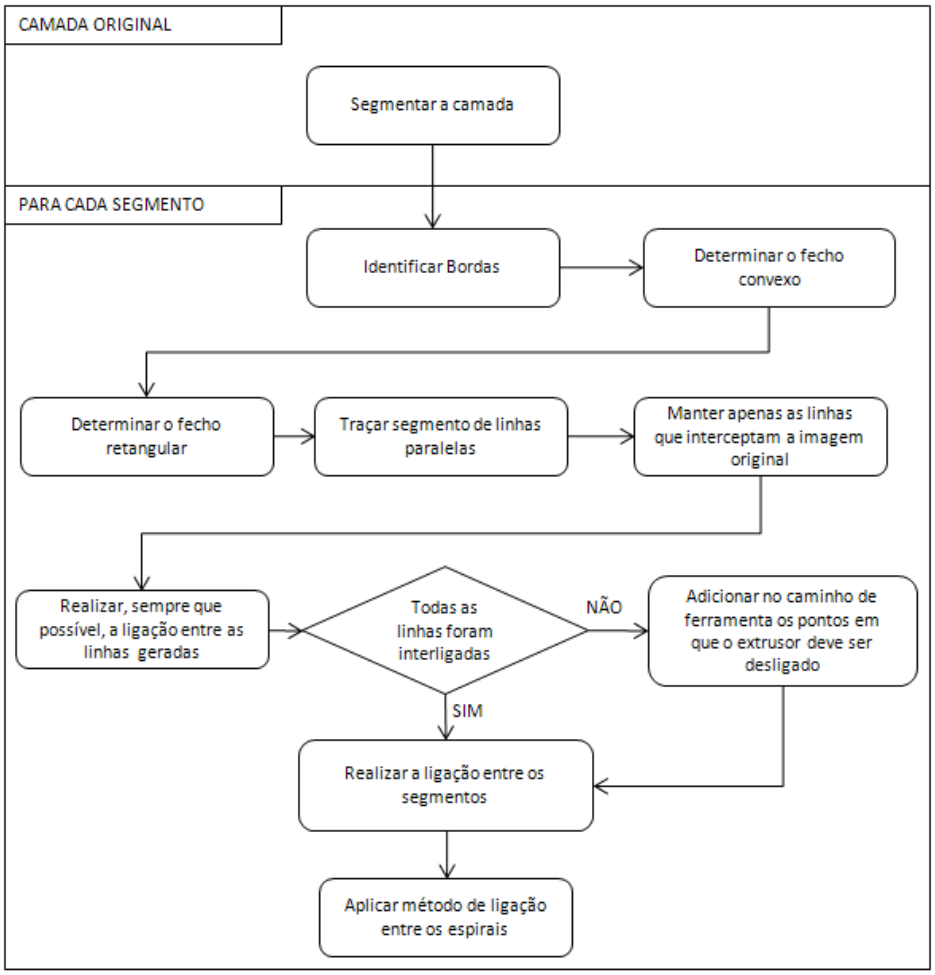


Figura 47 - Fluxo de execução do algoritmo em zigzag.

Para flexibilizar a utilização do algoritmo, este foi implementado recebendo dois parâmetros de entrada. Ambos os parâmetros são numéricos e definem distâncias que serão medidas em *pixels*. O primeiro definirá a distância entre os segmentos de linha paralelos. O segundo parâmetro definirá a distância mínima que as linhas de preenchimento podem ter em relação às bordas da fatia.

A Figura 48 demonstra o resultado do algoritmo com duas variações diferentes dos parâmetros citados. As imagens foram obtidas através do CNC Simulator (CNC SIMULATOR, 2009). Este software foi utilizado para interpretar as rotas geradas pelo sistema desenvolvido nesta dissertação.

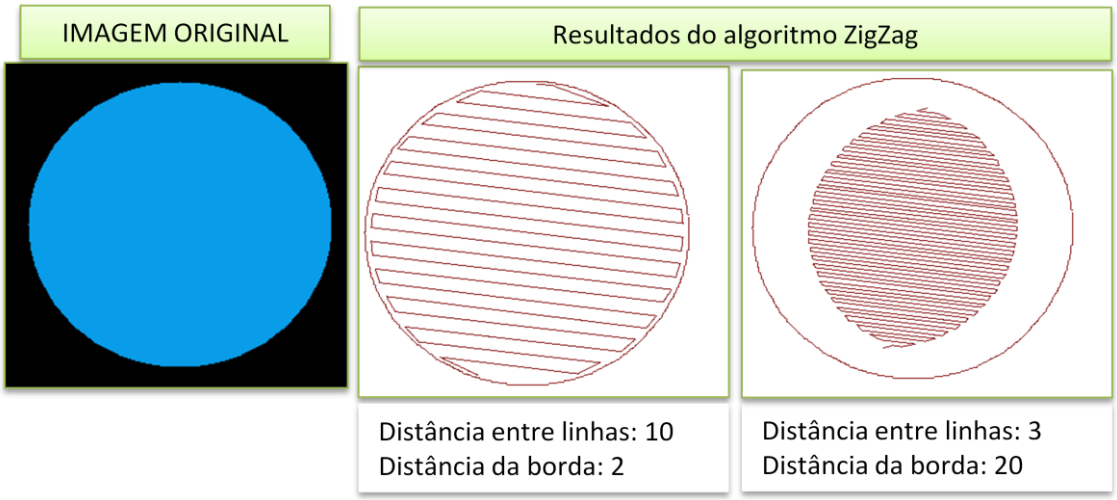


Figura 48 - Resultados obtidos com parâmetros diferentes.

3.5 Espiral

Segundo KAO (1999), o algoritmo espiral é bastante utilizado em máquinas CNC. Entretanto, o mesmo algoritmo não pode ser aplicado diretamente na deposição de material. Os maiores desafios estão no acúmulo de material em regiões específicas, preenchimentos incompletos e caminhos gerados de forma desconexa. Este tipo de algoritmo ainda é um desafio em andamento para as máquinas de prototipagem rápida e ainda não há uma solução ideal para este problema.

Este trabalho apresenta uma proposta para o preenchimento em espiral baseado em *offsets* recursivos. Na primeira etapa do processo, a fatia será segmentada em regiões e o algoritmo será aplicado para cada uma delas.

Para realizar as iterações do espiral, será utilizado o algoritmo de detecção de bordas já demonstrado. Aplicando-se este algoritmo N vezes, é possível obter uma figura proporcionalmente menor em relação à imagem original. Com isto, serão obtidas N espirais da imagem original. Este processo é executado até que se chegue ao centro da figura.

A Figura 49 ilustra as regiões geradas a partir do algoritmo de segmentação e os diversos espirais gerados através do algoritmo de detecção de bordas. As rotas em amarelo representam o que deve ser preenchido; as marcações na cor cinza representam o *offset*. O *offset* é uma distância fixa entre uma rota espiral e a próxima. É importante que este valor seja configurável para que possam ser realizados testes com espaçamentos diferentes.

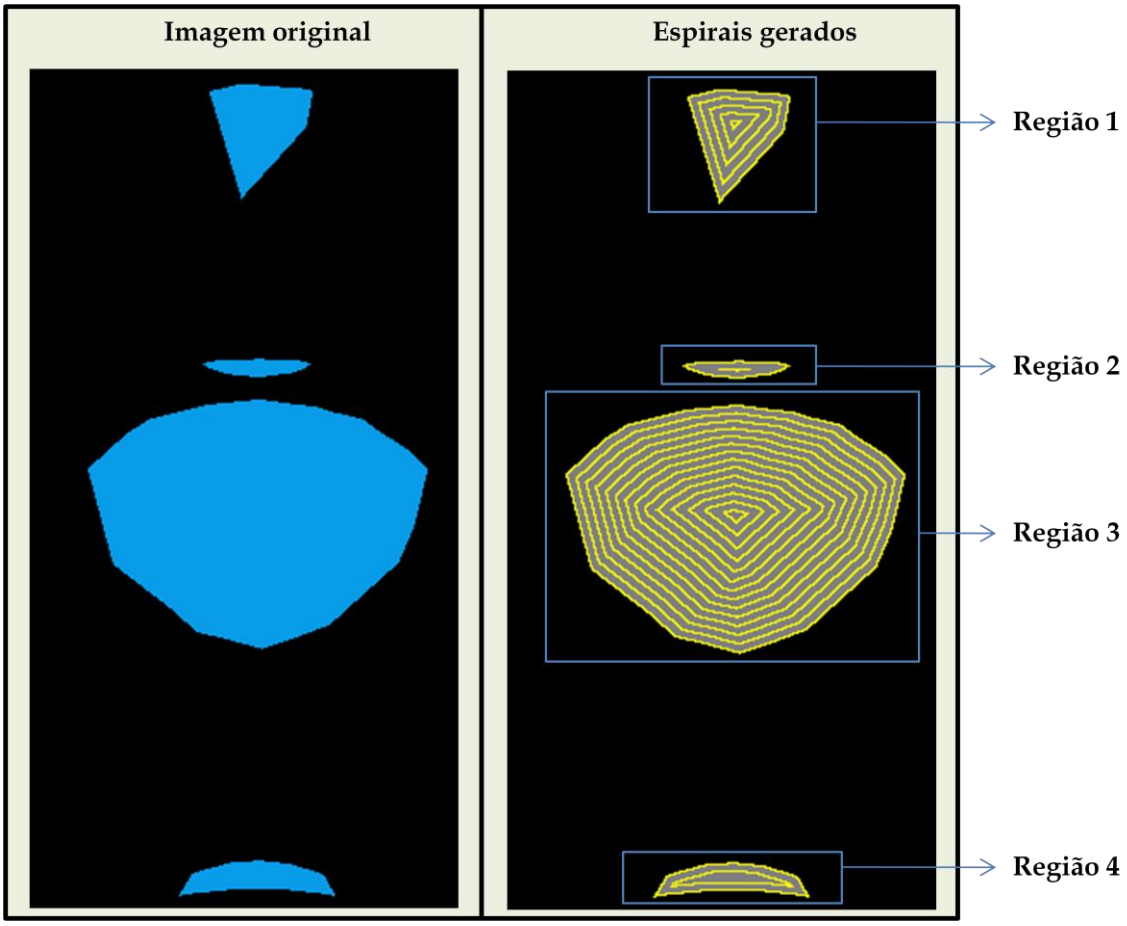


Figura 49 - Espirais gerados a partir da imagem original.

Depois que estas duas tarefas são completadas, é necessário utilizar algum método para realizar a ligação entre essas rotas. A proposta deste trabalho utiliza uma adaptação do método proposto por Park e Chung (2001), em que as rotas são identificadas e organizadas em uma estrutura de árvore. A partir desta estrutura, são feitas as ligações entre as rotas. Este método foi escolhido por suportar a ligação de imagens que contenham ilhas. Desta forma, o algoritmo contempla um universo muito maior de imagens possíveis.

A Figura 50 mostra uma camada hipotética a ser prototipada. Esta camada possui duas ilhas ao centro. A demonstração da utilização do método aqui adaptado de Park será feita com base neste exemplo.

Figura 50 - Imagem a ser processada

Ao aplicar o algoritmo de extração de bordas, seriam obtidos os três primeiros nós da árvore: E0-1 (borda externa à imagem), E0-2 e E0-3 (contornos identificados a partir das ilhas). A partir desta primeira iteração, novas aplicações do mesmo algoritmo irão gerar nós filhos. Estes nós serão colocados nos níveis subseqüentes até que não haja mais iterações possíveis para o algoritmo. O padrão para a nomenclatura dos nós segue o formato (EX–Y), em que Y é a identificação da folha e X é a quantidade de filhos; por exemplo: E1–2 é o primeiro filho da segunda ramificação da árvore.

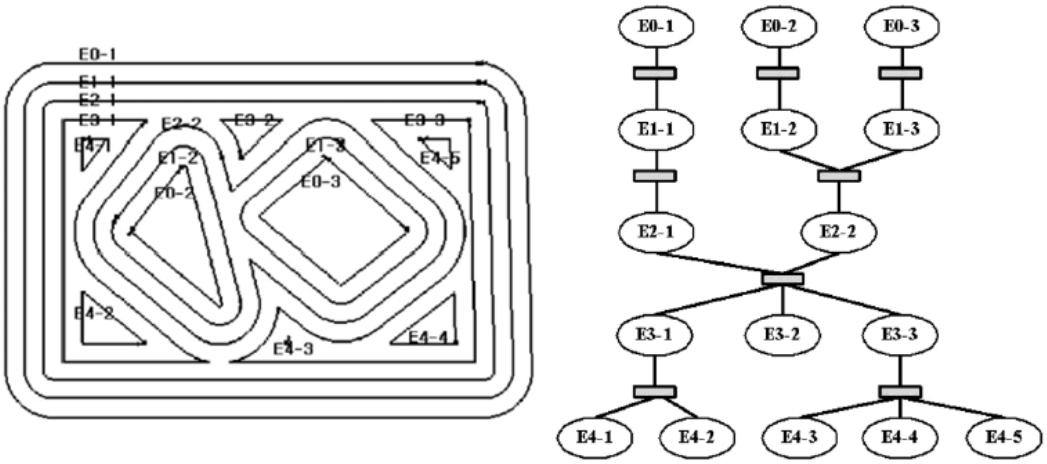


Figura 51 - Proposta para ligação entre as rotas (PARK e CHUNG, 2001)

Para determinar as ligações entre os nós da árvore, estes são avaliados a partir dos nós folha, ou seja, os nós que estão mais abaixo na hierarquia.

O método utilizado para determinar a correspondência pai/filho gera uma linha reta na vertical, a partir do ponto inicial do nó avaliado. A altura desta linha é correspondente ao *offset* determinado. Caso a linha gerada intercepte algum dos nós que estão um nível acima, é estabelecida uma ligação entre os dois nós. Por exemplo: considerando o nó E4-1 como o nó atual, os nós avaliados seriam os nós E3-1, E3-2 e E3-3.

Para ilustrar o que foi explicado anteriormente, caso na Figura 51 seja gerada uma linha vertical a partir do nó E4-1, esta linha interceptaria o nó E3-1. Neste caso, a ligação seria estabelecida. Fazendo o mesmo com o nó E3-1, a linha gerada interceptaria o nó E2-1. Este processo será repetido até chegar ao nó raiz, que é o nó E0-1.

Na Figura 52, é possível perceber as ligações obtidas após a execução do método proposto. À direita da imagem, estão com a mesma cor as rotas que estariam interligadas, com exceção das rotas na cor verde. Estas devem ser prototipadas de forma isolada por não haver ligação possível.

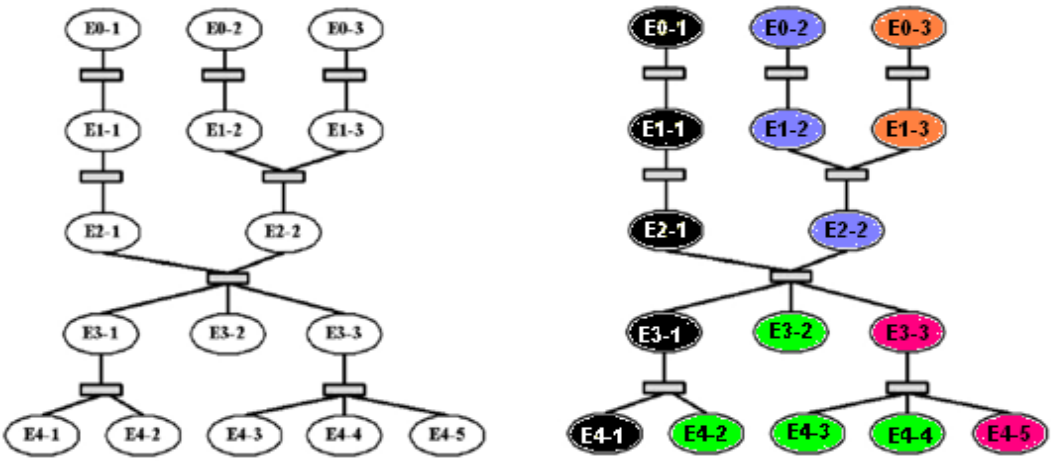


Figura 52 - Resultado após a aplicação do algoritmo.

A partir desta árvore e das ligações estabelecidas entre os nós, é possível definir a rota para prototipagem. A Figura 53 mostra o resultado da execução do algoritmo. No lado esquerdo da imagem, estão na mesma cor as rotas que foram associadas, enquanto do lado direito (em vermelho) as linhas demonstram como foram feitas as ligações.

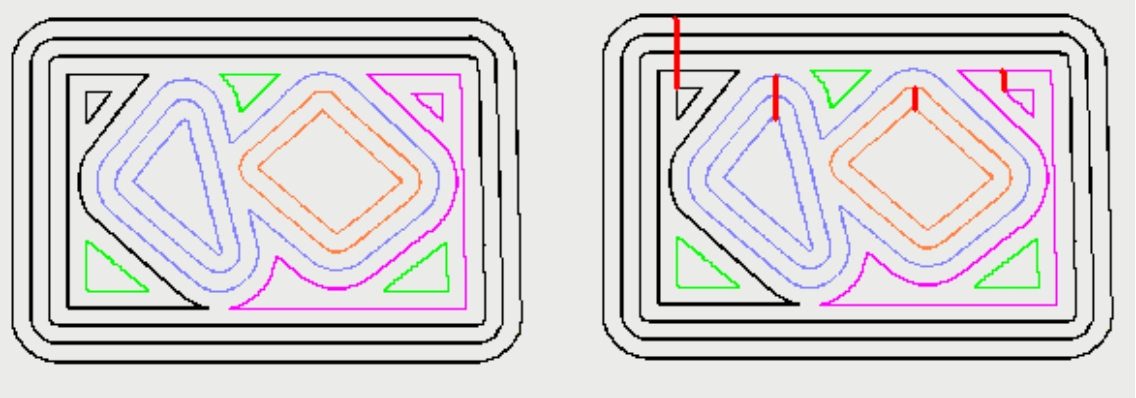


Figura 53 - Rota em espiral gerada para a camada.

Finalizando o planejamento para deposição

O resultado da execução dos processos descritos anteriormente é a geração de diversas rotas de deposição. Além de tais rotas, precisam ser definidos saltos entre as mesmas. Ou seja, no

momento em que uma rota de deposição chega ao final, o extrusor deve ser desligado e movido até o ponto inicial da próxima rota de deposição para que possa ser acionado novamente.

No algoritmo em ZigZag, a prioridade será dada para a deposição dos contornos. Para esta tarefa, será escolhido o contorno cujo ponto inicial estiver mais próximo do "ponto zero" da máquina. O próximo contorno escolhido é aquele que possuir o ponto inicial mais próximo do ponto final do contorno atual. Essa ação será feita até que não haja mais nenhum contorno a ser depositado.

Esta mesma forma de organizar a deposição dos contornos será utilizada para ordenar as rotas de preenchimento. Depois que esta ordenação for realizada, a fatia atual está pronta para ser prototipada. Uma visão geral deste fluxo está ilustrado na Figura 54.

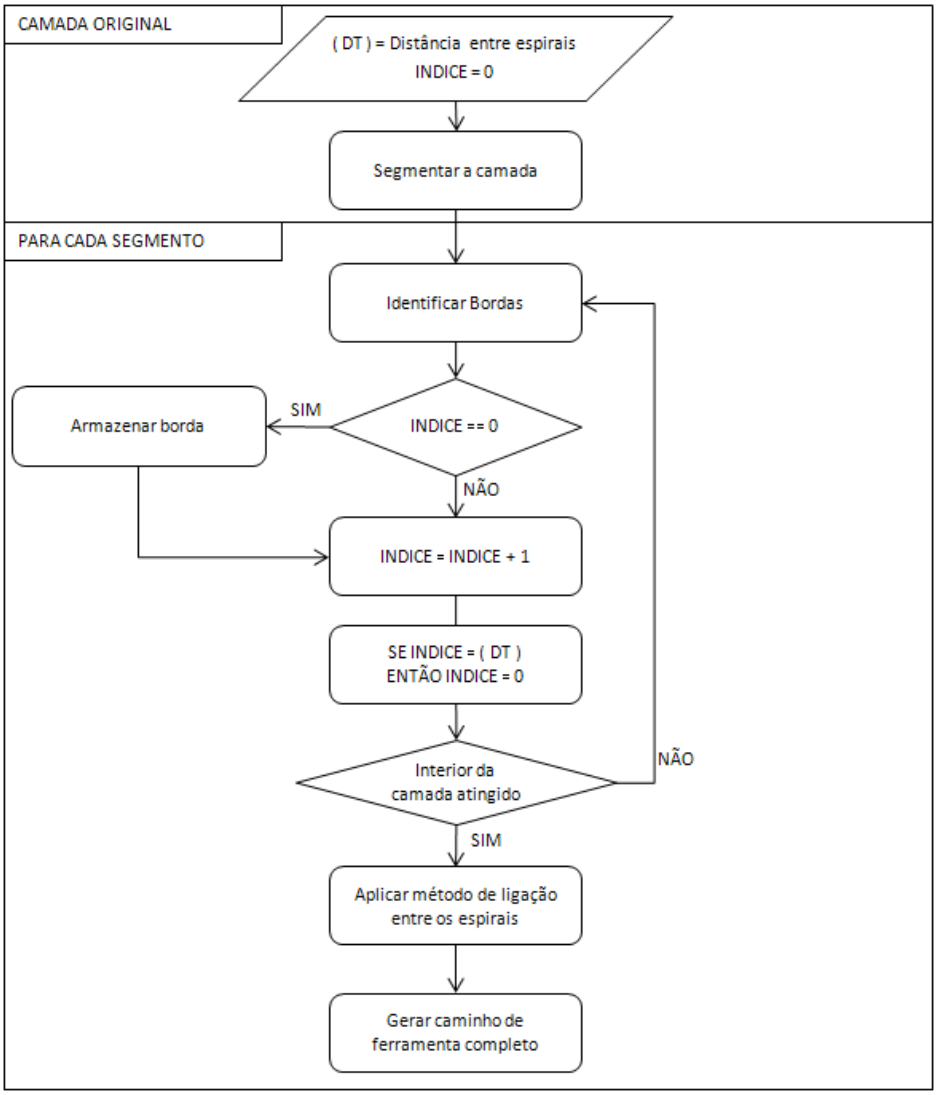


Figura 54 - Fluxo do algoritmo em espiral.

Duas opções serão disponibilizadas como saída deste processo. A primeira consiste em gerar um arquivo G para simulação no CNC Simulator (CNC Simulator, 2009). Esta opção tem o objetivo de validar as rotas que são geradas pelo software. Desta forma, diversos testes podem ser feitos de forma mais rápida, sem precisar do hardware da máquina de prototipagem.

Na segunda opção, será gerado um arquivo G seguindo o padrão estabelecido pelo EMC2 (*Enhanced Machine Controller*). Este é um sistema de código aberto, baseado em um *kernel* Linux de tempo real, que permite o controle de máquinas como fresadoras, tornos, robôs, máquinas de corte, etc. (EMC2, 2010). Desta forma, será possível realizar a prototipagem real dos modelos selecionados.

A integração do sistema desta dissertação com o EMC2 será detalhado no capítulo 5.

4 Software CAD para máquinas de prototipagem rápida FDM

O capítulo anterior mostrou soluções para as tarefas de orientação, fatiamento e suporte. Estas atividades foram analisadas de forma isolada. Este capítulo discute a integração dessas soluções e define a arquitetura do software para o sistema de prototipagem rápida com base em um hardware para o método FDM.

A linguagem de modelagem utilizada para descrever o sistema foi a UML (*Unified Modeling Language*). Esta escolha justifica-se por esta linguagem possuir grande aceitação na comunidade de desenvolvedores de sistemas, além de ser mantida pela OMG (*Object Management Group*), consórcio internacional de empresas que define os padrões na área de orientação a objetos (BEZERRA, 2002). Os seguintes diagramas foram elaborados visando descrever o sistema:

- **Diagrama de casos de uso:** visa demonstrar as possibilidades de interação do usuário com o sistema e a dependência entre os casos de uso levantado;
- **Diagrama de componentes:** este diagrama visa demonstrar o agrupamento lógico das classes. Nele, as classes estão agrupadas em módulos de software; cada módulo é responsável por atender uma determinada funcionalidade;
- **Diagrama de classes:** demonstrará a estrutura das diversas classes do sistema. Através deste diagrama é possível visualizar as dependências entre as classes do sistema;
- **Diagrama de sequência:** com este diagrama, é possível demonstrar a sequência lógica de processos específicos executados pelo sistema. Com base na troca de mensagens entre os objetos (instância de uma classe), demonstra-se o comportamento do sistema ao executar uma determinada atividade.

A Figura 55 mostra os seis casos de uso levantados para o sistema.

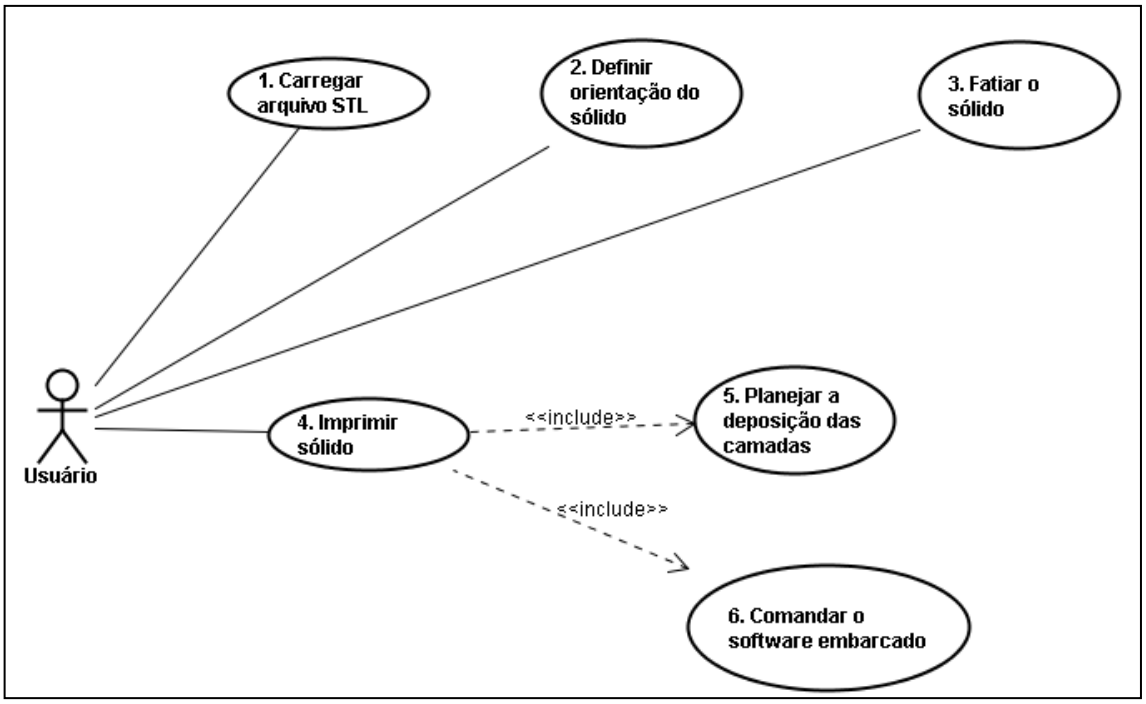


Figura 55 - Casos de uso do sistema.

Os casos de usos apresentam as interações do usuário com o sistema. A seguir, será realizado um detalhamento de cada caso de uso:

- **Carregar arquivo STL:** permite que o usuário selecione um arquivo tridimensional para ser trabalhado pelo sistema;
- **Definir a orientação do sólido:** possibilita que o usuário rotacione o arquivo tridimensional de acordo com a posição que achar mais conveniente para a prototipagem;
- **Fatiar o sólido:** inicia o processo de fatiamento do sólido;
- **Imprimir sólido:** esta função inicia o processo de interação com a máquina de prototipagem. Ao ser solicitado, este caso de uso realiza o planejamento das rotas e envia a sequência de comandos para o software embarcado na máquina de prototipagem.

O sistema foi projetado visando possibilitar ao usuário a realização destes casos de uso de forma simples. A interface homem-máquina possui apenas uma tela em que é possível o acionamento dos casos de uso citados anteriormente. A Figura 56 mostra a interface proposta para o sistema.

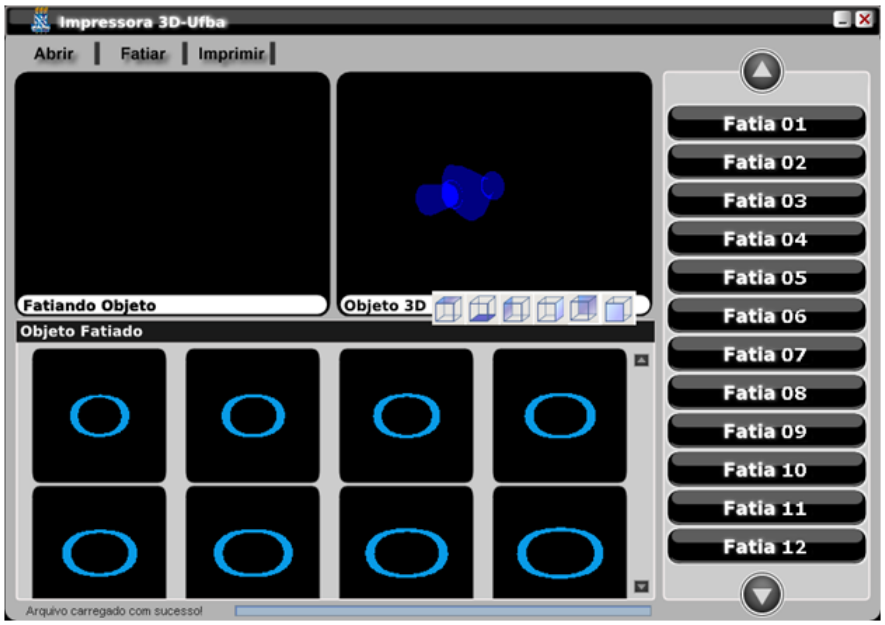


Figura 56 - tela do sistema.

A tela é dividida em quatro painéis básicos:

- **Fatiando Objeto:** no momento em que o usuário solicitar o fatiamento do arquivo tridimensional, este painel mostra em tempo real o processo de fatiamento;
- **Objeto 3D:** neste painel, é possível visualizar o objeto tridimensional que foi carregado pelo sistema. Além disso, possibilita que o usuário utilize o mouse para rotacionar este objeto;
- **Objeto Fatiado:** após a conclusão do processo de fatiamento, todas as fatias obtidas para o objeto ficarão visíveis neste local;
- **Painel na direita:** este painel mostrará diversos parâmetros de configuração da máquina de prototipagem. O usuário poderá ajustar os valores desejados para realizar novos testes.

O primeiro painel, denominado *Fatiando Objeto*, mostra em tempo real o processo de fatiamento, quando este está sendo executado. O painel, cujo título é *Objeto 3D*, mostra na tela o objeto tridimensional que foi carregado, possibilitando que o usuário utilize o *mouse* para rotacioná-lo.

Na Figura 57, é feita uma associação entre os casos de usos do sistema e a tela apresentada. Assim, é possível perceber qual caso de uso o usuário estará ativando ao clicar nos botões do sistema.

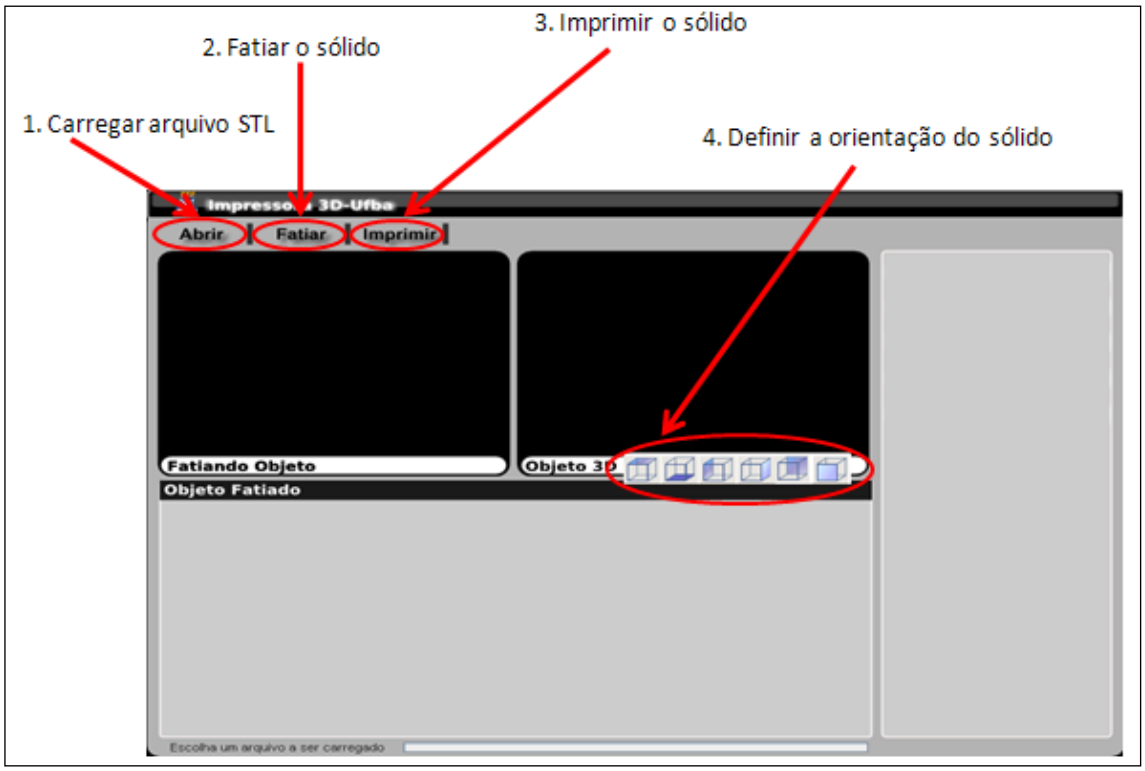


Figura 57 - associação entre os casos de uso e a interface.

Os módulos do sistema foram construídos visando atender às necessidades levantadas a partir do diagrama de casos de uso. A Figura 58 demonstra os componentes do sistema e em seguida será realizado um detalhamento da responsabilidade de cada um dos módulos.

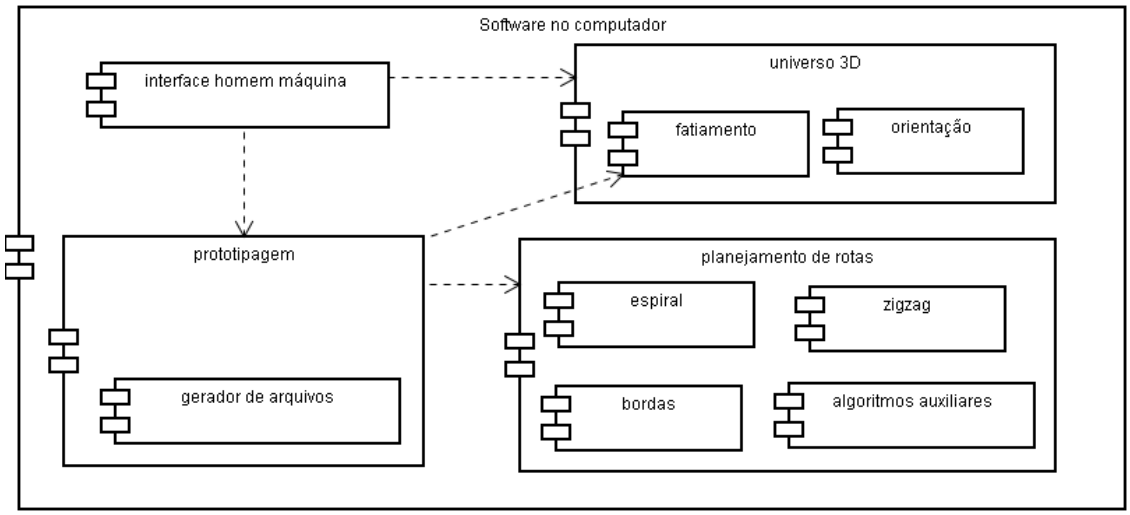


Figura 58 - Diagrama de componentes do sistema.

Quatro módulos principais foram especificados. A seguir, tem-se uma descrição das responsabilidades atribuídas a cada um destes:

- **Universo 3D:** responsável por realizar tarefas que envolvam o domínio tridimensional. As classes deste módulo são responsáveis por processos como:

carregar o arquivo STL a partir do computador, converter tal arquivo em um objeto tridimensional para o Java, criar o universo virtual em que este objeto será inserido, etc. Neste componente, há dois sub-módulos: o primeiro, denominado orientação, é responsável pelas transformações aplicadas ao objeto 3D, para que este fique de acordo com a opção do usuário. O segundo (fatiamento) é responsável por fazer a conversão do objeto tridimensional em uma lista de fatias bidimensionais através do processo de fatiamento;

- **Planejamento de rotas:** responsável por todas as tarefas associadas ao domínio bidimensional. Os algoritmos associados ao processamento das fatias estão todos agrupados neste módulo. Há três submódulos para os principais algoritmos implementados: detecção de bordas, preenchimento em zigzag e preenchimento em espiral. O módulo denominado “algoritmos auxiliares” agrupa funções de uso comum e que dão suporte à obtenção das rotas de preenchimento;
- **Prototipagem:** este terceiro módulo é responsável por realizar a integração entre as tarefas dos módulos anteriores. Ele recebe as requisições a partir da interface com o usuário e faz solicitações aos módulos responsáveis para que sejam realizadas tarefas como o fatiamento e o planejamento de cada camada. Ao final de sua atividade, este deve gerar um arquivo seguindo o padrão G para ser utilizado na próxima etapa da prototipagem;
- **Interface homem máquina:** contém os componentes de tela que permitem ao sistema interagir com o usuário.

Com base nesta visão geral, será feito um detalhamento maior em cada um dos módulos. Para viabilizar o entendimento da estrutura interna dos módulos, serão mostrados diagramas de classes. Ademais, serão utilizados diagramas de sequência visando à documentação da dinâmica dos processos considerados mais importantes para o funcionamento do sistema.

4.1 Universo 3D

A Figura 59 apresenta o diagrama das principais classes que compõem o universo tridimensional. Este módulo é responsável pelas seguintes funcionalidades: transformar um arquivo STL em um objeto tridimensional, aplicar transformações (rotação, translação, redimensionamento) ao objeto STL e gerar as diversas fatias do objeto.

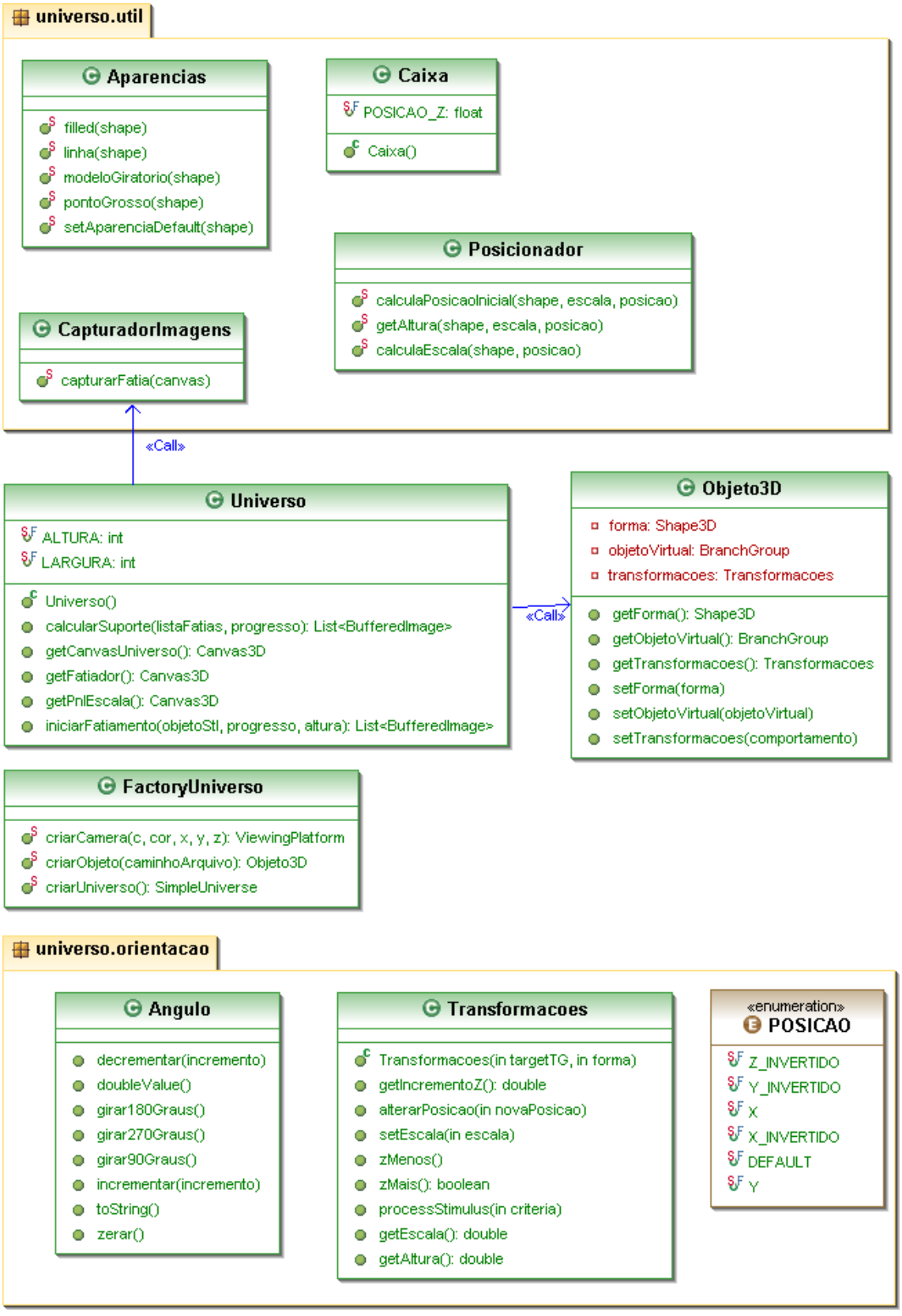


Figura 59 - Diagrama de classes do universo 3D.

A classe *Universo* é o ponto central deste módulo. Uma consideração importante a ser feita é a relação entre as classes *Universo* e *VirtualUniverse* (classe da API Java3D). A classe

VirtualUniverse provê uma infraestrutura genérica, através da qual é possível criar variados tipos de universos tridimensionais. Ao passo que a classe *Universo*, parte integrante deste trabalho, é uma especialização de *VirtualUniverse*. Em outras palavras, a classe Universo contém um *VirtualUniverse*, que foi criado apenas com o objetivo de atender às necessidades do sistema de prototipagem rápida. Existem outras classes que servem de suporte, encapsulando algumas tarefas. A classe *FactoryUniverso*, por exemplo, possui três métodos e é responsável pela criação dos principais componentes do universo. Os métodos são enumerados a seguir:

1. *criarUniverso()*: realiza a criação do universo 3D no estado inicial;
2. *criarCamera():* cria uma nova câmera que será posicionada na posição x, y e z especificada;
3. *criarObjeto():* recebe como entrada o caminho do arquivo no computador e o transforma em uma instância da classe *Objeto3D* para ser tratado pelo sistema.

A Figura 60 contém o processo de inicialização do universo. Como pode ser observado, inicialmente a classe *FactoryUniverso* recebe uma mensagem para criação do universo. Em seguida, uma instância da classe Caixa é criada e, posteriormente, duas câmeras são criadas. A primeira é responsável pelo fatiamento dos objetos, enquanto a segunda é criada com o objetivo de permitir a visualização destes objetos na tela.

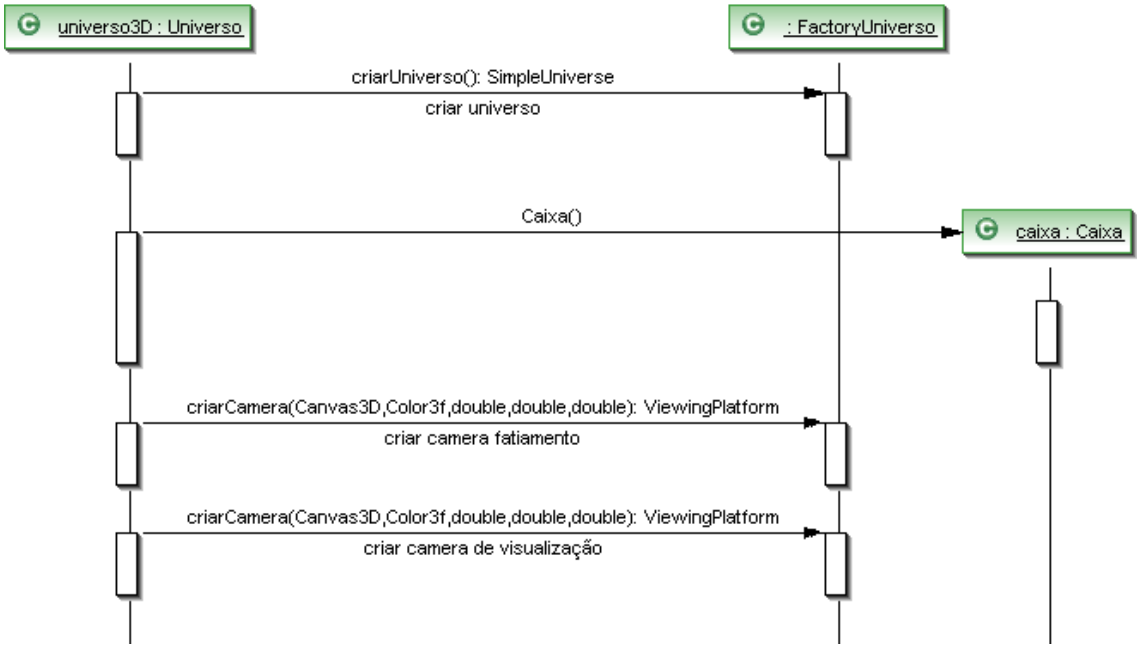


Figura 60 - Diagrama de sequência da criação do universo 3D.

O motivo da criação de duas câmeras é que a câmera de visualização dos objetos para a interface do usuário não precisa ter a mesma definição da câmera de visualização do

fatiamento. Desta forma, as propriedades do fatiamento podem ser facilmente modificadas sem alterar a visualização do usuário do sistema. Um exemplo de modificação que pode ser feita é o tamanho em *pixels* de cada imagem que a câmera do fatiamento irá gerar.

A necessidade da criação da classe *Caixa* se deu devido a testes efetuados com o software nos sistemas operacionais Windows Vista e XP, com placas de vídeo de fabricantes diferentes. No Windows Vista, mesmo com o volume de visão da câmera reduzido, a visualização das fatias do objeto era maior do que o planejado. Este fator atrapalharia as próximas etapas do processo de prototipagem. Com o objetivo de forçar o Windows Vista a manter a mesma profundidade de visualização das fatias que o Windows XP, foi criada uma caixa completamente preta, que fica posicionada exatamente no término do volume de visão da câmera responsável pelo fatiamento. Deste modo, esta inconsistência foi sanada e o Windows Vista passou a visualizar as fatias com a mesma altura do Windows XP.

5 *Hardware* e controle da máquina de prototipagem

Neste capítulo será abordada a integração do presente trabalho com os outros componentes envolvidos no projeto: *hardware* e software de controle da máquina de prototipagem. A descrição de tal modelo será feita em paralelo às especificações do *hardware*.

A Figura 61 mostra a integração do sistema CAM desenvolvido nesta dissertação com outros componentes do processo de prototipagem rápida. O código G, utilizado como padrão para saída do sistema, possibilitou a integração com outras aplicações existentes, a fim de validar o sistema desenvolvido.

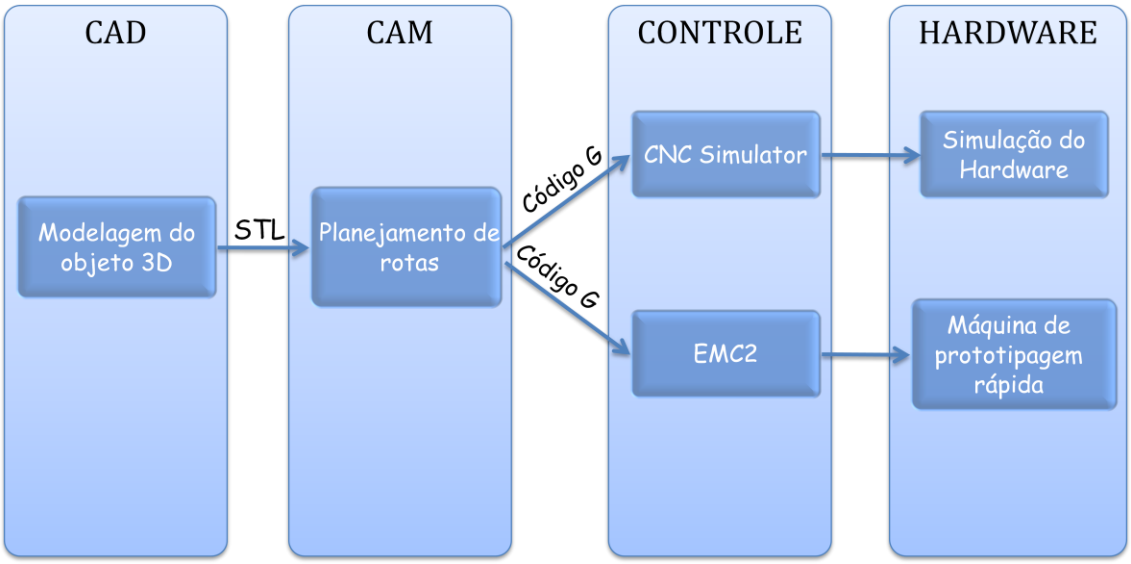


Figura 61 - Integração deste trabalho com outros sistemas.

Os primeiros testes feitos com o sistema CAM foram realizados em ambiente simulado, utilizando o CNC Simulator (CNC SIMULATOR, 2009). Este software permite a simulação do comportamento de máquinas CNC de acordo com o arquivo G utilizado como entrada. Estes testes foram úteis, pois facilitaram a visualização de erros durante o processo de desenvolvimento, permitindo ajustes em questões como: má formação de rotas, caminhos desconexos, erros de implementação dos algoritmos, dentre outras.

Depois que o sistema apresentou um comportamento estável no simulador, uma segunda etapa de testes foi feita utilizando uma máquina de manufatura aditiva, cujo desenvolvimento foi o foco da dissertação de Costa (2011).

O controle numérico dessa máquina foi feito por um sistema de tempo real baseado em Linux denominado EMC2 (EMC, 2010). A responsabilidade do EMC2 é, portanto, interpretar o arquivo G gerado pelo sistema CAM desta dissertação e controlar a máquina de prototipagem rápida.

A seguir, será feita uma breve descrição do protótipo do *hardware* utilizado. Os componentes do sistema mecatrônico serão detalhados segundo o modelo descrito por Isermann (1996). Os itens pertencentes a cada subconjunto serão listados (mecânica, eletrônica e tecnologia da informação). Em seguida, será demonstrado como o EMC2 atua como processo responsável pela integração entre estes mecanismos.

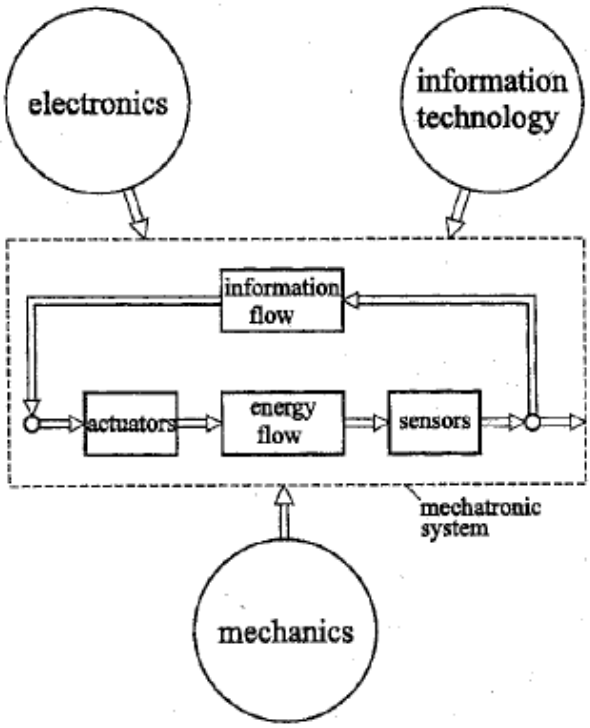


Figura 62 - Sistema mecatrônico, integração da tecnologia da informação, componentes mecânicos e eletrônicos (Isermann, 1996).

A máquina de manufatura aditiva é um robô cartesiano que possui três juntas prismáticas. Estas juntas são responsáveis pelo deslocamento de um atuador (extrusor) em uma área de trabalho tridimensional. Os componentes mecânicos estão distribuídos em quatro categorias principais:

Extrusor: sua responsabilidade é realizar a deposição do material. O extrusor é alimentado por grãos de termoplásticos e a deposição ocorre devido à rotação de uma rosca movida por um motor de corrente contínua. O giro da rosca promove o transporte do material da base até o bico do extrusor. O calor gerado pela rotação da rosca não é suficiente para derreter o material, por isso são adicionadas resistências elétricas responsáveis por complementar o aquecimento do sistema até os níveis requeridos para o tipo de material que estiver sendo processado;

Eixos X e Y: configuram juntas prismáticas que permitem avanços lineares da máquina. Há dois sinais que definem a direção e a intensidade da movimentação. O sinal de direção define em qual sentido será a rotação do motor e o PWM (*Pulse Width Modulation*) define a intensidade de rotação do motor. A fim de gerar a movimentação em malha fechada, foi utilizado um sensor incremental de fita (*Strip Encoder*) similar ao utilizado em impressoras jato de tinta. A informação de posição obtida deste *encoder* é utilizada como base para o controlador de posição do tipo PID (Proporcional Integral Derivativo).

Há um acoplamento físico entre os eixos X, Y e o extrusor. Qualquer movimento realizado no eixo X provoca a mudança de posição do extrusor. Ao passo que uma movimentação no eixo Y provoca o deslocamento do eixo X e do extrusor.

Eixo Z: foi concebido de forma independente. A movimentação deste não interfere nos eixos X e Y. Uma plataforma horizontal foi montada sobre este eixo. Esta plataforma possui sua posição inicial no alto da estrutura e vai sendo deslocada para baixo na medida em que são adicionadas novas camadas à peça em produção.

Sensores e atuadores

Tabela 2 - Sensores e atuadores da máquina de prototipagem.

	Eixos X e Y	Eixo Z	Extrusor
Sensores	*Um encoder* para medir o deslocamento do eixo; Sensor de fim de curso para definir a posição inicial do eixo;	Sensor de fim de curso para definir a posição inicial do eixo;	Dois sensores de temperatura; Um *encoder* para medir o deslocamento do motor;
Atuadores	Motor de corrente contínua;	Motor de passo;	Motor de corrente contínua; Duas resistências elétricas.

O projeto inicial para desenvolvimento do *software* embarcado pretendia utilizar o microcontrolador Arduino para gerenciar a máquina de prototipagem rápida (Arduino, 2010). Porém, os resultados apresentados nos testes com esta plataforma demonstraram que era necessário um maior poder de processamento, pois o Arduino não conseguia realizar as diversas tarefas de controle com uma taxa de repetição aceitável. Com base nestes resultados,

outras possibilidades foram analisadas e a solução que se mostrou mais promissora para as tarefas de controle da máquina foi a utilização do EMC2 (EMC, 2010).

5.1 EMC2 – Enhanced Machine Controller

O EMC2 é um software que foi desenvolvido com o propósito de controlar diversos tipos de máquinas, como fresadoras, tornos, robôs, etc. Este roda em plataforma Linux, utilizando extensões para tempo real. Neste sistema, há quatro módulos principais: controlador de movimentação, controlador discreto de E/S, executor de tarefas e interface gráfica com o usuário. Há também uma camada chamada HAL (*Hardware Abstraction Layer*) que permite a integração de diversos sensores e atuadores com o software, sem a necessidade de recompilação do software ou até mesmo do *kernel* do Linux.

O EMC2 foi utilizado como integrador entre o sistema proposto neste trabalho e o *hardware* desenvolvido para a máquina de prototipagem. Tal integração é possível por meio de código G. O sistema planeja todos os comandos necessários para prototipar o objeto e os converte em uma série de comandos G. Estes comandos devem ser salvos em um arquivo com a extensão NGC. Este arquivo é utilizado como entrada para o EMC2.

A Figura 63 mostra a tela inicial do *software*. Estão destacadas as ações para carregar um arquivo NGC. A primeira ação é clicar no *menu* sobre a opção *File* e em seguida no subitem *Open*. Esta opção fica no canto superior esquerdo da tela e está destacado de vermelho. Em seguida, o usuário deve escolher o arquivo NGC e clicar sobre a opção *Open.*

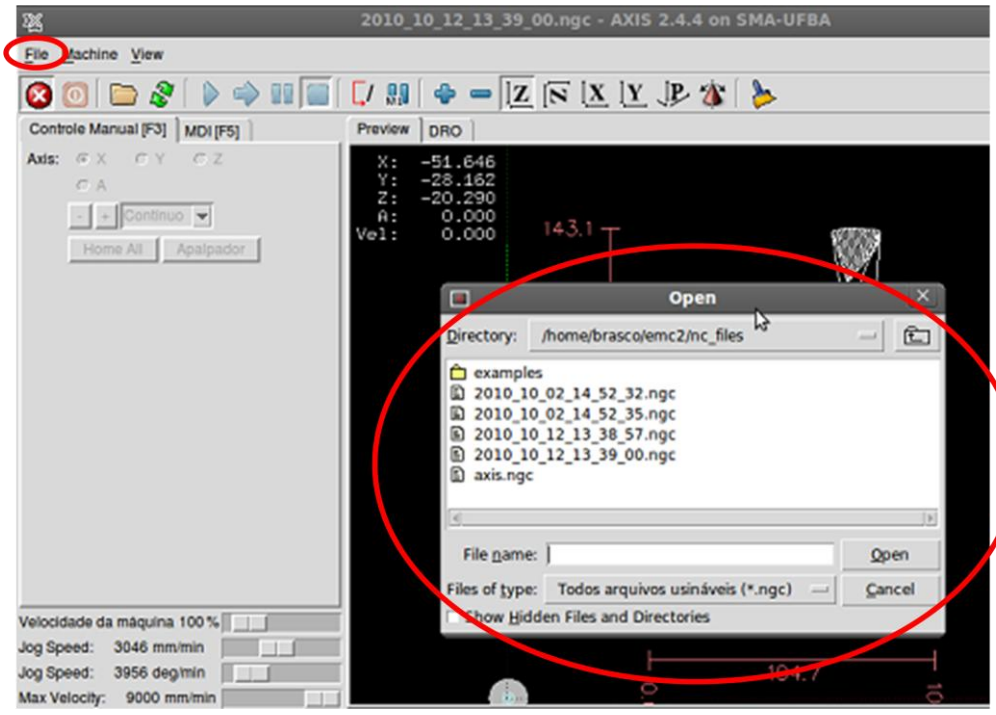


Figura 63 - Carregando um arquivo NGC no EMC2.

Com o arquivo carregado, é possível visualizar a trajetória que será percorrida pelo extrusor da máquina de prototipagem, além da posição física do extrusor na máquina de prototipagem. O painel com fundo preto ao centro da tela fornece essas informações. A Figura 64 mostra o estado do painel após carregar um arquivo gerado pelo sistema.

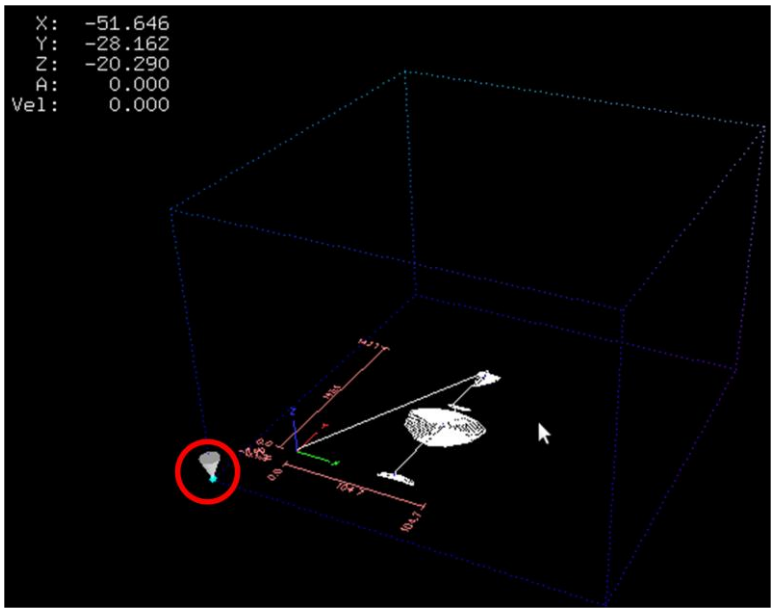


Figura 64 - Painel principal do EMC2 com um arquivo carregado.

No momento em que a máquina está funcionando e movendo o extrusor, o cone que está circulado em vermelho também se move na tela, a fim de permitir a visualização da movimentação através do software. É possível alternar entre diversas visões e até mesmo aplicar *zoom* às trajetórias. Estas funções estão circuladas em vermelho na Figura 65. As trajetórias descritas pelo arquivo NGC que ainda não foram percorridas ficam na cor branca. Depois que a máquina é acionada e os comandos vão sendo atendidos, a cor vai mudando para magenta.

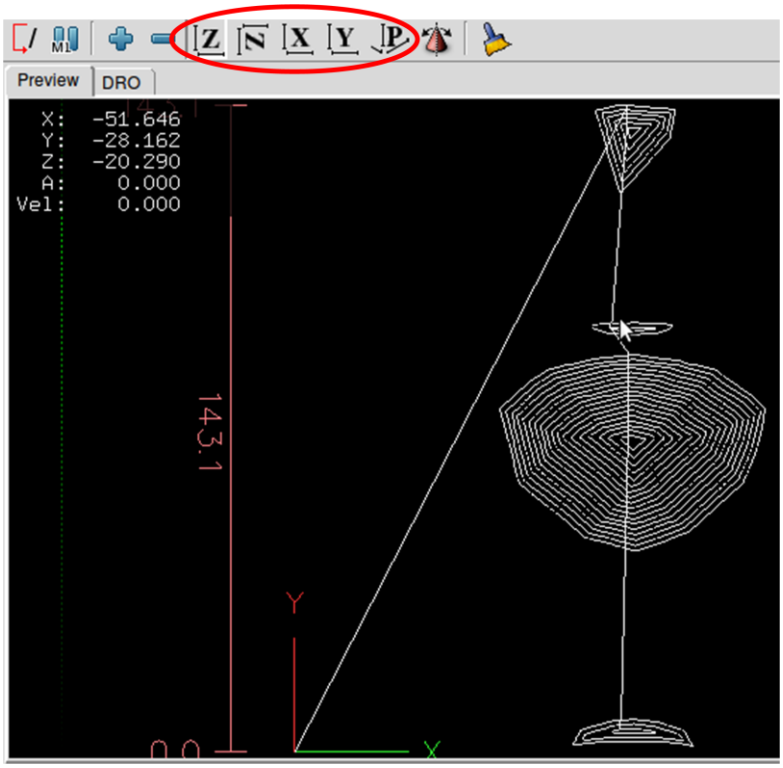


Figura 65 - Modificação na visão da trajetória.

Antes de iniciar a prototipagem, é necessário realizar a configuração dos eixos e enviá-los à "posição 0" da máquina. Na Figura 66, estão destacados em vermelho os botões que atendem esta funcionalidade. O primeiro botão é utilizado para parada de emergência. Quando acionado, a máquina é interrompida , desligando os motores e seus respectivos controladores. O botão em laranja ao lado permite ligar/desligar os controladores do motor. Para que o software funcione, é necessário que o botão esteja ativado.

O botão "*Home All*", segunda marcação em vermelho, faz com que a máquina coloque todos os seus eixos no ponto inicial. Por segurança, o EMC2 sempre inicia no modo "parada de emergência". Para iniciar o uso da máquina, três passos devem ser executados: desligar a parada de emergência, ligar o controlador e enviar todos os eixos para a posição inicial (*home*).

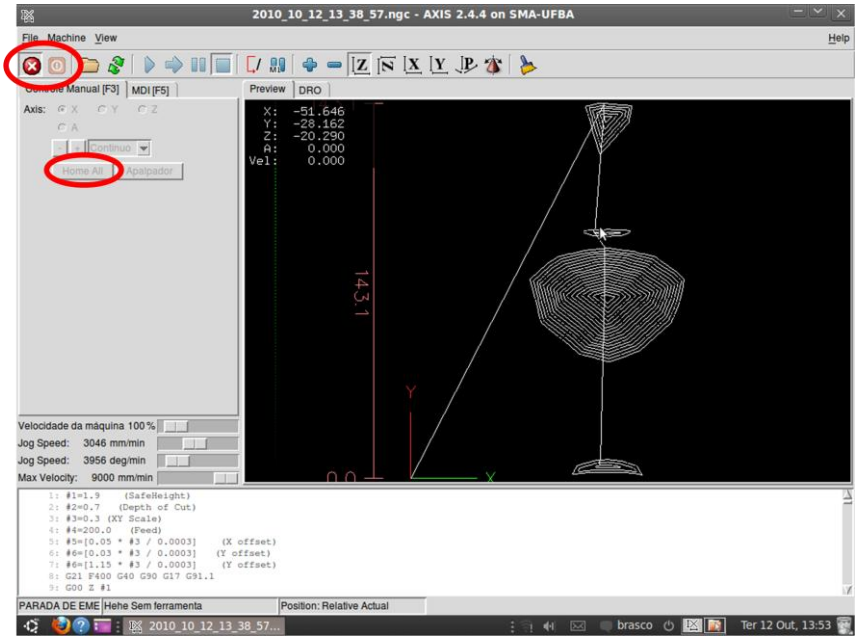


Figura 66 - Calibragem da máquina no EMC2.

Uma outra configuração importante a ser feita são os ganhos do controlador PID para cada eixo e os parâmetros de processo. Esta configuração pode ser feita uma única vez e em seguida salva em arquivo para as próximas utilizações. A Figura 67 mostra a tela em que os ganhos podem ser digitados pelo usuário. Para chegar a esta tela basta clicar na opção do *menu Machine* e, posteriormente, *Calibration.*

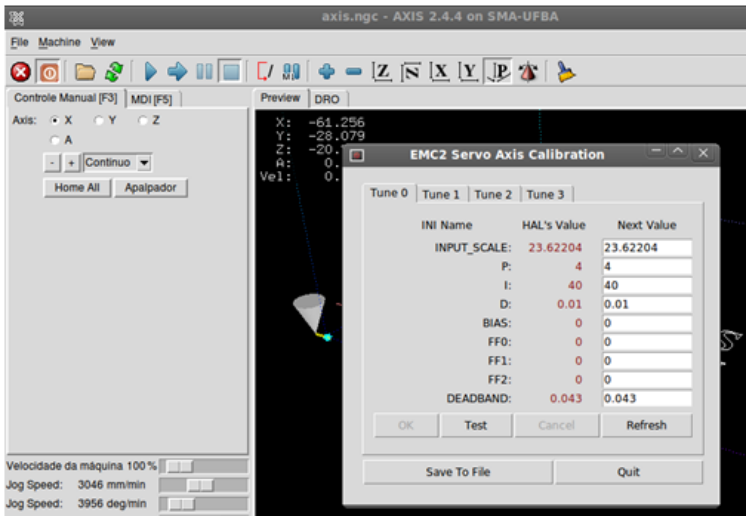


Figura 67 - Tela de configuração dos ganhos PID.

Caso os passos anteriores tenham sido realizados corretamente, o EMC2 pode iniciar seu funcionamento. Para tanto, basta carregar o arquivo NGC e solicitar o início da prototipagem clicando no botão *play* (quinto botão da esquerda para a direita). Seguindo a série de comandos do arquivo NGC, o EMC2 controla os eixos X, Y e Z e o extrusor, a fim de realizar a prototipagem do objeto.

6 Testes e validações

Neste capítulo, serão apresentados os métodos utilizados para testes e validação do sistema e dos algoritmos desenvolvidos. Os testes aplicados ao sistema podem ser divididos em quatro categorias:

1. **Fatiamento (software pc):** testes realizados para avaliar o resultado do método de fatiamento baseado em *bitmaps* proposto por este trabalho;
2. **Simulação de trajetórias (software pc):** visam validar principalmente os algoritmos de geração de trajetórias. Tais testes foram realizados através de simulações;
3. **Interpretação e capacidade de atender aos comandos (software embarcado):** o objetivo destes testes é validar o software embarcado atuando na máquina de prototipagem rápida. Comandos individuais passados para a máquina de prototipagem devem ser atendidos;
4. **Simulação de camadas na máquina de prototipagem:** estas simulações foram realizadas com uma caneta posicionada no lugar do extrusor da máquina de prototipagem. O objetivo destes testes era avaliar a capacidade da máquina de movimentar seus eixos (X, Y e Z), de forma a atender toda a trajetória determinada.

6.1 Fatiamento

Para avaliar o fatiamento, foi criada uma tela temporária no sistema para exibir todas as fatias obtidas após o processo. Para fins de demonstração, as camadas tiveram suas dimensões reduzidas. Isto implica em um menor nível de detalhes em relação à camada que é realmente tratada pelo sistema. O sistema trabalha com camadas de 500 *pixels* de largura e altura, sendo que este parâmetro é configurável. Quanto maior a quantidade de *pixels* das fatias, maior será o tempo de processamento; em compensação, elevará também a quantidade de detalhes da camada.

A Figura 68 mostra as 16 camadas obtidas após o fatiamento de uma garrafa. Esta foi posicionada na posição horizontal, “deitada”.

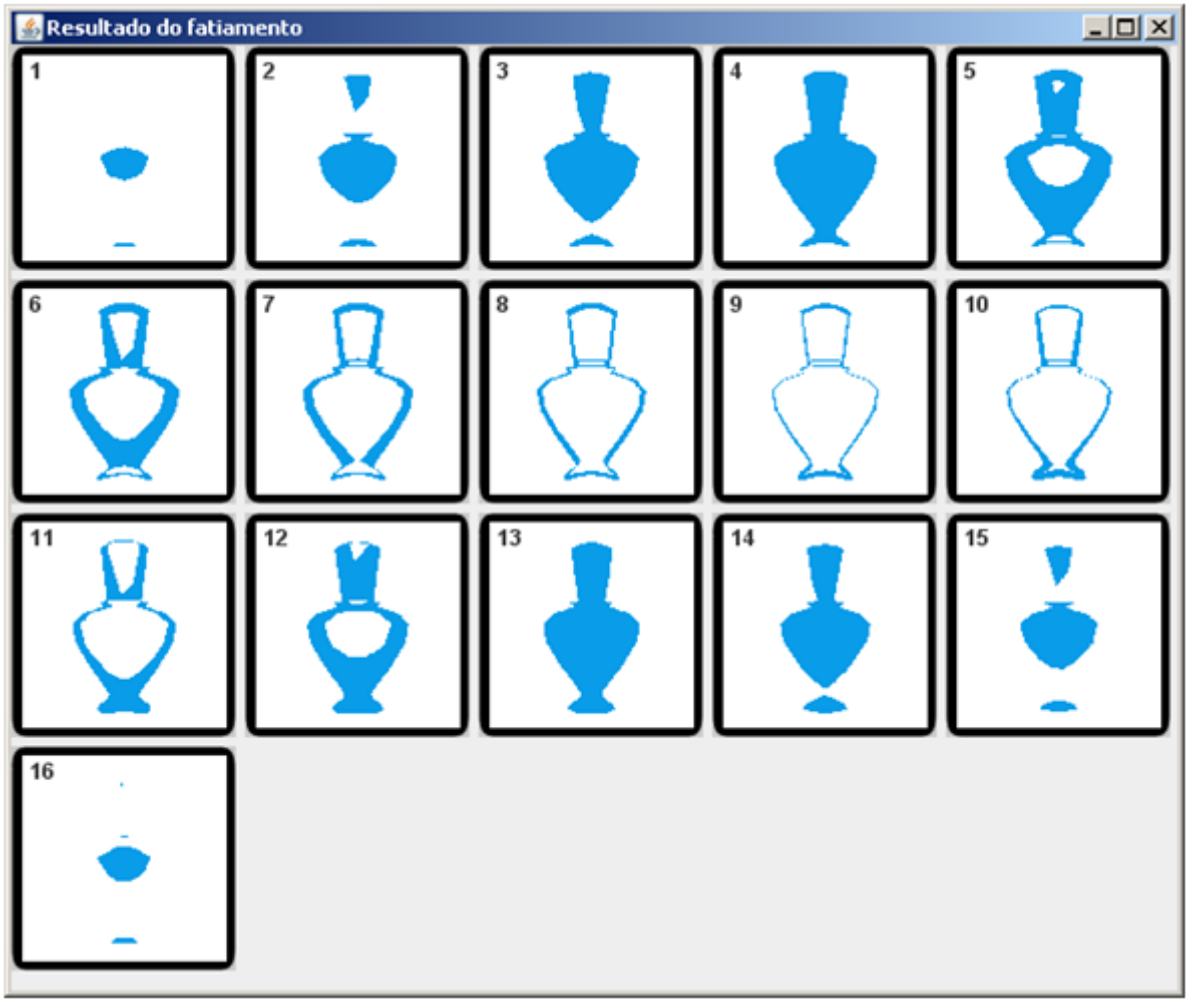


Figura 68 – Fatiamento realizado em uma garrafa.

Como já foi dito neste trabalho, existe um parâmetro no software que define a translação aplicada sobre o objeto durante o fatiamento. Quanto maior o valor definido, menor o número de camadas. Este teste utilizou o valor 0.01. No caso, o sistema gerou 16 fatias do sólido.

Uma segunda simulação no mesmo objeto (Figura 69) foi realizada, mudando o valor do parâmetro de translação para 0.001. O número de fatias passou de 16 para 160, sendo que apenas as 30 primeiras foram ilustradas na imagem.

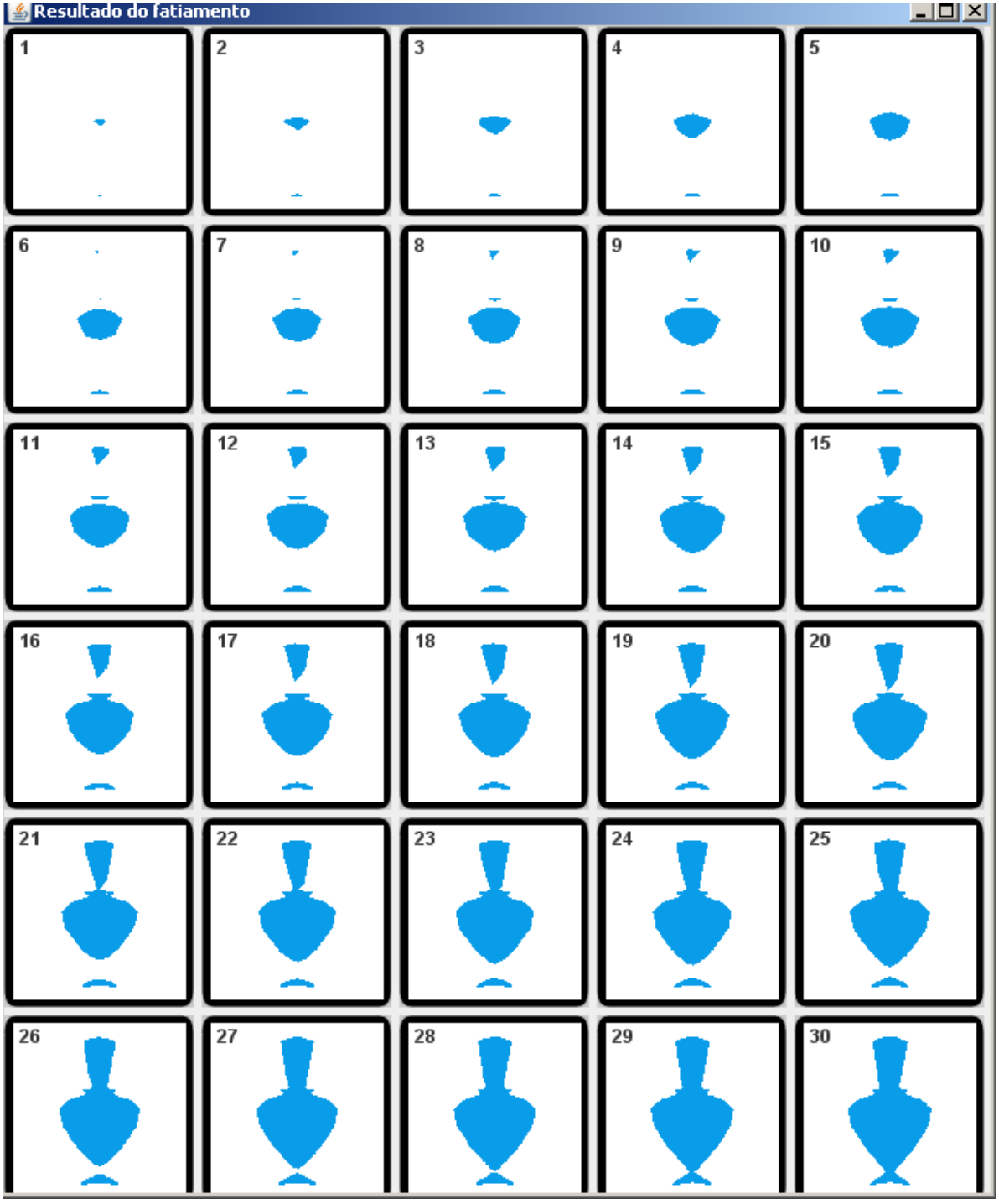


Figura 69 - Fatiamento realizado com mais detalhes.

O próximo teste (Figura 70) demonstra o fatiamento de um quadro com a palavra “NOY” inscrita.

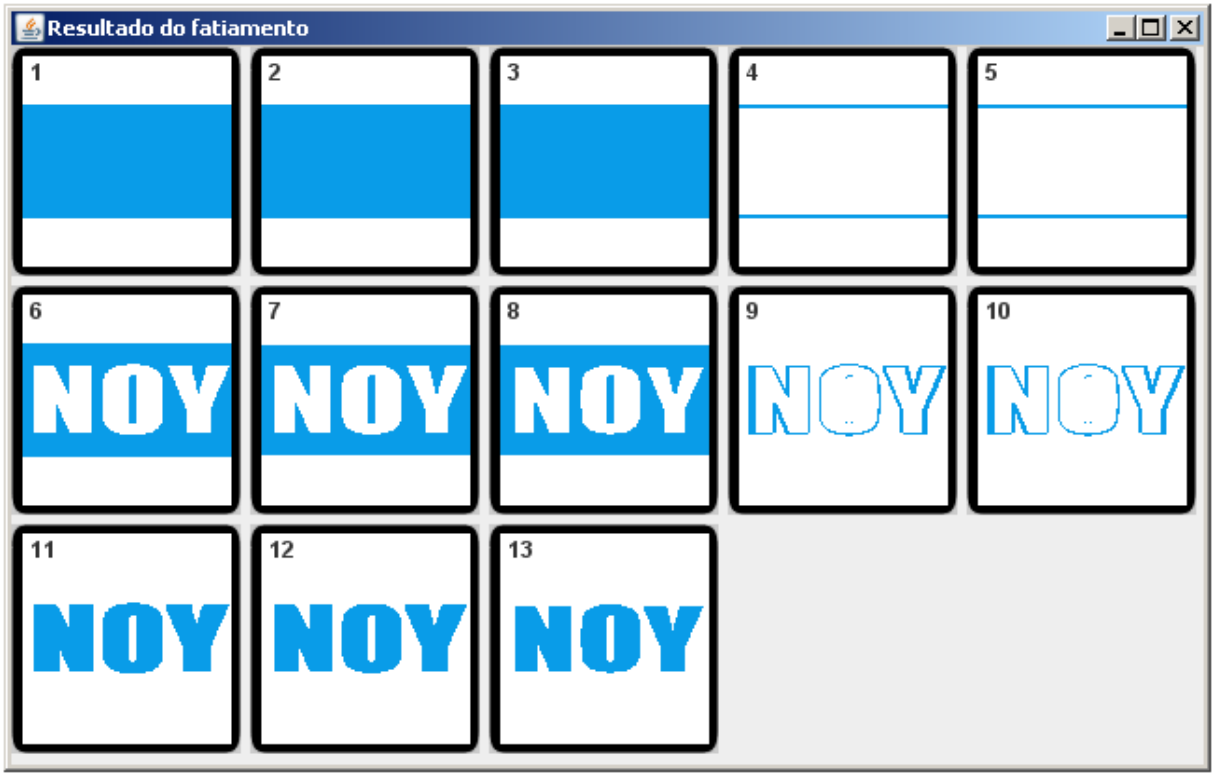


Figura 70 - Fatiamento do quadro "NOY".

Com a realização destes testes, percebeu-se que eventualmente pode surgir a necessidade do usuário descartar manualmente algumas das fatias obtidas. Em face disso, foi feita uma modificação no sistema para permitir a realização de tal tarefa. As fatias são exibidas com um *checkbox* na parte superior esquerda da camada, e, por padrão, este vem marcado. Caso seja desmarcado pelo o usuário, a fatia será descartada das próximas etapas do processo de prototipagem. A Figura 71 demonstra a melhoria descrita. Está destacado em vermelho uma camada que será desprezada.

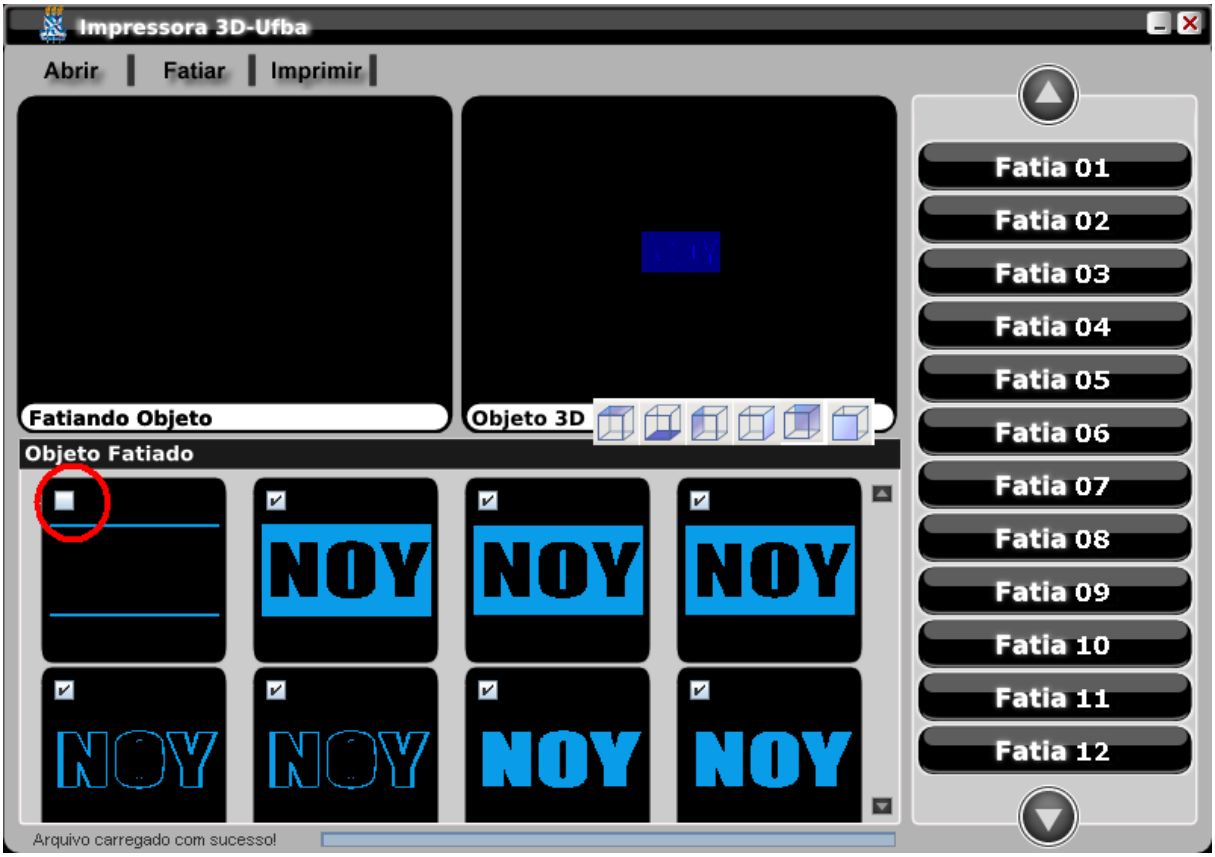


Figura 71 - Tela do sistema após o fatiamento.

Arquivos STL de tamanhos e detalhes variados foram carregados e testados. O sistema funcionou corretamente em todos os testes realizados. Variações foram feitas no parâmetro de translação e na posição de fatiamento do objeto (rotação). Os resultados do fatiamento aconteceram de acordo com o esperado em todas as situações.

6.2 Testes por simulação de trajetórias

Os testes realizados por meio de simulações utilizaram o CNCSimulator. Este *software* permite a demonstração em computador das rotas geradas. Para tanto, este simulador recebe como entrada um arquivo seguindo o padrão G e interpreta sequencialmente os comandos contidos neste arquivo, tal qual uma máquina real faria. Assim, é possível simular o comportamento da máquina de acordo com os comandos gerados pelo sistema CAM.

Este simulador foi utilizado para validar as seguintes funcionalidades do sistema:

- **Deposição das bordas**;
- **Preenchimento em zigzag;**
- **Preenchimento em espiral;**
- **Planejamento da deposição de toda a camada.**

Para o planejamento da deposição de uma camada, existe uma sequência estabelecida de rotas de preenchimento e intervalos em que há apenas a movimentação do extrusor, sem deposição. Todas estas rotas devem estar ordenadas, de modo que a máquina de prototipagem siga exatamente o planejado pelo sistema. A funcionalidade em questão abrange também os algoritmos anteriores; entretanto, está sendo testada a interação e organização da camada como um todo.

O software utilizado para a simulação dos resultados (CNCSimulator) foi projetado originalmente para a prototipagem rápida por subtração de material. Por este fato, foi necessário adaptá-lo para atender aos objetivos desta aplicação. A fim de visualizar as rotas durante as trajetórias de deposição, a ferramenta era abaixada de modo a entrar em contato com o bloco. O bloco consiste na massa inicial que é carregada no simulador. Aplicando o corte sobre este material, o sistema simula a usinagem. Desse modo, é possível visualizar a trajetória do extrusor. Nos momentos em que seriam geradas rotas sem deposição, a ferramenta era retirada do contato com o bloco e movida.

Com esta estratégia, foi possível simular tanto a deposição quanto o deslocamento do extrusor (sem extrusão). A Figura 72 mostra a tela inicial do simulador em questão e detalha a utilidade dos principais painéis.

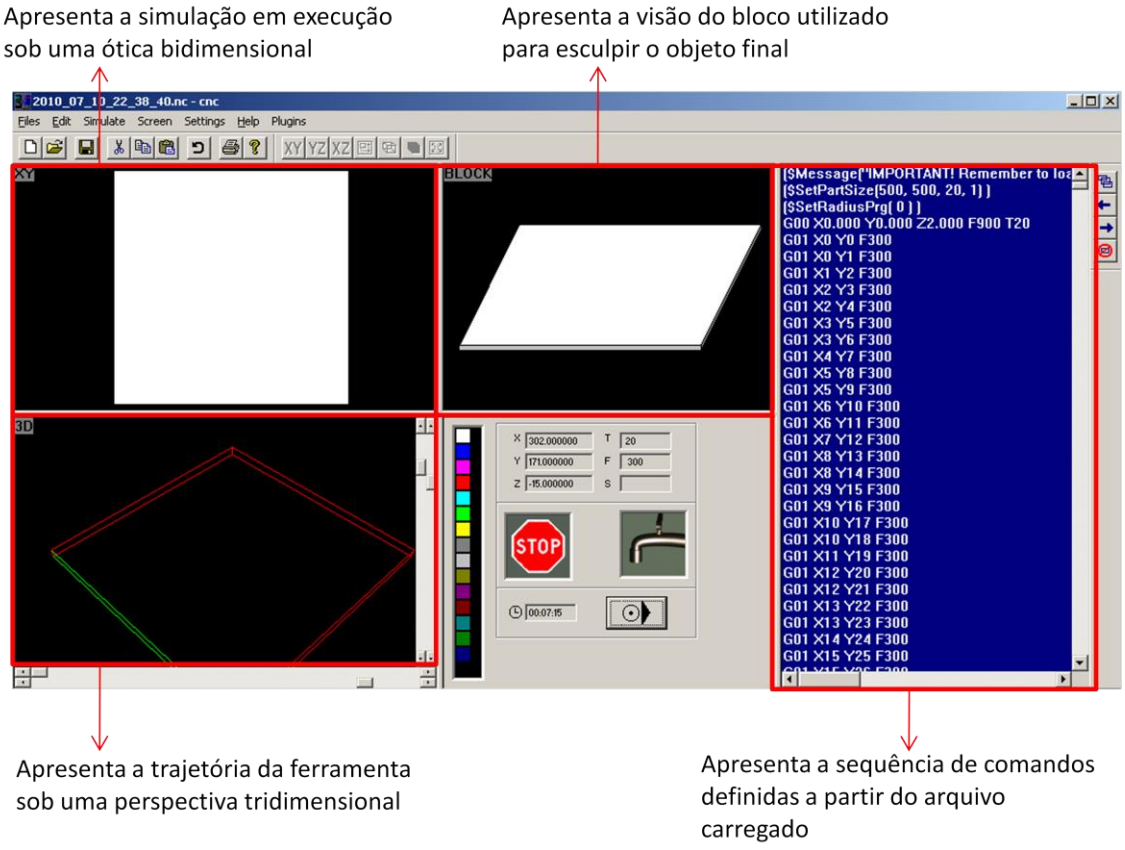


Figura 72 - Tela inicial do simulador.

6.2.1 Interface no sistema para geração dos arquivos G

Foi desenvolvida uma funcionalidade no sistema capaz de transformar as rotas geradas em arquivos seguindo o padrão G. Com isto, os testes de cada camada podem ser feitos de forma individual. A Figura 73 ilustra a tela desenvolvida com uma imagem escolhida dentre as várias fatias de objeto 3D. De acordo com o botão acionado, um dos algoritmos de geração de rotas é executado e a saída deste algoritmo é redirecionada para um arquivo. Tal arquivo, com a sequência de códigos G, é utilizado como entrada no CNCSimulator, a fim de apresentar a simulação do processo de prototipagem.

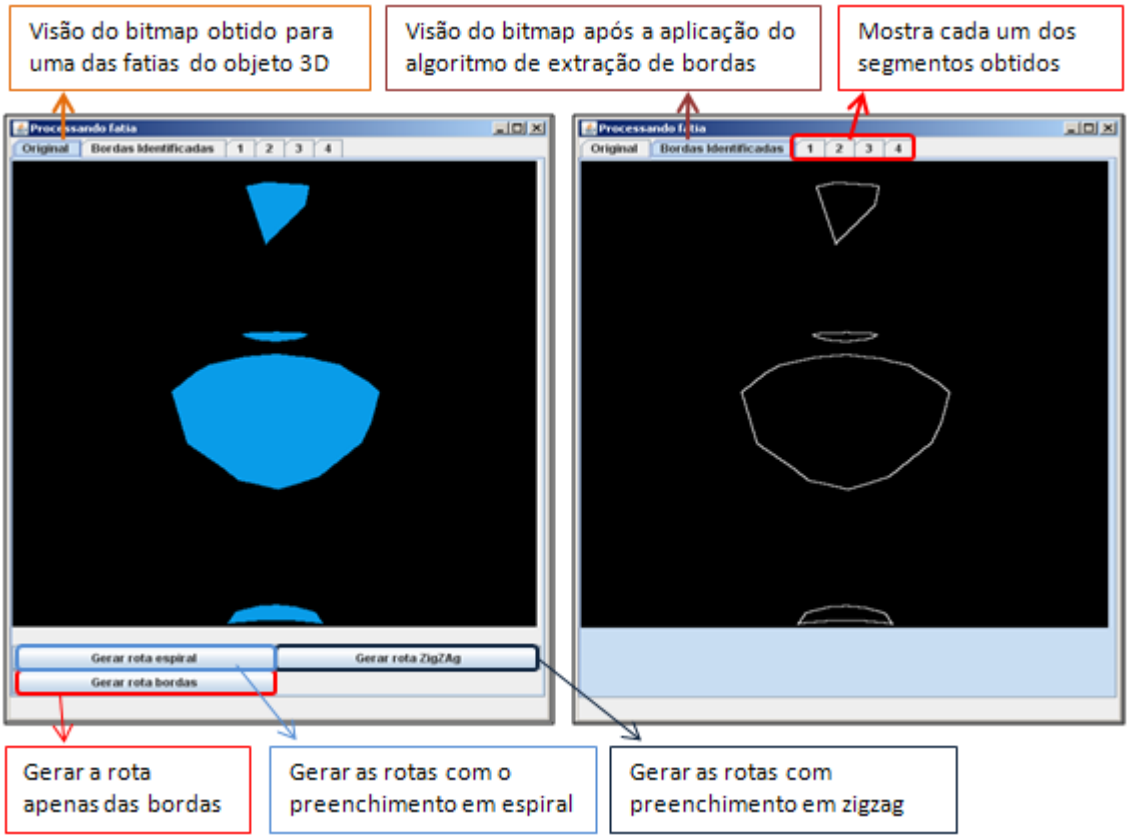


Figura 73 – Tela que permite a geração de código G para simulação.

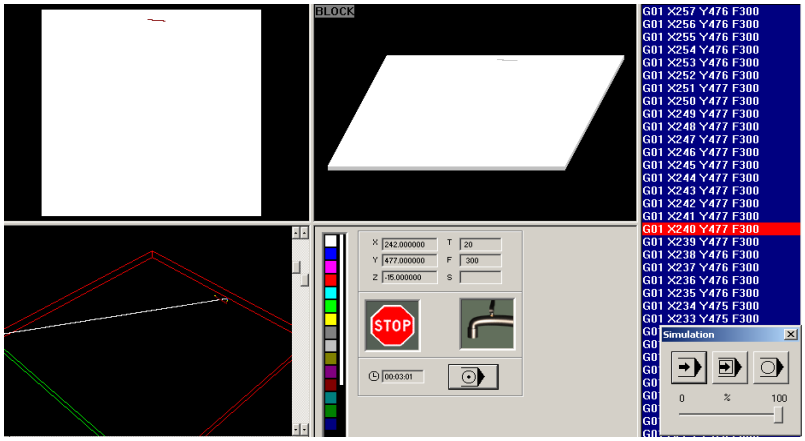
A primeira aba da tela (canto superior esquerdo) mostra a o *bitmap* obtido pelo processo de fatiamento. Esta aba mostra a camada em seu estado original. Nesta mesma aba há três botões na parte inferior da tela que permitem executar o algoritmo e gerar o arquivo. A descrição da funcionalidade que cada um dos botões ativa está descrita na imagem.

6.2.2 Geração das bordas

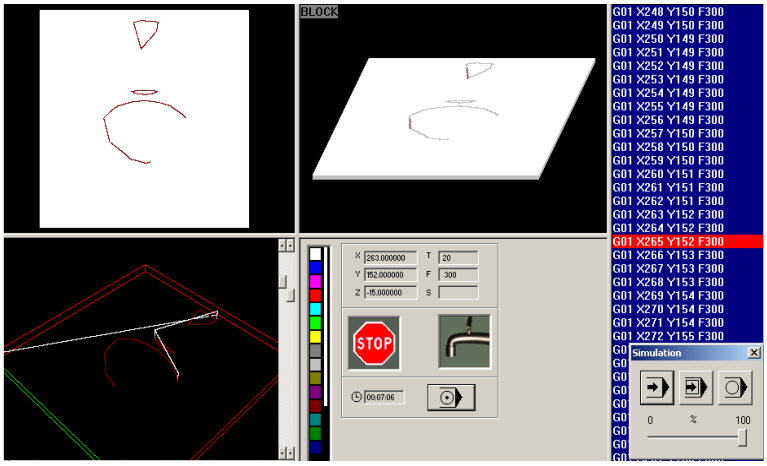
Os primeiros testes realizados foram justamente nos algoritmos de geração de bordas. Como visto, o primeiro passo no processo de deposição é identificar as bordas e gerar rotas para estas, a fim de se obter uma melhor qualidade superficial. Para testar se o algoritmo

desenvolvido para este fim estava funcionando de forma apropriada, diversas camadas foram escolhidas de vários objetos STL fatiados. Em seguida, o algoritmo era aplicado de forma a gerar toda a rota.

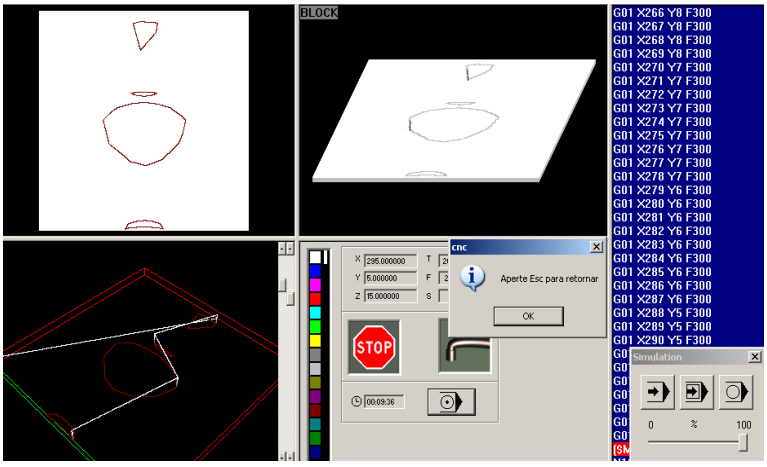
As imagens a seguir (Figura 74a, Figura 74b e Figura 74c) demonstram o processo de simulação. As imagens foram capturadas no início, meio e fim do processo.



a - início da simulação.



b - Imagem durante o processo de simulação.



c - fim da simulação.

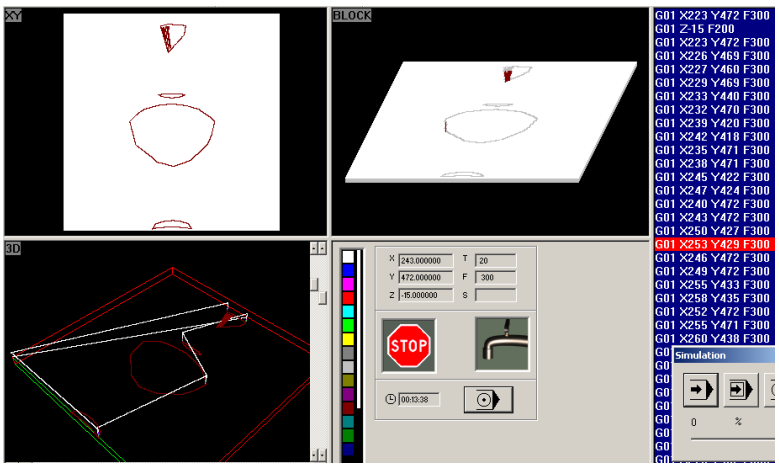
Figura 74 - Imagens do processo na simulação.

É possível perceber pela sequência da Figura 74 que a rota de deposição está de acordo com a informação da camada. A partir dos diversos testes realizados, ficou constatado que os algoritmos utilizados para esta funcionalidade atendiam às expectativas. Os algoritmos envolvidos desde o carregamento do arquivo 3D até esta funcionalidade são:

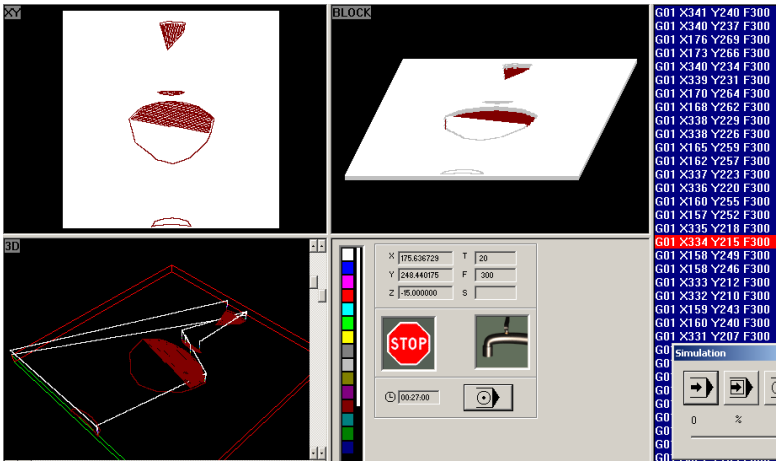
- Fatiamento de um sólido tridimensional (conversão em um bitmap 2D);
- Algoritmo de segmentação do bitmap;
- Identificação das bordas de cada segmento;
- Geração da rota para cada uma destas bordas (utilização do AStar de forma recursiva);
- Conversão da lista de pontos (rotas de deposição) em código G.

6.2.3 Preenchimento em zigzag

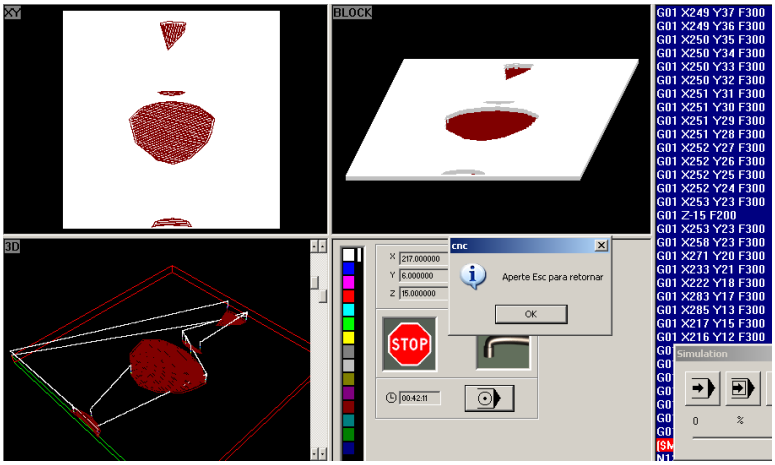
Como visto, para o planejamento em zigzag há também a necessidade da deposição das bordas. A diferença é que, após este processo, o interior de cada um dos segmentos existentes na camada deve ser preenchido. Dessa forma, esta funcionalidade reutiliza todos os algoritmos utilizados anteriormente, acrescido do algoritmo de deposição em zigzag. As imagens a seguir (Figura 75a, Figura 75b e Figura 75c), demonstram a simulação deste algoritmo.



a - Início do preenchimento em zigzag.



b –Preenchimento em zigzag.



c – Final do preenchimento em zigzag.

Figura 75 - Simulação do preenchimento em zigzag.

Na Figura 76, é apresentado o resultado do algoritmo em questão. Através do *zoom* aplicado no simulador foi possível perceber uma possibilidade de melhoria no algoritmo. O algoritmo foi projetado com um parâmetro configurável que determina a distância entre o preenchimento interno e as bordas da imagem. Inicialmente essa distância era testada apenas na direção em que a linha reta estivesse seguindo. Porém, como é possível observar na Figura 76, em alguns pontos o preenchimento ficava muito próximo das bordas, o que indicava que esta distância (*offset*) não estava sendo respeitada em todos os momentos.

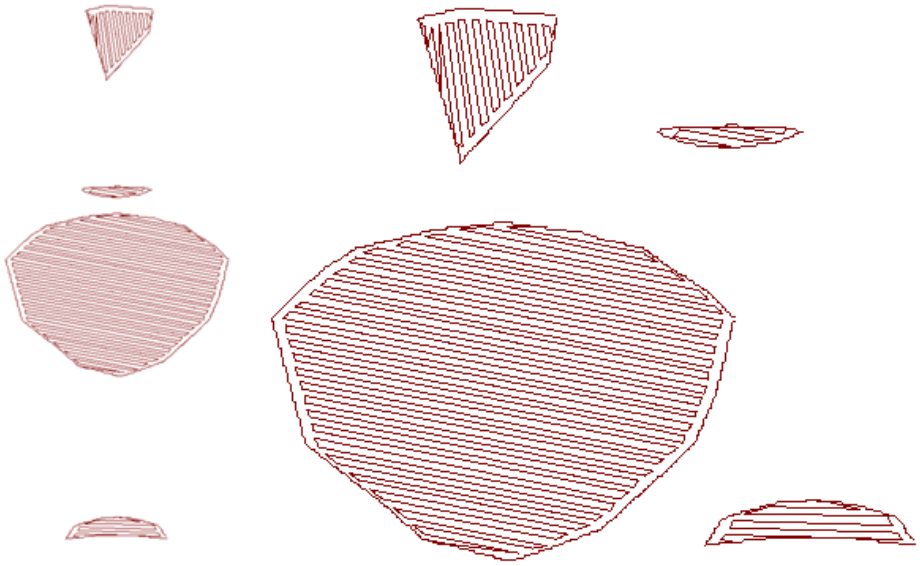


Figura 76 - *Zoom* aplicado ao resultado da simulação.

O algoritmo foi modificado a fim de incrementar essa verificação da distância em relação às bordas. Ao invés de verificar a distância apenas dos próximos pontos da reta, foi feito um teste ao redor do ponto em questão. Ou seja, o *offset* passou a ser um raio de distância em relação a qualquer ponto da borda; o que impede que seja gerada uma incoerência no preenchimento.

Na Figura 77, é possível comparar o resultado do preenchimento depois desta modificação. É perceptível que a modificação trouxe uma melhoria ao resultado do algoritmo, mantendo um *offset* constante em todas as direções, em relação às bordas interior e exterior da imagem.

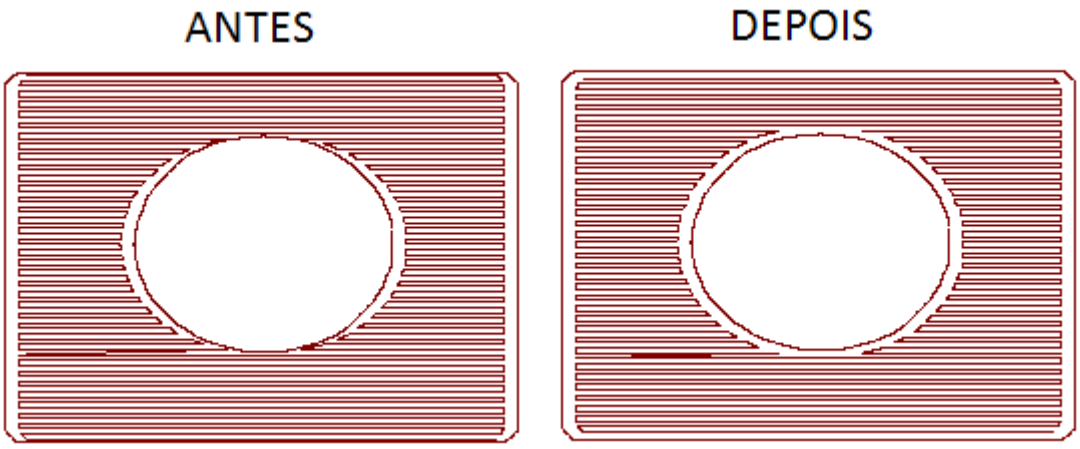
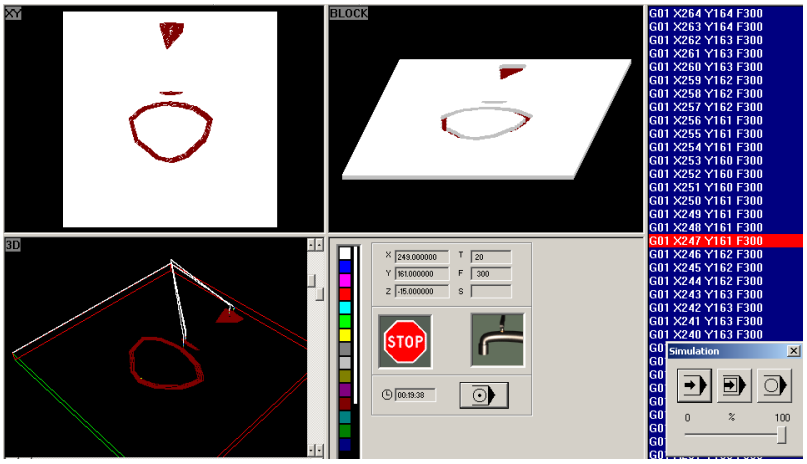


Figura 77 - Resultado da alteração do algoritmo.

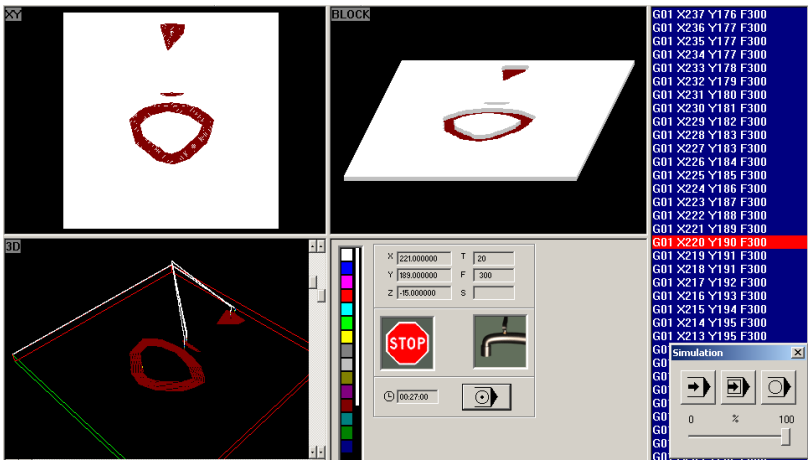
Diversas simulações foram realizadas e o algoritmo manteve uma constância nos resultados apresentados. Com base nestas simulações, é possível afirmar que o algoritmo de preenchimento em zigzag gera corretamente as rotas para as camadas.

6.2.4 Preenchimento em espiral

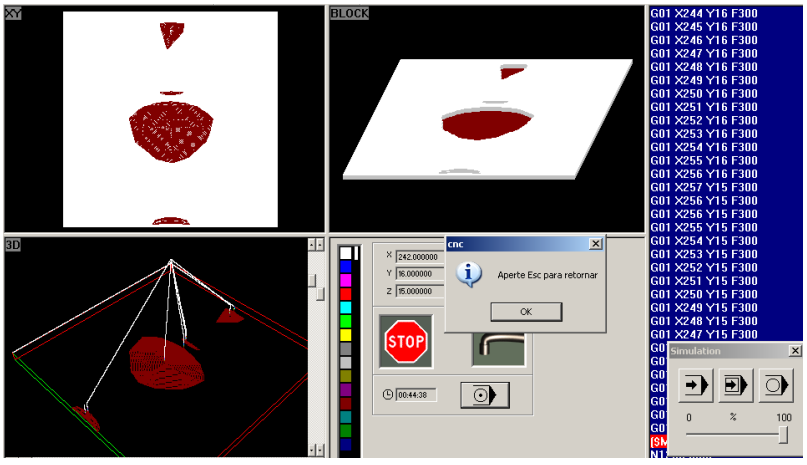
O algoritmo de preenchimento em espiral segue um principio diferente. A identificação das bordas do *bitmap* é aplicada de forma constante, a fim de se obter os vários espirais. Após a obtenção dos diversos espirais, estes são ligados. Os resultados das simulações deste algoritmo estão ilustrados abaixo (Figura 78a, Figura 78b e Figura 78c).



a – Início do preenchimento.



b - durante o processo.



c - Resultado final.

Figura 78 - Simulação do algoritmo de preenchimento em espiral.

Um zoom também foi aplicado ao resultado da simulação. À esquerda da Figura 79 é mostrada a imagem obtida ao final da simulação, enquanto à direita um *zoom* mais detalhado foi aplicado sobre cada um dos segmentos da imagem.

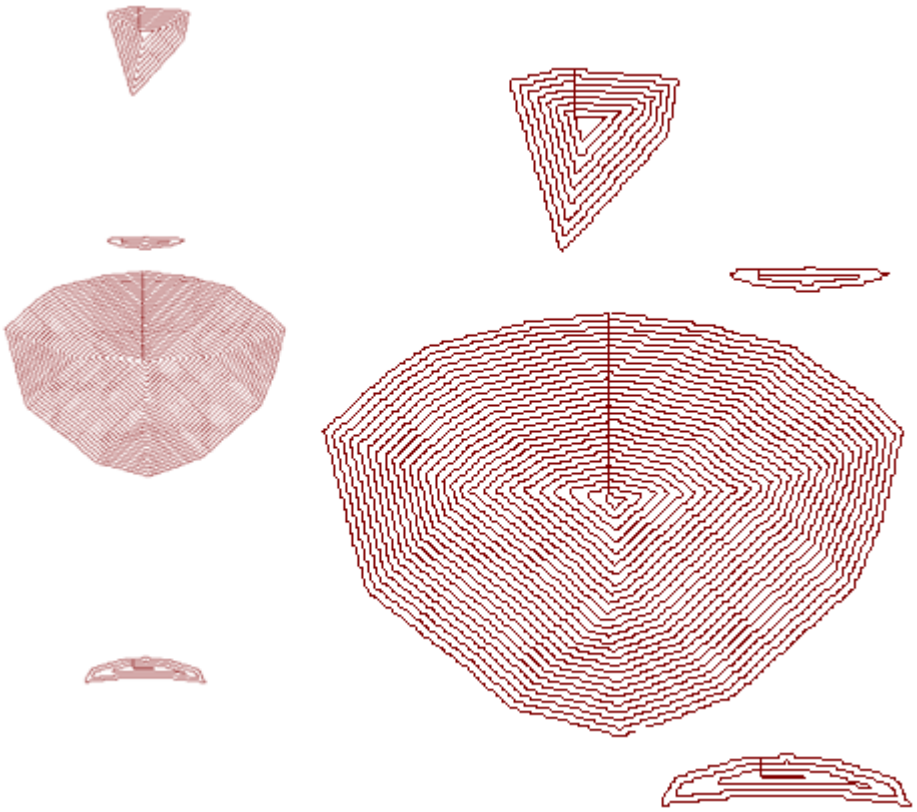


Figura 79 - Zoom aplicado ao resultado do preenchimento em espiral.

A principal observação a ser feita sobre o algoritmo em espiral refere-se à ligação entre os espirais. A implementação atual do algoritmo realiza a ligação sempre na vertical do espiral mais interno até o espiral mais externo; além disso, o sentido da ligação é sempre para cima. Esta é uma característica do algoritmo. Caso fosse considerada também a possibilidade de ligações entre as rotas para baixo, esquerda ou direita, haveria maiores possibilidades de ligações. Isto resultaria, em alguns casos, em rotas contínuas maiores em relação ao modelo atual, o que facilitaria e diminuiria o tempo de prototipagem do objeto. Esta é uma possibilidade a ser tratada em trabalhos futuros.

Uma questão a ser considerada é que atualmente nenhum outro software *open-source* implementa o preenchimento em espiral. De fato, o custo de implementação deste algoritmo é maior, porém um dos fatores que viabilizou a sua implementação foi a abordagem baseada em *bitmaps* do fatiamento. Além disso, o algoritmo desenvolvido permite a geração de rotas para imagens que contenham ilhas. Este outro fator é uma característica importante, pois existem diversas propostas presentes na literatura que não atendem a esta situação.

6.2.5 Comparativo espiral x zigzag

Os resultados de alguns dos testes feitos com os algoritmos de preenchimento serão ilustrados a seguir. O objetivo deste tópico é demonstrar que a resposta dos algoritmos foi satisfatória, mesmo com camadas em situações heterogêneas. A Figura 80 mostra algumas das camadas utilizadas para testes.

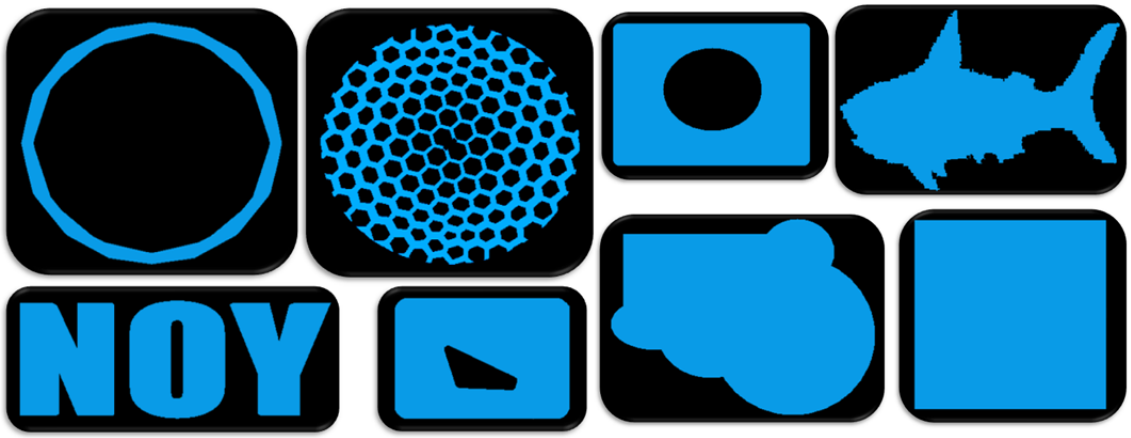
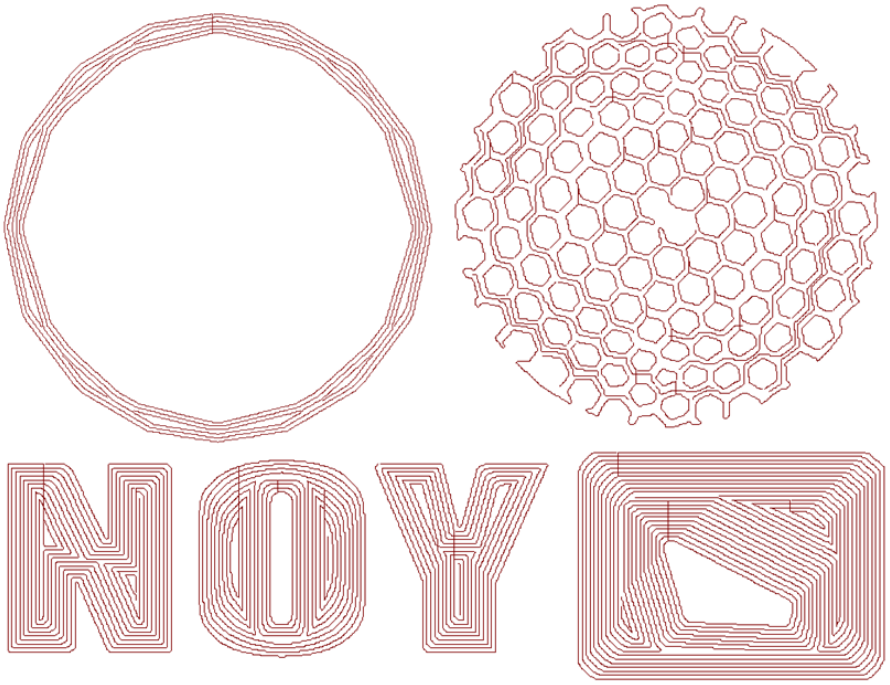
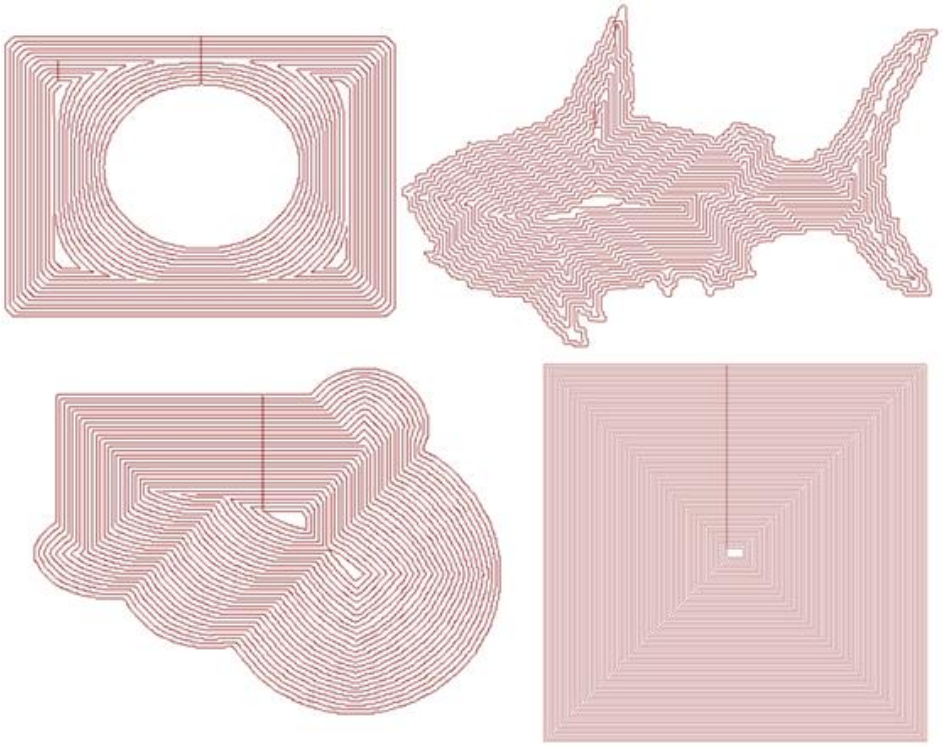


Figura 80 - Algumas das camadas utilizadas nos testes.

A Figura 81a e Figura 81b demonstram o resultado do algoritmo de preenchimento em espiral.



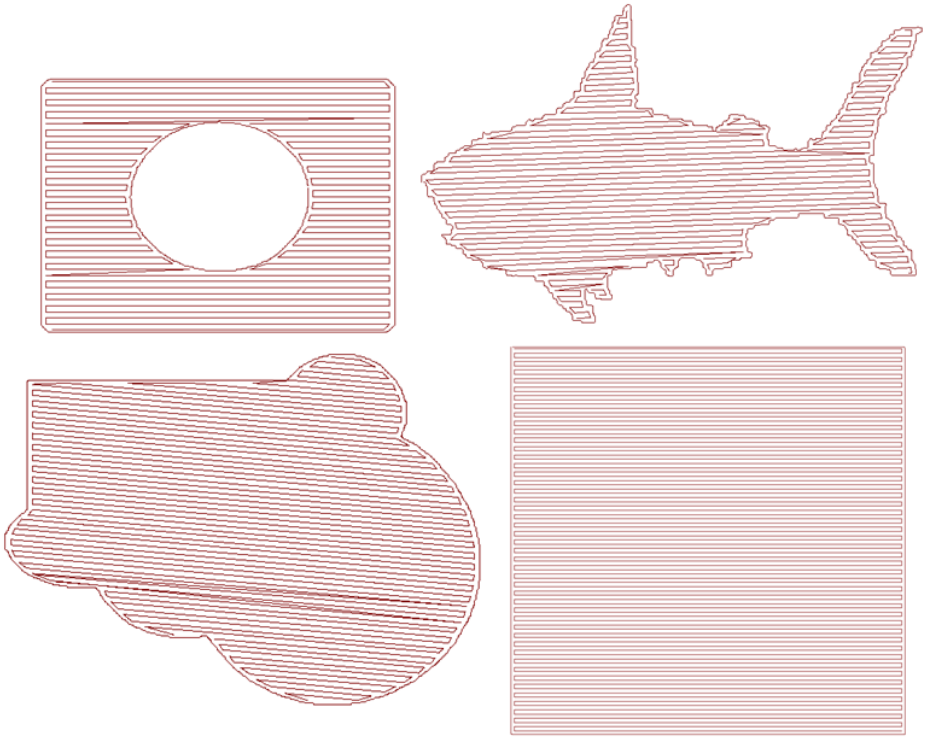
a – Preenchimento de quatro das camadas.



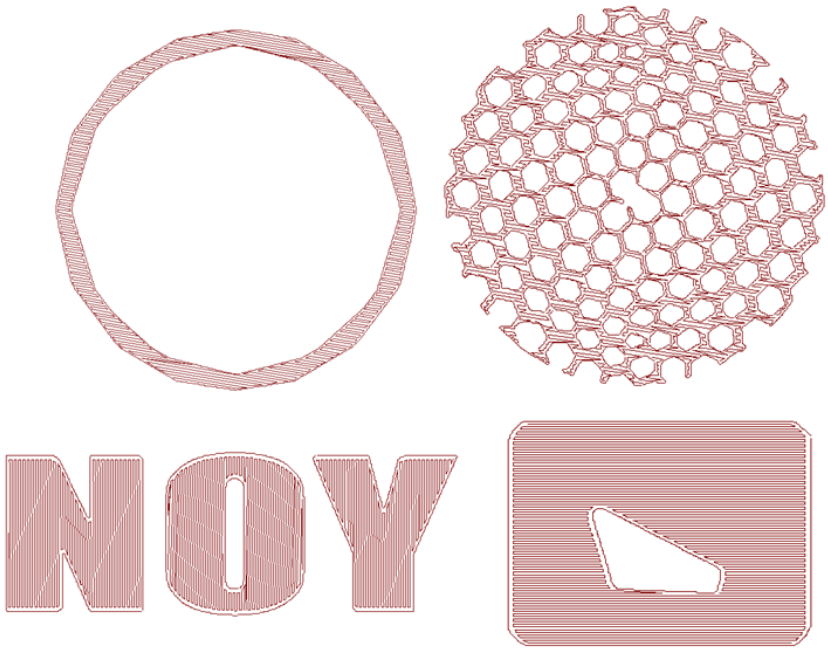
b – Preenchimento das últimas camadas.

Figura 81 - Preenchimento em espiral.

A Figura 82a e Figura 82b demonstram o resultado do algoritmo de preenchimento em zigzag.



a – Preenchimento de quatro das camadas.



b – Preenchimento das últimas camadas.

Figura 82 - Preenchimento em zigzag.

A partir dos resultados demonstrados, é possível perceber que o sistema conseguiu atender aos requisitos especificados para as duas opções de preenchimento.

6.3 Testes de movimentação dos eixos.

Nos primeiros testes efetuados com a máquina de prototipagem foi utilizada uma caneta ao invés do extrusor. O objetivo era analisar o comportamento dos eixos em relação aos comandos contidos nos arquivos NGC, os quais foram gerados a partir de camadas obtidas de arquivos STL.

A Figura 83 mostra a máquina de prototipagem rápida, destacando à direita a caneta utilizada para desenhar sobre o caderno. Para simular o peso do extrusor, foram adicionados dois *HDs* (*Hard Disks*) próximos à caneta.

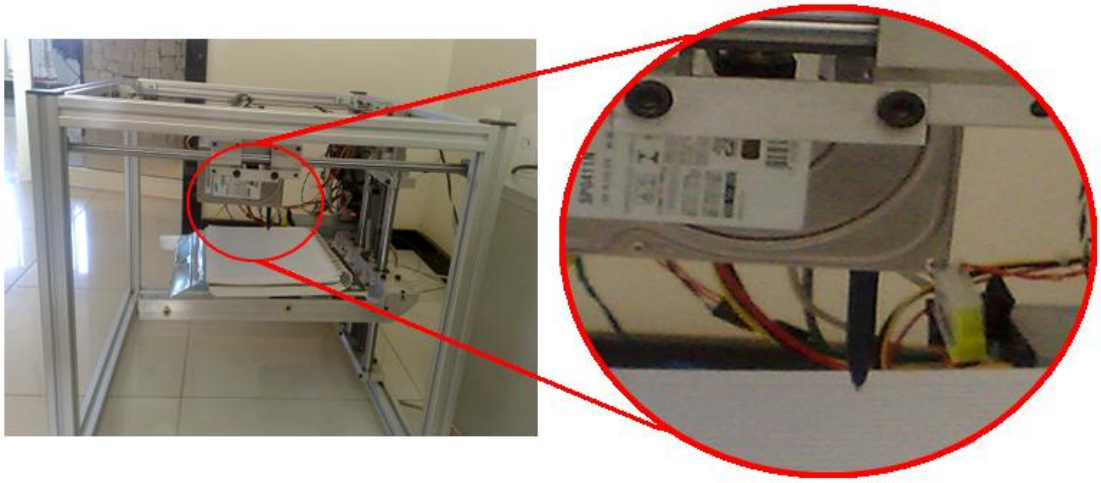


Figura 83 - Foto da máquina de prototipagem com a caneta em destaque.

O procedimento definido para os testes consiste em carregar no EMC o arquivo NGC, posicionar os eixos da máquina no ponto inicial e, em seguida, iniciar o teste. A Figura 84 mostra o programa EMC com um arquivo carregado e pronto para controlar a máquina.

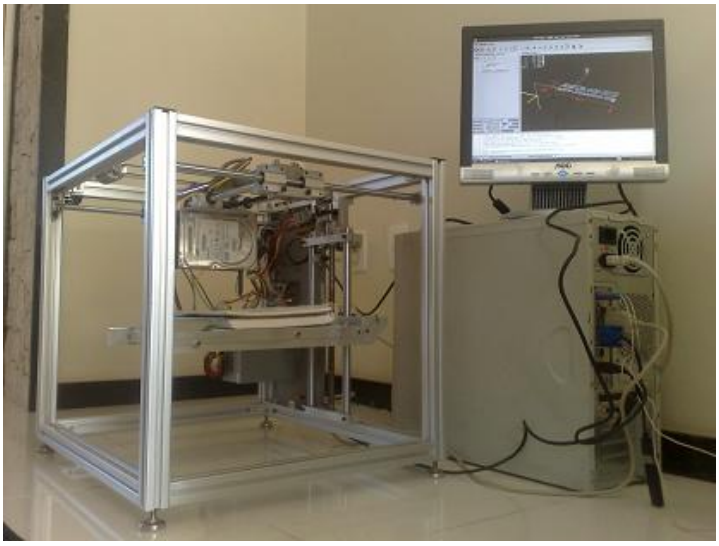


Figura 84 - Máquina de prototipagem.

Alguns testes foram realizados segundo o procedimento acima descrito. A Figura 85 mostra o resultado obtido após a realização do teste. Nesta ocasião, foi utilizado o preenchimento em zigzag.

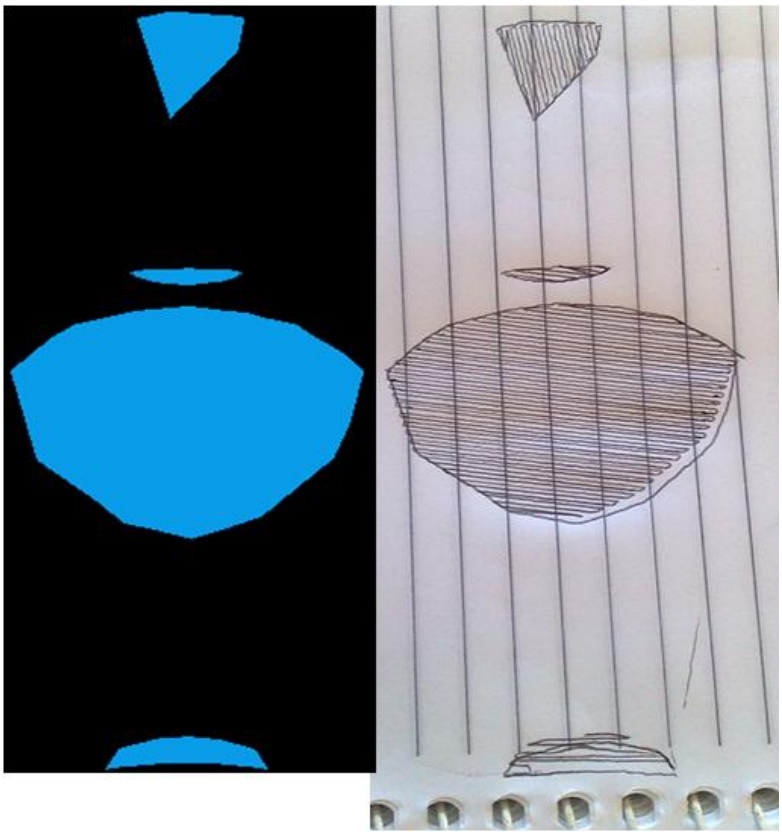


Figura 85 - Camada obtida com o preenchimento em zigzag.

Analisando a imagem, é possível perceber que houve dois desvios em relação ao objetivo. O primeiro ocorreu na terceira parte da camada: um afastamento excessivo do preenchimento em relação à borda pode ser observado. Este erro foi provocado porque sempre que a caneta entrava em contato com o caderno, provocava o deslocamento do papel. O acúmulo destes acontecimentos provocou o desvio.

A segunda distorção pode ser observada na última parte da camada. A mesma foi provocada pela inclinação que existe no espiral do caderno. Como esta parte do papel é mais alta, a caneta acabou prendendo-se mais fortemente ao papel e provocou um deslocamento muito maior.

A Figura 86 mostra a mesma camada da imagem anterior, com uma mudança na estratégia de preenchimento. Desta vez, foi utilizada a estratégia de preenchimento em espiral.

Figura 86 - Teste feito com a caneta utilizando o preenchimento em espiral.

A inconsistência que pode ser observada é que o eixo Z não obteve um deslocamento suficiente para retirar a caneta do contato com o caderno. Isto provocou as ligações entre os segmentos da camada.

A Figura 87 mostra os resultados de uma terceira avaliação feita no sistema.

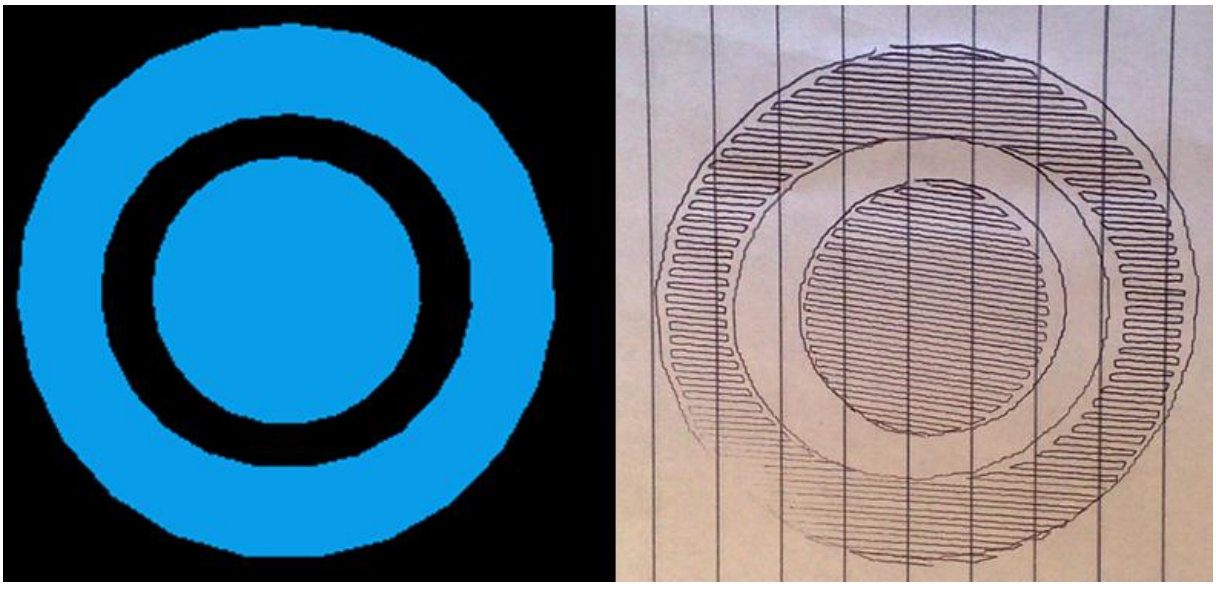


Figura 87 - Comparativo entre a camada e o resultado obtido no caderno.

Nos testes pode-se perceber pequenos desvios ou pontos em que o eixo Z não funcionou corretamente. Como observado anteriormente, estes problemas foram ocasionados por dois motivos principais: movimentação da folha do caderno quando entrava em contato com a caneta e inclinação do caderno em relação à mesa. Apesar dessas observações, o objetivo

traçado para esta etapa foi alcançado, pois a máquina conseguiu atender com a precisão necessária o processo de planejamento. Os eixos X, Y e Z obedeceram corretamente aos comandos definidos. A interpolação entre os eixos era outro ponto importante a ser avaliado e funcionou corretamente. Um exemplo da interpolação dos eixos está na Figura 87, durante o desenho do círculo (bordas da imagem).

6.4 Comparativo do sistema com o Insight, software comercial da Stratasys

A fim de analisar as possibilidades de melhorias no sistema e de trabalhos futuros, foi realizado um estudo comparativo entre a proposta deste trabalho e as principais funções contidas em um *software* de uma máquina FDM comercial da Stratasys, denominado Insight (STRATASYS, 2004). Este comparativo foi realizado com a ajuda de pesquisadores do CTI (CTI, 2010), que operam máquinas de prototipagem de diversas tecnologias e possuem grande experiência na utilização deste software.

No Insight, é possível definir diversos parâmetros que irão interferir no processo de prototipagem. O mesmo sistema é capaz de atuar em modelos diferentes de máquinas de prototipagem, desde que o *hardware* da máquina seja *Stratasys*.

Após a inicialização do software, o usuário deve informar qual o modelo da máquina de prototipagem que está utilizando e o tipo de filamento que servirá de insumo para o extrusor. Estes parâmetros vão interferir, dentre outras propriedades, na altura e espessura da camada, pois cada tipo de material possui propriedades físicas diferentes.

Após informar o modelo da máquina e o tipo de material que está sendo utilizado, o usuário pode carregar um arquivo STL no software. Uma função importante nos sistemas comerciais é a possibilidade de prototipar vários objetos em uma "corrida" da máquina (o termo corrida é utilizado para indicar um ciclo de prototipagem completo da máquina).

Depois que o arquivo STL é fatiado, há duas opções de preenchimento. A primeira é a opção em zigzag tradicional. Nesta opção, o usuário pode informar o valor de rotação entre o preenchimento da próxima camada em relação à anterior. A rotação entre as camadas é importante, pois irá interferir na resistência à tração do objeto prototipado. A segunda opção de preenchimento é a espiral. Neste caso, não há ligação entre as rotas em espiral geradas. Uma observação importante a ser feita é que, caso ocorresse a ligação das rotas, haveria um ganho em tempo e qualidade da prototipagem, pelo fato de não haver a necessidade de interromper e posicionar novamente o extrusor antes de reiniciar a deposição.

Outro parâmetro que pode ser configurado é a distância entre as linhas de deposição. Este parâmetro pode ser utilizado nas duas opções de preenchimento, determinando o quão próximo ficarão as linhas de deposição, influenciando no teor de porosidade do protótipo. Há também a possibilidade de determinar a espessura da camada. Este parâmetro irá influenciar na velocidade de deposição utilizada pela máquina (quanto maior for a espessura mais lento será o caminho de ferramenta).

Em relação ao algoritmo de suporte, o sistema gera malhas retangulares nos locais onde for necessária a presença deste tipo de material. A partir de todos estes parâmetros de preenchimento informados, é possível a criação de perfis de preenchimento no software. Tais perfis podem ser aplicados em uma determinada quantidade de fatias do modelo 3D, ou até mesmo em partes das camadas. Por exemplo: para preenchimento de um objeto 3D em que uma camada representasse a palavra "OI", poderia ser criado um perfil para preenchimento em espiral da letra O, enquanto um segundo perfil em zigzag poderia ser aplicado à letra I. Com esta função, o usuário pode percorrer as diversas fatias do objeto e aplicar perfis a cada uma delas, ou até mesmo aplicar um perfil em parte de uma fatia e outros perfis em outras partes.

Das funções analisadas, as que não são atendidas pela proposta deste trabalho e que podem ser implementados em trabalhos futuros são:

1. Possibilitar a prototipagem de vários objetos simultaneamente;
2. Permitir a parametrização da espessura da camada;
3. Permitir a criação de perfis e aplicá-los em camadas diferentes e em segmentos das camadas.

Esta dissertação está de acordo com o sistema avaliado nas seguintes funcionalidades:

1. Definir a altura da camada;
2. Configurar a distância entre os segmentos de rota gerados;
3. Permitir gerar rotas em espiral.

Das funções e características que este trabalho possui e que não são encontradas no sistema comercial avaliado estão:

1. No algoritmo espiral, as rotas geradas são interligadas, formando uma rota contínua de deposição;
2. Pelo fato de sua saída seguir o padrão G, pode ser utilizado em conjunto com outros sistemas abertos de prototipagem rápida. Alguns exemplos de projetos passíveis de compatibilização com o sistema desta dissertação são: MakerGear (MAKERGEAR,

2010), BitsFromBytes (BITSFROMBYTES, 2010), MakerBot (PETTIS et al., 2009) e RepRap (BOWYER, 2006);

3. É possível utilizar o sistema em outras tecnologias de prototipagem rápida, além da FDM. Em 2009, este trabalho foi aplicado a um protótipo de uma máquina LOM que estava em desenvolvimento por alunos do mestrado em mecatrônica da UFBA (GOMES et al., 2009). Em 2010, o sistema foi utilizado para realizar o planejamento em um protótipo de uma fresadora de corpo de guitarra (SOUZA e ACHY, 2010).

Como se vê, o sistema desenvolvido tem grande potencial de utilização, podendo ser comparado a um sistema comercial proprietário de alto custo, que vem sendo utilizado no mercado há alguns anos.

A Tabela 3 mostra um comparativo entre os sistemas CAM analisados:

Tabela 3 - Comparativo entre sistemas correlatos.

SOFTWARE	Fatiamento	Orientação	Planejamento em ZigZag	Planejamento Em Espiral	Planejamento por segmento
Skeinforge	Baseado no STL	Sim	Sim	Não	Não
RepRap	Baseado no STL	Sim	Sim	Não	Não
Fab@Home	Baseado no STL	Sim	Sim	Não	Não
Insight (Stratasys)	Baseado no STL	Sim	Sim	Sim	Sim
Dissertação	Baseado em bitmaps	Sim	Sim	Sim	Não

Analisando-se as dados da tabela, é possível destacar algumas considerações importantes. O primeiro item é o processo de fatiamento; a abordagem desta dissertação traz uma proposta de fatiamento baseada em *bitmaps*, diferenciando-se das outras abordagens, que tratam as informações diretamente do arquivo STL. O segundo ponto importante a ser destacado é o fato do planejamento em espiral ser atendido apenas pelo presente trabalho e por um software comercial (Insight).

7 Conclusão

A utilização da prototipagem rápida é uma tendência que tem ganhado força na indústria nos últimos anos. A propagação desta tecnologia para as PMEs tem sido buscada por diversos grupos de pesquisa e constitui um desafio que foi enfrentado neste trabalho.

Esta dissertação apresentou e avaliou o desenvolvimento de um sistema CAM para prototipagem rápida por adição de camadas. Este sistema viabiliza a solução de questões do processo da manufatura aditiva, como fatiamento, orientação e planejamento de trajetórias.

A validação deste sistema foi realizada inicialmente em ambiente simulado e, posteriormente, em um protótipo de uma máquina de manufatura aditiva. Arquivos tridimensionais com características variadas foram utilizados com o objetivo de atestar que o sistema apresentaria saídas coerentes, ainda que com variações em sua entrada. Os resultados apresentados foram utilizados para a realização de correções e melhorias até que fosse alcançada uma versão estável.

Das etapas que compõem o processo de manufatura aditiva, as seguintes funcionalidades do sistema desenvolvido podem ser destacadas:

- **Entrada de dados:** o formato de arquivo tridimensional utilizado para a entrada de dados no sistema foi o STL. Este formato foi escolhido por ser um padrão na maioria das máquinas de prototipagem rápida, sejam elas comerciais ou de código aberto. Esta medida possibilitou a utilização de uma série de sólidos tridimensionais disponíveis na internet, evitando-se a necessidade de criação de novos modelos. Mais importante, obteve-se uma interface para entrada do sistema CAM que é padrão na maioria dos sistemas CAD utilizados por PMEs.
- **Fatiamento:** foi apresentada uma proposta baseada em *bitmaps* com o método de fatiamento uniforme. O fator de incremento em z é parametrizável. Desta forma, pode-se definir o nível de detalhes e o número de camadas a ser obtido do arquivo tridimensional. A abordagem por *bitmaps* traz como vantagem a facilidade de compatibilizar o fatiamento com outros formatos tridimensionais, além do STL, pois não se trabalha diretamente sobre o conteúdo deste arquivo, e sim sobre a renderização do sólido na tela.
- **Orientação:** o sistema permite a definição de seis possíveis orientações para o sólido carregado. O usuário pode realizar as variações na orientação inicial de acordo com suas necessidades.

- **Planejamento de rotas:** para este processo, foram disponibilizadas duas opções de definição de trajetórias:
    - Zigzag: consiste na aplicação deste algoritmo às camadas do sólido fatiado. Constatou-se a necessidade de um pré-processamento envolvendo a deposição das bordas. Após esta etapa, inicia-se o preenchimento do interior da camada.
    - Espiral: a definição deste algoritmo consiste na geração de contornos recursivos aplicados à imagem inicial. Nesta dissertação, além da aplicação dos contornos, foi utilizado um método para ligação destes, a fim de gerar uma trajetória contínua de deposição.

  A disponibilidade destes recursos melhora muito o desempenho do sistema de prototipagem, inclusive no que se refere à resistência estrutural do protótipo gerado. A opção espiral, em particular, é alternativa inédita quando se trata de sistemas CAM para prototipagem de baixo custo
- **Saída:** o padrão utilizado como saída para o sistema foi o código G. A adoção deste padrão traz como vantagem a possibilidade de testar o sistema tanto em ambiente simulado quanto em um protótipo de uma máquina real. Além disso, facilita a integração deste sistema com máquinas de prototipagem pré-existentes, como MakerBot (PETTIS, 2009) ou RepRap (BOWYER, 2006).

Diante do que foi exposto, é possível afirmar que se atingiu o objetivo desta dissertação: desenvolver um sistema CAM capaz de atuar em uma máquina de manufatura aditiva. A seguir, os objetivos secundários que foram definidos no início deste trabalho serão avaliados:

- **Auxiliar o produto mecatrônico a atender requisitos de protótipos, como tolerância dimensional adequada, baixa rugosidade superficial e resistência mecânica**.

  Algumas das estratégias utilizadas durante o processo de planejamento foram empregadas visando atender a este objetivo. A razão de o sistema planejar inicialmente a deposição das bordas da camada é viabilizar a redução da rugosidade superficial. A resistência mecânica do protótipo advém da tecnologia de prototipagem utilizada (FDM), associada às opções de preenchimento do interior da camada disponibilizadas pelo sistema (espiral ou zigzag).
- **Disponibilizar um método simplificado de operação do software:** as três ações básicas que foram definidas na interface gráfica do sistema (abrir, fatiar, e imprimir) visam facilitar o uso do *software* por parte do usuário. Além disso, foi disponibilizada

a opção de visualização e alteração da orientação do objeto prototipado na tela principal do sistema.

- **Prover uma interface amigável para fins de parametrização e configuração:** Alguns parâmetros de entrada dos algoritmos são justamente para facilitar esse aspecto do sistema. Exemplos de parâmetros que podem ser facilmente alterados no sistema são: espessura da camada (definido através do parâmetro de translação do objeto carregado), distância entre os espirais, distância entre as linhas paralelas do zigzag e distância entre o preenchimento do interior e bordas da camada.
- **Validar o sistema desenvolvido num protótipo funcional, tendo como base o método FDM: c**omo demonstrado, foram realizados testes das rotas planejadas por este sistema em um protótipo de uma máquina de prototipagem rápida e esta atendeu de forma adequada o que foi definido pelo sistema.
- **Permitir a integração deste sistema com outras soluções disponíveis:** A utilização de código G como saída do resultado de planejamento é justamente para compatibilizar o sistema desenvolvido nesta dissertação com outras máquinas de prototipagem rápida. É possível integrar este sistema com o Replicator G (interpretador de código G do RepRap) ou qualquer hardware controlado pelo EMC.

Visando o aprimoramento dos resultados obtidos por esta dissertação, ficaram definidas algumas sugestões que podem ser exploradas em trabalhos futuros. Em seguida, são relacionados os resultados produzidos durante o desenvolvimento deste trabalho.

7.1 Trabalhos futuros

Como é de se esperar, um trabalho como este mais abre caminhos do que os conclui. Muitas possibilidades foram abertas, outras ficaram ainda inexploradas. Melhorias podem e devem ser implementadas. Destacam-se, então, como sugestões para trabalhos futuros os seguintes tópicos:

Entrada de dados:

1. Permitir que sejam carregados vários arquivos simultaneamente e que estes sejam prototipados em uma mesma corrida.
2. Possibilitar a utilização de outros formatos de arquivos tridimensionais, como o VRML.

Fatiamento: disponibilizar a opção de fatiamento adaptativo e avaliar o comportamento do sistema com esta abordagem.

Orientação: permitir a definição da orientação de forma automática, de acordo com os requisitos do cliente, por exemplo: rugosidade mínima da face, mínimo volume de suporte, altura mínima do objeto.

Planejamento de rotas:

1. Incrementar o algoritmo em espiral para permitir a ligação das rotas em todas as direções, ao invés de utilizar apenas a ligação na vertical.
2. Implementar outras trajetórias de preenchimento, como o *mesh* - disponibilizado pelo SkeinForge (BITSFROMBYTES, 2010).
3. Utilizar métodos de ordenação de trajetórias, a fim de reduzir o caminho total de ferramenta. Um método que poderia ser aplicado ao sistema seria o de Volpato et al. (2009).
4. Permitir a criação de perfis de planejamento. Essa função possibilitaria realizar uma variação de estratégias de preenchimento em camadas diferentes ou em partes diferentes da mesma camada.

Além dos pontos mencionados, há aspectos de integração do sistema com o protótipo da máquina que também podem ser trabalhados visando à melhoria de desempenho do processo de prototipagem rápida:

1. Parametrizar a espessura da camada depositada.
2. Parametrizar a altura física da camada.
3. Testar o planejamento de estruturas de suporte.
4. Promover uma integração transparente do sistema com o EMC2, permitindo uma calibração automática da máquina antes de iniciar o processo de prototipagem.

7.2 Publicações

O desenvolvimento deste trabalho ensejou desdobramentos de interesse acadêmico, com destaque para os seguintes:

1. Um sistema CAM para máquinas de prototipagem rápida de baixo custo. MARTINS, S.R.M.; LEPIKSON, H. A.; COSTA, B.L.S.. VI CONEM - Congresso Nacional de Engenharia Mecânica, 2010, Campina Grande, PB. Anais do VI CONEM - Congresso Nacional de Engenharia Mecânica. Rio de Janeiro, RJ : ABCM - Associação Brasileira de Ciências e Engenharia Mecânica, 2010.

Concurso Idéias Inovadoras - 1º lugar na categoria Mestrando, Fundação de Amparo à Pesquisa e Extensão do Estado da Bahia - FAPESB. 2008.

Portanto, o objetivo traçado para esta dissertação foi alcançado. Os resultados mostraram que o trabalho tem potencial para unir uma produção acadêmica ao desenvolvimento de um produto que poderá ser útil à industria.

REFERÊNCIAS

ARDUINO. **Open-source physical computing platform**. Disponível em: <http://www.arduino.cc>. Acesso em: 10 out. 2010.

ASTM. **Standard Terminology for Additive Manufacturing Technologies**. F 2792–09. Ann. Book ASTM Stand. Philadelphia, 2009.

BERG, de M.; CHEONG, O.; KREVELD, M. V.; OVERMARS, M. **Computational Geometry: Algorithms and Applications.** Third Edition. Berlin: Springer-Verlag, 2008.

BEZERRA, E. Princípios **de análise e projeto de sistemas com UML**. Rio de Janeiro: Campus, 2002.

BITSFROMBYTES. **Affordable 3D Printing.** Disponível em: <http://www.bitsfrombytes.com>. Acesso em 27 nov. 2010.

BOWYER, A. 2004. **Wealth Without Money: The Background to the Bath Replicating Rapid Prototyper Project**. Disponível em: <http://people.bath.ac.uk/ensab/replicator/background.html>. Acesso em 15 nov. 2010.

BOWYER, A. 2006. **The RepRap Project**. Disponível em: <http://reprap.org>. Acesso em 31 out. 2009.

CHUA C. K.; LEONG K. F.; LIM C. S. **Rapid Prototyping: Principles and Applications**. Second Edition. Singapore: World Scientific, 2003.

CNC Simulator. **CNC Simulator**. Disponível em: <http://www.cncsimulator.com>. Acesso em 10 dez. 2009.

CNPq. **Diretório dos Grupos de Pesquisa no Brasil**. Disponível em: <http://dgp.cnpq.br/buscaoperacional>. Acesso em 04 dez. 2010.

COSTA, B. L. S. **Desenvolvimento de um sistema de manufatura aditiva de baixo custo**. Dissertação (mestrado) - Universidade Federal da Bahia, Instituto de Matemática e Escola Politécnica, Salvador, 2011.

COSTA, L. da F.; CESAR JR, R. M. **Shape analysis and classification: theory and practice**. Boca Raton: CRC Press, 2001.

CRUMP, S. S. **Apparatus and method for creating three-dimensional objects**. United States Patent US5121329. 1989.

CTI. **Centro de Tecnologia da Informação Renato Archer**. Disponível em: <http://www.cti.gov.br>. Acesso em 03 dez. 2010.

DECKARD, C. R. **Method and apparatus for producing parts by selective sintering**. United States Patent US4863538. 1989.

DRAGOMATZ, D.; MANN, S. **A Classified Bibliography of Literature on NC Tool Path Generation.** Computer-Aided Design, Vol 29, No 3. 1997.

EMC2. **Enhanced Machine Controller**. Disponível em: <http://linuxcnc.org/>. Acesso em: 10 out. 2010.

FALCK, D. **EMC Handbook- G-Code Programming Basics**. Disponível em: <http://www.linuxcnc.org/handbook/gcode/g-code.html>. Acesso em 21 nov. 2010

FEYGIN, M.; ASHLAND, A. **Apparatus and method for forming an integral object from laminations**. United States Patent US4752352. 1988.

GIBSON, I. **Software Solutions for Rapid Prototyping.** London: Professional Engineering Publishing, 2002.

GIBSON, I.; STUCKER, B.; ROSEN, D. W. **Additive Manufacturing Technologies: Rapid Prototyping to Direct Digital Manufacturing**. Berlin: Springer, 2010.

GOMES, M. R.; LIMA, M. F.; BANDEIRA, A. M. **Projeto de sistema de prototipagem rápida LOM – Relatório Final**, Programa de pós-graduação em Mecatrônica, disciplina Sistemas Mecatrônicos, UFBA, 2009.

HART, P. E.; NILSSON, N. J.; RAPHAEL, B. **A formal basis for the heuristic determination of minimum cost paths**. IEEE Transactions on Systems Science and Cybernetic, p. 100–107, 1968.

HELD, M. **On the computational geometry of pocket machining**. Berlin: Springer-Verlag, 1991.

HOEKEN, Z.; PETTIS, B. 2010. Disponível em: <http://www.thingiverse.com>. Acesso em 27 nov. 2010.

HOPKINSON, N.; HAGUE R.; DICKENS P. **Rapid Manufacturing: An Industrial Revolution for a Digital Age**. New York: John Wiley, 2006.

HTH. **S.N.A.P - Scaleable Node Address Protocol**, 2002. Disponível em: <http://www.hth.com/snap/>. Acesso em: 15 nov. 2010.

HUANG, B. Development of a software procedure for Curved Layered Fused Deposition Modelling (CLFDM). Tese (Doutorado) - School of Engineering, AUT University, Auckland, New Zealand, 2009.

HULL, C. W. **Apparatus for production of three-dimensional objects by stereolithography**. United States Patent US4575330. 1986.

ISERMANN, R. **Modeling and design methodology for mechatronic systems**. Transactions on Industrial Eletronics, Vol.1, No.1, pp. 16-28, 1996.

KAMRANI, A. K.; NASR, E. A. **Rapid Prototyping: Theory and Practice**. New York: Springer, 2006.

KAO, J. **Process Planning for additive/subtractive solid freeform fabrication using medial axis transform.** Tese (Doutorado) - Stanford University, Stanford, California, 1999.

KOC, B. **Computational geometric analysis and planning for 3D rapid prototyping**

processes. Tese (Doutorado) - North Carolina State University, Raleigh, North Carolina, 2001.

KULKARNI, P.; MARSAN, A.; DUTTA, D. **A review of process planning techniques in layered manufacturing**. Rapid Prototyping Journal, 6(1), pp. 18-35, 2000.

LIPSON, H.; MALONE, E. **The Fab@Home Project**, 2006. Disponível em: <http://fabathome.org>. Acesso em 07 set. 2010.

LIPTON, J. **Fab2Gcode**, 2010. Disponível em: <https://launchpad.net/fab2gcode>. Acesso em 15 jan. 2011.

LIRA, V. M. **Desenvolvimento de Processo de Prototipagem Rápida via Modelagem por deposição de formas livres sob temperatura ambiente de materiais alternativos.** Tese (Doutorado) - Escola Politécnica, Universidade de São Paulo, São Paulo, 2008.

LIXANDRÃO FILHO, A. L.; CHEUNG, P. Y. C.; NORITOMI, P. Y.; SILVA, J. V. L.; COLANGELO, N.; LIPSON, H.; BUTCHER, J. T.; MALONE, E.; INFORCATTI NETO, P. **Construction and adaptation of an open source rapid prototyping machine for biomedical research purposes - a multinational collaborative development.** In: International Conference on advanced research in virtual and rapid prototyping, 2009, Leiria, Portugal. Proceedings. Leiria: CRC Press, 2009.

MAKERGEAR. Disponível em: <http://www.makergear.com>. Acesso em 27 nov. 2010.

MALONE, E.; LIPSON, H. **Fab@Home: the personal desktop fabricator kit**. Rapid Prototyping Journal, Vol. 13, No. 4, pp. 245-255, 2007.

MARCHAND-MAILLET S.; SHARAIHA Y. M. **Binary Digital Image Processing: A Discrete Approach.** San Diego: Academic Press, 2000.

MCDONALD, J. A.; RYALL, C. J.; WIMPENNY D. I. **Rapid Prototyping Casebook**. London: Professional Engineering Publishing, 2001.

PANDEY, P. M., REDDY, N. V., DHANDE, S. G. **Slicing procedures in layered manufacturing: a review**, Rapid Prototyping Journal, Vol. 9 No.5, pp.274-88, 2003.

PARK, S. C.; CHUNG, Y. C. **Offset tool-path linking for pocket machining**. Computer Aided Design, Vol. 34, No. 4, pp 299-308, 2001

PETTIS, B.; SMITH, Z.; MAYER, A. 2009. **MakerBot Industries: robots that make things**. Disponível em: <http://makerbot.com>. Acesso em 08 nov. 2010.

RABIN, S. **AI Game Programming Wisdom**. Hingham: Charles River Media, 2002.

RAMASWANI, K. **Process Planning for shape deposition manufacturing**. Tese (Doutorado) - Stanford University, Stanford, California, 1997.

ROBTEC. Desenvolvimento de produtos. Disponível em: <http://www.robtec.com>. Acesso em: 03 dez. 2010.

SACHS, E. M.; HAGGERY, J. S.; CIMA, M. J.; WILLIAMS, P. A. **Three-dimensional printing techniques**. United States Patent US5204055. 1993

SELLS, E. A. **Towards a Self-Manufacturing Rapid Prototyping Machine.** Tese (Doutorado) - University of Bath, Bath, 2009.

SELMAN, D. **Java 3D Programming**. Greenwich: Manning Publishers, 2002.

SOUZA, R. Q. S.; ACHY, A. R. A. **Projeto de Fresadora de Corpo de Guitarra – Relatório Final**. Programa de pós-graduação em Mecatrônica, disciplina Sistemas Mecatrônicos, UFBA, 2010.

STRATASYS, Insight. Stratasys, Eden Prairie, MN, 2004. Disponível em: <www.stratasys.com>. Acesso em 09 dez. 2010.

SUN, Microsystems. **Java 3D API specification**, 2000. Disponível em: <http://java.sun.com/products/java-media/java3d>. Acesso em 28 nov. 2009.

VOLPATO, N. **Prototipagem rápida: tecnologias e aplicações.** São Paulo: Edgard Blücher, 2007.

VOLPATO, N.; BABA, Julio Seiji Okada ; MANCZAK, Tiago. **A New 2D Inner/Outer Contour Identification Method for Layer Manufacturing**. In: 3rd International Conference on Advanced Research in Virtual and Rapid Prototyping (VRAP), 2007a, Leiria. Virtual and Rapid.

VOLPATO, N.; FOGGIATTO, J. A.; LIMA, M. V. A. de; MANCZAK, T. **Uma Otimização da Estratégia de Preenchimento do Processo FDM**. 4º Congresso Brasileiro de Engenharia de Fabricação, 2007b.

WALLICH, P. **3-D Printers Proliferate: But desktop manufacturing isn't yet ready for your desk.** IEEE Spectrum, 2010

WOHLERS, T.; GRIMM T. **Is CNC Machining Really Better Than RP?**, Time-Compression Technologies, 2003. Disponível em: <http://www.tagrimm.com/publications/perspectives-jan2003.html>. Acesso em 08 dez. 2010.

YAN, X.; GU, P. **A review of rapid prototyping technologies and systems.** Computer Aided Design, Vol. 28, No. 4, pp. 307-318, 1996.

GLOSSÁRIO

3DP - Three Dimensional Printing

API - Application Programming Interface

ASCII - American Standard Code for Information Interchange

CAD - Computer-Aided Design

CAM - Computer-Aided Manufacturing

CNC - Controle Númérico Computadorizado

EMC - Enhanced Machine Controller

FDM - Fused Deposition Modeling

HD - Hard Disk

LM - Layered Manufacturing

LOM - Laminated Object Manufacturing

OMG - Object Management Group

PC - Personal Computer

PID - Proporcional Integral Derivativo

PME - Pequenas e Médias Empresas

SFF - Solid Freeform Fabrication

SLA - Stereolithography Apparatus

SLS - Selective Laser Sintering

SNAP - Scaleable Node Address Protocol

STL - Stereo Lithography (STL)

UML - Unified Modeling Language

VRML - Virtual Reality Modeling Language