PRÉ E PÓS-PROCESSAMENTO DE MODELOS RETICULADOS E DE SEÇÕES TRANVERSAIS COMPOSTAS NO INSANE

Carlos Henrique Belém Vilela; Débora Coelho Cordeiro; Gabriel de Oliveira Ribeiro;

Roque Luiz Pitangueira; Samuel Silva Penna

Departamento de Engenharia de Estruturas, Universidade Federal de Minas Gerais, Av. Antônio Carlos, 6627, Bloco 1, Belo Horizonte - MG - Brasil
31270-901
e-mail: carlosbelem@ufmg.br, deboracordeiro@ufmg.br, gabriel@dees.ufmg.br, roque@dees.ufmg.br, spenna@dees.ufmg.br

Resumo. *Este artigo apresenta os recursos desenvolvidos para o pré e pós processamento de modelos estruturais reticulados do INSANE – INteractive Structural ANalysis Environment, bem como as tecnologias relacionadas a esse processo. Por meio do paradigma de programação orientada a objetos, com a utilização da linguagem JAVA, de suas APIs (Application Program Interface), de padrões de projeto de software apropriados e da biblioteca gráfica JOpenGL, foram criadas aplicações gráficas interativas para modelagem geométrica, pré e pós-processamento de modelos reticulados e seções transversais compostas. Como forma de armazenamento e organização dos dados do programa, foi utilizada uma estrutura de semi-arestas que possibilita a manipulação acessível e estruturada de construções e operações geométricas, além de outras informações relevantes. Para a etapa de pré-processamento, foram criados recursos para a geração de malha de elementos unidimensionais e prescrição de atributos ao modelo, tais como restrições, carregamentos, materiais e seções transversais. Ainda, foi desenvolvida uma aplicação gráfica específica que permite a modelagem de seções transversais compostas por diversos materiais, além de permitir o cálculo de suas propriedades geométricas, seja por meio do Teorema de Green ou por meio da discretização de seu domínio. Para o processamento de modelos reticulados lineares e geométrica ou fisicamente não-lineares em um ambiente unificado, foram utilizados os recursos já existentes no núcleo numérico do sistema, sem qualquer alteração nas implementações dos mesmos. Na etapa de pós-processamento, são fornecidos diagramas de esforços solicitante e isofaixas, sendo este também disponível no pós-processamento das seções transversais compostas.*

Palavras-chave: *Modelos Estruturais Reticulados, Método de Elementos Finitos, Computação Gráfica, Programação Orientada a Objetos.*

1. INTRODUÇÃO

A utilização de modelos discretos de análise estrutural compreende três etapas principais inter-relacionadas: (1) criação do modelo, (2) montagem e resolução do modelo e (3) avaliação de resultados. Na criação do modelo, o analista informa as hipóteses simplificadoras relativas à geometria, material, carregamento e condições de contorno. Estas são representadas com entidades matemáticas apropriadas, gerando a malha e os atributos do modelo. Na etapa de montagem e resolução do modelo, combinam-se as informações matematicamente representadas, de modo a produzir equações algébricas que, quando solucionadas, permitem obter as diversas grandezas. Na avaliação de resultados, o analista faz a crítica e verifica a adequação dos mesmos ao problema em estudo. Para disponibilização deste processo em computadores, normalmente a referida divisão é adotada através de programas de pré-processamento (para a criação dos modelos com recursos gráficos interativos), processamento (para a montagem e resolução numérica do modelo) e pós-processamento (para visualização gráfica de resultados).

A análise estrutural por meio de modelos discretos do Método dos Elementos Finitos (MEF) pressupõe a divisão do domínio de estudo em sub-domínios interconectados, denominados elementos finitos. Na formulação do MEF em deslocamentos, cada um dos elementos finitos tem uma função de aproximação de deslocamentos a ele associada. Nestes elementos também é necessário estabelecer hipóteses relativas ao regime de deformações (relações Deformações *versus* Deslocamentos) e ao comportamento do material (relações Tensões *versus* Deformações). Utilizando princípios físicos apropriados, estas hipóteses são consideradas na obtenção das equações de equilíbrio nodais do elemento, gerando assim, sua matriz de rigidez. Os vários elementos de uma discretização do MEF são interconectados entre si através de nós. A imposição das condições de equilíbrio a cada um dos nós da malha, considerando as equações de equilíbrio do elemento e a compatibilidade de deslocamentos nos nós, permite obter o sistema de equações de equilíbrio do modelo. Este sistema, inicialmente singular, pode ser resolvido através da imposição das condições de contorno.

Modelos unidimensionais para vigas, pórticos, treliças e grelhas, genericamente designados modelos reticulados, são representativos de peças cujas seções transversais apresentam dimensões muito inferiores ao comprimento e que, portanto, podem ser degeneradas em um ponto. Neste caso, o cálculo de grandezas internas, dados os deslocamentos nodais, envolve a obtenção da variação das mesmas, tanto no domínio unidimensional do reticulado, quanto no domínio bidimensional das seções transversais degeneradas. Assim, a utilização de recursos gráficos interativos para pré e pós-processamento, tanto do reticulado quanto das seções degeneradas, é imprescindível, principalmente se as seções são compostas por vários materiais. Estes recursos foram incorporados ao sistema `INSANE` com o trabalho de iniciação científica desenvolvido pelos dois primeiros autores, cujos resultados este artigo apresenta.

2. O SISTEMA INSANE

As possibilidades que os recursos tecnológicos para desenvolvimento de software oferecem para cada uma das etapas da resolução de modelos estruturais constituem amplo campo de pesquisa na área de métodos numéricos e computacionais aplicados à engenharia. O domínio destes recursos e a aplicação dos mesmos no aprimoramento progressivo dos modelos, sem ter que recomeçar o processo a cada novo aperfeiçoamento, requer um ambiente computacional segmentado, amigável a mudanças e escalável em complexidade.

O `INSANE` é um projeto de software livre, em desenvolvimento no Departamento de Engenharia de Estruturas da Universidade Federal de Minas Gerais, que visa disponibilizar um ambiente computacional de código aberto, com características tais que facilitem sua expansão

e manutenção, evitando o recomeço, tão comum no desenvolvimento de softwares acadêmicos.

Os seus pré e pós-processadores são aplicações gráficas interativas, implementadas na linguagem JAVA, que disponibilizam, respectivamente, ferramentas de pré e pós - processamento de diferentes modelos discretos. O seu processador é uma aplicação, também implementada em Java, que representa o núcleo numérico do sistema, sendo o responsável pela obtenção dos resultados de diferentes modelos discretos de análise estrutural. A linguagem JAVA é utilizada no desenvolvimento das aplicações do `INSANE` por possuir vantagens como a portabilidade, que possibilita a criação de um software cujo funcionamento independe do sistema operacional, e o suporte ao paradigma da Programação Orientada a Objetos, facilitando a expansão e o trabalho em equipe.

Cada uma destas aplicações é baseada em uma combinação dos padrões de projeto de software *Model-View-Controller* (*MVC*), *Command* e *Observer* (Gamma et al., 1995), como ilustrado na Fig. 1.

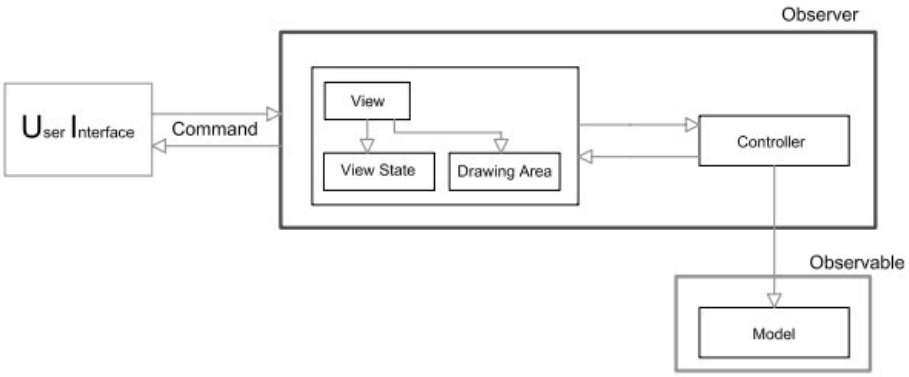


Figura 1: Arquitetura das aplicações do INSANE.

Visando separar o modelo de sua representação, a utilização do *MVC* permite que o controle da interação com o usuário e a visualização das respostas sejam implementados independente do modelo adotado, minimizando as tarefas de manutenção e expansão da aplicação. A implementação segundo o padrão *MVC* permite o aperfeiçoamento gradual da aplicação através de mudança de plataforma, criação de diversas vistas sincronizadas com o modelo, substituição ou atualização das diversas vistas e disponibilização "on-line" do sistema.

O uso do padrão *Command* na implementação do sistema permite o encapsulamento de rotinas de execução em objetos, a associação destes objetos a elementos de interface gráfica com o usuário (GUI) e dispositivos de entrada (teclado e mouse), a execução de uma mesma rotina disparada por diferentes elementos de GUI e possibilita um incremento na modularidade de seu código. O encapsulamento das rotinas de execução possibilita também que a realização de alterações nas mesmas não provoque modificações nas classes existentes (Gonçalves, 2004).

Como mostra a Fig. 1, um mecanismo de propagação de mudanças garante a consistência e a comunicação entre os componentes *Controller* e *View* com o componente *Model*. Este mecanismo baseia-se na observação de alterações do modelo e ocorre segundo o padrão de projeto *Observer-Observable*. Quando um objeto dito *observador* é criado, ele é inscrito na lista de observadores dos objetos ditos *observados*. Quando alguma mudança ocorre no estado de um objeto *observado*, é disparado então o mecanismo de propagação de mudanças, que se encarrega de notificar os objetos *observadores* para se atualizarem. Isto garante a consistência e a comunicação entre o componente *observador* e os componentes *observados*.

O trabalho de iniciação científica que aqui se apresenta envolveu: criação de interfaces

gráficas de pré e pós-processamento específicas para modelos estruturais reticulados e a incorporação das mesmas no repositório de aplicações gráficas do sistema; alteração dos recursos de desenho de todo o sistema que passaram a utilizar a *JOpenGL*, versão adotada pela *SUN* da biblioteca nativa *OpenGL*; a utilização da implementação `INSANE` da estrutura de semi-arestas para tratamento dos dados de modelos reticulados;

3. REPOSITÓRIO DE APLICAÇÕES GRÁFICAS

O Repositório de Aplicações Gráficas do `INSANE` é reponsável por integrar, em uma interface gráfica genérica, os elementos de interação com o usuário definidos por cada aplicação específica. Conforme mostra a Fig. 2, os elementos que compõem o *Barra de Menu*, as *Barras de Ferramentas* (superior, lateral esquerda e lateral direita), a *Área de Desenho* e a *Barra de Status* são definidos por cada aplicação específica e exibidos na interface genérica.

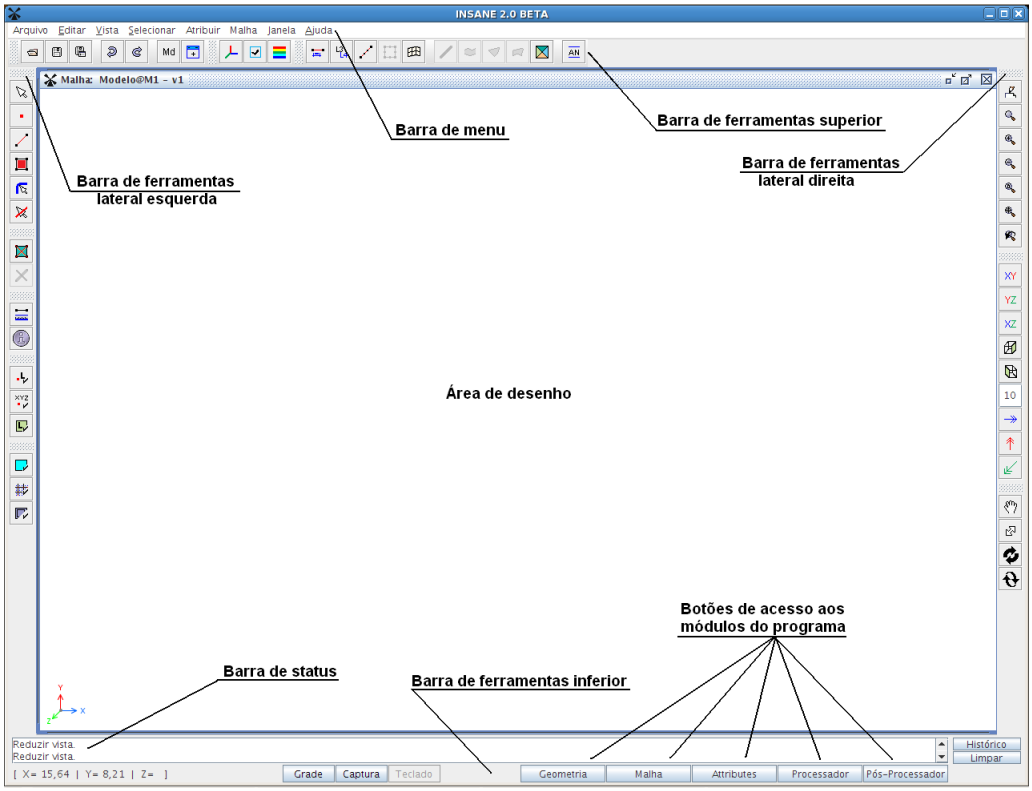
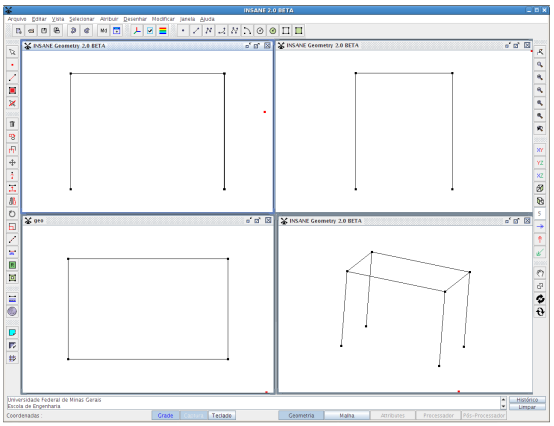


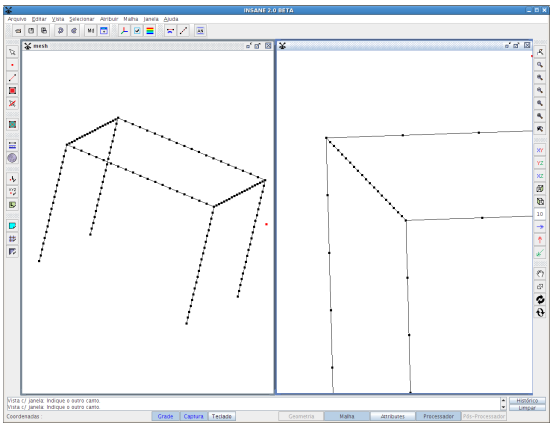
Figura 2: Esquema da interface gráfica que integra as aplicações INSANE.

A exibição dos elementos de interação com o usuário é gerenciada pelo Repositório de Aplicações Gráficas, através dos botões da *Barra de Ferramentas Inferior* (Fig. 2). Estes botões estão distribuídos em dois (2) grupos, a saber: os botões *Grade*, *Captura* e *Teclado* formam o grupo que gerencia a área de desenho para um aplicação específica; os botões *Geometria*, *Malha*, *Atributos*, *Processamento* e *Pós-Processamento* formam o grupo que gerencia as etapas de modelagem e resolução para uma aplicação específica.

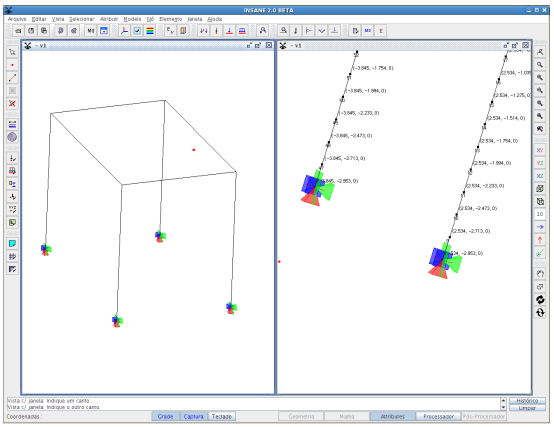
A Fig. 3 ilustra a capacidade de integração proporcionada pelo Repositório de Aplicações Gráficas, destacando-se a mudança automática das barras de ferramentas e menus de acordo com a aplicação ativa, o uso comum dos recursos de visualização (barra lateral direita) e de seleção (parte superior da barra lateral esquerda) por todas as aplicações, a visualização simultânea de várias vistas de uma mesma aplicaccão, bem como a visualização simultânea de várias aplicações.



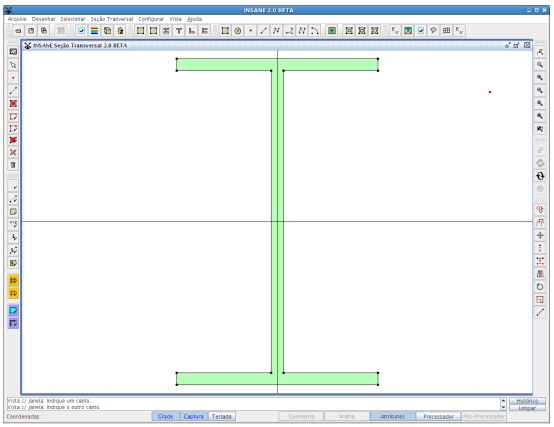
(a) Geometria do reticulado



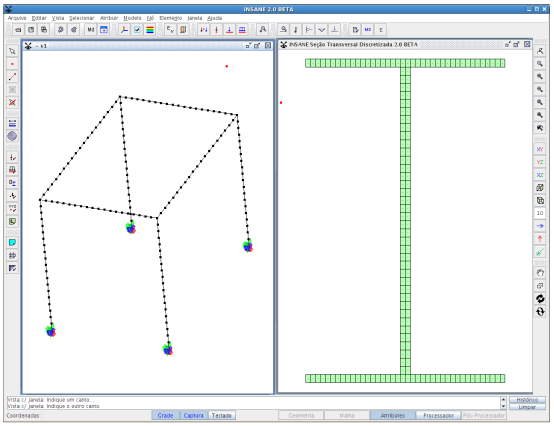
(b) Malha do reticulado



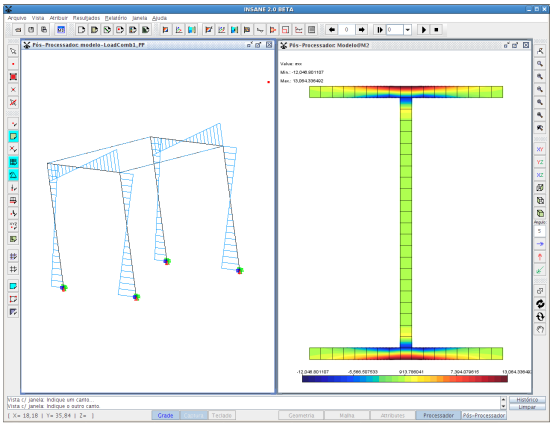
(c) Atributos do reticulado



(d) Geometria de seção



(e) Aplicações simultâneas



(f) Vistas simultâneas

Figura 3: Adaptação da interface à aplicação corrente.

O Repositório de Aplicações Gráficas foi implementado segundo o projeto orientado a objetos da Fig. 4. Conforme mostra a Fig. 4, a classe abstrata *Interface* implementa métodos de *InternalFrameListener* (pacote *javax.swing.event*) para permitir o uso de múltiplas janelas internas (relação um para n). No momento em que uma janela de uma aplicação específica é ativada, um método chamado *updateOwner()*, definido na classe *InternalInterface* é acionado.

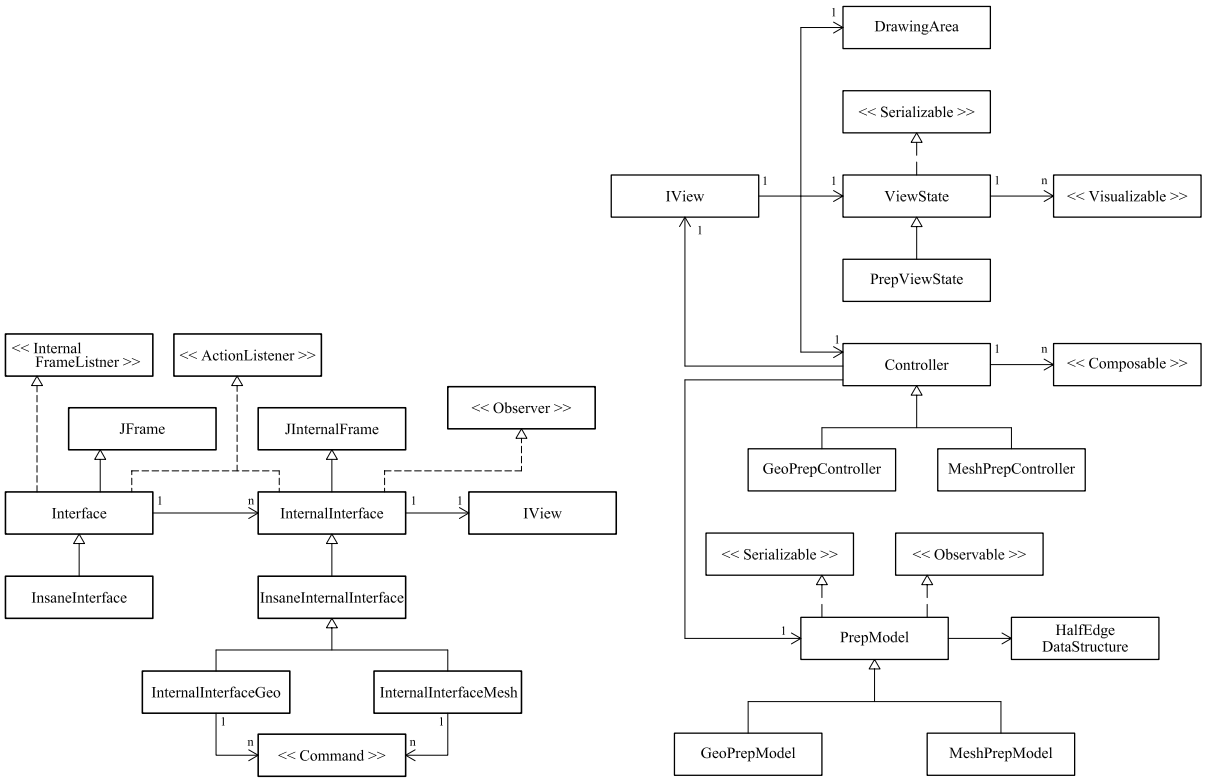


Figura 4: Projeto OO da interface gráfica que integra as aplicações INSANE.

Esse método atualiza a interface gráfica (GUI) com menus e barras de ferramentas específicos da aplicação corrente. Nas classes das aplicação específicas (*InternalInterfaceGeo* e *InternalInterfaceMesh* na Fig. 4) são declarados vários comandos que são associados à cada item de menu ou botão das barras de ferramentas, conforme estabelece o padrão *Command*.

A classe *InternalInterface* é derivada de *JInternalFrame* (pacote *javax.swing*) e implementa a interface *ActionListener* (pacote *java.awt.event*) para acionar o método *execute()* de cada comando. A classe *InternalInterface* implementa também a interface *Observer* (pacote *java.util*) para ser notificada quando da atualização do modelo.

A observação do modelo é feita através de uma vista, representada pela classe *IView* (detalhada na Fig.4). A *vista* é formada por uma área de desenho (classe *DrawingArea*), por um estado da vista (classe *ViewState*) e por um controlador (classe *Controller*). O estado da vista é o responsável pelo visual dos elementos gráficos e pode ser formado por várias classes que implementam a interface *Visualizable*. O controlador cuida da composição dos desenhos que representam o modelo e pode ser formado por vários "compositores" (classes que implementam a interface *Composable*).

4. DESENHOS COM *JOpenGL*

As novas interfaces gráficas desenvolvidas neste trabalho contaram com a biblioteca gráfica, a *JOpenGL (Java Open Graphics Library)*, uma especificação aberta e multiplataforma de uma biblioteca de rotinas gráficas e de modelagem. A *JOpenGL* é bastante eficiente, uma vez que muitos dos seus algoritmos são implementados em hardware nas placas gráficas mais modernas. Além do desenho de primitivas gráficas, tais como linhas e polígonos, *JOpenGL* dá suporte a iluminação, colorização, mapeamento de textura, transparência, animação e tranformações geométricas. A utilização desta biblioteca no desenvolvimento do INSANE melhorou o desempenho de toda a interface gráfica. Além da melhora de desempenho, a adoção de *JOpenGL*

em lugar de *Graphics2D*, usada nas versões anteriores, resultou em melhora significativa de qualidade dos desenhos. A Fig. 5 ilustra essa melhora comparando os desenhos de cargas concentradas implementados com as duas bibliotecas.

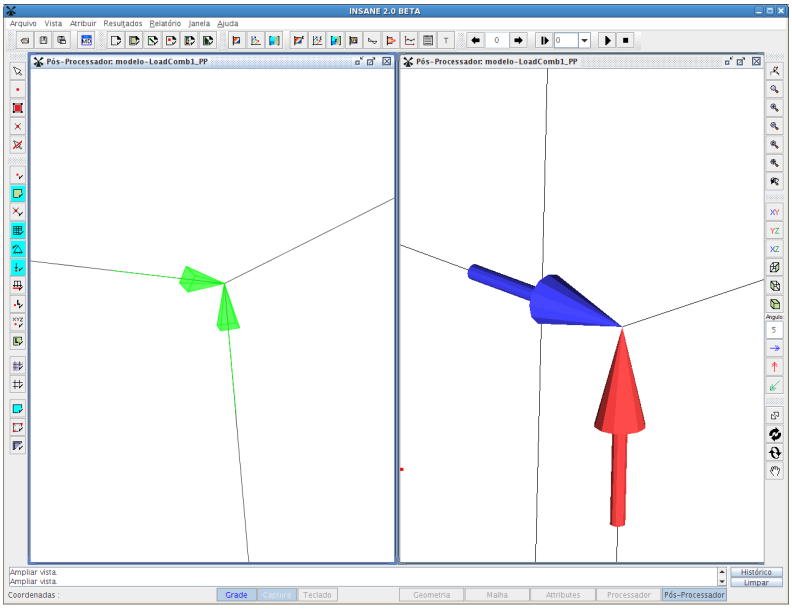


Figura 5: Comparação entre os recursos das bibliotecas *Graphics2D* e *JOpenGL*

5. ESTRUTURA DE DADOS

Como forma de armazenamento e organização dos dados do programa `INSANE`, é utilizada uma estrutura de semi-arestas (*Half-Edge Data Structure*), que possibilita a manipulação acessível e estruturada de construções e operações geométricas, além de outras informações relevantes. A estrutura de semi-arestas apresenta um melhor desempenho na busca sendo sua implementação no `INSANE` feita por Penna (2007).

Essa estrutura de dados é desenvolvida em uma hierarquia, ilustrada na Fig. 6, composta por cinco diferentes níveis, sendo eles a *Subdivisão Plana*, a *Face*, o *Loop*, a *Semi-aresta* e o *Vértice*. Esses elementos são armazenados na forma de listas duplamente encadeadas, o que facilita operações com os objetos, tais como adição, remoção e organização. A *Subdivisão Planar* possui referências que permitem o acesso ao nível imediatamente inferior, nível *Face*. Este nível, por sua vez, representa as faces da subdivisão planar, sendo ainda interligado às demais faces do modelo. Cada face é composta por um ou mais *loops*, que podem ser externos ou internos. As faces possuem, portanto, referências tanto para a subdivisão planar à qual pertencem quanto aos *loops* de que são formadas. O nível *Loop* contém *loops* que, como dito anteriormente, compõem as faces do modelo. Assim, esse nível é interligado ao nível imediatamente superior, nível *Face*. Ainda, um *loop* é composto de uma ou mais semi-arestas, possuindo referências ao nível *Semi-Aresta*. Quando um *loop* é externo, este faz parte do contorno de uma face. Por outro lado, quando o *loop* é interno, ele está associado ao contorno de um furo no interior da face a que está associado. O nível *Semi-Aresta*, por compor *loops*, possui referências ao nível *Loop*, além de ser interligado ao nível *Vértice*. Cada semi-aresta possui um vértice, que representa o início da respectiva semi-aresta. Por fim, o nível *Vértice* é interligado ao nível imediatamente superior, nível *Semi-Aresta*, e é composto por vértices em uma lista duplamente encadeada. A fim de compor o modelo geométrico através da conexão entre as faces, é empregado um nível intermediário *Aresta*, como mostra a Fig. 6. Esse nível possui ligação com

os níveis *Semi-Aresta*, pois uma aresta é formada de duas semi-arestas em sentidos opostos e incidentes no mesmo vértice, e também com o nível *Subdivisão Planar* (Penna, 2007).

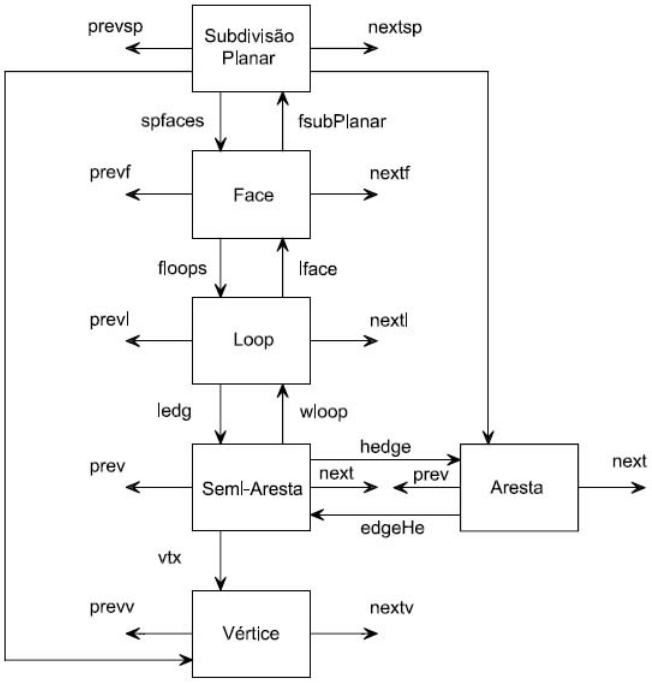


Figura 6: Elementos da estrutura de dados.

6. RECURSOS DO PRÉ-PROCESSADOR DE RETICULADOS

Foram desenvolvidos recursos de desenho, edição e visualização para o pré - processamento dos modelos reticulados.

Conforme mostra a Fig. 7, a barra de ferramentas superior conta com comandos de manipulação de arquivos e configurações. Para auxiliar na criação e edição do desenho, a aplicação conta com outras duas barras de ferramentas: a lateral esquerda, composta por comandos de seleção e edição, e a lateral esquerda, que agrupa os comandos de visualização. A seleção de desenhos é feita através de clique com mouse, interseção com uma linha, seleção por janela ou seleção por nome (identificador do desenho). Ainda, é possível restringir a seleção a um ou mais grupos de desenhos, como por exemplo selecionar apenas vértices no desenho da geometria. Dentre os comandos de visualização, é possível afastar, aproximar, rotacionar e enquadrar a vista, assim como utilizar projeções ortogonais, como plano XY, YZ ou XZ, ou vistas em perspectiva. Todos os comandos disponibilizados nas barras de ferramentas, assim como outros, podem ser também acessados por meio de menus, localizados na parte superior da interface (ver Fig. 7).

Os comandos de desenho podem ser executados de duas maneiras: via mouse, o que possibilita maior agilidade de desenho, ou via teclado (ex: entrada de coordenadas), que por sua vez possibilita maior precisão e controle. Estes comandos são específicos para cada uma das três etapas do pré-processamento, a saber: o desenho da geometria (Fig. 8), a geração de malhas (Fig. 9) e a especificação de atributos (Fig. 10).

Na etapa de definição da geometria (Fig. 8), são disponibilizados comandos de desenho de pontos, linhas, multi-linhas, quádricas, cúbicas, arcos, circunferências e retângulos. São disponibilizados também comandos de edição, como copiar, espelhar, mover, rotacionar, escalonar, dividir e remover, assim como comandos de inferência, como medir a distância entre dois

vértices ou informações sobre um elemento.

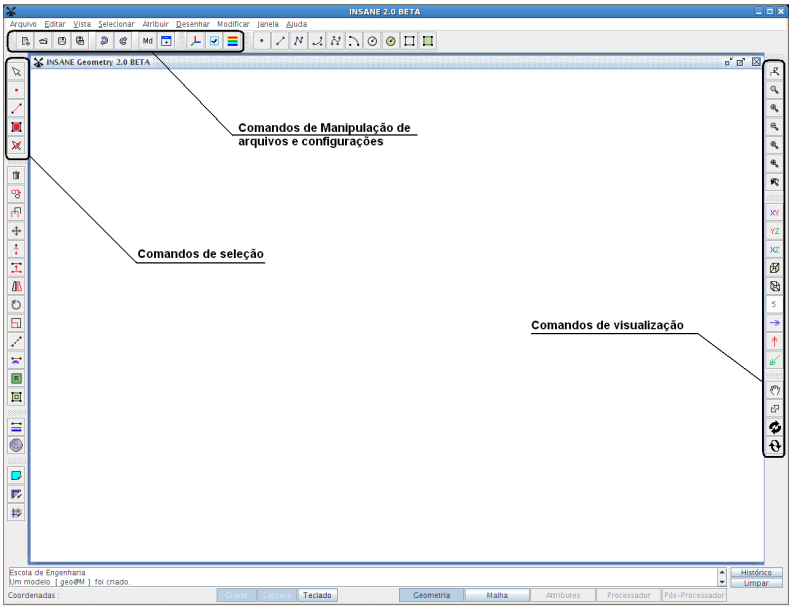


Figura 7: Recursos comuns a todas as etapas do pré-processador.

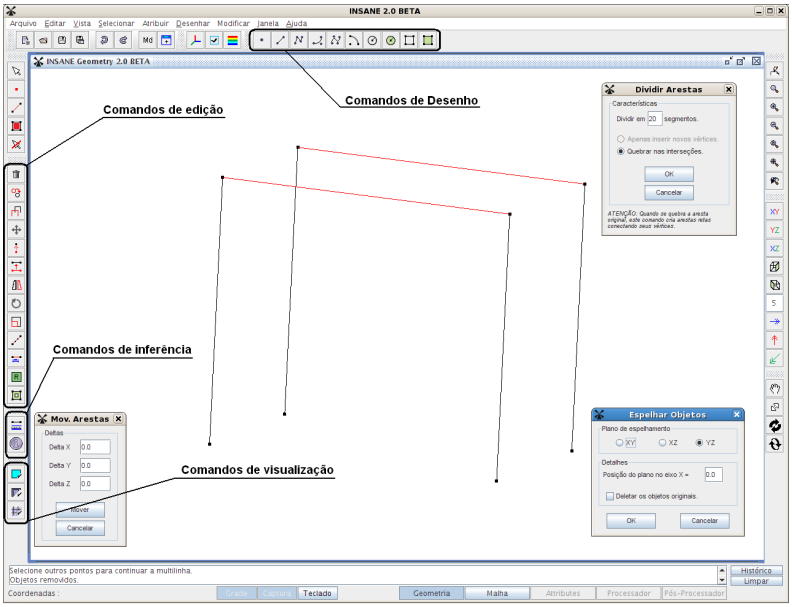


Figura 8: Barras de ferramentas da interface `Geometria`.

A etapa seguinte compreende a geração de malhas (Fig. 9), que possui comandos de divisão de arestas, assim como os demais comandos de seleção, inferência e visualização. Um comando exclusivo desta etapa é o de seleção do modelo de análise, que é necessário para definir os atributos compatíveis, assim como para o processamento e pós-processamento do modelo.

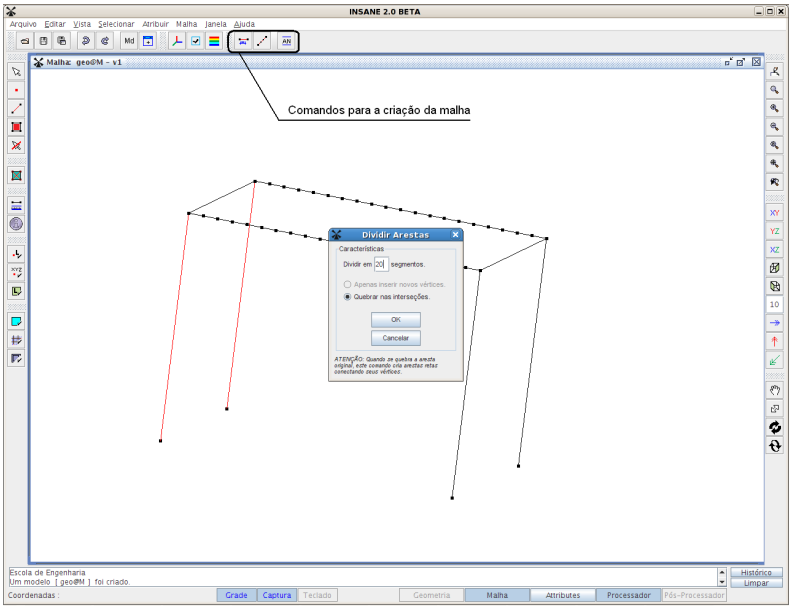


Figura 9: Barra de ferramentas da interface `Malha`.

Após a geração de malhas, o usuário é levado à etapa de especificação de atributos (Fig. 10). Nesta, ele conta com comandos de atribuições nodais e de elementos, tais como restrições e cargas nodais, liberação de extremidade, variação de temperatura, pré-deslocamentos e carregamento de barras. Existem, ainda, comandos de atribuição de seções transversais e materiais às barras.

A criação de seções tranversais pode ser feita por meio do fornecimento de suas propriedades geométricas via teclado ou por meio de um editor de seções especialmnete criado para este fim.

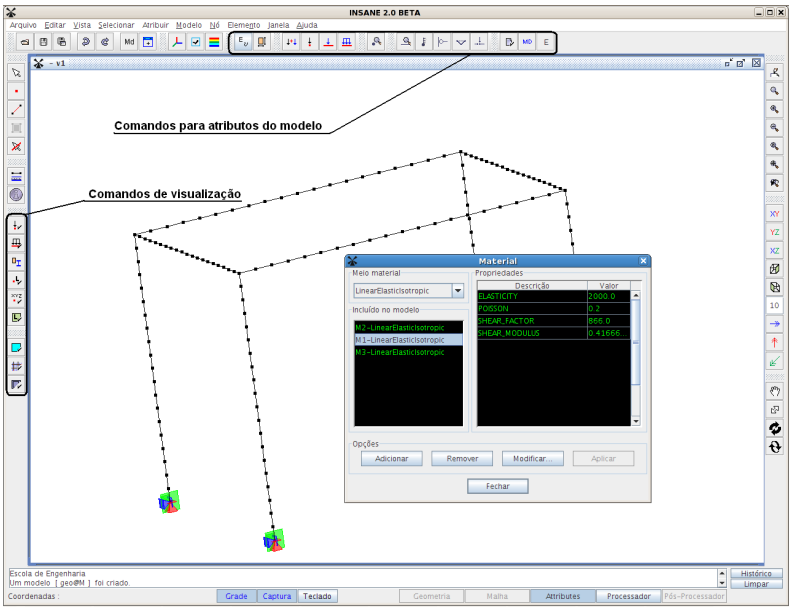


Figura 10: Barras de ferramentas da interface `Atributos`.

7. RECURSOS DO PRÉ-PROCESSADOR DE SEÇÕES TRANSVERSAIS

O editor de seções transversais é capaz de realizar as seguintes ações: (1) definir a geometria da seção; (2) criar materiais e atribuí-los à seção; (3) discretizar seu domínio em pontos materiais; (4) calcular suas propriedades geométricas; (5) armazenar os dados na forma de uma estrutura de semi-arestas e (6) exportar a seção. Essas medidas foram úteis para que se obtivessem os dados necessários para o processamento e consequente pós-processamento da seção.

Conforme ilustra a Fig. 11, para a modelagem geométrica de seções (ação 1), estão disponíveis comandos de desenho de seções retangulares (fornecendo base e altura ou os dois vértices da diagonal), poligonais e circulares (fornecendo centro e raio), além dos perfis I, T, L, C (fornecendo as espessuras e dimensões da alma e na mesa). Os mesmos comandos de desenho de arestas e vértices existentes na interface *Geometria* estão disponíveis nesta interface, sendo utilizados para a posterior criação regiões. Existe também a possibilidade de criação de furos nas regiões já existentes.

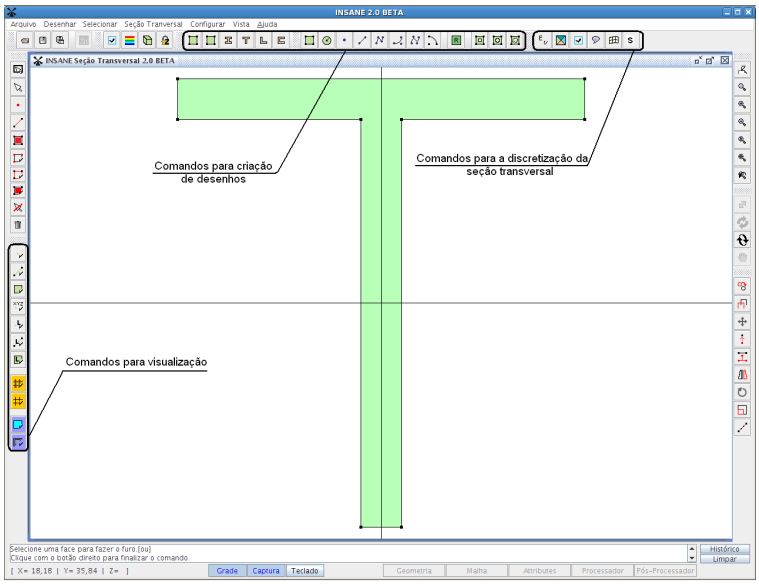


Figura 11: Barras de ferramentas da interface `Seção Transversal`.

Após a criação da geometria da seção transversal, é possível utilizar os comandos que realizam as ações (2), (3) e (4) listadas acima. Esses comandos estão dispostos à direita da barra de desenho, na extremidade superior da interface (Fig. 11). O comando de criação de materiais abre um diálogo que permite que o usuário crie ou modifique materiais ao escolher suas propriedades. A Fig. 12 mostra o diálogo de criação de um material. Uma vez criados, os materiais podem ser atribuídos às faces do modelo, sendo possível a atribuição de um material para cada uma delas, formando seções compostas.

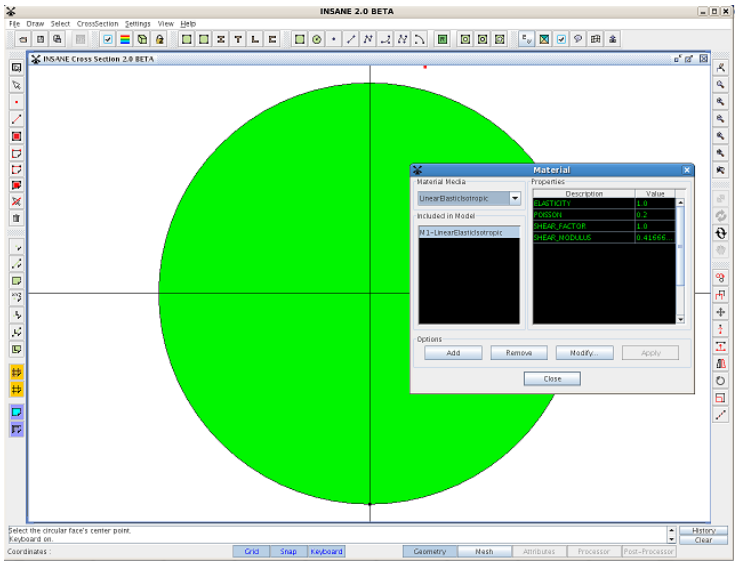


Figura 12: Diálogo de criação de materiais.

A aplicação também possui um comando de discretização do domínio da seção em pontos materiais. Quando esse comando é acionado, pergunta-se ao usuário em quantas divisões em cada direção ele deseja que a discretização ocorra. Em seguida, uma nova interface é aberta (Fig. 13) e, com base nos valores obtidos no diálogo e nas coordenadas das extremidades do modelo, são estabelecidas as coordenadas x e y e as dimensões dx e dy de cada ponto material. A cada ponto material criado corresponde uma face da estrutura de semi-arestas. A discretização é realizada levando em conta os materiais já associados às faces e as cores associadas a eles, entretanto, mesmo na nova interface, é possível criar novos materiais e atribuí-los a cada um dos pontos materiais.

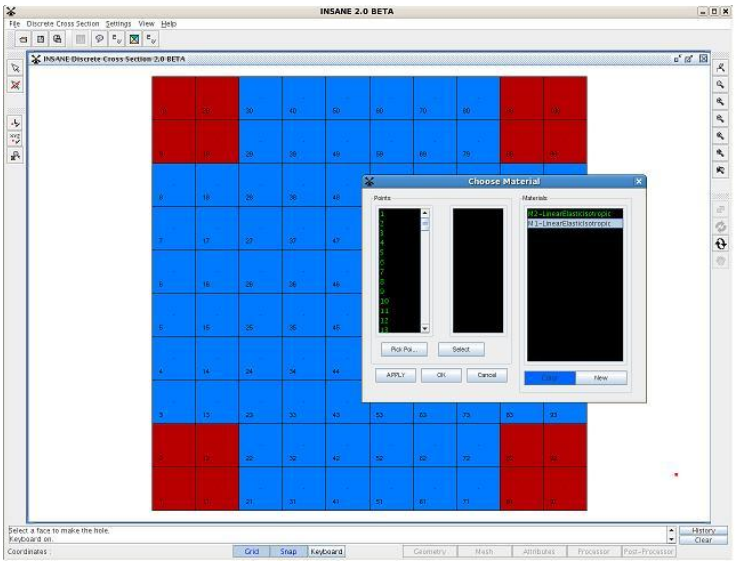
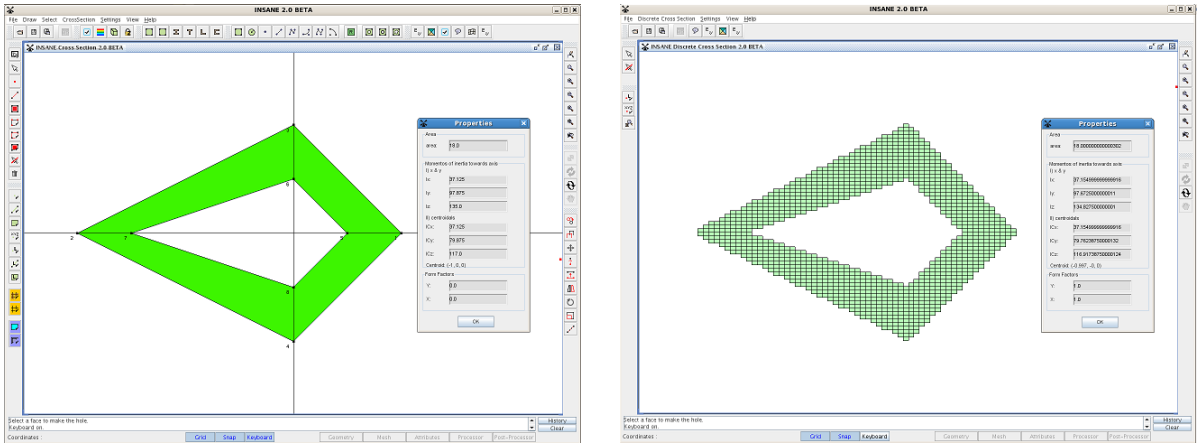


Figura 13: Seção composta discretizada em 100 pontos materiais.



(a) Seção Poligonal (b) Seção Poligonal Discretizada

Figura 14: Cálculo das propriedades da seção.

O comando de cálculo das propriedades geométricas da seção mostra sua área, as coordenadas de seu centróide e seus momentos de inércia em um diálogo (Fig. 14). O cálculo dessas propriedades pode ser realizado a partir da aplicação do Teorema de Green no plano (Fig. 14(a)) ou utilizando a discretização da seção em pontos materiais (Fig. 14(b)). A Fig. 14 mostra os dois quadros de propriedades obtidos. Ao se comparar os resultados de antes e depois da discretização, observa-se que os valores obtidos por último são bem próximos da solução analítica obtida pelo Teorema de Green.

Uma vez criada, a seção tranversal pode ser exportada para a interface de atributos. No caso da seção discretizada, os dados geométricos e de material armazenados na estrutura de semi-arestas são utilizados na criação de pontos materiais. No caso da seção não discretizada, são exportadas as propriedades da seção calculadas através do Teorema de Green.

8. RECURSOS DO PÓS-PROCESSADOR DE RETICULADOS

O núcleo numérico do `INSANE` possui recursos para análise linear e geometrica ou fisicamente não-linear de modelos reticulados. Os recursos para análise linear foram implementados por Fonseca et al. (2004). A análise fisicamente não-linear de reticulados com seções compostas foi implementada por Fonseca (2006). Algumas formulações para análise geometricamente não-linear foram incorporadas por Fonseca (2008). Os processos de obtenção de trajetórias de equilíbrio, fundamentais nas análises não-lineares, foram implementados por Fuina (2009).

Para integração dos recursos de processamento feita pela aplicação gráfica aqui apresentada, foram utilizados os recursos já existentes no núcleo numérico do sistema, sem qualquer alteração nas implementações dos mesmos.

O pós-processamento de modelos reticulados envolve a visualização de grandezas, tanto no domínio unidimensional do reticulado, quanto no domínio bidimensional das seções transversais. As Fig. 15, Fig. 16 e Fig. 17 ilustram os recursos disponibilizados: visualização da estrutura deformada, dos diagramas de deslocamentos e esforços solicitantes nas barras, das trajetórias de equilíbrio dos nós e de tensões e deformações, feita por meio de isofaixas, para as seções transversais.

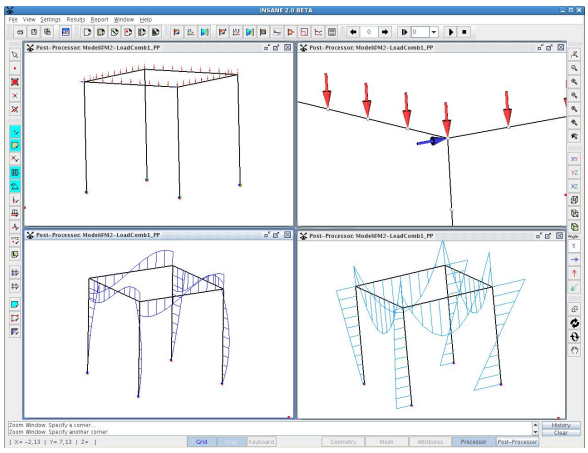


Figura 15: Visualização de resultados: análise linear de pórtico espacial.

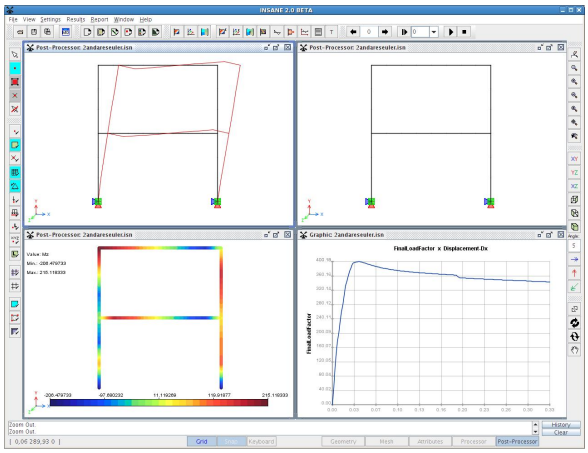


Figura 16: Visualização de resultados: análise fisicamente não-linear de pórtico plano.

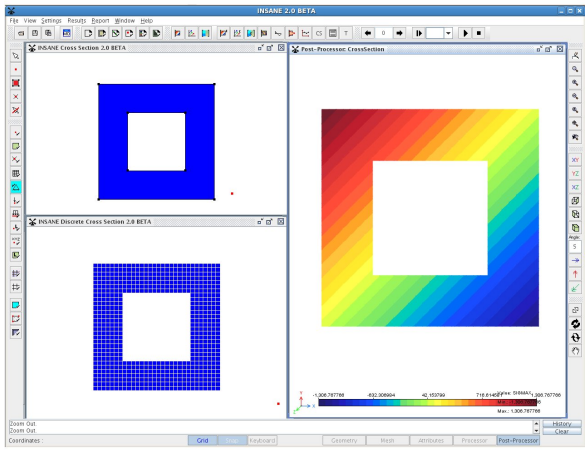


Figura 17: Visualização de resultados de seções transversais (isofaixas).

9. CONSIDERAÇÕES FINAIS

O desenvolvimento deste trabalho, seguindo os já antes adotados padrões de projeto de software e substituindo a biblioteca gráfica *Graphics2D* pela *JOpenGL*, vialibilizou a criação de no-

vos recursos de pré e pós-processamento de modelos estruturais reticulados e suas degenerações de forma organizada e eficiente.

Com a criação do Repositório de Aplicações Gráficas, qual nova interface gráfica para qualquer nova aplicação pode ser facilmente incorporada ao sistema.

Os recursos do editor de seções transversais mostraram-se extremamente úteis, facilitando muito o pré-processamento de modelos reticulados, principalmente no caso de análise fisicamente não-linear com seções compostas.

A disponibilização dos diversos recursos em um ambiente gráfico unificado permite dar saltos qualitativos relevantes no processo de análise e interpretação de resultados dos modelos.

Os recursos disponíveis na nova biblioteca gráfica adotada ainda não foram explorados por completo, o que possibilita a contínua melhora dos recursos implementados.

Direitos Autorais

Os autores são os únicos responsáveis pelo conteúdo do material impresso incluído no seu trabalho.

Agradecimentos

Os autores agradecem à *FAPEMIG (Fundação de Amparo à Pesquisa do Estado de Minas Gerais)* e ao *CNPq (Conselho Nacional de Pesquisa)* pelo apoio financeiro nas formas de fomento à pesquisa e de bolsas de iniciação científica, respectivamente.

Referências

Fonseca, F. T., 2008. Sistema computacional para análise dinâmica geometricamente não-linear através do método dos elementos finitos. Master's thesis, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.

Fonseca, F. T., Pitangueira, R. L., & Vasconcellos Filho, A., 2004. Implementação de modelos estruturais de barras como casos particulares do método de elementos finitos. In *Simpósio Mineiro de Mecânica Computacional - SIMMEC 2004*, Itajubá, Minas Gerais.

Fonseca, M. T., 2006. Aplicação orientada a objetos para análise fisicamente não-linear com modelos reticulados de seções transversais compostas. Master's thesis, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.

Fuina, J. S., 2009. *Formulações de Modelos Constitutivos de Microplanos para Contínuos Generalizados*. PhD thesis, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J., 1995. *Design Paterns - Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Gonçalves, M. A. B., 2004. Geração de malhas bidimensionais de elementos finitos baseada em mapeamentos transfinitos. Master's thesis, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.

Penna, S. S., 2007. Pós-processador para modelos bidimensionais não-lineares do método dos elementos finitos. Master's thesis, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.