## Referências Bibliográficas


[ACIS09] **www.spatial.com/products/3d/modeling/acis.html**. 1.3

[Alves06] ALVES, G. T. M.. **Projeto integrado de estabilidade de unidades flutuantes**. Master's thesis, Departamento de Engenharia Civil,PUC-Rio, 2006. (document), 1.2, 2.3, 2.2, 2.3, 2.4

[Amorim01] AMORIM, F. A. S.; MARTINS FILHO, P. D.. **Sobre a importância do estudo da metodologia de projeto no ensino de engenharia**. XXIX Congresso Brasileiro de Ensino de Engeharia-COBENGE, p. 1–10, 2001. 2.1

[Ansys09] **www.ansys.com**. 1.3, 3.1

[AutoCad09] **www.autodesk.com**. 2.1, 3.1

[Baumgart74] BAUMGART, B. G.. **Geometric Modeling for Computer Vision.** DSc thesis, CS Department, Stanford U., 1974. D

[Beckmann90] BECKMANN, N.; KRIEGEL, H.-P.; SCHNEIDER, R. ; SEEGER, B.. **The r*-tree: an efficient and robust access method for points and rectangles**. SIGMOD Rec., 19:322–331, 1990. D.2

[Beer92] BEER, G.; WATSON, J.. **Introduction to Finite and Boundary Element Methods for Engineers**. John Wiley & Sons, 1992. 1

[Birk98] BIRK, L.. **Hydrodynamic Shape Optimization of Offshore Structures**. DSc thesis, Technische Universit at Berlin, 1998. 1.3

[Birk06] BIRK, L.. **Parametric modeling and shape optimization of offshore structures**. International Journal of CAD/CAM, 6:29–40, 2006. 1.3

[Bole06] BOLE, M.; LEE, B. S.. **Integrating parametric hull generation into early stage design**. Ship Technology Research, 53, 2006. (document), 2.6, 2.7

[Coelho96] COELHO, L. C.. **MG: Manual do Usuário - Versão 3.0**. Grupo de Tecnologia em Computação Gráfica-Puc-Rio, 1996. 1.3

[Coelho98] COELHO, L. C.. **Modelagem de Cascas com Interseções Paramétricas**. DSc thesis, Departmento de Informatica, PUC-Rio, 1998. (document), 1.2, 1.3, 1.4, C.24, D, D.2, D.3

[Coelho03] COELHO, L.; JORDANI, C.; OLIVEIRA, M. ; MASETTI, I.. **Equilibrium, ballast control and free-surface effect computations using the sstab system**. International Conference of Stability of Ships and Ocean Vehicles -Stab, 8:377–388, 2003. 1.2, D

[Coons64] COONS, S. A.. **Surfaces for computer aided design**. Technical report, Cambridge, Massachusetts, 1984. 4.2

[Diaz04] DIAZ, R. G.. **Uma compressão simples de malhas irregulares com alças**. Master's thesis, Departamento de Engenharia Mecânica,PUC-Rio, 2004. D

[Evans59] EVANS, J. H.. **Basic design concepts**. Technical report, 1959. (document), 1.2, 1.6, 1.4

[Farin84] BOHM, W.; FARIN, G. ; KAHMANN, J.. **A survey of curve and surface methods in cagd. computer aided geometric design**. 1:1–60, 1984. 1.3

[FastShip09] **www.proteusengineering.com**. 2.1

[Fonseca05] FONSECA, M. M.. **Arte Naval**. Serviço de Documentação da Marinha, Rio de Janeiro-Brasil, 2005. (document), 2.5

[Hoffman89] HOFFMAN, C. M.. **Geometric & Solid Modeling: An Introduction**. Purdue University, Indiana, 1989. 1.3

[IMO78] **The international convention for the prevention of pollution from ships**. Technical report, International Maritime Organization, Protocolo, 1978. 2.5, 4.2, 5

[Ierusalimschy04] IERUSALIMSCHY, R.; FIGUREIDO, L. H. ; CELES, W.. **A linguagem lua e suas aplicações em jogos**. 1.3, 1.5, 3, 3.2, 3.3

[Lira02] LIRA, W. M.. **Modelagem Geométrica para Elementos Finitos Usando Mutlti-Regiões e Superfícies Paramêtricas**. DSc thesis, Departmento de Engenharia Civil, PUC-Rio, 2002. 1.3, 4.2

[Lop96] LOPES, H.. **Algorithm to build and unbuild 2 and 3 dimensional manifolds**. DSc thesis, Department of Mathematics, PUC-Rio, 1996. D

[Lua09] **www.lua.org**. 3.3, 3.4

[Mantyla88] MANTYLA, M.. **An Introduction to Solid Modelling Computer**. Science Press Rockville, Maryland, 1988. 1.3, 5, D

[Martins02] MARTINS FILHO, P. D.. **Projeto de engenharia: um jogo intelectual entre livre criação e ação disciplinada**. In: ANAIS DO VIII ENCONTRO DE EDUCAçãO EM ENGENHARIA, p. 1–14, 2002. 1.2, 2.1

[Mendonça04] DE MENDONÇA, C. E. L. R.. **Um Sistema Computacional para Otimização Através de Algortimos Genéticos e Redes Neurais**. DSc thesis, COPPE-UFRJ, 2004. 1.3, 1.4, 3

[Miranda99] MIRANDA, A. C. O.. **Integração de algoritmos de geração de malhas de elementos finitos**. Master's thesis, Departamento de Engenharia Civil,PUC-Rio, 1999. C.3, C.3

[Miranda00] MIRANDA, A. C. O.; MARTHA, L. F.. **Uma biblioteca computacional para geração de malhas bidimensionais e tridimensionais de elementos finitos**. Anais do XXI Ibero Latino Americano Sobre Métodos Computacionais para Engenharia, 1:1–10, 2000. 1.3, C.1, C.2, C.2.1

[Miranda02] MIRANDA, A. C. O.; MARTHA, L. F.. **Mesh generation on high-curvature surfaces based on a background quadtree structure**. In: IMR, p. 333–341, 2002. (document), C.3, C.20

[Mortenson85] MORTENSEN, M. E.. **Geometric Modeling**. John Wiley & Sons, Inc., New York, NY, USA, 1985. 1.3

[NavCad09] **www.hydrocompinc.com/navcad/default.htm**. 2.1

[Neto07] NETO, A. C.; CASTANHARO, C. A.; MORAES, C. E. M. ; LOPES, R. D. C.. **AJUSTE E SIMULAÇÃO DE CONTROLADORES DE LASTRO DE UMA PLATAFORMA SEMI-SUBMERSÍVEL DE PEQUENAS DIMENSÕES**. Monografia em automação industrial, Universidade do Estado do Rio de Janeiro - UERJ, 2007. (document), 1.2

[Oliveira04] DE OLIVEIRA, V. C. C.. **Análise da segurança em operações marítimas de exploração e produção de petróleo**. Master's thesis, Faculdade de Engenharia Mecânica e Instituto de Geociências,UNICAMP, 2004. 1.1

[Oliveira08] DE OLIVEIRA, M. C.. **Offshore platforms sizing optimization through genetic algorithms**. In: 20TH DEEP OFFSHORE TECHNOLOGY INTERNATIONAL CONFERENCE(DOT 2008), 2008. 1.3, 1.4, 3, 5

[Ousterhout94] OUSTERHOUT, J. K.. **Tcl and the Tk Toolkit**. Addison-Wesley Professional, 1994. 3.2

[Parasolid09] **http://www.plmsolutionseds.com/products/parasolid**. 1.3

[Parson03] PARSON, M. G.. **Parametric design ch. 11 in ship design and construction**. Society of Naval Architects and Marine Engineers, 2000. 2.4

[Patran09] **http://www.mscsoftware.com/**. 3.1

[Piegl91] PIEGL, L.. **On nurbs: A survey.** IEEE Computer Graphics and Applications, 10:55–71, 1991. 1.3

[Piegl99] PIEGL, L.; TILLER, W.. **The Nurbs Book**. Springer-Verlag, 1999. 1.3

[Rossignac90] ROSSIGNAC, J.; O'CONNOR, M.. **A dimensional-independent model for pointsets with internal structures and incomplete boundaries**. Geometric Modeling for Product Engineering, p. 145–180, 1990. 1.3

[Rossum03] ROSSUM, G. V.. **The Python Language Reference Manual**. Network Theory Ltd., London, 2003. 3.2

[Shah95] SHAH, J. J.; MANTYLA, M.. **Parametric and Feature Based CAD/Cam: Concepts, Techniques, and Applications**. John Wiley & Sons, Inc., New York, NY, USA, 1995. 2

[ShipWeight09] **www.shipweight.com**. 2.1

[Teixeira03] TEIXEIRA, F. G.. **Modelamento Paramétrico e Geração de Malha em Superficies para Aplicação em Engenharia**. DSc thesis, Departmento de Mecânica, UFRGS, 2003. 1.3

[Teminko97] TEMINKO, J.. **Step-by-Step QFD: Customer-Driven Product Design**. Second Edition, St. Lucie Press, Boca Raton, Florida, USA, 1997. 1.2

[Velho02] VELHO, L.; GOMEZ, J. ; FIGUEREIDO, L. H.. **Implicit objects in computer graphics**. Technical report, Springer Verlag, New York, 2002. 1.3

[Wamit08] **www.wamit.com**. 1.2, C

[Watson76] WATSON, D.; GILFILLAN, A.. **Some ship design methods**. Trans RINA, 118:279–321, 1976. (document), 1.2, 1.7

[Watson98] WATSON, D.. **Practical ship design**. Elsevier Ocean Engineering, 1, 1998. 2.4

[Zienkiewicz00] ZIENKIEWICZ, O.; TAYLOR, R.. **The Finite Element Method**. Butterworth-Heinemann, 2000. 1

# A
# Comandos de Modelagem Baseados em Histórico

Neste apêndice são detalhados os comandos de modelagem implementados no MG. Cada comando possui o número de parâmetros necessários para ser executado. Caso seja fornecido para determinado comando mais parâmetros que o necessário, uma mensagem de erro será mostrado para o usuário.

## A.1
## Comandos de geração de vértices e curvas

– mgNewEntityType(*etype*)

*etype* é um código que pode ser do tipo vértice ou curve.

Após a execução deste comando, todas as entidades que serão criadas a seguir serão do tipo especificado por *etype*. Caso seja necessário criar entidades diferentes deve-se executar novamente este comando especificando o novo tipo de entidade a ser criada.

– mgCreateVertex(X,Y,Z)

Este comando gera um vértice no espaço 3D na posição indicada pelas coordenadas X,Y e Z.

– mgSetCurveType(*curvecode*)

*curvecode* indica o tipo de curva que será criada: *line*, *polyline*, *arc* e *spline*.

Após executado este comando, todos os pontos adicionados a seguir serão os pontos que gerarão a curva especificada.

– mgAddPoint(X,Y,Z)

Este comando permite adicionar pontos a uma curva qualquer. Caso a curva selecionada seja do tipo linha poligonal ou spline, poder-se-á adicionar n pontos a curva. Caso o tipo de curva selecionada seja um arco somente serão aceitos três pontos (centro, raio e tamanho da curva). Os demais pontos adicionados a uma curva do tipo arco serão ignorados.

– mgEndVertex()

Este comando finaliza a criação de vértices. Caso deseja-se criar novas entidades devera-se selecionar primeiramente o tipo de entidade a ser criada, utilizando o comando mgNewEntityType().

– mgEndEditCurve()

Este comando finaliza a criação de curvas, sejam estes do tipo: polilinha, linha, arco ou spline.

## A.2
## Comandos de seleção e edição

Descreve-se a seguir os comandos utilizados para selecionar e interagir com as entidades do MG.

– mgSelect(<"etypeid">/<all>)

Este comando permite selecionar uma ou todas as entidades criadas. Caso deseja-se selecionar uma entidade curva basta passar como parâmetro entre aspas a letra C em maiúscula seguida do id da curva a selecionar:

Ex: mgSelect("C0"), mgSelect("S0"), ... , mgSelect("T0"), mgSelect("all").

Os exemplos acima, mostram a seleção das entidades curva, superfície e volume cujo id é igual a 0. Caso deseja-se selecionar um vértice basta passar como parâmetro "V<id>"seguido do id do vértice. O parâmetro *all* é muito útil quando se deseja criar um volume, pois é necessário selecionar todas as entidades pertencentes a este volume.

– mgUnSelect(<"etypeid">/<"all">)

Este comando permite remover a seleção de uma entidade previamente selecionada. Todos os parâmetros válidos para o comando *mgSelect*, também são válidos para este comando.

– mgInvertSelection()

Este comando inverte a seleção das entidades previamente selecionadas, e não recebe nenhum parâmetro para ser executado.

– mgDeleteSelected()

Este comando permite apagar todas as entidades previamente selecionadas. Igual que o comando anterior, não recebe nenhum parâmetro.

– mgSetCurveSubdivision(*nsdv*,*ratio*)

Este comando permite definir a subdivisão da poligonal que a representa em n partes. Os parâmetros significam:

*nsdv* - número de subdivisões da curva.

*ratio* - razão entre o tamanho das subdivisões.

O parâmetro *ratio* permite gerar uma subdivisão gradativa na curva dependendo do valor que for passado entre 0 e 1.

– mgInvCurveSubdivision()

Este comando permite inverter a orientação e sentido das subdivisões de uma curva respectivamente.

– mgJoinCurve()

Este comando permite que duas curvas ou mais possam ser unidas em uma única curva. Caso uma curva for subdivida em n partes, pode-se reconstruir a curva inicial fazendo a junção das n subdivisões (curvas), que geraria a curva inicial antes de ser subdividida.

– mgSplitCurve()

Este comando permite dividir uma curva seja este do tipo quaisquer em um número n de partes iguais, utilizando para isto a discretização da poligonal interna desta curva.

– mgTurn2Poly()

Este comando permite transformar uma linha do tipo *arc* ou *spline* em um conjunto de linhas poligonais dependendo do número de subdivisões que esta curva possua. Isto é muito útil dependo do modelo a ser criado (Ex: navios por seções transversais)

## A.3
## Comandos de geração e manipulação de superfícies e volumes

A seguir são descritos os comandos para gerar e manipular superfícies e volumes.

– mgSetSurfaceType(*surfcode*)

*surfcode* pode ser do tipo: *BILINEAR*, *TRILINEAR*, *PLANAR*, *TRANSLATIONALSWEEP*, *ROTATIONALSWEEP* e *GENERICSWEEP*.

Este comando permite definir o tipo de superfície que será criada, que é definida pelo parâmetro *surfcode*.

– mgCreateSurface()

Este comando permite criar uma superfície, a partir do conjunto de curvas previamente selecionadas, e o tipo de superfície previamente definida pelo comando *mgSetSurfaceType*.

– mgCreateVolume()

Este comando permite criar um volume segundo as curvas e superfícies selecionadas.

– mgInvertSurfOrientation()

Este comando inverte a orientação de todas as superfícies selecionadas.

– mgCutSurfToPlane()

Este comando permite calcular a curva de *trimming* de um ou várias superfícies que se interceptam com o plano de interface.

## A.4
## Comandos geração de wizards

São descritos a continuação os comandos para gerar wizards ou entidades automáticas como cone, cilindro, esfera entre outros, os quais são definidos de acordo aos seus parâmetros correspondentes.

– mgCreateBox(length,width,height,cx,cy,cz)

Onde:

**length** comprimento da caixa.

**width** largura da caixa.

**height** altura da caixa.

**cx, cy, cz** coordenadas do centro da caixa.

– mgCreateCylinder(radius,height,cx,cy,cz)

Onde:

**radius** raio do cilindro.

**height** altura do cilindro.

**cx, cy, cz** coordenadas do centro do cilindro.

– mgCreateCone(radius,height,cx,cy,cz)

Onde:

**radius** raio do cone.

**height** altura do cone.

**cx, cy, cz** coordenadas do centro do cone.

– mgCreateSphere(radius,cx,cy,cz)

Onde:

**radius** raio da esfera.

**cx, cy, cz** coordenadas do centro da esfera.

– mgCreateRectangle(width,height,cx,cy,cz)

Onde:

**width** comprimento no eixo x do retângulo.

**height** largura no eixo y do retângulo.

**cx, cy, cz** coordenadas do centro do retângulo.

– mgCreateTriangle(width,height,cx,cy,cz)

Onde:

**width** comprimento no eixo x do triângulo.

**height** largura no eixo y do triângulo.

**cx, cy, cz** coordenadas do centro do triângulo.



## A.5
## Comandos de transformações afins

Neste tópico serão especificados os comandos do MG utilizadas para realizar transformações (translação, copia, rotação, etc) em entidades previamente selecionadas.

– mgEnableCopies()

Este comando permite ativar a copia de todas as entidades selecionadas. Isto é, quando uma transformação é executada (translação, rotação, etc.), será criada uma copia de todas as entidades selecionadas.

– mgTranslateX(), mgTranslateY(), mgTranslateZ() e mgTranslateXYZ().

Estes comandos realizam translações das entidades previamente selecionadas nos eixos X, Y, Z, XYZ respectivamente. De acordo ao comando utilizado, e o eixo de transformação, será necessário fornecer um ou no máximo três parâmetros de entrada referentes aos eixos X,Y ou Z respectivamente.

– mgRotateX(), mgRotateY(), mgRotateZ() e mgRotateXYZ().

Estes comandos realizam rotações das entidades previamente selecionadas nos eixos X, Y, Z, XYZ respectivamente. De acordo ao comando utilizado será necessário fornecer um ou no máximo três parâmetros de entrada referentes aos eixos X,Y ou Z respectivamente.

– mgScaleXYZ()

Este comando realiza escala nos eixos XYZ das entidades selecionadas. Este comando requer três parâmetros de entrada nos seus respectivos eixos X, Y e Z.

– mgRepeatTransformation()

Este comando repete a ultima transformação que foi executada e as aplica nas entidades previamente selecionadas. Pode ser aplicado n vezes repetindo assim n vezes a ultima transformação executada e este comando não precisa de parâmetros de entrada para sua execução.

– mgMirrorToPlane()

Este comando realiza o espelhamento das entidades selecionadas em relação ao plano de desenho. Caso o comando *mgEnableCopies*() estiver ativado será feito o espelhamento, e uma copia de todas as entidades selecionadas. Caso contrario todas as entidades selecionadas serão espelhadas (transladas) para sua nova posição em relação ao plano de desenho. Não é necessário fornecer parâmetros de entrada.

– mgFixModel()

Este comando elimina entidades redundantes ou sobrepostas num determinado modelo.

## A.6 Comandos de entrada e saída de dados

A seguir apresenta-se os comandos de entrada e saída utilizados para carregar um script (arquivo Lua), carregar um arquivo .mg, imprimir valores de variáveis, medir o tempo de execução de determinado script ou função.

– mgLoadMGFile(<filename>)

Este comando carrega um arquivo do modelador MG via linha de comandos. Precisa como parâmetro de entrada o nome do arquivo a ser carregado

– mgLoadLuaFile(<filename>)

Este comando carrega um script Lua via linha de comandos e gera o modelo 3D a partir de um conjunto de comandos. Precisa como parâmetro de entrada o nome do arquivo a ser carregado

– mgPrint()

Este comando permite visualizar os possíveis valores que uma variável, tabela, entidade tem assim como também escrever uma mensagem de texto os quais serão mostrados na interface de comandos Lua.

## A.7
## Comandos de transformações do plano de interface

Nesta seção são citados os comandos que mexem com transformações diretas no plano de interface.

– mgTranslatePlane(x,y,z)

Este comando translada a posição do plano para a nova posição indicada pelos parâmetros de entrada definidos por x, y e z.

– mgRotatePlane(alfa,x,y,z)

Este comando rotaciona a posição do plano para a nova posição indicada pelo ângulo de rotação definido por alfa e pelo eixo que define a direção da rotação definidos por x, y e z.

– mgPutPlaneAtXY()

Este comando posiciona o plano de interface no plano XY com z=0.

– mgPutPlaneAtXZ()

Este comando posiciona o plano de interface no plano XZ com y=0.

– mgPutPlaneAtYZ()

Este comando posiciona o plano de interface no plano YZ com x=0.

# B
# Comandos de Modelagem Baseados em Script

Neste apêndice são detalhados os comandos de modelagem implementados no MG para cada uma das entidades do tipo vértice, curva, superfície e volume. Cada comando possui o número de parâmetros necessários para ser executado. Caso seja fornecido para um comando mais parâmetros que o necessário, uma mensagem de erro será mostrado para o usuário.

## B.1
## Comando para gerar vértices

– mgVertex()

Este comando pode receber parâmetros ou não. Caso não seja passado nenhum parâmetro, o vértice gerado sempre estará na origem das coordenadas do MG. Caso contrario deve ser passado 3 parâmetros numéricos nas coordenadas X, Y e Z.

Este comando possui os seguinte métodos:

**Cross** Cria um vértice cujas coordenas é o resultado do produto vetorial entre dois vértices.

**Print** Imprime os dados do vértice.

**Select** Seleciona o vértice corrente.

**UnSelect** Retira a seleção do vértice corrente.

**TranslateX** Translada o vértice no eixo X.

**TranslateY** Translada o vértice no eixo Y.

**TranslateZ** Translada o vértice no eixo Z.

**TranslateXYZ** Translada o vértice no eixos XYZ respectivamente.

**RotateX** Rotaciona o vértice no eixo X.

**RotateY** Rotaciona o vértice no eixo Y.

**RotateZ** Rotaciona o vértice no eixo Z.

São implementados para esta classe os seguintes meta-métodos que a linguagem Lua permite:

- __eq
- __add
- __sub
- __unm
- __mul
- __pow
- __index
- __newindex

## B.2
## Comando para gerar curvas

- mgCurve()

Este comando cria curvas do tipo: linha, polilinha, arc e spline. Para criar uma curva é necessário passar como parâmetro o tipo de linha a ser criada e os ids dos vértices ou os objetos do tipo vértice.

**Print** Imprime os dados da curva.

**Select** Seleciona a curva corrente.

**UnSelect** Retira a seleção da curva corrente.

**TranslateX** Translada a curva no eixo X.

**TranslateY** Translada a curva no eixo Y.

**TranslateZ** Translada a curva no eixo Z.

**TranslateXYZ** Translada a curva no eixos XYZ respectivamente.

**RotateX** Rotaciona a curva no eixo X.

**RotateY** Rotaciona a curva no eixo Y.

**RotateZ** Rotaciona a curva no eixo Z.

**ScaleXYZ** Escala a curva nos eixos XYZ respectivamente.

**SetNormal** Seta a orientação das curvas do tipo arco.

**SetSubdivision** Subdivide uma curva em n segmentos.

São implementados para esta classe os seguintes meta-métodos:

- __index
- __newindex

## B.3
## Comando para gerar superfície

- mgSurface("surftype","meshtype",ncurves,curveref1,..,curverefn)

Este comando cria superfícies, onde é necessário passar como parâmetros:

***surftype*** Tipo de superfície a ser gerada : bilinear, trilinear, planar, etc.

***meshtype*** Tipo de malhar a ser gerada: quadrilat, contraction, etc.

***ncurves*** Número de curvas que compõem a superfície

***curveref*** Conjunto de curvas que formam a superfície. Pode ser fornecido o id da curva ou um objeto do tipo *mgCurve*.

Os métodos implementados para este comando são:

**Print** Imprime os dados da superfície.

**Select** Seleciona a superfície corrente.

**UnSelect** Retira a seleção da v corrente.

**TranslateX** Translada a superfície no eixo X.

**TranslateY** Translada a superfície no eixo Y.

**TranslateZ** Translada a superfície no eixo Z.

**TranslateXYZ** Translada a superfície no eixos XYZ respectivamente.

**RotateX** Rotaciona a superfície no eixo X.

**RotateY** Rotaciona a superfície no eixo Y.

**RotateZ** Rotaciona a superfície no eixo Z.

**ScaleXYZ** Escala a superfície nos eixos XYZ respectivamente.

São implementados os mesmos meta-métodos da classe curva.

## B.4
## Comando para gerar volume

– mgTank("name","voltype",nsurfaces,surfref0,..,surfrefn)

Este comando cria volumes a partir de um conjunto de superfícies e os seus parâmetros são:

***name*** Nome do volume a ser criado.

***voltype*** O modelador geométrico utilizado neste trabalho define os seguintes tipos de volumes: *Hull_Volumes*, *Ballast_Tanks*, *Fuel_Oil*, *Fresh_Water*, *Pump_Room*, *Void_Spaces*, *Chain_Lockers*, *Acces_Trunk*, *Columns*, *Mass Compartment*, *Hull & Mass Compartment*.

***nsurfaces*** Número de superfícies que compõem o volume.

***surfref*** Pode ser fornecido os ids de cada superfície ou um objeto do tipo *mgSurface*.

São implementados os mesmos métodos e meta-métodos da classe *mgSurface*.

# C
# Geração de Malhas

Nesta seção, são apresentados os ambientes gráficos, para a geração de malhas, implementados nos programas MG e Sstab. Estes novos ambientes permitem subdividir, cortar num determinado calado e gerar uma nova malha, de forma gradativa, de acordo com os parâmetros de entrada do usuário. A gradação de malhas consiste em dividir um determinado domínio (faces ou superfícies) em sub-domínios, gerando novas malhas para a análise de estabilidade dinâmica, porém preservando as malhas originais.

As malhas geradas neste trabalho são utilizadas pelo programa Wamit (Wave MIT) [Wamit08], para a análise dinâmica da influência das ondas do mar na plataforma. Conseqüentemente, o processo de geração de malhas, influência bastante na eficiência dos resultados finais: uma discretização mal feita pode comprometer todo o resultado final, gerando uma resposta com erros significativos.

O Wamit[Wamit08] é um programa de análise de radiação/difração por painéis, que faz a análise linear da interação do casco de estruturas *offshore* com as ondas, no domínio da freqüência. O Wamit exige que todas as faces sejam convexas e tenham sempre 4 vértices (quad ou triângulo com um vértice repetido).

O processo de gradação de malha é feito de tal forma que haja um número grande de painéis próximo a linha da água e vá gradativamente reduzindo até a quilha do modelo.

## C.1
## Geração de malhas no MG

A figura C.1 mostra o ambiente gráfico implementado no MG, para cortar e gerar malhas para o Wamit. É possível ver o modelo original e as malhas geradas pelo processo de corte.

A subdivisão das curvas cortadas (gradação) preserva a integridade do modelo. Isto é, as curvas que não tenham uma curvatura definida, linhas e polilinhas, poderão ser subdivididas de acordo com os parâmetros fornecidos pelo usuário. As curvas cuja curvatura forem altas, spline e arcos, serão subdivididas até um certo limite máximo de tolerância, para poder preservar integralmente a geometria do modelo original com o modelo cortado.

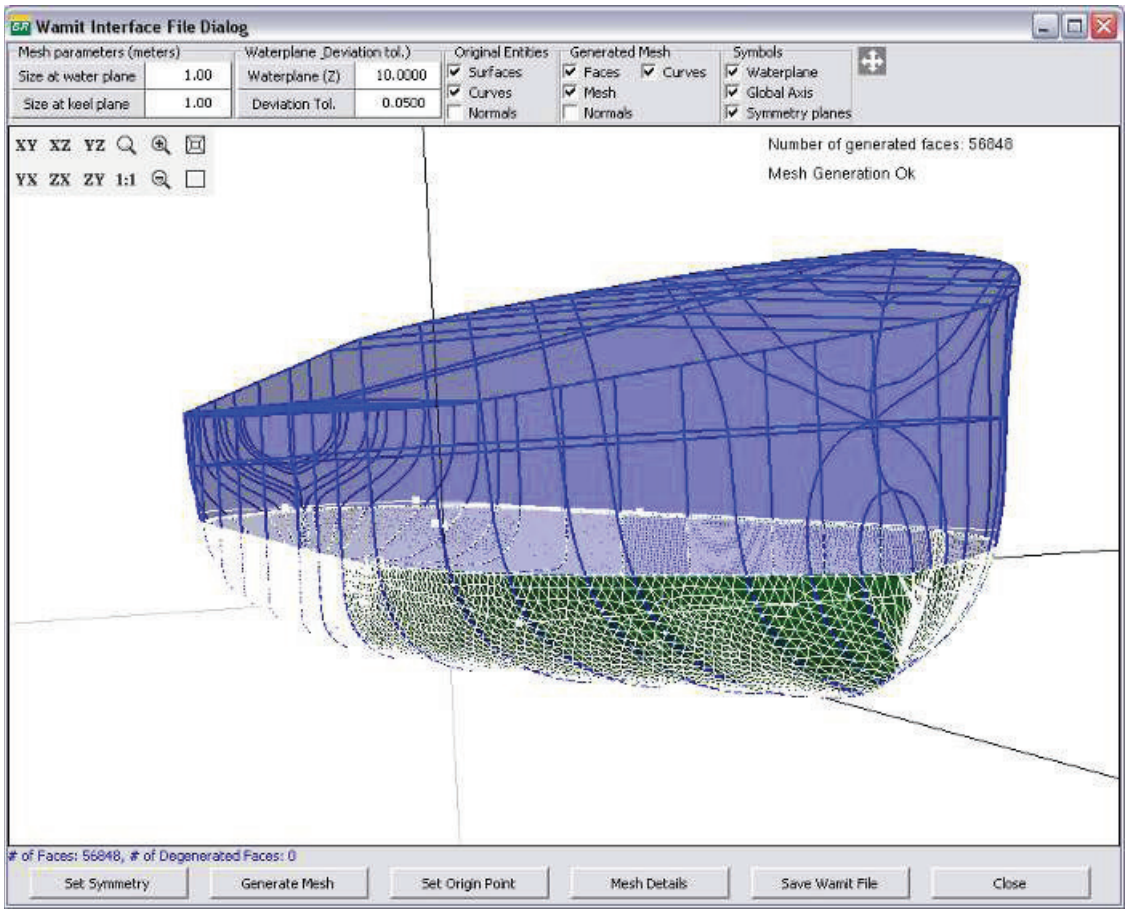



Figura C.1: Interface do Gerador de malhas para o Wamit no MG.

A seguir, descrevem-se os passos necessários para processar todas as curvas e superfícies. Este processo consta de 3 fases:

- Corte das superfícies e geração das curvas de *trimming*;
- Subdivisão gradativa das curvas e
- Geração das malhas respeitando o tamanho médio de painel.

**Corte das superfícies e geração das curvas de *trimming*** A geração de malhas para o Wamit é feita em todas as superfícies e curvas que estão abaixo do plano da água. Se uma superfície se intercepta com o plano de água, corta-se esta superfície, assim como as suas curvas de bordo. As curvas de *trimming* e as curvas cortadas definem uma nova malha que está totalmente abaixo do plano de corte (figura C.1).

Para gerar as curvas de *trimming*, de cada superfície com o plano de corte, percorrem-se todas as faces da malha. Isto é, para cada face, encontra-se uma curva de corte com o plano, e no final unem-se todas as curvas cortadas.

Eventualmente, as curvas de *trimming* de uma superfície podem gerar mais de um *loop*, onde cada *loop* define uma superfície cortada. A figura C.2 mostra uma superfície bilinear, cujo corte gerou dois *loops*, resultando em duas novas superfícies. O algoritmo de corte deve ser capaz de identificar estes casos e criar os *loops* corretos que definam cada superfície cortada.

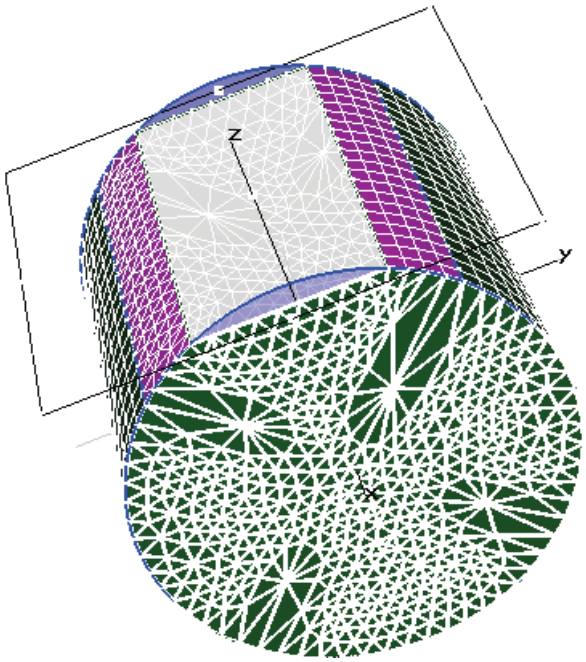



Figura C.2: *Loops* de curvas cortadas numa superfície.

**Subdivisão gradativa das curvas** Uma vez calculadas todas as curvas de corte, o próximo passo é gerar uma subdivisão gradativa definida pelo usuário.

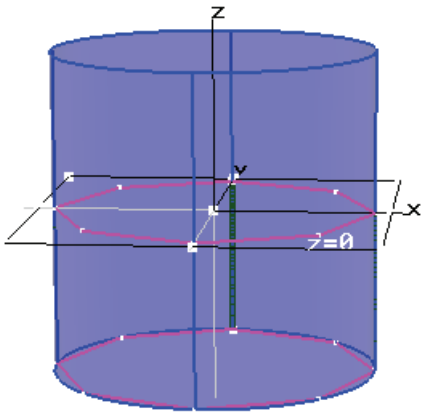



Figura C.3: Problema na geração das subdivisões das curvas cortadas.

Os parâmetros que controlam a subdivisão das curvas são: o tamanho médio de painel no plano da água (*Size at water plane*), tamanho médio de painel no plano da quilha (*Size at keel plane*) e uma tolerância de desvio de curvatura (*Deviation tolerance*). Estes parâmetros podem ser vistos na figura C.1.

A figura C.3 mostra um cilindro cortado na cota Z=0, subdividido com tamanho médio de painel de 6 metros e tolerância de desvio igual a 1. Pode-se perceber que a subdivisão das curvas não respeita o modelo original. Já a figura C.4 mostra o mesmo cilindro, com um critério da curvatura igual a 0.05, o que privilegia a geometria da curva original, realizando subdivisões automáticas quando necessário.

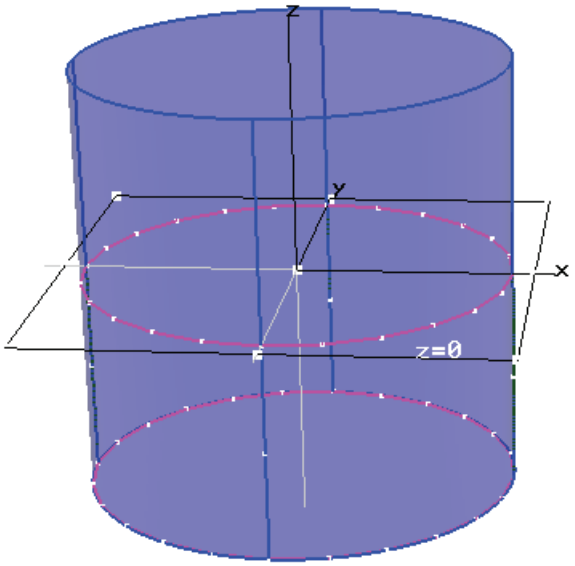



Figura C.4: Geração das subdivisões das curvas respeitando o critério de curvatura.

A tolerância de desvio de curvatura, da subdivisão de uma curva, é um parâmetro muito importante a ser considerado. Este parâmetro é definido da seguinte forma:

$$devtoler = length(C) - lengthgrad(C) \qquad \text{(C-1)}$$

Onde *length* é o comprimento da curva original $C$ e *lengthgrad* é o comprimento da nova subdivisão da curva $C$. Desta forma, enquanto a tolerância de desvio da curva não for menor que uma dada tolerância, subdivide-se esta a curva de forma automática aplicando-se a equação C-1 até alcançar a tolerância definida.

### Geração das malhas respeitando o tamanho médio de painel

Definido o processo de corte e subdivisão das curvas, procede-se à geração de todas as malhas das superfícies que foram cortadas. Utiliza-se, neste ambiente, o gerador de malhas implementado por [Miranda00]. Um resultado deste gerador de malhas pode ser apreciado na figura C.1.

O algoritmo de Miranda [Miranda00] é utilizado para gerar malhas em contornos bilineares, trilineares ou planares, onde a subdivisão das curvas de bordo de uma superfície pode ser definida pelo usuário. Dependendo da situação, uma malha triangular não estruturada será gerada ou um método transfinito será aplicado.

## C.2
## Geração de malhas no Sstab

O método de geração de malhas para o Wamit, implementado no Sstab, é muito parecido com o método implementado no MG. A abordagem, neste caso, é um pouco diferente, pois não existe o conceito de retalhos de superfícies, uma vez que o Sstab armazena diretamente as malhas extraídas de cada retalho ou patch modelado pelo MG.

A figura C.5 apresenta a interface do gerador de malhas implementado, mostrando um modelo de uma plataforma já cortado pelo plano da linha da água.

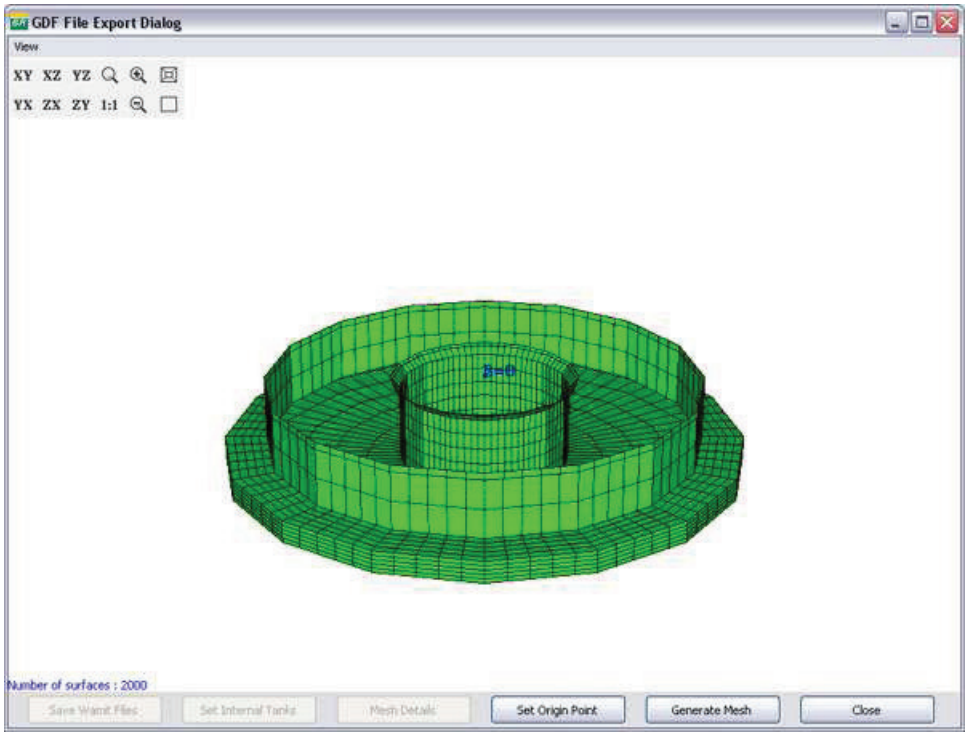



Figura C.5: Interface do Gerador de Malhas para o Wamit no Sstab.

Como não existem retalhos no Sstab, todas as faces são convexas (quads ou triângulos), além disto a subdivisão das curvas aqui se dá por cada face (aresta) da malha e não pela definição geométrica das curvas que definiram cada retalho na fase de modelagem com o MG. O critério de desvio de curvatura utilizado no módulo de geração de malhas para o Wamit, implementado no MG, não é mais necessário. Além disto, se for aplicado o algoritmo de simplificação de faces, o Sstab passa a armazenar também um conjunto de superfaces (face com n arestas) o qual agrupa várias faces que compartilham uma mesma propriedade de coplanaridade, e estas podem conter um ou vários contornos (*loops*) internos.

Pode-se dizer que os passos mais importantes para gerar as malhas do Wamit no Sstab são:

- Corte de todas as faces do modelo;
- Subdivisão gradativa de todas as arestas de cada face;
- Tratamento de faces simplificadas
- Geração de Malhas.

**Corte de todas as faces do modelo** Quando um modelo é carregado no Sstab, o programa automaticamente classifica todas as faces de acordo com a posição em que se encontra com o plano da linha da água. Isto é, internamente o Sstab classifica e armazena vetores de faces que estão cortadas abaixo do plano (*cut below*) ou cortadas acima do plano (*cut above*) e as faces que estão sobre o plano são classificadas como (*cut on*). Esta classificação se realiza da seguinte forma:

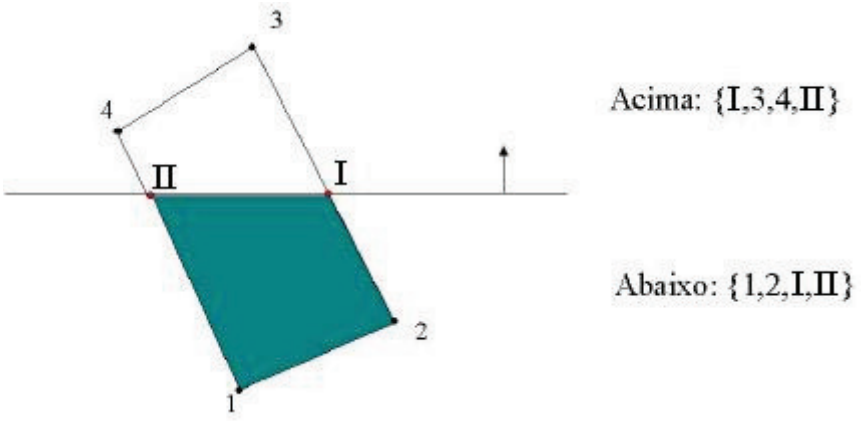



Figura C.6: Classificação de faces no Sstab.

Seguindo a seqüência mostrada na figura C.6 para todas as arestas da face, classificam-se os seus vértices de início e fim, os quais podem ser: *below-below*, *below-above*, *above-above*, e *above-below*. Para as arestas que

são classificados *below-above* e *above-below* são calculados os vértices de corte I e II (figura C.6), criando uma nova aresta I-II, que por sua vez cria as faces classificadas como acima e abaixo, mostradas na figura C.6. Desta forma, o módulo de geração de malhas para o Wamit considerará somente as faces classificadas como abaixo do plano da linha da água.

**Subdivisão gradativa de todas as arestas de cada face** A subdivisão gradativa de todas as arestas, que pertencem às faces que estão abaixo da linha do plano da água, é feita da mesma forma que no módulo de geração de malhas para o Wamit, implementado no MG. Isto é, o usuário indica um valor de tamanho mínimo e máximo de painéis entre o plano da linha da água e o plano da quilha, e o programa interpola pontos internos de subdivisão para cada aresta, quando possível.

**Tratamento de faces simplificadas** O módulo de simplificação de malhas do Sstab elimina faces completamente coplanares entre si, gerando também superfaces, da mesma forma como é feito no MG.

As superfaces resultantes da simplificação que contenham *loops* devem sempre manter uma orientação consistente do *loop* externo com relação aos *loops* internos e viceversa. Para isto, são inseridas duas arestas de conexão por *loop* interno a partir de um determinado vértice do *loop* externo, conforme mostrado na figura C.7.

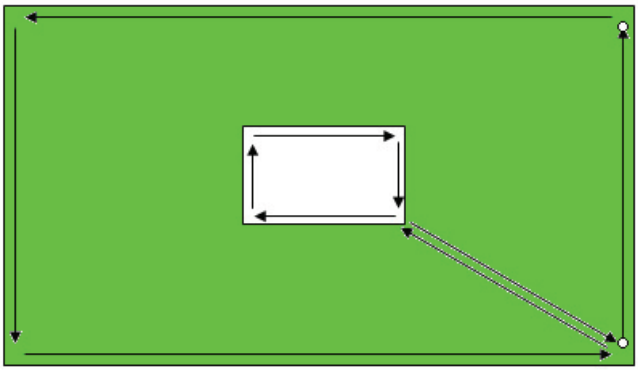


Figura C.7: Orientação da fronteira de uma superface com furo no Sstab.

**Geração de Malhas** Uma vez eliminadas as arestas que conectam *loops* externos aos *loops* internos de uma superface, podem-se gerar as malhas tanto das faces convexas quanto das faces simplificadas utilizando o mesmo algoritmo de geração de malhas desenvolvido por Miranda [Miranda00].

### C.2.1
### Resultados de Geração de Malhas

Nesta seção apresentam-se alguns resultados de corte e geração de malhas no MG.

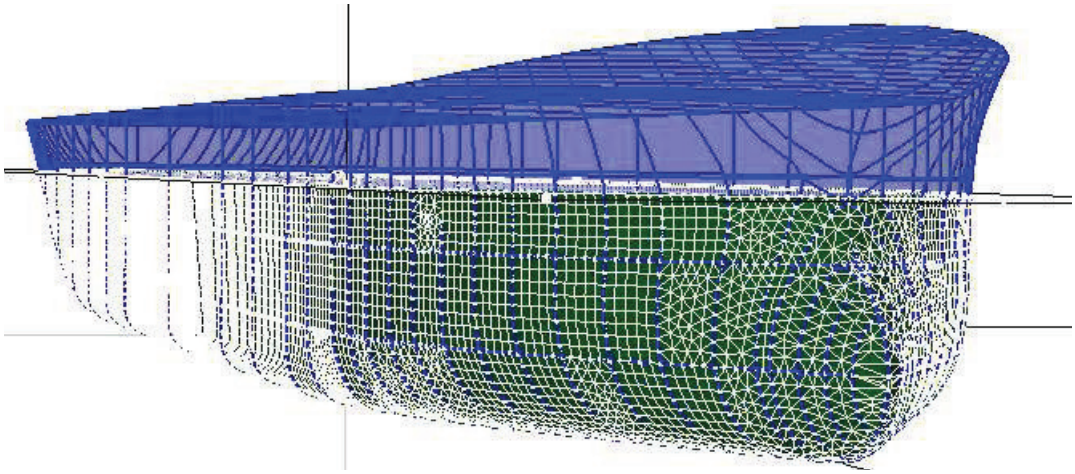


Figura C.8: Corte e geração de malhas num navio.

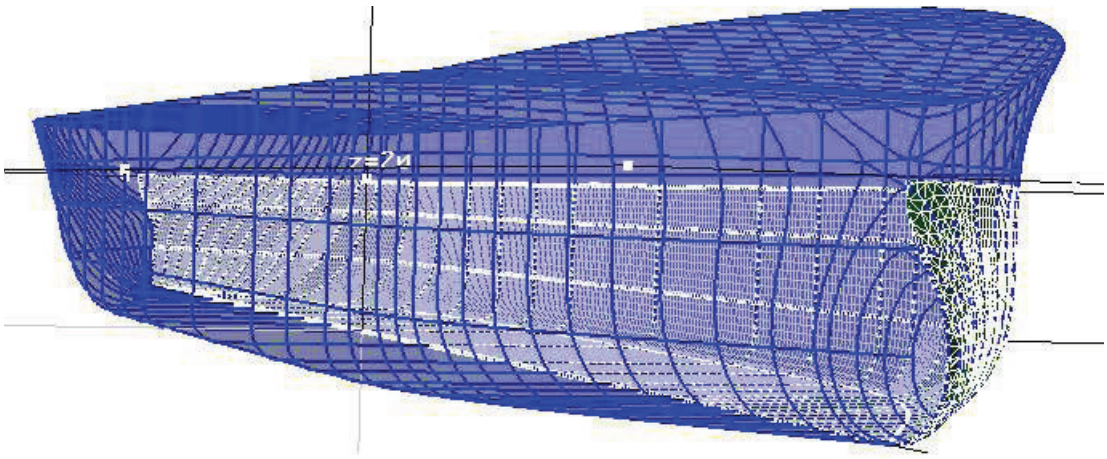



Figura C.9: Corte e geração de malhas em um dos planos de simetria do modelo.

As figuras C.8 e C.9 mostram o corte e a geração das malhas cortadas de um navio no calado igual a 20. Os tamanhos de painel escolhidos foram de 1 para 1 tanto no valor de *size at waterplane* como *size at keelplane*. O critério de tolerância de curvatura é 0.0500. O número de painéis gerados na figura C.8 foi de 39890 enquanto que a figura C.9 gerou 19952 painéis.

A mudança destes parâmetros gera diferentes instâncias de malhas. Os exemplos mostrados, a seguir, consideram somente a parte simétrica do navio.

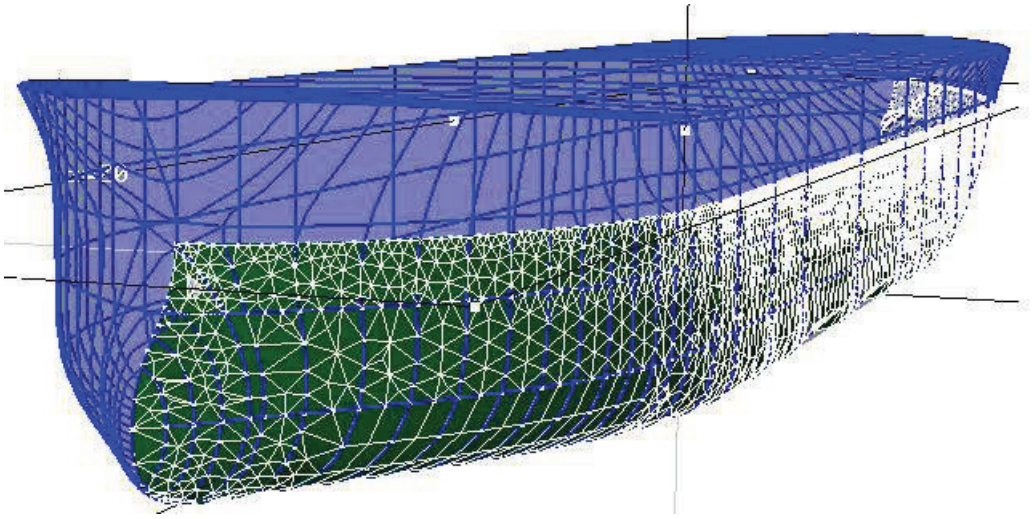


Figura C.10: Gradação de malha de 1 para 4 da parte simétrica do navio.

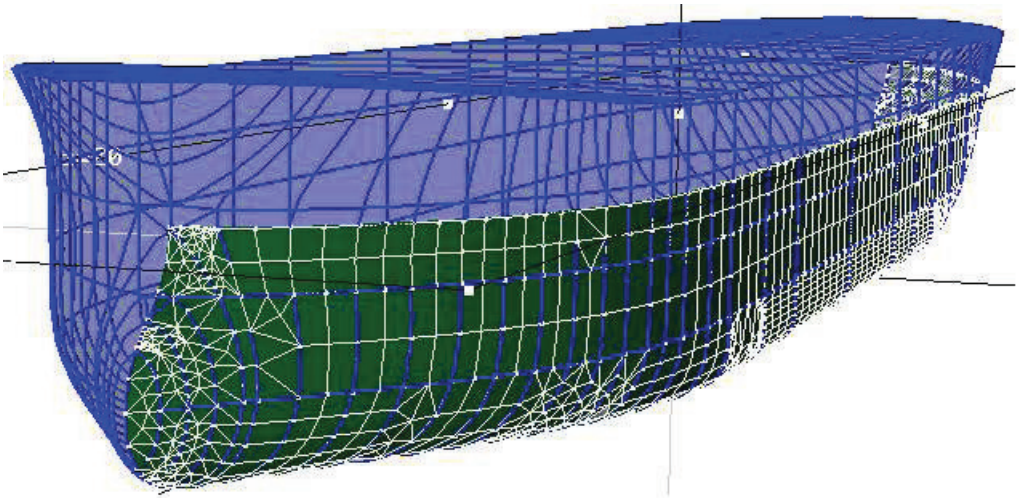


Figura C.11: Gradação de malha de 4 para 4 da parte simétrica do navio.

As figuras C.10 e C.11 mostram o resultado da variação dos parâmetros do ambiente de corte de malhas do MG. A malha gerada na figura C.10 possui um total de 6294 painéis gerados, considerando uma gradação de 1 para 4 desde o plano da linha da água até a quilha do navio. Já a malha gerada na figura C.11 possui somente 2437 painéis, com uma gradação de 4 para 4. É necessário mencionar que a tolerância de desvio de curvatura foi mantida, assim como a altura do plano de corte igual a 20. A mudança da tolerância de desvio também influência a qualidade da malha gerada, como será mostrado na seguinte figura.

A figura C.12 mostra a gradação das malhas cortadas com tamanho de painel de 1 para 7. Nota-se que a geometria do modelo original não foi mantida, (figura C.8), pois o raio de boio do navio foi completamente eliminado, utilizando uma tolerância de desvio igual a 1. O número de painéis gerados nesta instância foi de 3020.

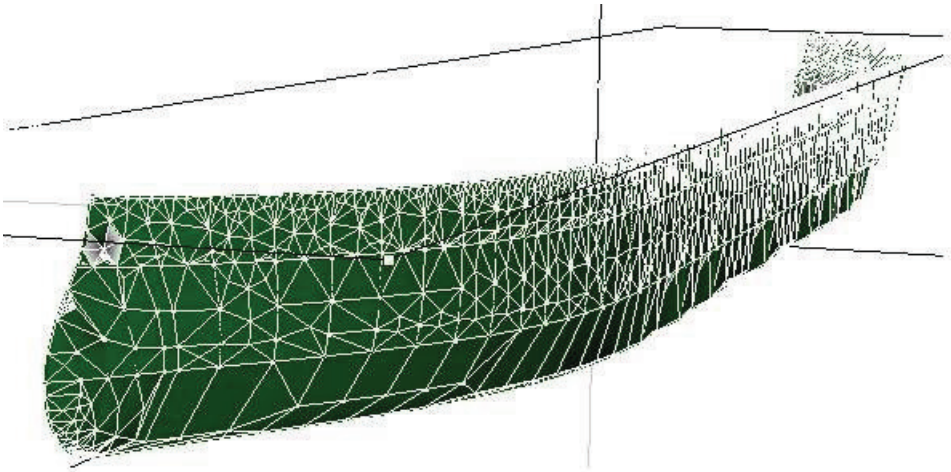


Figura C.12: Gradação de malha de 1 para 7 da parte simétrica do navio.

A seguir mostra-se o processo de corte e geração de malhas em plataformas.

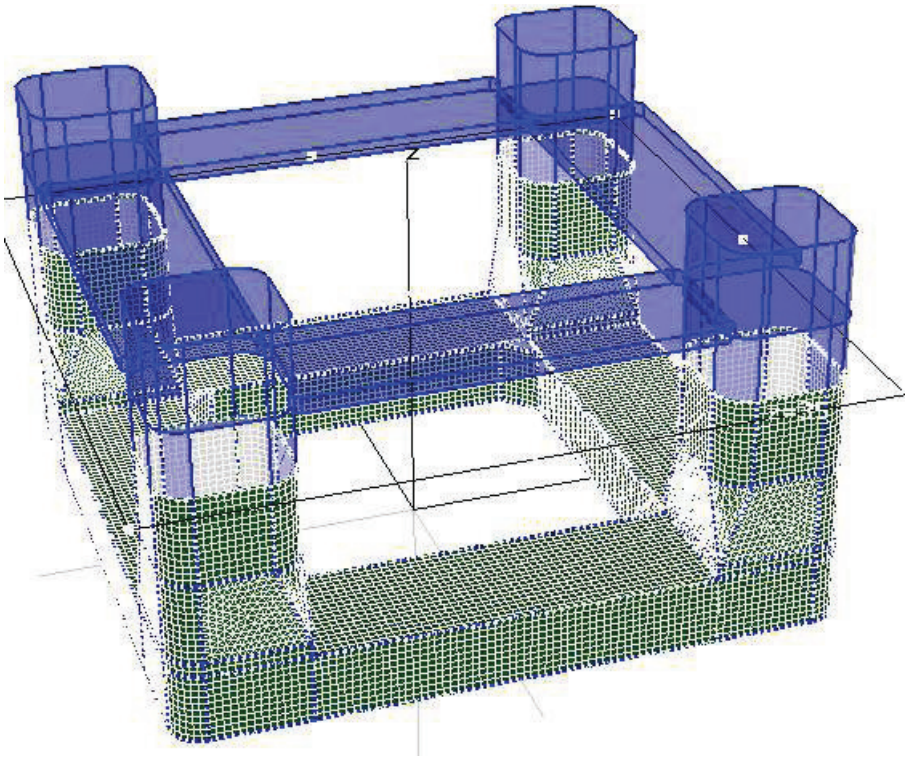


Figura C.13: Plataforma cortada com gradação de 1 para 1.

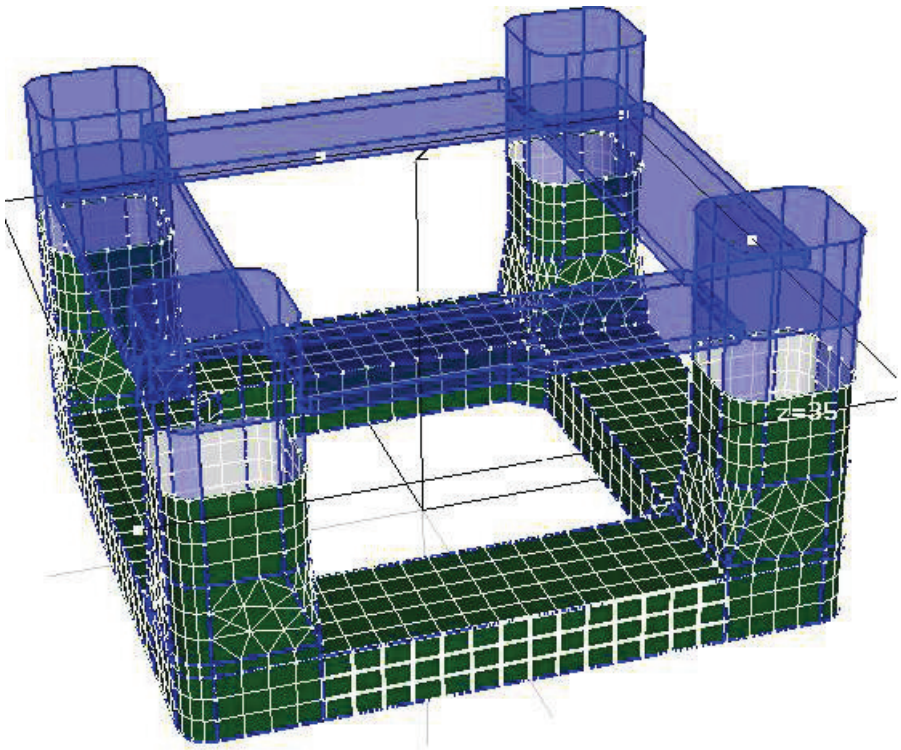



Figura C.14: Plataforma cortada com gradação de 4 para 4.

A figura C.13 mostra o corte e geração de malhas com tamanho de painéis de 1 para 1 e calado igual a 35. O número de painéis gerados é 46136. Já a figura C.14 mostra a mesma plataforma, porém com uma gradação de malhas de 4 para 4, obtendo um total de painéis igual a 3900. A tolerância de desvio de curvatura em ambos casos é igual 0.0500.

O uso do ambiente de corte e geração de malhas no MG é muito simples. O usuário simplesmente escolhe a cota Z, onde deseja posicionar o plano de corte, e, automaticamente, as curvas são cortadas e a malha do modelo cortado é gerado. É possível, ainda, selecionar um conjunto de curvas e mudar, localmente, os parâmetros de subdivisão de cada curva, assim como escolher um conjunto de superfícies e poder modificar o tipo de malha a ser gerada.

Estas opções, proveêm um controle bastante grande da malha a ser gerada, para as simulações de análise dinâmica, feitas com o programa Wamit. Nos exemplos apresentados, as melhores malhas para poder realizar simulações no Wamit são as das figuras C.11 e C.14, as quais representam muito bem a geometria do modelo com uma quantidade de painéis razoável.

Nesta seção apresentam-se alguns resultados de corte e geração de malhas obtidos no Sstab.

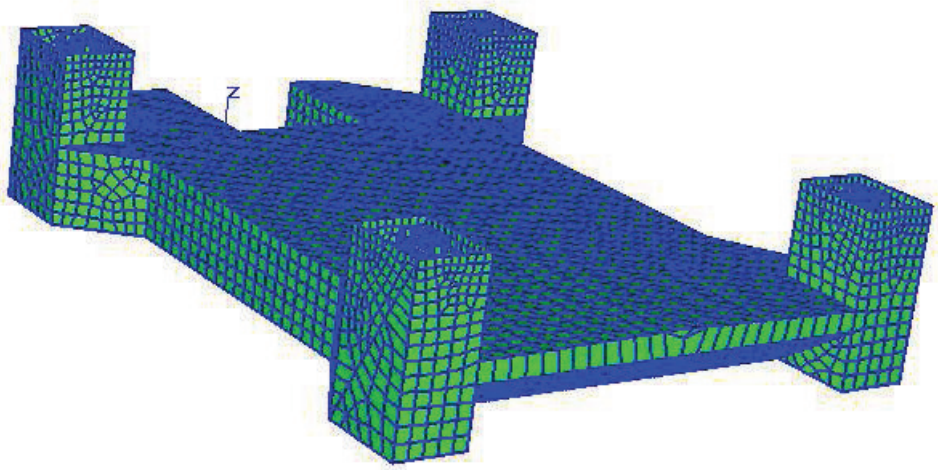



Figura C.15: Malha gerada no Sstab do corte do modelo simplificado mostrado na figura D.12.

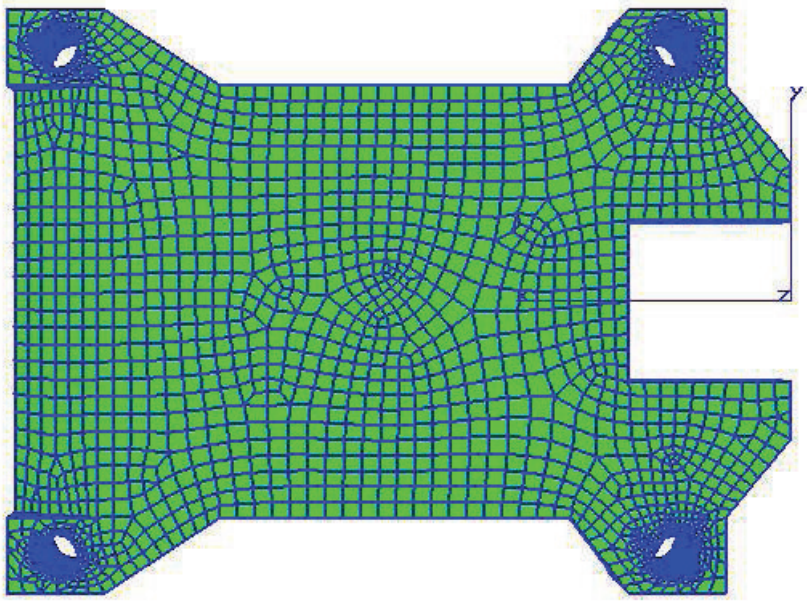



Figura C.16: Malha gerada no Sstab do corte do modelo simplificado mostrado na figura D.12, vista do plano XY.

As figuras C.15 e C.16, mostram o resultado da geração de painéis de uma malha simplificada. As malhas geradas possuem uma tamanho médio de painel de 0.5 no plano da linha da água e 1 na quilha. O número total de painéis gerados foi de 8072. Pode-se perceber que módulo de geração de malhas para

o Wamit desconsidera as arestas "falsas" introduzidas pela aplicação do algoritmo de simplificação em superfaces que possuem *loops* internos.

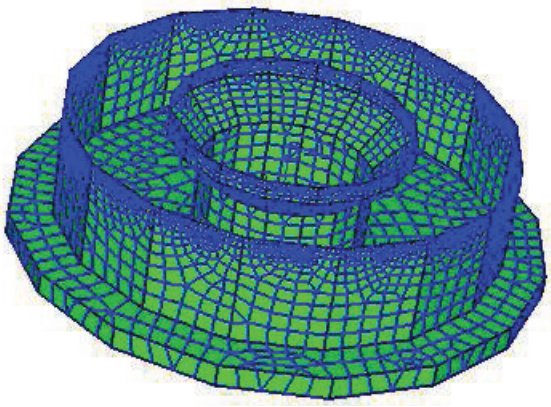


Figura C.17: Malha gerada no Sstab do corte do modelo simplificado mostrado na figura D.10.

A figura C.17 mostra o resultado da malha gerada utilizando um tamanho médio de painel de 0.5 para 1, obtendo-se um total de 4892 painéis.

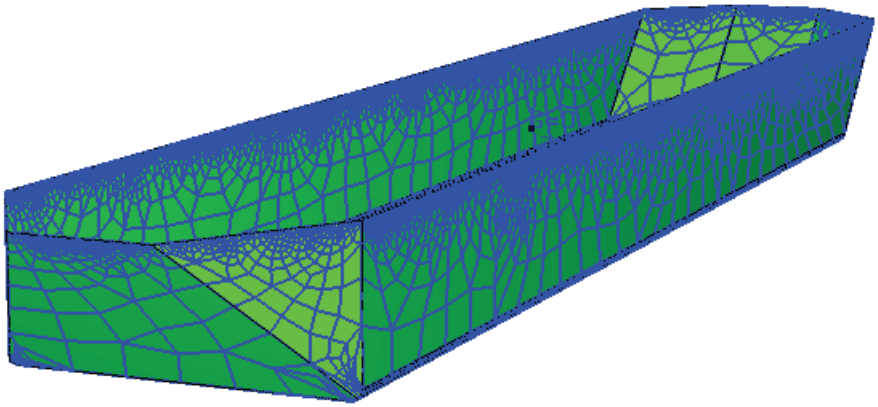


Figura C.18: Malha gerada com tamanho de painéis de 0.5 para 10.

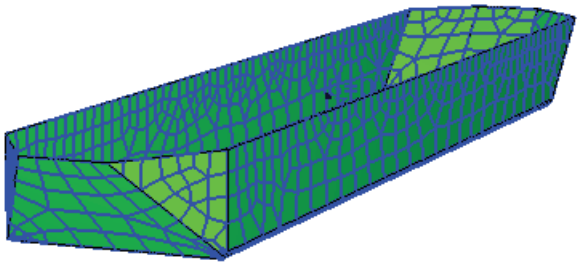


Figura C.19: Malha gerada com tamanho de painéis de 5 para 10.

A figura C.18 mostra a malha gerada para o casco cortado de um navio, cujos parâmetros médios de painéis são de 0.5 no plano da linha da água e 10 no plano da quilha. A malha gerada possui um total de 7715 painéis. A figura C.18 mostra o mesmo modelo, porém com tamanho médio de painel variando de 5 para 10, cujo número total de painéis é 1862.

O módulo de corte do Sstab gera malhas com elementos triangulares ou quads, em geometrias tridimensionais compostas por faces planas. A idéia geral consiste em deslocar cada face que compõe a geometria tridimensional para o espaço bidimensional, usando-se transformações geométricas e, em seguida, aplicar o algoritmo de triangulação. Depois de gerados os elementos, novas transformações geométricas são aplicadas a fim de enviar a face de volta para sua posição original no espaço tridimensional. Utiliza-se também neste módulo um algoritmo implementado por Miranda [Miranda00] que gera malha em contornos planares, podendo-se escolher entre gerar triângulos ou quads.

## C.3
## Pós-processamento das Malhas Geradas

Em simulações numéricas dois importantes aspectos que devem ser considerados são a geração da malha e a definição do grau de refinamento associado a essa malha. Esse refinamento deve ocorrer de modo que os elementos gerados varie de acordo com a curvatura de cada superfície que compõe um determinado modelo.

O modelador geométrico utilizado neste trabalho utiliza o algoritmo de geração de malhas implementado por Miranda [Miranda99] [Miranda02]. Este algoritmo utiliza a descrição paramétrica da discretização das curvas bordo de cada superfície, combinando a técnica de avanço de fronteira com decomposição espacial recursiva. Essa decomposição espacial é realizada utilizando-se uma árvore quaternária (*quadtree*) para armazenar métricas e desenvolver diretrizes locais usadas na definição do tamanho dos elementos que serão gerados. Isto é para cada superfície é gerada uma malha de fundo (*quadtree*) que captura a variação de curvatura da superfície, sendo mais refinada a *quadtree* onde existe maior curvatura (figura C.20).

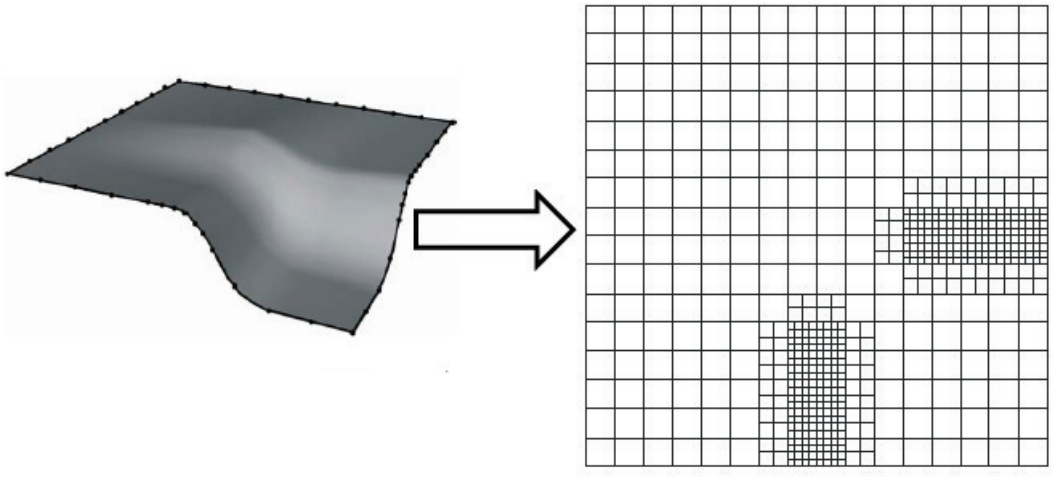


Figura C.20: Malha de fundo de uma superfície [Miranda02].

Desta forma, a distribuição do tamanho dos elementos triangulares resultantes para uma superfície é deduzida pelo tamanho das arestas de contorno fornecida como dado de entrada, a qual é utilizada para gerar a *quadtree*.

Dependendo da subdivisão das curvas de bordo de uma superfície, a malha resultante gerada pelo algoritmo de Miranda, pode não ter bons resultados. Isto é, a malha gerada não representa corretamente as variações de curvatura da superfície, devido a que subdivisão inicial da malha de fundo (*quadtree*) não foi ótima. Este resultado é mostrado na figura C.21.


Figura C.21: Malhas com diferentes discretizações.

A figura C.21a mostra uma superfície bilinear cujas linhas retas possuem uma única subdivisão enquanto que a figura C.21b mostra as linhas retas com 10 subdivisões. Pode-se perceber que o grau de subdivisão das curvas de bordo de uma superfície influência a malha gerada.

Geralmente o plano de balizas de um navio é composto por linhas retas e splines como mostrado na figura C.22. O problema da qualidade da malha

gerada depender do critério de subdivisão da fronteira de cada superfície gera vários amassamentos na modelagem de navios.

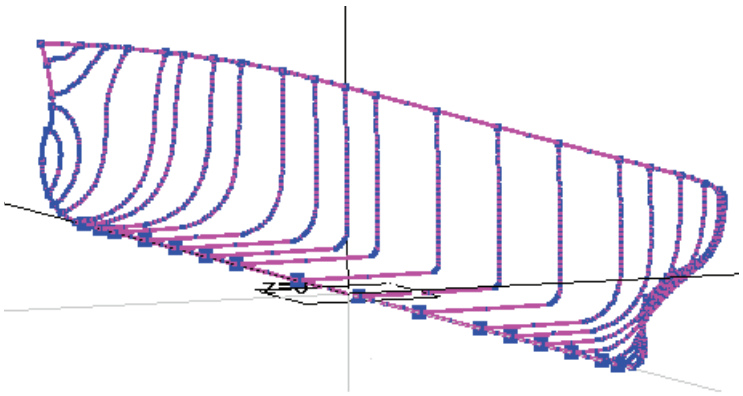


Figura C.22: Plano de balizas de um navio.

Desta forma, as malhas geradas que definem o casco de um navio podem precisar de um pos-processamento para corrigir estes erros, onde a discretização da variação da curvatura de cada superfície não for boa o suficiente.

A figura C.23 mostra o resultado da geração de malhas da seção transversal do casco de um navio, onde pode-se perceber o problema de amassamentos que surge devido a má discretização das superfícies que possuem grande variação de curvatura. As superfícies planas mesmo tendo uma discretização baixa na malha gerada não gera problemas pois todos os elementos estão no mesmo plano. É possível ainda ver na figura C.23 o número de vértices, número de faces, e a contribuição de volume calculado de cada face.

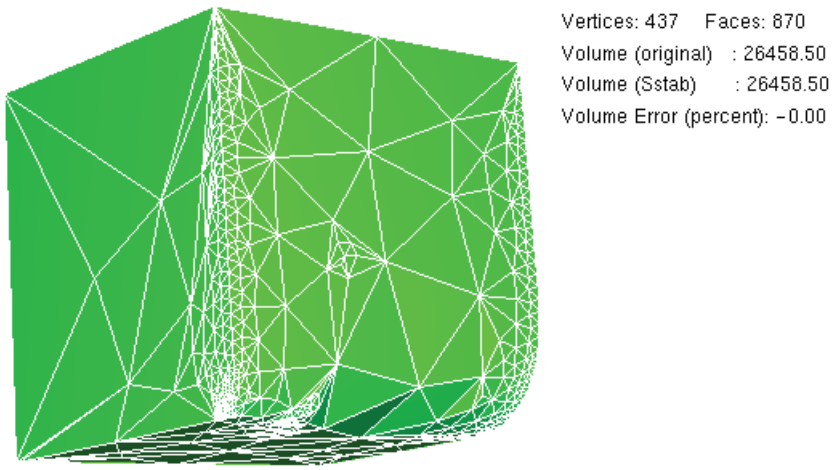
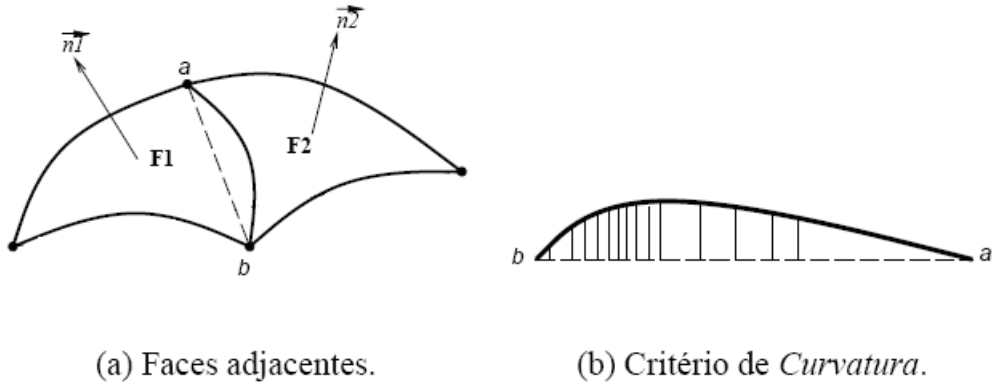



Figura C.23: Malha seção transversal.

O problema de amassamentos gera perda de volume do modelo final, pois o volume é calculado pela integração de cada face aplicando-se o *Teorema de*

*Stokes*. Uma idéia para resolver o problema de amassamentos da malha seria atrair as arestas dos triângulos que foram mal discretizados para à superfície.




(a) Faces adjacentes. (b) Critério de *Curvatura*.

Figura C.24: Critério geométricos para avaliação de arestas [Coelho98].

Desta forma pode-se dizer que o algoritmo de pós-processamento das malhas consta de 4 fases:

1. Subdivisão das arestas da malha de cada superfície.
2. Calculo da tolerância de área de corda.
3. Ordenação das arestas ruins.
4. Split das faces adjacentes a uma aresta ruim.

**Subdivisão das arestas da malha de cada superfície** Todas as arestas da malha cujas normais das faces adjacentes não são coplanares, são subdivididas em dez subdivisões para poder garantir que o calculo de tolerância da área de corda seja bom o suficiente, e o ponto onde a aresta será dividida seja o melhor possível (figura C.24a).

**Calculo da tolerância de área de corda** Cada ponto da subdivisão da aresta é atraída para à superfície, onde cada ponto atraido para a superfície forma um pequeno trapézio (figura C.24b). Estas pequenas áreas são somadas até que a soma seja maior que uma dada tolerância. A aresta cuja área de corda for maior que uma dada tolerância é armazenada em um vetor de arestas chamada de *rankededges*.

**Ordenação das arestas ruins** O vetor de arestas *rankededges* é ordenado pelo número de trapézios cuja área total foi maior que uma dada tolerância que definem um peso. Considerando neste caso 10 subdivisões por aresta, o maior

peso sera 10 e o menor sera 1. Desta forma , o vetor *rankededges* é ordenado colocando os maiores pesos no topo do array e os menores pesos no final do array.

**Split das faces adjacentes a uma aresta ruin** Uma vez identificadas as arestas ruins e o ponto ótimo de subdivisão, procede-se a dividir a aresta adjacente a duas faces, conforme mostrado na figura C.25, onde uma aresta adjacente a duas faces torna-se adjacente a quatro faces.

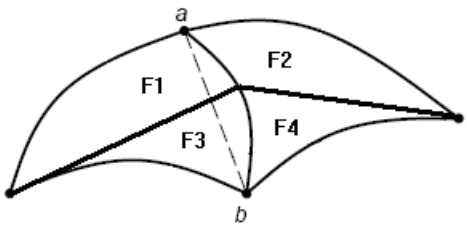



Figura C.25: Subdivisão da aresta.

O split das faces priorizará primeiramente todas as faces cujas arestas tiverem o maior peso, as quais são as faces que não discretizam corretamente a curvatura da superfície.

O resultado deste pós-processamento pode ser visto na figura C.26, onde pode-se perceber a diferença de volume do modelo original com a malha melhorada, assim como também pode-se perceber o incremento do número total de vértices e faces que definem a nova malha.

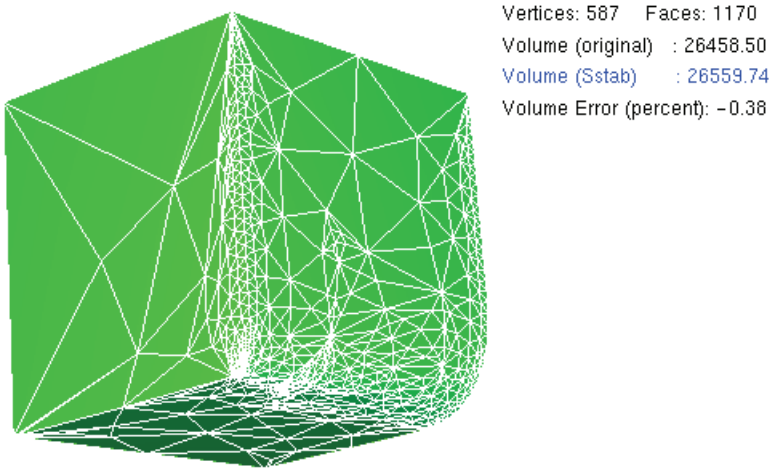



Figura C.26: Malha após pós-processamento das faces.

O pós-processamento das malhas é um processo iterativo, pois a cada fase de pos-processamento novas faces serão geradas. Estas novas faces precisam ser novamente testadas e caso não passem o critério de tolerância de área da corda, estas devem ser novamente subdivididas. O pós-processamento termina quando nenhuma aresta da malha viola a tolerância de área.

Um outro exemplo mostra-se na figura C.27 onde a malha gerada de um navio possui o efeito de amassamento onde há variação de curvatura na malha.

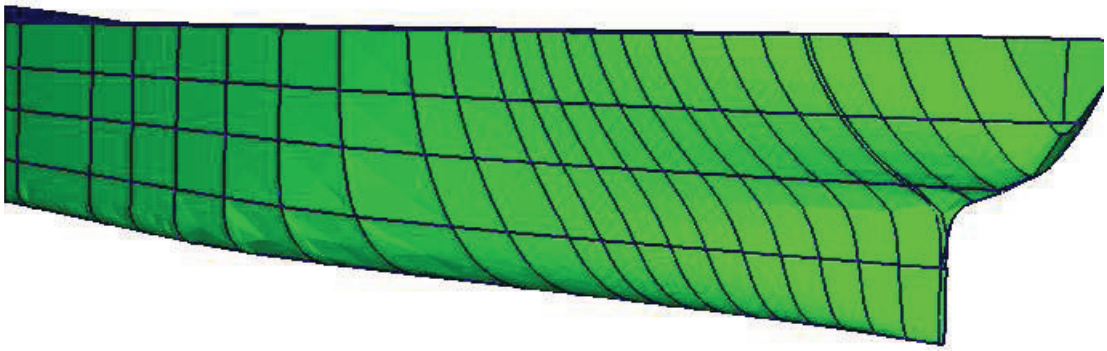


Figura C.27: Malha de um navio com efeito de amassamentos.

A figura C.28 mostra a mesma malha do navio mostrado na página anterior, porém após realizado o pós-processamento das malhas. Nota-se que há uma grande melhoria na qualidade da malha sendo quase imperceptível o problema dos amassamentos que surge devido a má discretização da malha nas regiões onde existe variação de curvatura.

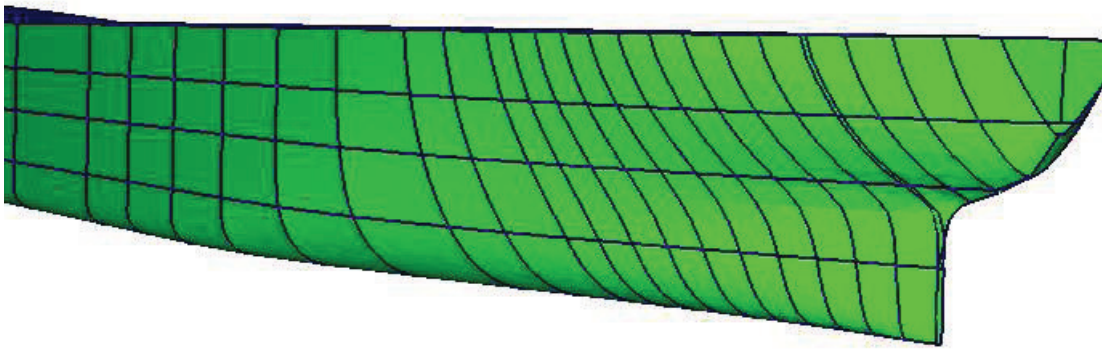


Figura C.28: Malha de um navio sem efeito de amassamentos.

O pós-processamento das malhas resolve vários problemas da qualidade da malha onde o algoritmo de Mirando [Miranda99] não consegue discretizar corretamente os elementos a serem gerados onde há variação de curvatura.

## C.4
## Composição de Volumes

A modelagem de uma estrutura flutuante é composta pelo casco e seus compartimentos internos. Cada um destes compartimentos representa um volume único e é independente dos outros volumes. Em alguns casos, definir a modelagem das malhas, de alguns compartimentos internos, pode se tornar uma tarefa muito complexa, devido ao grande número de superfícies adjacentes entre si. Isto é, para definir um volume, deve-se selecionar o conjunto de superfícies que forma a geometria do volume desejado. A figura C.29 mostra a seleção das superfícies que compõem um compartimento interno de um navio.

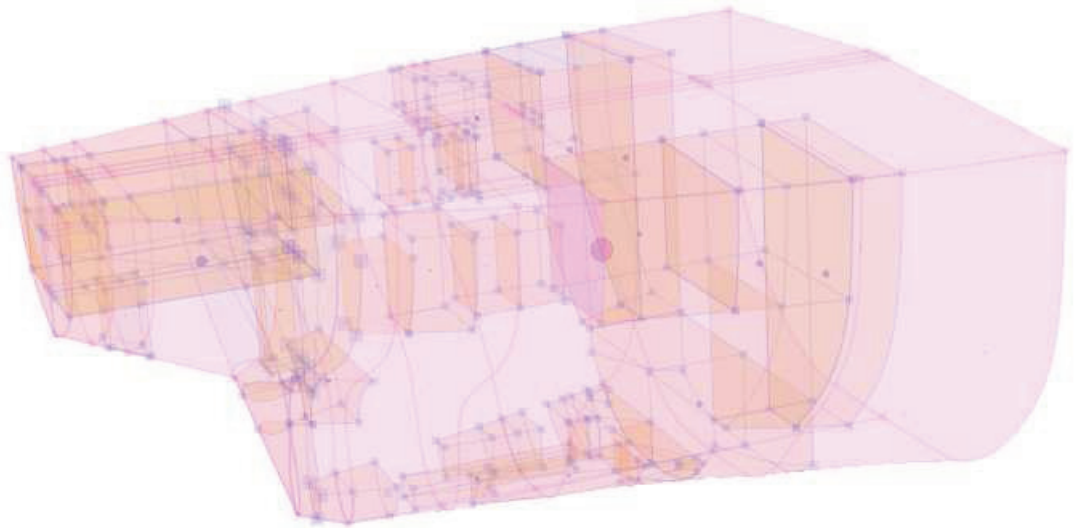


Figura C.29: Compartimentos internos de um navio.



Uma forma de resolver este problema é a possibilidade de poder definir o novo compartimento interno, a partir de alguns volumes previamente criados. Isto é, o volume do novo compartimento interno poder ser definido pela adição ou subtração de volumes, gerando um volume composto.

Pode-se dizer que o processo de adição ou subtração de volumes se assemelha à modelagem por CSG, onde, a partir de algumas primitivas básicas e a aplicação de operações booleanas, pode-se definir um volume mais complexo. É necessário ressaltar que a modelagem por CSG considera a fronteira e o interior de uma primitiva básica de forma implícita, enquanto que a modelagem neste trabalho utiliza a representação por contornos ou BRep.

Desta forma, o ambiente gráfico incorporado no MG, além de verificar a consistência topológica de uma malha e simplificar a malha, pode ainda realizar composição de volumes. Todo volume composto no MG é referenciado por uma

árvore de entidades chamada de *Tank Sets*, conforme mostrado na figura D.2, aqui reproduzida.

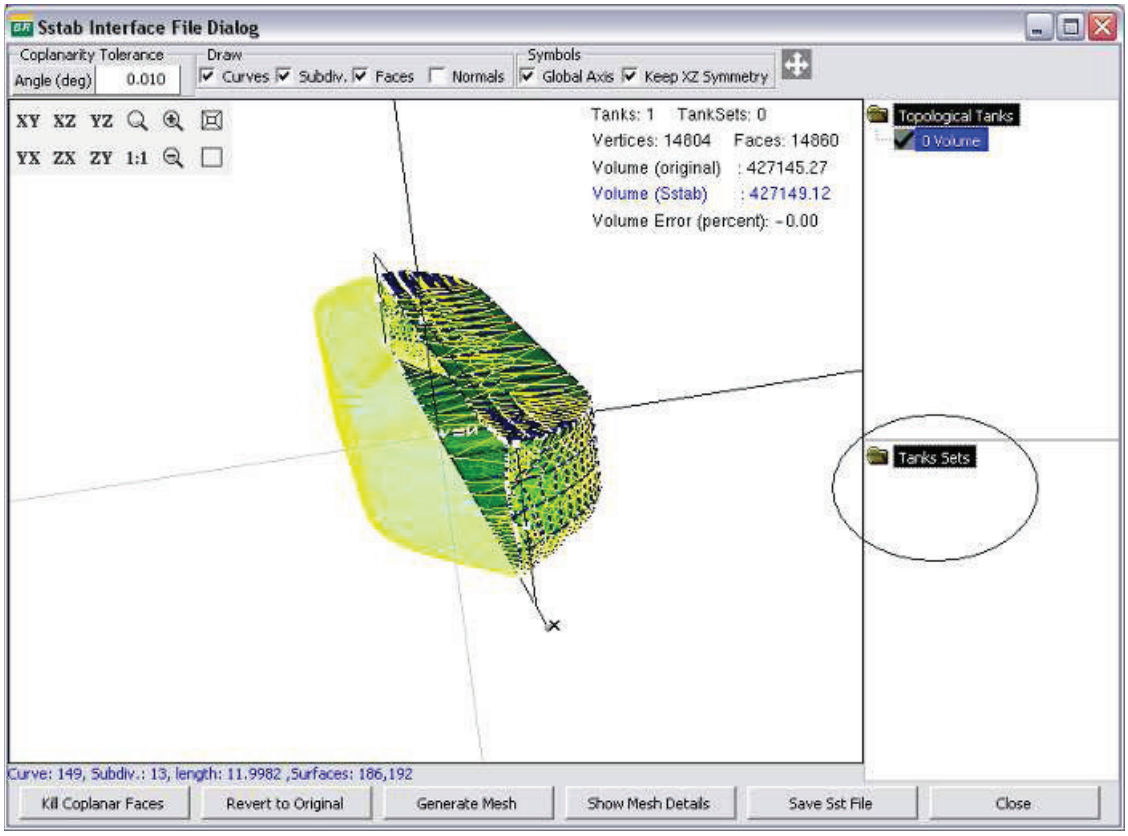



Figura C.30: Árvore de composição de volumes.

O resultado do volume composto também é uma malha topologicamente válida, pois a composição somente considera volumes topologicamente corretos, podendo estes ser simplificados ou não. Esta composição de volumes será chamada de Pseudo-CSG.

A figura C.29 mostra um caso típico de modelagem de estruturas flutuantes, onde é necessário compor vários volumes para poder criar um novo volume que seja definido pela diferença do casco externo, contra todos os compartimentos internos da estrutura flutuante. Para isto, selecionam-se todos os volumes a serem compostos e define-se a orientação das malhas individualmente para cada volume, conforme mostrado na figura C.31, onde observa-se que o número de volumes do modelo mostrado na figura C.29 é 36.

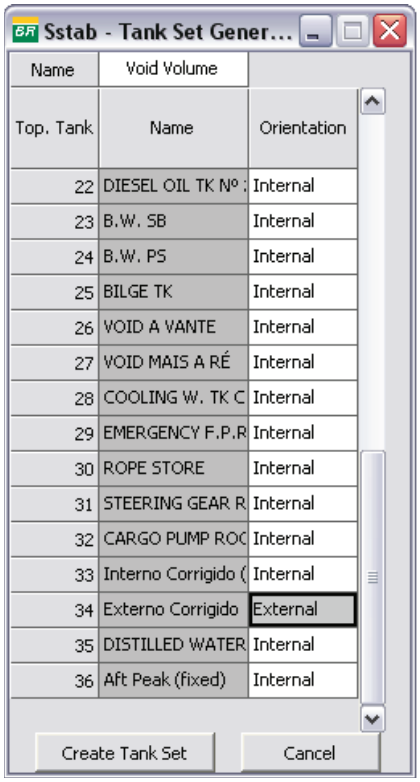



Figura C.31: Interface de composição de volumes.

Há duas orientações possíveis para cada volume: *External* e *Internal*, que definem a contribuição de cada face para o volume composto. É possível definir ainda um nome para este novo volume.

***External Volume*** Esta orientação indica que a contribuição de todas as faces deste volume será positiva. Isto é, todas as faces terão as suas normais orientadas para fora do modelo.

***Internal Volume*** Esta orientação indica que a contribuição de todas as faces deste volume será negativa. Isto é, todas as faces terão as suas normais orientadas para dentro do modelo.

A figura C.32 mostra o resultado da composição de volumes, a partir da figura C.29 definindo um novo compartimento interno do casco do navio.

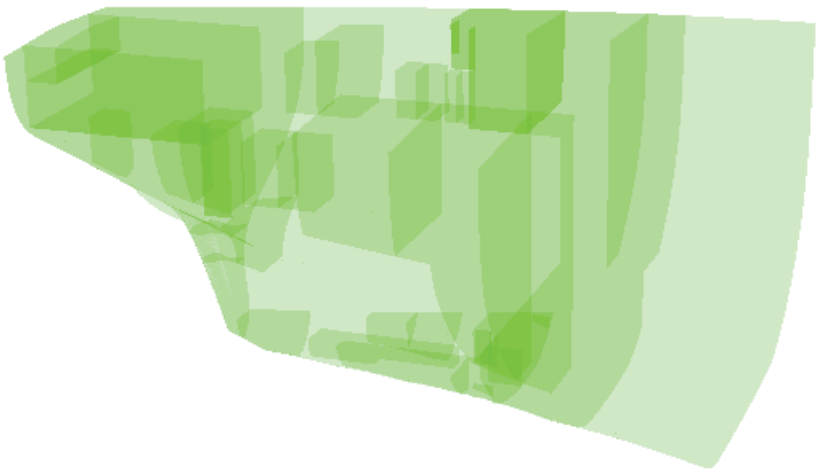


Figura C.32: Compartimento interno composto de um navio.

A tabela C.1 mostra a contribuição de volume do casco externo, volumes dos cascos internos, e o volume composto, respectivamente.

| ***Nome Volume*** | Volume($m^3$) |
|---|---|
| Casco Externo | 61496.71 |
| Cascos Internos | 17187.24 |
| Void Volume | 44309.47 |

Tabela C.1: Contribuição de volumes após a composição de volume.

Pode-se dizer que a composição de volumes é uma excelente ferramenta que facilita a criação de novos modelos, a partir de um conjunto de volumes. Cada volume é modelado de forma independente, como é feito na modelagem por CSG, porém a modelagem de composição neste caso se faz de forma explícita.

# D
# Simplificação de Malhas

Este apêndice, apresenta as novas ferramentas incluídas nos programas MG [Coelho98] e Sstab [Coelho03], as quais dizem respeito à verificação da consistência topológica das malhas e simplificação.

Diversos modelos de representação têm sido desenvolvidos para sólidos em $\mathbb{R}^3$ e a representação por modelos de contorno, especialmente aquela baseada em polígonos. Existem muitas estruturas de dados que manipulam malhas poligonais com eficiência como a *Winged-Edge* [Baumgart74], a *Half-Edge* [Mantyla88] e a *Handle-Edge* [Lop96], por exemplo.

Pode-se citar, também, o trabalho de Diaz [Diaz04], onde é utilizado uma estrutura de dados do tipo *Half-Edge*. A estrutura é chamada *CHalfEdge* e é uma versão compacta da *Half-Edge* original, em termos de representação e custo computacional.

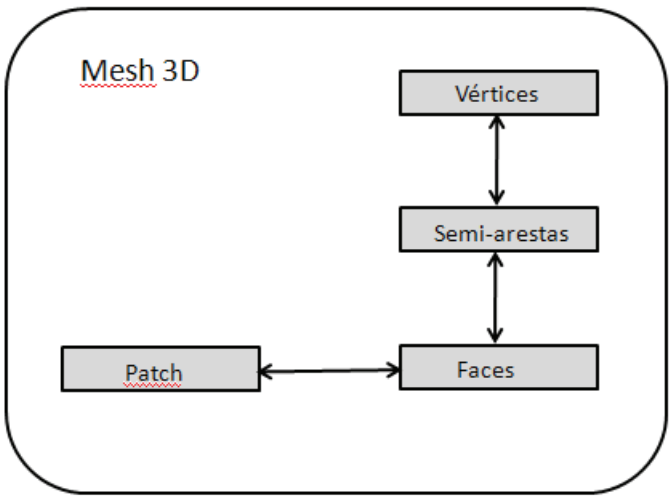



Figura D.1: Estrutura de dados topológica.

A estrutura de dados implementada neste trabalho pode ser vista na figura D.1 e trata-se de uma versão modificada da *CHalfEdge*. Da mesma forma que a *CHalfEdge*, utiliza-se também o conceito de semi-arestas para detectar inconsistências topológicas numa malha.

Nesta estrutura de dados, os objetos que compõem o modelo topológico de uma malha ou sólido 3D são 4 *arrays* dinâmicos: **Vértices**, **Faces**, **Semi-arestas** e **Patch**. Cada vértice possui uma lista de todas as semi-arestas que saem deste vértice. Cada face possui uma lista das semi-arestas que definem a fronteira de uma determinada face. Um *Patch* possui uma lista para um conjunto de faces, que representa uma superfície. A lista de semi-arestas para cada vértice e face são referências do vetor global de semi-arestas.

## D.1
## Simplificação de Malhas Utilizando a Estrutura de Semi-Arestas

A figura D.2 mostra o ambiente gráfico, incorporado no modelador geométrico, utilizado para realizar a verificação da consistência topológica e simplificação de malhas. Cada malha pertence a um volume e os volumes são armazenados numa árvore chamada de *Topological Tanks*.

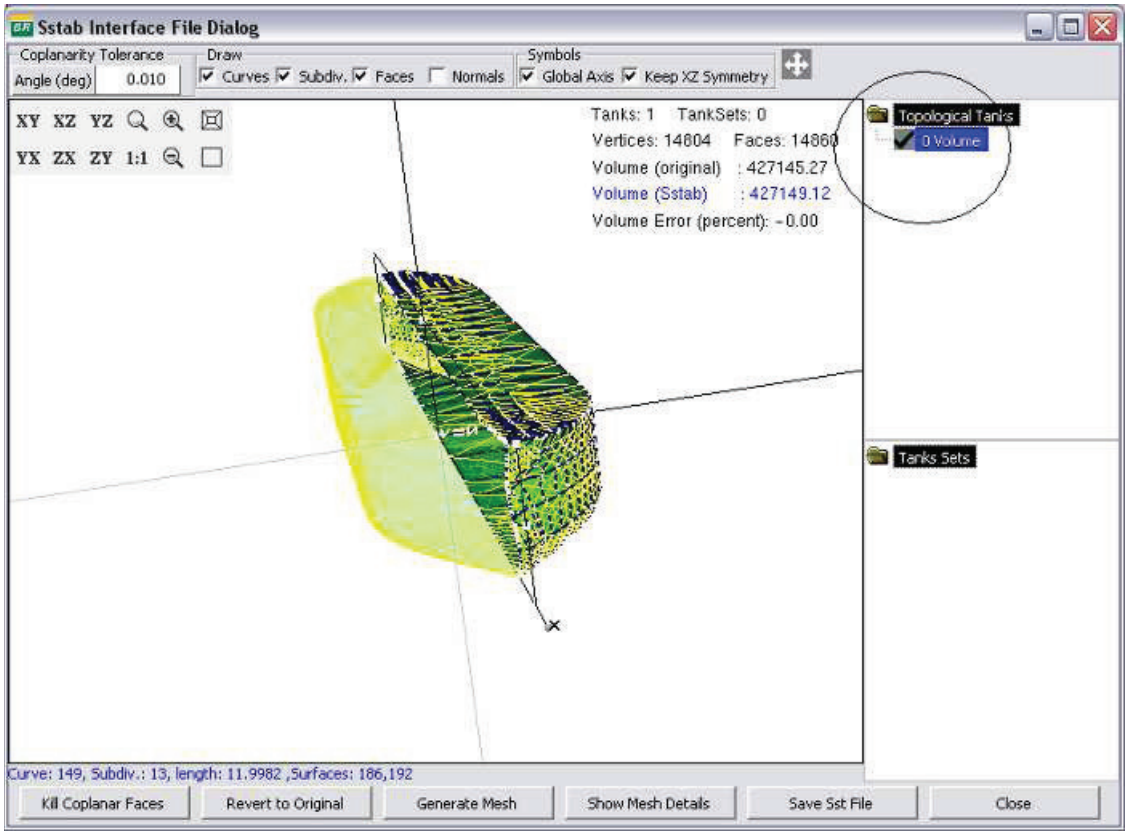



Figura D.2: Simetria de uma unidade flutuante.

Na maioria dos casos, estruturas flutuantes são construídas por simetria com relação a determinados planos. Isto é, somente metade ou um quarto do modelo é modelado quando existe simetria. O modelo final é obtido realizando-se processos de espelhamento nos planos de simetria.

Inclinações permanentes, como *trim* ou *banda*, podem aparecer nas simulações e cálculos de estabilidade estática. Para evitar este problema, a malha resultante do processo de simplificação deverá manter as características do modelo original. Isto é, se o modelo for simétrico, a malha simplificada também deverá ser simétrica.

Considera-se, neste trabalho, que o processo de simplificação será feito por retalhos. Cada retalho mantém a sua fronteira de bordo após a simplificação.

## D.2
## Construção das Semi-Arestas

Para poder simplificar, ou verificar se a malha está consistente, é necessário encontrar, para cada semi-aresta, a sua correspondente simétrica (*mate*). Para isto, utilizam-se estruturas do tipo R*-tree [Beckmann90], que obtêm as arestas simétricas realizando poucos testes.

Cada semi-aresta possui uma *bounding box*, a qual é usada para determinar, na R*-tree, o conjunto de semi-arestas que potencialmente intercepta uma dada semi-aresta (figura D.3) [Coelho98].

Uma semi-aresta que tem sua semi-aresta simétrica instanciada é removida da árvore, pois esta não será mais referenciada por outra semi-aresta. Esta remoção acelera o processo de encontrar todas as semi-arestas do modelo, onde somente consideram-se semi-arestas válidas aquelas cujos *mates* ainda não foram instanciados.

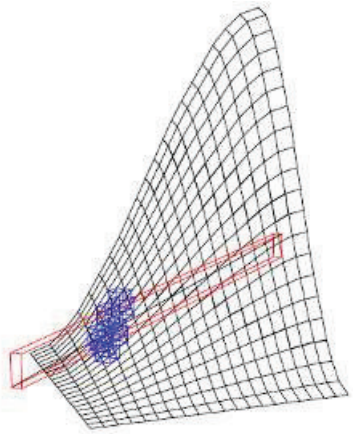


Figura D.3: Faces de um retalho que potencialmente interceptam uma aresta [Coelho98].

O algoritmo 3 mostra o processo utilizado para encontrar as semi-arestas simétricas para uma malha qualquer utilizando estruturas de dados espaciais do tipo Rtree.

```
algoritmo void FindMateEdges()
 1 for( i = 0 ; i < vEdges.size() ; i++ )
 2 {
 3 // get current halfedge
 4   h = vEdges[i];
 5   RtreeInitSearchBox(r3alledges,box(h));
 6 //find mate of h in r3alledges
 7   while(edge=RtreeSearchBox(r3alledges)!=NULL)
 8   {
 9     if (!vertexidentical(v1(h),v2(edge)) continue;
10     if (!vertexidentical(v2(2),v1(edge)) continue;
11     mate(h)=edge;
12     mate(edge)=h;
13     RtreeDelete(r3alledges,edge);
14     break;
15   }
16 }
fim
```

**Algoritmo 9:** Procura do mate para cada semi-aresta



O processo de encontrar todas as semi-arestas simétricas é uma fase muito importante, pois sem esta informação seria muito difícil extrair informações topológicas de uma malha.

## D.3 Processo de Simplificação

A simplificação consiste dos seguintes passos:

1. Identificar se o modelo possui simetria.
2. Simplificar retalhos da parte simétrica.
3. Espelhar faces simplificadas nos planos de simetria.

No passo 1, a simetria do modelo é, geralmente, aparece no plano XZ, como no navio mostrado na figura na figura D.2.

No passo 2, são feitos 3 sub-passos para a simplificação das faces simétricas, os quais são: Eliminar faces coplanares, Eliminar vértices colineares e Construir fronteira simplificada (superface).

**Eliminar faces coplanares** O processo de simplificação das faces coplanares é feito utilizando um critério de tolerância para poder verificar a igualdade das normais de duas faces adjacentes. Este critério considera o ângulo entre as normais das faces. Isto é, a diferença entre os dois vetores deve ser menor que uma dada tolerância definida pelo usuário.

A unificação das faces somente será feita para cada retalho, individualmente, preservando as curvas de bordo de cada superfície. O algoritmo apresentado a seguir mostra a simplificação para um determinado retalho, passado como parâmetro para a função EliminarFacesCoplanares (algoritmo 2).

A tolerância de coplanaridade de normais (*Coplanarity tolerance*) está contemplada na interface de simplificação, mostrada na figura D.2. O usuário possui total controle sobre este parâmetro. Quanto maior for a restrição de ângulo entre normais, fornecida pelo usuário, mais faces serão eliminadas do modelo e vice-versa.

```
algoritmo EliminarFacesCoplanares(Patch * patch)
 1 for ( he = 0 ; he < patch->Vedges.size(); he++ )
 2 {
 3  CHalfEdge *h1 = Vedges[i]; CHalfEdge *h2 = mate(h1);
 4  if h1 is marked continue;
 5  if (face(h1) == face(h2)) /* dangling edge*/
 6  {
 7    mark as visited h1 and h2;
 8    delete pointer h1 from list of edges of h1->vertex;
 9    delete pointer h2 from list of edges of h2->vertex;
10    delete pointer h1 from face(h1);
11    delete pointer h2 from face(h1);
12  }
13  else
14  {
15    /*simplify faces*/
16    if !(EqualNormal(face(h1),face(h2))) continue;
17    mark as visited h1 and h2;
18    delete pointer h1 from list of edges of h1->vertex;
19    delete pointer h2 from list of edges of h2->vertex;
20    mark face(h2) to be deleted;
21    delete edge h1 from face(h1)
22    add all edges from face(h2) into face(h1)
23  }
24 }
25 elimina todas as faces marcadas para serem deletadas;
fim
```

**Algoritmo 10:** Eliminação de faces coplanares

**Eliminar vértices colineares** Este módulo remove todos os vértices colineares de uma superface ou face simplificada. Todos os vértices internos foram removidos pelo sub-passo 1, somente resta remover os vértices da fronteira que são colineares entre si.

Desta forma, dada uma semi-aresta h, verifica-se se existe colinearidade com uma semi-aresta adjacente a h. Isto é, testa-se o vértice da semi-aresta h, o vértice da semi-aresta simétrica de h, e o vértice da semi-aresta adjacente a h.

**Construir fronteira simplificada** A construção de cada fronteira simplificada orienta todas as arestas num único sentido. Caso haja *loops* internos, estes terão um sentido contrario à orientação do loop externo.

Se o modelo possui simetria, o passo 3 é executado espelhando-se todas as faces simplificadas com relação ao planos de simetria do modelo, e invertendo a orientação de cada face. Isto é feito, para evitar o surgimento de inclinações de trim ou bando nas simulações de análise estática do modelo (Sstab).

O módulo de simplificação de malhas ainda fornece ao usuário uma comparação em porcentagem de erro, entre o volume real e o volume do modelo simplificado (figura D.2). Caso o erro do volume seja muito grande, deve-se realizar uma nova simplificação mudando a tolerância de coplanaridade de normais. Além disso, será criada a árvore de volumes (*Topological Tanks*). Caso não haja erro de topologia, o volume será mostrado em azul, caso contrario, será mostrado em vermelho.

## D.4 Resultados de modelos navais com a simplificação proposta

Nesta seção apresentam-se alguns resultados do algoritmo de simplificação de malhas proposto. A seguir são apresentados alguns modelos gerados no MG.

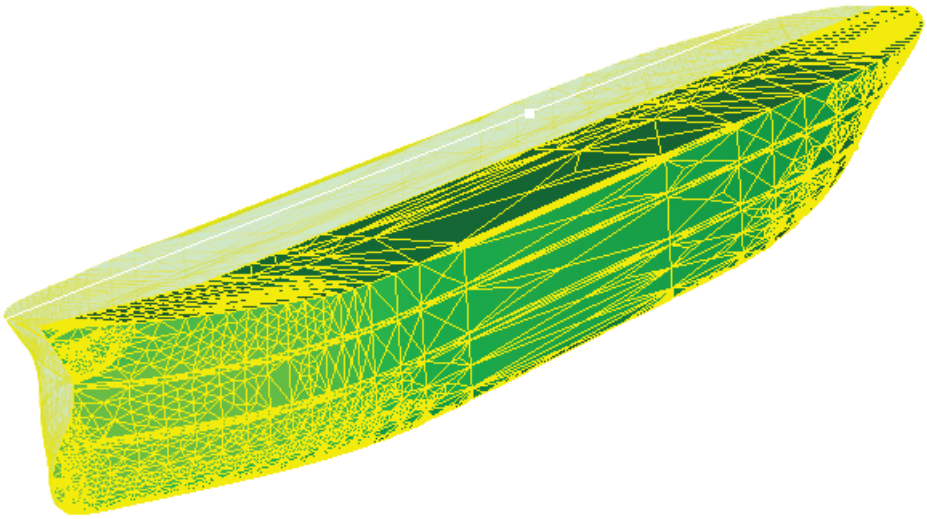


Figura D.4: Malha original de um navio.

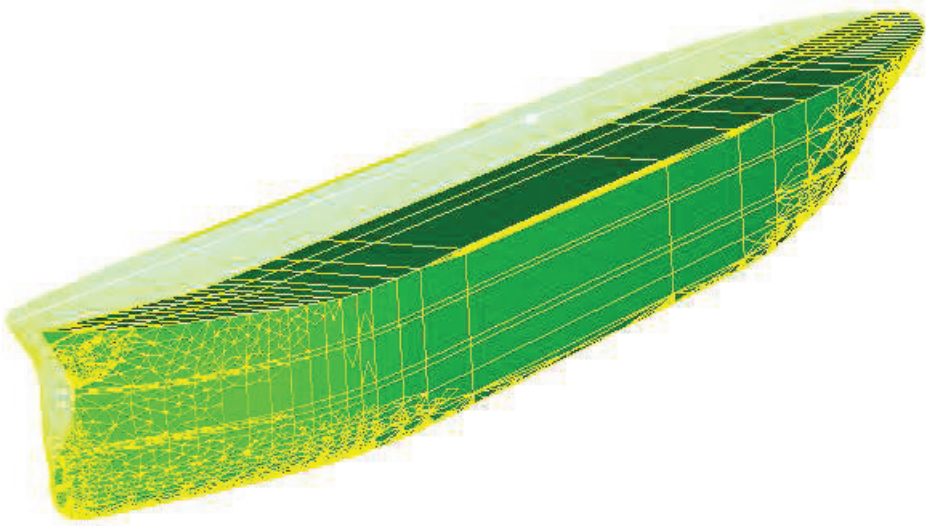


Figura D.5: Malha simplificada de um navio.

Como mencionado anteriormente, o ambiente de simplificação de malhas detecta, automaticamente, a simetria do modelo. Caso exista simetria, somente metade do modelo é considerado e simplificado. A outra metade é obtida por espelhamento e desenhada de forma transparente (figura D.4). A simplificação (figura D.5) é feita individualmente para cada retalho, e somente da parte simétrica do modelo. Após a simplificação as faces são espelhadas de forma a manter a simetria do modelo.

As figuras D.6 e D.7, mostram a malha original de uma plataforma e, o resultado do processo de simplificação, respectivamente.

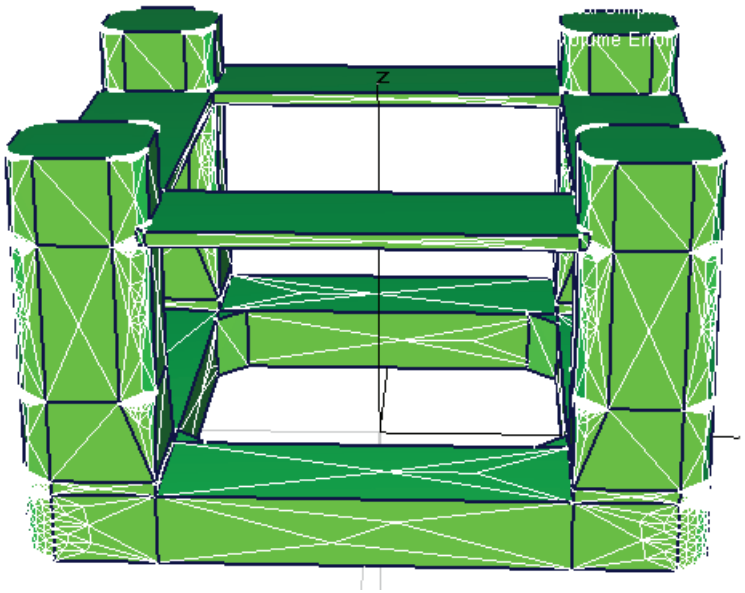



Figura D.6: Malha original de uma plataforma.

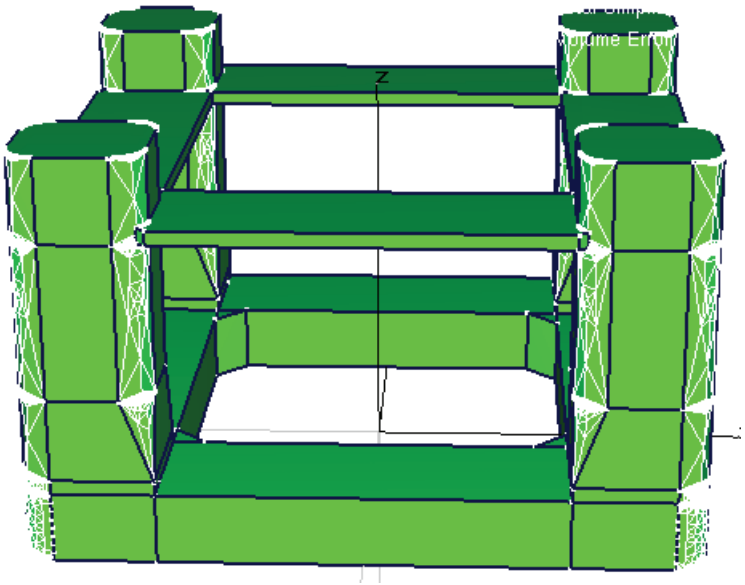



Figura D.7: Malha simplificada de uma plataforma.



Determinados modelos não possuem simetria, pois algumas superfícies não estão totalmente contidas num plano de simetria. O MG considera o plano XZ como plano de simetria de referência.

As figuras D.8 e D.9 mostram o compartimento interno de um navio e o seu casco respectivamente, os quais possuem superfícies não simétricas ao plano XZ. As superfícies que não são simétricas são desenhadas em cor amarela, de tal forma que o usuário possa identificá-las rapidamente. Caso seja necessário manter a simetria do modelo, deve-se remodelar esta superfície de tal forma a torná-la simétrica.

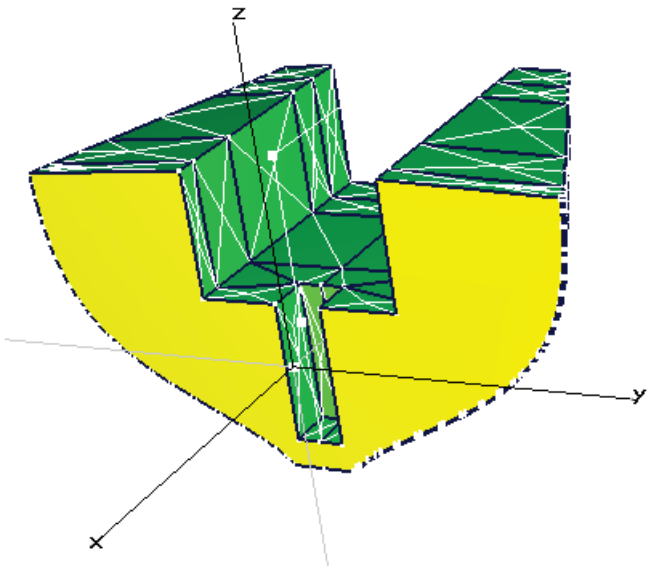



Figura D.8: Compartimento interno de um navio não simétrico ao plano XZ.

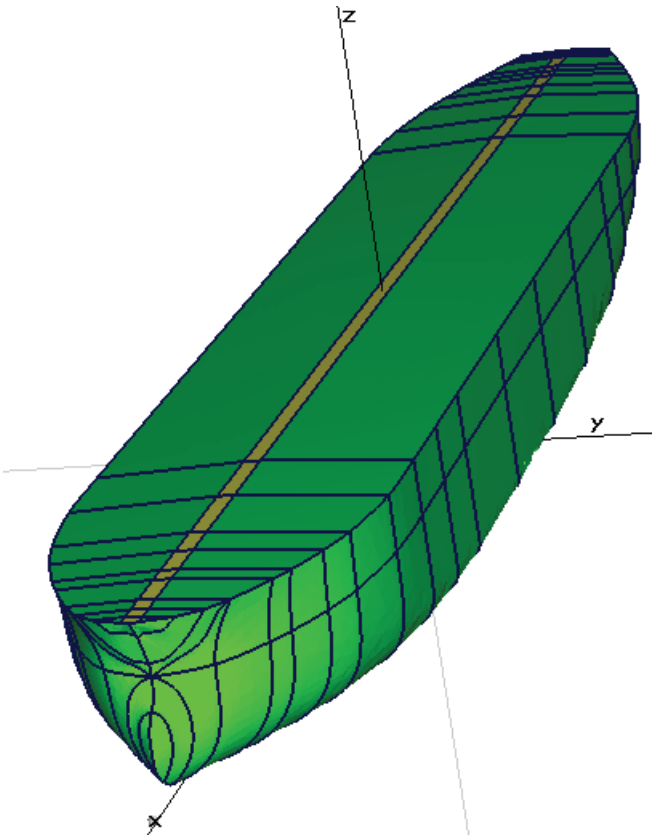



Figura D.9: Casco de um navio não simétrico ao plano XZ.

A seguir são apresentados alguns resultados da simplificação de malhas implementado no Sstab.

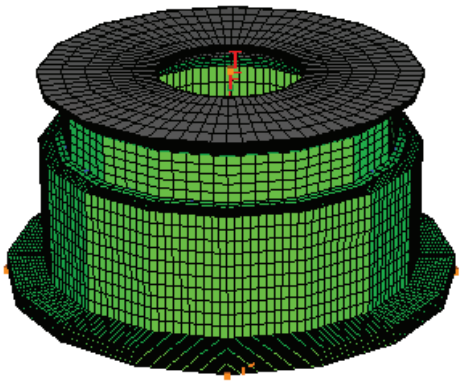


Figura D.10: Malha original de uma monocoluna.

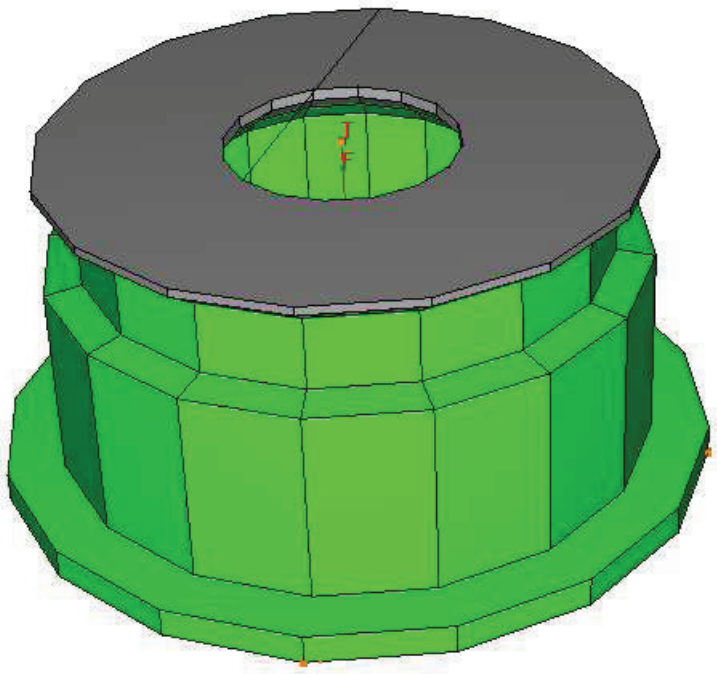


Figura D.11: Malha simplificada de uma monocoluna.

O resultado da simplificação da figura D.10, mostra o aparecimento de *loops* internos a uma superface simplificada (figura D.11). Estes *loops* são conectados por uma nova aresta criada, artificialmente, de forma a manter a consistência topológica da face. A figura D.11 possui uma única superface com um *loop* interno, enquanto que a simplificação da figura D.12 possui múltiplos *loops*, para uma única dada superface (figura D.13 e figura D.14).

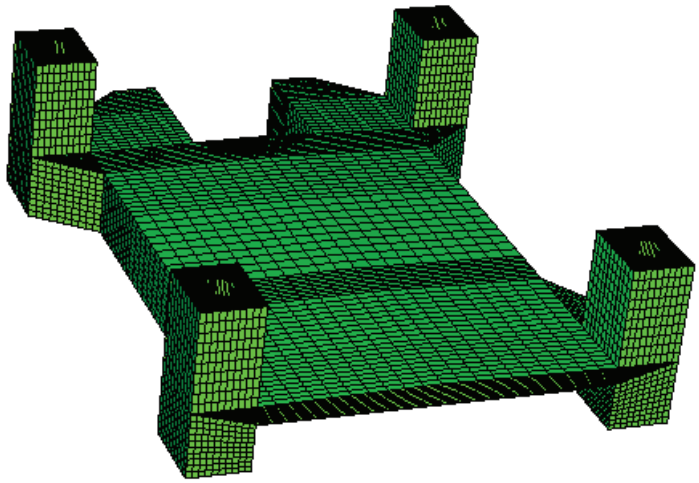


Figura D.12: Malha original do casco de uma plataforma.

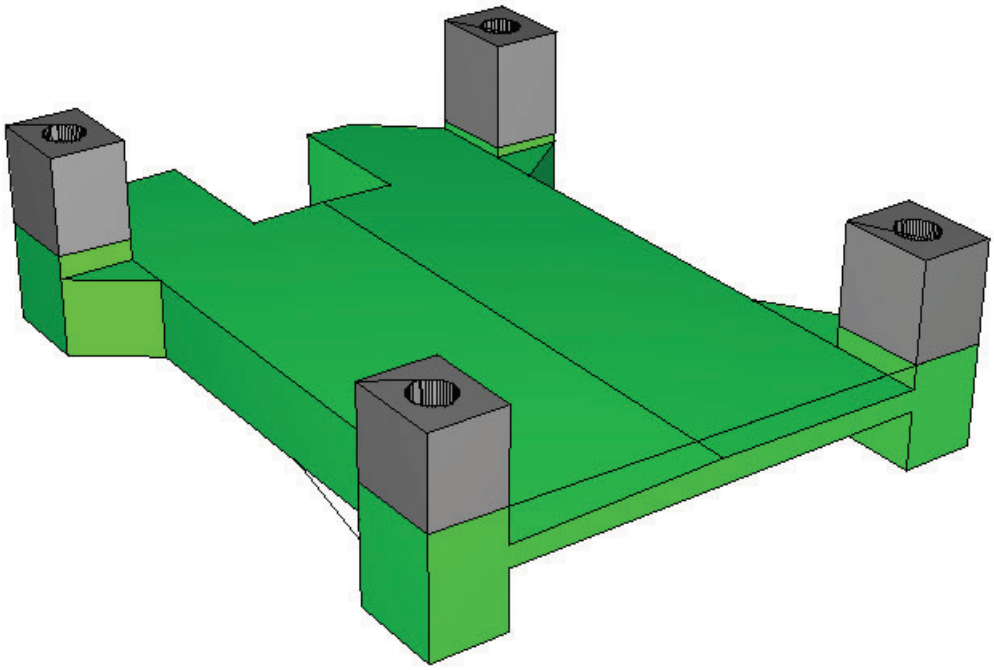


Figura D.13: Malha simplificada do casco de uma plataforma.

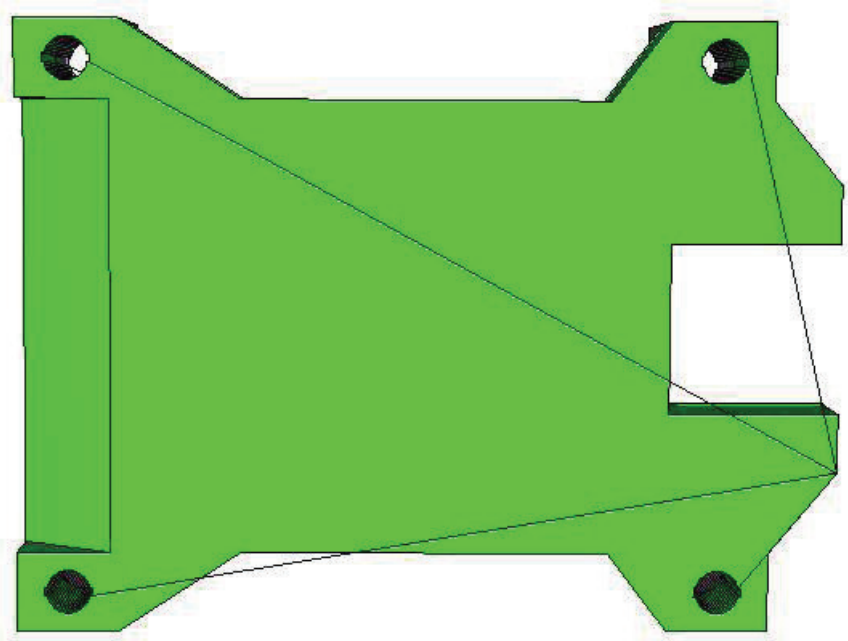


Figura D.14: Malha simplificada do casco de uma plataforma, vista inferior.

Como pode-se ver, o módulo de simplificação no Sstab funciona muito bem, na maioria dos casos, porém para modelos com alta curvatura não é possível ter uma boa simplificação. Dependendo das tolerâncias utilizadas nos testes de

coplanaridade entre duas faces pode-se, ou não, ter um bom resultado (figura D.15 e figura D.16).

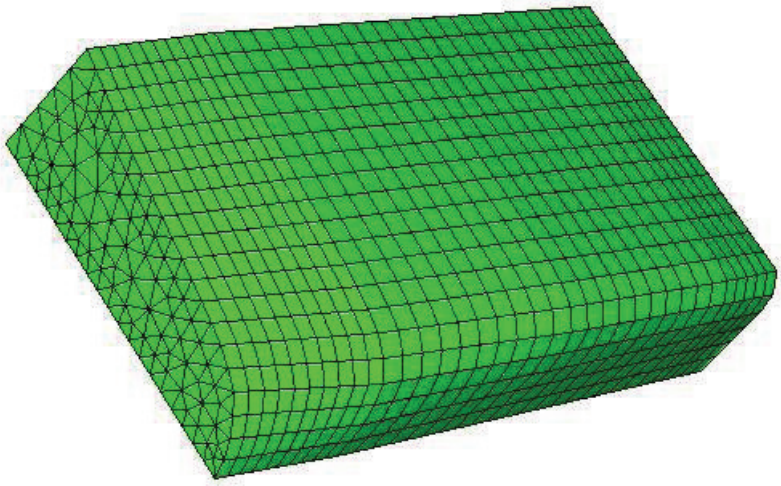


Figura D.15: Malha com uma boa variação de curvatura.



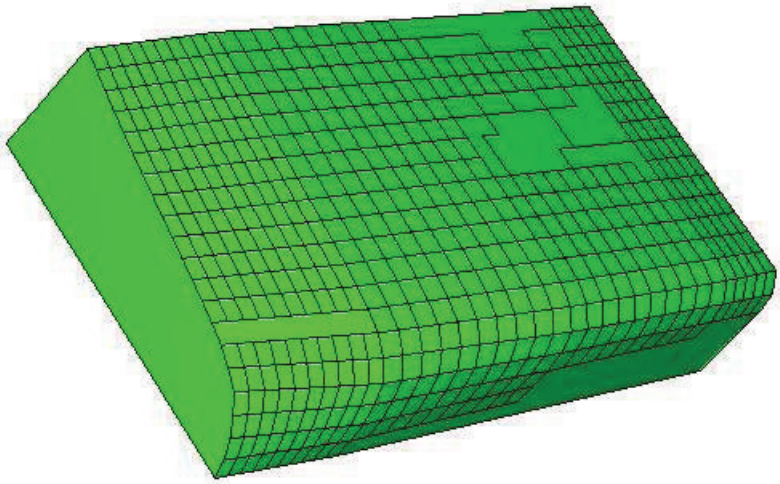


Figura D.16: Malha simplificada com uma boa variação de curvatura.

A figura D.16 mostra a malha simplificada da figura D.15, com poucas simplificações em regiões com variação de curvatura.

Problemas de topologia podem ser identificados, com o uso da estrutura de semi-arestas. A figura D.17 mostra uma superfície cuja malha está com alguns problemas topológicos. Isto é, existe uma face cujas arestas cruzam outras faces adjacentes. Uma forma de corrigir este erro é modificando a subdivisão da curva ou curvas de bordo da superfície que apresenta problemas (figura D.18).

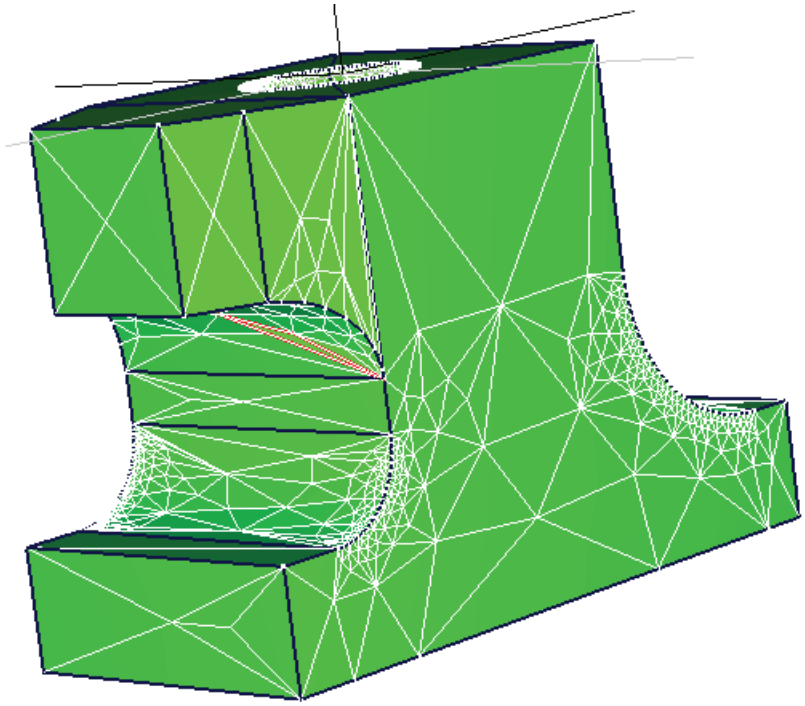


Figura D.17: Malha com erro de topologia.

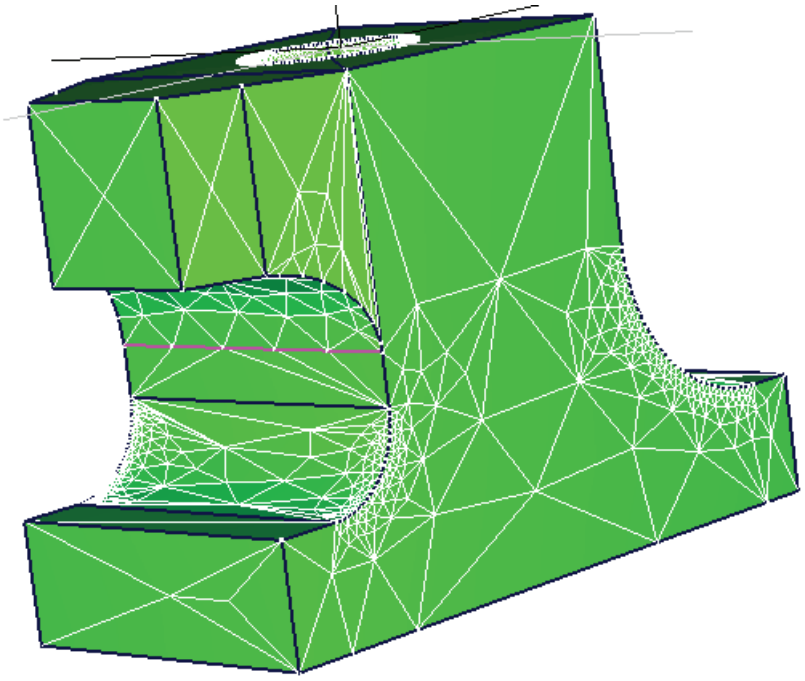


Figura D.18: Malha com erro de topologia corrigido.