TESE
802

PROGRAMA COMPUTACIONAL PARA SIMULAÇÃO E ANÁLISE DE SISTEMAS ELÉTRICOS DE POTÊNCIA COM RECURSOS GRÁFICOS INTERATIVOS

Biblioteca
MAUÁ
BIM
de Engenharia
de Itajubá - EFEI

MESTRANDO: José Geraldo Paiva

ORIENTADOR: Prof. Júlio César Tibúrcio

DISSERTAÇÃO APRESENTADA À COORDENAÇÃO DOS CURSOS DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA DA ESCOLA FEDERAL DE ENGENHARIA DE ITAJUBÁ, COMO PARTE DOS REQUISITOS PARA OBTENÇÃO DO TÍTULO DE "MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA".

ITAJUBÁ, MAIO DE 1995

CLASS. 621.316:519.674 : 681.322(043.2)

CUTTER. P142p

TOMBO. 802

EFEI / BIBLIOTECA

ESTE LIVRO DEVE SER DEVOLVIDO NA ÚLTIMA DATA CARIMBADA.

05.4.96		
26.4.96		
07.8.96		
16.8.96		
13.8.96		
08/09/98		

Ministério da Educação e do Desporto
ESCOLA FEDERAL DE ENGENHARIA DE ITAJUBÁ
Reconhecida Lei 3232 - 05/01/1917

A N E X O I I

FOLHA DE JULGAMENTO DA BANCA EXAMINADORA

Título da Dissertação: "Programa Computacional para Simulação e Análise de Sistemas Elétricos de Potência com Recursos Gráficos Interativos"

Autor: José Geraldo Paiva

JULGAMENTO

EXAMINADORES	CONCEITO	RUBRICA
1º	A	
2º	A	
3º	A	

Resultado Médio: Conceito A, ou seja, APROVADO

Observações: _____

Itajubá, 10 de maio de 1995.

Dr. Eduardo Márcio Teixeira Nery
1º Examinador - CEMIG

Prof. Renato de Aquino Faria Nunes
2º Examinador - EFEI

Prof. Júlio César Tibúrcio
3º Examinador - EFEI - Orientador

Pró-Diretoria de Pesquisa e Pós-Graduação - EFEI - Escola Federal de Engenharia de Itajubá
Av BPS, 1303 - Cx Post 50 - 37500-000 - ITAJUBÁ/MG - BRASIL - Tel.: (035)629-1118 - 629-1121;
Fax (035)629-1148 - International Dialing: Please replace (035) by (+5535)
e-Mail:PPG@EFEI.UUCP@DCC.UFMG.BR

Este trabalho é dedicado à minha esposa, Maria de La Luz e aos meus filhos Diego e André, pelo permanente encorajamento e compreensão pelo tempo que deixei de dedicar-lhes para levar a cabo esta empreitada.

AGRADECIMENTO

Ao orientador, Prof. Júlio César Tibúrcio, pela dedicação, amizade e auxílio, prestados através de suas orientações, críticas e sugestões, para que o desenvolvimento deste trabalho pudesse ser executado da melhor forma.

À CPFL (Companhia Paulista de Força e Luz - Campinas - S.P.), especialmente ao Diretor de Operações à época, Eng. Arlindo Gonçalves Araújo, e ao Gerente da Divisão de Estudos da Operação, Eng. José Antônio Rennó Bittencourt, pela oportunidade que me proporcionaram de participar do curso de Pós-graduação da EFEI, sem a qual este trabalho não poderia realizar-se.

Aos meus colegas da CPFL pelo apoio e sugestões para melhoria do trabalho, bem como pelo auxílio para dirimir dúvidas relativas às linguagens de programação utilizadas.

SUMÁRIO

A presente dissertação descreve o desenvolvimento de um programa computacional capaz de explorar a capacidade gráfica dos microcomputadores atuais para criar rotinas de interação entre o usuário e programas de simulação e análise de sistemas elétricos, tais como Fluxo de Carga. O objetivo principal deste trabalho foi o de tornar disponível uma moderna ferramenta de simulação, para utilização pelas empresas do setor elétrico brasileiro, bem como pelas universidades, centros de pesquisa e empresas de consultoria e serviços que atuam na área de eletricidade.

A partir da pesquisa de trabalhos similares publicados anteriormente e das linguagens de programação gráfica disponíveis, chegou-se a conclusão de que seria conveniente utilizar a linguagem C para processamento da parte gráfica e a linguagem Fortran para execução da simulação do sistema elétrico, com utilização do método para cálculo de fluxo de potência conhecido por "Desacoplado Rápido".

O produto final mostrou-se simples de ser utilizado, de execução rápida e capaz de fornecer ao usuário as principais informações necessárias à análise em forma gráfica, sobre o diagrama do sistema elétrico.

ÍNDICE

INTRODUÇÃO 1

CAPÍTULO 1 - HISTÓRICO 5

CAPÍTULO 2 - DESCRIÇÃO DA FERRAMENTA DESENVOLVIDA ... 12

2.1 - ALGUNS COMENTÁRIOS SOBRE TÉCNICAS GRÁFICAS 12

2.1.1 - Sistema de Coordenadas 12

2.1.2 - Transformação de Coordenadas 15

2.1.3 - Primitivas Gráficas 17

2.1.3.1 - Traçado de Retas em Dispositivos do Tipo "Raster" 18

2.1.3.2 - Traçado de Círculos em Dispositivos do Tipo "Raster" 21

2.1.4 - Recorte 24

2.1.5 - Preenchimento de Polígonos 26

2.1.6 - Considerações 27

2.2 - ESTRUTURA BÁSICA DO PROGRAMA 28

2.2.1 - Inicialização 29

2.2.2 - Leitura/Gravação dos Dados 29

2.2.3 - Edição Gráfica 30

2.2.4 - Execução 31

2.2.5 - Gerenciamento da Transformação de Coordenadas 32

2.2.6 - Impressão dos Resultados 35

2.3 - DETALHAMENTO DO PROGRAMA 36

2.3.1 - Funções do Módulo PRINCIP.C 38

2.3.2 - Funções do Módulo FMOUSE.C 42

2.3.3 - Funções do Módulo LER.C 43

2.3.4 - Funções do Módulo GRAVAR.C 45

2.3.5 - Funções do Módulo EDIÇÃO.C 46

2.3.6 - Funções do Módulo DESENHO.C 48

2.3.7 - Funções do Módulo ALTERA.C 49

2.3.8 - Funções do Módulo MODIFICA.C e MODIFIC1.C 51

2.3.9 - Funções do Módulo MENUTECL.C 54

2.3.10 - Funções do Módulo EXECUTAR.C 54

2.3.11 - Funções do Módulo TRANSFOR.C 55

2.3.12 - Funções do Módulo IMPRIME.C 58

CAPÍTULO 3 - RESULTADOS E CONCLUSÕES 61

3.1 - RESULTADOS 61

3.1.1 - Testes Realizados 61

3.1.2 - Apresentação dos Resultados Obtidos 63

3.1.2.1 - Edição do Sistema 67

3.1.2 .2- Entrada de Dados 72

3.1.2.3 - Execução 73

3.1.2.4 - Impressão dos Resultados 79

3.2 - CONCLUSÕES 80

REFERÊNCIAS BIBLIOGRÁFICAS 83

ANEXO I - EXEMPLO DE DIAGRAMA IMPRESSO

ANEXO II - EXEMPLO DE SAÍDA NO FORMATO TEXTO

INTRODUÇÃO:

Durante muitos anos o planejamento da operação e da expansão de sistemas elétricos de potência dependeu dos cálculos manuais e do uso de simuladores analógicos, restritos a grandes empresas e centros de pesquisa. A introdução da simulação do comportamento do sistema elétrico em computadores digitais trouxe um grande avanço nessa área, possibilitando uma significativa redução nos custos necessários para sua implementação, proporcionando assim a difusão de sua utilização.

A evolução constante e rápida dos computadores digitais permitiu que mais algoritmos fossem implementados e que fosse aprimorada a interface com o usuário, porém, esta interface permaneceu, na maioria dos programas, na forma de textos, o que dificulta sua utilização por usuários não especializados.

O contato com especialistas da área de análise de sistemas da CPFL e de outras empresas do Sistema Elétrico Brasileiro mostrou que ainda temos poucos aplicativos que utilizam os recursos gráficos disponíveis atualmente nos micro computadores, que já dispõem de grande capacidade de processamento.

A maior utilização de recursos gráficos no Brasil tem ocorrido em implementações de centros de controle automatizados, porém com funções específicas e a maioria sem o suporte de ferramentas de análise para o operador ou analista.

Uma frase bastante conhecida: "uma imagem vale por mil palavras", retrata o fascínio que uma imagem exerce sobre o ser humano, bem como o conteúdo de informações que pode transmitir.

Essas observações levaram à proposição deste trabalho. A ferramenta desenvolvida permite explorar os recursos dos computadores na área gráfica,

integrando a simulação do sistema elétrico à sua representação visual, possibilitando uma simplificação em sua utilização, com consequente ampliação no universo de usuários.

Foi escolhido o processamento de fluxo de carga para o desenvolvimento deste trabalho, por ser uma ferramenta básica para o analista de sistemas elétricos, sendo assim de utilização mais ampla. Foi considerado ainda que o resultado de seu processamento pode ser implementado de forma mais intuitiva sobre um diagrama gráfico, o que facilita um desenvolvimento inicial. A idéia, no entanto, pode ser estendida a outros cálculos e simulações usuais na análise de sistemas elétricos, tais como: curto circuito, segurança operativa, estabilidade, confiabilidade, sem grandes dificuldades.

Assim, procurou-se com este trabalho propiciar um produto que possibilite o processamento do fluxo de carga de forma simples pelo usuário, exteriorizando todos os resultados relevantes no diagrama unifilar do sistema elétrico, deixando apenas os detalhes (como número de iterações, desvios, totalizações e outros) na forma de textos.

As principais vantagens observadas na aplicação de uma ferramenta tal como a desenvolvida neste trabalho são:

- indicação visual dos sentidos dos fluxos nas linhas;
- indicação, através de cores diferenciadas, da ultrapassagem do carregamento admissível em um equipamento;
- possibilidade de representação do diagrama unifilar em forma geográfica;
- abertura e fechamento de linhas, com atuação direta sobre a representação dos equipamentos de manobra;
- maior agilidade na simulação de contingências;
- facilidade da visualização dos diversos níveis de tensão, pela utilização de cores diferenciadas.

Desta forma, a utilização do processamento de fluxo de carga torna-se mais simples de ser implementada por técnicos não habituados a lidar com esta ferramenta, estendendo sua aplicação a programas de treinamento, tomadas de decisões em centros de operação não automatizados, utilização didática no meio universitário e também em áreas que já utilizam outros programas de fluxo de carga, pois traz vantagens para solução de problemas isolados, que exigem grande quantidade de casos a analisar.

O desenvolvimento do trabalho ocorreu de forma lenta no início, devido principalmente à dificuldade de definição de uma linguagem de programação que atendesse às necessidades do desenvolvimento. Por fim optou-se pela linguagem C, por ser bastante portátil e já incorporar um razoável número de funções, com suporte a implementações gráficas [4, 8].

Por fim obteve-se um produto que propicia toda a interface com o usuário em forma gráfica, através da utilização de botões, "mouse" (não obrigatório) e janelas, oferece opções de edição gráfica, impressão, leitura de dados, alteração de dados gráficos e elétricos, aproximação e afastamento da exibição do sistema ("zoom"), e é totalmente portátil entre equipamentos que utilizam processadores da família 8086 ou 80386 da Intel, sob gerenciamento do sistema operacional DOS e que possuam, de preferência, adaptador gráfico VGA.

A dissertação que aqui apresentamos está organizada da seguinte forma:

No Capítulo 1 faz-se uma revisão cronológica, com a apresentação de alguns trabalhos que nos antecederam e que serviram de roteiro inicial, quando da definição conceitual da interface gráfica interativa.

No Capítulo 2 é apresentada inicialmente uma introdução teórica sobre a representação gráfica em computadores, para que se possa conhecer como são as rotinas básicas de desenho e ter uma idéia do esforço computacional (entenda-se como número de operações necessárias) para sua execução.

A seguir é feita uma abordagem global do programa, procurando apresentar a conceituação utilizada no desenvolvimento de suas principais funções.

Por fim é feita uma apresentação detalhada de cada função, para que possa ser analisada a metodologia utilizada, bem como servir de referência para futuras alterações no programa.

No Capítulo 3 faz-se a apresentação dos resultados através de uma sequência dos principais comandos do programa, utilizando o recurso de fotografias tomadas diretamente do monitor, para avaliação da apresentação visual obtida.

Por fim é apresentada a conclusão do trabalho onde ressalta-se a potencialidade da expansão futura do "software", com o acréscimo de novas funções de análise de sistemas elétricos.

CAPÍTULO 1

HISTÓRICO

Interfaces gráficas para programas de análise de sistemas elétricos tem sido pesquisadas já há alguns anos, desde que os computadores passaram a dispor de algum recurso para introdução de rotinas gráficas.

O primeiro trabalho pesquisado para o desenvolvimento desta dissertação e que originou a idéia da mesma, foi elaborado igualmente a título de dissertação de mestrado [11] na Universidade de Manchester (UMIST).

O trabalho descreve a implementação do programa "IPSA - Interactive Power System Analysis", em computador CDC Cyber 72, e o desenvolvimento de uma interface gráfica para o mesmo, implementada através dos recursos gráficos dos terminais Tektronics 4010.

Para interação com o usuário foi utilizada uma combinação da posição do cursor na tela com o pressionamento de teclas pré-determinadas no teclado, resultando em um diálogo de fácil utilização.

A representação gráfica do sistema elétrico permite o desenho dos equipamentos em um diagrama unifilar, sendo sua montagem feita na tela, de forma interativa. Os dados elétricos são fornecidos, ou alterados, através de telas padronizadas, facilitando sua atualização.

A exteriorização dos resultados foi feita sobre o diagrama gráfico, para os fluxos nas linhas e para os valores de tensão, geração e carga, em uma outra tela na forma de texto.

Pode-se considerar que o programa desenvolvido era bastante complexo para os recursos disponíveis na época, já incorporando conceitos modernos de interação com o usuário.

Escola Federal de Engenharia de Itajubá - EFEI
Biblioteca MAUÁ BIM

Em trabalhos mais recentes pode-se constatar o grande avanço verificado nos dispositivos de representação gráfica, bem como nas ferramentas para suporte ao desenvolvimento.

Um trabalho apresentado na "International Conference on Large High Voltage Eletric System" do CIGRÉ, de 1989 [13], divulga o sistema denominado PLATINE desenvolvido pela Eletricité de France, para planejamento de sistemas de transmissão. O trabalho implementa simulações de fluxo de carga, curto circuito, otimização e segurança no controle de tensão, confiabilidade, análise de custos e estabilidade transitória.

Para interação com o usuário foi desenvolvida uma interface gráfica, utilizando biblioteca no padrão GKS-2D [1, 2, 9]. É possível a exteriorização do sistema elétrico em forma geográfica, utilizando a divisão da tela em três janelas. A primeira apresenta o sistema de forma regionalizada. A segunda, um "zoom" sobre a seção estudada e a terceira, um diagrama da subestação em forma de blocos.

Nos diagramas geográficos, as subestações são representadas por nós, sendo indicados nas linhas de transmissão os fluxos resultantes da análise e seu sentido. Outros resultados (como tensão, geração e carga) são obtidos na terceira janela referenciada anteriormente, reservada ao detalhamento da subestação.

Os programas de simulação são escritos na linguagem de programação FORTRAN e a base de dados na linguagem SQL.

Os autores ressaltam que a portabilidade do sistema Platine é total, com exceção da base de dados, que pode ser utilizada apenas em ambiente IBM.

Isto, entretanto, é discutível, pois a portabilidade da interface gráfica utilizando a padronização GKS está limitada a equipamentos onde foi desenvolvida biblioteca nesta padronização. Durante a fase de pesquisa da linguagem a ser utilizada para o desenvolvimento de nossa dissertação, utilizamos inicialmente a padronização GKS. Porém, a mesma foi abandonada posteriormente, devido à dificuldade de obter-se

uma biblioteca atualizada para o sistema operacional DOS, uma vez que a mesma é totalmente dependente do compilador utilizado.

Um outro trabalho, publicado pela universidade de Canterbury [12], apresenta um programa denominado "Display Power", que implementa uma interface gráfica para interação com programas de fluxo de carga, curto circuito e análise harmônica, desenvolvido com o objetivo de utilização didática por alunos dos cursos de graduação na área de sistemas de potência.

Foi desenvolvido em "Workstation" MicroVaxII e, na época da edição do artigo, estava sendo portada para micro computadores padrão PC. Foi utilizada a linguagem de programação Módula-2 para a interface gráfica e, para os módulos de simulação, a linguagem FORTRAN.

Os autores ressaltam as qualidades de interatividade do programa, através do uso de "mouse", janela "pop-up" e janela de "help", em cada etapa do processamento.

A sua utilização é feita na disciplina de sistemas de potência e propicia a exemplificação aos alunos dos conceitos passados através do ensino formal. Foi observado um maior interesse pelos alunos nessa área, tendo os autores mostrado intenção em incluir a utilização do programa pela turma de 2° ano (anterior ao início das disciplinas específicas para sistemas de potência), para despertar o interesse na área.

O exemplo apresentado no trabalho possui boa disposição gráfica dos equipamentos, sendo o "lay-out" de tela bem definido.

No entanto, a linguagem utilizada para elaboração da interface poderá resultar em problemas de portabilidade do "software", não tendo os autores registrado qualquer comentário mais detalhado sobre a sua transferência para plataforma PC, em desenvolvimento na época da edição do artigo.

Um interessante trabalho, publicado pelo instituto de tecnologia de Chicago [14] apresenta uma implementação de interface gráfica sobre ambiente "Windows

Microsoft" que possibilita interação com "softwares" comerciais, escritos em várias linguagens de programação. O programa foi denominado "Power System Graphics Package - PSGP".

A interface propicia a entrada de dados, edição gráfica e execução de programas externos. Foram utilizados programas comercialmente disponíveis e escritos nas linguagens de programação C, Fortran, Basic, Nexpert, Object Shell e Brain Maker Neural Network. A passagem de dados entre a interface e esses programas é feita por meio de arquivos no formato ASCII e executados por meio de "scripts" onde são passados os parâmetros necessários.

É implementada, além da análise do sistema de transmissão, a estimação de estado, a programação de geração multi-área, com otimização de custo e a análise de segurança. São utilizados conceitos de sistemas especialistas e de redes neurais em alguns procedimentos para redução do tempo de processamento, como o caso da otimização na programação de geração.

A exteriorização dos resultados é feita em páginas gráficas, sobre o diagrama unifilar do sistema (caso do fluxo nas linhas de transmissão), ou em páginas de texto (no caso de programação de geração, onde é apresentada a programação hora a hora, nas 24 horas do dia).

Para montagem das páginas gráficas são utilizadas apenas imagens no formato "bit-map", o que impede a implementação de efeito "zoom" do diagrama no sistema. Isto limita o número de barras representadas graficamente. Nos exemplos apresentados pelos autores na análise de um sistema de seis barras foi possível a representação dos fluxos nas linhas no diagrama unifilar. Para um sistema de trinta barras foi necessária a apresentação dos fluxos nas linhas em uma outra página, no formato texto.

É implementada ainda a animação da representação gráfica, através da mudança de cores dos equipamentos que entram em sobrecarga para amarelo e vermelho, dependendo do gráu de severidade.

Outra abordagem de implementação de interfaces gráficas foi publicada pela universidade do estado do Arizona [15]. Essa implementação explora o ambiente gráfico "X Window" (padrão MOTIF/OSF) sobre sistema operacional "UNIX", apresentando uma discussão sobre a utilização da "programação orientada a objetos" no desenvolvimento de interfaces gráficas.

É apresentado um protótipo experimental de interface gráfica desenvolvido sobre ambiente "X Window" e linguagem de programação C, para um sistema de gerenciamento de energia.

Os autores optaram por esta metodologia devido à uma experiência anterior de implementação de um sistema de simulação em tempo real, desenvolvido utilizando várias rotinas gráficas específicas da "Workstation" Apollo (Hewlett Packard), o que inviabilizou a portabilidade do "software".

A opção pela programação orientada a objetos foi feita por suas características de modularidade e por incorporar conceitos de herança e hierarquia de classes, o que proporciona facilidades na construção e manutenção do "software".

Foi utilizada a linguagem de programação C, embora não sendo uma linguagem orientada a objetos, obedecendo estritamente a metodologia OOP. Não foi utilizada uma linguagem mais adequada, como C++, que facilitaria sua implementação, devido à não disponibilidade de compiladores em algumas plataformas de "hardware".

O novo sistema desenvolvido explora a capacidade multi-janela do ambiente "X Window", o que possibilita a superposição de vários níveis do sistema elétrico na tela, facilitando a utilização pelo usuário. Utiliza também sua capacidade de interligação em rede, o que propicia que mais de um usuário participe , através de um terminal, de forma compartilhada da simulação.

É apresentada a utilização de um micro computador padrão PC, interligado como terminal, rodando a interface, através de "software" adequado de emulação de terminal "X Window".

A abordagem dada pelos autores sobre a programação orientada a objetos é coerente com a aplicação desenvolvida. Os ambientes para desenvolvimento de interfaces gráficas, como "X Window" (MOTIF), "Windows Microsoft", "Open Look" (SUN), são padrões estruturados nesta metodologia. Assim, qualquer aplicativo que utilize essas interfaces, aproveita melhor suas capacidades seguindo a metodologia de programação orientada a objetos.

No Brasil, alguns trabalhos tem sido apresentados em seminários promovidos no âmbito do setor elétrico nacional, onde pode-se verificar algumas pesquisas visando explorar a capacidade gráfica dos micro computadores.

Foi apresentado no IV EDAO - Encontro para Debates de Assuntos da Operação, em 1993, o trabalho "Sistemas de Análise de Fluxo de Potência para a Operação em Tempo Real com Apresentação dos Resultados Plotados em Diagramas Unifilares" [16], que descreve um aplicativo computacional implantado no CNOS - Centro Nacional de Operação do Sistema. Este aplicativo permite a obtenção de casos de fluxo de potência atualizados de acordo com as condições de geração e carga estabelecidas para cada horário no Programa Diário da Operação, sendo os resultados exibidos em diagramas gráficos idênticos ao esquemáticos de área das consoles de supervisão em tempo real.

É utilizado o programa ANAREDE desenvolvido pelo CEPEL - Centro de Pesquisas da Eletrobrás, para processamento do fluxo de potência, que utiliza como dados de entrada arquivos textos gerados automaticamente (produção das usinas e carga das empresas) ou manualmente através da manipulação de arquivos tipo texto (programação de desligamentos e exclusão de equipamentos indisponíveis devido a contingência), pelo operador.

Os resultados gerados pelo processamento do fluxo de carga são então exteriorizados em diagramas unifilares, criados através de telas semi-gráficas (montadas a caracteres), idênticas às utilizadas no sistema de operação em tempo real. A montagem das telas gráficas é feita a partir de arquivos textos. Para representar uma barra, por exemplo, é colocada uma sequência de asteriscos no arquivo texto, que será representada como uma linha grossa contínua.

Pode-se considerar que o produto obtido é um grande avanço na apresentação do processamento de fluxo de carga, principalmente com o objetivo de utilização por operadores de centros de controle, mas que ainda pode ser aperfeiçoado através da implementação da interatividade com o usuário, uma vez que qualquer alteração nos dados de entrada depende da manipulação de arquivos textos.

CAPÍTULO 2

DESCRIÇÃO DA FERRAMENTA DESENVOLVIDA

Este capítulo faz uma introdução às técnicas mais comuns desenvolvidas para a representação gráfica em computadores, com o objetivo de propiciar uma melhor compreensão do trabalho, que passa a ser descrito após esse preâmbulo.

2.1 - ALGUNS COMENTÁRIOS SOBRE TÉCNICAS GRÁFICAS

Este item descreve algumas técnicas gráficas utilizadas no desenvolvimento do trabalho. Embora façam parte da biblioteca gráfica utilizada, o seu conhecimento auxilia no melhor entendimento do trabalho.

Serão descritas apenas as técnicas para desenho em duas dimensões (2D), uma vez que o trabalho não se estendeu ao desenho em 3 dimensões (3D), embora muitas das técnicas descritas também se apliquem a este tipo de desenho. A descrição limita-se às técnicas utilizadas no trabalho, uma vez que o campo de aplicações gráficas é bastante vasto.

2.1.1 - Sistemas de Coordenadas

A base de qualquer gráfico é o sistema de coordenadas, a partir do qual o mesmo é montado.

Cada dispositivo gráfico para interface com o computador possui seu sistema de coordenadas. O sistema mais utilizado é o cartesiano, variando entre cada dispositivo os limites de coordenadas.

Por exemplo, no adaptador para vídeo CGA ("Color Graphics Adapter") tem-se dois modos de desenho: o de alta-resolução, com coordenadas de 0 a 640 na abscissa e de 0 a 200 na ordenada (comumente referido como resolução 640x200), e o de baixa resolução, com a abscissa variando de 0 a 320 pontos, e a ordenada, de 0 a 200 pontos (resolução 320x200).

No adaptador de vídeo VGA ("Video Graphics Array"), tem-se, no modo de alta resolução (padronizado pela IBM),a abscissa variando de 0 a 640 pontos e a ordenada, de 0 a 480 pontos (resolução 640x480) [10], podendo cada ponto assumir até 256 cores.

Esses adaptadores atingem atualmente resolução de até 1024 pontos na horizontal por 768 pontos na vertical; resoluções superiores a esta são obtidas com adaptadores especiais, que variam de fabricante para fabricante.

Assim, se um determinado programa desejar desenhar uma linha do ponto (0,0) ao ponto (640,200), a exteriorização desta linha em um adaptador CGA aparecerá cortando a tela de uma extremidade a outra, enquanto a exteriorização em um adaptador VGA aparecerá unindo uma extremidade do vídeo ao meio da tela, como pode ser visualizado na figura 2.1:

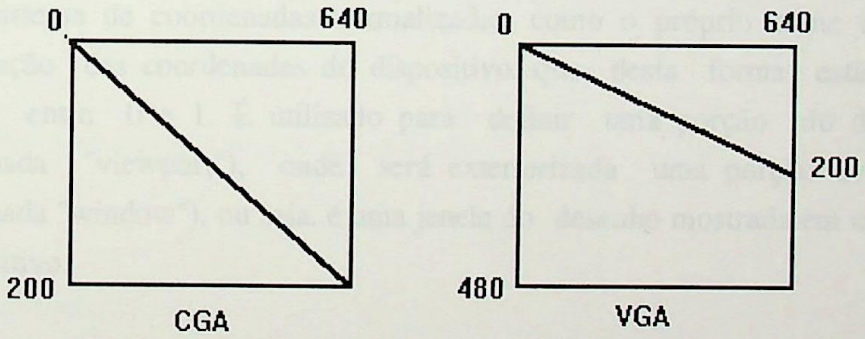


FIGURA 2.1

Obs: Convém ressaltar, neste ponto, que a padronização utilizada pelos adaptadores gráficos para vídeo tem a coordenada (0,0) no canto superior esquerdo.

Este efeito é desfavorável, quando se deseja a portabilidade do "software" entre equipamentos com dispositivos adaptadores de vídeo, impressoras ou "plotters" diferentes daqueles para os quais o mesmo foi inicialmente projetado.

Este problema pode ser solucionado através da definição de sistemas de coordenadas distintos para utilização pelo "software" e para utilização pelo equipamento, sendo feita uma transformação de coordenadas sempre que for necessária a exteriorização de um ponto.

Na padronização GKS foram definidos 3 sistemas de coordenadas:

- Coordenadas Globais
- Coordenadas Normalizadas
- Coordenadas de Dispositivo

O sistema de coordenadas globais é onde o usuário definirá todos os pontos que compõem o objeto, que pode utilizar unidades reais, como metros, pés, milímetros, etc. Desta forma o desenho resultante torna-se independente do dispositivo em que será exteriorizado.

O sistema de coordenadas normalizadas, como o próprio nome diz, faz a normalização das coordenadas do dispositivo, que desta forma estão sempre limitadas entre 0 e 1. É utilizado para definir uma porção do dispositivo (denominada "viewport"), onde será exteriorizada uma porção do desenho (denominada "window"), ou seja, é uma janela do desenho mostrada em uma janela do dispositivo.

Por fim, o sistema de coordenadas de dispositivo é dependente do número de pontos que o mesmo utiliza para formar a imagem, o que varia de dispositivo para dispositivo. Quanto maior o número de pontos, melhor a qualidade da imagem resultante.

Escola Federal de Engenharia de Itajubá - EFEI
Biblioteca
MAUÁ
BIM

2.1.2 - Transformação de Coordenadas

Sendo a imagem gráfica mapeada em um sistema de coordenadas real (coordenadas globais), para que possa ser exibida de forma adequada em um dispositivo é necessária a conversão entre esta coordenada e a coordenada do dispositivo. Esta conversão é denominada Transformação de Coordenadas, e é realizada com a utilização de matrizes. A figura 2.2 ilustra o processo.

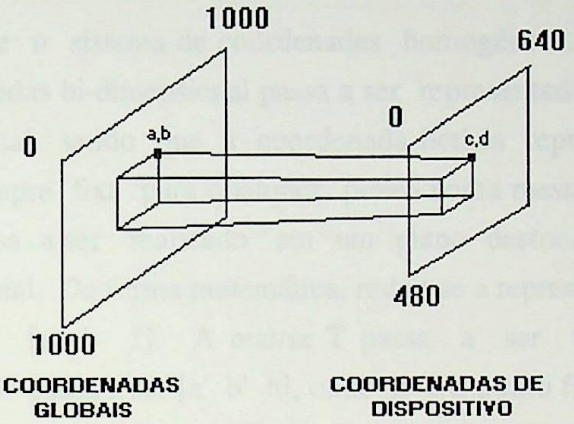


FIGURA 2.2

Assim, o ponto (a,b) no sistema de coordenadas globais passará a ser o ponto (c,d) no sistema de coordenadas de dispositivo, e a transformação do ponto (a,b) no ponto (c,d) é feita através da multiplicação por um escalar, ou seja:

$$c = a \cdot t1$$
$$d = b \cdot t2$$

que em forma matricial pode ser escrito como:

$$[c\ d] = [a\ b] \cdot \begin{bmatrix} t1 & 0 \\ 0 & t2 \end{bmatrix}$$

e de forma genérica

$$B = A \cdot T$$

sendo B o ponto transformado, A o ponto original e T a matriz de transformação.

Caso os elementos fora da diagonal (1,2 e 2,1) sejam diferentes de zero teremos, além da transformação de coordenadas, o deslocamento do ponto com

relação a sua posição original, porém, com as coordenadas iniciais influenciando neste deslocamento.

Para que a utilização de matrizes no cálculo de transformações seja genérico, é necessária uma alteração na equação anterior, uma vez que não existe uma matriz T tal que o produto [a b].T resulte em [a+m b+n] , que é o deslocamento do ponto inicial de m no sentido horizontal e n no sentido vertical.

Assim, define-se o sistema de coordenadas homogêneas, em que o ponto no sistema de coordenadas bi-dimensional passa a ser representado por um ponto no sistema tridimensional, sendo que a coordenada fictícia representada no sistema tridimensional é sempre fixa para qualquer ponto numa mesma transformação, ou seja, o desenho passa a ser realizado em um plano deslocado da origem no sistema tridimensional. De forma matemática, reduz-se a representar a matriz inicial [a b] pela matriz [a b 1]. A matriz T passa a ser tridimensional, e o resultado [a b 1] .T passa a ser [a' b' h], onde h é o ponto fictício tridimensional, e os pontos a' e b' são os pontos transformados.

Se a matriz T for igual a:

$$\begin{bmatrix} t1 & 0 & 0 \\ 0 & t2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

tem-se o mesmo efeito descrito anteriormente, ou seja, realizou-se apenas uma translação.

Se a matriz T passa a ser:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ m & n & 1 \end{bmatrix}$$

o resultado obtido é $a' = a + m$ e $b' = b + n$, ou seja, obtem-se a translação dos pontos.

Para a matriz:

$$\begin{bmatrix} \cos(Y) & -\text{sen}(Y) & 0 \\ \text{sen}(Y) & \cos(Y) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

tem-se a rotação do ponto (a, b) de um ângulo Y sobre a origem do sistema, como pode-se demonstrar com auxílio da figura 2.3:

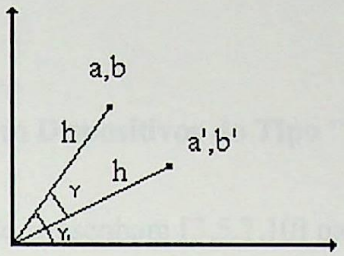


FIGURA 2.3

$$a = h \cdot \cos(Y1)$$
$$b = h \cdot \sen(Y1)$$
$$a' = h \cdot \cos(Y1 - Y)$$
$$b' = h \cdot \sen(Y1 - Y)$$

ou seja:

$$a' = a \cdot \cos(Y) + b \cdot \sen(Y)$$
$$b' = -a \cdot \sen(Y) + b \cdot \cos(Y)$$

2.1.3 - Primitivas Gráficas

Os dispositivos mais comuns para exteriorização de informações gráficas em computadores são do tipo "Raster", nos quais a imagem é formada por uma matriz de pontos.

No dispositivo vídeo, cada ponto dessa matriz corresponde a um "pixel" (ou seja, ponto iluminado) na tela do monitor. Assim, toda estrutura gráfica está baseada em grupos de pontos.

Para que seja facilitada a programação de figuras complexas, são definidos alguns grupamentos de pontos, que formam as figuras geométricas mais utilizadas. Essas figuras são chamadas de primitivas gráficas e são, principalmente: a reta, o círculo, a elipse, o retângulo, o polígono e o arco.

Para ilustrar a forma de desenho dessas primitivas serão descritas a seguir, as rotinas básicas para o desenho de retas e círculos, que compõem a estrutura básica de qualquer objeto geométrico.

2.1.3.1 - Traçado de Retas em Dispositivos do Tipo "Raster"

Será descrito o algoritmo de Bresenham [3,5,7,10] para traçado de linhas, por ser o mais utilizado e o que apresenta melhor desempenho em termos de tempo de execução. A qualidade visual das retas obtidas equivale à das obtidas com algoritmos que utilizam a equação linear da reta.

Este algoritmo torna-se eficiente devido ao fato de não requerer multiplicações ou operações com variáveis de ponto flutuante, ao contrário de algoritmos que utilizam a equação linear.

É um algoritmo incremental, que desenha a reta a partir de uma extremidade, e tomando como base a inclinação da mesma vai "ligando pixels" a partir de um fator de decisão, no eixo x ou y, de modo a obter uma sequência de pontos o mais próximo possível da reta.

Tomemos como exemplo a figura 2.4:

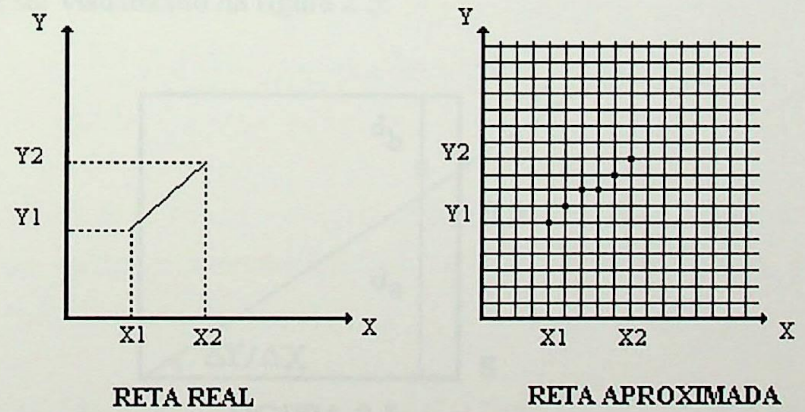
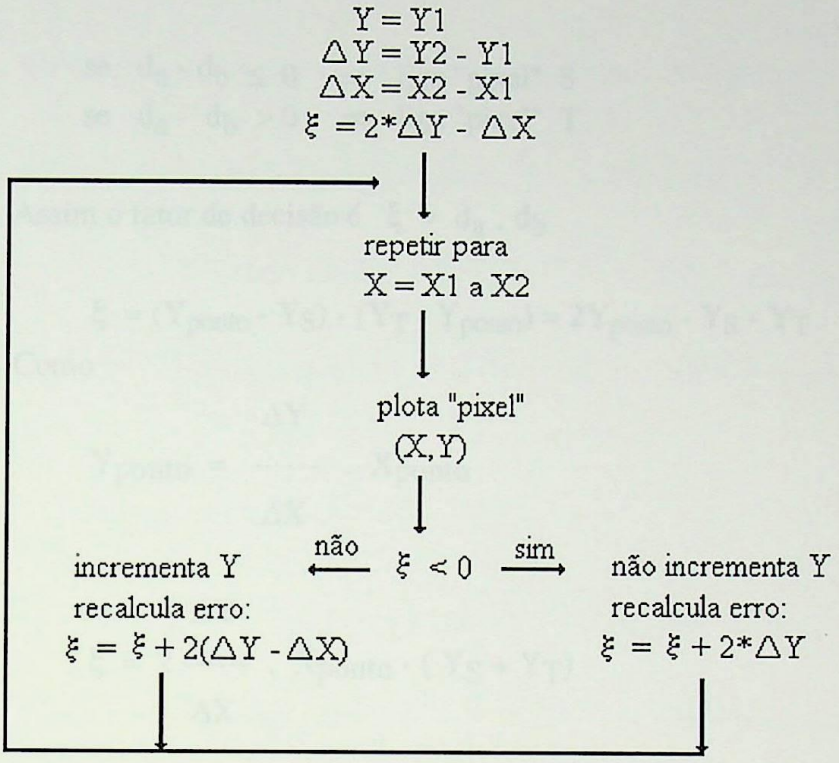


FIGURA 2.4

Escrevendo o algoritmo de forma simplificada:



Obs: Os incrementos em X e Y são sempre iguais a 1, pois referem-se a incrementos de um "pixel".

O fator de decisão ξ permite determinar qual "pixel" está mais próximo da reta, como pode ser visualizado na figura 2.5:

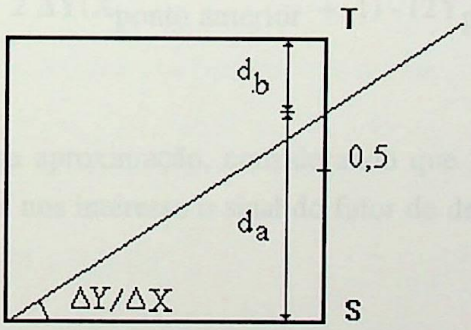


FIGURA 2.5

As expressões $\xi = 2*\Delta Y - \Delta X$, $\xi = \xi + 2\Delta Y - 2\Delta$ e $\xi = \xi + 2\Delta Y$ são obtidas seguindo o seguinte raciocínio:

se $d_a - d_b \leq 0 \Rightarrow$ liga "pixel" S
se $d_a - d_b > 0 \Rightarrow$ liga "pixel" T

Assim o fator de decisão é $\xi = d_a - d_b$

$$\xi = (Y_{ponto} - Y_S) - (Y_T - Y_{ponto}) = 2Y_{ponto} - Y_S - Y_T$$

Como

$$Y_{ponto} = \frac{\Delta Y}{\Delta X} \cdot X_{ponto}$$

$$\xi = 2\frac{\Delta Y}{\Delta X} \cdot X_{ponto} - (Y_S + Y_T)$$

Tomando $X_{ponto} = X_{ponto\ anterior} + 1$, $Y_T = Y_{ponto\ anterior} + 1$ e $Y_S = Y_{ponto\ anterior}$

$$\xi = \frac{1}{\Delta X}\left[2\Delta Y(X_{ponto\ anterior} + 1) - (2Y_{ponto\ anterior} + 1)\Delta X\right]$$

É feita agora uma aproximação, considerando que ΔX é positivo (reta no 1° octante) e que apenas nos interessa o sinal do fator de decisão, então, o termo $1/\Delta X$ é desconsiderado.

$$\xi = 2X_{ponto\ anterior}\Delta Y + 2\Delta Y - 2\Delta X.Y_{ponto\ anterior} - \Delta X$$

Para o ponto inicial, ou seja $X_{ponto\ anterior} = 0$ e $Y_{ponto\ anterior} = 0$

$$\xi = 2\Delta Y - \Delta X$$

Calculando agora o fator de decisão do próximo incremento:

Se $\xi \geq 0 \Rightarrow Y_{\text{próximo ponto}} = Y_{\text{ponto atual}} + 1$

$$\xi = 2(X_{\text{ponto atual}} + 1)\Delta Y + 2\Delta Y - 2\Delta X(Y_{\text{ponto atual}} + 1) - \Delta X$$

$$\xi = 2X_{\text{ponto atual}}\,\Delta Y + 2\Delta Y + 2\Delta Y - 2\Delta X.Y_{\text{ponto atual}} - 2\Delta X - \Delta X$$

ou seja

$$\xi_{\text{novo}} = \xi_{\text{atual}} + 2\Delta Y - 2\Delta X$$

Se $\xi < 0 \Rightarrow Y_{\text{próximo ponto}} = Y_{\text{ponto atual}}$

$$\xi = 2(X_{\text{ponto atual}} + 1)\Delta Y + 2\Delta Y - 2\Delta X(Y_{\text{ponto atual}}) - \Delta X$$

ou seja:

$$\xi_{\text{novo}} = \xi_{\text{atual}} + 2\Delta Y$$

Assim, se $\xi \geq 0$ a reta está mais próxima de T então incrementa-se Y, se $\xi < 0$ a reta está mais próxima de S, então Y não é incrementado.

2.1.3.2 - Traçado de Círculos em Dispositivo do Tipo "Raster"

Para o traçado de círculos pode-se utilizar as equações trigonométricas que o descrevem:

$$X_{n+1} = X_n\cos\theta + Y_n\text{sen}\theta$$
$$Y_{n+1} = Y_n\cos\theta - X_n\text{sen}\theta$$

Onde n e n+1 são os incrementos entre dois "píxels" e θ o deslocamento angular entre ambos.

Escola Federal de Engenharia de Itajubá - EFEI
Biblioteca MAUÁ BIM

Porém, a utilização destas expressões pode resultar em erros acumulativos, devido aos arredondamentos das operações envolvidas.

Bresenham desenvolveu um algoritmo incremental para a geração de arcos de circunferência, usando apenas operações de adição e subtração, semelhante ao algoritmo desenvolvido para linhas [18, 19]. Por ser complexa sua demonstração, será apresentada somente a sua idéia básica.

A figura 2.6 ilustra o princípio do algoritmo:

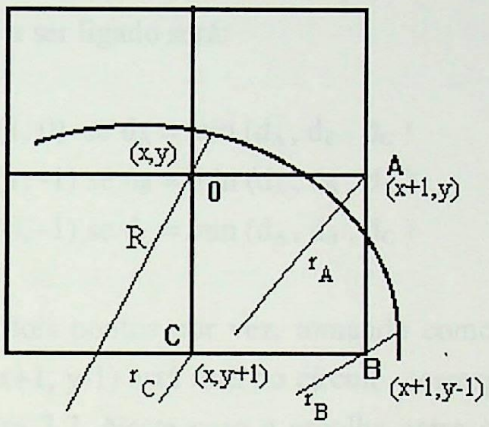


FIGURA 2.6

Estando o ponto 0 representado, o problema é qual dos pontos, A, B ou C marcar para representação da circunferência no próximo incremento do algoritmo.

A determinação de qual dos três "píxels" deverá ser ligado parte de uma análise de erro mínimo, calculando-se qual dos três pontos leva ao valor mais próximo de R, o raio da circunferência.

Considerando que a circunferência está centrada no ponto zero e que um determinado ponto (X,Y) encontra-se sobre a circunferência, podemos dizer que:

$$X^2 + Y^2 = R^2$$

E que a distancia entre este ponto e a circunferência é zero, ou seja:

$$d_{ponto} = X^2 + Y^2 - R^2 = 0$$

Assim, para os pontos A, B e C podemos dizer que:

$$d_A = x_A^2 + y_A^2 - R^2$$
$$d_B = x_B^2 + y_B^2 - R^2$$
$$d_C = x_C^2 + y_C^2 - R^2$$

Quanto mais próximo de zero estiver d, mais próximo da circunferência estará o ponto. Então, o "pixel" a ser ligado será:

$$(1, 0) \text{ se } d_A = \min (d_A, d_B, d_C)$$
$$(1, -1) \text{ se } d_B = \min (d_A, d_B, d_C)$$
$$(0, -1) \text{ se } d_C = \min (d_A, d_B, d_C)$$

O algoritmo avalia dois pontos por vez, tomando como partida o ponto B. Se $d_B > 0$ então o ponto (x+1, y-1) está fora do círculo, correspondendo aos casos dos círculos 1 e 2 da figura 2.7. Neste caso a escolha entre os pontos B e C é feita determinando-se qual deles é o mais próximo da circunferência real. Da mesma forma, se $d_B < 0$, a escolha fica entre os pontos A e B (casos dos círculos 3 e 4 da figura 2.7).

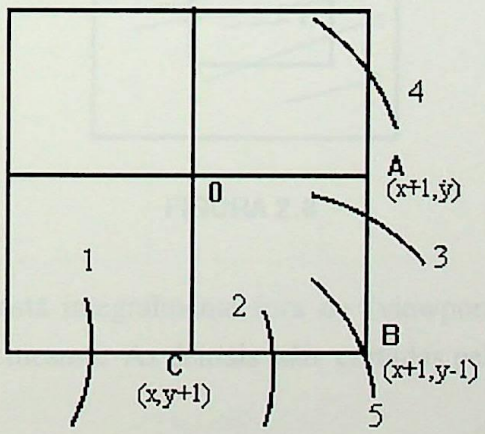


FIGURA 2.7

A vantagem deste algoritmo é que os cálculos de distância, originalmente envolvendo operações de multiplicação se reduzem a fórmulas incrementais simples, utilizando somente operações de adição e subtração.

2.1.4 - Recorte

Denomina-se recorte à metodologia para limitar o desenho em uma "viewport", recortando assim a figura geométrica fora dos limites estabelecidos pela mesma.

Um algoritmo bastante difundido para recorte de linhas, aplicável a figuras geométricas, foi desenvolvido por Dan Cohen e Ivan Sutherland, sendo, por isso, denominado algoritmo de Cohen-Sutherland. [3, 6]

O algoritmo determina quais linhas pertencem à "viewport" e também, os pontos de interseção das linhas com as arestas da mesma.

Tomando como exemplo a figura 2.8:

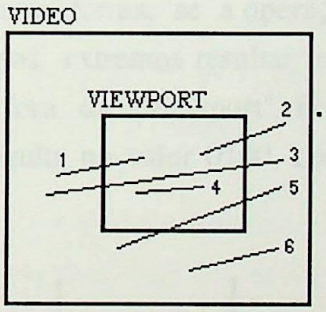


FIGURA 2.8

vemos que a reta 6 está integralmente fora da "viewport" e que a reta 4 está integralmente dentro da mesma. As demais são cortadas pelas suas arestas.

O algoritmo define 9 regiões no vídeo e para cada região associa 4 "bits", como mostra a figura 2.9:

1001	1000	1010
0001	0000	0010
0101	0100	0110

FIGURA 2.9

Cada "bit" define como a região está posicionada com relação à "viewport", ou seja:

1° bit = 1, se está acima
2° bit = 1, se está abaixo
3° bit = 1, se está a direita
4° bit = 1, se está a esquerda

Assim, a cada ponto do desenho é associado um conjunto de 4 "bits".

Se os dois pontos extremos da reta possuem o código 0000, então a reta está cortando a "viewport". Da mesma forma, se a operação lógica "AND", executada entre os códigos dos dois pontos extremos resultar em valor diferente de zero, a linha encontra-se totalmente fora da "viewport". Por exemplo, para dois pontos nas regiões 0100 e 0110, resulta no valor 0100, conforme pode ser visto na figura 2.10:

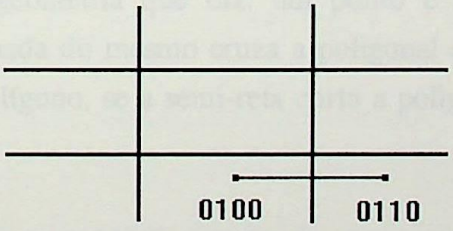


FIGURA 2.10

Aplicando-se esta metodologia consegue-se eliminar a maior parte das linhas. As demais, podem ou não pertencer à "viewport", como ilustrado na figura 2.11:

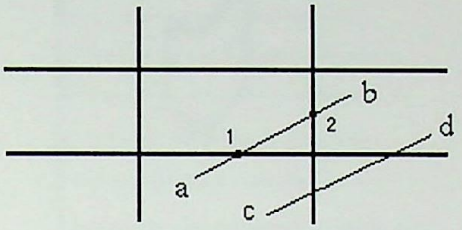


FIGURA 2.11

Para a reta $\overline{ab}$, basta determinar os pontos 1 e 2 de interseção com as arestas da "viewport" e aplicar o método anterior aos segmentos resultantes. Desta forma, determina-se que somente o segmento 1-2 pertence à "viewport". Ao aplicar-se esse método à reta $\overline{cd}$, verifica-se que todos os segmentos resultantes são externos à "viewport".

2.1.5 - Preenchimento de Polígonos

Será apresentado o método de preenchimento por linhas de varredura (método "Scan-Line") [3], por ser simples e de boa eficiência.

Este método é aplicado apenas em polígonos fechados, para que seja utilizada uma propriedade básica da geometria que diz: um ponto é interno ao polígono, se qualquer semi-reta emanada do mesmo cruza a poligonal em um número ímpar de vezes, e é externo ao polígono, se a semi-reta corta a poligonal em um número par de vezes.

Assim, pelo critério de paridade, cada linha de varredura da tela irá cortar a poligonal de fronteira em um número par de pontos. Se tomarmos estes pontos de cruzamento, ordenados aos pares pelas abscissas, podemos afirmar que os

segmentos que os unem estão contidos no interior da área. A figura 2.12, a seguir ilustra:

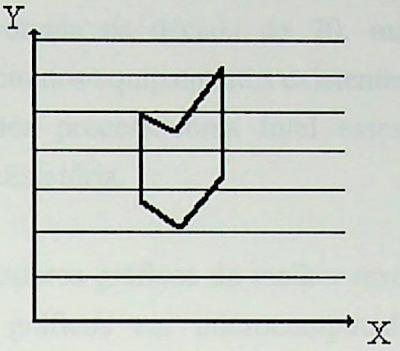


FIGURA 2.12

A estratégia de preenchimento baseada neste critério é feita através da montagem de uma lista com os pontos de interseção das linhas de varredura com as arestas do polígono, formando a lista de arestas ativas.

A coleção de pontos computados é classificada pela ordenada y e, para pontos de mesma ordenada, pela abscissa x. Os pontos assim classificados são considerados aos pares, que determinam os extremos dos segmentos horizontais a serem traçados. As arestas horizontais da poligonal podem ser simplesmente ignoradas.

Assim, logo que uma aresta deixa de ser ativa, ela é retirada da tabela, e assim que a tabela estiver vazia é considerado terminado o processo.

2.1.6 - Considerações

Como pode ser observado na descrição dos métodos apresentados, o esforço computacional para desenho de figuras em dispositivos gráficos é bastante elevado, uma vez que é necessário um grande número de operações para determinar se um "pixel" deve permanecer apagado ou ser aceso para formar uma imagem.

Por esta razão, a utilização dos recursos gráficos em microcomputadores padrão PC levou bastante tempo para ser disseminada, não por falta de algoritmos, uma vez que a maioria foi desenvolvida na década de 70, mas sim devido à pequena capacidade de processamento dos equipamentos existentes. Somente com o início da utilização da série 386 dos processadores Intel esses algoritmos puderam ser implementados de forma satisfatória.

A inexistência de adaptadores gráficos de melhor resolução também dificultou a implantação de sistemas gráficos em microcomputadores, principalmente com relação aos adaptadores de vídeo, que somente se tornaram satisfatórios após a definição do padrão VGA, em meados da década de 80.

2.2 - ESTRUTURA BÁSICA DO PROGRAMA

Este item apresenta a estrutura global do programa, e procura fazer uma abordagem apenas dos principais conceitos utilizados na sua elaboração.

O programa é constituído por seis blocos básicos, que são:

- Inicialização;
- Leitura/Gravação dos Dados;
- Edição Gráfica;
- Execução;
- Gerenciamento da Transformação de Coordenadas;
- Impressão dos Resultados.

Convém ressaltar que todas as funções referenciadas a seguir pertencem à linguagem utilizada, Turbo C, mas possuem equivalentes em outras linguagens de programação que dispõem de biblioteca gráfica.

2.2.1 - Inicialização

O principal objetivo da inicialização do programa é obter os parâmetros do dispositivo de exteriorização em vídeo, através de chamadas a funções do Turbo C (transparente para o usuário), armazenando-os nas variáveis *MaxX* e *MaxY*.

Estas variáveis são utilizadas em todo o programa e são responsáveis pela portabilidade do "software" entre diferentes adaptadores de vídeo (CGA, EGA e VGA).

Se os valores *MaxX* e *MaxY* , obtidos através das funções getmaxx( ) e getmaxy( ) do Turbo C, forem 640 e 200 respectivamente, o computador está equipado com adaptador CGA e todo o mapeamento daí em diante será feito dentro dessas coordenadas de vídeo.

É também responsável por inicializar o sistema no modo gráfico, através da função initgraph( ) do Turbo C, que instrui a BIOS (Basic Input/Output System) do sistema operacional a passar a utilizar o modo de vídeo 12 Hexa, no caso do adaptador VGA (modo 640x480 - 16 cores) [10].

2.2.2 - Leitura/Gravação dos Dados

O armazenamento dos dados foi separado em dois arquivos, um contendo os dados necessários a apresentação gráfica do sistema e outro contendo os parâmetros elétricos necessários ao processamento do fluxo de carga.

Essa separação foi adotada para que se possa processar sistemas elétricos com um maior número de barras, sendo a apresentação gráfica limitada a uma sub-área do mesmo.

Com isso, áreas do sistema que não interessam diretamente à análise (como áreas externas) podem ser excluídas da representação gráfica, sendo porém levadas em conta no processamento do fluxo de carga.

Outra vantagem desta abordagem é aproveitar-se de arquivos já existentes de parâmetros elétricos, sendo necessária assim somente a edição do arquivo gráfico.

O arquivo de dados elétricos é armazenado no formato ASCII, o que permite sua edição, além da interface do programa (entenda-se por interface a interação gráfica com o usuário), por qualquer "software" de edição de textos. Já o arquivo de dados gráficos, por ser restrito à utilização pelo programa, é armazenado na forma binária (para redução do tamanho do arquivo), e desta forma sua edição é restrita à interface do programa.

2.2.3 - Edição Gráfica

As rotinas de edição permitem a inclusão dos seguintes equipamentos na forma gráfica:

- Barra;
- Linha;
- Gerador;
- Carga;
- Capacitor;
- Reator;
- Disjuntor;
- Seccionador;
- Transformador

que são os equipamentos normalmente necessários para a composição dos sistemas elétricos. A inclusão de outros equipamento, além destes, pode ser executada com pequena alteração nas rotinas de desenho e edição.

O posicionamento do equipamento pode ser feito em qualquer ponto da tela. Após seu posicionamento, é solicitada a referência que o liga ao dado elétrico, como, por exemplo, o número da barra ao qual está conectado eletricamente. É possível ainda a alteração de cor do equipamento, o que possibilita a diferenciação visual dos níveis de tensão do sistema. É também possível efetuar alterações em sistemas já representados como, por exemplo, a movimentação ou o apagamento de um grupo de equipamentos.

A movimentação é realizada apenas com alteração das coordenadas do equipamento e o apagamento é feito com a ativação de um "flag" na estrutura de dados do equipamento. Quando o arquivo é gravado no final da edição, este equipamento é excluído do sistema armazenado no arquivo gráfico.

Esta forma de tratamento foi adotada para evitar manipulações na estrutura de armazenamento dos dados gráficos dos equipamentos, o que envolve pesquisas que são onerosas, em termos de tempo de computação e considerando que as alterações em arquivos existentes devidas ao apagamento de equipamentos, ocorrem em número reduzido. Porém, a principal vantagem desta metodologia é que os equipamentos removidos acidentalmente podem ser recuperados durante a edição, uma vez que os mesmos continuam nos vetores de dados.

2.2.4 - Execução

O cálculo do fluxo de carga do sistema elétrico é feito externamente à interface, através da execução de um programa escrito na linguagem Fortran, que utiliza o método Desacoplado Rápido de resolução [17].

A interface cria, a cada execução, um arquivo temporário para ser lido pelo programa de Fluxo de Carga, que exclui as linhas abertas pelo usuário, bem como alterações nos valores de carga e geração, possibilitando a simulação de contingências no sistema.

Da mesma forma, após o final do processamento, o programa de Fluxo de Carga cria um arquivo temporário que é lido pela interface e exteriorizado no ambiente gráfico para o usuário.

A execução externa do fluxo de carga possibilita uma grande facilidade na utilização deste programa, que pode ser elaborado e modificado utilizando-se qualquer linguagem de programação, independentemente da linguagem utilizada para a interface.

O inconveniente desta abordagem ocorre em "hardwares" com tempo de acesso a disco elevado (o tempo de gravação e leitura dos arquivos temporários passa a ser preponderante), quando da representação de sistema elétricos de grande porte, ou seja, com muitas barras. Porém, através de testes realizados em computadores tipo PC XT, onde o tempo de acesso a disco é bastante elevado, não houve constatação de tempo elevado o suficiente para inviabilizar a utilização desta metodologia.

2.2.5 - Gerenciamento da Transformação de Coordenadas

Esta parte do programa é a responsável pelo efeito de aproximação/afastamento (efeito "zoom") na visualização gráfica.

Todos os pontos do sistema são mapeados e armazenados utilizando as coordenadas globais 0 a 10.000 nos eixos x e y. Essas coordenadas são atribuídas às variáveis *MaxXwc* e *MaxYwc*, que são utilizadas pelo programa sempre que necessária a conversão entre as coordenadas do dispositivo (dependente do adaptador gráfico) e as coordenadas globais utilizadas (0 a 10.000).

A definição da região da janela em coordenadas globais a ser apresentada (ou seja, a área do sistema elétrico a ser visualizada) é feita através das variáveis *xdmin*, *xdmax*, *ydmin* e *ydmax*, conforme mostra a figura 2.13:

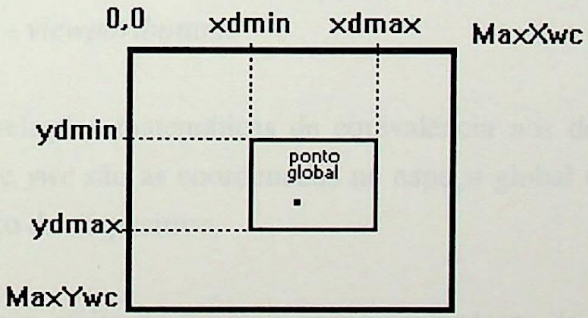


FIGURA 2.13

Esta janela é apresentada em uma "viewport" no vídeo (área delimitada do vídeo), que é determinada pela função setviewport( ) do Turbo C e que atribui às variáveis *viewportleft*, *viewportright*, *viewporttop* e *viewportbottom* os limites para desenho no vídeo, como se pode visualizar na figura 2.14:

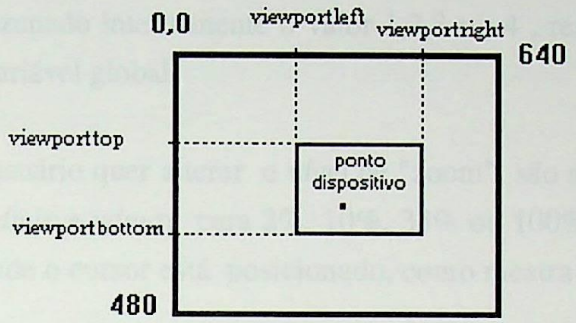


FIGURA 2.14

Assim, foram desenvolvidas funções de conversão de um ponto em coordenadas globais para coordenadas de dispositivos, que implementam as relações:

$$xwc = \frac{xdc - viewportleft}{viewportright - viewportleft} . (xdmax - xdmin) + xdmin$$

$$ywc = \frac{ydc - viewporttop}{viewporttop - viewportbottom} . (ydmax - ydmin) + ydmin$$

e que são obtidas das relações matemáticas de equivalência nos dois sistemas de coordenadas, onde *xwc* e *ywc* são as coordenadas no espaço global e *xdc* e *ydc* são as coordenadas no espaço do dispositivo.

Para que seja obtido o efeito de "zoom", apenas os valores de *xdmin*, *xdmax*, *ydmin* e *ydmax* são alterados, ou seja, a janela de visualização no espaço global é aumentada ou diminuída. Com isto, o usuário tem a sensação de estar vendo o

Escola Federal de Engenharia de Itajubá - EFEI
Biblioteca
MAUÁ
BIM

sistema elétrico mais próximo ou mais longe, uma vez que uma maior ou menor porção do mesmo é apresentada na tela.

Foram definidos quatro níveis de apresentação do sistema: 2%, 10%, 33% e 100% do espaço global. Para se saber em que nível de "zoom" está a apresentação no momento é armazenado internamente o valor 1,2,3 ou 4 , respectivamente para cada nível, em uma variável global.

Assim, quando o usuário quer alterar o nível de "zoom", são alterados os valores de *xdmin* , *xdmax* , *ydmin* e *ydmax* para 2%, 10%, 33% ou 100% do espaço global, em torno do ponto onde o cursor está posicionado, como mostra a figura 2.15:

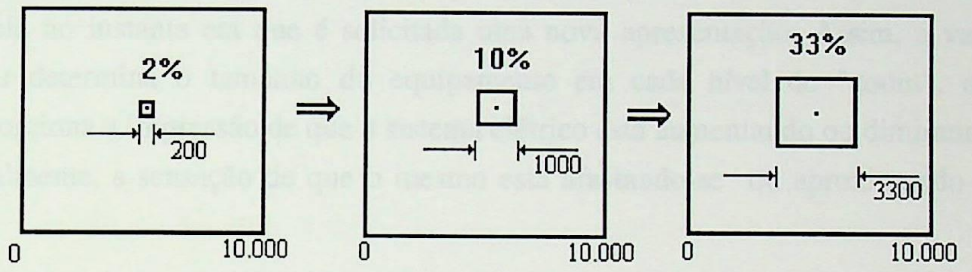


FIGURA 2.15

A expressão utilizada, no caso de uma aproximação de 10%, por exemplo, é:

$$xdmin = xwc - 0.1*MaxXwc/2$$

onde *xwc* é a abscissa do ponto central, em coordenadas globais, da área do sistema elétrico que se deseja visualizar. Como descrito no início deste item, sendo adotado o valor *MaxXwc* =10.000, temos:

$$xdmin = xwc - 500$$

As expressões equivalentes paras as outras variáveis, por analogia, são:

$$xdmax = xwc + 500$$
$$ydmin = ywc - 500$$
$$ydmax = ywc + 500$$

Com isso tem-se uma janela 1000x1000 sobre o ponto desejado.

Para os outros níveis de "zoom" basta alterar o fator de multiplicação de *MaxXwc* para 0.02, 0.33 e 1, respectivamente.

Para completar o efeito de "zoom" é definida a variável auxiliar *offset* cujo valor altera-se com o nível de "zoom" e é utilizada para desenhar os equipamentos. Todos os equipamentos são armazenados apenas pelo seu ponto central e são desenhados na tela no instante em que é solicitada uma nova apresentação. Assim, a variável *offset* determina o tamanho do equipamento em cada nível de "zoom", o que proporciona a impressão de que o sistema elétrico está aumentando ou diminuindo, e visualmente, a sensação de que o mesmo está afastando-se ou aproximando-se da tela.

2.2.6 - Impressão dos Resultados

A impressão dos resultados pode ser feita na forma gráfica ou na forma texto.

Para impressão na forma gráfica, é feito apenas um "hardcopy" da tela para a impressora. Para isto, são utilizadas as funções getpixel( ) e fprintf( ) do Turbo C, ou seja, é verificada a condição (apagado/aceso) de cada "píxel" da tela, sendo a informação enviada para a impressora. Assim, se um "píxel" estiver aceso, aparece um ponto impresso; se estiver apagado, o papel permanece em branco. Desta forma, o efeito de cores não é percebido na impressão. Este procedimento está descrito em [4].

A tela é percorrida no sentido horizontal, sendo verificados 8 "píxels" verticais de cada vez, informações que na impressora, controlam as agulhas 0 a 7 da cabeça de impressão. Assim, existe uma limitação da utilização do "software" apenas

impressoras matriciais com cabeça de 9 pinos. Para utilização em outro tipo de impressora é necessária a alteração do código fonte do programa. Ressalta-se, porém, foi possível a impressão no modelo Epson de 24 agulhas, pois este tipo de impressora permite o controle de apenas 9 agulhas, devido à mesma ter mantido compatibilidade com estes modelos.

Para impressão em texto é criado um arquivo em formato ASCII pelo programa de fluxo de carga, que é apenas lido pela interface e apresentado no vídeo ou na impressora. Este arquivo contém a apresentação dos dados de entrada, um resumo da configuração do sistema, um sumário de desvios e limites violados e, por fim, os resultados de barras e de linhas, tais como tensão, geração de reativos e fluxos nas linhas.

2.3 - DETALHAMENTO DO PROGRAMA

Para facilitar o desenvolvimento do programa, o mesmo foi dividido em módulos, que foram sendo agregados à medida em que novas funções eram desenvolvidas.

Os principais módulos são:

MÓDULO	FUNÇÕES	
PRINCIP.C	Main	SaveImage
	Initialize	ReadImage
	TelaInic	gprintf
	TelaPrinc	Mensagem
	Contorno	JanelaSaida
FMOUSE.C	Mreset	Mmotion
	Mpos	Mmoveto
	Mshow	Mresetbut
	Mxlimit	Mcursorcross
	Mylimit	

LER.C	GerenciaLer	Converte
	LerGraf	LerElet
	FixaPontoFlutuante	
GRAVAR.C	GerenciaGravar	Erasestr
	GravarGraf	Letexto
	GravarElet	MenuNomeArquivo
EDIÇÃO.C	Editar	Posiciona
	Desenhar	Sentido
	OpçãoCor	DesenhaLinha
DESENHO.C	Disj	Traf
	Ger	Reat
	Cap	Carga
	Barra	Linha
	Carrega	Desist
ALTERA.C	Apaga	Move
	LocalizaEquip	
MODIFICA.C	Modificar	Drawbutton
	IncluiBarra	JanelaModifica
	AlteraTitulo	DadoGráfico
	AbreDisjuntor	MenuLinhaLocal
	MenuBarraLocal	
MODIFIC1.C	IncluiLinha	
MENUTECL.C	JanelaMenu	MenuEquipBarConect
	MenuLinha	MenuBarra
EXECUTAR.C	Executar	LerSaidaFlow

TRANSFOR.C	xdcwc	ydcwc
	xwcdc	ywcdc
	compxwcdc	compxdcwc
	compywcdc	compydcwc
	Zoom	Janela
	Scrollbar	
IMPRIME.C	Imprimir	OpçãoGraf
	OpçãoText	PrintGraf
	ImprimirText	MostrarVideo

A seguir, será apresentado o detalhamento das principais funções do programa, procurando apresentar as principais lógicas de controle utilizadas.

2.3.1 - Funções do Módulo PRINCIP.C

Este módulo define as variáveis globais utilizadas e as funções que fazem o controle principal do programa.

As principais variáveis globais definidas são as estruturas de armazenamento dos dados dos equipamentos, definidas no cabeçalho ESTRUT.H. Cada tipo de equipamento apresentado na tela (carga, reator, barra, etc) possui, armazenadas em uma estrutura (indexada do elemento zero ao número máximo de equipamentos permitido), suas coordenadas no eixo x e no eixo y em valores globais (0 a 10.000), sua cor para desenho, o seu alinhamento (horizontal/vertical), se está aberto/fechado etc.

Assim, por exemplo, a estrutura de armazenamento do equipamento barra é:

```
struct Barra { int      Numelet;
               int      PosX;
               int      PosY;
               int      Comp;
               unsigned Esp:1;
               unsigned Hor_vert:1;
               unsigned Apagado:1;
               unsigned Cor:4;  }   Bar_Graf[MaxBar];
```

ou seja, é armazenado o número elétrico a que se refere este desenho, as coordenadas x e y do ponto inicial de desenho da barra, seu comprimento e espessura, a posição horizontal/vertical, se foi apagada na edição corrente e a cor de desenho.

Nota-se que é utilizado um campo de "bit" para armazenar a espessura, direção de desenho, deleção e cor, uma vez que foi arbitrado que uma barra tem apenas duas espessuras (0 ou 1), a direção é apenas horizontal ou vertical (0 ou 1), está apagada ou não (0 ou 1), e todos os desenhos poderão ter apenas 16 cores (o que pode ser armazenado em apenas 4 "bits"). Com isto é reduzida a necessidade de memória para armazenamento da parte gráfica.

É também definido no cabeçalho DEFINE.H o número máximo de equipamentos suportados pelo programa, com base na limitação da memória em modo real do microcomputador padrão PC, que é 640 kilobytes. Este dimensionamento leva em consideração ainda o espaço disponível necessário para executar, posteriormente, o programa de fluxo de carga em Fortran.

Assim, foram definidas as seguintes limitações:

Barras Gráficas	=	100
Barras Elétricas	=	100
Linhas Gráficas	=	1000
Linhas Elétricas	=	200
Disjuntores	=	400

Seccionadores	=	400
Capacitores	=	30
Reatores	=	20
Transformadores	=	50
Geradores	=	30
Cargas	=	100
Carregamentos	=	400
Objetos Gráficos	=	2530

Outras variáveis globais definidas neste módulo e já descritas nos itens 2.2.1 e 2.2.5 são:

MaxX	*xdmin*
MaxY	*xdmax*
MaxXwc	*ydmin*
MaxYwc	*ydmax*

Algumas variáveis globais para controle do programa também são definidas neste módulo:

inout: informa, em qualquer ponto do programa, em que nível de "zoom" está o desenho;

janela: informa se o desenho está sendo feito na janela principal (tela cheia), ou na janela de pesquisa (canto superior direito);

offset: define, para cada nível de "zoom", qual o distanciamento dos pontos para desenho, ou seja, aumenta ou diminui um desenho para dar o efeito de "zoom".

Neste módulo é feito ainda o controle principal do programa, através da atuação do cursor de "mouse" sobre o menu de botões, em uma estrutura de "loop", como pode ser exemplificado:

```
do {
      Mpos( );
      switch (m.button_status)
      case 1:
            if (m.xaxis > 0 && m.xaxis < 0.1*MaxX &&
               m.yaxis> 0.02*MaxY &&m.yaxis < 0.107*MaxY)
             { GerenciaLer();              /* cursor sobre o botao LER */
               break;}
         ...
            if ( ... )                     /* cursor sobre botão RETORNO */
             { ch='C';}
      } while (ch!='C');
```

Este tipo de estrutura é utilizado para gerenciar toda a espera por um comando de execução. A função Mpos( ) retorna a posição do cursor através da estrutura:

```
struct Mstatus { int button_status;
                 int button_count;
                 int xaxis;
                 int yaxis;
                 int x_count;
                 int y_count;  } m;
```

ou seja, a posição do cursor no momento em que é apertado um botão do "mouse" e qual esse botão. Com isto, através de testes condicionais "if", verifica-se qual função deve ser executada.

O desenho dos botões é armazenado no arquivo MENUPRIN.IMG, que é um arquivo tipo "bitmap" criado pela função SaveImage( ) e lido pela função ReadImage( ) definidas neste módulo.

A opção por guardar a imagem de botões em um arquivo "bitmap", e não por desenhá-los no momento em que é solicitada a montagem da tela, é devida apenas à redução no número de linhas de código do programa. Assim, um programa auxiliar

desenha previamente todos os botões necessários ao programa e, a partir daí, já não é mais necessária a sua execução, o que diminui o tamanho do programa executável.

2.3.2 - Funções do Módulo FMOUSE.C

As funções definidas neste módulo são responsáveis pelo gerenciamento do movimento do cursor de "mouse" na tela e pelo controle de seus botões.

É utilizado o "driver" de controle de "mouse" fornecido juntamente com os "mouses" comerciais padrão.

Para que a portabilidade do "software" ocorra, são utilizados apenas dois botões de "mouse" (padrão Microsoft) para o controle do programa, o que permite sua utilização com "drivers" de "mouse" com três botões (padrão Logitech).

O acesso ao "mouse" é feito acionando-se a interrupção 33 hexa do DOS, que executa o "driver". A comunicação entre o programa e o "driver" é feita através de valores adequados ajustados nos registros AX, BX, CX e DX da CPU.

Assim, se é passado o valor 0 (zero) ao registro AX, o "driver" executa "reset" no "mouse" e retorna, no próprio registro AX, a informação se o "mouse" está presente no equipamento e, no registro BX, se é de 2 ou 3 botões.

Se é passado o valor 1 ao registro AX, o "driver" mostra o cursor de "mouse" na tela. Se é passado o valor 2, o cursor é apagado.

Assim, para cada valor no registro AX, uma função é desempenhada pelo "driver", sendo utilizadas no programa as seguintes funções:

valor 3 → retorna a posição atual do cursor na tela e o "status" dos botões (se algum foi apertado e qual);

valor 4 → move o cursor para uma determinada posição na tela;

valor 7 → fixa limites para o movimento do cursor no eixo x;

valor 8 → fixa limites para o movimento do cursor no eixo y;

valor 9 → altera a forma do cursor (seta, cruz, ampulheta, etc). Foi implementada, nesta aplicação, apenas a alteração de seta para cruz, para o desenho de linhas;

valor 11 → retorna o comprimento deslocado pelo "mouse", desde a última chamada desta função;

Está disponível ainda no programa a opção de simulação do "mouse" através do teclado, quando aquele não está presente no equipamento. Assim, uma imagem de cursor é apresentada na tela e deslocada através das teclas de seta, sendo os botões do "mouse" simulados pelas teclas + ou -. Esta simulação é feita de forma simples, através de solicitação ao teclado e de condições "if", determinando qual tarefa será executada.

A comunicação ao programa dos valores retornados pelo "driver" de "mouse" é feita através da estrutura Mstatus, descrita no item 2.3.1.

2.3.3 - Funções do Módulo LER.C

As funções deste módulo são responsáveis pela leitura dos dados gráficos e elétricos.

A leitura de dados do programa é dividida em duas funções, que lêem os dados gráficos (LerGraf) e elétricos (LerElet) separadamente, e são gerenciadas pela função GerenciaLer( ).

A função básica de GerenciaLer( ) é fornecer ao usuário a opção de ler uma nova base de dados, sobrepondo-a à atual, ou de acrescentar dados à base

atualmente em uso. Isto é feito apenas com a reinicialização dos indexadores da estrutura de dados gráficos, quando a opção LER CASO NOVO é escolhida. Quando a opção ACRESCENTAR AO ATUAL é escolhida, os indexadores são mantidos nos valores atuais e assim os novos dados lidos são acrescentados aos existentes pelo incremento dos indexadores associados às estruturas de dados dos equipamentos.

Uma restrição a este tipo de acréscimo ocorre se houver numeração de barra repetida, o que provocará erro no processamento do fluxo de carga. Outro erro que ocorrerá é a superposição de equipamentos desenhados sobre uma mesma coordenada.

Assim, o programa não prevê a verificação de consistência para este tipo de erro, deixando a critério do usuário, através do conhecimento da base de dados, o gerenciamento da superposição. Esta opção foi adotada devido à verificação de consistência já estar implementada no processamento do fluxo de carga, porém, poderá também ser incorporada ao programa nesta etapa, com pequenas alterações nas rotinas de leitura dos dados.

A rotina LerGraf( ) é responsável pela leitura dos dados gráficos, que são armazenados em um arquivo com extensão .GRA, na forma binária.

O arquivo é lido na mesma sequência em que é criado. Após a leitura da estrutura de cada equipamento, são armazenadas no vetor *LocalizaEquip* (descrito no item 2.3.7) as coordenadas do mesmo. Este vetor é utilizado pelo programa sempre que é preciso localizar um equipamento na tela, para sua alteração ou movimentação.

Esta rotina lê o arquivo de dados elétricos no formato do programa de fluxo de carga conhecido como PECO e utilizado pelas empresas concessionárias brasileiras, aceitando como controle apenas os cartões 1 (título), 6 (controle de processamento), 5 (dados de barras), 4 (dados de linhas) e 30 (fim de leitura), que devem ser inseridos nesta ordem.

Escola Federal de Engenharia de Itajubá - EFEI
Biblioteca MAUÁ BIM

Para que a leitura seja realizada no mesmo padrão e formato da linguagem FORTRAN, foi necessária uma série de adaptações no padrão de leitura da linguagem C. Assim, os dados são lidos como caracteres ASCII e convertidos para inteiros/reais após serem testados pela rotina auxiliar FixaPontoFlutuante. Esta rotina testa e ajusta o ponto flutuante de cada campo para o formato FORTRAN do programa PECO. Desta forma, a leitura torna-se totalmente dependente do formato especificado no programa. Uma eventual alteração nesse formato implicaria em reescrever a rotina.

2.3.4 - Funções do Módulo GRAVAR.C

O módulo é iniciado com a função GerenciaGravar( ), que tem apenas a finalidade de solicitar ao usuário o nome do arquivo a ser gravado, sendo as extensões .GRA e .ELE colocadas automaticamente pelo programa, para os arquivos de dados gráficos e elétricos respectivamente.

A função GravarGraf( ) armazena os dados dos equipamentos no arquivo .GRA no formato binário, na forma:

N° de Cargas+(dadoscarga1,dadoscarga2, ... dadoscargaN)+
N° Capacitores+(dadoscapac1,dadoscapac2, ... dadoscapacN)+
...
N° Carregamento+(dadoscarreg1,dadoscarreg2, .. dadoscarregN)

ou seja, é armazenado o número de equipamentos de cada tipo, seguido dos dados dos mesmos em número igual ao especificado. Caso algum erro de armazenamento seja detectado, isto é informado ao usuário.

A função GravarElet( ) tem a atribuição de gravar os dados elétricos para armazenamento histórico e para processamento pelo programa de fluxo de carga.

O arquivo de armazenamento histórico é criado com a extensão .ELE no padrão ASCII e obedece ao formato de dados do programa PECO. Para tal, é necessário

transformar todas as variáveis inteiras/reais em cadeia de caracteres, para que possa ser fixado o tamanho do campo no formato FORTRAN.

O arquivo temporário para processamento do fluxo de carga é criado sempre com o nome ENTRADA.DAD. Assim, uma solicitação de execução do fluxo de carga causa a sobreposição do arquivo temporário criado sobre o anterior.

2.3.5 - Funções do Módulo EDIÇÃO.C

As funções deste módulo permitem a edição do desenho gráfico do sistema elétrico.

A função Editar( ) tem a finalidade de gerenciar a chamada às funções encarregadas de desenhar o sistema elétrico ou de modificar um desenho já existente, através da chamada às funções Apaga( ) e Move( ), descritas no módulo ALTERA.C.

A função Desenhar( ) é a responsável pelo posicionamento dos equipamentos na tela, bem como pela determinação dos atributos de cor, direção e espessura, e pela correlação com os dados elétricos de cada equipamento.

O posicionamento de cada equipamento, com exceção de barras e linhas, é feito da seguinte forma:

Inicialmente deve ser escolhida a região (espaço global) em que se quer desenhar o sistema elétrico, para que sejam definidas as variáveis que delimitam a janela de desenho (*xdmin, xdmax, ydmin e ydmax*) e a variável *offset* que define o tamanho do desenho, através da chamada à função Zoom( ).

Os equipamentos carga, capacitor, reator, transformador, disjuntor e gerador são criados através da seleção dos ícones correspondentes, localizados no canto inferior esquerdo da tela. Isto cria uma imagem, que é transferida para a memória no formato "bitmap". Esta imagem, quando superposta sobre ela mesma, através da

função putimage( ) do TurboC, no modo ou-exclusivo, provoca o seu desaparecimento na tela e o imediato reaparecimento na posição desejada. Assim, a chamada à função putimage( ), de forma coordenada com o deslocamento do cursor, provoca a sensação de deslocamento da imagem no monitor.

Este deslocamento é gerenciado pela função auxiliar Posiciona( ), que utiliza o botão 1 do "mouse" enquanto está deslocando a imagem, e o botão 2 para confirmar o posicionamento final na tela pelo usuário.

Uma vez acionado o botão 2, as coordenadas do ponto posicionado são passadas à função Desenhar( ), que atribui estes valores à estrutura de dados do equipamento (após sua conversão para coordenadas globais) nas variáveis *PosX* e *PosY*.

A função Sentido( ) é a responsável pela inicialização das coordenadas que criam o objeto a ser deslocado na tela, em função de seu sentido de desenho. Por exemplo, o capacitor pode ser desenhado no sentido direita/esquerda ou acima/abaixo, o que determina quatro modos de desenho. Estes modos são armazenados através da atribuição dos valores 0, 1, 2 ou 3 no campo *Pos* da estrutura de dados dos equipamentos.

Finalmente é chamada a função MenuEquipBarConect( ), que solicita ao usuário informar a qual barra elétrica o equipamento está conectado, para que os dados elétricos do mesmo possam ser consultados pelo programa, na estrutura de armazenamento de dados elétricos.

A cor de desenho do equipamento pode ser alterada através da função OpçãoCor( ), que através de um menu e de testes "if", atribui valores de 0 a 15 (16 cores) à variável Cor. Essa variável é utilizada para atribuição do campo Cor nas estruturas de dados dos equipamentos. Para o usuário, esse menu aparece na forma de uma barra com 16 segmentos coloridos, que permite escolher a cor desejada com auxílio do "mouse".

O desenho de barras difere do desenho dos demais equipamentos, pois nele não ocorre o deslocamento da imagem na tela. Seu desenho é feito através da solicitação

apenas do ponto inicial e final da barra, após a escolha de sua espessura e orientação horizontal/vertical na rotina Sentido( ). Assim, são armazenados na estrutura de dados de barra, apenas o ponto inicial do desenho, seu comprimento, espessura e orientação.

O desenho de linhas é mais complexo que o dos demais equipamentos descritos, pois permite que sejam desenhados segmentos de linhas em sequência, ou seja, a formação de polígonos, sendo que cada aresta é armazenada como uma linha independente. A função DesenhaLinha( ), responsável por este gerenciamento, permite que seja visualizado todo o espaço global, através de chamadas às funções Zoom( ) e Scrollbar( ), o que permite que uma linha possa ser desenhada entre quaisquer pontos do espaço global e não apenas no espaço apresentado na tela no instante do início do desenho, como acontece no caso dos demais equipamentos.

2.3.6 - Funções do Módulo DESENHO.C

As funções descritas neste módulo são responsáveis pela montagem do desenho dos equipamentos e do sistema elétrico.

Convém relembrar que cada equipamento é armazenado em sua estrutura de dados apenas pelo ponto central do desenho. Assim, as funções deste módulo desenham o equipamento em torno deste ponto, utilizando segmentos de reta e arcos de círculo.

Por exemplo, o disjuntor é formado por 6 segmentos de reta, que são desenhados a partir do ponto central na forma mostrada na figura 2.16, onde as setas indicam a sequência de desenho dos segmentos:

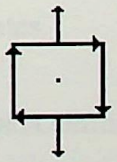


FIGURA 2.16

O tamanho dos segmentos de reta é definido de acordo com o nível de "zoom" em que se encontra o desenho , através de múltiplos da variável *offset*, pré-definida na função Zoom( ).

Após o desenho do equipamento, são apresentados juntamente com o mesmo as grandezas elétricas e os demais atributos associados (potência ativa, reativa, tensão, nome, etc). Isto, porém, somente quando o nível de "zoom" é de 2% do espaço global, pois nos demais níveis não é possível a visualização de caracteres alfanuméricos, devido à redução no desenho.

A função Desist( ) é responsável pelo desenho de todo o sistema, a cada solicitação de redesenho (mudança do nível de "zoom" ou deslocamento na tela).

O sistema é montado através do desenho apenas dos equipamentos que pertencem à janela corrente. Assim, através de testes condicionais "if", são excluídos os equipamentos que possuem coordenadas fora da janela corrente, o que aumenta consideravelmente a velocidade de montagem do sistema, pois embora todos os equipamentos sejam testados, exclui-se a necessidade de desenho dos mesmos.

É testado também se o equipamento foi excluído do sistema através da função Apaga( ), definida no módulo ALTERA.C. Em caso afirmativo, sua apresentação é também excluída.

2.3.7 - Funções do Módulo ALTERA.C

Este módulo possui apenas as funções, Apaga( ), Move( ) e LocalizaEquip( ), que são gerenciadas no módulo EDIÇÃO, pois são utilizadas durante a edição gráfica para modificações em sistemas existentes.

A função Apaga( ) é responsável pela eliminação de um ou vários equipamentos. Esta eliminação não ocorre fisicamente na estrutura de dados dos equipamentos. Para isso, é ajustado para 1 o "bit" definido pela variável *Apagado* no "campo de

"bits"" da estrutura de dados. Ao serem salvos em arquivo os dados correntes, os equipamentos marcados para deleção são excluídos do armazenamento.

Para seleção dos equipamentos a serem eliminados, é solicitado ao usuário que defina, com os botões do "mouse", os vértices superior (direito ou esquerdo) e inferior (oposto ao definido para o vértice superior) do retângulo que contém os equipamentos.

Com estes valores, o programa passa a pesquisar se existem equipamentos dentro do retângulo definido. Esta pesquisa é feita no espaço global 0x10.000, usando como incremento o valor 11, na horizontal e na vertical. Assim, para uma área de procura de 1100x1100 (11% do espaço global) serão necessários 100 incrementos na horizontal e 100 incrementos na vertical, totalizando 10.000 pesquisas.

Embora o número de pesquisas seja elevado, o tempo de processamento foi considerado adequado, mesmo ao se apagar áreas maiores, considerando ainda que não é comum eliminar-se grandes áreas de sistemas previamente elaborados.

A função Move( ) é responsável pela movimentação de um ou mais equipamentos na janela gráfica.

A definição da área a ser deslocada é feita da mesma forma que na função Apaga( ). Após a definição do retângulo a mover, é criada uma imagem do mesmo na memória, através da função getimage( ) do Turbo C. Com a utilização da função putimage( ) do turbo C, coordenada com o deslocamento do "mouse", de forma similar à descrita no item 2.3.5, no módulo EDIÇÃO.C, provoca-se o efeito de movimentação da imagem na tela. O ponto final para deslocamento é informado pela pressão do botão 2 do "mouse".

Ao ser definido o novo ponto para localização dos equipamentos, é calculada a diferença entre as coordenadas anteriores e as atuais. É feita a seguir uma pesquisa em toda a região transportada, de forma idêntica à feita na função Apaga( ), para verificação de quais equipamentos estão contidos na mesma. Após serem obtidos

todos os equipamentos, suas coordenadas são incrementadas com esta diferença, atualizando as variáveis *PosX* e *PosY* das estruturas de dados.

A função LocalizaEquip( ) utiliza a seguinte estrutura de dados:

```
struct Procura { int PosX;
                 int PosY;
                 int Tipo;
                 int Indice; };
```

Esta estrutura é utilizada para armazenar a posição de cada equipamento, através de seu tipo e índice. Assim, quando qualquer outra função necessita obter a informação de qual equipamento está em um determinado ponto no vídeo, basta pesquisar no vetor.

Através do comando switch( ) da linguagem C é realizada a inclusão de um novo equipamento no vetor, ou a pesquisa no vetor existente. Esta pesquisa é feita através da comparação das coordenadas armazenadas no vetor com as coordenadas passadas à função e acrescidas de um incremento (variável *increm*), o que determina uma faixa onde não há possibilidade de existir mais de um equipamento, devido ao seu próprio tamanho. Com isto reduz-se o esforço computacional, durante a pesquisa.

2.3.8 - Funções do Módulo MODIFICA.C e MODIFIC1.C

As funções contidas neste módulo são responsáveis pela alteração dos dados elétricos de linhas e barras, das referências de conexão entre dados elétricos e gráficos, pela alteração do título do caso em estudo e comando para abertura/fechamento de disjuntor/seccionador.

O controle da chamada das funções de modificação é realizado pela função Modificar( ) que gerencia as funções responsáveis pelas alterações de dados gráficos e elétricos.

A apresentação do menu de opções na forma de botões é realizada pela função auxiliar JanelaModifica( ) que desenha os botões com auxílio da função Drawbutton( ), através da superposição de três figuras geométricas: um retângulo maior, um polígono e um retângulo menor, conforme pode ser visualizado na figura 2.17:

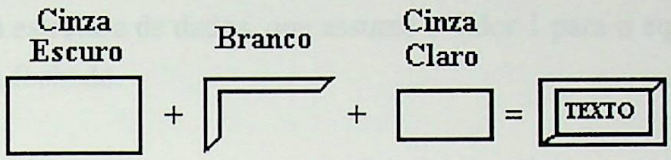


FIGURA 2.17

A alteração de dados de barra é realizada pela função IncluiBarra( ), que cria um menu gráfico onde todos os campos de um cartão de dados no formato utilizado pelo programa PECO podem ser incluídos ou alterados.

Esta rotina é iniciada com a criação de uma janela com todos os campos discriminados e vazios, apenas com a utilização das funções outtextxy( ) e bar( ) do Turbo C. A seguir, a rotina espera pela entrada do número da barra pelo usuário. Quando este número é digitado, a rotina entra em um "loop", pesquisando se esta é uma nova barra ou uma já existente.

Se for encontrada a barra, todos os dados da mesma são apresentados nos seus respectivos campos e a rotina entra em um novo "loop", esperando do usuário a opção de qual campo deve ser alterado, pela pressão do botão 1 do "mouse". Se o botão 2 é pressionado, a rotina é encerrada.

Se a pesquisa nas barras existentes não resultar em sucesso, é considerado que o usuário deseja entrar com os dados de uma nova barra. Então, a função passa a aceitar somente o preenchimento dos campos vazios, em ordem seqüencial, até a entrada de dados em todos os campos, retornando no final ao menu de gerenciamento de modificações.

A função AlteraTitulo( ) apenas abre uma janela com chamada à função JanelaMenu( ) do módulo MENUTCL.C, espera a digitação do novo titulo pelo usuário, retornando após fechar a janela na tela.

A função AbreDisjuntor( ) é responsável pela mudança de estado do disjuntor ou seccionador. Isto é feito através da alteração do valor da variável *Ab_Fech* (unsigned:1) da estrutura de dados, que assume o valor 1 para o equipamento aberto e o valor 0 para fechado.

Esta função solicita do usuário as coordenadas do disjuntor/seccionador, através da pressão do botão 1 do "mouse" sobre o mesmo. Estas coordenadas são passadas à função LocalizaEquip( ) do módulo ALTERA.C, que devolve o tipo de equipamento nesta posição. Caso não seja um disjuntor ou seccionador, isso é informado ao usuário e a função retorna. Caso seja um desses equipamentos, é alterado o valor da variável *Ab_Fech* e redesenhado o sistema, já com o equipamento na nova condição.

A função DadoGráfico( ) é responsável pela alteração da variável (int) *BarConect* na estrutura de dados dos equipamentos carga, capacitor, reator, gerador, e da variável *NumElet* na estrutura de dados de barra. Estas variáveis são responsáveis pelo relacionamento entre dados gráficos e elétricos. É através destas variáveis que o desenho do equipamento busca os valores de carregamento (MW e MVAr), tensão e ângulo, nome da barra, etc, no arquivo de dados elétricos.

A obtenção da coordenada do equipamento é similar à feita na função AbreDisjuntor( ), sendo o valor retornado pela função LocalizaEquip( ) direcionado pelo controle switch( ) da linguagem C, para cada tipo de equipamento. São utilizadas as rotinas auxiliares MenuBarraLocal( ) e MenuLinhaLocal( ) para a digitação dos novos valores de referência pelo usuário. Estas rotinas abrem uma janela onde são exteriorizadas as referências atuais e obtém o novo valor digitado pelo usuário, através da função auxiliar Letexto( ), do módulo GRAVAR.C.

O módulo MODIFIC1.C contém apenas a função IncluiLinha( ), que é similar à função IncluiBarra( ), porém sendo os campos da mesma associados aos campos do cartão de dados de linha do formato utilizado no programa PECO.

2.3.9 - Funções do Módulo MENUTECL.C

As funções descritas neste módulo são funções auxiliares, chamadas pela função Desenhar( ), do módulo EDIÇÃO.C e têm a finalidade de criar janelas para entrada da referência dos dados elétricos associados aos desenhos gráficos dos equipamentos.

São definidas as funções MenuBarra( ) e MenuLinha( ), que possuem o mesmo tipo de construção, porém, a primeira solicita do usuário apenas o nome da barra ao qual o equipamento está conectado (para carga, capacitor, reator e gerador) e a segunda solicita o número das barras nas extremidades do equipamento e o número do circuito (caso de transformador, disjuntor e carregamento).

É definida ainda a função JanelaMenu( ), que abre um retângulo de dimensões definidas pelos parâmetros de chamada, salvando a região da tela anterior em memória. Após a execução, uma nova chamada com parâmetro *abrefecha*=0, recupera a região salva em memória, através da função putimage( ) do Turbo C, restaurando a tela inicial, ou seja, fechando a janela.

2.3.10 - Funções do Módulo EXECUTAR.C

Este módulo possui apenas duas funções que são responsáveis pela execução do programa de fluxo de carga escrito na linguagem Fortran, e pela leitura do arquivo de saída dos resultados obtidos pelo mesmo.

A função Executar( ) cria inicialmente o arquivo temporário ENTRADA.DAD através de chamada à função GravarElet( ) do módulo GRAVAR.C, e utiliza a função spawnl( ) do Turbo C para proceder à execução externa do programa

LOADFLOW.EXE. Caso algum erro de sistema operacional ocorra, é enviada uma mensagem ao usuário, através da função strerror( ) do Turbo C.

Após a execução do programa de fluxo de carga, a função spawnl( ) retorna o controle de execução à interface gráfica, que redesenha o sistema, atualizando os valores de carregamento, tensão, tap etc, que são obtidos pela função LerSaidaFlow( ).

A função LerSaidaFlow( ) lê o arquivo temporário SAIDA.DAD criado pelo programa LOADFLOW.EXE, no formato ASCII, que contém apenas os valores resultantes de tensão, ângulo, geração (MW e MVAr) e potência reativa em capacitores e reatores, para cada barra, e os dados de carregamentos (MW e MVAr) e tap (no caso de transformador), para cada linha do sistema. Caso o programa tenha divergido, o programa de fluxo de carga coloca a palavra DIVE no início do arquivo SAIDA.DAD, informando que o caso divergiu. Isso força o retorno sem atualização dos dados elétricos, com a mensagem "SISTEMA DIVERGIU" ao usuário.

2.3.11 - Funções do Módulo TRANSFOR.C

As funções deste módulo são responsáveis pelo gerenciamento da transformação de coordenadas globais para coordenadas de dispositivo e vice-versa, bem como pelo efeito "zoom" no desenho do sistema elétrico.

As funções xdcwc( ), ydcwc( ), xwcdc( ), ywcdc( ), compxwcdc( ), compxdcwc(), compywcdc( ) e compydcwc( ) implementam as expressões:

$$xwc = \frac{xdc\ (xwmax - xwmin)}{viewport.right - viewport.left} + xwmin$$

$$ywc = \frac{ydc\ (ywmax - ywmin)}{viewport.bottom - viewport.top} + ywmin$$

$$xdc = (xwc - xwmin)\ \frac{xwc\ (viewport.right - viewport.left)}{xwmax - xwmin}$$

$$ydc = (ywc - ywmin)\ \frac{ywc\ (viewport.bottom - viewport.top)}{ywmax - ywmin}$$

$$compxdc = compxwdc\ \frac{viewport.right - viewport.left}{xwmax - xwmin}$$

$$compydc = compywdc\ \frac{viewport.bottom - viewport.top}{ywmax - ywmin}$$

$$compxwc = compxdc\ \frac{xwmax - xwmin}{viewport.right - viewport.left}$$

$$compywc = compydc\ \frac{ywmax - ywmin}{viewport.bottom - viewport.top}$$

respectivamente.

Estas expressões são obtidas através dos conceitos descritos no item 2.2.5, utilizando as coordenadas de janela do dispositivo, através de chamada à função getviewsettings( ) do Turbo C, e as coordenadas da janela no espaço global definidas na função Zoom( ).

A execução da função Zoom( ) é orientada pelo parâmetro (int) *zoom*, que informa o nível de aproximação desejado, através da atualização da variável global *inout*.

A atualização da janela global (variáveis *xwmin, ywmin, xwmax e ywmax*) e da variável *offset* é realizada através das expressões seguintes, para os níveis de apresentação 2%, 10%, 33% e 100%, respectivamente:

Escola Federal de Engenharia de Itajubá - EFEI
Biblioteca MAUÁ BIM

$xwmin = x - MaxXwc/100$
$xwmax = x + MaxXwc/100$
$ywmin = y - MaxYwc/100$
$ywmax = y + MaxYwc/100$
$offset = (viewport.right - viewport.left)/204$

$xwmin = x - MaxXwc/20$
$xwmax = x + MaxXwc/20$
$ywmin = y - MaxYwc/20$
$ywmax = y + MaxYwc/20$
$offset = (viewport.right - viewport.left)/1020$

$xwmin = x - MaxXwc/6$
$xwmax = x + MaxXwc/6$
$ywmin = y - MaxYwc/6$
$ywmax = y + MaxYwc/6$
$offset = (viewport.right - viewport.left)/3400$

$xwmin = 0$
$xwmax = MaxXwc$
$ywmin = 0$
$ywmax = MaxYwc$
$offset = (viewport.right - viewport.left)/20400$

O valor 204 no denominador do cálculo de *offset* na primeira expressão é utilizado para resultar no valor *offset* = 2.8, quando a janela no vídeo é cheia, e *offset* = 0.9, quando é utilizada a janela reduzida no canto superior direito. Estes valores 2.8 e 0.9 são as constantes que definem os segmentos de reta que formam cada equipamento, e foram escolhidos de forma empírica, de forma a resultar numa melhor apresentação visual do sistema elétrico. Os valores 1020, 3400 e 20400 nos denominadores das demais expressões são decorrência do valor 204 adotado, e são obtidos, respectivamente, a partir das operações: 100/20*204, 100/6*204 e 100/1*204.

A função Janela( ) apenas altera o valor das coordenadas de dispositivo para criar a janela reduzida de pesquisa no canto superior direito da tela. Após fixar os valores de coordenada é capturada a região da tela referente à janela reduzida, através da função getimage( ) do Turbo C. Uma nova chamada à função Janela( ) restaura a tela inicial, com utilização da função putimage( ), que apaga a janela reduzida.

A função scrollbar( ) é responsável pelo deslocamento do sistema na tela, quando é pressionado o botão 1 do "mouse" sobre a barra de rolagem horizontal ou vertical. Esta função apenas atualiza o valor da variável (static int) *eixo_x_novo_ws* e *eixo_y_novo_wc*, conforme o nível de "zoom" atual, para redesenho do sistema pela função Zoom( ).

O deslocamento é feito pela adição ou subtração, dependendo da direção desejada, de meia janela em coordenadas globais, em relação aos valores atuais.

Quando é pressionado um ponto qualquer no meio da barra de rolamento, é calculado o valor percentual deste ponto em relação ao comprimento total da barra e o desenho do sistema é realizado com base neste percentual, em relação à coordenada global, através das expressões:

$$eixo_x_novo_wc = ((m.xaxis - 0.22MaxX)/0.838MaxX)MaxXwc$$
$$eixo_y_novo_wc = ((m.yaxis - 0.05MaxY)/0.79MaxY)MaxYwc$$

onde $0.838MaxX$ e $0.79MaxY$ são os comprimentos da barra horizontal e vertical, respectivamente e $0.12MaxX$ e $0.05MaxY$ são os pontos de início das mesmas. As variáveis *m.xaxis* e *m.yaxis*, apresentadas no item 2.3.1, informam o ponto em que foi pressionado o "mouse" sobre a barra de rolamento.

2.3.12 - Funções do Módulo IMPRIME.C

Este módulo contém as funções gráficas responsáveis pela impressão dos resultados do processamento em impressora matricial.

A impressão é gerenciada pela função Imprimir( ), que através de chamada à função JanelaMenu( ), descrita no módulo MENUTECL.C, abre uma janela com as opções GRAFICO ou TEXTO. Através de condições "if" sobre o retorno da posição do "mouse", obtido pela função Mpos( ), é executada a função OpçãoGraf( ) ou OpçãoText( ).

A função OpçãoGraf( ) é responsável pelo gerenciamento da impressão em forma gráfica. É similar à função Imprimir( ), abrindo uma nova janela de menus com as opções NORMAL ou LANDSCAPE. Estas opções são passadas à função PrintGraf( ), que efetivamente irá processar a execução da impressão em impressora matricial.

A função PrintGraf( ) inicia com a verificação de estado da impressora pela chamada à função biosprint( ) do Turbo C. Caso esteja conectada, o processo prossegue com redesenho do sistema utilizando a tela toda, através de chamada à função Zoom( ).

Como descrito no item 2.2.6, esta função irá imprimir a tela através de "hardcopy", "píxel" a "píxel". Para a impressão no formato normal (portrait) são feitos 3 "loops" com o comando "for" da linguagem C:

- o primeiro subdivide a tela ao longo do eixo Y em linhas de 8 "píxels":

```
for ( j = 0; j < =  ((MaxY+1)/8-1);  j++);
```

- o segundo percorre cada linha de 8 ""píxels"" até o final da tela no eixo X:

```
for ( i = 0; i < = MaxX;  i++);
```

- o terceiro verifica o estado de cada "píxel" na coluna de 8 "píxels" de uma linha, obtido através de chamada à função getpixel( ) do Turbo C. É utilizada uma variável caracter *m*, inicializada em 0, para enviar à impressora a cadeia de "píxels" lida. Conforme o "loop" é executado, *m* é deslocada um "bit" para

a esquerda. Se o "píxel" corrente não for 0 (cor de fundo), *m* é incrementada (ligado o "bit" mais a direita), obtendo assim a reprodução da coluna de "píxels" na variável *m*:

```
m = 0;
for (k = 0; k < 8;  k++);
  {
    m << = 1;                          /* shift de 1 na variável m          */
   if (getpixel(i,j*8+k)) m++;   /* incrementa em 1 a variável m */
  }                                    /* se o "pixel"está ligado           */
```

Para impressão no formato "landscape" é seguido o mesmo procedimento, apenas trocando-se o sentido de varredura, ou seja, o primeiro "loop" é feito ao longo do eixo X e o segundo ao longo do eixo Y.

Para impressão dos dados em formato Texto é utilizada a função OpçãoText( ) que permite a opção de escolha entre impressora ou vídeo através de chamada às funções ImprimirText( ) ou MostrarVideo( ). A primeira apenas abre o arquivo SAIDA.PRN, que é criado pelo programa LOADFLOW.EXE, e o imprime linha por linha na impressora, através da função fprintf( ) do Turbo C. A segunda, ativa o programa auxiliar SAIDA.EXE, que lê o arquivo SAIDA.PRN e o exibe na tela. Neste caso é feito o chaveamento entre os modo gráficos e texto, através das funções restorecrtmode( ) e setgraphmode( ) do Turbo C.

CAPÍTULO 3

RESULTADOS E CONCLUSÕES

3.1 - RESULTADOS

3.1.1 - Testes Realizados

Durante o desenvolvimento do programa, foram realizados inúmeros testes, utilizando-se diferentes sistemas elétricos, obtidos a partir de equivalentes do sistema interligado brasileiro e também diversos sistemas teste encontrados nas publicações do IEEE.

Esses sistemas foram sendo escolhidos com o intuito de testar cada uma das novas funções que eram incorporadas ao programa, visando explorar as suas características, como por exemplo: diferentes níveis de tensão, ocorrência de transformador com comutação automática de tap, representação de bancos de capacitores e reatores, etc.

Uma observação importante feita durante os testes realizados com o programa foi a limitação dos compiladores utilizados, cujos programas gerados fazem uso apenas da memória real do microcomputador, que permite uma disponibilidade de, no máximo, 640 Kbytes de memória para o processamento. Essa restrição ocorre devido à forma utilizada pelo sistema operacional DOS para gerenciamento da memória. Como os microcomputadores atualmente têm disponíveis valores bem superiores a este limite, é necessário que o compilador crie o programa executável com capacidade de gerenciamento da memória estendida (nome dado à memória superior a 1Mbyte do microcomputador).

Como foram utilizados os compiladores Turbo C 2.0 da Borland e Fortran 5.0 da Microsoft para o desenvolvimento deste trabalho, e ambos não possuem

capacidade de gerenciamento da memória estendida, os limites máximos para representação do sistema elétrico foram de 100 barras e 200 linhas.

Devido a este fato, foi feito um teste utilizando o compilador Fortran produzido pela Lahey Computer System Inc, que utiliza a memória estendida dos microcomputadores padrão Intel 386. Com o programa de fluxo de carga compilado nessa ferramenta foi possível a simulação de um sistema elétrico com 1000 barras e 2000 linhas, sendo que pode-se chegar a valores ainda superiores a esses, dependendo da quantidade de memória estendida presente no equipamento.

Ressalta-se que a representação gráfica do sistema elétrico permaneceu limitada a 100 barras e 200 linhas, devido ao programa escrito na linguagem C, responsável pela exteriorização gráfica, manter seu processamento na memória real. Esse fato, porém, não limita a representação gráfica do sistema, pois pode ser feito o particionamento dos dados gráficos em vários arquivos (limitados a 100 barras e 200 linhas), permanecendo os dados elétricos representados de forma completa. Assim, durante uma simulação, para se ter a visualização dos resultados de forma gráfica, bastaria a utilização do arquivo de dados gráficos contendo a área de interesse, sendo o processamento do fluxo de carga realizado sobre o sistema completo.

Com o sucesso do teste realizado, conclui-se que a utilização de compiladores que explorem a capacidade de gerenciamento da memória estendida permite a representação de sistemas elétricos de grande porte.

Outro teste realizado, para verificação da portabilidade do "software" desenvolvido, foi o processamento sobre o sistema operacional UNIX, utilizando emulação do sistema operacional DOS. Foi utilizado o sistema operacional UnixWare, produzido pela Novell, instalado em um microcomputador 486 DX 33, com 16 Mbytes de memória, disco rígido particionado apenas no formato Unix e utilizando monitor controlado pelo adaptador gráfico padrão VGA.

O programa foi executado a partir do comando de emulação DOS, na janela de controle de execução do Unix, passando a processar sem apresentar erros de execução. O inconveniente observado foi que o programa passou a controlar

totalmente os dispositivos de vídeo e "mouse" do microcomputador, perdendo-se com isso a vantagem multitarefa do ambiente Unix.

3.1.2 - Apresentação dos Resultados Obtidos

Em sua versão final, os seguintes arquivos (programas executáveis e dados) compõem a ferramenta desenvolvida:

ARQUIVO	**TAMANHO** (bytes)	
BOTAO.EXE	126.590	
FLOWGRAF.EXE	356.904	
LOADFLOW.EXE	193.522	
SAIDA.EXE	4.217	
CAMPINAS.ELE	17.367	*
CAMPINAS.GRA	14.246	*
EGAVGA.BGI	5.363	
GOTH.CHR	8.560	
LITT.CHR	2.138	
ENTRADA.DAD	17.367	*
CURCROSS.IMG	150	
CURSETA.IMG	142	
MENUCOR.IMG	2.606	
MENUDES.IMG	9.210	
MENUEDIT.IMG	12.030	
MENUPRIN.IMG	13.382	
POSBAR.IMG	2.970	
POSCAP.IMG	2.970	
POSCARG.IMG	2.894	
POSCARRE.IMG	2.970	
POSDISJ.IMG	3.046	
POSGER.IMG	2.970	
POSLIN.IMG	1.606	
POSLIN1.IMG	2.406	

POSREAT.IMG	2.970
POSSEC.IMG	3.046
POSTRAF.IMG	1.606
SCBAHRZG.IMG	3.366
SCBAVERG.IMG	3.270

* Estes arquivos referem-se ao caso apresentado como exemplo.

Para a apresentação neste trabalho dos resultados obtidos com o produto desenvolvido, foi escolhido um caso utilizando o sistema elétrico da Companhia Paulista de Força e Luz - CPFL, da região de Campinas, com um total de 95 barras e 145 linhas, das quais 6 barras representam o equivalente do sistema externo, obtido através do método Ward-Estendido, sobre o caso base do Sistema Interligado Brasileiro.

Esta escolha foi feita para facilitar uma "representação gráfica de forma geográfica" do sistema elétrico e também porque os números de barras e de linhas desse sistema são próximos dos limites máximos de representação gráfica, o que permite a visualização da distribuição de um sistema deste porte na tela do computador.

O equivalente foi realizado nas barras de interligação em 138kV, o que resultou em um sistema com os níveis de tensão de 138 e 69 kV, utilizando 4 autotransformadores entre estes subsistemas. Está representado, ainda, o banco de capacitores de 32 MVAr existente na barra de 138 kV da SE Piracicaba.

A figura 3.1, a seguir apresenta a área de concessão da CPFL, destacando seus três centros de operação de área. Pode-se identificar especificamente o Centro de Operação de Área Sul, composto básicamente da região de Campinas. A figura 3.2 apresenta o detalhamento do sistema elétrico supervisionado por esse Centro.

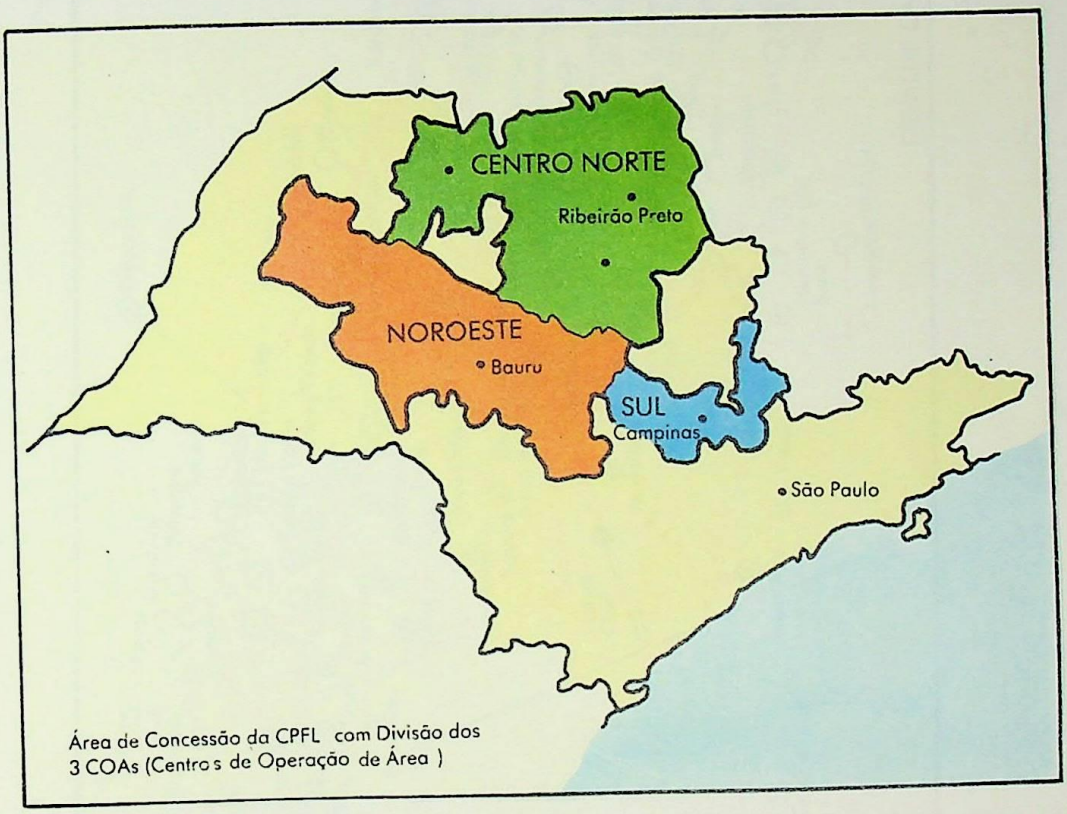

CENTRO NORTE
Ribeirão Preto
NOROESTE
Bauru
SUL
Campinas
São Paulo
Área de Concessão da CPFL com Divisão dos
3 COAs (Centros de Operação de Área)

FIGURA 3.1

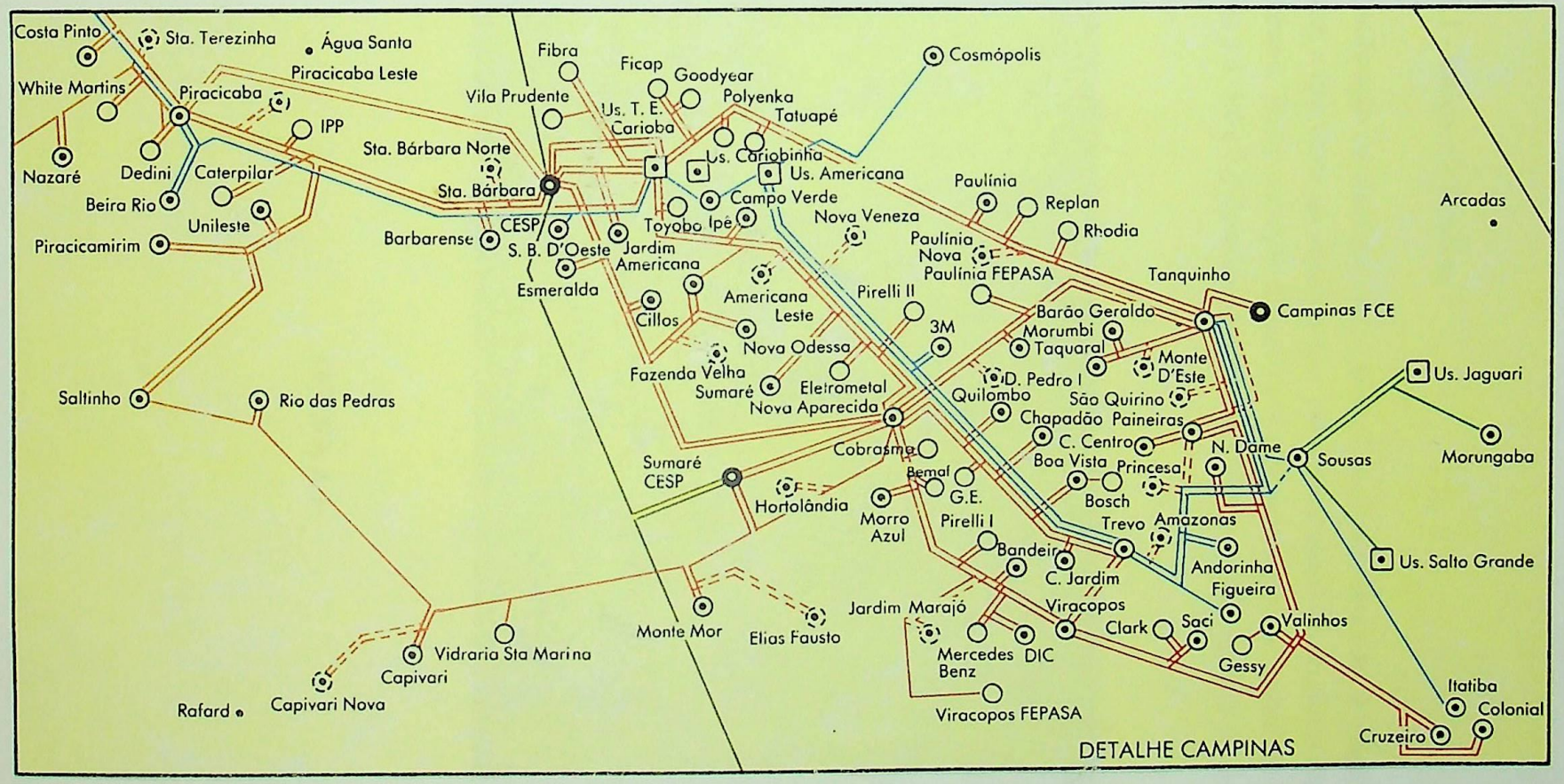


FIGURA 3.2

Escola Federal de Engenharia de Itajubá - EFEI
Biblioteca MAUÁ BIM

3.1.2.1 - Edição do Sistema

Para utilização da interface, inicialmente é necessário proceder à edição gráfica do sistema, o que demanda algum tempo e o conhecimento do diagrama representativo do mesmo. Neste exemplo foi utilizada a representação de forma geográfica. As figuras 3.3, 3.4 e 3.5 ilustram algumas etapas na construção gráfica do sistema:

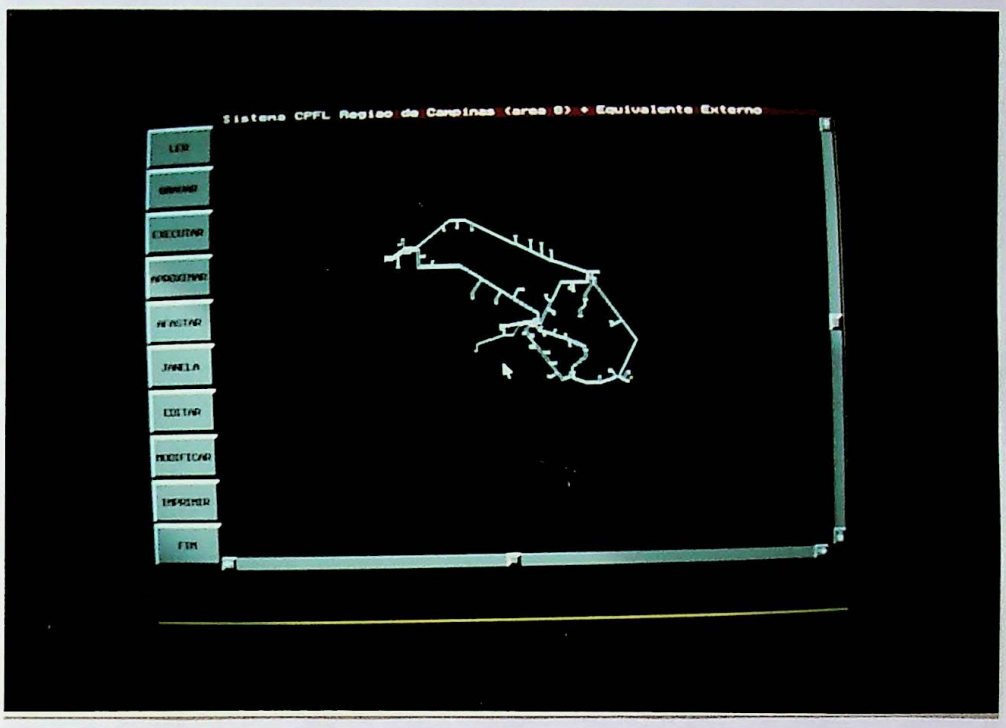


FIGURA 3.3

FIGURA 3.4

FIGURA 3.5

Quando o sistema elétrico já está representado na forma gráfica e armazenado em arquivo, pode ser recuperado através da opção LER, que exibe o menu apresentado na figura 3.6, com as opções NOVO ou ACRESCENTAR:

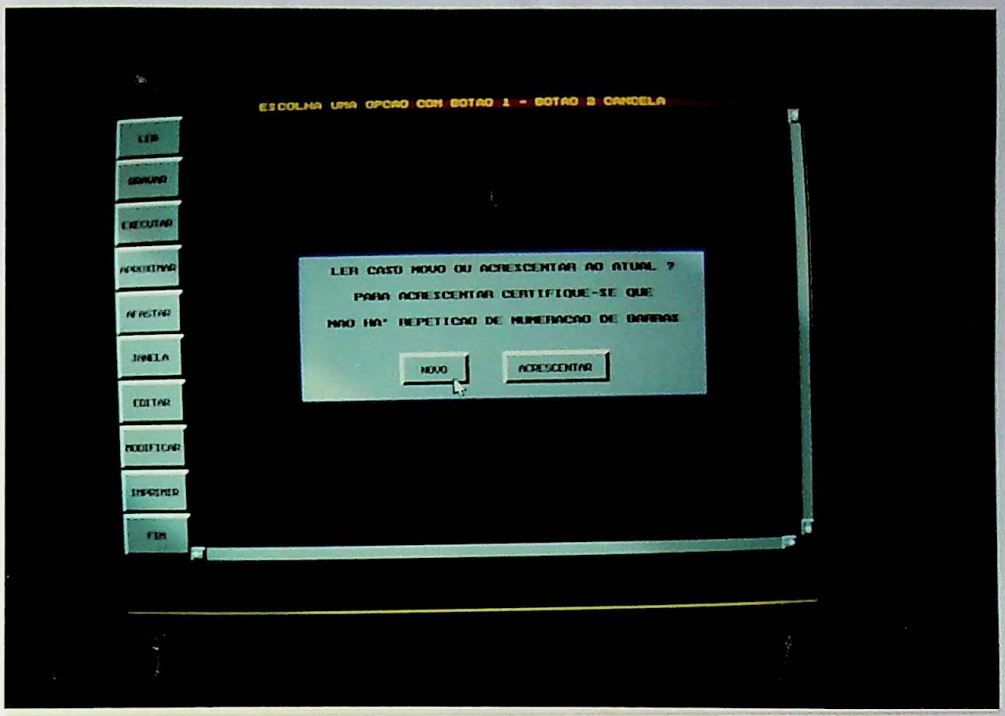


FIGURA 3.6

Para a inserção de um equipamento o usuário deve entrar no modo EDIÇÃO e escolher a opção DESENHO, quando aparecerá o menu inferior de objetos.

A figura 3.7 ilustra a seleção de um disjuntor, quando então aparece o menu auxiliar de escolha de sua orientação:

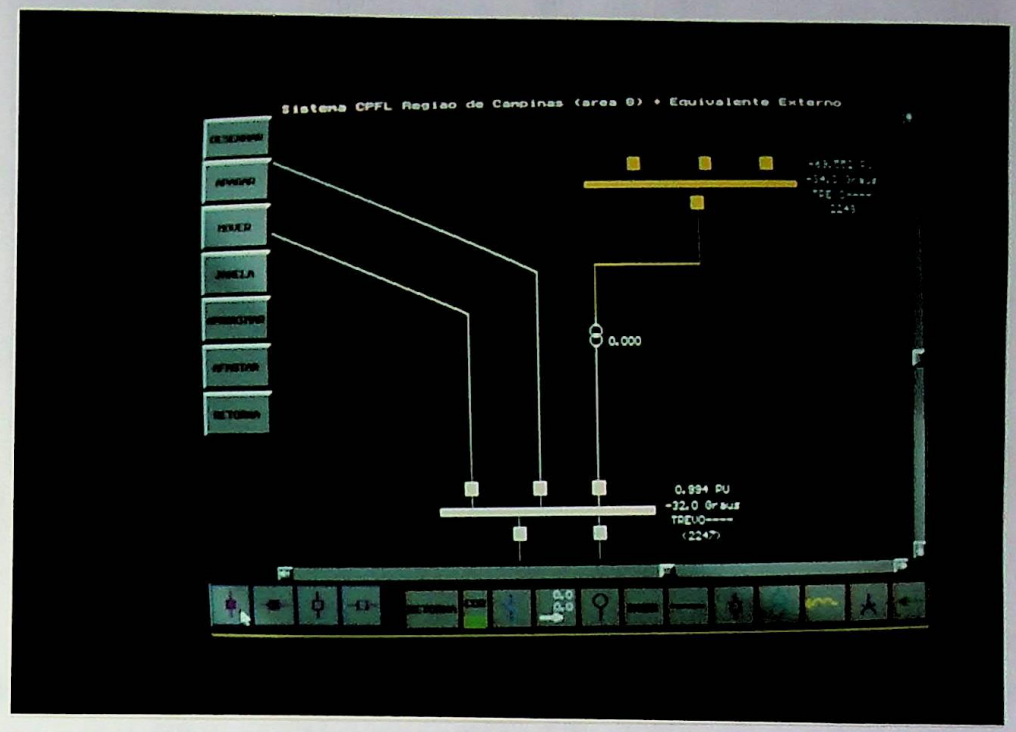


FIGURA 3.7

Após a escolha, com o botão 1 do "mouse" é feito o arraste do equipamento até a posição desejada, quando deve ser pressionado o botão 2 para sua fixação.

É então apresentada uma janela para entrada dos dados que fazem a amarração do equipamento gráfico com a linha à qual está conectado, como ilustrado na figura 3.8:

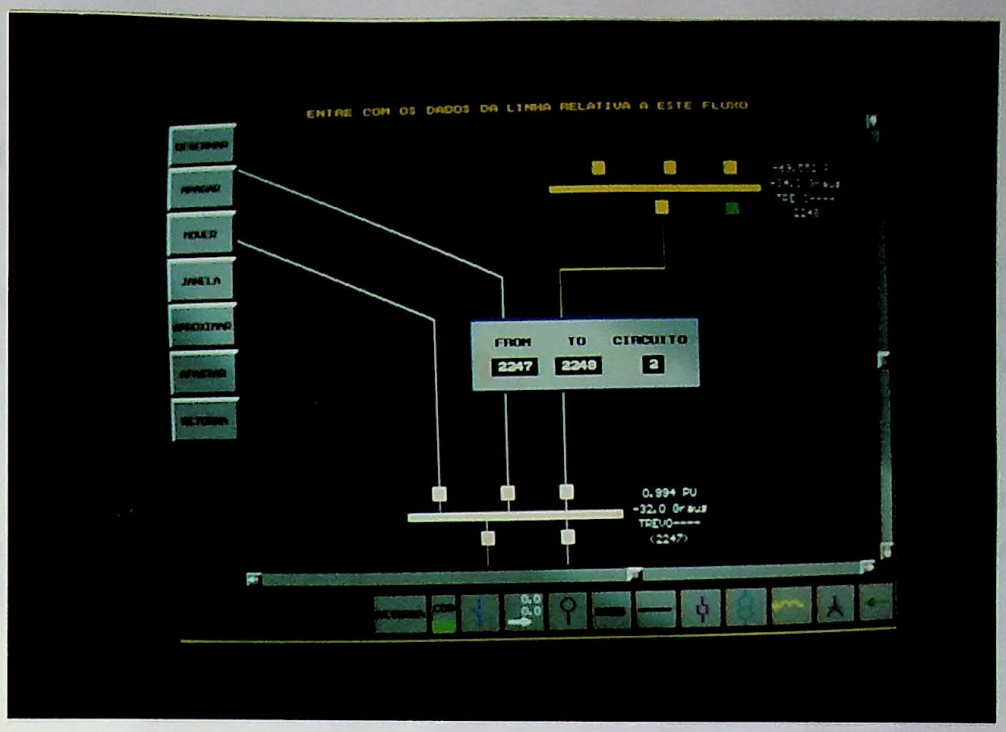


FIGURA 3.8

Como pode ser observado, a cor de desenho do equipamento foi alterada inicialmente para verde, utilizando a opção COR do menu anterior, que passa a ser a nova cor corrente, como ilustrado na figura 3.9:

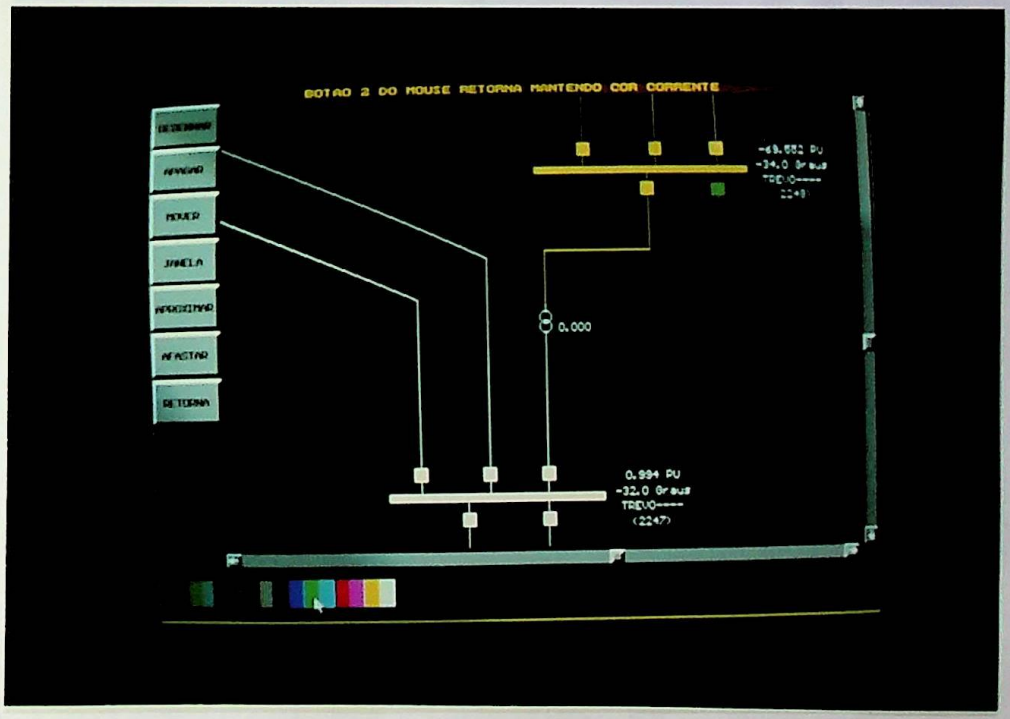


FIGURA 3.9

3.1.2.2 - Entrada de Dados

Através da opção MODIFICAR, é possível a entrada/modificação de dados elétricos, que serão alterados na base de dados no momento em que for salvo o caso estudado. Se o caso não é gravado pelo usuário ao final da seção, as alterações serão perdidas.

Estas telas são úteis também para a simulação de variação de geração, carga, reatâncias, taps e demais grandezas elétricas, sem manipulação do arquivo de dados elétricos.

As figuras 3.10 e 3.11 ilustram essas janelas:

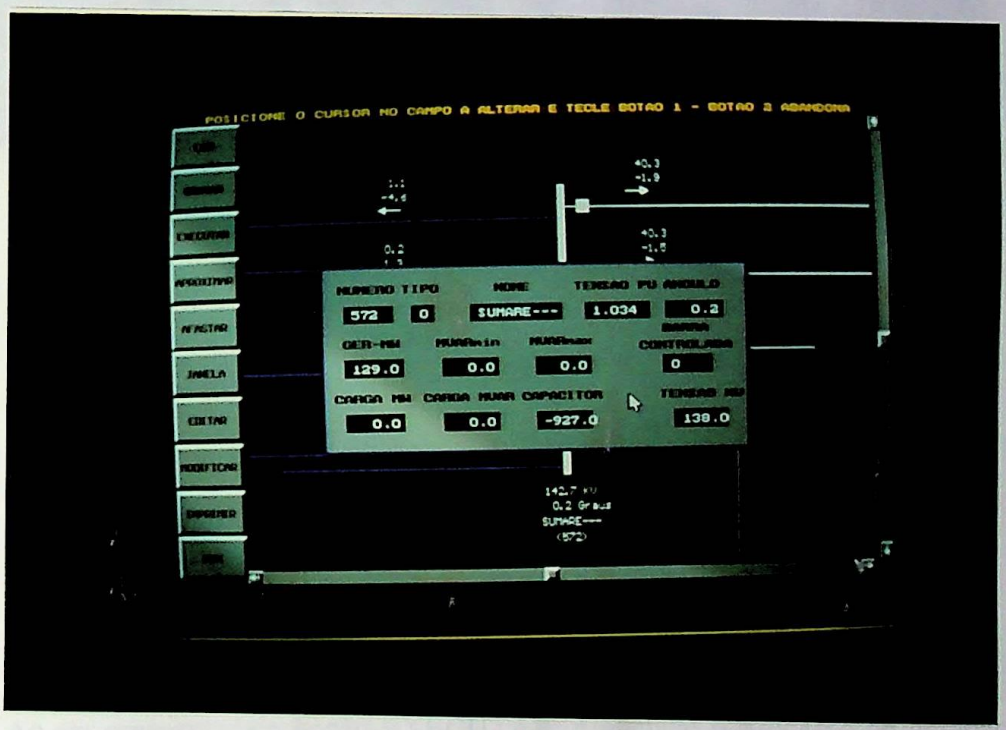


FIGURA 3.10

Escola Federal de Engenharia de Itajubá - EFEI
Biblioteca MAUÁ BIM

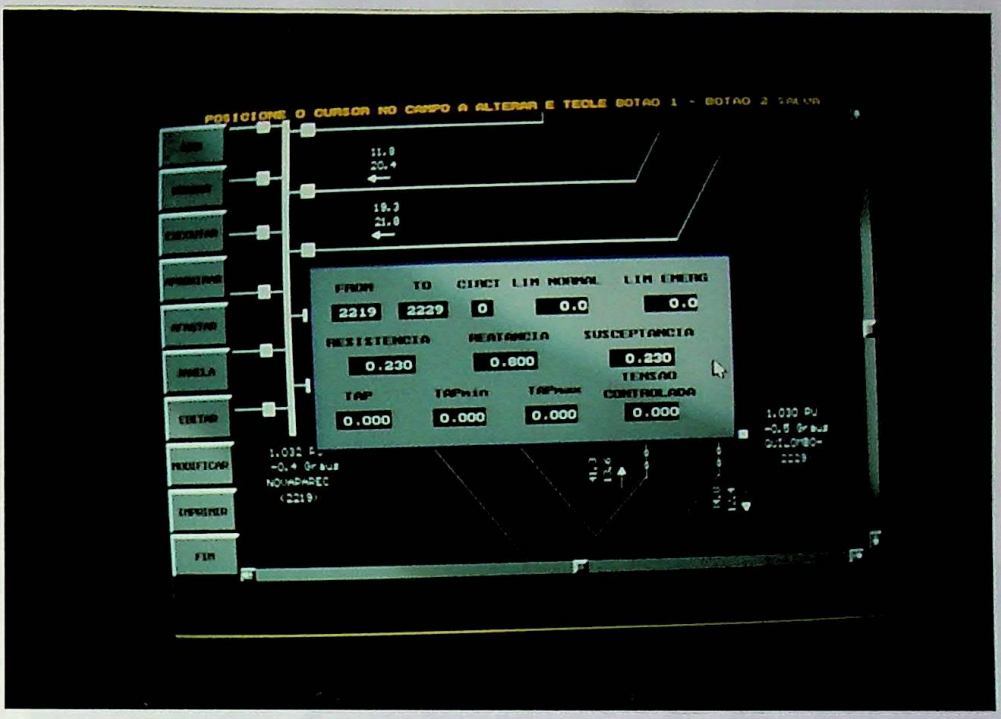


FIGURA 3.11

3.1.2.3 - Execução

Uma vez representado o sistema elétrico, passamos à execução da simulação.

Ao ser pressionada a opção EXECUTAR, é iniciado o processo de geração do arquivo de dados para o programa de fluxo de carga, que é executado imediatamente.

Os seguintes tempos de processamento foram verificados, a partir da pressão sobre o botão EXECUTAR, até o redesenho da tela gráfica, com os resultados, considerando o sistema em análise (95 barras, 145 linhas):

Micro 386SX - 40 MHZ - s/ coprocessador aritmético → 50 segundos
Micro 486DX - 33 MHZ - c/ coprocessador aritmético → 10 segundos

As figuras 3.12 e 3.13 apresentam os fluxos em linhas, antes e após a simulação de abertura de um disjuntor, onde é destacada a ultrapassagem do limite de carregamento em emergência de uma LT, utilizando a cor vermelha:

.

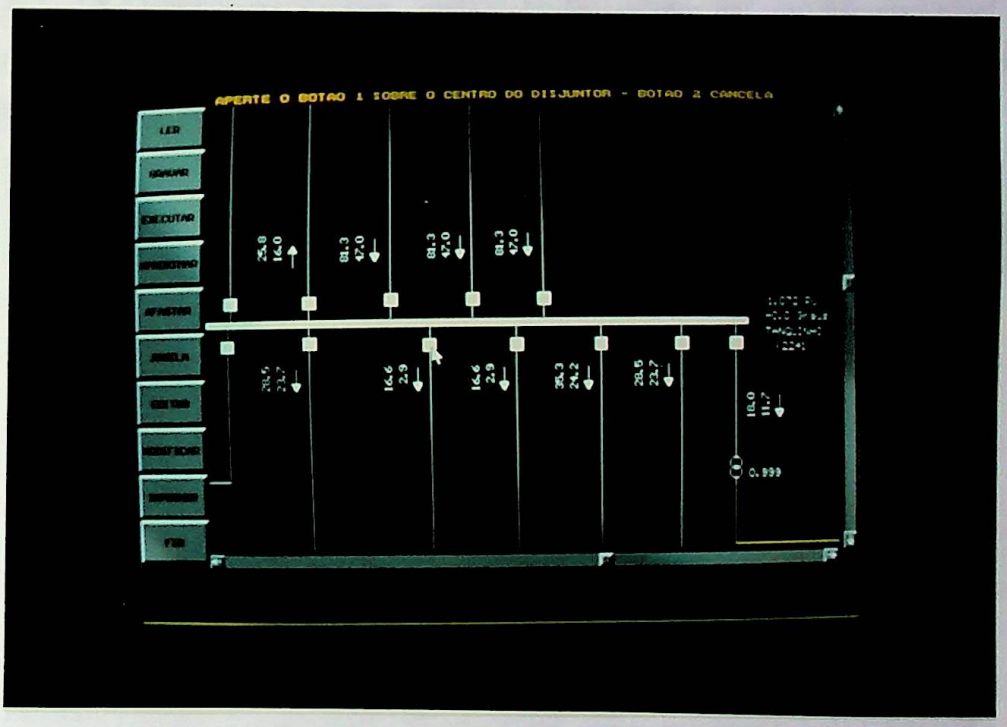


FIGURA 3.12

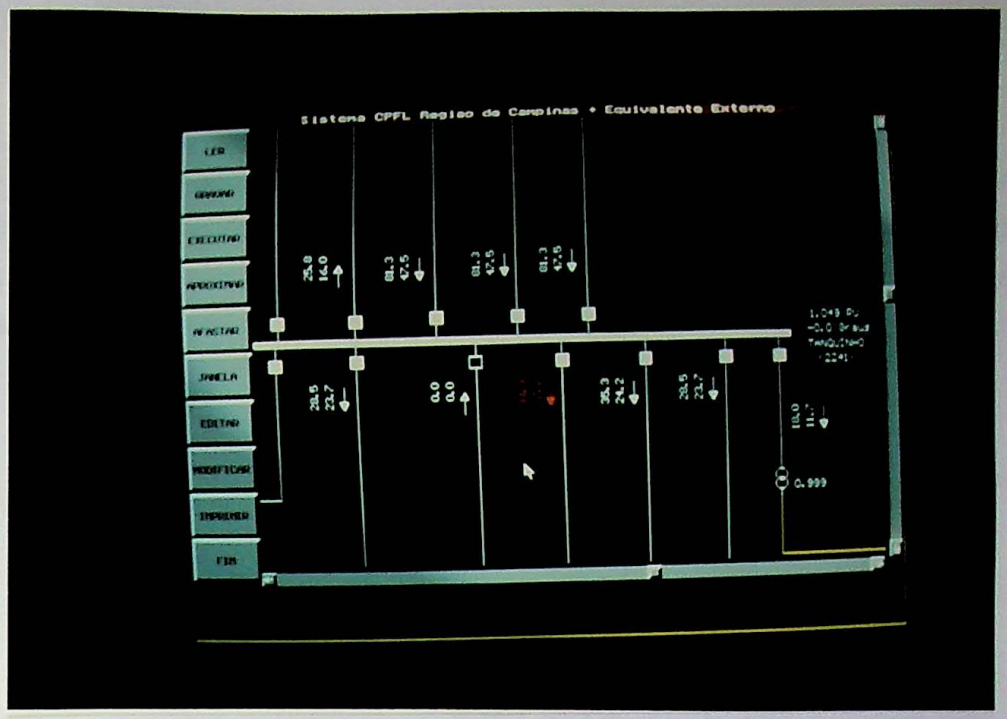


FIGURA 3.13

A figura 3.14 apresenta a mesma região elétrica, porém em um primeiro nível de afastamento ("zoom out"), podendo ser visualizada a linha onde foi ultrapassado o limite de carregamento:

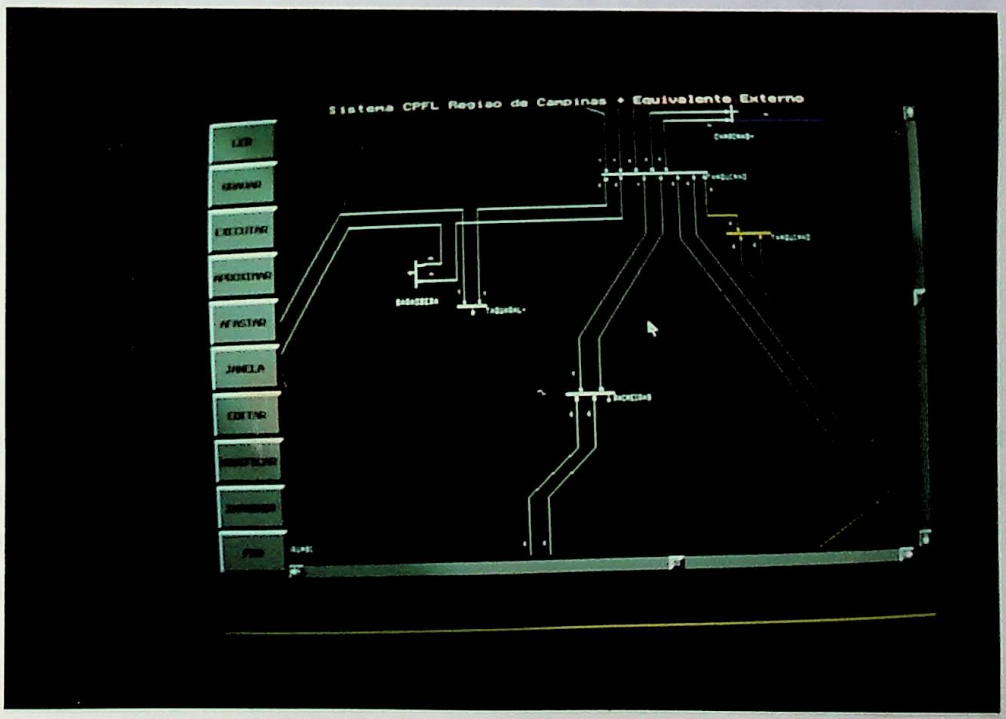


FIGURA 3.14

A figura 3.15 apresenta um caso onde os limites de carregamento em regime normal e em emergência na LT N.Aparecida-Viracopos foram reduzidos propositadamente, com o objetivo de ilustrar a representação da ultrapassagem desses limites:

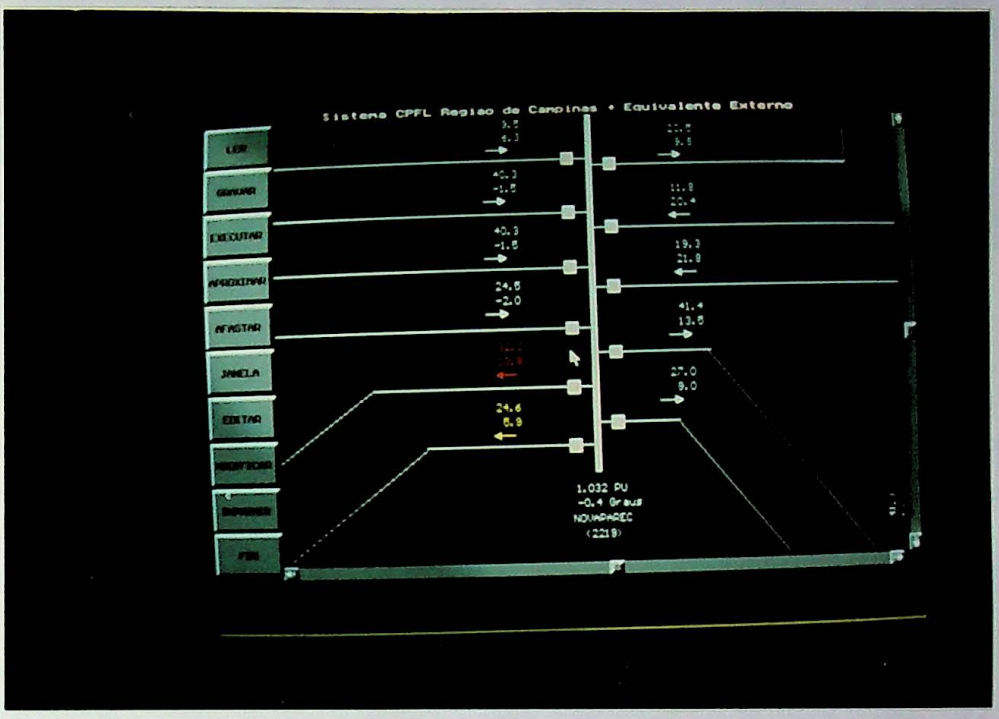


FIGURA 3.15

Para visualização dos valores verificados em outros pontos do sistema, pode-se modificar a área apresentada com auxílio da janela de visualização reduzida, como mostrado na figura 3.16. O sistema será desenhado novamente em torno do ponto indicado pela seta na janela, ou seja, sobre o ponto onde o botão do "mouse" for pressionado:

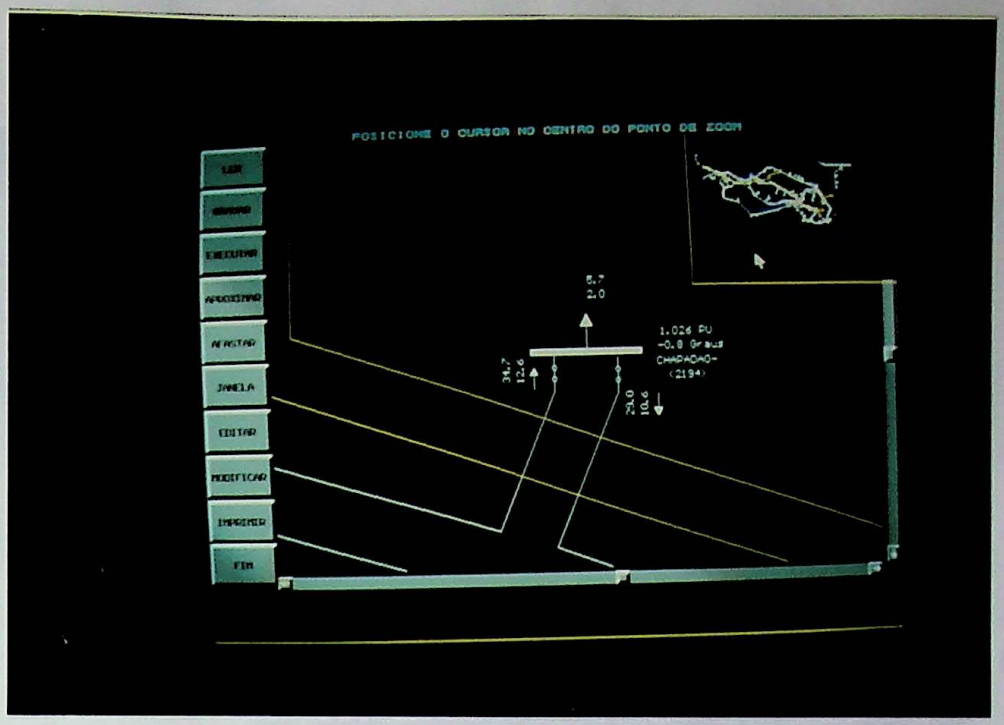


FIGURA 3.16

Para a visualização de outras áreas do sistema elétrico, podem também ser utilizados os recursos "scroll-bars", ou ainda através de afastamentos/aproximações sucessivas, como representado nas figuras 3.17, 3.18 e 3.19, onde são feitas 3 aproximações sobre a SE Viracopos:

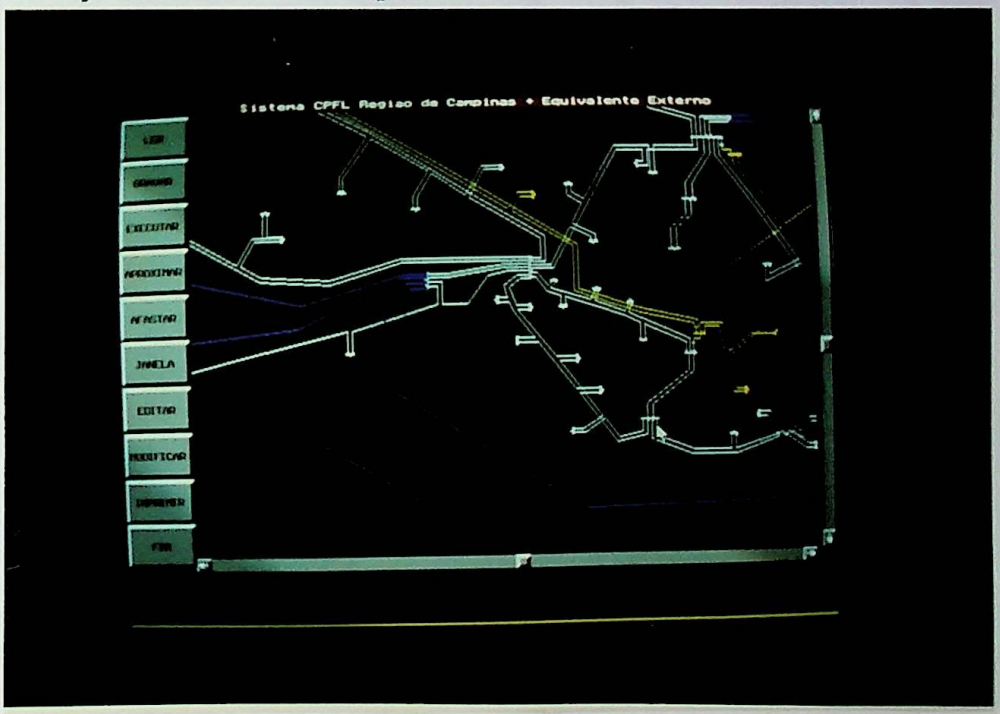


FIGURA 3.17

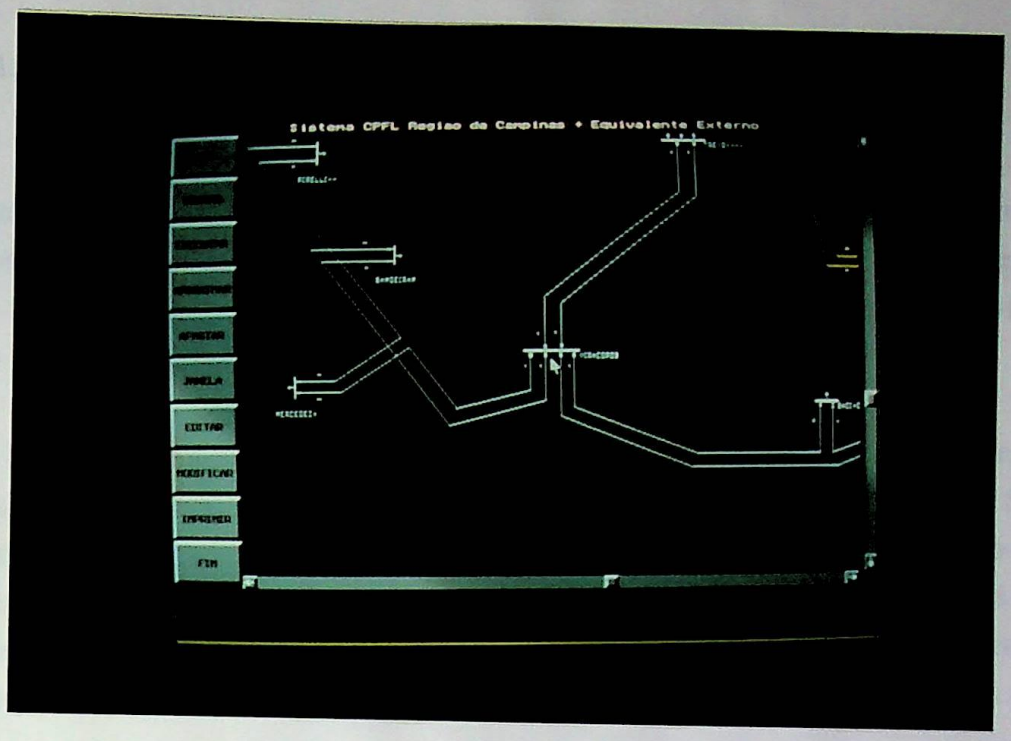


FIGURA 3.18

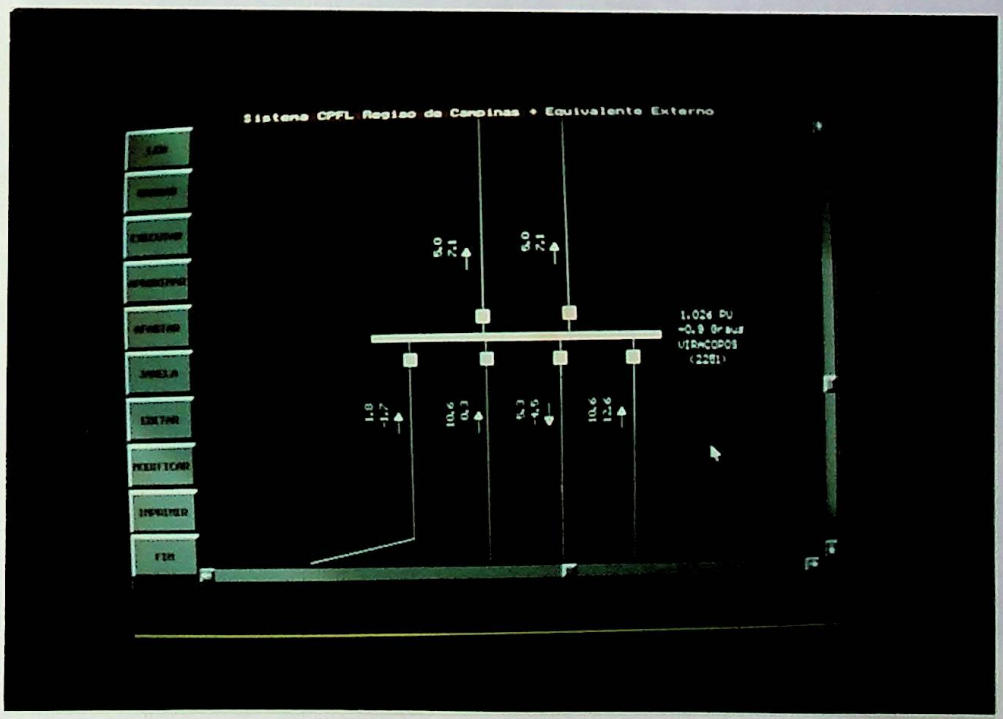


FIGURA 3.19

3.1.2.4 - Impressão dos Resultados

A impressão dos resultados pode ser feita de forma completa, em vídeo, no formato texto, como apresentado na figura 3.20 (no formato "landscape"), ou de forma gráfica em impressora matricial, conforme apresentado em alguns exemplos no Anexo I. No Anexo II são apresentados alguns exemplos de relatório completo do processamento de fluxo de carga, em impressora matricial.

Cabe destacar que foi utilizado o mesmo formato de relatório impresso elaborado em Fortran pelo programa de fluxo de carga, que é bastante completo. São apresentados os dados lidos, seguidos de um sumário totalizador, número de iterações realizadas, resultados de barras, resultados de linhas (incluindo perdas em cada linha e percentual de sobrecarga, se houver) e, ao final, o total de perdas no sistema.

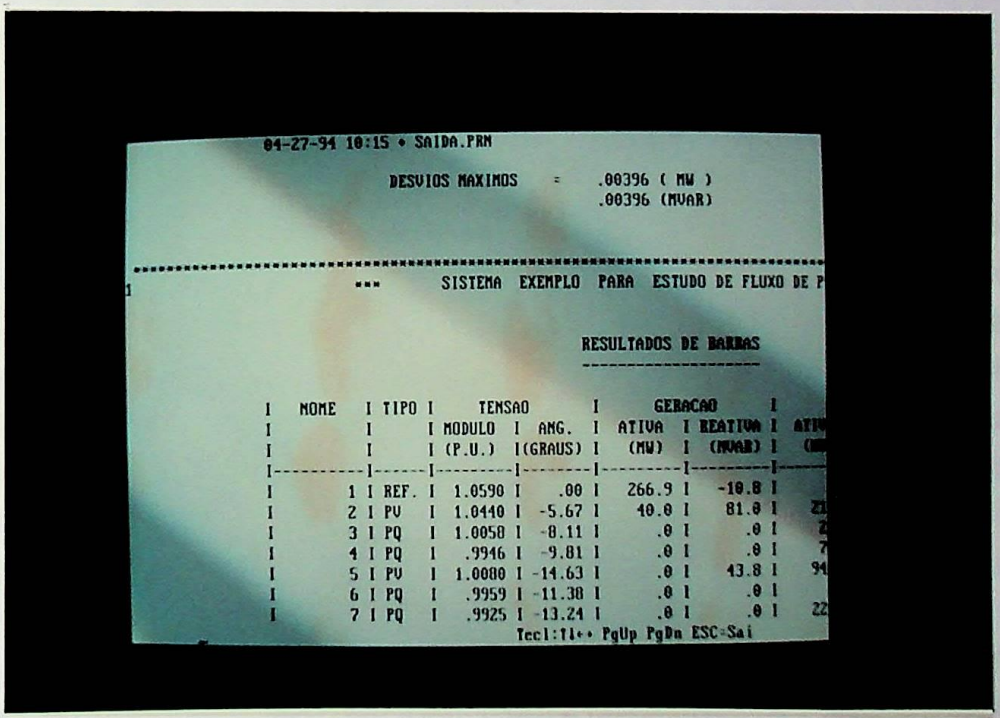


FIGURA 3.20

3.2 - CONCLUSÕES

No desenvolvimento deste trabalho, procurou-se, desde o início, obedecer ao princípio de criar um programa que fornecesse toda a potencialidade dos recursos gráficos ao processamento para análise de sistemas de potência, mas que fosse ao mesmo tempo, simples de ser utilizado.

Trabalhos similares foram utilizados como ponto de partida, contudo o produto final apresenta uma personalidade própria, que pode ser observada na apresentação visual, na sequência de comandos, na disposição dos resultados do processamento e na capacidade de alteração dos dados. A idéia básica é a propiciar o máximo de facilidade de manuseio ao usuário.

Com isso, o programa passa a ter uma grande abrangência de aplicação, tornando-se uma ferramenta que pode ser utilizada tanto pelo técnico não habituado ao uso de programas de simulação, como pelo especialista em análise de sistemas elétricos.

O programa torna-se também poderoso para utilização pelas áreas de análise das empresas do setor elétrico, pela facilidade na manutenção de uma base de dados gráfica, ou seja, os dados para representação gráfica do sistema elétrico. Isto deve-se ao fato do único vínculo entre dados gráficos e elétricos ser o número da barra. Normalmente a numeração das barras representativas do sistema não se alteram, pois as atualizações nos casos base (modelo do sistema elétrico consensado a nível de sistema interligado brasileiro) consistem nas alterações dos dados de carga e geração e em pequenas alterações na configuração do sistema elétrico. Assim, após a atualização do caso base, que normalmente é realizada com a participação de todas as empresas do setor elétrico e com a utilização de programas de simulação fornecidos pela Eletrobrás, ao retornar à sua Empresa o especialista pode utilizar-se do programa ora descrito para suas simulações, uma vez que a base de dados gráfica será compatível com o novo caso base, o que é fundamental para áreas de estudos, onde as atualizações são freqüentes.

Uma aplicação importante do progama, é sua utilização como ferramenta de treinamento. Os atuais programas de simulação, cuja interação com o usuário acontece através de relatórios, ficam limitados ao uso pelo analista especializado. Quando a interação é feita de forma gráfica, a simulação passa a ser mais intuitiva, exigindo menor qualificação do usuário, e assim viabilizando sua utilização em treinamento, principalmente de operadores de Centros de Operação.

Uma grande preocupação durante o desenvolvimento deste trabalho foi quanto à portabilidade do "software" final. Esse objetivo foi atingido, dentro do limite do sistema operacional DOS. Ressalta-se , entretanto, que foi conseguido o processamento do mesmo em ambiente Unix (UnixWare - Novell), com a utilização de emulação DOS, conforme os teste descritos no Capítulo 3.

Embora o programa fonte faça uso de funções exclusivas do compilador Turbo C da Borland, limitando a sua utilização em compiladores de outros fabricantes (devido às funções gráficas não serem padronizadas), o desenvolvimento a partir de dados estruturados e mapeamento de pontos gráficos em coordenadas globais, torna a base estrutural do programa portátil, viabilizando sua implantação em outros ambientes (como "Windows - DOS" ou "XWindows - Unix").

O desempenho dos equipamentos disponíveis atualmente no mercado (padrão Intel 386 ou superior), pode ser considerado totalmente adequado ao processamento exigido. Nos testes realizados, a velocidade de processamento e exibição em tela foi inteiramente satisfatória e até além das expectativas iniciais, quando os equipamentos disponíveis limitavam-se aos microcomputadores padrão Intel 8086 (XT). Foi encontrada apenas limitação quanto à memória requerida, porém devido aos compiladores utilizados, que não fazem uso da memória estendida dos microcomputadores 386, executando o processamento apenas na memória real (limitada a 640 Kbytes), gerenciada pelo sistema operacional DOS.

Esta limitação pode ser evitada com a utilização de compiladores que façam uso da memória estendida (superior a 1 Mbytes), o que permitiria elevar o número de barras representadas à ordem de 2000, suficiente para representação do Sistema

Interligado Sul/Sudeste Brasileiro, de forma completa, conforme os testes detalhados no item 3.1 do Capítulo 3..

Foi constatada durante a elaboração do trabalho uma evolução extremamente rápida na utilização de interfaces gráficas em microcomputadores, bem como de ferramentas para seu desenvolvimento.

No início efetivo dos trabalhos, em meados de 1991, o ambiente gráfico "Windows" era pouco utilizado, principalmente pelo custo dos equipamentos necessários ao seu processamento. Ao final deste trabalho já é o sistema mais utilizado na micro informática.

Assim, como uma evolução futura da interface desenvolvida, a sua integração a esse ambiente deve ser considerada. Ressalta-se, entretanto, que na forma apresentada, é possível a execução da interface sobre o ambiente Windows, porém não utilizando toda sua potencialidade, principalmente a nível de gerenciamento de memória.

A evolução da interface também deve considerar a integração de outros "softwares" para análise de sistemas elétricos, como processamento de curto-circuito, análise de contingências, confiabilidade, estabilidade, entre outros; a exemplo dos trabalhos similares pesquisados. Desta forma, toda a necessidade dos técnicos das empresas do setor elétrico, bem como dos profissionais do meio acadêmico na área de análise de sistemas elétricos de potência, poderá ser satisfeita através da interface gráfica.

REFERÊNCIAS BIBLIOGRÁFICAS:

[1] ENDERLE, G. Computer Graphics Programming. Berlin: Springer-Verlag, 1984. 542 p.

[2] IBM Graphical Kernel System - Users Guide- Programer's Guide. Boca Raton, Florida: IBM, 1984. 520 p.

[3] FOLEY, James e VAN DAN, Andrew. Fundamentals of Interactive Computer Graphics. New York: Addison-Wesley, 1982. 664 p.

[4] EZZELL, Ben. Graphics Programming in Turbo C ++. New York: Addison-Wesley, 1990. 603 p.

[5] PERSIANO, Ronaldo C. M. e OLIVEIRA, Antonio A. F. Introdução à Computação Gráfica. Rio de Janeiro: LTC, 1988. 225 p.

[6] RANKIN, John R. Computer Graphics Software Construction. New York: Prentice Hall, 1989. 543 p.

[7] NEWMAN, Willian M. e SPROLL, Robert F. Principles of Interactive Computer Graphics. London: McGraw-Hill, 1981. 2 ed. 541 p.

[8] SHILDT, Herbert . Turbo C ++ - Guia do Usuário. São Paulo: McGraw-Hill, 1990. 552 p.

[9] CUNHA, Gilberto J. et al. Computação Gráfica e suas Aplicações em CAD. São Paulo: Atlas, 1987. 199 p.

[10] FERRARO, Richard F. Programmer's Guide to the EGA and VGA Cards. New York : Addison-Wesley , 1989. 696 p.

[11] CALE, Kevin R. Dissertação de Mestrado: Interactive Power Systems Analysis. Manchester: The Victoria University of Manchester, 1976. 113 p.

[12] PAHALAWATHTHA, N. et al. A Power System CAD Package for the Workstation and Personal Computer Environment. New York: IEEE Power Engineering Review, Feb. 1991. p 77.

[13] BACK, H. e MESLIER, F. - PLATINE - A New Computerized System to Help Planning the Power Transmission Networks. In: International Conference on Large High Voltage Eletric Systems - CIGRÉ, Florence: 17 a 20 Oct 1989. CIGRÉ SC 37. 14 p.

[14] LI, S. e SHAHIDEHPOUR, S. M. An Object Oriented Power System Graphics Package for Personal Computer Environment. New York: IEEE Transactions on Power System, v. 8, n. 3, Aug. 1993. p 1054-1060.

[15] FOLEY, M. et al. An Object Based Graphical User Interface for Power Systems. New York: IEEE Transactions on Power System, v. 8, n. 1, Feb. 1993. p 97-104.

[16] OLIVEIRA, Adel M. S. e GOMES, José R. Sistema de Análise de Fluxo de Potência para Operação em Tempo Real com a Apresentação dos Resultados Plotados em Diagramas Unifilares. In: IV Encontro para Debates de Assuntos da Operação, 1993, São Paulo. Anais ... São Paulo: Item 4, 23 Jul. 1993. 11 p.

[17] STOTT, B. e ALSAC, O. Fast Decoupled Load Flow. In: IEEE PES Summer Meeting, Vancouver, Jul. 1973. Paper T43 463-7.

[18] BRESENHAM, J. A Linear Algorithm for Incremental Digital Display of Circular Arcs. New York: Communications of the ACM - Association for Computing Machinery, v 20, n2, Feb. 1977. p 100 - 106

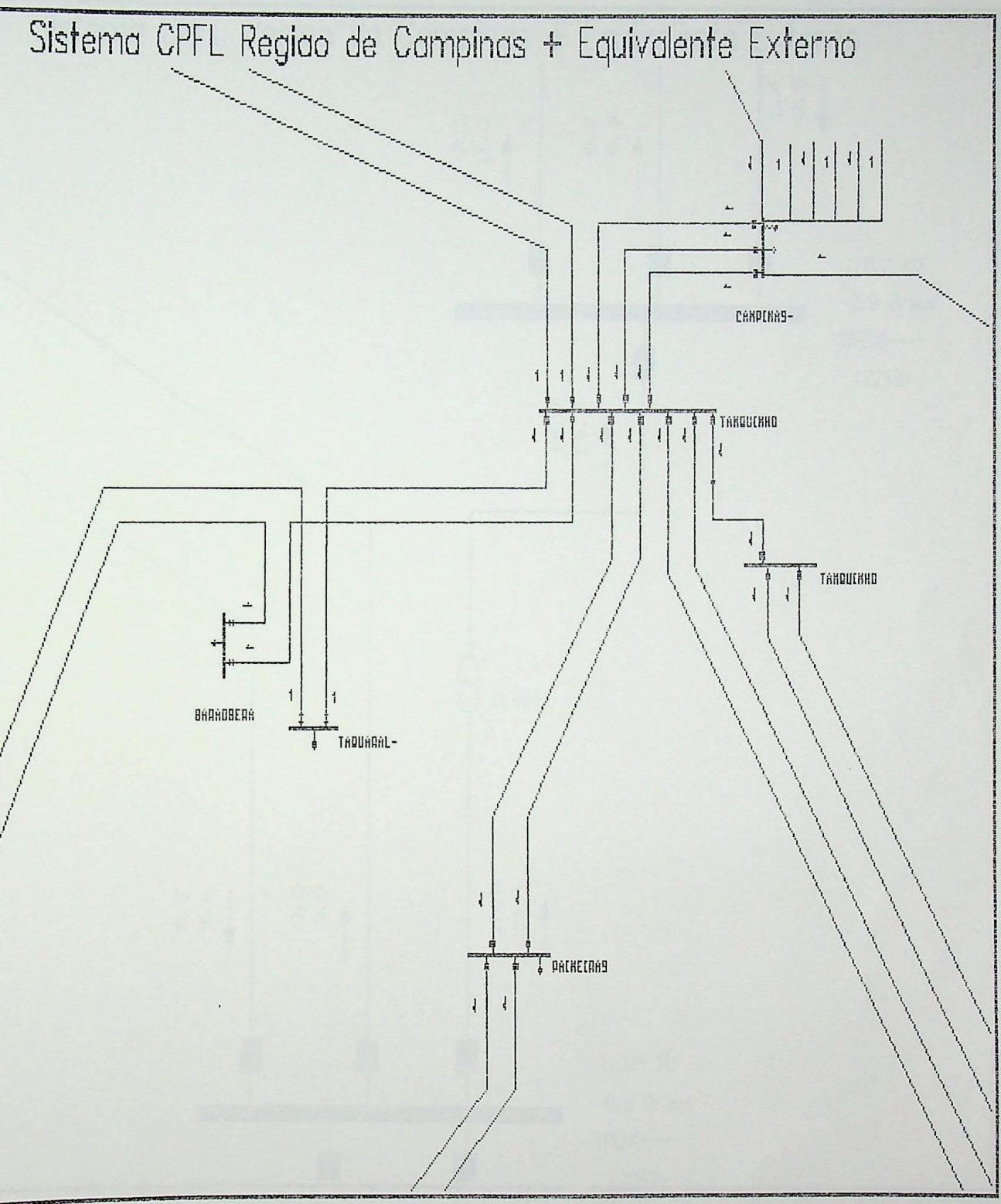
[19] TORI, R. et. al. Fundamentos de Computação Gráfica. Rio de Janeiro: LTC, 1987. 356 p.

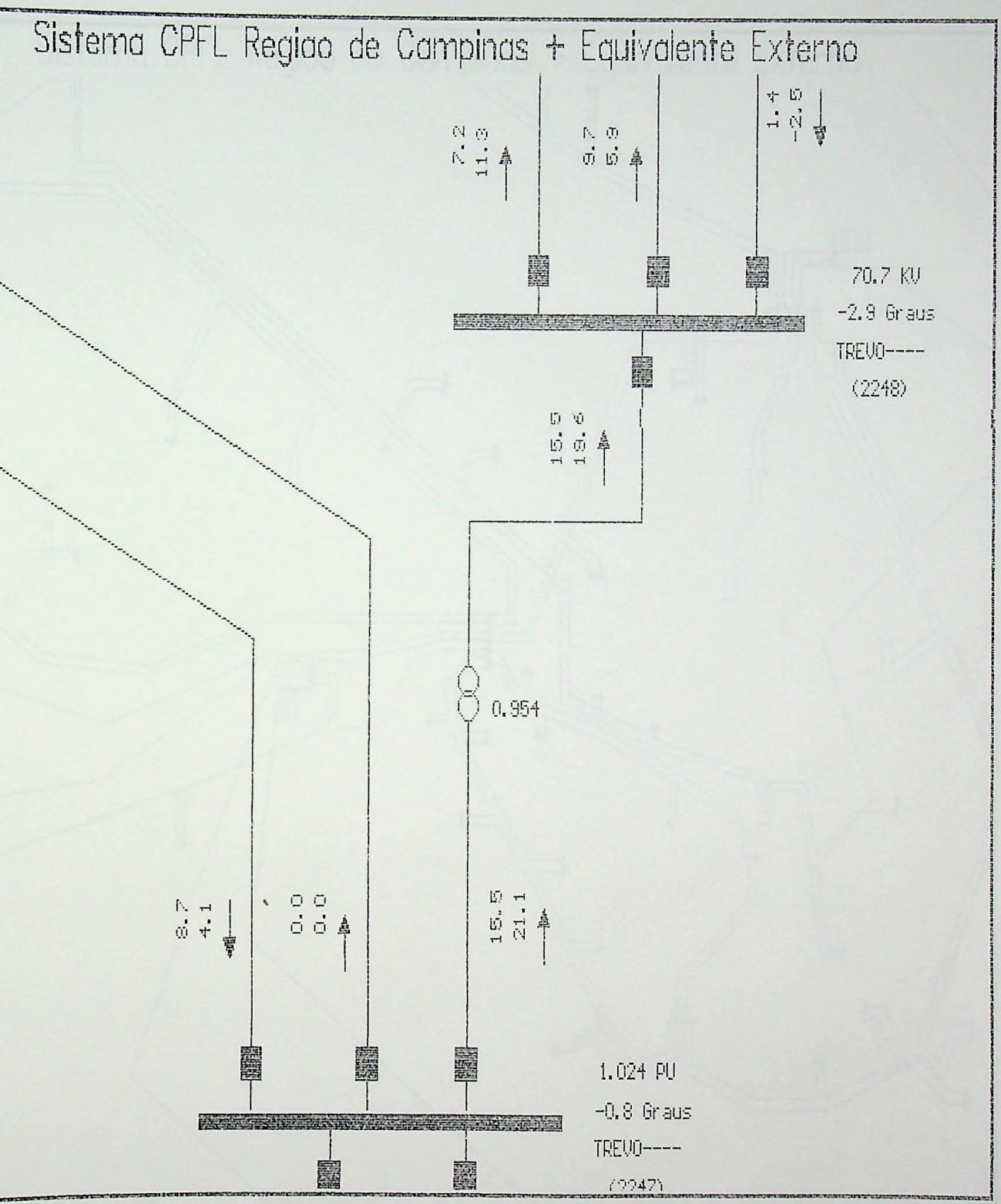
ANEXO I

EXEMPLO DE DIAGRAMA IMPRESSO

Escola Federal de Engenharia de Itajubá - EFEI
Biblioteca
MAUÁ
BIM

Sistema CPFL Regiao de Campinas + Equivalente Externo




Sistema CPFL Regiao de Campinas + Equivalente Externo
7.2
11.3
9.7
5.9
1.4
-2.5
70.7 KV
-2.9 Graus
TREVO----
(2248)
15.5
19.6
0.954
8.7
4.1
0.0
0.0
15.5
21.1
1.024 PU
-0.8 Graus
TREVO----
(2247)

Sistema CPFL Regiao de Campinas + Equivalente Externo

ANEXO II

EXEMPLO DE SAÍDA NO FORMATO TEXTO

*** Sistema CPFL Regiao de Campinas + Equivalente Externo ***

DADOS DE RAMOS

DA BARRA	PARA A BARRA	N	RESIST. P.U.	REATANCIA P.U.	SUSCEP. TOTAL P.U.	TAP P.U.	LIMITE INF. P.U.	INCR. P.U.	LIMITE SUP. P.U.	TENSAO E TOLER. ESPECIFICADAS P.U.	POT. LIM 1 MVA	POT. LIM 2 MVA
171	2241	1	.00010	.00040	.00010						0	0
171	2241	2	.00010	.00040	.00010						0	0
171	2241	3	.00010	.00040	.00010						0	0
568	2122	0	.00220	.01210	.00360						0	0
568	2185	0	.00250	.00860	.00240						0	0
568	2191	3	.00390	.02120	.00650						0	0
568	2191	4	.00390	.02120	.00650						0	0
568	2191	1	.00850	.02130	.00570						0	0
568	2212	0	.00610	.01520	.00400						0	0
568	2220	0	.00560	.03060	.00920						0	0
568	2224	1	.02770	.06860	.01830						0	0
568	2224	2	.02770	.06860	.01830						0	0
568	2234	0	.02730	.08590	.02390						0	0
572	2215	0	.00700	.02460	.00670						0	0
572	2219	1	.00480	.02440	.00730						0	0
572	2219	2	.00480	.02440	.00730						0	0
572	2219	3	.01020	.03930	.01070						0	0
680	2336	0	.00050	.00130	.00040						0	0
698	2344	0	.01690	.04430	.01060						0	0
700	2103	0	.00010	.00010	.00010						0	0
2102	2189	0	.02300	.05270	.00100						0	0
2102	2198	0	.12410	.18680	.00260						0	0
2102	2253	0	.13860	.18730	.00270						0	0
2104	2211	0	.18020	.23110	.00300						0	0
2104	2242	0	.12560	.18890	.00260						0	0
2107	2237	0	.10660	.27010	.06660						0	0
2113	2195	0	.30560	.47790	.00690						0	0
2122	2145	0	.00160	.00890	.00270						0	0
2139	2221	0	.00900	.02240	.00590						0	0
2139	2250	0	.00010	.00010	.00010						0	0
2145	2219	0	.00910	.04940	.01490						0	0
2150	2218	0	.09570	.25570	.06220						0	0
2164	2216	0	.00170	.00430	.00110						0	0
2164	2219	0	.00380	.00960	.00250						0	0
2170	2230	0	.00330	.00810	.00220						0	0
2170	2241	0	.01180	.02910	.00770						0	0
2173	2190	0	.00470	.01660	.00440						0	0
2173	2215	0	.01410	.05000	.01300						0	0
2179	2213	0	.00360	.01310	.00320						0	0
2179	2239	0	.01290	.03400	.00800						0	0
2180	2217	0	.00720	.01780	.00480						0	0
2180	2243	0	.01190	.02950	.00790						0	0
2181	2220	0	.00180	.00450	.00110						0	0
2182	2248	0	.02850	.07430	.00120						0	0
2183	2216	0	.01080	.02660	.00710						0	0
2183	2251	0	.00380	.00950	.00260						0	0
2184	2219	0	.02140	.05300	.01420						0	0
2184	2241	0	.00200	.00490	.00130						0	0
2185	2224	0	.02100	.06190	.01710						0	0
2186	2225	0	.02540	.03580	.00060						0	0

*** Sistema CPFL Regiao de Campinas + Equivalente Externo ***

DADOS DE BARRAS

NOME	TENSAO P.U.	P. GER. P.U.	Q. GER. P.U.	P.CARGA P.U.	Q.CARGA P.U.	Q. GER. MIN. P.U.	Q. GER. MAX. P.U.	TENSAO MIN. P.U.	TENSAO MAX. P.U.	BARRA CONTRO-LADORA	CAP MVAr
171	1.0490	2.378	30.990	.000	.000	-99.990	999.990				-2652.0
568		3.030	32.460	.000	.000						-2893.0
572		1.290	9.830	.000	.000						-927.0
680		.270	4.800	.000	.000						-460.0
698		.090	27.440	.000	.000						-2602.0
700		.000	6.960	.000	.000						-658.0
2102	1.0010	.060	-.144	.000	.000	-.180	.000				.0
2103		.030	.010	.037	.016						.0
2104	.9900	.030	-.071	.063	.025	-.072	.072				.0
2107		.210	18.240	.000	.000						-1742.0
2113		.000	.000	.020	.010						.0
2122		.000	.000	.125	.050						.0
2139		.000	.000	.097	.047						.0
2145		.000	.000	.076	.036						.0
2150		.000	12.030	.000	.000						-1188.0
2164		.000	.000	.017	.010						.0
2170		.000	.000	.058	.025						.0
2173		.000	.000	.018	.007						.0
2179		.000	.000	.000	.000						.0
2180		.000	.000	.000	.000						.0
2181		.000	.000	.098	.034						.0
2182		.000	.000	.097	.059						.0
2183		.000	.000	.065	.032						.0
2184		.000	.000	.090	.031						.0
2185		.000	.000	.060	.021						.0
2186		.000	.000	.060	.021						.0
2187		.000	.000	.202	.071						.0
2188		.000	.000	.269	.059						.0
2189		.000	.000	.051	.023						.0
2190		.000	.000	.086	.038						.0
2191		.000	.000	.000	.000						.0
2192		.050	.010	.000	.000						.0
2193		.000	.000	.053	.025						.0
2194		.000	.000	.057	.020						.0
2195		.000	.000	.011	.001						.0
2196		.000	.000	.076	.036						.0
2197		.000	.000	.002	.000						.0
2198		.000	.000	.051	.027						.0
2199		.000	.000	.096	.030						.0
2200		.000	.000	.085	.026						.0
2201		.000	.000	.291	.141						.0
2202		.000	.000	.197	.107						.0
2203		.000	.000	.218	.093						.0
2204		.000	.000	.037	.017						.0
2205		.000	.000	.014	.003						.0
2206		.000	.000	.075	.041						.0
2207		.000	.000	.187	.103						.0
2208		.000	.000	.265	.058						.0
2209		.000	.000	.178	.105						.0
2210		.000	.000	.053	.026						.0

*** Sistema CPFL Regiao de Campinas + Equivalente Externo ***

RESULTADOS DE RAMOS

DA BARRA	PARA A BARRA	N	POT. ENVIADA ATIVA (MW)	POT. ENVIADA REATIVA (MVAR)	POT. RECEBIDA ATIVA (MW)	POT. RECEBIDA REATIVA (MVAR)	PERDAS NA LINHA ATIVA (MW)	PERDAS NA LINHA REATV. (MVAR)	PORCENI-TAGEM	POTENCIA APARENTE REAL MVA	POTENCIA APARENTE LIM1 MVA	POTENCIA APARENTE LIM2 MVA	SOBRE I-CARGA	POSICAO DO TAP (P.U.)
2179	2239	0	3.0	.5	3.0	1.3	.00	-.83		3	0	0		
2191	2240	0	14.6	5.3	14.6	6.4	.04	-1.07		15	0	0		
2240	2202	0	4.6	2.9	4.6	3.1	.00	-.21		5	0	0		
171	2241	1	81.3	47.1	81.3	47.0	.02	.07		93	0	0		
171	2241	2	81.3	47.1	81.3	47.0	.02	.07		93	0	0		
171	2241	3	81.3	47.1	81.3	47.0	.02	.07		93	0	0		
2241	2170	0	25.8	16.0	25.7	16.5	.10	-.59		30	0	0		
2241	2184	0	28.5	23.7	28.5	23.8	.02	-.08		37	0	0		
2241	2200	0	25.6	14.0	25.4	15.2	.18	-1.20		29	0	0		
2241	2221	0	45.4	22.9	45.1	23.1	.33	-.18		50	0	0		
2241	2222	1	16.6	2.9	16.6	4.0	.02	-1.12		17	0	0		
2241	2222	2	16.6	2.9	16.6	4.0	.02	-1.12		17	0	0		
2241	2231	0	31.7	23.0	31.6	23.4	.16	-.42		39	0	0		
2242	2104	0	11.0	11.6	10.7	11.4	.31	.20		15	0	0		
2242	2204	0	5.2	-1.2	5.1	-.9	.03	-.36		5	0	0		
2241	2242	0	18.0	11.7	18.0	10.8	.00	.89		21	42	56		.9990:1
2243	2180	0	19.7	20.8	19.6	21.4	.09	-.63		29	0	0		
2241	2243	0	35.3	24.2	35.3	24.3	.03	-.06		42	0	0		
2228	2244	0	8.9	-1.9	8.9	-1.9	.00	-.02		9	0	0		
2231	2244	0	7.5	10.1	7.4	11.6	.04	-1.48		13	0	0		
2191	2245	0	30.0	7.4	30.0	7.5	.01	-.07		30	0	0		
2245	2208	0	24.2	5.4	24.1	5.7	.03	-.32		24	0	0		
2246	2238	0	9.2	1.5	9.2	2.6	.01	-1.06		9	0	0		
2187	2247	0	8.8	3.6	8.7	4.1	.01	-.51		9	0	0		
2196	2247	0	17.9	5.8	17.9	6.1	.01	-.22		18	0	0		
2248	2182	0	9.7	5.9	9.7	5.9	.04	-.03		11	0	0		
2204	2248	0	1.4	-2.6	1.4	-2.5	.00	-.06		2	0	0		
2247	2248	0	15.5	21.1	15.5	19.6	.00	1.51		26	42	42		.9540:1
2193	2249	0	1.0	2.2	1.0	2.4	.00	-.19		2	0	0		
2227	2249	0	11.9	2.8	11.8	3.1	.01	-.32		12	0	0		
2139	2250	0	29.4	16.7	29.4	16.7	.00	-.01		33	0	0		
2183	2251	0	10.6	.0	10.6	.3	.00	-.26		10	0	0		
2214	2251	0	1.8	-1.8	1.8	-1.7	.00	-.10		2	0	0		
2233	2251	0	10.7	12.1	10.6	12.6	.02	-.47		16	0	0		
2251	2247	1	5.0	7.1	5.0	7.6	.01	-.48		9	0	0		
2251	2247	2	5.0	7.1	5.0	7.6	.01	-.48		9	0	0		
2251	2250	0	5.3	-4.6	5.3	-3.2	.01	-1.38		6	0	0		
2252	2218	0	1.5	4.7	1.5	5.8	.00	-1.06		5	0	0		
2224	2252	0	9.1	9.0	9.1	9.0	.00	-.09		12	0	0		
2102	2253	0	6.8	-4.5	6.7	-4.4	.09	-.15		8	0	0		
2248	2253	0	7.2	11.3	7.0	11.3	.20	-.03		13	0	0		
680	2336	0	27.5	6.0	27.5	6.1	.00	-.04		28	0	0		
2336	2246	0	15.2	3.1	15.1	3.8	.03	-.78		15	0	0		
698	2344	0	9.3	1.2	9.3	2.3	.01	-1.09		9	0	0		

PERDAS DO SISTEMA

	ATIVAS MW	REATIVAS MVAR
SERIE	5.17	17.55
SHUNT		-79.90
TOTAL	5.17	-62.35

DATA 19/10/1995
PROC.
PED PPG/EFEI
L.V.
RS Doação

I V N O
BIM - EFEI

DATA	Rubrica

FICHA 01 / T.802

EFEI / Biblioteca Mauá
621.316:519.674:681.322(043.2)
P142p
OSCAR NAVARRO: 1160
1161 → JULIO NAVARRO
PAIVA, J.G.
Programa Computacional para Simulaçao e Análise de Sistemas Elétricos de Potência com Recursos Gráficos Interativos.

...ada	N.º Registro	D... Devo

EFEI - BIBLIOTECA MAUÁ
8200802
NÃO DANIFIQUE ESTA ETIQUETA

EFEI
BIBLIOTECA MAUÁ

Esta publicação deverá ser devolvida dentro do prazo estipulado.

O leitor é responsável pela publicação em seu poder.