6
Visualização de superfícies de falha por malhas neurais abertas

6.1.
Introdução

A estratégia de segmentação apresentada no capítulo anterior foi capaz de extrair um grafo para cada superfície de falha existente nos dados de teste. Cada grafo obtido é fortemente alinhado com a falha que representa. A visualização de um grafo como um conjunto de pontos no espaço tridimensional ligados por linhas permite ter uma idéia aproximada da superfície correspondente. Entretanto, para que o observador seja capaz de identificar facilmente a forma da superfície em cada ponto é necessário gerar uma representação visual mais completa.

A melhor forma de se representar uma superfície é através de uma malha triangular: placas gráficas de uso geral possuem suporte eficiente para sua visualização; além disso, em um modelo construído com o uso de malhas triangulares, uma série de operações envolvendo superfícies pode ser facilmente implementada. Assim, neste trabalho será desenvolvida uma técnica, novamente baseada em aprendizado competitivo, de forma a construir malhas triangulares para superfícies abertas, como as encontradas em falhas.

A primeira questão que se coloca é como identificar a distribuição de probabilidades que permita selecionar sinais de entrada para um algoritmo desse tipo. Uma solução seria identificar os sinais de entrada como sendo simplesmente os nós dos grafos resultantes da segmentação da rede gerada pelo Growing Neural Gas. Uma alternativa é utilizar os grafos segmentados como “filtros” sobre os dados originais.

Retomando os resultados do capítulo anterior, os subgrafos gerados pelo processo de segmentação representam as regiões do volume onde a estrutura de falha está bem caracterizada. Cada subgrafo define um subvolume de atributos de falha ou de probabilidades. Como cada subvolume é obtido a partir dos grafos segmentados, a topologia de um subvolume é também compatível com uma superfície aberta. A Figura 6-1 mostra o volume de probabilidades

sendo dividido em três partes, cada uma definida a partir de um dos três subgrafos identificados por uma segmentação prévia. Os vértices e arestas de cada grafo são distinguidos pelas cores vermelha, azul e amarela. Desse modo, o grafo segmentado dá origem a uma segmentação do volume do atributo de falha (ou do volume de probabilidades derivado do atributo).

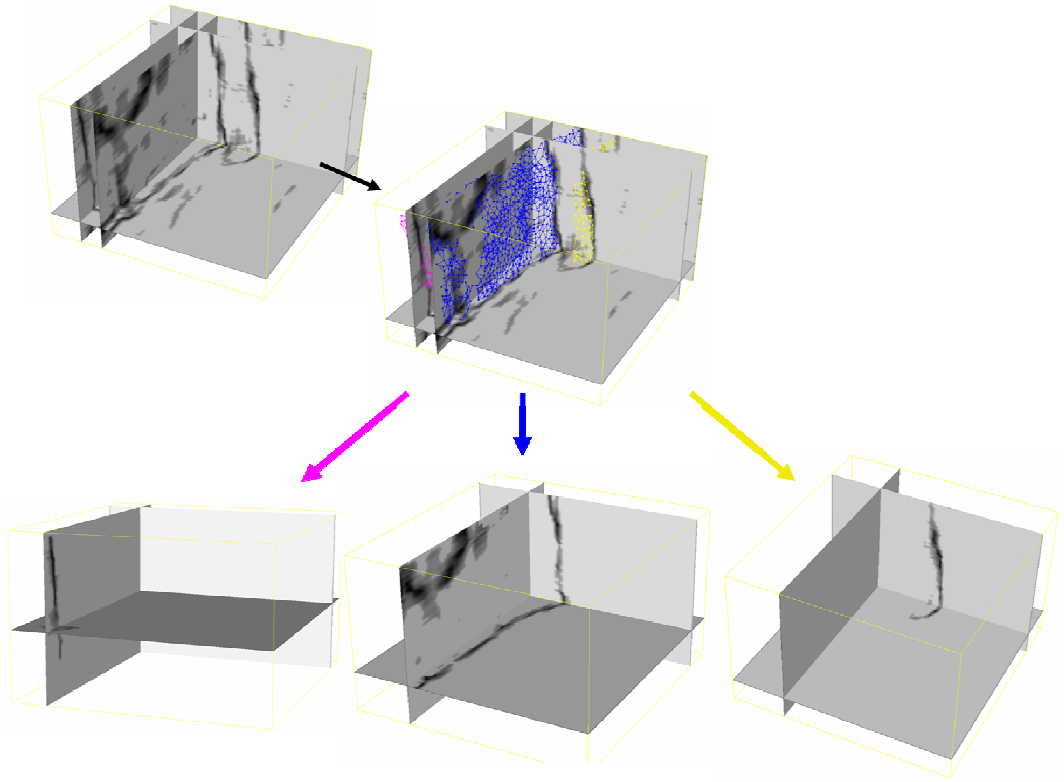

Figura 6-1: Volume de probabilidades é dividido em um conjunto de volumes de probabilidades específicos para cada falha identificada na segmentação do grafo (Mapeamento Grafo-Volume).

O mapeamento de cada subgrafo no volume de atributo de falha, denominado Mapeamento Grafo-Volume, é feito de forma que o valor de uma amostra seja preservado apenas se esta se encontra na vizinhança de algum nó do subgrafo. Caso contrário, o valor da amostra é zerado. Neste trabalho, a vizinhança de um nó é definida como sendo o cubo centrado nesse nó e de lado igual ao maior comprimento de suas arestas incidentes. Assim, cada um dos volumes de probabilidades resultantes desse mapeamento permite extrair amostras de uma mesma superfície.

Retomando a discussão sobre preservação de topologia apresentada no Capítulo 4, pode-se, agora, pensar na utilização de uma rede neural auto-organizável que mapeie o espaço de entradas em uma rede de estrutura

bidimensional. Esse tipo de abordagem foi proposto por Yizhou Yu (Yu, 1999). Yu utiliza uma rede SOFM (Kohonen, 1982) com nós organizados como vértices de uma triangulação para reconstruir superfícies a partir de um conjunto de pontos não-organizados. Além do método de Yu, a literatura apresenta uma segunda forma de utilização de redes auto-organizáveis para a reconstrução de superfícies chamada de **Neural Meshes** (NM). Esse método foi criado por Ivrissimtzis, Jeong e Seidel (Ivrissimtzis et al. 2003a, 2003b, 2004a, 2004b, 2005; Jeong et al. 2003) e é uma versão modificada do **Growing Cell Structures** (GCS)**,** uma rede auto-organizável proposta por Fritzke (o mesmo autor do GNG), combinada com uma técnica de suavização de malhas. Neste capítulo, são discutidos o GCS, a técnica de suavização de malhas pelo Laplaciano e o NM. Por fim, é apresentada uma proposta de adaptação do NM para o problema de visualização de falhas.

6.2. Growing Cell Structures

O modelo do Growing Cell Structures (Fritzke, 1993, 1994, 1997) é muito semelhante ao do Growing Neural Gas visto na seção 4.3. A principal diferença é que a topologia da rede é restrita a estruturas simpliciais de uma dimensão fixa escolhida a priori. A rede inicia como um simplex de dimensionalidade *k* e cresce mantendo os nós ligados de forma que toda a estrutura se mantenha composta apenas por esse tipo de bloco básico. Assim, a rede é formada apenas por uma linha para k=1, formada por triângulos para k=2 e por tetraedros para k=3.

O GCS é semelhante ao SOFM por ter a estrutura de conexões da rede fixa. Entretanto, difere deste por ser um método de aprendizado evolutivo, isto é, os nós da rede não são criados todos no início do processo, mas sim ao longo do aprendizado. Essa característica torna o GCS, em princípio, mais adequado para a tarefa de visualização do que o SOFM, pois o número de nós ao longo de cada uma das dimensões do mapa não precisa ser definido a priori (Fritzke, 1996). A distribuição dos nós no GCS resulta do próprio processo de aprendizado, sendo, portanto, mais adaptada aos dados de entrada do que no caso SOFM.

Como no GNG, tanto o nó vencedor como seus vizinhos topológicos sofrem o processo de adaptação. Cada nó mantém a informação de erro local; tal informação é utilizada para decidir onde inserir novos nós. O processo de inserção, entretanto, é diferente do GNG. Começa pela identificação do nó *q* de

maior erro acumulado; é também identificada a aresta de *q* com maior tamanho. Seja *f* o nó ligado a *q* por essa aresta; um novo nó *r* é então inserido, dividindo em duas a aresta que liga *q* a *f*. Além de ser ligado a *q* e *f*, o novo nó *r* também é ligado a todos os nós que são vizinhos simultâneos de *q* e *f*. Desta forma, a estrutura resultante consiste exclusivamente de simpliciais de dimensão *k*, novamente.

O algoritmo GCS é apresentado completo a seguir, conforme descrito por Fritzke (1997):

Algoritmo GCS

A entrada é um volume de probabilidades *p*. A saída é um conjunto de unidades *A*, onde cada unidade tem um peso **w** e um conjunto de conexões *C*.

1. Escolher a dimensionalidade da rede *k*. Iniciar conjunto *A* para conter *k*+1 unidades, com os vetores de referência escolhidos aleatoriamente de acordo com $p(\boldsymbol{\xi})$. Iniciar o conjunto de conexões *C*, $C \subset AxA$, de forma que cada unidade esteja conectada a todas as demais (a rede tem a topologia de um simplex de dimensionalidade *k*).
2. Gerar aleatoriamente um sinal de entrada $\boldsymbol{\xi}$ de acordo com $p(\boldsymbol{\xi})$.
3. Determinar a unidade vencedora *s*
$$s = \arg\min_{c \in A} \left\| \boldsymbol{\xi} - \mathbf{w}_c \right\|$$
4. Atualizar a variável de erro local da unidade vencedora. Conforme o objetivo do algoritmo, isto pode ser feito de diferentes formas (ver discussão na seção 4.1). No caso de visualização, o erro local deve ser simplesmente o contador de sinais para os quais o nó foi vencedor (maximização da entropia).
5. Adaptar os vetores de referência do vencedor e dos vizinhos topológicos diretos (conjunto N_s) por frações ε_b e ε_n, respectivamente, da distância total para o sinal de entrada:
$$\Delta \mathbf{w}_s = \varepsilon_b (\boldsymbol{\xi} - \mathbf{w}_s)$$
$$\Delta \mathbf{w}_i = \varepsilon_n (\boldsymbol{\xi} - \mathbf{w}_i) \quad (\forall i \in N_s)$$
6. Se o número de sinais de entrada gerados até o momento for um múltiplo inteiro de um parâmetro λ, inserir uma nova unidade da seguinte maneira:
   - Determinar a unidade *q* com o maior erro acumulado.

- Adicionar um novo vértice *r*, dividindo a maior aresta incidente a *q*. Seja *f* o vértice ligado por esta aresta; inserir as conexões (*q*,*r*) e (*r*,*f*) e também criar conexões com todos os vizinhos comuns de *q* e *f*, isto é, com todas as unidades de $N_q \cap N_f$.
- Interpolar seu vetor de referência a partir dos vetores de *q* e *f*.

$$\mathbf{w}_r = (\mathbf{w}_q + \mathbf{w}_f)/2.$$

- Diminuir as variáveis de erro de todos os vizinhos de *r* por uma fração que depende do número de vizinhos de *r*:

$$\Delta \mathrm{E}_i = -\frac{\alpha}{|N_r|} E_i \quad (\forall i \in N_r)$$

- Atribuir à variável de erro da unidade *r* a média dos valores dos vizinhos:

$$\mathrm{E}_r = \frac{1}{|N_r|} \sum_{i \in N_r} E_i.$$

7. Diminuir a variável de erro de todas as unidades:

$$\Delta \mathrm{E}_c = -\beta \mathrm{E}_c$$

8. Se um critério de parada (por exemplo: tamanho da rede ou alguma medida de desempenho) não tiver sido alcançado, voltar ao passo 2.

A seguir é apresentada uma tabela com os parâmetros do GCS, seus respectivos significados e valores típicos[44]:

Parâmetro	Descrição	Valores típicos
λ	Intervalo em ciclos entre 2 inserções de nós	200
ε_b	Fator de adaptação do nó vencedor	0.06
ε_n	Fator de adaptação dos nós vizinhos ao venc.	0.002
α	Fator de decremento dos erros na inserção	1.0
β	Fator de depreciação dos erros a cada ciclo	0.0005

Tabela 4: Parâmetros do algoritmo Growing Cell Structures

Observe que, à semelhança do Growing Neural Gas (ver seção 4.4), no passo 6, quando ocorre o crescimento da rede, é necessário identificar, dentre

[44] Neste trabalho se está interessado em malhas de triângulos, assim a dimensionalidade da rede é considerada constante com $k = 2$.

todas as unidades, a de maior erro. Essa pesquisa, junto com a diminuição de todos os erros que ocorre no passo 7, faz o algoritmo ter um desempenho computacional $O(n^2)$.

Uma variação do GCS inclui um passo de remoção de unidades (Fritzke, 1993). Uma unidade é considerada supérflua se sua posição (vetor de referência) estiver em uma região de baixa densidade de probabilidade. Periodicamente, o algoritmo varre o grafo, identificando unidades supérfluas. Para os casos em que $p(\boldsymbol{\xi})$ é desconhecida, Fritzke propõe uma forma de avaliar a probabilidade local da unidade, em função do número de vezes que a unidade foi ativada. Observe que esse não é o caso tratado neste trabalho, já que o atributo de falha é utilizado para gerar explicitamente a densidade de probabilidades. Um novo parâmetro η é utilizado como valor limite de probabilidade; caso a probabilidade local da unidade seja inferior a η, a unidade é removida. No presente trabalho, esse passo de remoção não foi considerado na implementação do GCS.

A seguir, é apresentado o resultado de um teste de utilização do GCS na visualização de superfícies com um dado sísmico sintético. O dado sintético foi gerado pela justaposição de um mesmo traço sísmico real ao longo de um volume de 50x50x50 amostras, de forma a produzir uma falha no centro do volume ao longo de uma das duas direções verticais do volume. A Figura 6-2 exibe duas fatias verticais da amplitude sísmica mapeando os valores de amplitude do mais negativo para o mais positivo em tons das cores azul, branca e vermelha (mapa de cores *flag*). Como atributo de falha, foi gerado o Cubo de Coerência por Auto-estrutura (seção 2.2.2). Na mesma Figura, o atributo de falha é exibido em duas fatias, uma vertical e outra horizontal, em tons de cinza. Enquanto a descontinuidade correspondente à falha tem espessura zero no volume de amplitudes, a espessura é de duas amostras no caso do atributo de falha.

O GCS foi executado para gerar uma estrutura composta apenas por triângulos (k=2). Os valores utilizados para os demais parâmetros são: ε_b=0.06, ε_n=0.002, λ=200, α=1.0, β=0.0005. A Figura 6-3 exibe o grafo gerado por GCS com 100 nós em dois ciclos diferentes de uma mesma execução do algoritmo. Triângulos extraídos dos grafos são exibidos em verde. Ao fundo é exibida, em tons de cinza, uma fatia vertical do atributo de falha, imediatamente anterior à região onde o grafo se dispõe. Observe que, embora a densidade de nós do grafo acompanhe a distribuição de valores do atributo de falhas, dois problemas

são claramente visíveis: ambos os grafos se apresentam “rasgados” nas bordas e o grafo da imagem de baixo possui um defeito na malha triangulada (realçado pelo círculo em vermelho).

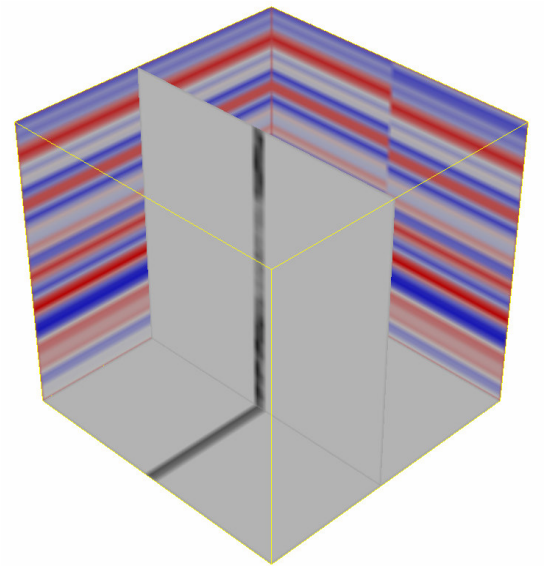

Figura 6-2: Duas fatias verticais da amplitude sísmica de um dado sísmico sintético com o mapa de cores *flag*. O atributo de falha é exibido em duas fatias, uma vertical e outra horizontal, em tons de cinza. A superfície de falha vista pelo atributo gera uma região volumétrica com duas amostras de espessura.

Os rasgos surgem quando um nó da borda é atraído para o interior da estrutura. Se não ocorrerem sinais de entrada que reposicionem o nó de volta na parte externa da estrutura, o nó corre o risco de ficar aprisionado no interior. Essa dinâmica pode ser observada na Figura 6-4. Vale lembrar também que o algoritmo GCS só gera conexões durante a criação de nova unidade. Nenhum esquema do tipo CHL é empregado que permita estabelecer novas conexões entre unidades já existentes. Assim, sinais de entrada obtidos de regiões sem conexões ativam nós ora de um lado ora do outro do rasgo, mas não criam novas conexões[45].

[45] O problema de superfícies abertas com bordas rasgadas em grafos gerados por GCS pode ser também observado nas imagens do trabalho de Bohn (2000); ver figuras 2.4, 2.5 e 2.7 nas páginas 24 e 25.

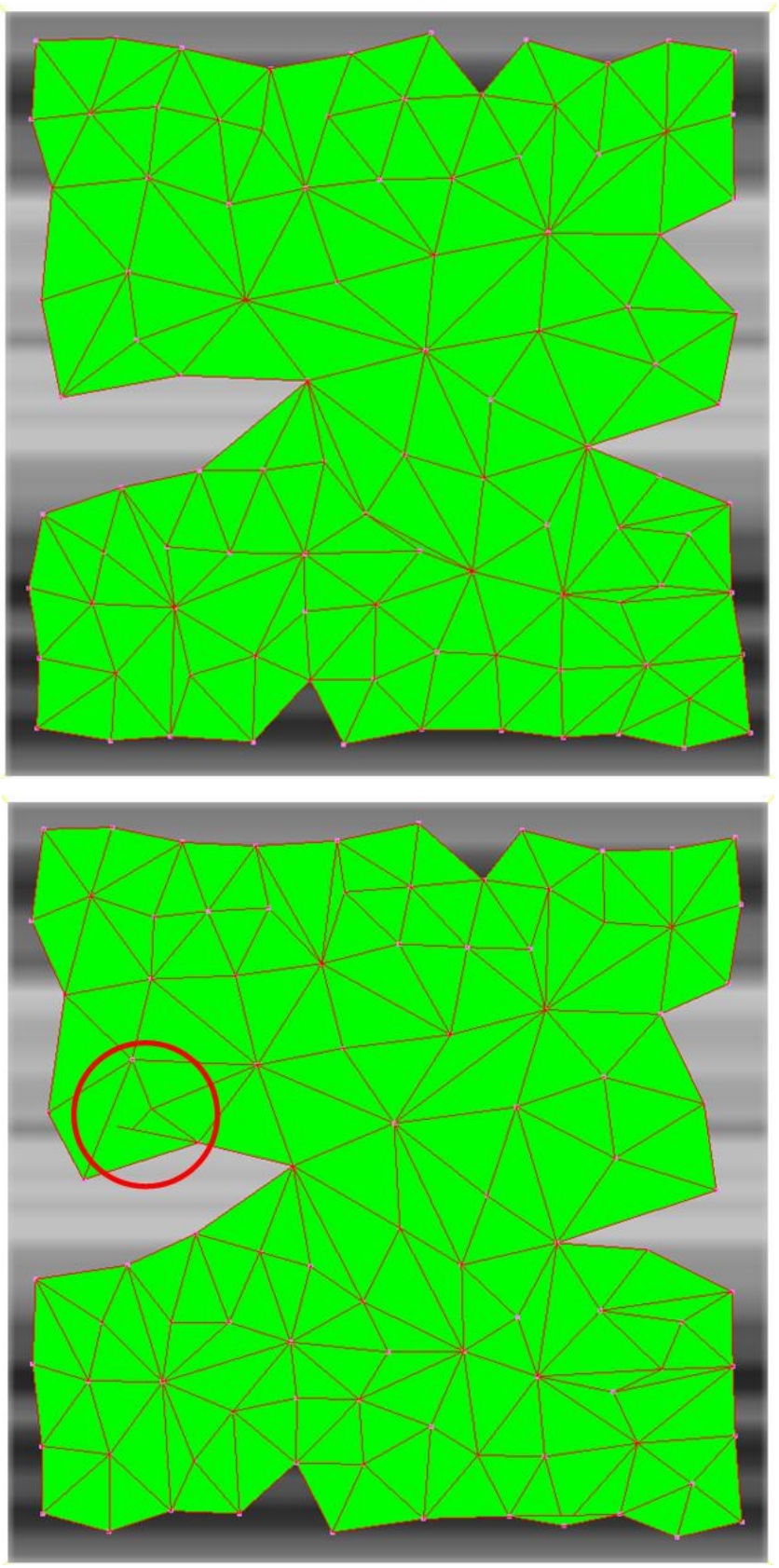

Figura 6-3: Visualização do grafo gerado por GCS com 100 nós em dois ciclos diferentes. Triângulos extraídos dos grafos são exibidos em verde. Os grafos se apresentam rasgados nas bordas. O grafo da imagem de baixo gera uma malha com defeito (círculo vermelho).

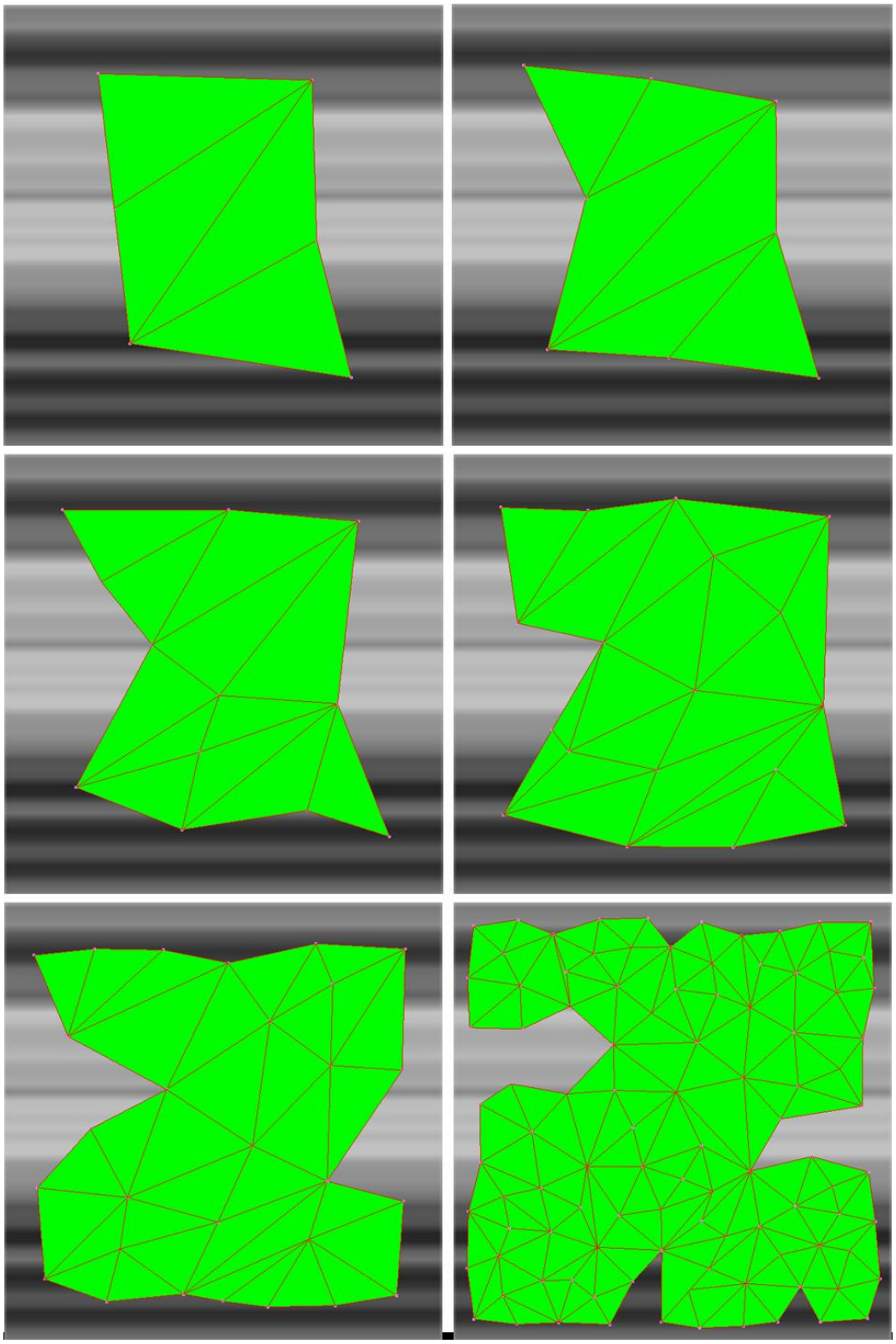

Figura 6-4: Formação de rasgos em superfícies planas geradas pelo GCS.

Voltando à Figura 6-3, o grafo da imagem de baixo apresenta arestas que atravessam triângulos vizinhos, isto é, a superfície gerada pelo grafo não é uma variedade (*non-manifold surface*). Isto se deve à instabilidade da posição dos nós em relação à superfície. Observe que esse tipo de problema ocorreu em

uma situação em que o espaço dos sinais de entrada formava um volume com apenas duas amostras de espessura na direção normal à superfície (ver Figura 6-2). Esse fato remete à necessidade de alteração do mecanismo de adaptação do GCS, de forma a estimular a preservação do caráter de variedade (*manifold*) do grafo em construção. Tal abordagem é explorada pelo algoritmo Neural Meshes. Um aspecto fundamental desse algoritmo é a utilização do método de suavização de malhas pelo Laplaciano, o qual é discutido na seção a seguir.

6.3. Suavização de malhas

Nos últimos anos, vários métodos de suavização de superfícies discretas[46] têm sido apresentados, tendo como modelo a equação da difusão (Pauly, 2003). Taubin foi o pioneiro na introdução desse tipo de método e apresentou várias aproximações de filtros passa-baixa para malhas triangulares, usando diferentes aproximações do operador Laplaciano (Taubin, 1995). Sua abordagem se baseia na generalização das técnicas de processamento de sinais para superfícies que são variedade.

A derivação do método do Laplaciano para suavização de malhas apresentada a seguir se baseia na discussão do problema feita por Hubeli (2002).

No contínuo, uma operação de suavização aplicada a uma função bidimensional $X(u,v)$ é modelada como a solução de um processo de difusão. No processo de difusão, uma função $Y(u,v,t)$ evolui ao longo do tempo de forma a satisfazer à seguinte equação diferencial parcial (EDP):

$$D\left(\frac{\partial^2 Y}{\partial u^2}(u,v,t)+\frac{\partial^2 Y}{\partial v^2}(u,v,t)\right)=\frac{\partial Y}{\partial t}(u,v,t), \qquad (6.1)$$

Onde D é o coficiente de difusão. A equação é definida sobre o domínio $u\in[a,b]$, $v\in[c,d]$, $t\in[0,\infty]$, com a condição inicial (que associa Y com X):

$$Y(u,v,0)=X(u,v)$$

e com as condições de fronteira:

$$Y(a,v,t)=Y(b,v,t)=Y(u,c,t)=Y(u,d,t)=0\,.$$

A equação de difusão pode ser resolvida de duas maneiras: de forma analítica, usando separação de variáveis ($Y(u,v,t)=U(u)V(v)T(t)$) ou pela discretização

[46] Na literatura em inglês, a expressão “suavização de superfícies” é conhecida por *surface fairing*.

do domínio. No segundo caso, o domínio contínuo das variáveis *u*, *v*, *t* é transformado pelo processo de amostragem uniforme (e pela restrição do domínio do tempo como $t \in [0, e]$) em uma grade regular:

$$y_{i,j,k} = Y(u_i, v_j, t_k)$$

$$\begin{aligned} u_i &= a + (b-a) \cdot i / N_u \quad & i &= 0,...,N_u \\ v_j &= c + (d-c) \cdot j / N_v \quad & j &= 0,...,N_v \\ t_k &= e \cdot k / N_t & k &= 0,...,N_t \end{aligned}$$

As condições de contorno são satisfeitas fixando os valores apropriados de amostras:

$$\begin{aligned} y_{i,j,0} &= X(u_i, v_j) \\ y_{0,j,k} &= y_{N_u,j,k} = 0 \\ y_{i,0,k} &= y_{i,N_v,k} = 0 \end{aligned}$$

A discretização da equação de difusão (6.1) facilita a construção de uma fórmula para avaliar os valores das demais amostras desconhecidas. Utilizando a aproximação de diferença à frente para as derivadas de primeira ordem

$$\frac{\partial Y}{\partial u}(u_i, v_j, t_k) = \frac{y_{i+1,j,k} - y_{i,j,k}}{h_u} + O(h_u) \tag{6.2}$$

e de diferenças de segunda ordem divididas para as derivadas de segunda ordem

$$\frac{\partial^2 Y}{\partial u^2}(u_i, v_j, t_k) = \frac{y_{i+1,j,k} - 2y_{i,j,k} + y_{i-1,j,k}}{h_u^2} + O(h_u^2) \tag{6.3}$$

obtém-se:

$$D \cdot \left(\frac{y_{i+1,j,k} - 2y_{i,j,k} + y_{i-1,j,k}}{h_u^2} + \frac{y_{i,j+1,k} - 2y_{i,j,k} + y_{i,j-1,k}}{h_v^2} \right) = \frac{y_{i,j,k+1} - y_{i,j,k}}{h_t} \tag{6.4}$$

$$h_u = \frac{b-a}{N_u}, \quad h_v = \frac{d-c}{N_v}, \quad h_t = \frac{e}{N_t}$$

O valor de $Y(u,v,t)$ na posição $[u_i, v_j, t_{k+1}]$ é então calculado resolvendo a equação 6.2 com respeito a $y_{i,j,k+1}$:

$$y_{i,j,k+1} = y_{i,j,k} + Dh_t \cdot \nabla^2 y_{i,j,k} \tag{6.5}$$

onde ∇^2 corresponde ao Laplaciano discreto:

$$\nabla^2 y_{i,j,k} = \frac{y_{i+1,j,k} - 2y_{i,j,k} + y_{i-1,j,k}}{h_u^2} + \frac{y_{i,j+1,k} - 2y_{i,j,k} + y_{i,j-1,k}}{h_v^2}$$

A suavização de malhas pode ser feita aplicando-se o método discreto apresentado acima para cada uma das componentes do vetor posição dos vértices da malha. Entretanto, essa aplicação não pode ser feita diretamente. O

passo fundamental do processo de construção da solução obtida na Equação 6.5 é a discretização dos operadores de derivada que é feita assumindo-se uma amostragem regular ao longo de cada um dos eixos. Esta hipótese não é válida no caso de malhas, onde tanto o número de vizinhos de cada vértice como a distância entre os vértices vizinhos não têm garantia de serem constantes. Além disso, a parametrização (u, v) deve ser global para toda a malha, o que não é possível em geral. O que se faz neste caso é preservar a solução obtida na Equação 6.5 e fornecer um novo operador para avaliar o Laplaciano $\nabla^2\mathbf{x}_i$ para um ponto $\mathbf{x}_i$ em uma superfície $\mathbf{X}$. Observe que a discretização da derivada parcial em relação ao tempo ainda é válida.

Uma parametrização local (u, v) na vizinhança do ponto $\mathbf{x}_i$ pode ser obtida selecionando-se dois vetores ortogonais no plano tangente de $\mathbf{X}$ em $\mathbf{x}_i$. Neste caso o Laplaciano em $\mathbf{x}_i$ é dado por

$$\nabla^2\mathbf{x}_i = \frac{\partial^2\mathbf{X}}{\partial u^2}(0,0) + \frac{\partial^2\mathbf{X}}{\partial v^2}(0,0) \tag{6.6}$$

já que nesta parametrização a posição de $\mathbf{x}_i$ é (0,0). Essa definição não é satisfatória já que o valor do Laplaciano depende da escolha da parametrização, que foi feita arbitrariamente. Essa limitação é eliminada calculando o Laplaciano como a média (integral) de todas as possíveis parametrizações:

$$\nabla^2\mathbf{x}_i = \frac{2}{\pi}\int_0^{\pi}\frac{\partial^2}{\partial r^2}\mathbf{X}(0,0)d\varphi \qquad (u,v) = r(\cos\varphi,\sin\varphi)\,. \tag{6.7}$$

Observe que, no cálculo acima, o limite de integração, de 0 a π e não de 0 a 2π, leva em conta o fato da aproximação que será utilizada para a derivada segunda ser simétrica em relação ao ângulo φ. Observe também que um dado valor de φ junto com seu par ortogonal, ($\varphi + \pi/2$) módulo π, geram o valor do Laplaciano para uma única parametrização (Equação 6.6); isto explica o fator de normalização como $\pi/2$.

A definição do Laplaciano expressa na Equação 6.7 ainda está baseada no contínuo e deve ser discretizada para poder ser aplicada a malhas triangulares. Isto é feito em dois passos: primeiro, a derivada de segunda ordem é aproximada como na Equação 6.4:

$$\nabla^2\mathbf{x}_i = \frac{2}{\pi}\int_0^{\pi}\frac{\mathbf{X}(h\cos\varphi,h\sin\varphi) - 2\mathbf{X}(0,0) + \mathbf{X}(-h\cos\varphi,-h\sin\varphi)}{h^2} \tag{6.8}$$

No caso discreto, a vizinhança de um ponto $\mathbf{x}_i$ é um conjunto de n triângulos t_k, k=1,...,n formado pelo conjunto de n pontos $\mathbf{x}_{j_k}$, como mostrado na Figura a seguir:

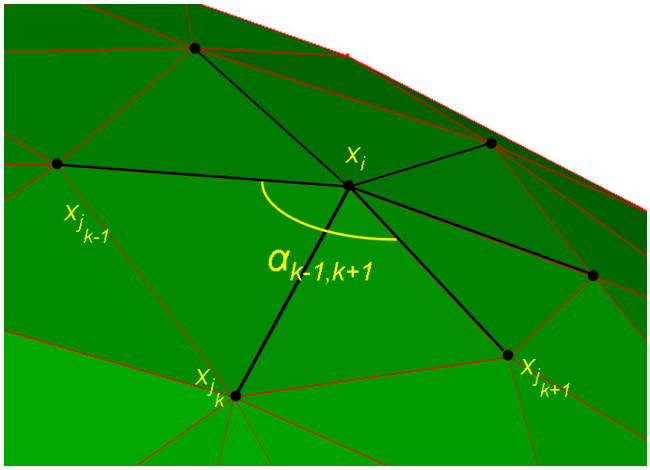

Figura 6-5: Vizinhança de um ponto da malha triangular

A integral da Equação 6.8 pode ser aproximada considerando-se apenas valores do integrando ao longo das arestas disponíveis:

$$\nabla^2 \mathbf{x}_i = \left(\sum_{k=1}^{n} \alpha_{k-1,k+1} \right)^{-1} \sum_{k=1}^{n} \frac{\mathbf{x}_{j_k} - \mathbf{x}_i}{\left\| \mathbf{x}_{j_k} - \mathbf{x}_i \right\|} \cdot \alpha_{k-1,k+1} \tag{6.9}$$

O ângulo $\alpha_{k-1,k+1}$ é definido pelos dois triângulos que compartilham a aresta $[\mathbf{x}_i, \mathbf{x}_{j_k}]$ como visto na Figura 6-5. O coeficiente de normalização foi alterado já que em geral os vértices vizinhos $\mathbf{x}_{j_k}$ de $\mathbf{x}_i$ não são coplanares.

A derivação da Equação 6.9 (Hubeli, 2002) mostra que é razoável utilizar como Laplaciano de um vértice de uma malha triangular uma média ponderada dos tamanhos das arestas incidentes. Na verdade, Taubin (1995) definiu que o Laplaciano de uma superfície discreta é, simplesmente, uma média ponderada, deixando em aberto que tipo de pesos utilizar:

$$\nabla^2 \mathbf{x}_i = \sum_{k=1}^{n} w_{ik} (\mathbf{x}_{j_k} - \mathbf{x}_i) \tag{6.10}$$

$$\sum_{k=1}^{n} w_{ik} = 1$$

Pode-se utilizar, por exemplo, o operador guarda-chuva (*umbrella*) onde $w_{ik} = 1/n, \forall k = 1,...,n, \forall i$, assumindo que todas as arestas incidentes têm tamanhos muito próximos, assim como os ângulos dos triângulos.

Assim, retomando a Equação 6.5 e utilizando a aproximação 6.10 do Laplaciano, tem-se que o processo de suavização de uma malha triangular consiste na alteração de cada vértice **x** para **x'** por:

$$\mathbf{x}' = \mathbf{x} + D \cdot \nabla^2 \mathbf{x} \tag{6.11}$$

O processo pode ser iterado várias vezes. O grau de suavização pode ser controlado pelo parâmetro *D* e pelo número de iterações.

O conceito de suavização de malhas como apresentado aqui pode ser também aplicado a nuvens de pontos, apenas substituindo o conceito de vizinhança em malhas (arestas incidentes) por alguma forma de identificar vizinhos sobre nuvens de pontos (Pauly, 2003).

A Figura 6-6 a seguir mostra o resultado da suavização com *D*=1 e com o Laplaciano aproximado pelo operador guarda-chuva. Em a, é exibida uma superfície enrugada gerada com o GCS a partir de uma distribuição de probabilidades com valores diferentes de zero em apenas 8 fatias verticais do volume. Em b, é exibido o resultado após um passo de suavização; em c, após dois passos e em d, após cinco passos.

Observe que, além da esperada diminuição das irregularidades, ocorre também um encolhimento da superfície. Esse efeito indesejado ocorre tanto em superfícies com bordas, como a da Figura 6-6, como em superfícies fechadas. Para tratar este problema, Taubin (1995) propôs que cada iteração da suavização fosse feita em dois passos: o primeiro, de encolhimento, com valor de *D* positivo, seguido de um passo de ampliação com valor negativo para o parâmetro. No caso de superfícies fechadas, outra forma de amenizar o problema consiste em aplicar a suavização apenas na direção tangencial da malha (Jeong, 2001), isto é, calcular:

$$\nabla_t^2 \mathbf{x} = \nabla^2 \mathbf{x} - \left(\nabla^2 \mathbf{x} \cdot \mathbf{n}\right)\mathbf{n}$$

onde **n** é o vetor normal em **x**. Dessa forma, em uma superfície fechada, os vértices se movem no plano tangente à superfície, evitando encolher na direção da normal. A Figura 6-7 mostra um vértice de um tetraedro, com seu vetor normal **n**, o Laplaciano $\nabla^2 \mathbf{x}$ e a componente tangencial $\nabla_t^2 \mathbf{x}$.

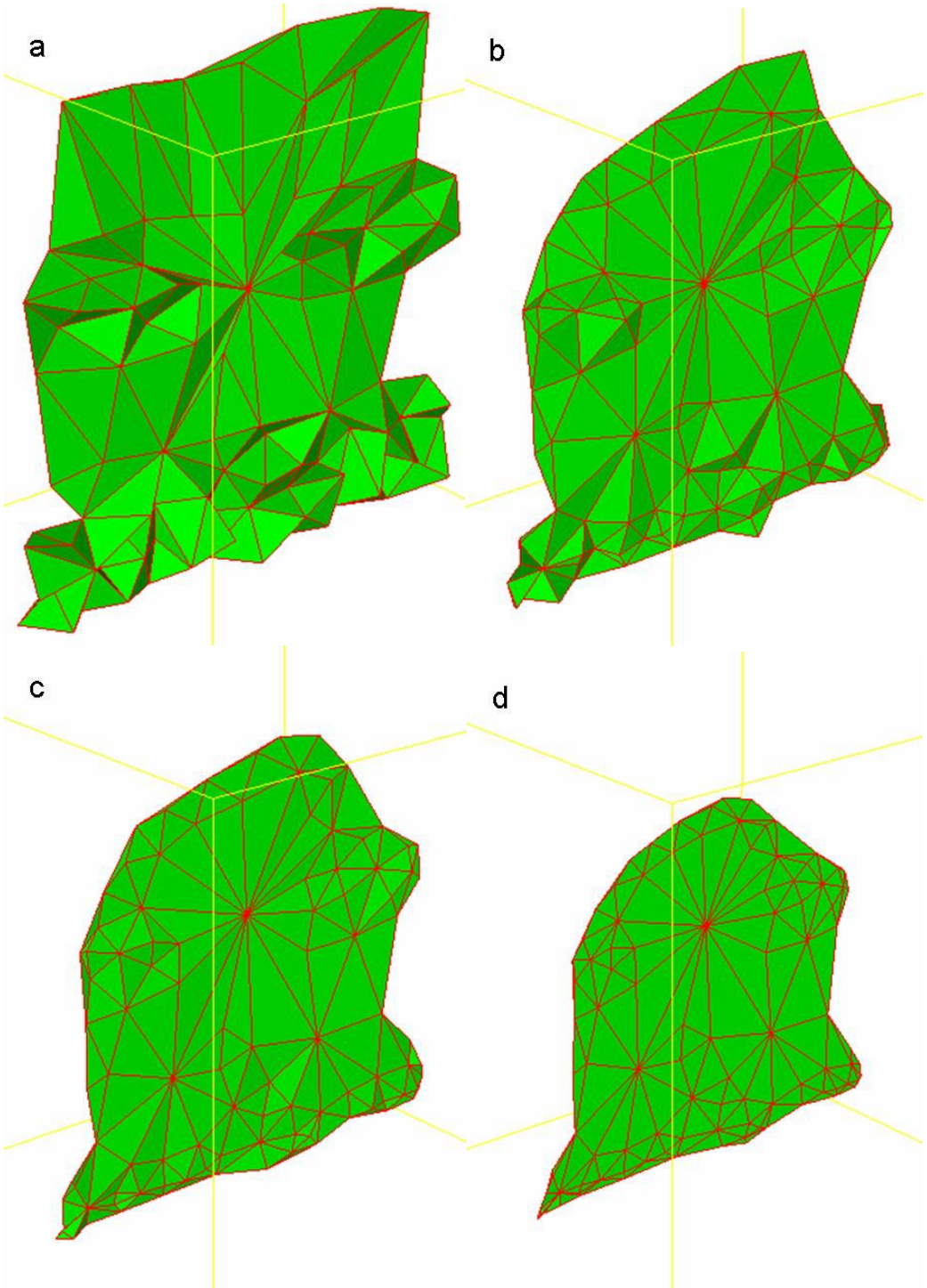

Figura 6-6: Suavização de uma superfície triangular enrugada. a: superfície original. b: após um passo de suavização. c: após dois passos. d: após cinco passos.

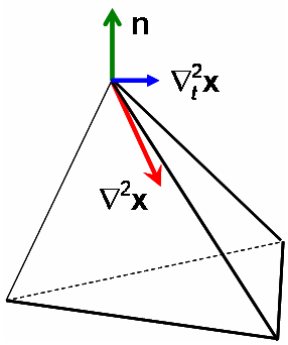

Figura 6-7: Componente tangencial do Laplaciano ($\Delta_T\mathbf{x}$).

6.4. Neural Meshes

O Neural Meshes (NM) foi proposto como um método para reconstrução de superfície a partir de uma nuvem de pontos ou para a geração de uma malha a partir de uma superfície implícita. Neste trabalho ele será apresentado como um método de criação de uma malha triangular a partir de uma distribuição de probabilidades $p(\boldsymbol{\xi})$ que tenha topologia de uma superfície fechada de genus 0 (sem buraco).

O NM possui três grandes diferenças em relação ao Growing Cell Structures (GCS). A primeira é a maneira como os vizinhos de um nó vencedor são adaptados. No NM, o vetor de referência do nó vizinho (ou seja, o vetor posição do vértice vizinho) é atualizado por uma fração da componente tangencial do Laplaciano. Tal ajuste aumenta a qualidade da malha e impede que a superfície fique presa a um mínimo local ou que se dobre sobre si mesma.

As outras diferenças estão na forma como um novo vértice é incluído no grafo e como um vértice inativo é retirado – em particular, como as conexões são alteradas nessas operações. No GCS as conexões são alteradas de forma a manter simplesmente a dimensionalidade da estrutura da rede. Assim, quando um novo vértice *r* é inserido entre os vértices *q* e *f*, são criadas arestas ligando *r* com todos os vizinhos comuns de *q* e *f*. Essa operação é muito simples e fácil de ser implementada para qualquer dimensionalidade da rede. Entretanto, no caso do NM, o grafo é visto como uma malha triangular que representa uma superfície. Assim, a operação de inserção de um novo nó é entendida como uma operação de refinamento da malha, enquanto a remoção corresponde a uma simplificação da malha.

A literatura de Computação Gráfica que trata de algoritmos de refinamento e simplificação de malhas utiliza largamente as operações de **divisão de vértice**

(*vertex split*) e **colapso de aresta** (*edge collapse*) (ver, por exemplo, Hoppe, 1996, 1998). As duas operações, que podem ser vistas como inversas uma da outra, são apresentadas na Figura 6-8. No colapso de aresta, dada uma aresta (v_t, v_s), as duas faces triangulares f_l e f_r que compartilham essa aresta são eliminadas, além de um vértice, v_t. Por outro lado, a divisão de vértice é definida pelo par de **arestas de corte**, na Figura 6-8 (v_l, v_s) e (v_r, v_s), e pelo vértice de divisão, v_s. É criado um novo vértice, v_t, e dois triângulos, f_l e f_r.

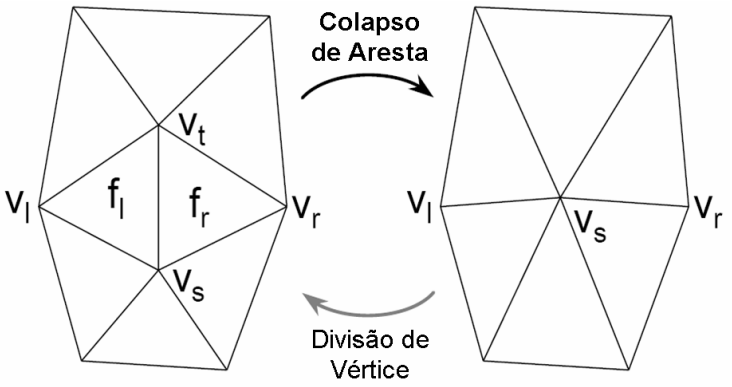

Figura 6-8: Operações de colapso de aresta e divisão de vértice. Adaptada de Hoppe (1998).

Antes de detalhar como as operações de colapso de aresta e divisão de vértice são utilizadas no NM, cabe observar que, como pode ser visto na Figura 6-8, essas operações alteram a posição do vértice v_s. Entretanto, no NM essas operações são utilizadas mantendo inalterada a posição deste vértice. A idéia é que, desta forma, menos triângulos sofrem distúrbios com as operações (Saleem, 2004).

A seguir, é detalhado como a operação de divisão de vértice é utilizada na inclusão de um novo vértice no NM. Como no GCS, o passo de inclusão de um novo nó se inicia identificando o nó v_i de maior contador de sinais e a maior aresta "e" adjacente a v_i. Então, percorrendo o anel dos primeiros vizinhos (*1-ring*) de v_i a partir de "e" nas duas direções igualmente, são identificadas as arestas e_1, e_2 que dividem a estrela de v_i aproximadamente na metade, como exibido na Figura 6-9. As arestas e_1, e_2 funcionam como arestas de corte da operação de divisão de vértice. O novo nó é inserido no meio da aresta "e". O nó v_i perde, para o novo nó, as conexões com os nós dispostos entre e_1, e_2. O novo nó também ganha arestas para gerar os dois novos triângulos, exibidos em cinza no lado direito da Figura 6-9.

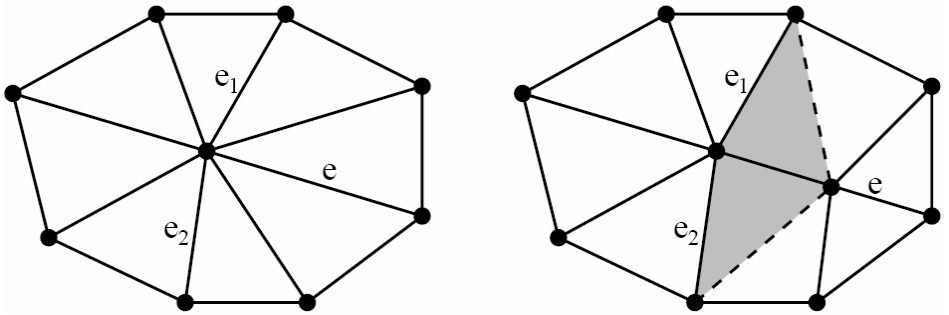

Figura 6-9: À esquerda: um vértice de valência 8 antes do processo de divisão. À direita: após o processo de divisão. O processo distribui as valências de forma mais igualitária possível. (Ivrissimtzis et al., 2003a)

A Figura 6-10 mostra como seria o resultado de uma inserção de um novo nó no GCS para a configuração exibida no lado esquerdo da Figura 6-9. Observe que enquanto, no caso do NM, o nó de maior atividade e o novo nó ficaram com valências iguais (6 para cada um dos dois), no caso do GCS, o nó de maior atividade permanece com valência 8 enquanto o novo nó tem valência 4, gerando um desequilíbrio na malha. Assim, pode-se dizer que a operação de divisão de aresta melhora a conectividade da malha.

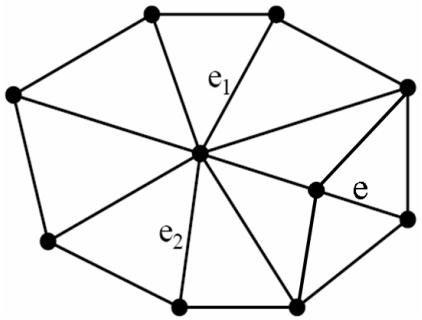

Figura 6-10: Resultado do processo de criação de um novo nó segundo a política do GCS para a situação da esquerda da Figura 6-9. O nó de maior atividade permanece com valência 8 e o novo nó tem valência 4.

No caso de remoção de um nó inativo, a utilização da operação de colapso de aresta requer que se identifique qual das arestas do nó será colapsada. No NM, essa identificação é feita levando em conta como as valências dos nós envolvidos na operação serão alteradas. A Figura 6-11 mostra os nós envolvidos na operação de colapso de uma aresta (a, b) que têm suas valências alteradas. A imagem da esquerda corresponde ao caso em que (a, b) é uma aresta interna; nesse caso, mais dois nós, “c” e “d”, estão envolvidos. Com a eliminação do nó “a” pelo colapso da aresta (a, b), a nova valência do nó “b” será val(a) + val(b) –

4, onde val(x) representa a valência do nó x. Os nós “c” e “d” por sua vez, terão suas valências alteradas para val(c) -1 e val(d) – 1, respectivamente.

O lado direito da Figura corresponde ao caso em que (a, b) é uma aresta de borda; nesse caso, apenas mais um nó, “c”, deve ser considerado. Com a eliminação do nó “a”, pelo colapso da aresta (a, b), a nova valência do nó “b” será val(a) + val(b) – 3 e do nó “c” será val(c) - 1.

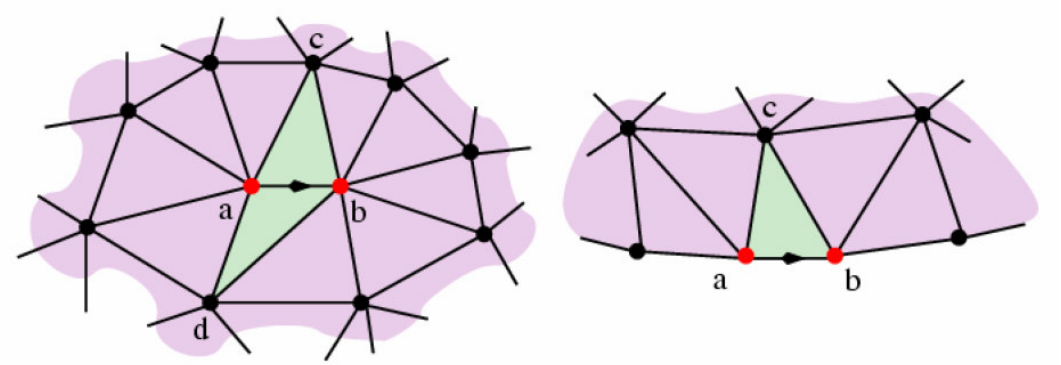

Figura 6-11: Aresta (a, b) sofre o processo de colapso de aresta. À esquerda, (a, b) é uma aresta interna. À direita, (a, b) é uma aresta de borda. (Ivrissimtzis et al., 2003b)

O NM trata a seleção da aresta a ser colapsada de forma a gerar uma configuração mais próxima de uma conectividade de boa qualidade. No caso de um nó interno, é considerada como referência de qualidade uma valência de 6 e, para um nó de borda, uma valência de 4. Assim, NM utiliza no critério de seleção uma medida de distância entre a conectividade gerada e a ideal, isto é, uma medida da irregularidade de conectividade. O algoritmo NM escolhe para colapsar a aresta que minimiza essa medida. No caso de aresta interna (o nó “a” ou o nó “b” é interno), a medida da irregularidade de conectividade é dada por:

$$\frac{1}{3}\sqrt{(\text{val}(a)+val(b)-10)^2+(\text{val}(c)-7)^2+(\text{val}(d)-7)^2} \qquad (6.12)$$

enquanto no caso de uma aresta de borda (“a” e “b” são nós de borda), é dada por:

$$\frac{1}{3}\sqrt{(\text{val}(a)+val(b)-7)^2+(\text{val}(c)-7)^2} \qquad (6.13)$$

Observe que as expressões acima correspondem à distância euclidiana entre o vetor de valências geradas e o vetor de valências ideal; cada componente dos vetores é a valência de cada um dos nós resultantes da operação de colapso de aresta.

Um tipo de situação que, aparentemente, os autores do algoritmo Neural Meshes não levaram em consideração é quando a aresta (a, b) é interna, mas um ou ambos os nós “c” e “d” estão na borda. Também a aresta (a, b) pode ser interna, mas o nó “a” ser um nó de borda. Nessas situações, a expressão 6.12 deve ser modificada.

O passo de remoção de nós do NM é chamado em intervalos de $\mu \cdot n$ ciclos, onde μ é um parâmetro e *n* é o número de unidades da rede no início do intervalo. Se os nós da rede estão distribuídos em bom acordo com a distribuição de probabilidades *p*(**ξ**), então, em $\mu \cdot n$ ciclos, espera-se que um nó qualquer *v* seja vencedor μ vezes em média. O parâmetro μ é um limiar abaixo do qual um nó inativo será considerado mal colocado e será removido.

A seguir, o algoritmo do Neural Meshes é apresentado como proposto por Ivrissimtzis et al. (2003a). Esta versão considera apenas a reconstrução de superfícies fechadas (sem bordas).

Algoritmo NM

A entrada é um volume de probabilidades *p*. A saída é um conjunto de vértices *A*, onde cada vértice tem um peso **w** e um conjunto de conexões *C*.

1. Iniciar conjunto *A* para conter 4 vértices, com os vetores de referência escolhidos aleatoriamente de acordo com *p*(**ξ**). Iniciar o conjunto de conexões *C*, $C \subset AxA$, de forma que cada vértice esteja conectado a todos os demais, formando um tetraedro.
2. Gerar aleatoriamente um sinal de entrada **ξ** de acordo com *p*(**ξ**).
3. Determinar o vértice vencedor *s*

$$s = \arg\min_{c \in A} \left\| \boldsymbol{\xi} - \mathbf{w}_c \right\|$$

4. Incrementar de um o contador de sinais τ do vértice vencedor.
5. Adaptar o vetor de referência do vértice vencedor *s* por uma fração ε_b da distância total para o sinal de entrada:

$$\Delta \mathbf{w}_s = \varepsilon_b (\boldsymbol{\xi} - \mathbf{w}_s) .$$

6. Adaptar os vetores de referência dos vizinhos topológicos diretos do vértice vencedor *s* (conjunto N_s) por uma fração ε_n da componente tangencial do Laplaciano ∇_t^2:

$$\Delta \mathbf{w}_i = \varepsilon_n \nabla_t^2 (\mathbf{w}_i) \quad (\forall i \in N_s)$$

7. Se o número de sinais de entrada gerados até o momento for um múltiplo inteiro de um parâmetro λ, inserir um novo vértice da seguinte maneira:
   - Determinar o vértice *q* com o maior valor de contador de sinais.
   - Identificar a maior aresta *e* incidente a *q*. Seja *f* o vértice ligado a *q* por *e*.
   - Varrer o anel de *q* em ambas as direções a partir da aresta *e*, identificando as duas arestas e_1, e_2 que dividem a estrela de *q* aproximadamente na metade.
   - Executar a operação de divisão de vértice (*vertex split*) para *q* tendo e_1, e_2, como arestas de corte, e posicionando o novo vértice *r* no meio de *e*:

$$\mathbf{w}_r = (\mathbf{w}_q + \mathbf{w}_f)/2 .$$

   - O valor original do contador de sinais de *q* é dividido entre os vértices *r* e *q* proporcionalmente às suas regiões de Voronoi. O volume da região de Voronoi de um vértice é aproximado pelo quadrado do comprimento médio de suas arestas adjacentes.
8. A cada intervalo de $\mu \cdot n$ ciclos, onde μ é um parâmetro e *n* é o número de unidades da rede no início do intervalo, remover os vértices que não foram ativados durante este intervalo. A remoção de um vértice *a* é feita pelo colapso de uma de suas arestas. O outro vértice *b* da aresta a ser removida é identificado da seguinte maneira:
   - Determinar o vértice *b* que tenha dois vértices adjacentes *c*, *d* também adjacentes ao vértice *a* (formando dois triângulos) e que minimize a seguinte medida de irregularidade de conectividade:

$$\frac{1}{3}\sqrt{(\text{val}(a) + val(b) - 10)^2 + (\text{val}(c) - 7)^2 + (\text{val}(d) - 7)^2}$$

9. Diminuir o contador de sinais de todas as unidades:

$$\Delta\tau_c = -\alpha\tau_c$$

10. Se um critério de parada (por exemplo: tamanho da rede ou alguma medida de desempenho) não tiver sido alcançada, voltar ao passo 2.

A seguir é apresentada uma tabela com os parâmetros do NM, seus respectivos significados e valores típicos:

Parâmetro	Descrição	Valores típicos
λ	Intervalo em ciclos entre 2 inserções de nós	200
ε_b	Fator de adaptação do nó vencedor	0.06
ε_n	Fator de suavização pelo Laplaciano tang.	0.002
α	Fator de depreciação dos τ's a cada ciclo	0.0005
μ	Fator de intervalo entre remoção de nós	10

Tabela 5: Parâmetros do algoritmo Neural Meshes

Novamente, a pesquisa do vértice de maior contador de sinais (passo 7) junto com a diminuição dos contadores de sinais de todos os vértices acarreta um desempenho computacional $O(n^2)$ para o NM. Uma novidade em relação ao GNG e ao GCS é a inclusão de um passo de remoção de vértices inativos (passo 8), onde se varrem todos os nós da rede. Entretanto, esse passo é executado $O(\log n)$ vezes (em intervalos de $\mu \cdot n$ ciclos), gerando uma contribuição total de $O(n \log n)$.

Na Figura 6-12 e Figura 6-13 são exibidas malhas geradas por NM para uma esfera. A função de probabilidades foi definida a partir da função implícita da esfera, amostrada em uma grade regular tridimensional de 100 elementos em cada direção.

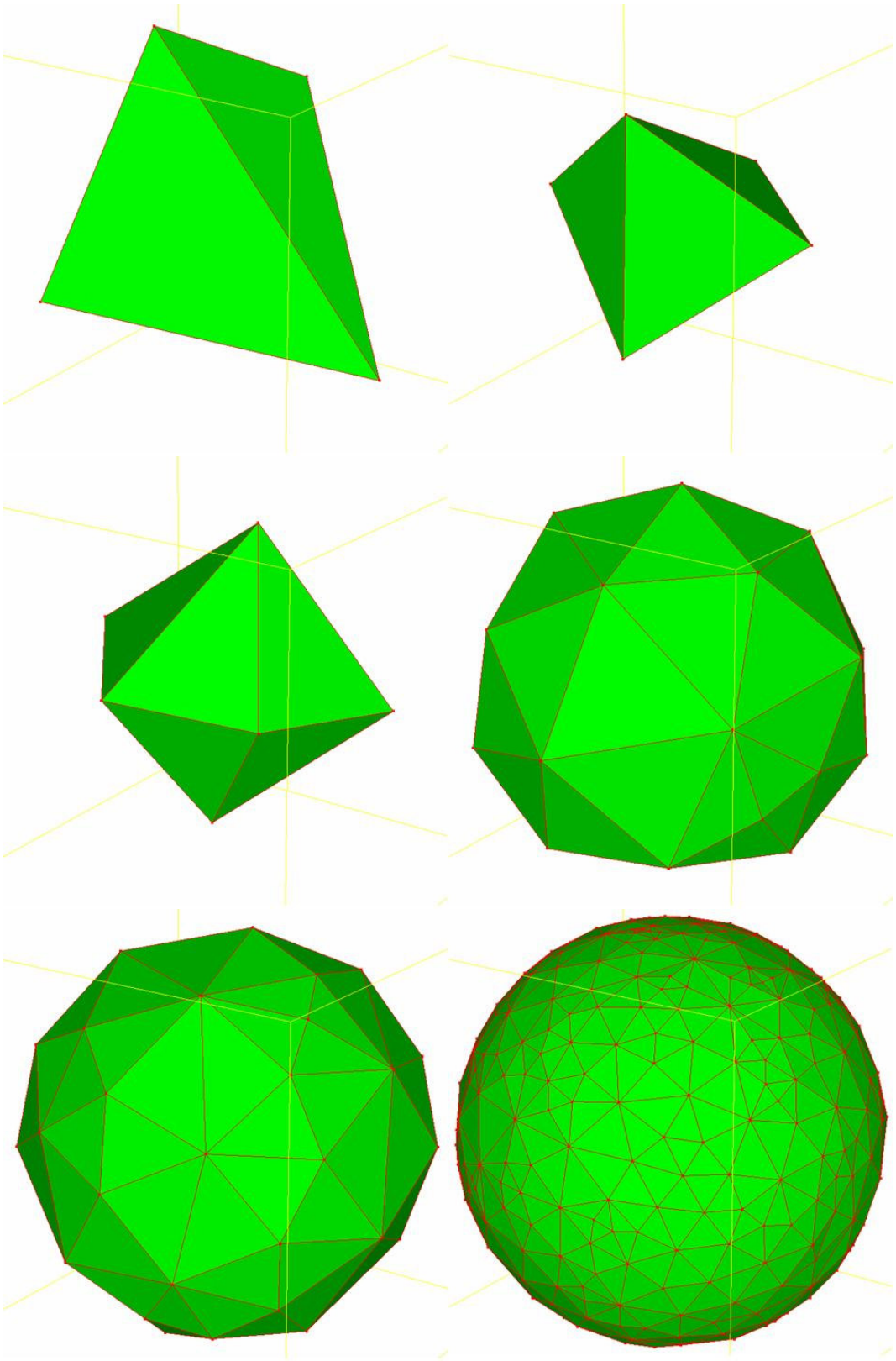

Figura 6-12: Esfera gerada por NM. Da esquerda para a direita e de cima para baixo: tetraedro inicial, malha com 5, 7, 27, 49 e 351 nós.

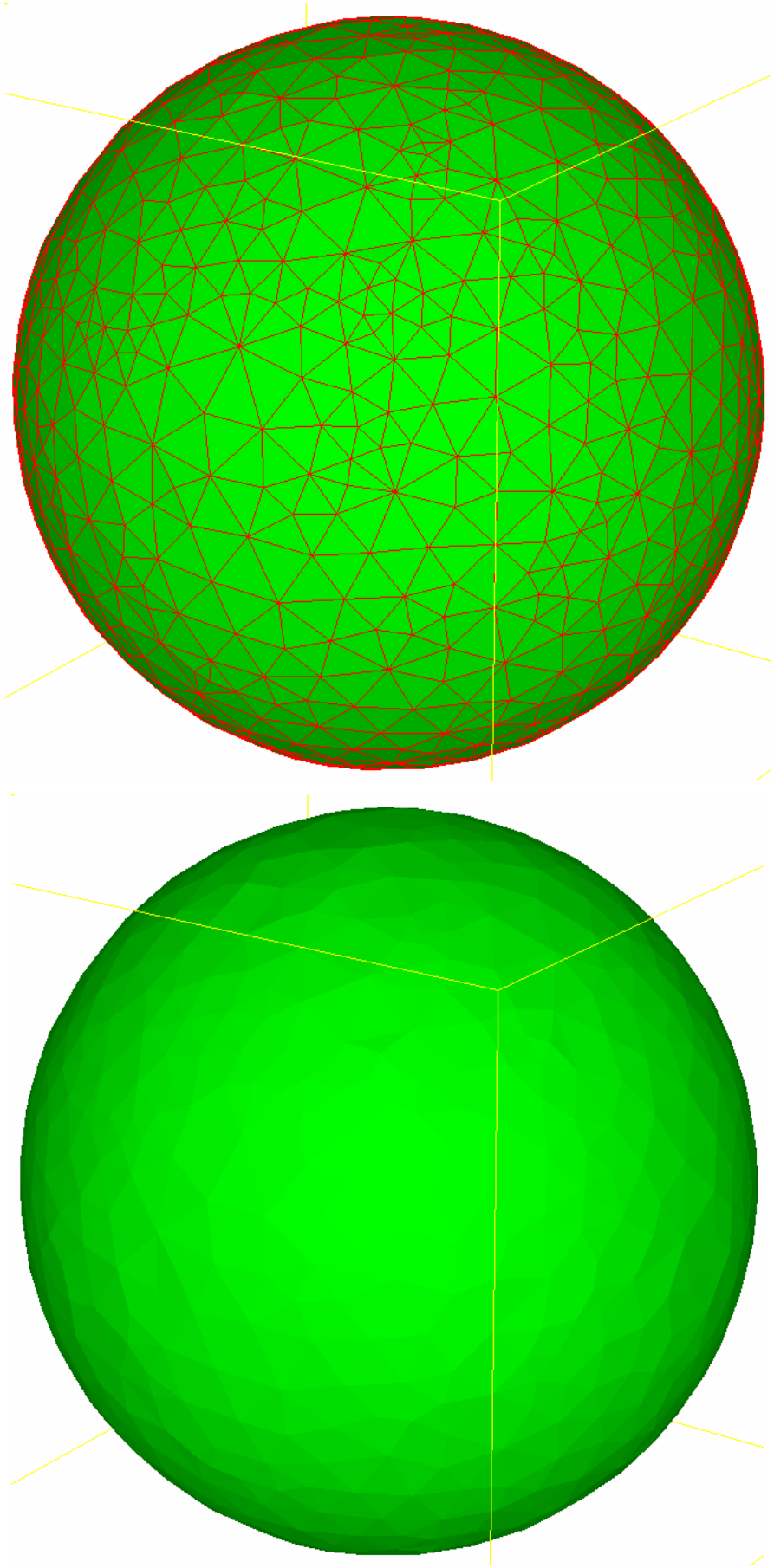

Figura 6-13: Esfera gerada por NM com 1000 nós.

Foram propostas também algumas variações sobre esse algoritmo. Jeong et al. (2003) propuseram a inclusão de um contador de normais para cada vértice da rede. Tal contador mantém uma medida da variação do vetor normal do vértice. A utilização da política de dividir os vértices com maior atividade da normal leva a uma reconstrução da superfície com nível de detalhe adaptativo. Isto é, as áreas com maior curvatura são mais densamente representadas enquanto que as áreas mais suaves ou planas têm menos detalhes.

Ivrissimtzis et al. (2003b) incluíram um passo de remoção de triângulos que possuem área maior do que a média (o quão maior definido por parâmetro). Esse passo pode criar bordas na superfície em construção. A topologia da superfície fechada originalmente de *genus* 0 pode ser alterada, gerando por exemplo um toro. Também é proposto um passo de fusão de bordas muito próximas, permitindo criar alças.

Ivrissimtzis et al. (2004a) propuseram ainda uma filtragem dos sinais de entrada de forma a tornar o algoritmo mais robusto com relação a "pontos fora da curva" (*outlier noise*). Na atualização da posição do nó vencedor é considerado o valor da distância entre o vencedor e a amostra. Se a distância for muito grande em relação à média das distâncias, então seu valor sofre uma diminuição, de forma a impedir um grande deslocamento do nó.

Em Ivrissimtzis et al. (2004b) é utilizada a técnica estatística de *ensemble*. Esta técnica se baseia na idéia de se fazer uma média sobre um conjunto de diferentes saídas de um algoritmo. No caso do NM, o objetivo do uso de *ensemble* é evitar a convergência para um mínimo local, isto é, capturar corretamente a forma global dos dados de entrada. Assim, o NM é executado várias vezes de forma a criar um conjunto contendo vários modelos grosseiros (malhas com número de vértices inferior ao desejado) para a mesma função de probabilidades. Cada modelo é voxelizado, isto é, transformado em um vetor tridimensional de voxels, todos utilizando uma mesma grade. Cada voxel tem um valor 1 ou 0, conforme pertença ou não ao modelo. Um modelo médio é obtido com os valores mais votados de cada voxel. Uma malha média grosseira é obtida com o uso do algoritmo de Cubos Marchantes (*Marching Cubes*) (Lorensen et al., 1987). Por fim, uma nova instância do NM é executada, tendo essa malha como grafo inicial.

Por fim, Saleem (2004) propõe uma alteração no NM de forma a diminuir seu custo computacional para $O(n \log n)$. O esquema de atribuir a cada vértice um contador de sinais que sofre uma depreciação a cada ciclo é substituído por um esquema em que os vértices são apenas dispostos em uma fila. Toda vez

que um nó é vencedor, ele executa um salto de um número fixo de nós a frente na fila. No passo de adição de um novo nó é necessário apenas identificar o vértice no início da fila. Nas palavras do autor, a abordagem de “aprendizado exato” do algoritmo de Neural Meshes original é substituída pela de “aprendizado comparativo”.

6.5. Malhas Neurais Abertas

O algoritmo de Neural Meshes tem o grande mérito de combinar a estratégia de aprendizado competitivo evolutivo do Growing Cell Structures com operações de suavização, refinamento e simplificação válidas para malhas triangulares. A suavização feita com o uso da componente tangencial do Laplaciano aumenta a qualidade da malha e age no sentido de impedir que a superfície fique presa a um mínimo local ou que se dobre sobre si mesma. As operações de divisão de vértice e colapso de aresta possibilitam uma boa estrutura de conectividade para a malha. O processo evolutivo do GCS, por sua vez, permite capturar os aspectos importantes da geometria dos dados de entrada.

Entretanto, considerando a estratégia de visualização de falhas proposta neste trabalho, a utilização do NM não é possível. Os volumes de probabilidades para cada falha formam superfícies abertas, enquanto o NM só aceita sinais de entrada que tenham topologia de superfície fechada. Além disso, o nível de irregularidades presente no volume de probabilidades exige uma suavização mais agressiva. A Figura 6-14 mostra um detalhe de fatias do volume de probabilidades associado com uma superfície de falha. Observe as irregularidades na direção da normal à superfície. Nesse caso, é necessário que a suavização tenha uma atuação também na direção da normal.

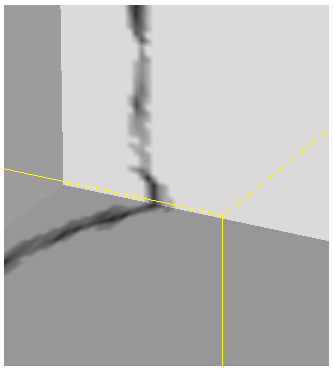

Figura 6-14: Detalhe de fatias do volume de probabilidades para uma superfície de falha.

Neste trabalho é proposto um novo algoritmo que será chamado de Malhas Neurais Abertas (*Open Neural Meshes*). Malhas Neurais Abertas (MNA) derivam do NM com as seguintes alterações:

- O grafo é iniciado contendo apenas um triângulo.
- Movimentos de adaptação de vértices da borda são restritos.
- Vértices da borda que são vizinhos do nó vencedor sofrem um processo de suavização com o uso da componente normal do Laplaciano.
- Vértices internos que são vizinhos do nó vencedor são suavizados pelo Laplaciano, tanto na direção tangencial como na direção da normal.

O algoritmo MNA assume que os dados de entrada têm uma topologia de superfície aberta, sem buracos. Assim, o processo de reconstrução começa com a criação de um único triângulo.

O ponto fundamental do MNA é a restrição dos movimentos dos vértices da borda, quando são nós vencedores. Essa restrição é feita de forma a impedir que a área da superfície reconstruída diminua por um movimento de adaptação de um vértice da borda. Isto evita a ocorrência de rasgos como os gerados pelo Growing Cell Structures. Outro efeito dessa restrição é tornar possível, na adaptação dos vértices internos, a utilização de todas as componentes do Laplaciano, sem o risco de encolhimento da superfície reconstruída. Como comentado anteriormente, isto é importante quando se trabalha com dados pouco suaves como atributos de falha de dados sísmicos.

A Figura 6-15 mostra a geometria envolvida no processo de adaptação de um nó da borda *s*. Nas duas imagens da Figura, *a* e *b* são os nós vizinhos da borda, enquanto *i* é um nó interno. O plano α é o plano tangente ao grafo na vizinhança do nó *s*. O lado esquerdo da Figura corresponde ao caso em que as arestas da borda formam localmente um polígono convexo, enquanto o lado direito corresponde ao caso em que as arestas da borda geram localmente uma concavidade. A heurística proposta neste trabalho identifica o plano π que separa o espaço em torno do vértice *s* em duas regiões: a interna ao grafo, para a qual não é permitido qualquer movimento de *s*, e a externa, para a qual os movimentos são permitidos. O plano π passa pelo vértice *s,* é paralelo ao segmento de linha que liga os vértices *a*, *b* e é perpendicular ao plano α. Tendo

calculado o vetor normal ao plano $\mathbf{n}_\pi$ e dado um sinal de entrada $\boldsymbol{\xi}$, a regra de adaptação do vetor de referência $\Delta\mathbf{w}_s$ de um nó *s* da borda é reescrita como:

$$\mathbf{d}_s = \varepsilon_b(\boldsymbol{\xi} - \mathbf{w}_s)$$
$$\Delta\mathbf{w}_s = \mathbf{d}_s, \; se\, \mathbf{d}_s \cdot \mathbf{n}_\pi \geq 0$$
$$\Delta\mathbf{w}_s = 0, \quad se\, \mathbf{d}_s \cdot \mathbf{n}_\pi < 0$$

Isto é, o movimento só é possível se o vetor deslocamento aponta para a região externa à malha.

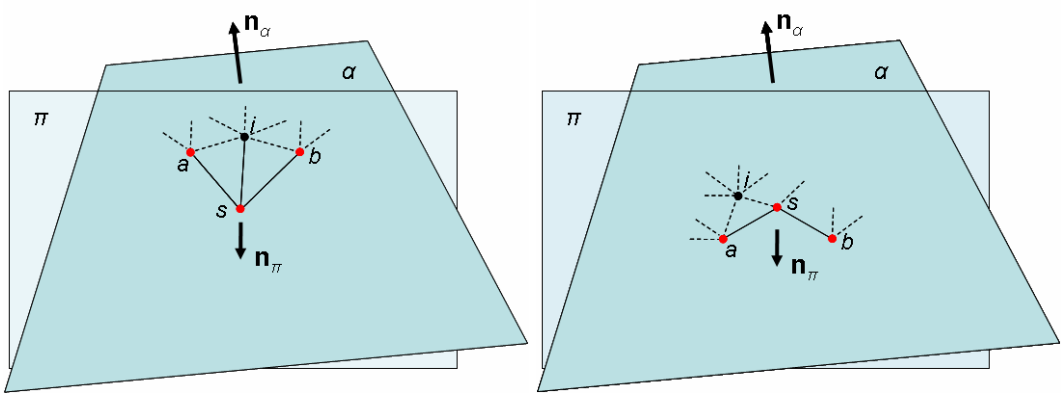

Figura 6-15: Geometria do processo de adaptação de um nó da borda (*s*). Do lado esquerdo, situação em que o nó forma uma vizinhança convexa com os nós vizinhos da borda *a*, *b*. Do lado direito, o nó *s* forma uma concavidade com seus vizinhos.

O Laplaciano calculado para uma aresta da borda gera, em geral, uma componente extremamente forte na direção tangente, no sentido do interior. Assim, para superfícies abertas de pequena curvatura, é a componente tangencial, e não a componente normal, que leva ao encolhimento da superfície em construção. O que se propõe neste trabalho, então, é a utilização apenas da componente normal. Entretanto, um vértice de borda tem naturalmente uma conectividade deficiente o que torna a avaliação do vetor normal nesse tipo de vértice muito pouco confiável. A solução do MNA é aplicar a suavização de um nó de borda apenas nos casos em que este é vizinho de um nó vencedor interno com valência maior ou igual a 5; a normal utilizada para calcular a componente do Laplaciano do vértice de borda é a normal do vértice interno. Essa escolha torna o processo de suavização dos nós de borda mais confiável.

A suavização dos nós internos é feita de forma independente para a componente tangencial e a normal do Laplaciano. Isto é, na adaptação dos nós vizinhos ao vencedor são utilizados fatores diferentes para cada uma das componentes do Laplaciano. Tal decisão está relacionada com o fato de que as duas componentes desempenham papéis diferentes no MNA. A componente

tangencial age no sentido de preservar a qualidade da malha enquanto a componente normal age no sentido de manter a superfície localmente com baixa curvatura.

A seguir são apresentados o algoritmo de cálculo da normal do plano de borda e o algoritmo de Malhas Neurais Abertas.

Algoritmo Normal do Plano de Borda

A entrada é um grafo G e um nó de borda *s* pertencente a G.
A saída é um vetor $\mathbf{n}_\pi$.

1. Identificar o vértice interno *i* conectado a *s*, de valência mais próxima de 6. Em caso de empate, escolher o de maior distância.
   1.1. Em caso de sucesso, calcular o vetor normal $\mathbf{n}_\alpha$ ao vértice *i*.
   1.2. Caso não haja vértice interno, criar *i* como um vértice auxiliar posicionado no baricentro do grafo. Calcular a normal do grafo $\mathbf{n}_\alpha$ usando o método de autovetores da matriz de covariância.
2. Identificar os vértices da borda *a*, *b* conectados a *s*.
3. Calcular o vetor normal do plano de borda como $\mathbf{n}_\pi = ab \times \mathbf{n}_\alpha$ (produto vetorial do vetor *ab* com a normal do plano tangente).
4. Identificar se o ângulo interno *ang* formado pelo vértice *s* com *a*, *b* é agudo ou obtuso. O ângulo é agudo se $((as \times ab) \cdot (as \times ai)) > 0$.
5. Corrigir o sentido de $\mathbf{n}_\pi$:
   5.1. Se vértices *a*, *s*, *b* são aproximadamente colineares ou *ang* é agudo então
      5.1.1. Se $si \cdot \mathbf{n}_\pi > 0$ então inverter o sentido de $\mathbf{n}_\pi$ (*si* é confiável para o teste).
   5.2. Senão
      5.2.1. Se $as \cdot \mathbf{n}_\pi > 0$ então inverter o sentido de $\mathbf{n}_\pi$.

Algoritmo MNA

A entrada é um volume de probabilidades *p* com topologia de uma superfície aberta. A saída é um conjunto de vértices *A*, onde cada vértice tem um peso **w** e um conjunto de conexões *C*.

1. Iniciar o conjunto *A* para conter 3 vértices, com os vetores de referência escolhidos aleatoriamente de acordo com *p*(**ξ**). Iniciar o conjunto de

conexões *C*, $C \subset AxA$, de forma que cada vértice esteja conectado a todos os demais, formando um triângulo.

2. Gerar aleatoriamente um sinal de entrada **ξ** de acordo com *p*(**ξ**).
3. Determinar o vértice vencedor *s*

$$s = \arg\min_{c \in A} \left\| \boldsymbol{\xi} - \mathbf{w}_c \right\|$$

4. Incrementar de um o contador de sinais τ do vértice vencedor.
5. Adaptar o vetor de referência do vértice vencedor *s*.

   5.1. Se o vencedor *s* for um vértice interno então adaptar por uma fração ε_b da distância total para o sinal de entrada:

$$\Delta \mathbf{w}_s = \varepsilon_b (\boldsymbol{\xi} - \mathbf{w}_s) .$$

   5.2. Se *s* for um nó da borda então calcular a normal do plano de borda $\mathbf{n}_\pi$ e adaptar o vetor de referência do vencedor como:

$$\begin{aligned} \mathbf{d}_s &= \varepsilon_b (\boldsymbol{\xi} - \mathbf{w}_s) \\ \Delta \mathbf{w}_s &= \mathbf{d}_s, \ se\ \mathbf{d}_s \cdot \mathbf{n}_\pi \geq 0 \ . \\ \Delta \mathbf{w}_s &= 0, \quad se\ \mathbf{d}_s \cdot \mathbf{n}_\pi < 0 \end{aligned}$$

6. Adaptar os vetores de referência dos vizinhos topológicos diretos do vértice vencedor *s* (conjunto N_s).

   6.1. Se o vértice vizinho *j* for um nó interno então adaptar por uma fração ε_{nit} da componente tangencial do Laplaciano, $\nabla^2{}_t$, e por uma fração ε_{nit} da componente normal do Laplaciano, $\nabla^2{}_n$:

$$\Delta \mathbf{w}_j = \varepsilon_{nit} \nabla^2{}_t (\mathbf{w}_i) + \varepsilon_{nin} \nabla^2{}_n (\mathbf{w}_i) .$$

   6.2. Se o vértice vizinho *j* for um nó da borda então

      6.2.1. Se *s* for um nó interno com valência maior ou igual a 5, então calcular o vetor normal em *s*, $\mathbf{n}_s$, e adaptar o vértice *j* por uma fração ε_{nb} da componente na direção da normal $\mathbf{n}_s$ do Laplaciano ∇^2_n:

$$\Delta \mathbf{w}_j = \varepsilon_{nb} \nabla^2_n (\mathbf{w}_i) .$$

7. Se o número de sinais de entrada gerados até o momento for um múltiplo inteiro de um parâmetro λ, inserir um novo vértice da seguinte maneira:
   - Determinar o vértice *q* com o maior valor de contador de sinais.
   - Identificar a maior aresta *e* incidente a *q*. Seja *f* o vértice ligado a *q* por *e*.
   - Varrer o anel de *q* em ambas as direções a partir da aresta *e*, identificando as duas arestas e_1, e_2 que dividem a estrela de *q* aproximadamente na metade.

- Executar a operação de divisão de vértice (*vertex split*) para *q* tendo e_1, e_2, como arestas de corte, e posicionando o novo vértice *r* no meio de *e*:

$$\mathbf{w}_r = (\mathbf{w}_q + \mathbf{w}_f)/2 .$$

- O valor original do contador de sinais de *q* é dividido entre os vértices *r* e *q*, proporcionalmente às suas regiões de Voronoi. O volume da região de Voronoi de um vértice é aproximado pelo quadrado do comprimento médio de suas arestas adjacentes.

8. A cada intervalo de $\mu \cdot n$ ciclos, onde μ é um parâmetro e *n* é o número de unidades da rede no início do intervalo, remover os vértices que não foram ativados durante esse intervalo. A remoção de um vértice *a* é feita pelo colapso de uma de suas arestas. O segundo vértice *b* da aresta a ser removida é escolhido de forma a minimizar a seguinte medida de erro:
   - Caso *ab* seja uma aresta interna: determinar os dois vértices *c*, *d* adjacentes simultaneamente aos vértices *a*, b (formando dois triângulos) e calcular

$$\frac{1}{3}\sqrt{(\text{val}(a)+\text{val}(b)-4-\text{id}(a))^2+(\text{val}(c)-1-\text{id}(c))^2+(\text{val}(d)-1-\text{id}(d))^2}$$

     onde

$$\text{id}(x)=\begin{cases}6, \text{caso } x \text{ nó interno}\\ 4, \text{caso } x \text{ nó de borda}\end{cases}$$

   - Caso *ab* seja uma aresta de borda: determinar o vértice *c* adjacente simultaneamente aos vértices *a*, b (formando um triângulo) e calcular

$$\frac{1}{3}\sqrt{(\text{val}(a)+\text{val}(b)-7)^2+(\text{val}(c)-1-\text{id}(c))^2}$$

9. Diminuir o contador de sinais de todas as unidades:

$$\Delta\tau_c = -\alpha\tau_c$$

10. Se um critério de parada (por exemplo: tamanho da rede ou alguma medida de desempenho) não tiver sido alcançada, voltar ao passo 2.

A seguir é apresentada uma tabela com os parâmetros do MNA, seus respectivos significados e valores típicos:

Parâmetro	Descrição	Valores típicos
λ	Intervalo em ciclos entre 2 inserções de nós	200
ε_b	Fator de adaptação do nó vencedor	0.06
ε_{nit}	Fator suav. nó interno pelo Laplaciano tang.	0.09
ε_{nin}	Fator suav. nó interno pelo Laplaciano normal	0.05
ε_{nb}	Fator suav. nó borda pelo Laplaciano normal	0.002
α	Fator de depreciação dos τ's a cada ciclo	0.0005
μ	Fator de intervalo entre remoção de nós	10

Tabela 6: Parâmetros do algoritmo de Malhas Neurais Abertas.

Por fim, um comentário sobre a implementação e o funcionamento do algoritmo de MNA. Como o Neural Meshes, o algoritmo de Malhas Neurais Abertas gera grafos cuja configuração em um determinado instante pode não corresponder a uma variedade (por exemplo, com a ocorrência de uma aresta que perfura um triângulo, como na Figura 6-3). As operações de suavização e colapso de arestas inativas tendem a corrigir defeitos eventualmente existentes. Entretanto, existem situações em que esses mecanismos não são suficientes para recuperar a estrutura danificada. A função que identifica o conjunto ordenado de triângulos incidentes a um determinado vértice (ver Figura 6-5) tem um papel central no código do MNA. Essa função é invocada durante a divisão de vértice (inclusão de novo vértice), no cálculo do vetor normal de um vértice[47],[48] e, após a remoção de vértice inativo, na identificação do tipo, se interno ou de borda, do nó sobrevivente do colapso de aresta. No código construído para MNA, nos casos em que essa função é incapaz de identificar a lista de triângulos incidentes, o programa reinicia o algoritmo. Isto é, a função funciona como um filtro para identificar situações irrecuperáveis. Nos testes executados neste trabalho, situações de reinício do algoritmo se mostraram raras.

[47] A normal de um vértice é calculada, por exemplo, no passo 1.1 do algoritmo Normal do Plano de Borda e no passo 6 do MNA.

[48] A normal de um vértice *v* é obtida, tomando-se o conjunto de vetores normais dos triângulos da estrela do vértice *v* e calculando a média ponderada pela área de cada triângulo.

6.6. Resultados

Nesta seção, é apresentada uma série de resultados que mostram o comportamento do algoritmo MNA na reconstrução de superfícies abertas definidas a partir de dados sintéticos e reais.

A Figura 6-16 mostra um teste da evolução dos nós de borda para grafos gerados por MNA. O dado utilizado é o mesmo dado sintético do teste feito com o GCS (ver Figura 6-2). Da esquerda para a direita, de cima para baixo, são exibidos o grafo (triângulo) inicial, grafo inicial após 196 ciclos, e grafos com 4, 5, 20 e 100 vértices. Observe que os problemas de rasgamento e defeitos de topologia apresentados na Figura 6-3 não ocorrem no teste com MNA.

Além do dado sintético utilizado na Figura 6-16, foi também modelado outro dado sintético a partir de uma função implícita que define uma superfície ondulada. Foi construído um volume de tamanho 256x256x256, tal que todas as suas amostras tenham valor zero, com exceção daquelas interceptadas por uma superfície definida pelo produto de duas funções seno. Este volume, que será chamado de "ondulada1", define uma superfície com apenas um voxel de espessura. O volume é utilizado como volume de probabilidades para o algoritmo MNA. A Figura 6-17 mostra a evolução da malha sobreposta a duas fatias desse volume de probabilidades. Observe novamente a inexistência de problemas de rasgamento ou defeitos de topologia. MNA foi executado com:

$\varepsilon_b = 0.06$, $\varepsilon_{nit} = 0.03$ $\varepsilon_{nin} = 0.03$, $\varepsilon_{nb} = 0.003$, $\lambda = 300$, $\mu = 20$, $\alpha = 0.05$,

gerando uma malha de 300 nós.

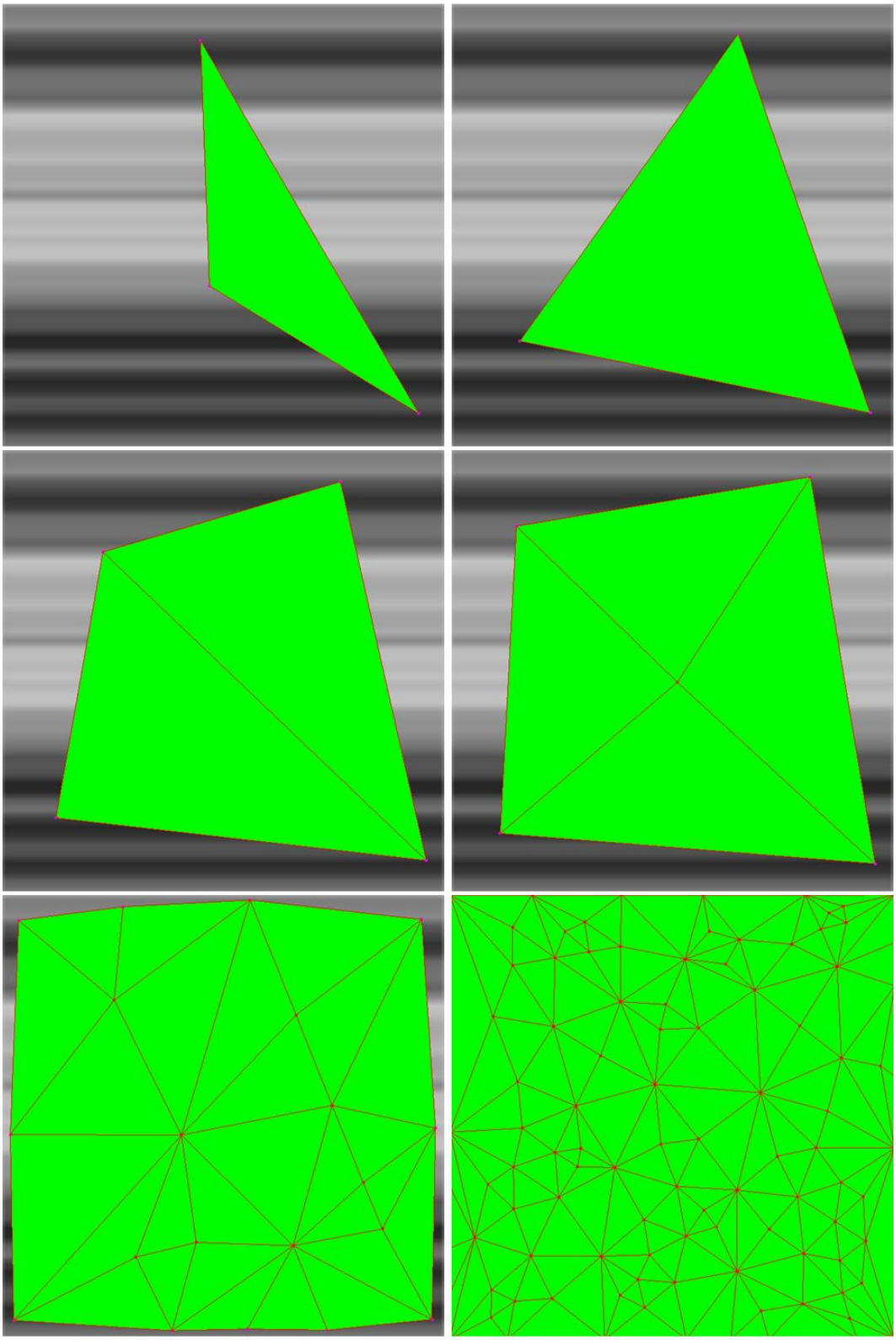

Figura 6-16: Restrição de movimento dos nós da borda. Grafos gerados por MNA para o mesmo dado sintético utilizado no teste do GCS (ver Figura 6-2). Da esquerda para direita, de cima para baixo: grafo inicial, grafo inicial após 196 ciclos, e grafos com 4, 5, 20 e 100 vértices (comparar com Figura 6-3).

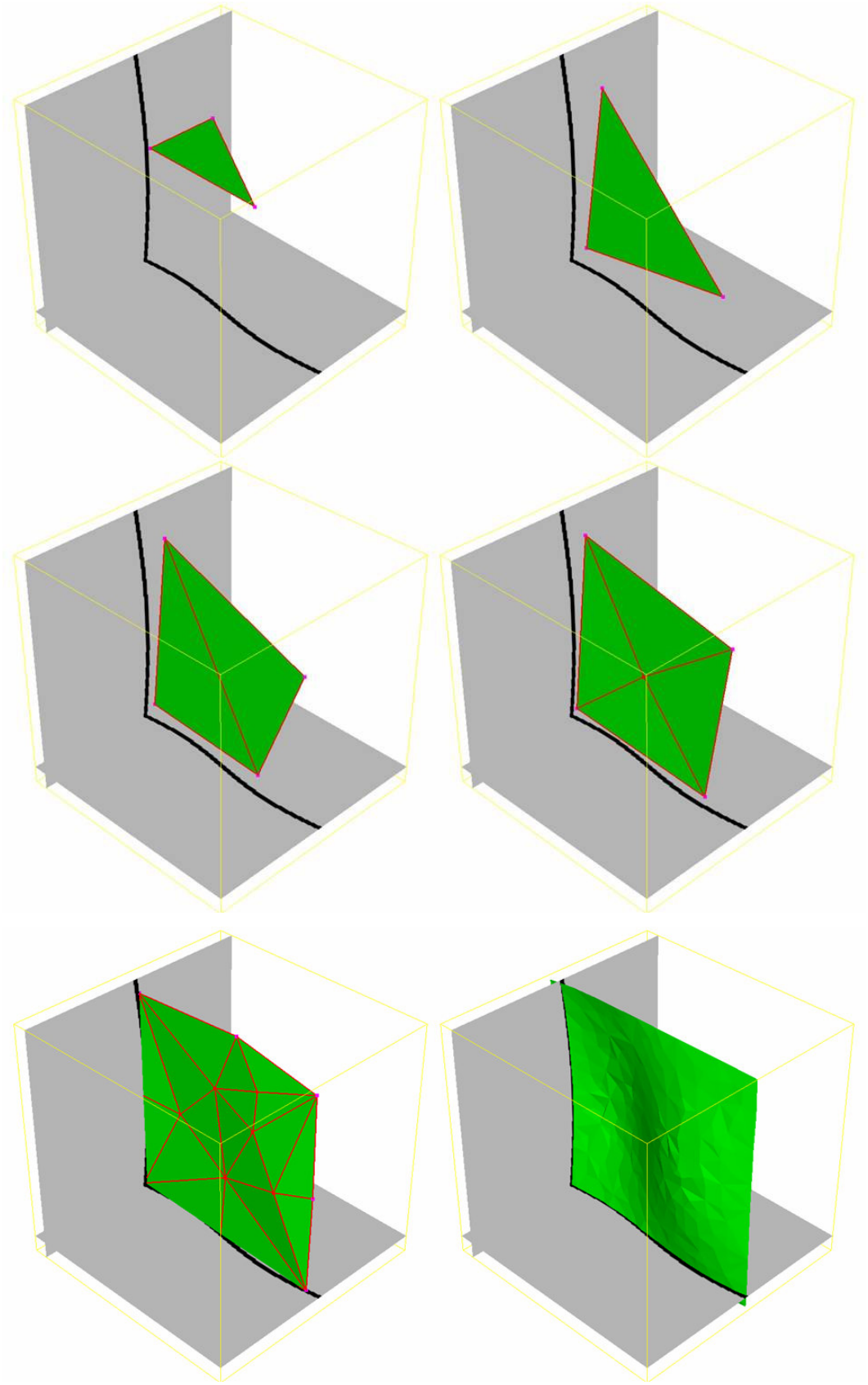

Figura 6-17: Evolução de malha gerada por MNA para o volume de probabilidades ondulada1.

Foi também construído um segundo volume de mesmas dimensões e definido a partir das mesmas funções seno de ondulada1, mas gerando uma estrutura ondulada com 25 voxels de espessura. Este segundo volume de probabilidades é denominado neste documento “ondulada25”. Observe que ele simula situações onde existe uma grande imprecisão sobre a posição da superfície a ser reconstruída.

A imagem de cima da Figura 6-18 foi gerada a partir de ondulada25 com os seguintes valores de parâmetros do MNA:

$$\varepsilon_b = 0.06, \varepsilon_{nit} = 0.9\ \varepsilon_{nin} = 0.0, \varepsilon_{nb} = 0.0, \lambda = 600, \mu = 10, \alpha = 0.05,$$

gerando uma malha de 500 nós. A imagem de baixo, por sua vez, foi gerada com os mesmos parâmetros, com exceção de:

$$\varepsilon_{nin} = 0.05\ .$$

Observe o efeito de suavização obtido pelo uso da componente normal do Laplaciano para os vértices internos. Assim, a suavização pelo Laplaciano, sem a restrição da componente tangencial do NM, permite ao algoritmo MNA gerar superfícies suaves, mesmo com dados de entrada muito irregulares como visto na Figura 6-14.

Como comentado acima, a segunda imagem da Figura 6-18 foi gerada com os seguintes valores de parâmetros do MNA:

$$\varepsilon_b = 0.06, \varepsilon_{nit} = 0.9\ \varepsilon_{nin} = 0.05, \varepsilon_{nb} = 0.0, \lambda = 600, \mu = 10, \alpha = 0.05.$$

A imagem da Figura 6-19, por sua vez, foi gerada com os mesmos parâmetros, com exceção da seguinte alteração:

$$\varepsilon_{nb} = 0.001$$

Observe o efeito de suavização obtido pelo uso da componente normal do Laplaciano, agora para os vértices da borda. Esse efeito é mais pronunciado na parte central do topo da superfície.

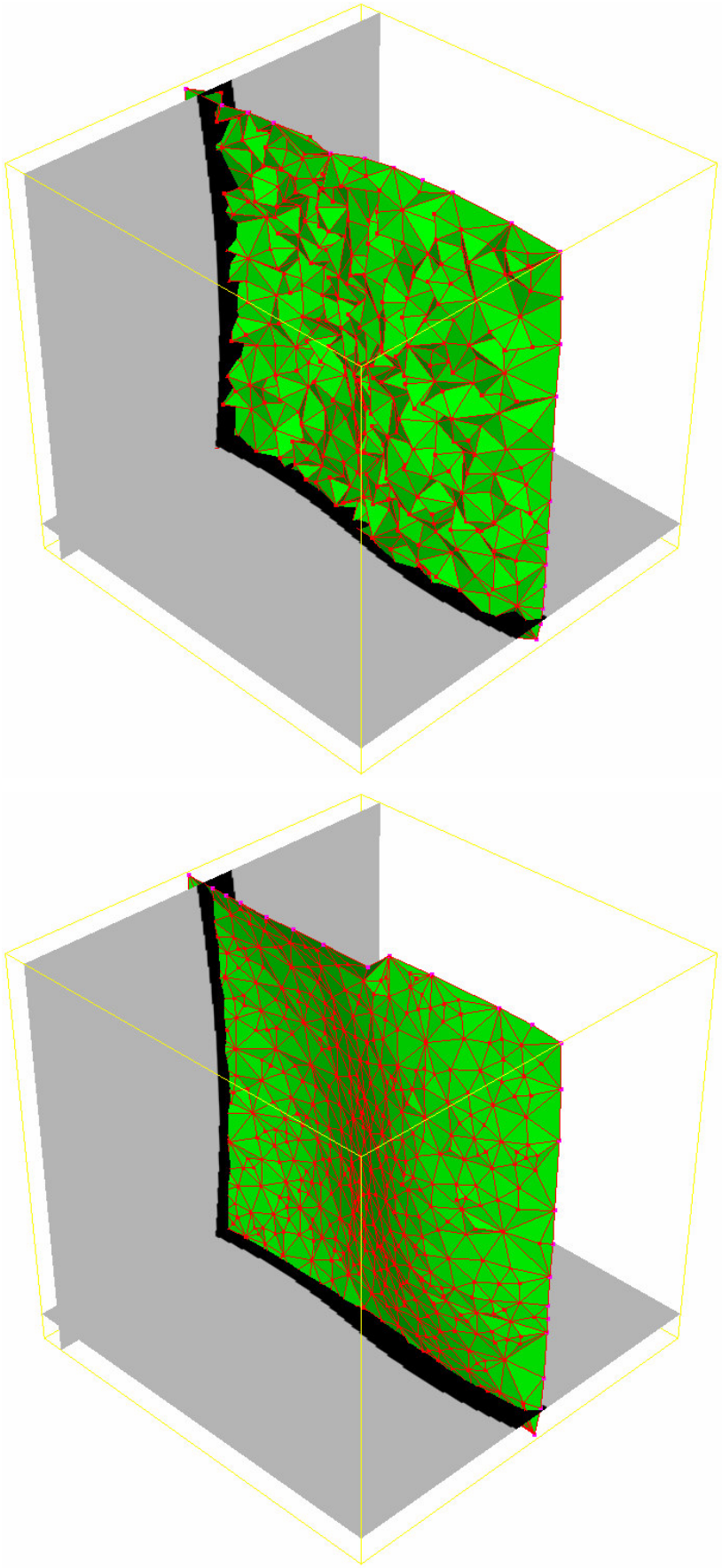

Figura 6-18: Malha gerada por MNA para o volume de probabilidades ondulada25, sem (em cima) e com (embaixo) suavização pela componente normal do Laplaciano.

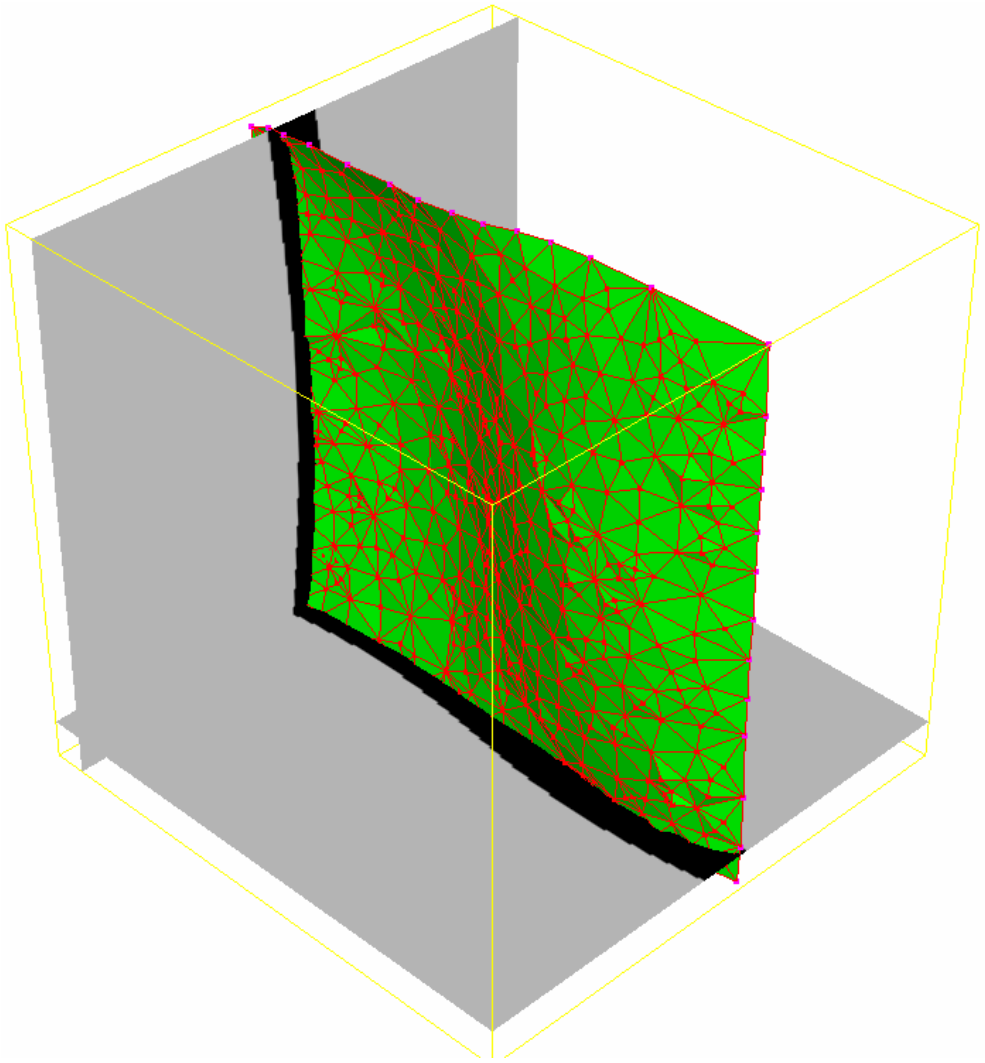

Figura 6-19: Malha gerada por MNA para o volume de probabilidades ondulada25 com suavização de bordas pela componente normal do Laplaciano.

A Figura 6-20 apresenta duas malhas com 300 vértices geradas por MNA a partir do atributo de falha segmentado do volume sísmico real de teste (o mesmo utilizado nos Capítulos anteriores). O segmento utilizado corresponde ao grafo exibido em cor azul nas imagens da Figura 5-9 até a Figura 5-12. Na Figura 6-20 os vértices são também exibidos; os internos em verde claro e os vértices de borda em vermelho. A imagem da esquerda foi gerada por MNA com os seguintes parâmetros:

$$\varepsilon_b = 0.06, \varepsilon_{nit} = 0.06\ \varepsilon_{nin} = 0.06, \varepsilon_{nb} = 0.0, \lambda = 300, \mu = 20, \alpha = 0.05,$$

isto é, sem suavização da borda. A imagem da direita foi gerada com suavização da borda, com uso do seguinte valor:

$$\varepsilon_{nb} = 0.006.$$

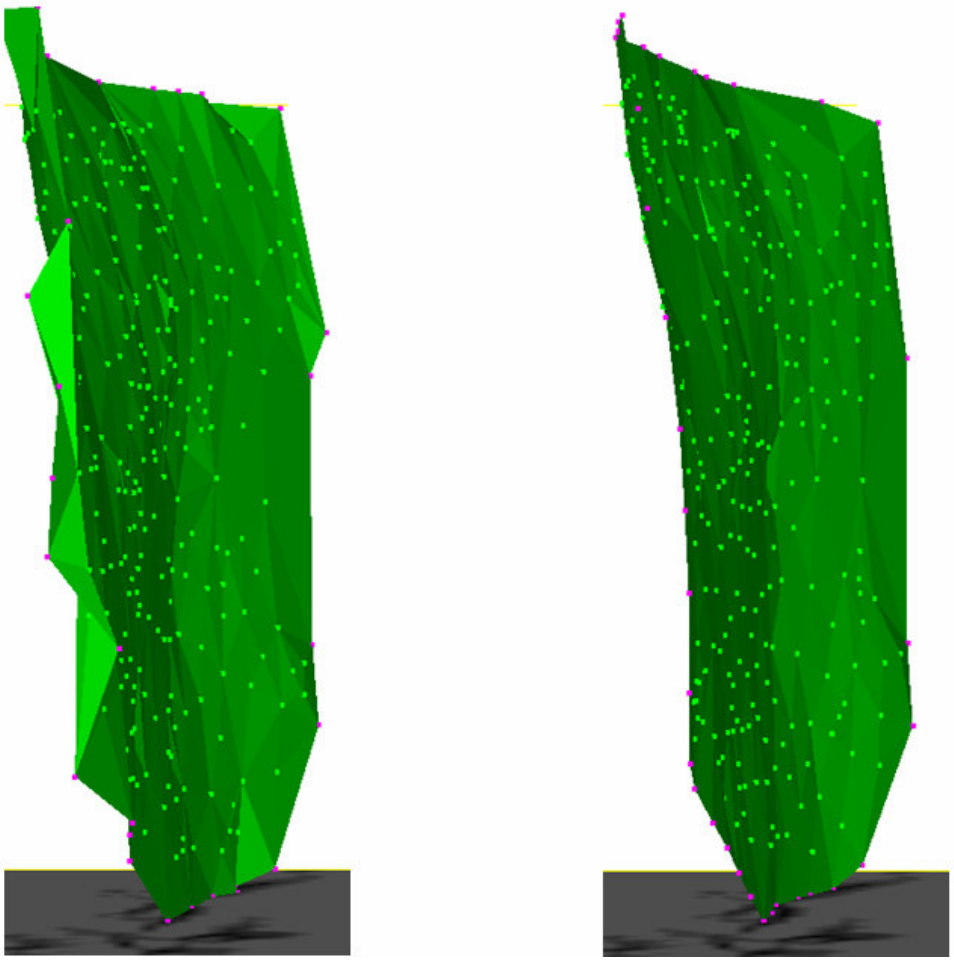

Figura 6-20: MNA sem suavização de bordas (à esquerda) e com suavização de bordas (à direita).

Como comentado anteriormente, a suavização dos nós internos é feita de forma independente para a componente tangencial e a normal do Laplaciano (ver passo 6.1 do algoritmo MNA). A componente tangencial age no sentido de preservar a qualidade da malha. O fator ε_{nit} pode assumir valores altos (próximos a 1.0) como visto nos exemplos anteriores para ondulada25. Valores baixos desse parâmetro aumentam a possibilidade de colapso da malha durante o processo de evolução do MNA. O efeito da componente normal do Laplaciano, por sua vez, foi observado na Figura 6-18. Entretanto, nesse caso, valores altos do fator ε_{nin} podem eliminar aspectos importantes da superfície. Na imagem de cima da Figura 6-21 foi utilizado $\varepsilon_{nin} = 0.05$, enquanto na de baixo foi utilizado $\varepsilon_{nin} = 0.5$. Observe a perda de qualidade da reconstrução no caso do valor mais alto do fator.

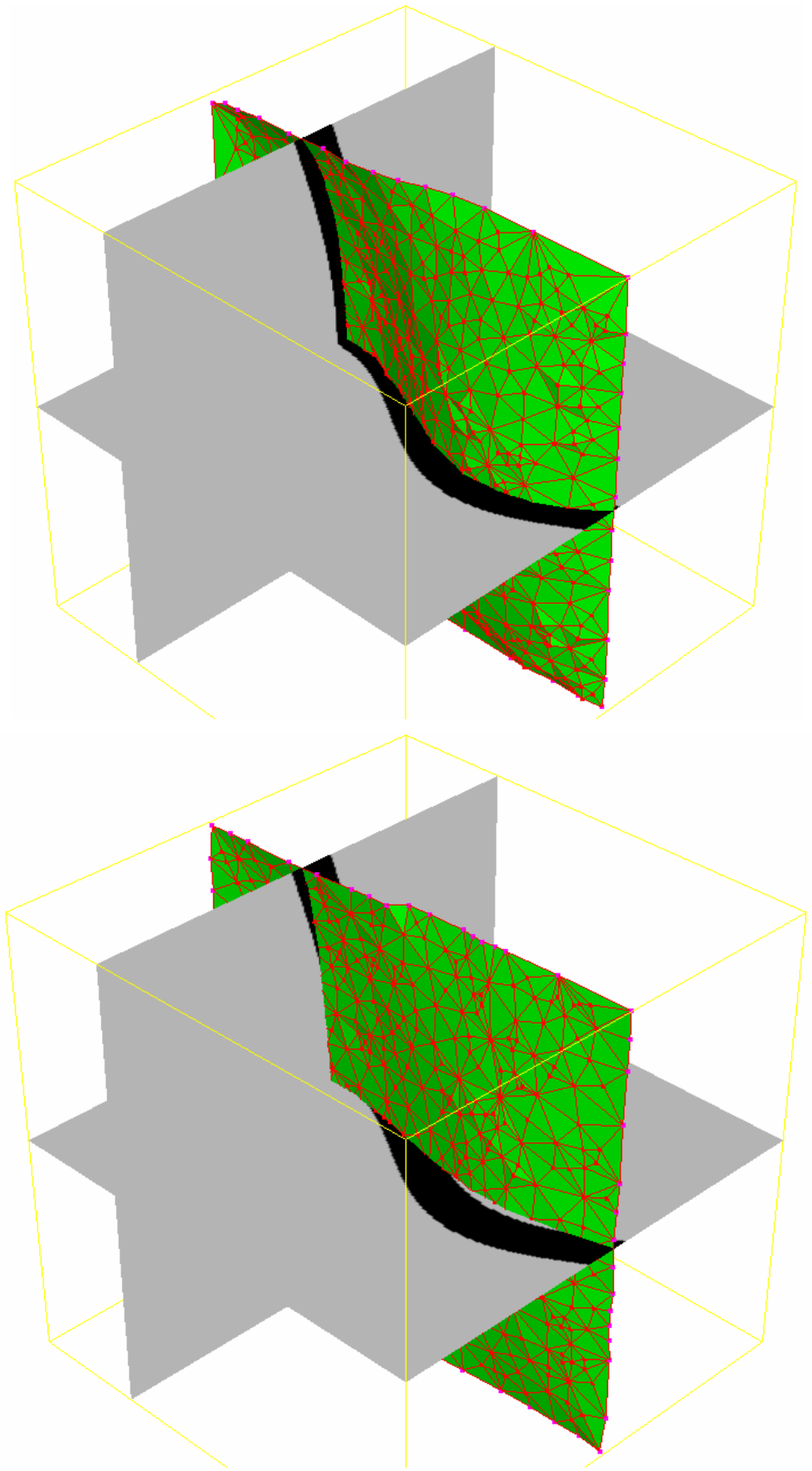

Figura 6-21: Efeito do aumento da componente normal do Laplaciano: na imagem de cima, $\varepsilon_{nin} = 0.05$ e na de baixo, $\varepsilon_{nin} = 0.5$

Por fim, a Figura 6-22 mostra a boa adequação da reconstrução por MNA da superfície para o dado ondulada25. A Figura mostra a malha resultante contra diferentes conjuntos de fatias do volume de probabilidades. A malha foi criada com 500 nós e com os seguintes valores de parâmetros do MNA:

$$\varepsilon_b = 0.06,\ \varepsilon_{nit} = 0.9\ \varepsilon_{nin} = 0.05,\ \varepsilon_{nb} = 0.001,\ \lambda = 600,\ \mu = 10,\ \alpha = 0.05\,.$$

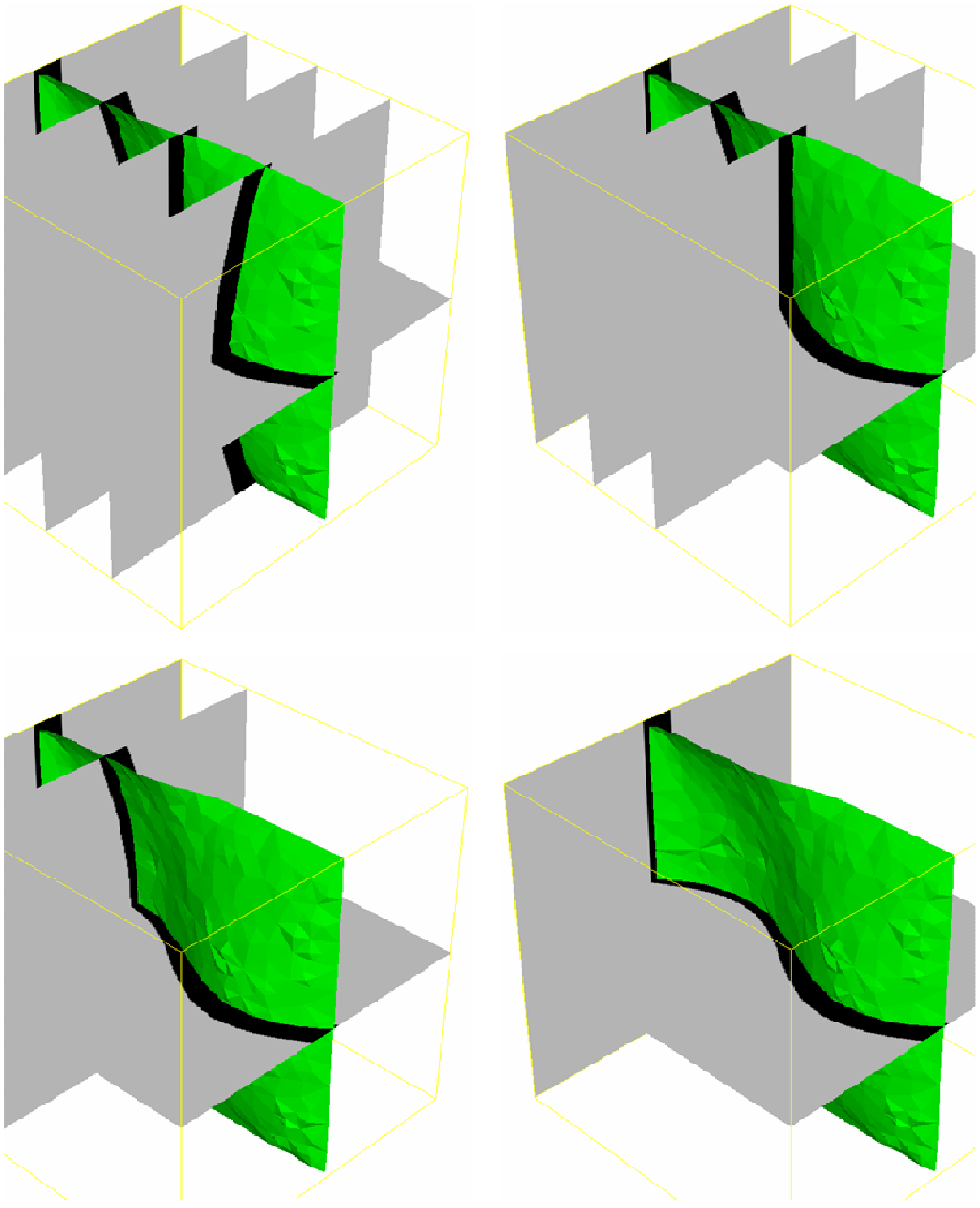

Figura 6-22: Superfície reconstruída por MNA para o volume de probabilidades ondulada25.

6.7. Exemplos de visualização de falhas por MNA

Com a derivação do algoritmo MNA está finalmente completo o conjunto de ferramentas necessárias para implementar o método proposto de Extração e Visualização de Falhas por Aprendizado Competitivo. Esse método foi inicialmente descrito na Seção 3.5. A Figura 6-23 apresenta um diagrama que o resume.

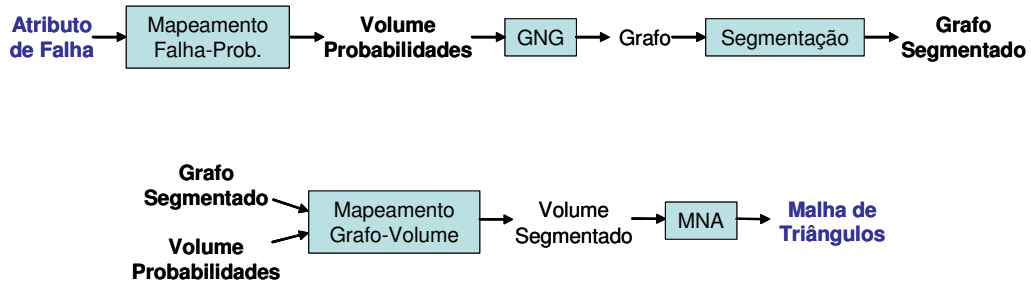

Figura 6-23: Diagrama de Extração e Visualização de Falhas por Aprendizado Competitivo.

A seguir, são apresentadas Figuras com os resultados da aplicação do método proposto sobre o Cubo de Coerência gerado para o volume sísmico de teste. Na seqüência de execuções foram utilizados os seguintes valores de parâmetros:

- Mapeamento do Atributo de Falha em Probabilidades:

  $c = 0.2$

- GNG:

  $a_{max} = 44$, $\varepsilon_b = 0.05$, $\varepsilon_n = 0.0006$, $\lambda = 300$, $\alpha = 0.5$, $\beta = 0.0005$,

  número de nós = 2000;

- Segmentação passo 1 (FH):

  $k = 15$;

- Segmentação passo 2 (orientação):

  $p = 2$, *limAng* = 0.3rad, *limConf* = 0.45, *mergMin* = 1rad, *minNos* = 40;

- MNA:

  $\varepsilon_b = 0.06$, $\varepsilon_{nit} = 0.06$, $\varepsilon_{nin} = 0.06$, $\varepsilon_{nb} = 0.006$, $\lambda = 300$, $\mu = 20$,

  $\alpha = 0.0005$.

O passo de segmentação gerou três subgrafos que são exibidos nas cores azul, amarela e vermelha. Na geração das superfícies por MNA foram gerados 300 nós para o segmento azul, 100 para o amarelo e 20 para o vermelho. Os tempos de execução são da ordem de 70s, 25s e 3s, respectivamente, totalizando 1min e 38s. Estes valores de tempo foram obtidos com a utilização de equipamento com processador Pentium 4 de 3.4GHz.

A Figura 6-24 e a Figura 6-25 dão destaque à superfície gerada a partir do segmento de grafo azul. As fatias exibidas nas Figuras mapeiam o volume de probabilidades gerado para esse segmento. Observe que a superfície se mantém suave apesar das irregularidades do volume de probabilidades.

A Figura 6-26 e Figura 6-27 destacam detalhes do segmento amarelo e vermelho, respectivamente.

A Figura 6-28 e a Figura 6-29 apresentam, além das superfícies geradas por MNA, o volume do atributo de falha mapeado em fatias horizontais e verticais. Essas Figuras permitem uma avaliação da qualidade das superfícies geradas pelo método.

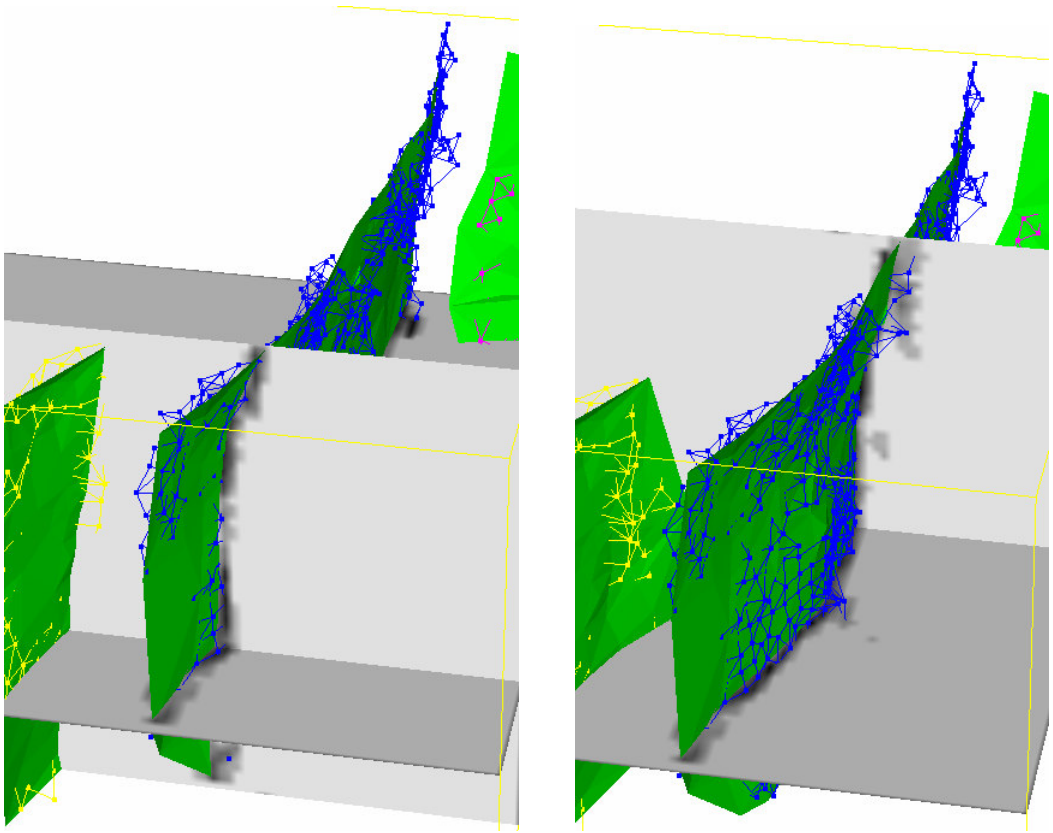

Figura 6-24: Superfície gerada por MNA para o segmento azul. Volume de probabilidades do segmento mapeado em fatias horizontais e verticais.

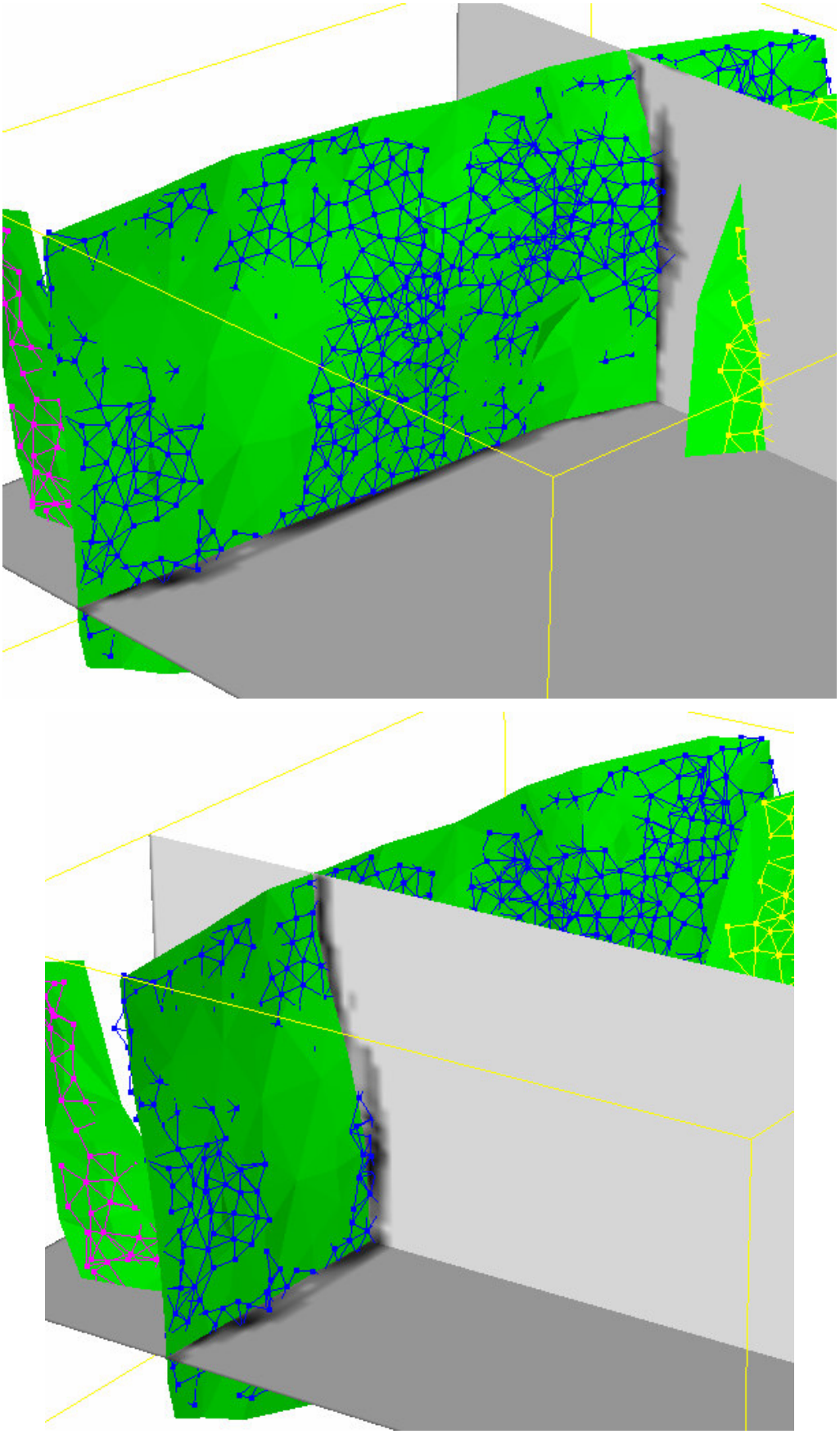

Figura 6-25: Outro ângulo da superfície gerada por MNA para o segmento azul. Volume de probabilidades do segmento mapeado em fatias horizontais e verticais.

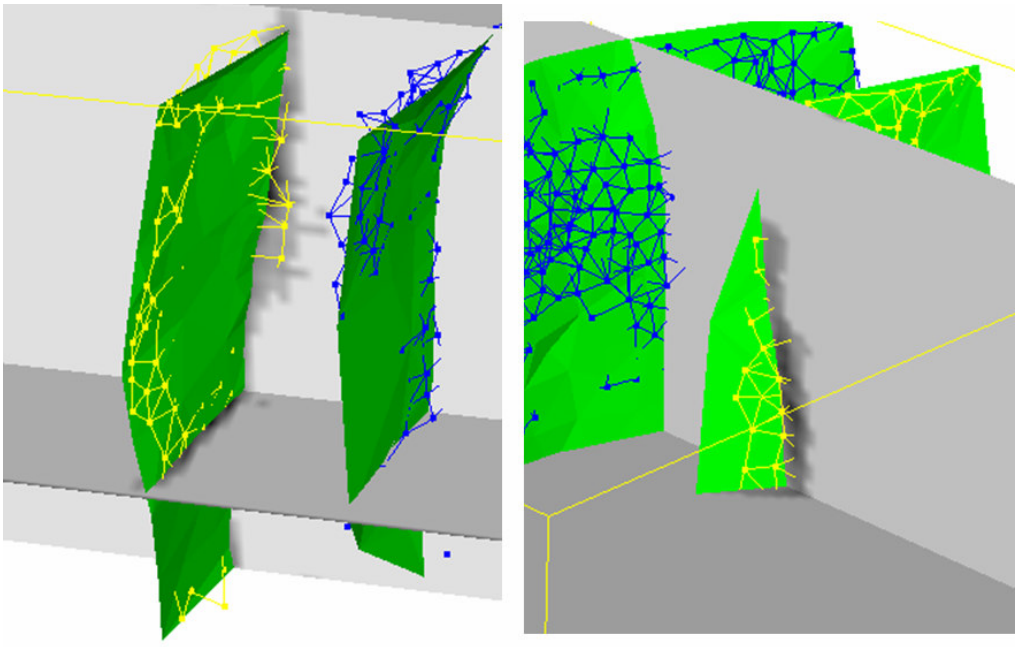

Figura 6-26: Superfície gerada por MNA para o segmento amarelo. Volume de probabilidades do segmento mapeado em fatias horizontais e verticais.

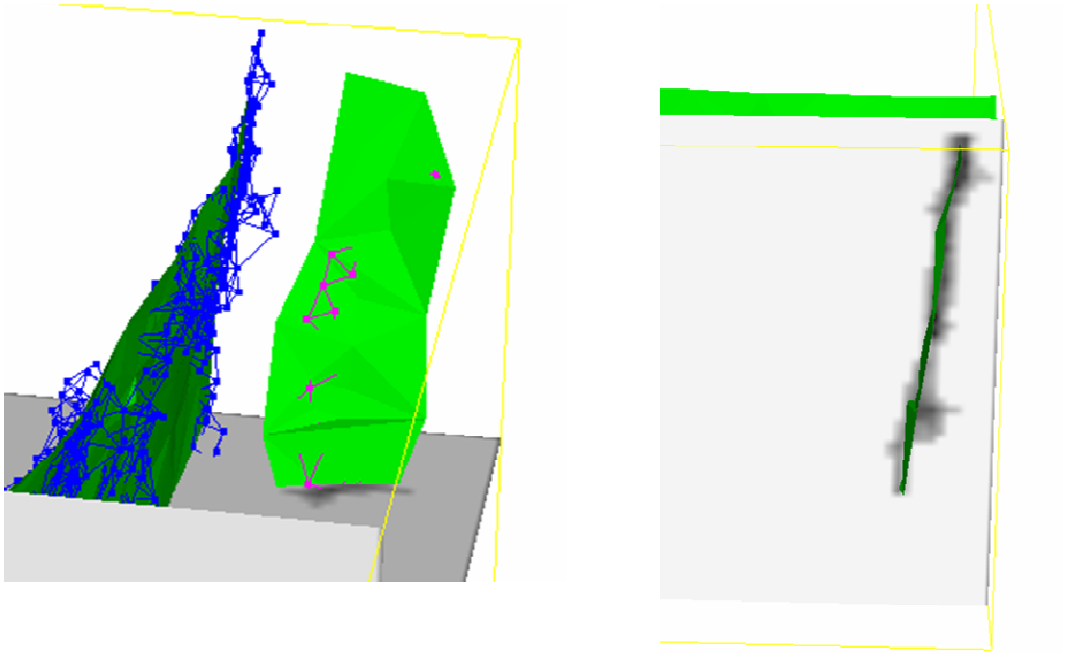

Figura 6-27: Superfície gerada por MNA para o segmento vermelho. Volume de probabilidades do segmento mapeado em fatias horizontais e verticais.

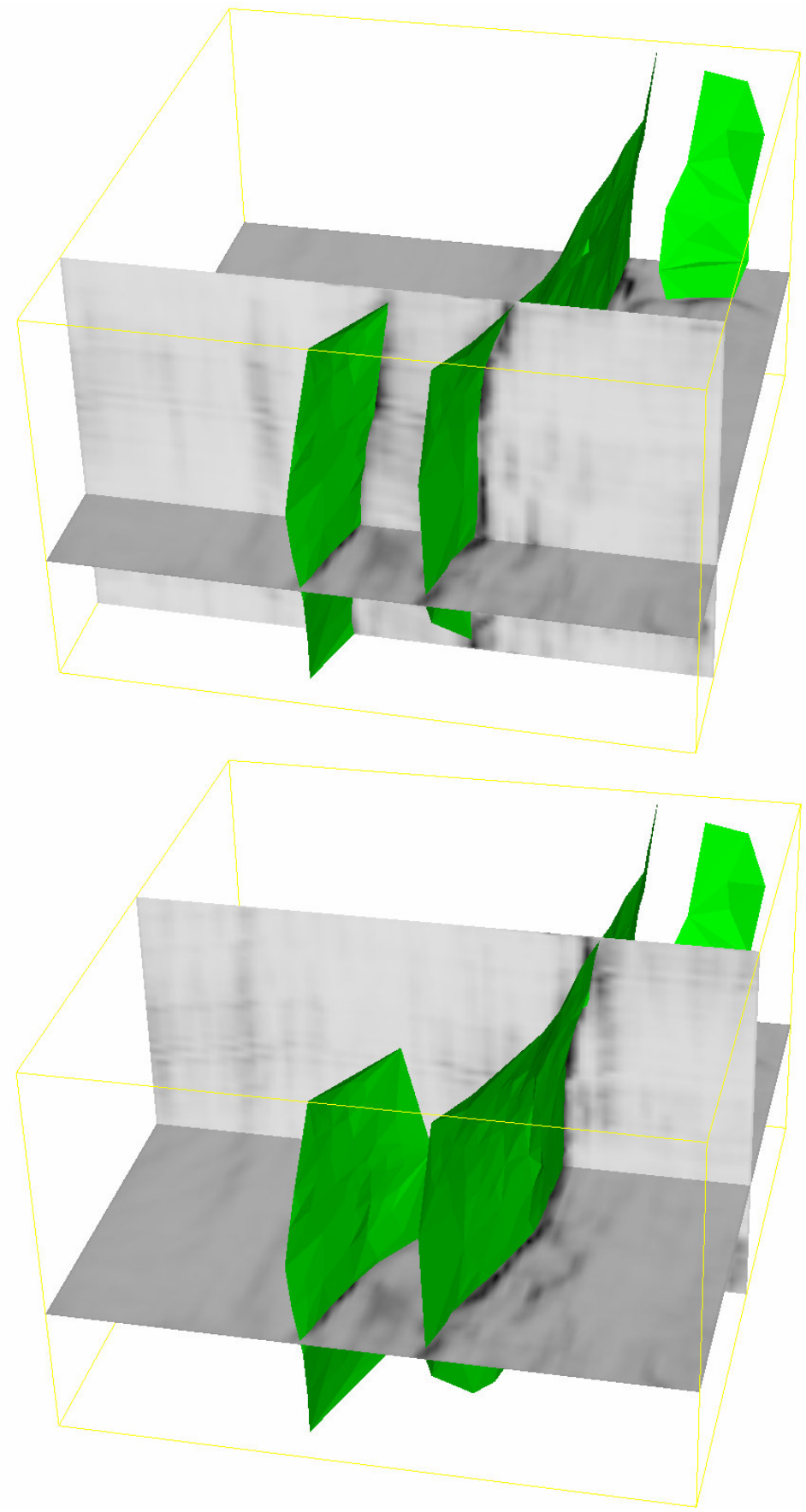

Figura 6-28: Superfícies de falha geradas por MNA com o volume de atributo de falha mapeado em fatias horizontais e verticais.

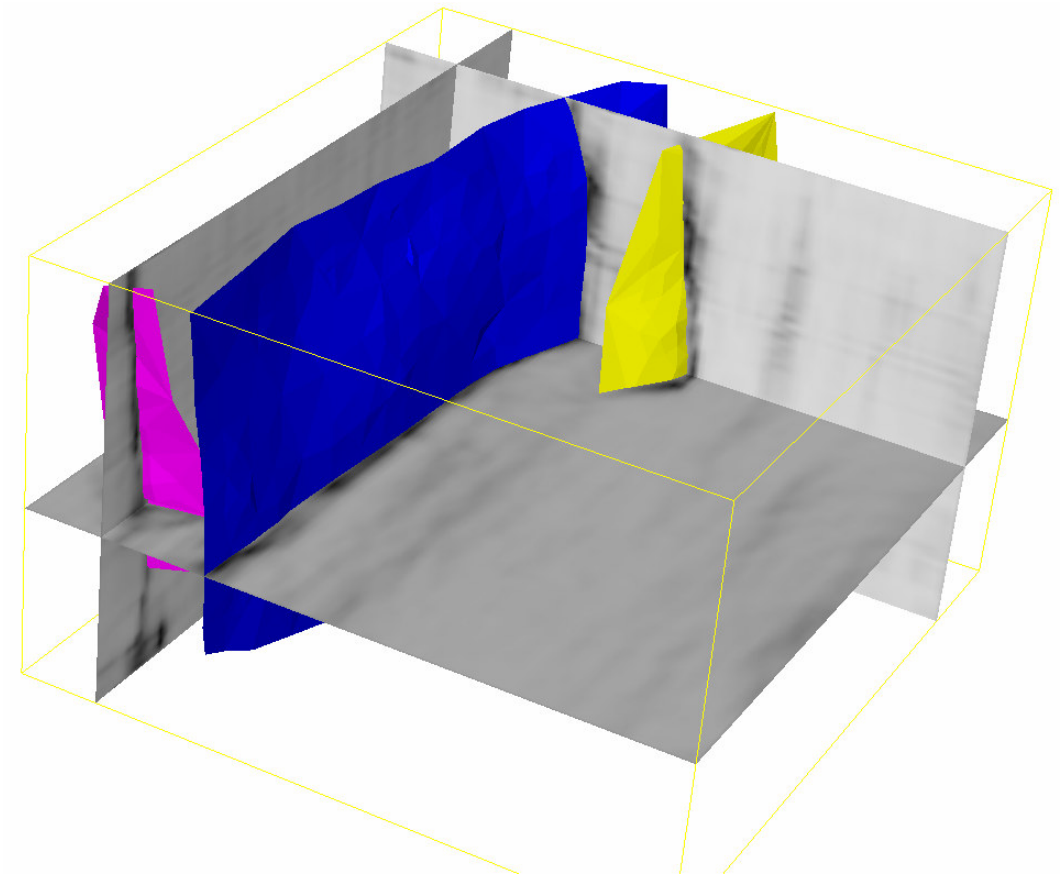

Figura 6-29: Superfícies de falha geradas por MNA exibidas em vermelho, azul e amarelo com o volume de atributo de falha mapeado em fatias horizontais e verticais.

A Figura 6-30 mostra fatias horizontais do volume de amplitudes sísmicas utilizado para gerar o atributo de falha com o algoritmo de Cubo de Coerência por Auto-estrutura sem correção de mergulho. Vale lembrar que o algoritmo de geração do atributo de falha não faz parte do método proposto. Novamente as imagens da Figura mostram uma boa adequação da superfície gerada.

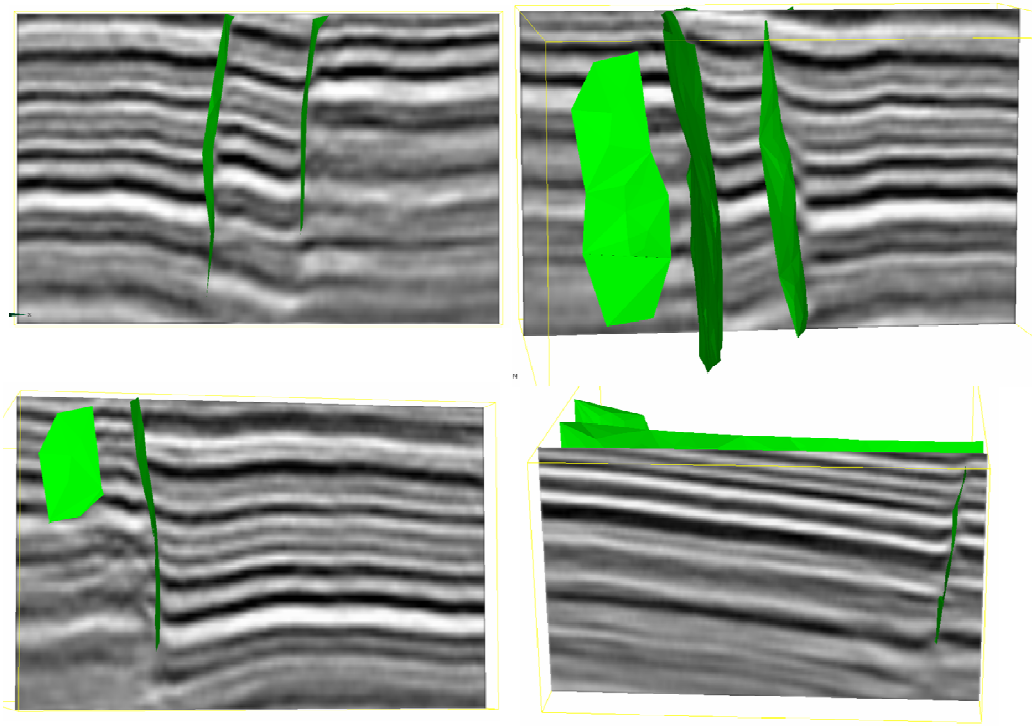

Figura 6-30: Superfícies geradas por MNA com volume de amplitudes sísmicas mapeado em fatias verticais.

Por fim, a Figura 6-31 apresenta na imagem de cima a visualização dos voxels identificados pelo método de supressão de pontos não-máximos de Canny (SNM) (seção 3.2.1); a imagem de baixo apresenta as superfícies geradas por MNA superpostas pelos voxels gerados por SNM. Os valores dos parâmetros utilizados para executar o processo SNM foram os mesmos utilizados para efetuar o pré-processamento do atributo de falhas do método de malhas atômicas de Hale (seção 3.4.1.1), isto é, σ_g = 3 amostras e σ_T = 12 amostras. Observe que as superfícies MNA fecham os vários buracos gerados por Canny.

Para efeito de comparação, o método SNM gerou para o mesmo volume de atributo de falha 8082 amostras não-nulas. Os 420 vértices gerados com o método proposto neste trabalho correspondem a 5% desse total.

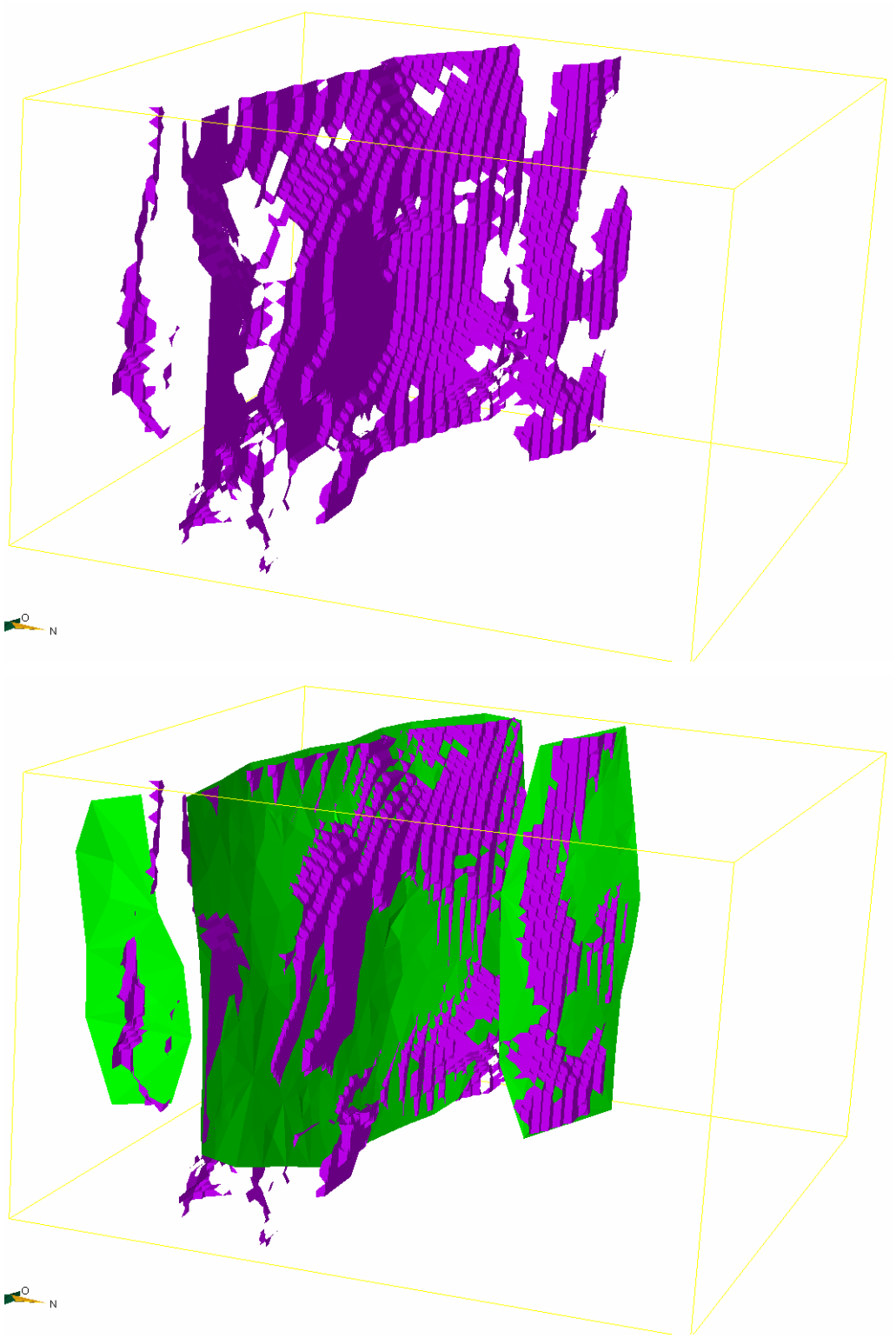

Figura 6-31: Na imagem de cima, visualização dos voxels identificados pelo método de SNM de Canny. Embaixo, superfícies geradas por MNA superpostas pelos voxels de Canny.