UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
TRABALHO DE CONCLUSÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Fem2014 - Software Multiplataforma para Análise de Campos Elétricos e Magnéticos 2D baseado no Método dos Elementos Finitos

Autor: Elizandro G. Roos

Orientador: Prof. Luís Alberto Pereira

Porto Alegre, dezembro de 2014

Resumo

O trabalho aborda o desenvolvimento de uma nova versão do sistema Fem2000, um software de análise de campos eletromagnéticos baseado no Método dos Elementos Finitos. O Fem2000 é um sistema utilizado como ferramenta de apoio nas aulas de máquinas e dispositivos eletromagnéticos, além do seu emprego em pesquisas e projetos. Entretanto, a versão atual funciona somente em plataforma Windows e utiliza ferramentas de software proprietário. Além disso o sistema apresenta baixo desempenho gráfico e numérico, principalmente na manipulação de modelos de dados muito complexos. Propôs-se a implementação de uma versão baseada em software livre, multiplataforma, e uma análise de otimizações no código a fim de tornar o sistema mais robusto. A nova versão implementada roda em sistemas Windows e Linux, possui interface aprimorada, e possui nova opção de algoritmo de geração de malha triangular. O novo algoritmo apresentou melhor desempenho, principalmente devido ao menor número de elementos da malha gerada.

Palavras-chaves: MEF, multiplataforma, triangulação, .

Abstract

This work presents a new version of the Fem2000 system, a Finite Element Method software for electromagnetic field analysis. Fem2000 is a tool used in the field of academic research, as well as in part of electromagnetic devices class. However, it only works on Windows systems and uses proprietary software. Moreover, the system presents low graphical and numerical performance, especially when working with complex data models. The proposed implementation is based on open source e cross-platform software libraries, like OpenGL and wxWidgets. The new version implemented runs on Windows and Linux systems, has improved interface, e faster graphical response. In addition, a faster algorithm for triangular mesh generation was implemented.

Key-words: FEM. cross-platform. .

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

MEF	Método dos Elementos Finitos (*Finite Element Method*).
API	Interface de programação de aplicações (*Application Programming Interface*).
PSLG	Grafo Planar de Segmentos de Retas (*Planar straight-line graph*).

Sumário

1 Introdução

Atualmente, em um contexto de rápido avanço na capacidade de cálculo dos computadores, diversos métodos numéricos são utilizados na solução de problemas científicos. Particularmente, o estudo de campos eletromagnéticos, assunto essencial na física e engenharia, na maior parte dos casos práticos envolve geometrias complexas onde muitas vezes não é possível o cálculo analítico. O Método dos Elementos Finitos (MEF) é o mais utilizado nesses casos, gerando soluções aproximadas suficientemente acuradas para os propósitos da engenharia (SADIKU, 2009).

A determinação dos campos é essencial no dimensionamento e otimização de dispositivos eletromagnéticos (PEREIRA, 2000). Logo, uma ferramenta computacional adequada, que ofereça recursos de cálculo e simulação, é fundamental para o aprendizado e para a redução de custos e tempo em projetos e pesquisas. Existem diversos softwares no mercado para suprir essa demanda, no entanto, em sua maioria são sistemas comerciais de custo elevado que requerem licenças anuais.

De autoria do professor Luís Alberto Pereira, o software Fem2000 é um sistema de análise de campos eletromagnéticos que utiliza o MEF como método de cálculo. Ele vem servindo como base para trabalhos científicos em nível de graduação e pós-graduação, e também é utilizado como ferramenta de apoio nas aulas da disciplina de máquinas elétricas. Entretanto, o software atual possui alguns problemas e limitações, como baixo desempenho gráfico. Surgiu então a necessitando de manutenção e adequação às tecnologias mais modernas, além de rodar somente em Windows.

Devido à essas e outras limitações, surgiu a necessidade de uma versão mais moderna, independente de plataforma e baseada em software livre. Felizmente, hoje existem algumas bibliotecas de *software* disponíveis na Internet para essa finalidade, como a GTK+, Qt e wxWidgets, sendo a última a escolhida para o presente trabalho. A biblioteca wxWidgets possui diversos componentes para interface gráfica e outras funções, e tem foco em auxiliar o desenvolvedor na criação e manutenção de sistemas portáveis. Além disso, conta com ampla documentação e suporte pela comunidade de *software* livre.

Também será utilizada a API livre OpenGL (Open Graphics Library), que é um padrão utilizado há mais de duas décadas na computação gráfica para desenvolvimento de aplicativos gráficos, ambientes 3D, jogos, entre outros. Essa API padroniza a interface de software para desenho gráfico 2D e 3D, permitindo acessar os recursos de hardware das placas gráficas existentes nos computadores atuais.

1.1 Justificativa

O sistema Fem2000 software atual utiliza componentes e recursos da proprietários da biblioteca de classes *Microsoft Foundation Classes* (MFC) e, por isso, roda somente em ambiente Windows. Além disso, o ambiente de desenvolvimento (IDE) Visual Studio da Microsoft necessita de licenças anuais. Já, as tecnologias baseadas em software livre são mais atrativas em relação a esses aspectos, pois além de serem livre de custo, tendem a ser independentes de plataforma (MENG, 2003).

Outra limitação do sistema atual é o baixo desempenho gráfico devido ao uso de rotinas que não são adequadas para processamento intensivo. A manipulação de desenhos de peças de geometrias complexas, prejudica a usabilidade do programa. Além disso, o processamento numérico requerido para a geração de malhas triangulares, etapa automática e fundamental na análise de elementos finitos, utiliza um algoritmo pouco eficiente. Em determinados casos são necessários minutos para o término do processo.

1.2 Objetivos

O objetivo principal do trabalho é o desenvolvimento de um sistema multiplataforma para análise de campos eletromagnéticos em duas dimensões, baseado no sistema Fem2000. Os principais objetivos do trabalho enfocam a utilização de Software Livre, portabilidade do código, desempenho e usabilidade do novo sistema. A seguir são detalhadas as características do sistema novo:

- alta performance gráfica: desenhos com geometria complexa, com muitos segmentos, podem causar atrasos significantes ao atualizar a tela. Propõe-se o uso de rotinas gráficas do padrão OpenGL, a fim de utilizar recursos de aceleração gráfica por hardware;
- melhorias na etapa de geração da malha: a geração da malha triangular é a etapa do Fem2000 que mais demanda processamento computacional. Assim, a implementação de um novo gerador de malhas baseado em um algoritmo de refinamento de melhor desempenho é uma das principais necessidades;
- portabilidade: o novo sistema deve rodar em Windows, Linux e MacOS, e ser implementado com código portável, de modo a facilitar a manutenção do sistema. Deve também utilizar bibliotecas e algoritmos de código livre;
- interface aprimorada: o sistema deve ser de fácil utilização pelo usuário, tendo uma curva de aprendizado com recursos comumente encontrados em sistemas modernos similares. Como exemplo, serão implementadas os seguintes recursos: *zoom* aprimorado, histórico de comandos (desfazer/refazer), área de transferência (copiar/colar), impressão, ajuda, entre outros;

- implementar um novo gerador de malhas.

1.3 Organização do Trabalho

O capítulo 2 apresenta uma breve revisão teórica sobre os assuntos relevantes para o trabalho. São apresentadas algumas definições sobre malhas triangulares, o método dos elementos finitos, e algoritmos sobre geração de malhas triangulares. O capítulo 3 descreve o funcionamento geral do software, destaca os novos recursos da interface gráfica, descreve as estruturas de dados e algoritmos utilizados, bem como peculiaridades em relação a portabilidade do sistema para diferentes plataformas. O capítulo 4 descreve brevemente os testes que foram feitos. O capítulo 5 apresenta resultados através de tabelas e figuras apresentando as diferenças entre o sistema atual e a nova versão do software.

2 Revisão Bibliográfica

Neste capítulo são apresentadas

2.1 Método dos Elementos Finitos

O Método dos Elementos Finitos - MEF - encontra soluções aproximadas para equações diferenciais parciais (PDE) e também equações integrais. Segundo Sadiku (2009), o MEF consiste em quatro passos:

- discretização da região da solução em um número finito de subregiões, ou elementos;
- determinação das equações relevantes ao problema para cada elemento;
- união das equações de todos elementos da região total;
- resolução do sistema de equações obtido.

Quando implementado na forma de um sistema computacional, o processo de obtenção da solução pode ser dividido nas seguintes etapas (JIN, 1993):

2.1.1 Pré-processamento

Inicialmente, é necessário entrar com os dados geométricos do problema, subdividir a geometria e especificar as características de cada parte, como permeabilidade magnética ou elétrica, corrente, material. Após, devem ser escolhidas as condições de contorno, podendo-se citar as duas mais utilizadas: Dirichlet e Neumann. Nessa etapa também são escolhidas as características da malha de elementos a ser gerada. A discretização então é feita automaticamente pelo programa, subdividindo a geometria em elementos menores, tais como triângulos ou retângulos. O elemento 2D mais utilizado é o triângulo, devido a sua simplicidade de implementação. Esse é o tipo de elemento utilizado no Fem2000.

2.1.2 Processamento

Nesta etapa é realizado o cálculo da solução nos nós da malha. O cálculo pode ser aproximado por uma função de aproximação da grandeza principal (potencial do campo magnético ou elétrico), que pode ser linear ou não-linear. A solução do sistema de equações lineares é em geral realizada através de métodos iterativos até que o erro esteja dentro do valor estipulado pelo usuário.

2.1.3 Pós-processamento

Esta etapa envolve a representação gráfica das grandezas de interesse através de linhas equipotenciais e/ou escala de cores. As principais grandezas são o campo magnético e o

campo elétrico.

2.2 Geração de malhas triangulares

Primeiramente, apresentam-se algumas definições geométricas que são utilizadas nesta seção:

- Circuncírculo: o circuncírculo de um triângulo é o círculo único que passa sobre os três vértices de um triângulo;
- Círculo diametral: é o menor círculo que engloba um segmento;
- PSLG: grafo Planar de Segmentos de Retas é um conjunto de segmentos e pontos no qual os segmentos se cruzam somente em suas extremidades (do inglês *Planar straight-line graph*)
- Triangulação: uma triangulação de um conjunto de pontos é um caso específico de PSLG. No plano bidimensional, é uma subdivisão de uma geometria em triângulos.

A geração de malha é uma etapa crucial no MEF, pois a qualidade da malha interfere diretamente na precisão da solução numérica obtida. Uma malha possui vários requerimentos normalmente difíceis de serem conciliados: deve ser compatível à geometria do objeto; não deve ter um número muito grande de elementos nem elementos muito grandes; o tamanho dos elementos pode variar em curtas distâncias; e os elementos não devem ter ângulos muito grandes ou pequenos. Particularmente no MEF, deve-se evitar ângulos muito grandes, pois acarretam em grande erro na aproximação da solução (SHEWCHUK, 2012).

Existem diversas técnicas para geração de malhas, porém para o MEF os algoritmos mais utilizados se baseiam em malhas triangulares obtidas através do algoritmo de Delaunay. Uma triangulação Delaunay tem a seguinte definição: seja um conjunto de pontos S, uma triangulação T é Delaunay se e somente se nenhum ponto de S está no interior de qualquer circulo-circundante de um triângulo de T (Figura 1).

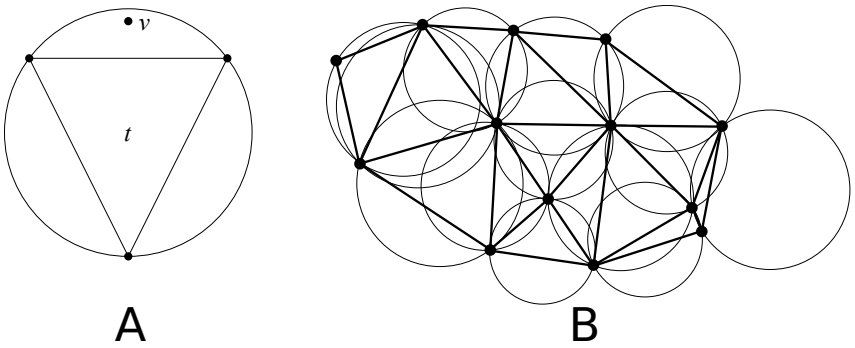


Figura 1 – A: O triângulo t não é Delaunay devido ao ponto v no interior de seu circuncírculo. B: Uma triangulação Delaunay. Fonte: Shewchuk (2012).

A qualidade de uma triangulação pode ser estimada segundo diferentes critérios (SHEWCHUK, 1996; YVINEC, 2009). Para Shewchuk a relação raio-aresta r/l_{min} é a medida mais apropriada de análise da qualidade dos algoritmos de refinamento Delaunay, onde r é o raio do circuncírculo e l_{min} a menor aresta de um triângulo. Essa medida deve ser minimizada, e se relaciona com o menor ângulo de acordo com a equação:

$$\frac{r}{l_{min}} = \frac{1}{2\sin\theta_{min}} \tag{1}$$

A principal operação dos algoritmos de refinamento Delaunay é a inserção de um ponto no circuncentro de um triângulo de má qualidade. Um dos algoritmos mais utilizados é o de Ruppert, descrito na próxima seção.

2.2.1 Algoritmo de Ruppert

A partir de uma triangulação Delaunay como entrada, o algoritmo de Ruppert adiciona vértices até que todos triângulos satisfaçam as restrições de qualidade e tamanho escolhidas. Alguns vértices são inseridos ao centro dos circuncírculos, e outros sobre segmentos, subdividindo-os. A partir de uma malha inicial Delaunay, o algoritmo iterativamente refina a malha subdividindo os segmentos inválidos e retriangulando as porções afetadas da malha. As duas operações principais, segundo a ordem de prioridade, são:

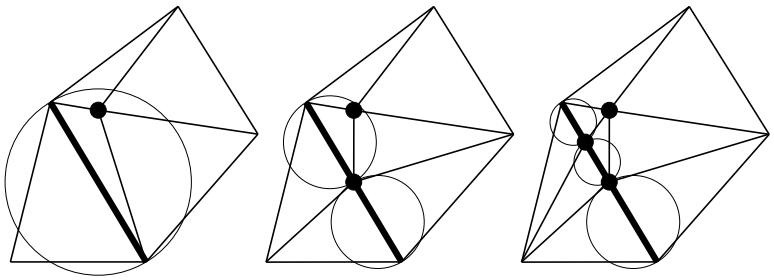


Figura 2 – Subdivisão recursiva de segmentos. Fonte: Shewchuk (2012).

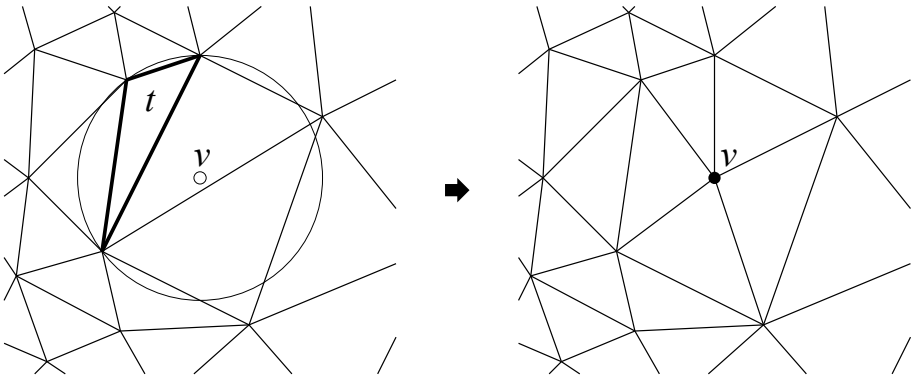


Figura 3 – Um ponto é inserido no circumcentro de t. O espaço criado pela remoção dos triângulos é preenchido por novos triângulos. Fonte: Shewchuk (2012).

a) Os segmentos inválidos são subdivididos ao meio, criando 2 novos subsegmentos e 1 novo ponto. Os novos subsegmentos tem círculos diametrais menores e também podem ser inválidos. A subdivisão continua até que todos segmentos e triângulos sejam válidos (Figura 2).

b) Cada triângulo com relação raio-aresta menor que um limite pré-estabelecido é eliminado através da inserção de um ponto em seu circumcentro, a não ser que isto torne algum de seus segmentos inválidos. Nesse caso, os segmentos que forem inválidos são subdivididos através da regra anterior. Se o triângulo for removido, uma "cavidade" surge na malha, e novos triângulos são inseridos no lugar conectando o ponto inserido às arestas da cavidade criada (Figura 3).

Essas operações são repetidas enquanto existirem segmentos inválidos ou triângulos fora das condições de qualidade estabelecidas. Sem modificações o algoritmo tem garantia de término para PSLG's de entrada que não tenham ângulos agudos e para uma condição de ângulo mínimo de até $20,7°$. O pseudo-código a seguir descreve o algoritmo de Ruppert.

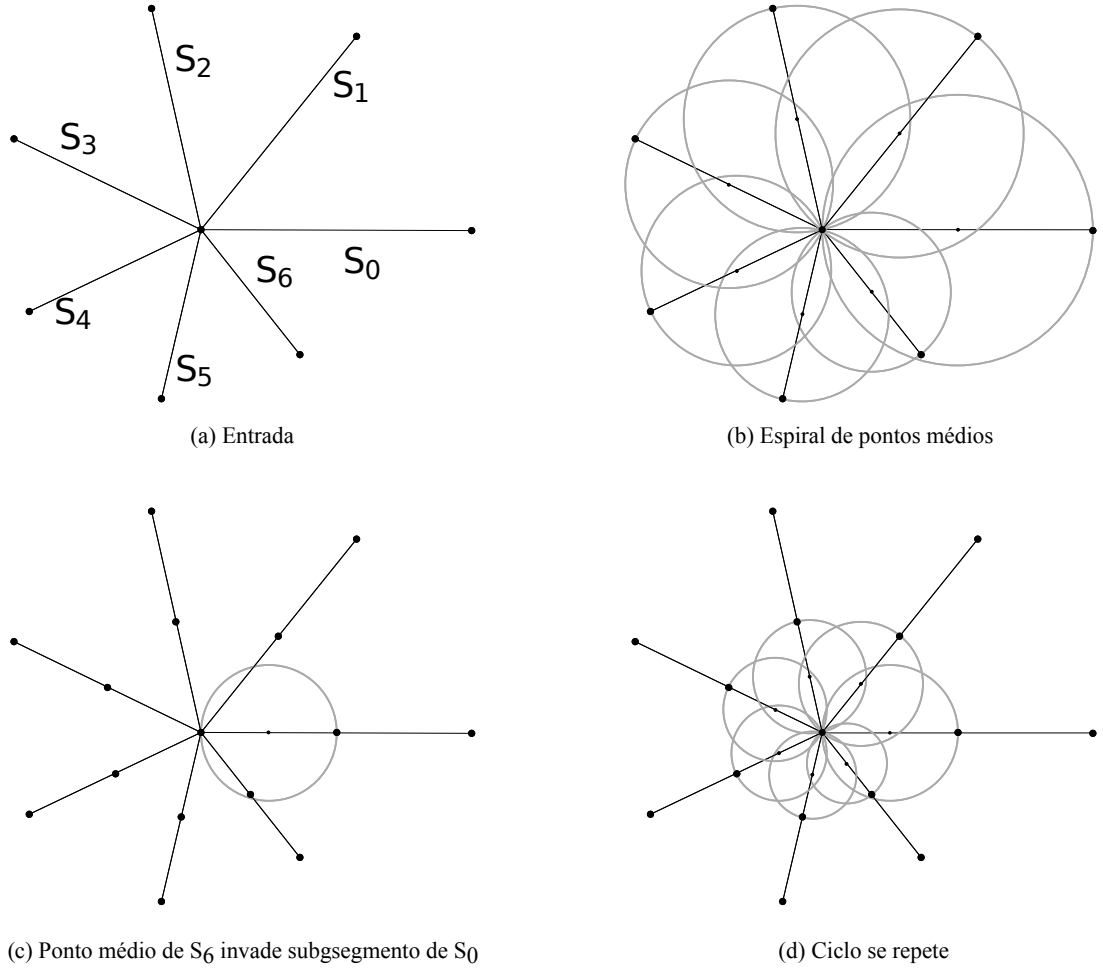
Entrada: lista de pontos Lp e segmentos Ls
Saída: Malha refinada
CalculaTriangulaçãoDelaunay(Lp)
enquanto *existe segmento S inválido ou triângulo T de má qualidade* **faça**
  se *algum ponto invade invade algum segmento S* **então**
    DivideSegmento(S)
  senão
    $P_c \leftarrow$ CalculaCircumcentro(T)
    se *P_c invade algum segmento da vizinhança* **então**
      DivideSegmento(S)
    senão
      insere circumcentro de t em D
fim

Algoritmo 1: Algoritmo de Ruppert

2.2.2 Algoritmo de Pav

O algoritmo proposto por Pav (2003) é uma variação do algoritmo de Ruppert, porém mais robusto, com a finalidade de permitir ângulos pequenos na geometria de entrada. É feita uma verificação dos ângulos nos dados de entrada, e a marcação de pontos e segmentos onde possam ocorrer problemas de recursividade infinita ou geração de elementos de má qualidade.

A Figura 4 ilustra um caso de recursão em espiral. Os comprimentos dos segmentos formam uma progressão geométrica, e após a subdivisão dos segmentos em seus pontos médios, ocorre apenas uma redução de escala do desenho, de modo que tem-se um ciclo infinito.



(a) Entrada

(b) Espiral de pontos médios

(c) Ponto médio de S_6 invade subgsegmento de S_0

(d) Ciclo se repete

Figura 4 – Caso de recursividade infinita. Fonte: Pav (2003)

2.3 Estruturas de dados

No campo da computação, uma estrutura de dados é um modo particular de organizar dados em um computador, normalmente em memória, de modo a permitir sua eficiente utilização. Alguns tipos são mais adequados para certos tipos de aplicação, sendo algumas estruturas bastante especializadas. A escolha do tipo de estrutura depende do tipo de dado a armazenar, memória disponível e desempenho dos algoritmos de manipulação dos elementos de dados, como busca, inserção e remoção de elementos. Um exemplo de estrutura é a lista encadeada simples, em que os dados são alocados dinamicamente em memória, e ligados através de ponteiros.

Esse tipo de estrutura é utilizada no sistema Fem2000 para armazenar a geometria do desenho e da malha, sob a forma de pontos, retas, arcos e triângulos. Para pequenos conjuntos de dados ou para sistemas que apenas necessitem de busca sequencial esse método é adequado para uso nos computadores modernos atuais, porém esse não é o caso deste trabalho, pois várias rotinas exigem acesso aleatório aos dados. A busca nesses casos é lenta pois o algoritmo deve varrer a lista elemento a elemento até encontrar o dado

procurado.

Um método mais apropriado de armazenagem de dados para acesso aleatório são as estruturas em árvore. Diferentemente das listas encadeadas, em que os dados se encontram numa sequência, em uma estrutura em árvore os dados simulam uma estrutura hierárquica por meio de nós ligados entre si. Cada nó da estrutura pode ter um dado ou uma condição, para facilitar a busca. Cada nó pode ter um nó pai, ou nenhum se ele for o primeiro nó, ou nó raiz; e nenhum ou vários filhos. Os nós finais, os que não possuem filhos, são denominados folhas.

2.4 Interface Gráfica do Usuário

A interface é uma parte essencial de um sistema de computador que influencia diretamente na usabilidade do mesmo. Através de elementos gráficos, como botões, menus e janelas, permite o acesso as funções oferecidas pelo programa. Atualmente existem várias bibliotecas que facilitam a implementação de interfaces gráficas. Foi escolhida a biblioteca de componentes wxWidgets, por atender aos seguintes critérios de projeto:

- Multiplataforma;
- suporte à linguagem C/C++, e outras linguagens;
- acesso livre ao código;
- interface gráfica nativa.

As diversas funções disponíveis possuem grande similaridade na sintaxe em relação as rotinas MFC da Microsoft, o que facilita o processo de adaptação do código do atual sistema para o novo toolkit. A maior parte das funcionalidades do sistema nativo são mantidas, pois a implementação do wxWidgets se baseia em sua maior parte em chamadas a rotinas do sistema operacional. Assim, um sistema implementado em wxWidgets tem *look and feel*[1] nativo do sistema operacional em que está executando.

Além da interface gráfica, o wxWidgets facilita o desenvolvimento de diversas outras funções comumente utilizadas em sistemas modernos, mantendo o foco na portabilidade do código. Dentre elas pode-se citar: manipulação de arquivos e *streams*, *threads*, arquivos de configuração, comunicação, *help*, acesso a banco de dados, e outros.

2.5 OpenGL

O OpenGL é uma interface para acessar ou controlar o subsistema gráfico do dispositivo em que está executando. A padronização aumenta a portabilidade, permitindo ao

[1] Termo utilizado para definir um padrão relacionado à aparência e aspectos funcionais de uma interface.

desenvolvedor se concentrar em aspectos importantes da aplicação sem se preocupar com as especificidades das diferentes plataformas que deseja que a sua aplicação seja compatível (WRIGHT; LIPCHAK; HAEMEL, 2013).

O padrão OpenGL não especifica ou implementa diretamente as janelas da GUI, deixando esse trabalho a cargo de outras bibliotecas do subsistema gráfico do sistema operacional. É uma interface padronizada Cada ambiente de janelas tem uma biblioteca de extensão do OpenGL. Em sistemas Unix e Linux, o sistema X possui o GLX, enquanto o Windows fornece a extensão WGL, já o Mac OS X fornece o CGL. No entanto isto é transparante para o programador, pois estas bibliotecas já vêm instaladas com o sistema operacional, ou distribuídas juntamente com a implementação do OpenGL. Em plataformas Linux a biblioteca Mesa 3D é uma implementação de código aberto do OpenGL, podendo utilizar recursos de software ou hardware. No Windows os fabricantes de placa de vídeo disponibilizam a implementação em conjunto com os drivers do dispositivo.

O OpenGL possui três primitivas geométricas básicas de desenho - pontos, linhas ou triângulos - que podem ser combinadas para formar vários tipos de geometria 2D e 3D. A primitiva mais básica é o vértice, que pode especificar um ponto, extremidade de linha, ou o vértice que une duas linhas. A especificação de coordenadas de um vértice é feita através da rotina *glVertex*.

Os métodos *glBegin(modo)* e *glEnd* delimitam o início e o fim, respectivamente, de um trecho de código de desenho, onde podem ser especificadas uma ou mais primitivas do mesmo tipo. Entre esses duas chamadas devem ser especificados os vértices através de chamadas ao método *glVertex*. De acordo com o argumento *modo* os vértices especificados são conectados formando o desenho. Através do modo GL_POINTS são especificados apenas pontos, já com os modos GL_LINES e GL_TRIANGLES, os vértices são conectados formando respectivamente linhas e triângulos. Há também o modo GL_TRIANGLE_STRIP, que gera triângulos em sequência, necessitando menor número de vértices. Para cada triângulo apenas um vértice é necessário, com exceção do primeiro triângulo que é formado por 3 vértices. Para N vértices, são gerados $N - 2$ triângulos.

```
glBegin(GL_TRIANGLE_STRIP);
glVertex2f(0., 0.);
glVertex2f(10., 0.);
glVertex2f(10., 10.);
glVertex2f(0., 10.);
glEnd();
```

É possível conectar os vértices de forma a formar um polígono através do modo GL_LINE_LOOP. Cada vértice é conectado ao seguinte, na sequência em que se encontram no código. Ao final da lista, o último vértice é conectado ao primeiro fechando um *loop*. O código a seguir gera uma quadrado com lado de 10 unidades.

```
glBegin(GL_LINES_LOOP);
glVertex2f(0.0, 0.0);
glVertex2f(10.0, 0.0);
glVertex2f(10.0, 10.0);
glVertex2f(0.0, 10.0);
glEnd();
```

A função *glVertex2f* aceita argumentos do tipo ponto flutuante para especificar as coordenadas 2D dos vértices. No exemplo anterior, cada par de chamadas consecutivas da função *glVertex2f* determina uma linha a partir de dois vértices. As rotinas em OpenGL possuem diversas variações de acordo com o tipo de argumento (*float*, *int*, e outros) e o tipo de coordenada (2D ou 3D). No exemplo anterior, o sistema de coordenadas é em duas dimensões e o tipo de dado *float*.

No próximo capítulo são apresentados aspectos de implementação prática dos conceitos apresentados aqui.

3 Ferramentas e Métodos

3.1 Ambiente e ferramentas de desenvolvimento

Uma das premissas do novo sistema é a utilização de software livre. Seguindo essa idéia foi escolhido o CodeBlocks, um ambiente integrado de desenvolvimento (IDE) multiplataforma e livre de custos. O editor de código possui diversos recursos modernos comumente encontrados em IDE's, como: auto completar código em C++, realce de sintaxe (*syntax highlight*), busca e substituição de texto, busca por métodos e variáveis. Quanto ao gerenciamento de projetos é bastante flexível, pois além de gerenciar os arquivos de código fonte do projeto, permite a configuração de diferentes perfis de compilação. A curva de aprendizado do ambiente é rápida, sendo recomendado também para iniciantes. A instalação do software tem a opção de instalar também o MinGW (*Minimalist GNU for Windows*), um conjunto de ferramentas de compilação que contém uma implementação do compilador GNU C++, debugador GDB, e outros programas necessários para compilação.

3.1.1 Otimização de código

Utilizou-se a ferramenta de análise dinâmica de código *gprof*, que faz parte do grupo de ferramentas binárias GNU, juntamente com o compilador GCC e o debugador GDB. Primeiramente compila-se o código com a opção *-p* do GCC, e após, executa-se o programa normalmente. Um arquivo é gerado automaticamente com dados de *profile*, ou seja, um perfil de consumo dos recursos computacionais, que pode ser analisado pela ferramenta *gprof*.

3.2 Estrutura do sistema

O software Fem2000 é organizado em 3 módulos principais: o módulo de edição de desenho, onde o usuário entra com os dados geométricos do problema da geometria a ser analisada. No módulo de desenho é feita a definição geométrica, enquanto as características de material são escolhidos no módulo de dados. No módulo de malha podem ser visualizados diferentes tipos de soluções disponibilizadas pelo programa, tais como linhas equipotenciais, gráficos numéricos, valores pontuais de força, fluxo magnético, entre outros.

As principais modificações feitas no software se encontram no módulo de desenho, e no gerador de malhas. Um maior detalhamento dos módulos é feito nos próximos itens, destacando os aprimoramento do novo sistema em relação ao Fem2000.

3.2.1 Editor de desenho

Este módulo é um editor de desenho onde o usuário pode entrar com os dados geométricos da peça a ser analisada. São oferecidas ferramentas básicas de desenho como inserção de linha, retângulo, arco, círculo e cópia. A entrada de dados pode ser via mouse ou teclado.

Há também a opção de importação de dados de outros sistemas em formatos DXF, um padrão de arquivo de geometria bastante utilizado em software como o AutoCad e Solidworks. Isso permite que peças muito complexas que exijam um editor com mais recursos possam ser importadas.

3.2.1.1 Histórico de comandos

Um dos recursos acrescentados ao sistema é o histórico de comandos. Esse recurso salva uma lista de comandos recentemente executados pelo usuário. Esse é um recurso encontrado na maioria das aplicações que de alguma forma trata com edição de dados, como editores de texto, editores gráficos, CAD, entre outros. Através do menu ou pelo teclado através de teclas de atalho, o usuário pode desfazer e refazer ações executadas. Os comandos desfazer e refazer são usualmente ativados pelas combinações de teclas Ctrl+Z e Ctrl+Y, respectivamente, podendo mudar de acordo com as configurações específicas da plataforma.

A implementação se baseou em duas classes disponibilizadas pela biblioteca wxWidgets que auxiliam o desenvolvedor nesse tipo de tarefa: *wxCommandProcessor* e *wxCommand*. A primeira implementa um processador de comandos, que mantém o histórico dos últimos comandos executados (tamanho configurável). E a segunda é uma classe abstrata, que serve como base para modelar as ações a serem executadas na aplicação. Quando há uma ação, ela deve ser executada indiretamente por meio do processador de comandos. As ações devem ser encapsuladas em um objeto da classe apropriada e passado como argumento para o método *wxCommandProcessor::Submit*. O *framework* se encarrega dos detalhes do gerenciamento do histórico e do acesso aos comandos desfazer/refazer que são apresentados na interface por meio de itens no menu da aplicação. Na classe filha criada, os métodos virtuais *Do* e *Undo* devem ser implementados, pois o processador invoca esses métodos na execução das respectivas ações. O método *Undo* deve executar as ações exatamente contrárias àquelas do método *Do*. É de responsabilidade do programador a correta manipulação das estruturas de dados relacionadas aos comandos executados. A Figura 5 mostra o menu editar da aplicação executando em ambiente Linux.

3.2.1.2 Configurações

A classe *wxFileConfig* possui métodos que auxiliam o gerenciamento de arquivos de configuração. São disponibilizados métodos que permitem a manipulação de dados de

Figura 5 – Opções do histórico de comandos

arquivo de configuração, como *Read* e *Write* que permitem recuperar e salvar dados.

3.2.2 Vista de dados e propriedade das regiões

Esse módulo permite o ajuste da malha e das características de material da peça. No exemplo da Figura 6, é apresentada uma peça que representa um corte transversal de parte de um conjunto rotor e estator. Pode-se ver uma região que representa uma bobina com corrente elétrica, em vermelho, e o núcleo de ferro, em verde. A cada região do desenho são atribuídas as características do material como: permeabilidade magnética, corrente, indução, curva de magnetização (no caso de material não-linear). A Figura 7 apresenta a tela de interface onde o usuário pode entrar com esses dados.

3.2.3 Módulo de análise de resultados

Nesse módulo são feitos executados os cálculos e a visualização dos resultados sob a forma de gráficos e valores numéricos pontuais, de acordo com a escolha do usuário. São disponibilizadas diversas opções, como cálculo de correntes induzidas, visualização de equipotenciais, força sobre uma seção, etc. A Figura 8 mostra um exemplo de visualização de linhas equipotenciais.

3.3 Novo gerador de malhas

A principal limitação do gerador de malhas do sistema Fem2000 é que os elementos da malha gerada são aproximadamente do mesmo tamanho. Isso é desvantajoso na maioria dos casos, pois geralmente há regiões do desenho que necessitam maior precisão que outras.

Utilizando-se algoritmos como o de Ruppert, pode-se reduzir o número de elementos gerados sem prejuízo da qualidade do resultado. Entretando, quando os dados de entrada violam as restrições de ângulo (ângulos não podem ser muito pequenos), o algoritmo não tem garantia de término, podendo entrar em recursividade infinita.

O algoritmo proposto, apresentado a seguir, baseado no algoritmo de Pav (2003, pg.42), detecta os casos críticos de ângulo automaticamente, e assegura a parada do algoritmo mesmo nesses casos. A malha gerada possui elementos de tamanho variáveis, se ajustando

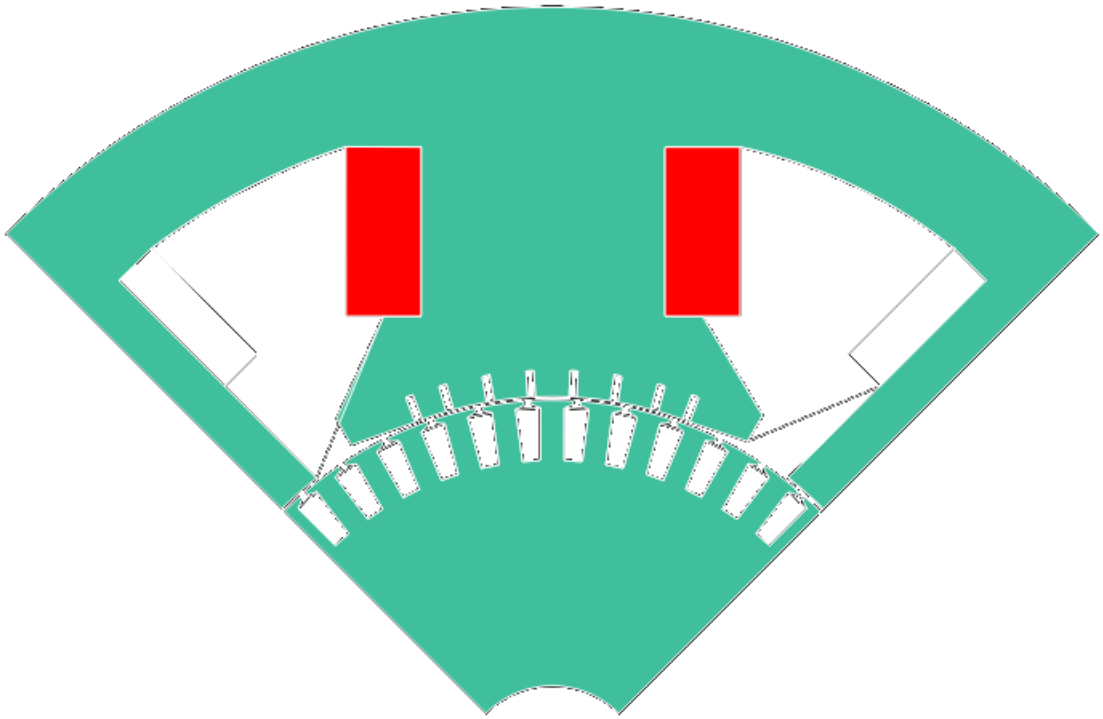


Figura 6 – Desenho em corte de rotor e estator de máquina de corrente contínua. As áreas em vermelho representam bobinas com corrente elétrica e, em verde, um material com alta permeabilidade magnética, como o ferro.

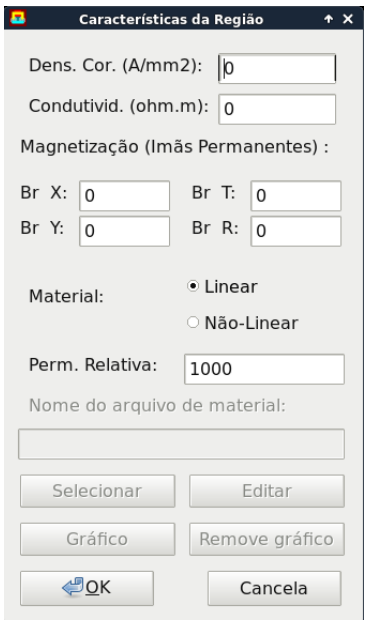


Figura 7 – Tela de edição de características de material.

à geometria do problema. São criados elementos menores onde os pontos e segmentos estão mais próximos uns dos outros.

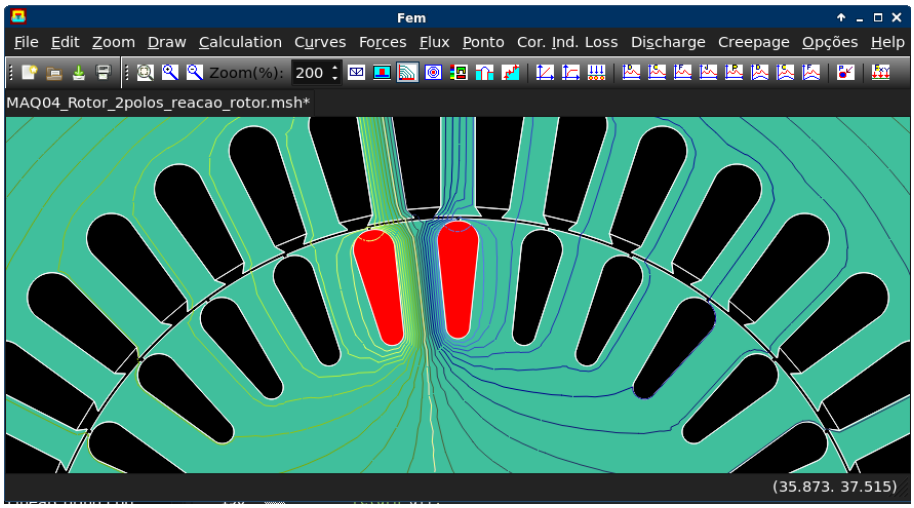


Figura 8 – Linhas equipotenciais.

Primeiramente é criada uma malha que inicia com a criação de um triângulo que envolva os pontos e segmentos de entrada. Então, a cada passo, os pontos do conjunto de entrada são adicionados um a um, verificando-se quais triângulos se tornam inválidos. Os triângulos que forem inválidos são removidos, e a porção afetada da malha é retriangulada.

3.3.1 Metodologia de teste

A fim de analisar o desempenho e funcionalidade do sistema foram utilizados alguns casos de uso típicos do programa. Fez-se algumas comparações entre a versão atual e a nova versão do software em relação as malhas geradas, levando-se em conta a qualidade e o desempenho. O critério de qualidade utilizado nos testes a maioria dos testes é de 24°, o que equivale a uma relação raio-aresta de aproximadamente $r/l_{min} = 1.229$ (Eq. 1). Os dois geradores de malha são bastante distintos, tornando difícil a comparação entre os dois. Nos testes foram escolhidos parâmetros de modo que as malhas apresentassem números de elementos similares.

Entrada: ListaPontos Lp, ListaSegmentos Ls,
Saída: Malha M refinada
`DelaunayPav();`
Triangulo T;
Segmento S;
Ponto P;
Preprocessa(Lp, Ls);
enquanto *algum S é invalido ou algum T inválido* **faça**
    enquanto *algum T é inválido* **faça**
        T = EncontraTrianguloInvalido();
        P = CalculaCircumCentro(T);
        DivideSegmentosInvalidos(P);
        se *não dividiu segmentos* **então**
            RetiraCavidade(P);
            PreencheCavidade(P);
    fim
    InsereSegmentosFaltantesMalha()
fim
// Impõe as fronteiras internas dividindo ao meio os segmentos
// que não estão na malha
`InsereSegmentosFaltantesMalha`
se *encontrar* $S : S \in Ls \ e \ S \notin M$ **então**
    Divide S P_{medio};
    gerando os subsegmentos $S_1 e S_2$;
    Insere P_{medio} em Lp;
    Remove S de Ls;
    Insere $S_1 e S_2$ em Ls;
    RetiraCavidade(P_{medio});
    PreencheCavidade(P_{medio});
fim

Algoritmo 2: Algoritmo de Pav

4 Resultados e Análise

A seguir será feita uma análise de alguns casos de teste, em que são demonstradas as principais funcionalidades do sistema. São verificadas algumas condições em que podem representar problemas na geração de malha. Os resultados apresentam comparações entre os dois geradores de malha. As variáveis de controle de geração da malha são ajustadas pelo usuário; elas tem influência na qualidade da malha obtida e no tempo de processamento.

4.1 Caso de uso típico

O primeiro teste é bastante utilizado nas aulas de máquinas e dispositivos eletromagnéticos. O desenho da Figura 9 ilustra uma peça de ferro (em cinza claro) com uma parte móvel que pode ser trocada de posição, e uma bobina (cinza escuro) que é percorrida por corrente elétrica. Um campo magnético é gerado, se concentrando na peça de ferro, por possuir maior permeabilidade magnética. Conforme pode ser visto na Figura 10, para este caso, a malha fica mais detalhada em torno do entreferro, que é onde há maior concentração de pontos.

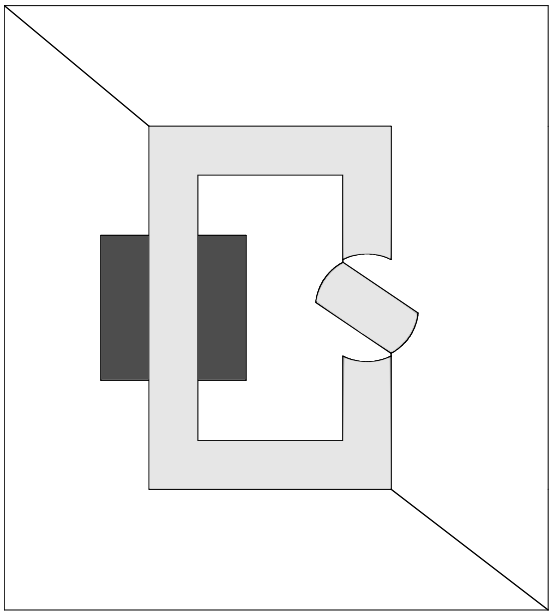


Figura 9 – Caso 1: Geometria da peça.

4.2 Alguns casos ilustrativos

Esse teste tem o objetivo de verificar a eficiência e a qualidade da malha obtida pelo novo malhador. Os dois geradores de malha foram executados para a geometria apresentada na Figura 9, utilizando parâmetros similares, de modo que se possa comparar padrões. As

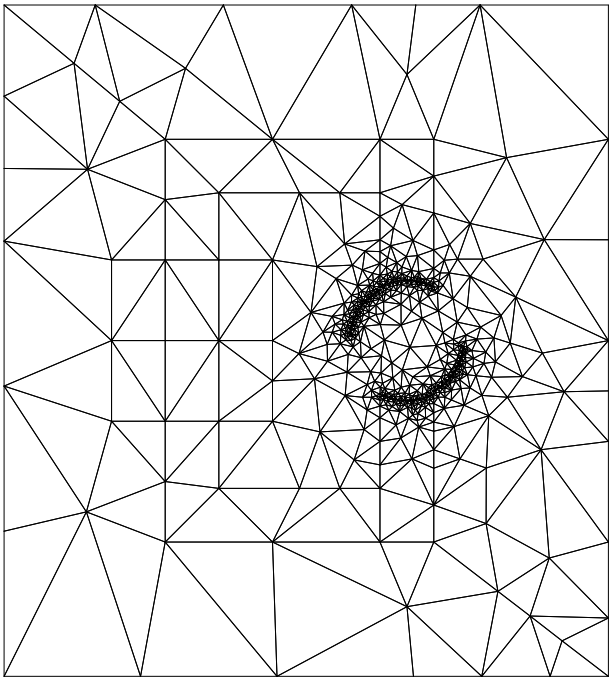


Figura 10 – Malha gerada pelo novo algoritmo.

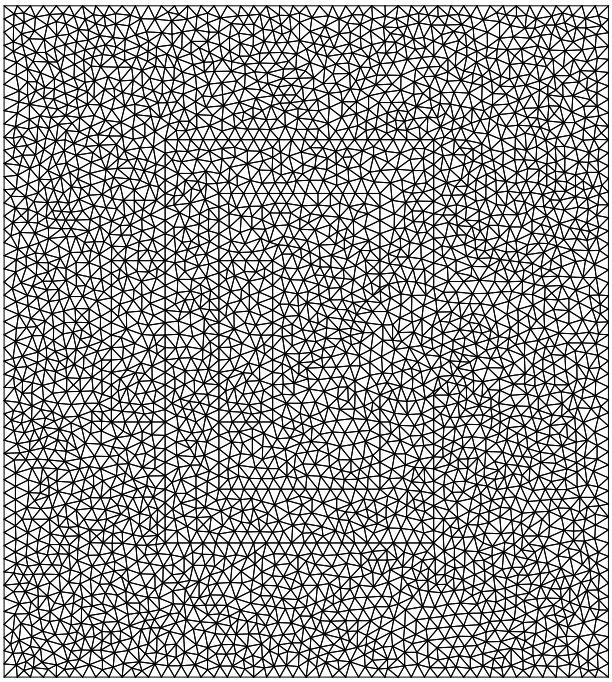


Figura 11 – Malha gerada pelo Fem2000.

figuras 13 e 14 apresentam o desenho das malhas geradas pelo algoritmo atual e o novo algoritmo implementado, respectivamente. A Tabela 1 apresenta resultados numéricos para algumas características da malha gerada pelos dois algoritmos.

O terceiro caso de teste envolve malha com mais de 10 mil elementos. As figuras 15 e 16 são as malhas geradas pelo Fem2000 e pelo novo algoritmo no Fem2014, respectivamente. A Tabela 2 apresenta os dados obtidos pelos algoritmos.

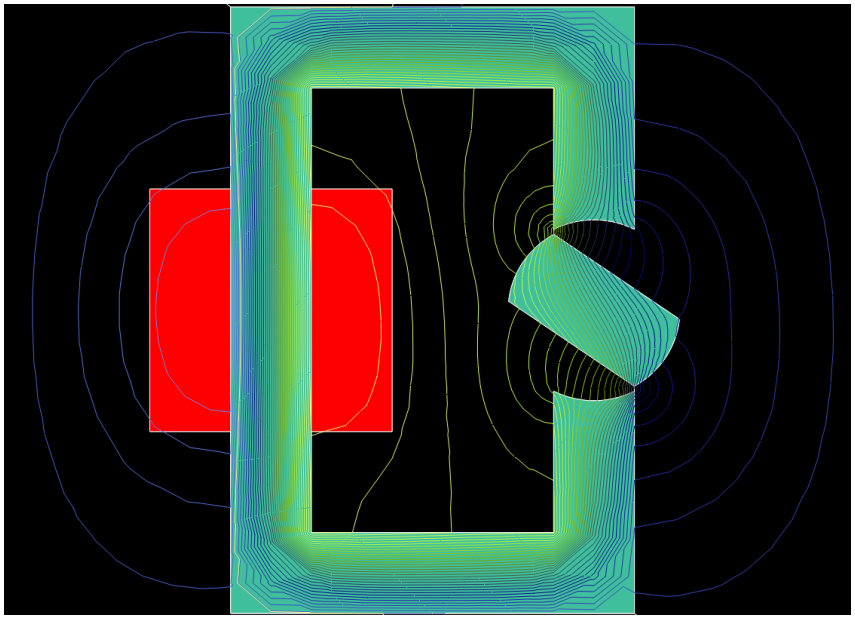


Figura 12 – Desenho de linhas equipotenciais da peça da Figura 9.

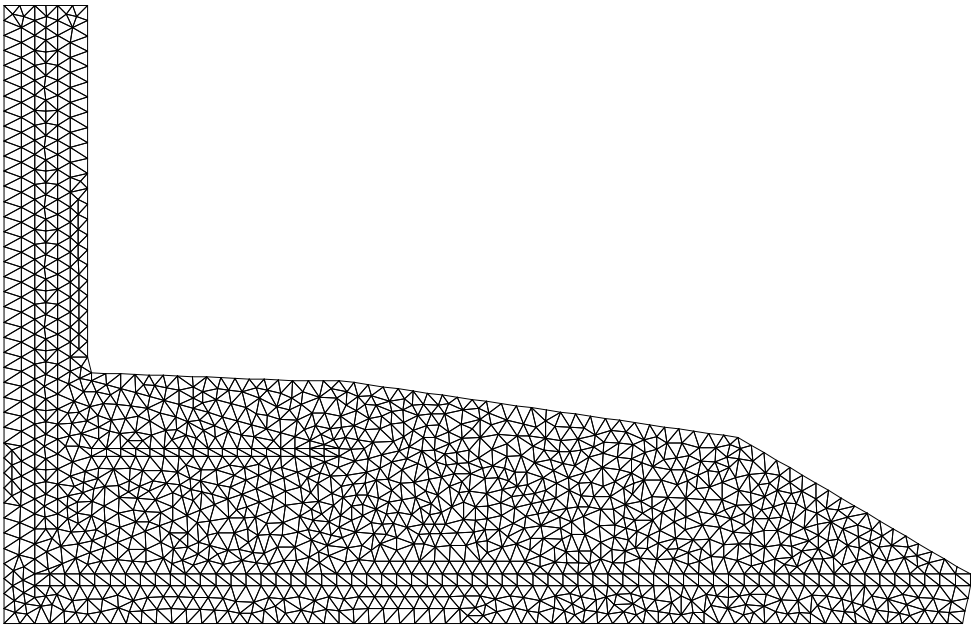


Figura 13 – Caso 2: Malha gerada pelo algoritmo antigo.

Verifica-se a partir dos dados da Tabela 2 que o algoritmo do Fem2014 leva a metade do tempo em comparação ao algoritmo do Fem2000. Além disso a qualidade da malha gerada é superior, pois o ângulo médio dos elementos é maior. A comparação entre as duas malhas geradas evidencia a melhor qualidade dos elementos próximo a regiões curvas da geometria. O algoritmo utilizado no gerador de malhas anterior cria um grande número de elementos.

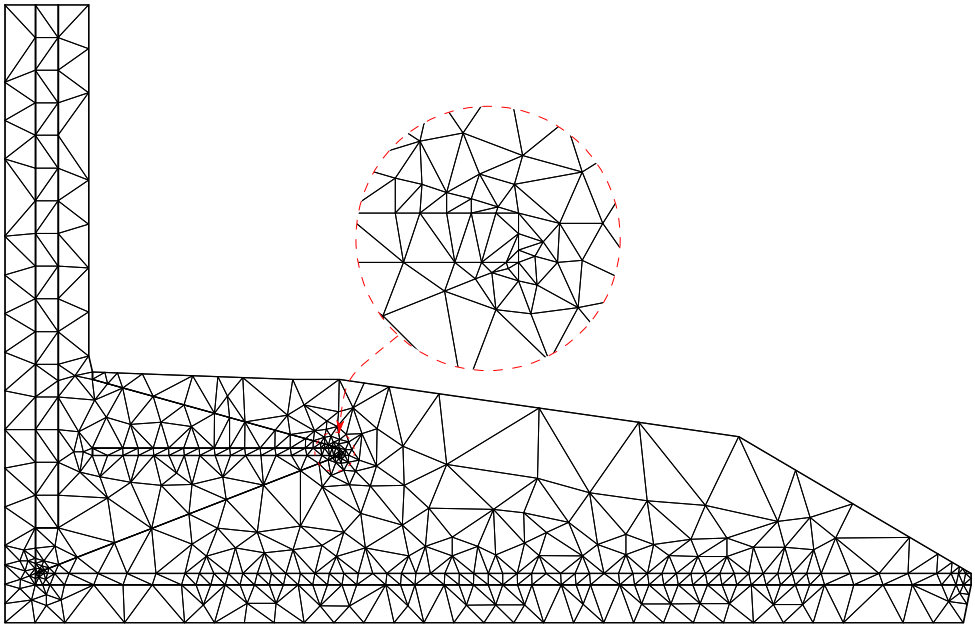


Figura 14 – Malha gerada pelo novo algoritmo de refinamento.

Tabela 1 – Dados comparativos de malha gerada para os dois algoritmos (Figuras 13 e 14)

	Malhador Fem2014	Malhador Fem2000
Área menor elemento	0,043617	2,02
Área média elementos	93,2801	5,88
Menor ângulo	$16,84°$	$16,78°$
Ângulo médio	$24,85°$	$20,38°$
Nº elementos	824	2666
Nº nós	933	1434
Fat. qual. médio	0,809161	0,91
Fat. qual mín	0,280771	0,46
Tempo geração	1,4 s	9,0 s

Tabela 2 – Dados para geração de malha com mais de 10 mil elementos

	Malhador Fem2014	Fem2000
Área menor elemento	0,0004	0,036
Área média elementos	0,44	0,49
Menor ângulo	$16,8°$	$16,7°$
Ângulo médio	$24,8°$	$21,3°$
Nº elementos	14713	13140
Fat. qual. médio	0,81	0,89
Fat. qual mín	0,25	0,18
Tempo geração	55s	2min 29s

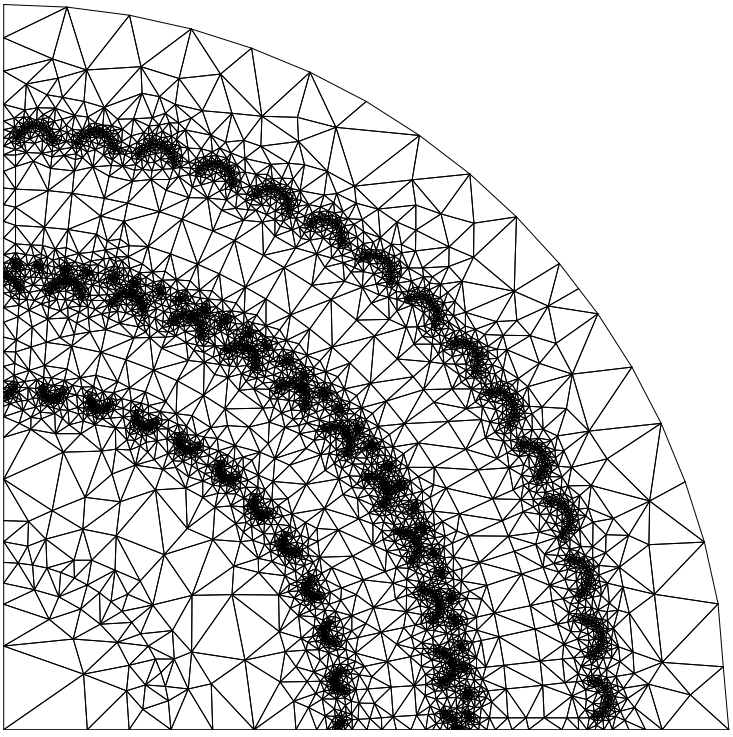


Figura 15 – Malha com aproximadamente 14 mil elementos - Fem2014.

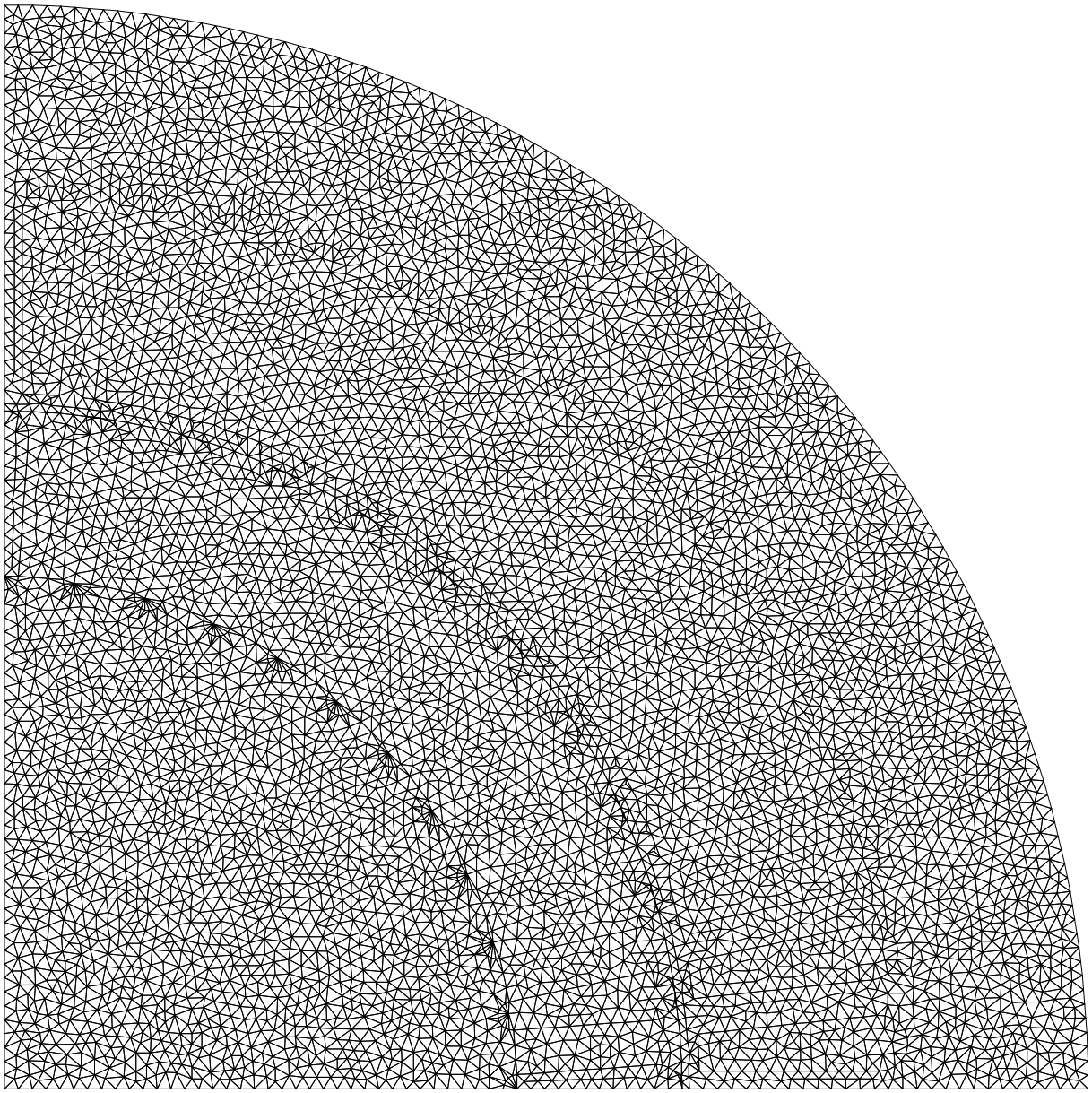


Figura 16 – Malha com aproximadamente 13 mil elementos - Fem2000.

5 Conclusões e Trabalhos Futuros

5.1 Conclusões

A nova versão do sistema Fem2000, denominada de Fem2014, foi satisfatoriamente implementada com as bibliotecas e ferramentas de software multiplataforma que foram selecionadas. Foram feitos testes nas plataformas Windows e Linux, e verificou-se que o software funciona satisfatoriamente em ambos sistemas operacionais. Além das funcionalidades já existentes, agora o software possui novos recursos que aumentaram sua usabilidade. O sistema novo é mais fácil e rápido de utilizar, tornando-o mais atrativo para utilização nas aulas de laboratório, conforme pode-se verificar pelos casos apresentados.

Os resultados obtidos confirmam que o algoritmo de refinamento implementado é superior ao antigo em vários aspectos, entre os quais o principal é o menor tempo de geração da malha. Os resultados obtidos com o novo algoritmo apresentarem uma melhor relação entre a qualidade da malha gerada e os recursos computacionais utilizados.

Uma malha mais fina possibilita maior precisão no resultado, porém aumenta o tempo computacional. Na prática, verificou-se que os melhores resultados são atingidos com o uso de um bom algoritmo de refinamento, aliado à decisão técnica do usuário.

5.2 Trabalhos Futuros

Apesar do algoritmo implementado ser mais rápido que o anterior, é necessária a implementação de algoritmos mais eficientes para manipulação de dados. A utilização de um índice espacial, como árvore R, pode aumentar consideravelmente o desempenho.

Referências

JIN, J. *The finite element method in electromagnetics*. [S.l.]: John Wiley & Sons, 1993.

MENG, T. *The Case for Open Source: OSS vs Proprietary Software*. 2003. Disponível em: <http://www.opensos.net/>.

PAV, S. E. *Delaunay Refinement Algorithms*. Tese (Doutorado) — Carnegie Mellon University, 2003.

PEREIRA, L. A. *Método dos Elementos Finitos Aplicado ao Eletromagnetismo*. [S.l.], 2000.

SADIKU, M. N. O. *Elements of Electromagnetics*. 3ª. ed. [S.l.]: Oxford University Press, 2009. ISBN 9780195134773.

SHEWCHUK, J. R. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In: LIN, M. C.; MANOCHA, D. (Ed.). *Applied Computational Geometry: Towards Geometric Engineering*. [S.l.]: Springer-Verlag, 1996, (Lecture Notes in Computer Science, v. 1148). p. 203–222.

SHEWCHUK, J. R. *Lecture Notes on Delaunay Mesh Generation*. 2012. Disponível em: <http://www.cs.berkeley.edu/~jrs/meshpapers/delnotes.pdf>.

WRIGHT, R. S.; LIPCHAK, B.; HAEMEL, N. *OpenGL(R) SuperBible: Comprehensive Tutorial and Reference*. 6ª. ed. [S.l.]: Addison-Wesle, 2013.

YVINEC, M. *Triangulations and Meshes*. 2009.