5
Resultados

Neste capítulo serão apresentados dados recolhidos sobre os modelos gerados em pré-processamento. A seguir, serão apresentados os testes realizados para verificar o desempenho do visualizador.

Os resultados obtidos podem ser divididos em dois grupos. No primeiro, temos os resultados que tentam analisar a qualidade da estrutura hierárquica gerada durante o pré-processamento. São eles os dados sobre distribuição de alturas da árvore gerada e as estatísticas de triângulos e voxels gerados por nó. No segundo grupo, temos testes de desempenho realizados usando o visualizador, que visam verificar se os diversos componentes funcionam bem em conjunto.

Tanto o pré-processamento quanto os testes de desempenho foram realizados em um computador com processador Athlon X2 64 4200+, 4 GB de memória RAM, placa 3D nVidia GeForce 8800 GTX com 768 MB de memória. A etapa de pré-processamento não fez nenhum uso da placa gráfica e foi executada no Windows Professional 64-bits, para poder usar toda a memória disponível. Os testes de desempenho foram realizados numa janela de 854x641 pixels e o driver da placa 3D foi configurado para usar filtro de anti-serrilhamento 4x e anti-serrilhamento de transparência por multi-amostragem, e foram rodados no Windows Professional 32-bits.

5.1
Modelos usados nos testes

Para testar o funcionamento e o desempenho do visualizador desenvolvido, foram usados 3 modelos de CAD de estruturas marítimas em operação pela Petrobras. Na tabela 5.1 apresentamos as características de cada um desses modelos. Nessa tabela também apresentamos as características do modelo otimizado gerado a partir deles para a visualização e o tempo que a etapa de pré-processamento gastou em cada um deles.

Modelo	Faces	Vértices	Tempo de geração	Nós de Voxels	Nós de Geometria	Faces por nó (média)	Voxels por nó (média)
P-38	8.4 milhões	9.7 milhões	4 h	195	150	56.237	24.313
P-40	22.3 milhões	19.3 milhões	22 h	843	701	31.811	19.986
P-50	29.8 milhões	37.9 milhões	27 h	945	770	38.701	21.237

Tabela 5.1. Características dos modelos usados nos testes.

O primeiro resultado do pré-processamento analisado visa medir o balanceamento da árvore gerada na etapa de criação da hierarquia. Como foi explicado na seção 3.1, uma KD-Tree é recomendada nesse tipo de divisão já que garante esse balanceamento. A *octree*, por outro lado, gera naturalmente uma árvore com menos níveis, o que é interessante para a nossa implementação. Os histogramas das figuras 5.1, 5.2 e 5.3 são criados a partir das alturas dos nós obtidas na divisão feita por *octree* nos nossos modelos de teste. Em todos os casos, a altura ficou limitada a no máximo 12 níveis com a maior parte dos nós folha concentrados em níveis próximos na hierarquia.

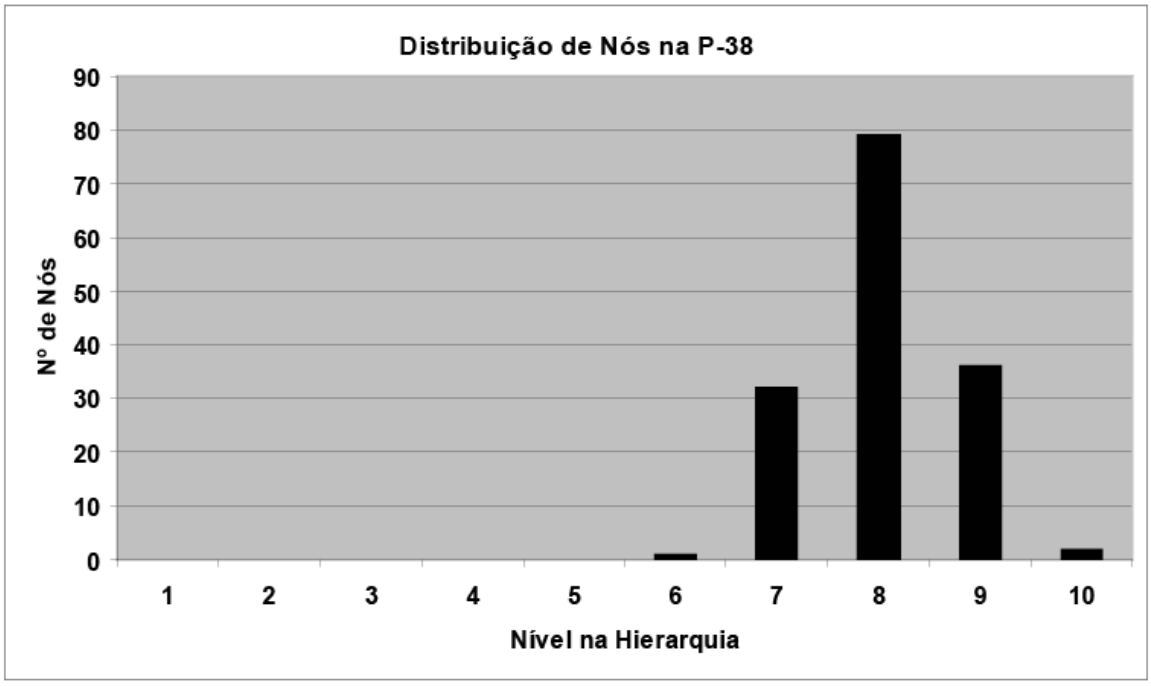

Figura 5.1 - Distribuição de nós na P-38

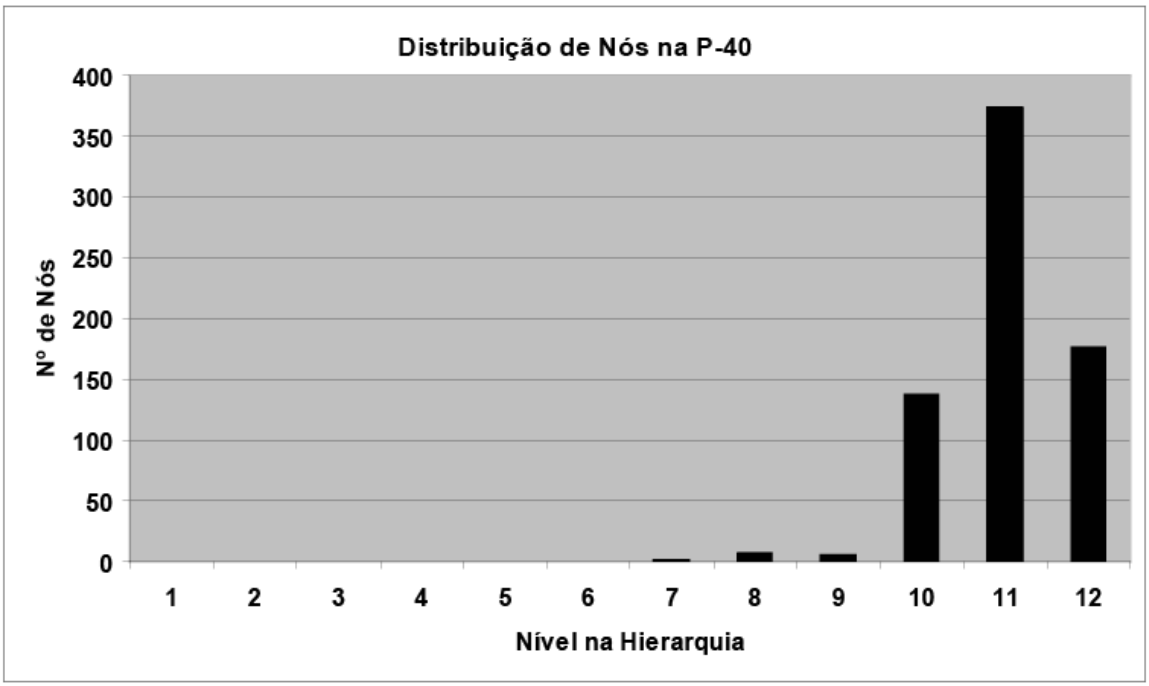

Figura 5.2 - Distribuição de nós na P-40

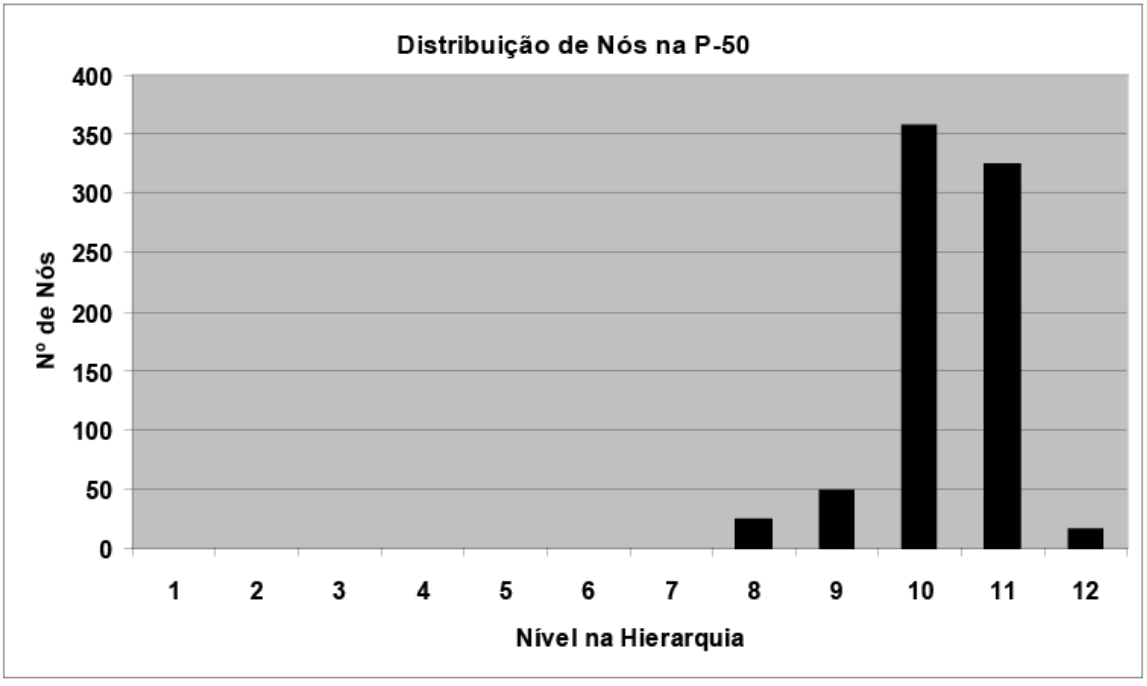

Figura 5.3 - Distribuição de nós na P-50

5.2
Testes de desempenho do algoritmo

O principal teste realizado foi o teste de desempenho. Esse teste simula um caminhamento típico dentro do modelo usando uma animação de câmera

pré-gravada. Para cada quadro renderizado durante o percurso recolhemos dados sobre a visualização e calculamos a taxa de quadros para gerar os gráficos comparativos. A seguir, realizamos o mesmo percurso modificando diversos parâmetros da renderização para descobrir qual estágio da cadeia de renderização está limitando o desempenho durante cada trecho da navegação.

Os dados recolhidos durante a renderização para testar o desempenho geram um gráfico formado por três linhas, denominado “Desempenho de Visualização” e apresentados a seguir. Essas linhas comparam o desempenho obtido pelo algoritmo para visualização usando voxels desenvolvido nesse trabalho com um algoritmo de visualização básico, sem suporte a LODs, mas com algumas otimizações mínimas para conseguir lidar com modelos massivos. Além disso, o desempenho do visualizador também é comparado com o obtido pelo mesmo algoritmo de voxels sem os testes de oclusão em hardware. Isso nos permite analisar separadamente o ganho obtido com o uso do LOD Hierárquico usando voxels e com os testes de oclusão em hardware. Nos gráficos que serão apresentados, o algoritmo com testes de oclusão é identificado como “Voxels com testes de oclusão” e o algoritmo sem esses testes é identificado como “Voxels sem testes de oclusão”.

O algoritmo de visualização básico usado nessa comparação usa apenas triângulos para representar o modelo, agrupados em clusters para otimizar o envio de sua geometria para a placa gráfica. Assim, ele também consegue evitar que seja criado um gargalo em CPU durante o percurso da cena para seleção de objetos a serem renderizados. Esse algoritmo também descarta objetos contidos fora da pirâmide de visão, ordena por materiais as primitivas a serem renderizadas, para evitar a troca desnecessária de estados no OpenGL e envia os dados para a placa gráfica usando VBOs (*VertexBufferObjects*) para evitar o envio desnecessário desses dados em quadros posteriores através do barramento do PC.

Os testes realizados para identificar o gargalo da visualização são apresentados divididos em dois gráficos para cada modelo, sob o título “Análise de Gargalo”. Esses testes envolvem eliminar ou reduzir de forma controlada o processamento nos diversos estágios da cadeia de renderização. A comparação

dos dados obtidos com essas modificações aplicadas permitem isolar com bastante segurança qual estágio é responsável por limitar o desempenho do visualizador. Essa informação pode ser usada posteriormente para guiar novos desenvolvimentos ou otimizações no mesmo. Foram realizadas navegações com 4 variações do algoritmo.

Na primeira variação, chamada de “Voxels sem desenho” nos gráficos que serão apresentados a seguir, todo o envio de dados à placa foi suprimido, mas o visualizador continua percorrendo normalmente a estrutura hierárquica gerada. Um ganho de desempenho nesse teste em comparação com os testes usando toda a cadeia de renderização indica que a placa gráfica é a responsável pelo gargalo. Caso o desempenho continue inalterado, temos a indicação de que a renderização está sendo limitada pelo desempenho da CPU.

Na segunda variação, denominada “Voxels sem iluminação” o cálculo de iluminação, que é realizado por vértices tanto para voxels como para geometria, foi removido. Assim, o estágio de transformação de vértices foi simplificado de forma que pudesse trabalhar sob menos carga. A geometria formada por triângulos passou a ser desenhada sem iluminação e a parte do *shader* responsável por calcular a iluminação do voxel foi removida. Outros cálculos realizados no *shader* dos voxels como a transformação da posição do vértice para coordenadas de *clipping* e o cálculo do tamanho do voxel tiveram que continuar sendo realizados, para não alterar a carga exercida sobre o estágio de rasterização. Um ganho de desempenho durante os testes com essa variação do algoritmo indica que o gargalo está sendo causado pelo estágio de transformação e iluminação de vértices.

A terceira variação, chamada de “Voxels sem anti-serrilhamento em placa” nos gráficos a seguir, consiste em usar o mesmo algoritmo usado na navegação com testes de oclusão, mas com o anti-serrilhamento da placa gráfica desabilitado. Um ganho de desempenho nesse teste indica que a rasterização é um gargalo.

Procurar o gargalo em uma placa gráfica como a GeForce 8800 GTX pode ser uma tarefa mais complicada do que em placas mais antigas. Por ser construída com uma arquitetura unificada, em que processadores genéricos

podem atuar tanto na etapa de transformação de vértices como na etapa de rasterização, é esperado que essas placas nunca apresentem gargalos em apenas uma dessas etapas. Como será demonstrado nas análises apresentadas a seguir, esse comportamento em que o gargalo está em dois estágios simultaneamente só foi observado em um dos três modelos usados.

5.2.1
Testes de desempenho com o modelo da P-38

O primeiro modelo testado foi o modelo CAD da P-38, nosso modelo mais leve. A Figura 5.4 mostra a trajetória da câmera usada durante a navegação, a Figura 5.5 mostra imagens tiradas em cada instante da visualização e a Figura 5.6 mostra a taxa de quadros obtida em cada instante da trajetória.

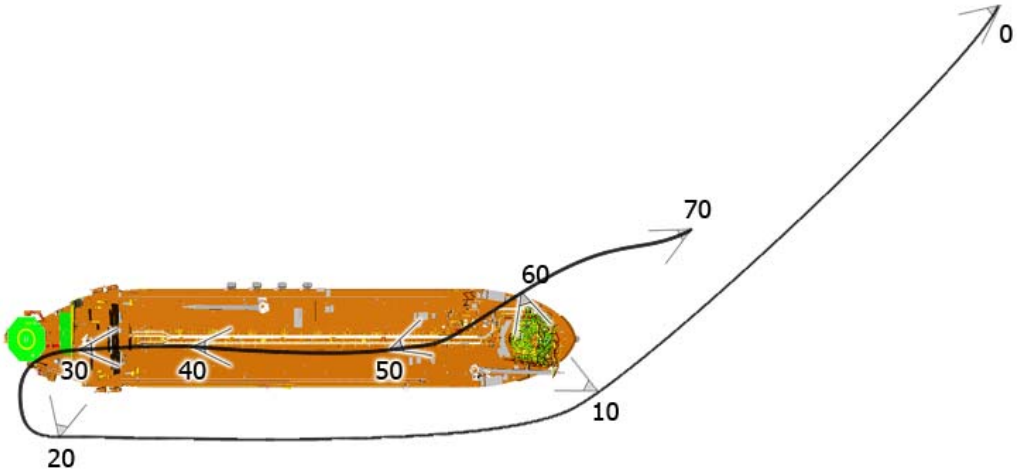

Figura 5.4 - Caminho percorrido durante o teste de desempenho na P-38. Os números indicam onde a câmera está em cada instante da animação

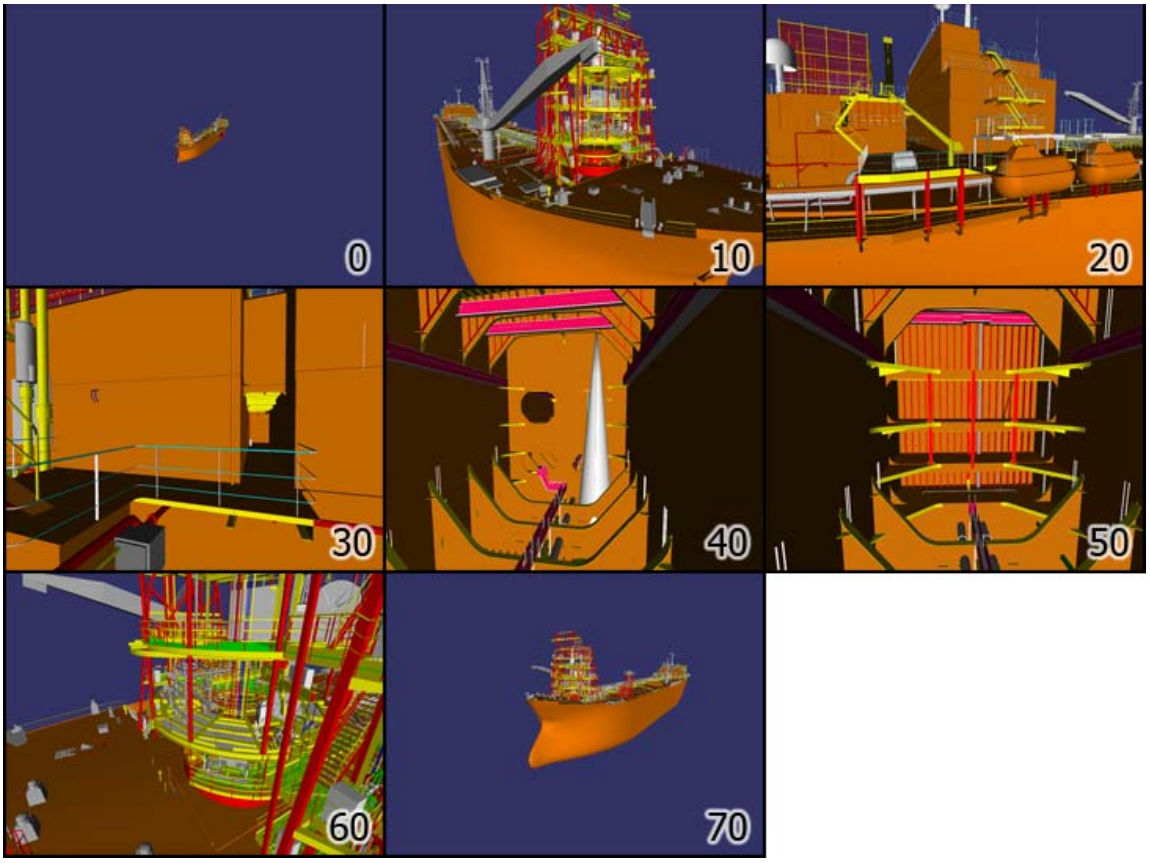

Figura 5.5 – Imagens de cada trecho do caminho percorrido durante os testes na P-38

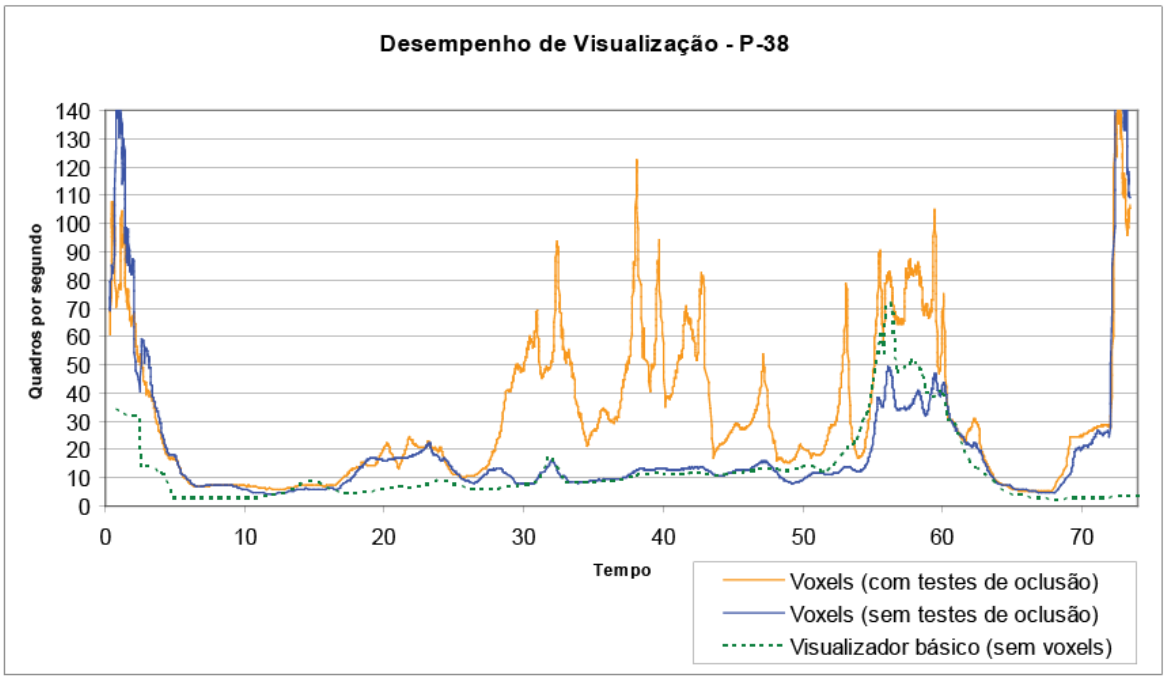

Figura 5.6 - Resultado em quadros por segundo dos testes de desempenho na P-38. Os valores de tempo se relacionam com as posições indicadas na Figura 5.4 e 5.5

O resultado mostra que, para modelos muito leves, a otimização usando voxels implementada nesse trabalho apresenta pouca vantagem. As maiores

diferenças são sentidas quando o modelo é visualizado de longe, como pode ser visto antes do tempo 10 e após o tempo 70, na figura 5.6. Os testes de oclusão conseguem dar um bom ganho quando a navegação ocorre em partes internas do modelo (entre os instantes 30 e 50).

Os próximos testes realizados visam identificar em quais etapas da cadeia de renderização ocorreram gargalos durante a navegação nesse modelo. Os resultados são apresentados nas Figuras 5.7 e 5.8 e são analisados a seguir.

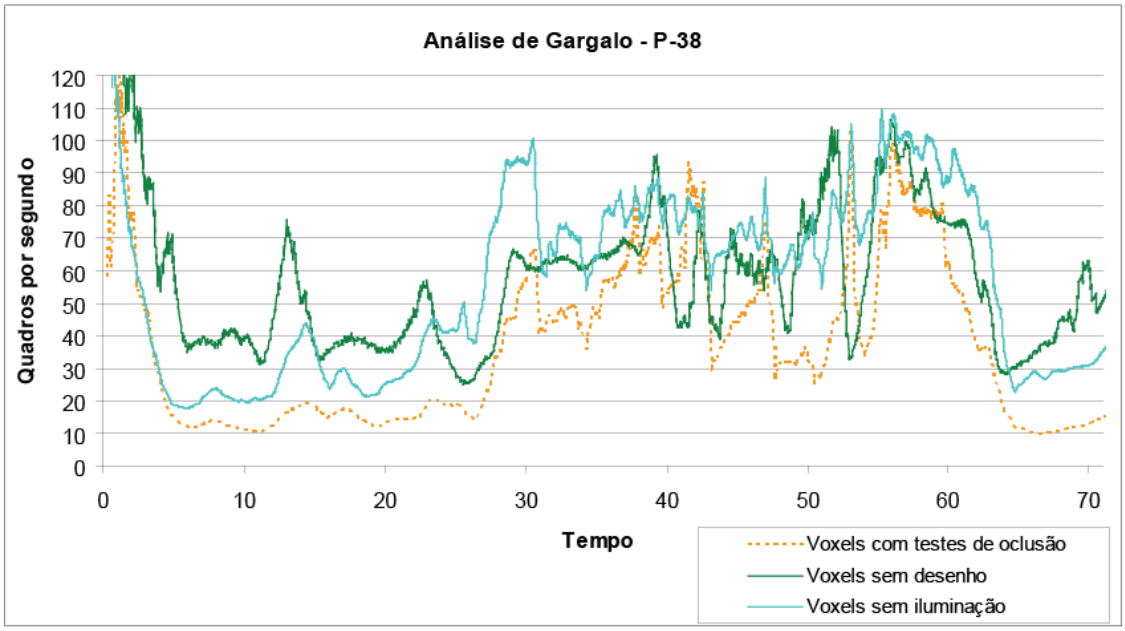

Figura 5.7 – Primeiro gráfico com resultados da análise realizada para localizar o gargalo na navegação da P-38. Esse gráfico compara os resultados obtidos na visualização normal, os resultados obtidos sem que nenhuma primitiva fosse enviada à placa e os resultados obtidos com o estágio de vértice simplificado

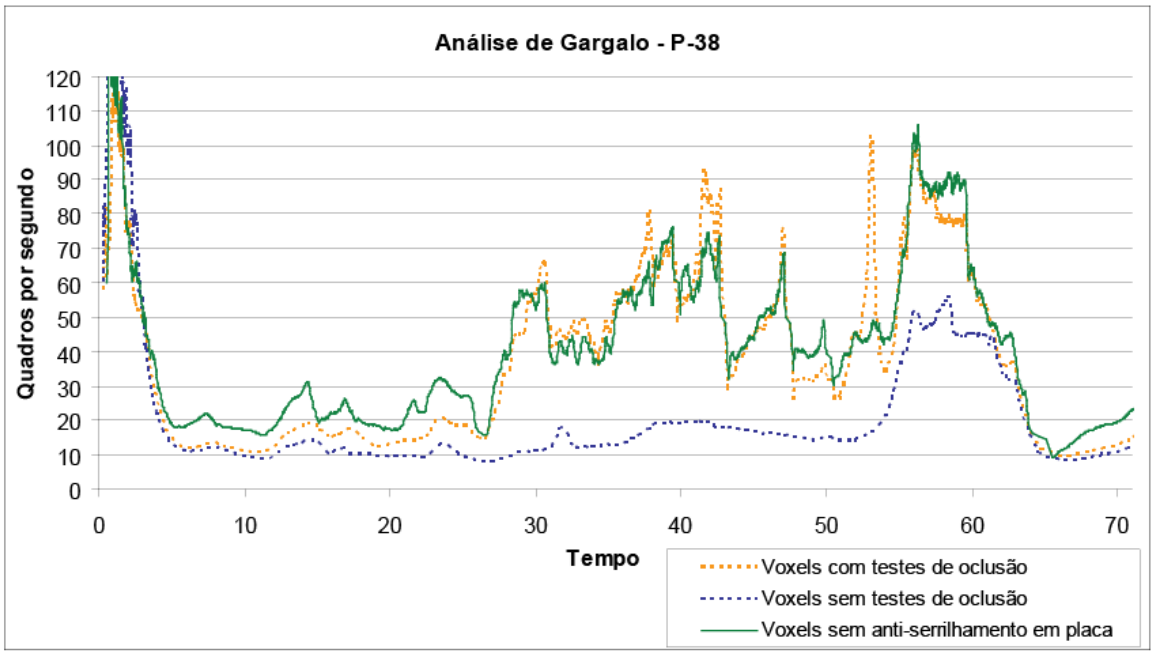

Figura 5.8 – Segundo gráfico com resultados da análise realizada para localizar o gargalo na navegação da P-38. Esse gráfico compara os resultados obtidos na visualização normal com os resultados obtidos sem o uso de anti-serrilhamento em hardware

Os primeiros resultados, apresentados na Figura 5.7, permitem afirmar que o gargalo não estava na CPU durante toda a navegação, já que desabilitar o desenho dos objetos e reduzir a complexidade do estágio de transformação de vértices trouxe ganhos de desempenho significativos durante todo o tempo do teste.

O ganho de performance obtido com a simplificação do estágio de transformação de vértices indica que este é um gargalo.

Os resultados da Figura 5.8 indicam dois resultados diferentes para o estágio de rasterização. A partir do tempo 25 do percurso, não há ganho de desempenho quando o anti-serrilhamento é desligado na placa gráfica, o que indica que o estágio de rasterização não é um gargalo durante a visualização nesse trecho.

Por outro lado, entre o tempo 5 e 25, houve ganho no desempenho quando o anti-serrilhamento foi desligado. Isso indicaria que o gargalo, durante essa parte da navegação, é a rasterização. Por outro lado, também há ganho quando a etapa de transformação de vértices é simplificada. Isso pode indicar que, nessa situação, a placa gráfica está conseguindo balancear a carga entre os estágios de

transformação de vértice e de rasterização como esperado na nova arquitetura unificada.

É interessante notar que a perda de desempenho por causa do anti-serrilhamento na placa gráfica só é percebida nos trechos em que os testes de oclusão não conseguem eliminar uma parte significativa do modelo, ou seja, nos trechos em que o desempenho com testes de oclusão não é muito maior que a o desempenho sem esses testes.

5.2.2 Testes de desempenho com o modelo da P-40

O segundo modelo testado foi o modelo de CAD da P-40, um modelo intermediário em termos de números de polígonos. Uma característica a ser notada nesse modelo é seu formato quadrado, que contribui para aumentar a concentração de geometria em torno da câmera em comparação com os outros dois modelos, em que a geometria se distribui ao longo de um comprimento maior. A Figura 5.9 mostra a trajetória da câmera usada durante a navegação, a figura 5.10 mostra imagens tiradas em cada instante da visualização e a Figura 5.11 mostra a taxa de quadros obtida em cada instante da trajetória.

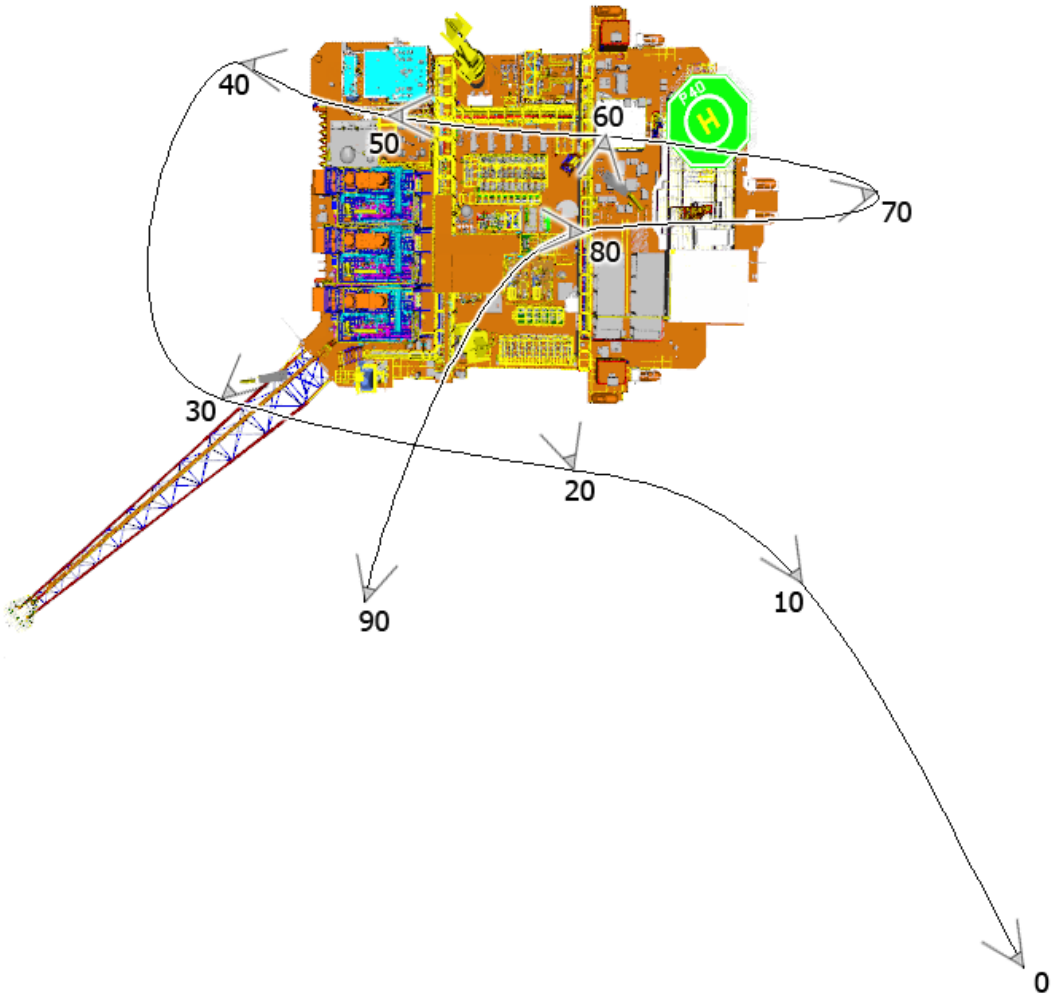

Figura 5.9 - Caminho percorrido durante o teste de desempenho na P-40. Os números indicam onde a câmera está em cada instante da animação

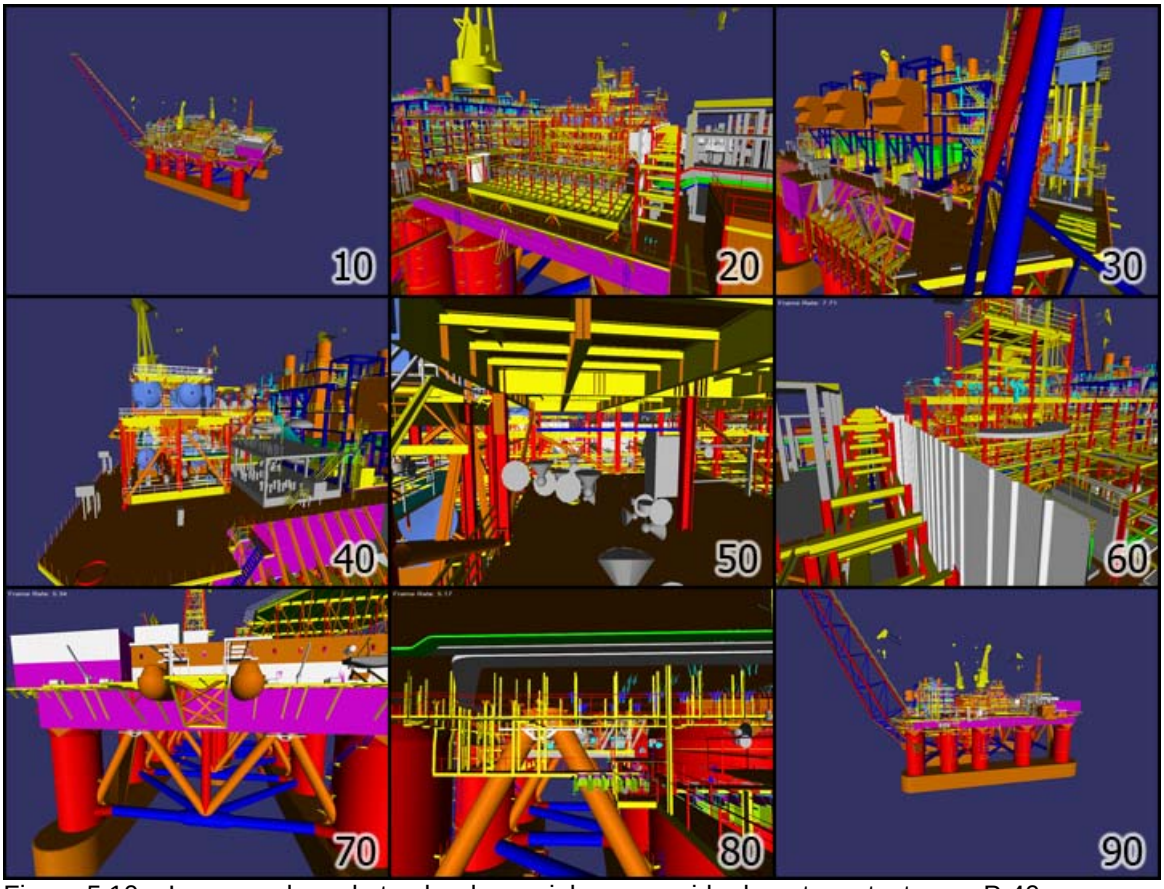

Figura 5.10 – Imagens de cada trecho do caminho percorrido durante os testes na P-40

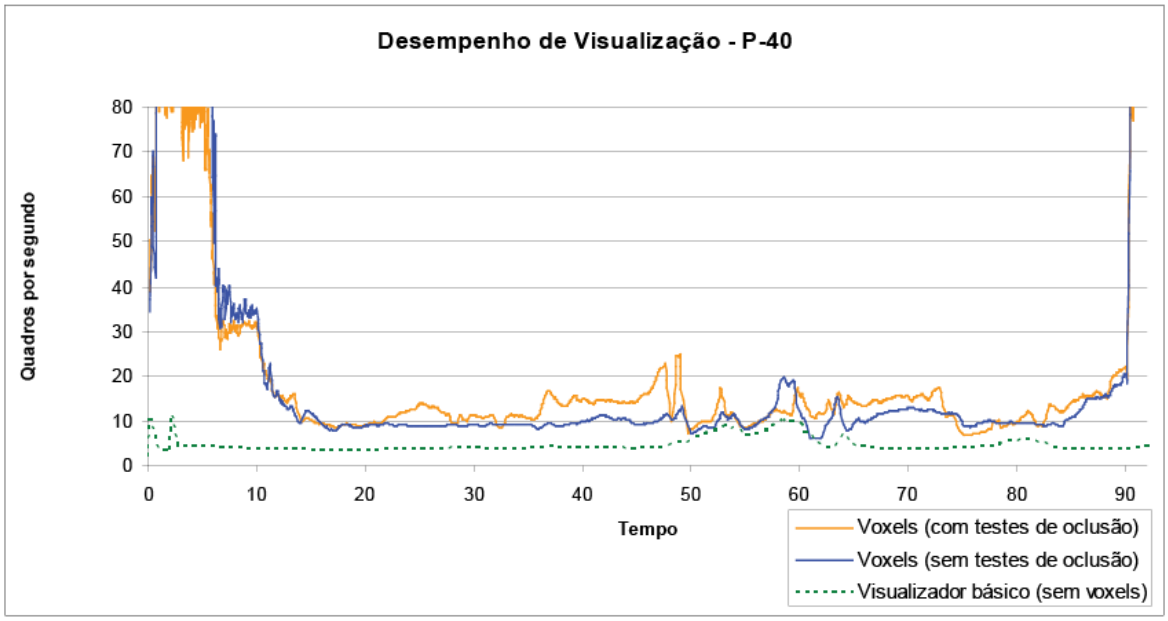

Figura 5.11 - Resultado em quadros por segundo dos testes de desempenho na P-40.
Os valores de tempo se relacionam com as posições indicadas nas Figura 5.9 e 5.10

O resultado apresentado mostra que, para modelos razoavelmente complexos, a otimização usando Voxels Distantes implementada nesse trabalho

apresenta um ganho de desempenho razoável em praticamente todos os trechos da navegação. Os testes de oclusão realizados conseguem melhorar ainda mais o desempenho em diversos trechos, principalmente naqueles em que a navegação ocorre dentro de áreas do modelo que estejam envolvidas por estruturas.

Os próximos testes realizados visam identificar em quais etapas da cadeia de renderização ocorreram gargalos durante a navegação nesse modelo. Os resultados são apresentados nas Figuras 5.12 e 5.13 e são analisados a seguir.

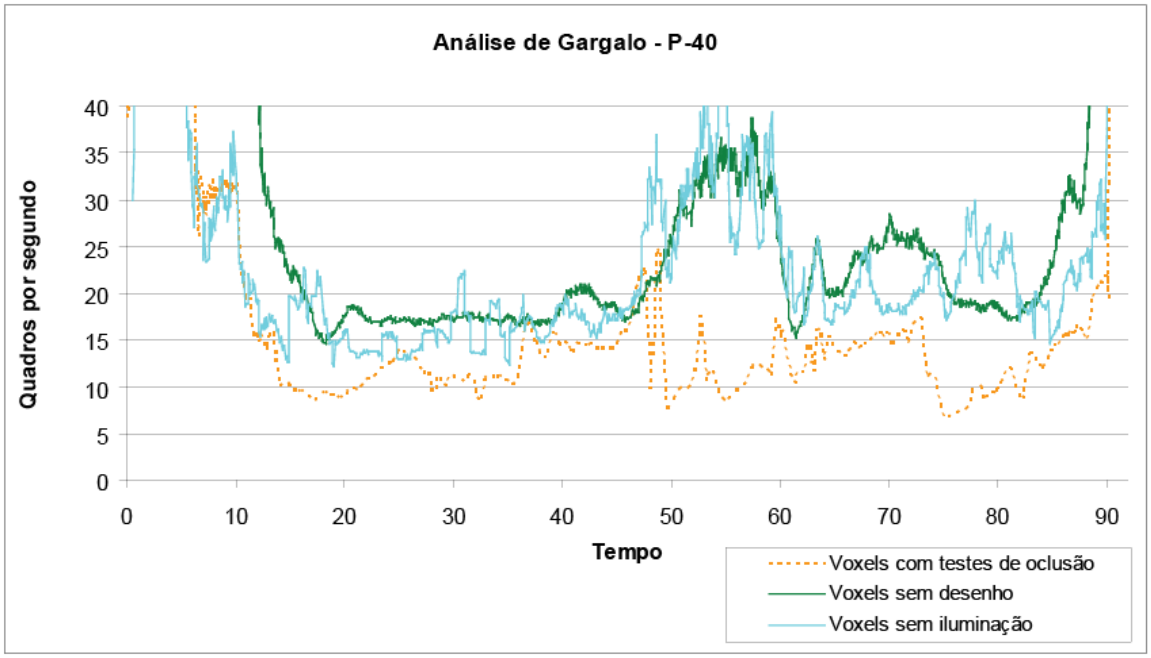

Figura 5.12 – Primeiro gráfico com resultados da análise realizada para localizar o gargalo na navegação da P-40. Esse gráfico compara os resultados obtidos na visualização normal, os resultados obtidos sem que nenhuma primitiva fosse enviada à placa e os resultados obtidos com o estágio de vértice simplificado

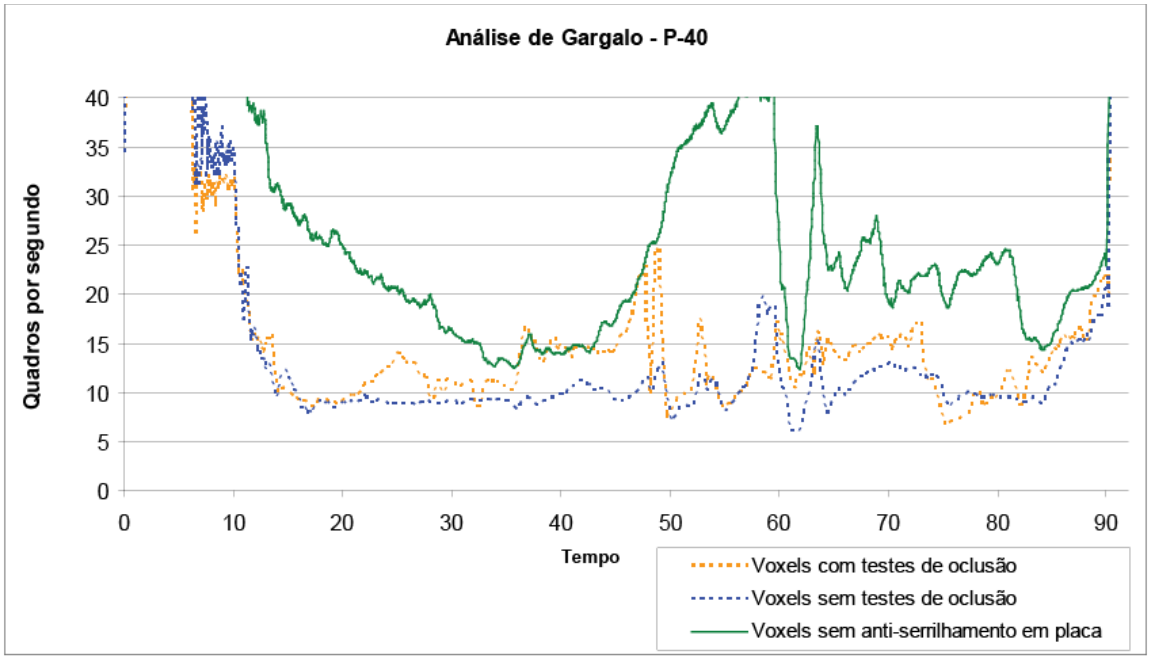

Figura 5.13 – Segundo gráfico com resultados da análise realizada para localizar o gargalo na navegação da P-40. Esse gráfico compara os resultados obtidos na visualização normal com os resultados obtidos sem o uso de anti-serrilhamento em hardware

Pelo gráfico apresentado na Figura 5.12, podemos concluir que o gargalo não está no processamento realizado em CPU, já que ao desabilitarmos o desenho, sempre houve ganho de desempenho. O mesmo gráfico aponta a sobrecarga no estágio de transformação de vértices, que pode ser comprovada pelo ganho de desempenho obtido com a navegação realizada com o estágio de vértices simplificado.

O gráfico apresentado na Figura 5.13 apresenta um resultado bem diferente do observado nos outros dois modelos. Aqui, desligar o anti-serrilhamento trouxe um ganho de desempenho considerável em todos os trechos do caminho percorrido. Assim, podemos afirmar que para a navegação nesse modelo, o desempenho está sendo limitado pela capacidade de rasterização da placa gráfica, além de estar sendo limitado pelo desempenho do estágio de transformação de vértices.

5.2.3
Testes de desempenho com o modelo da P-50

O terceiro modelo testado foi o modelo de CAD da P-50, o modelo com mais polígonos dentre os três testados nessa dissertação. A Figura 5.14 mostra a trajetória da câmera usada durante a navegação, a figura 5.15 mostra imagens tiradas em cada instante da visualização e a Figura 5.16 mostra a taxa de quadros obtida em cada instante da trajetória.

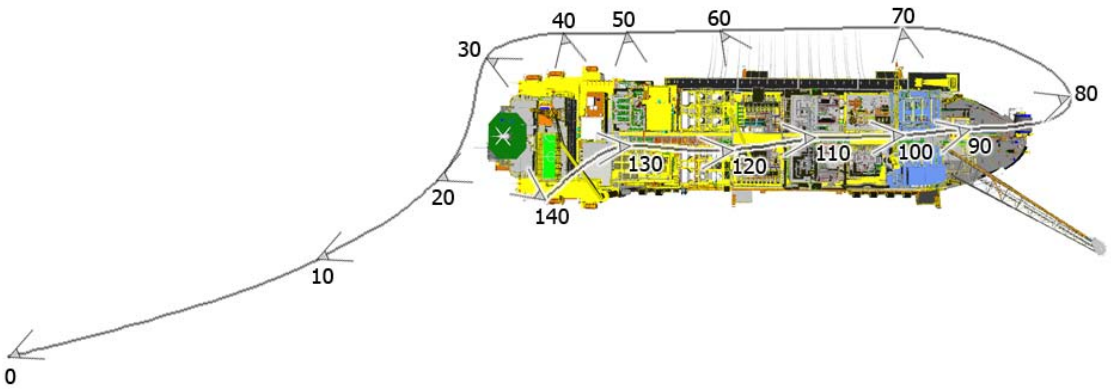

Figura 5.14 - Caminho percorrido durante o teste de desempenho na P-50. Os números indicam onde a câmera está em cada instante da animação

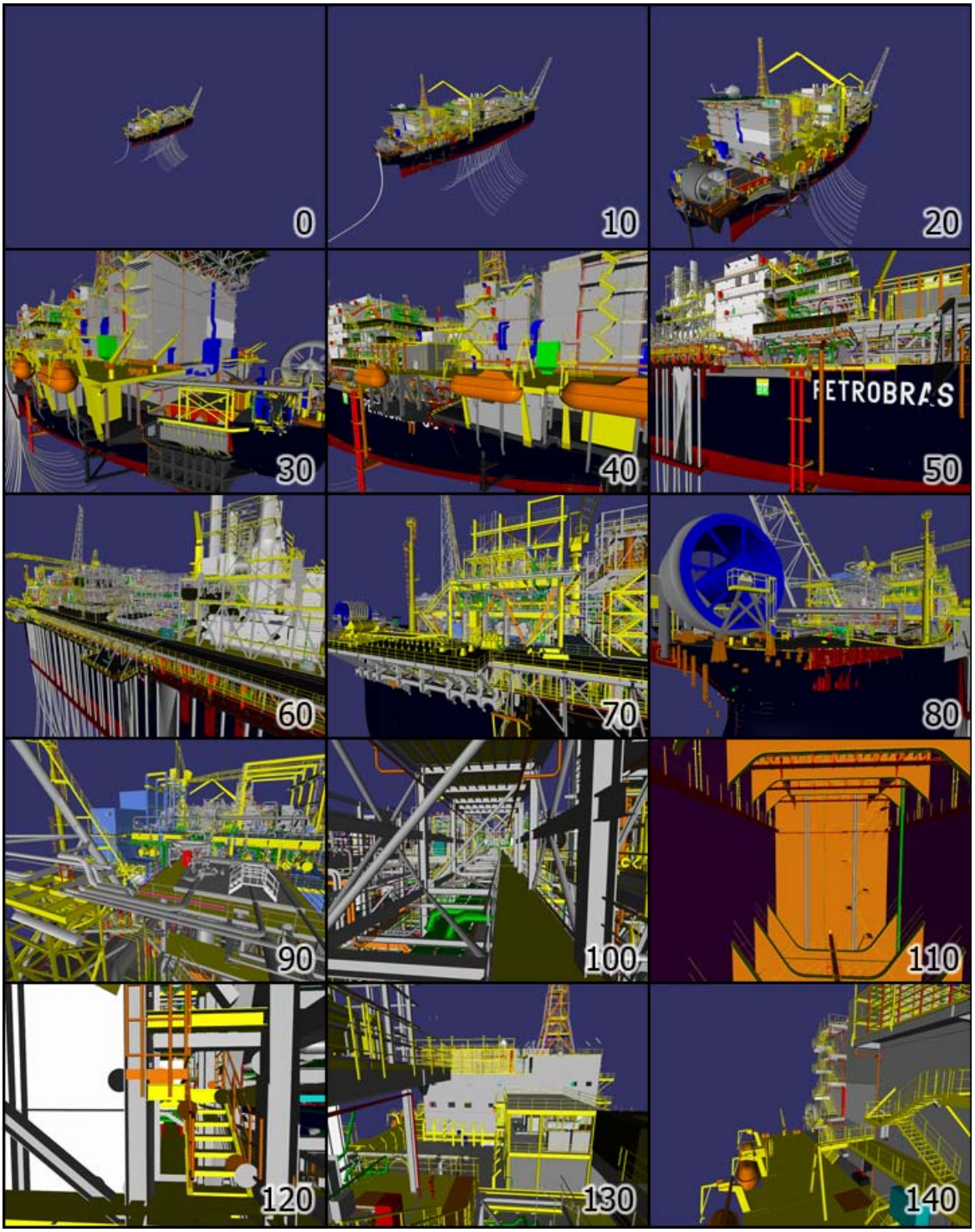

Figura 5.15 – Imagens de cada trecho do caminho percorrido durante os testes na P-50

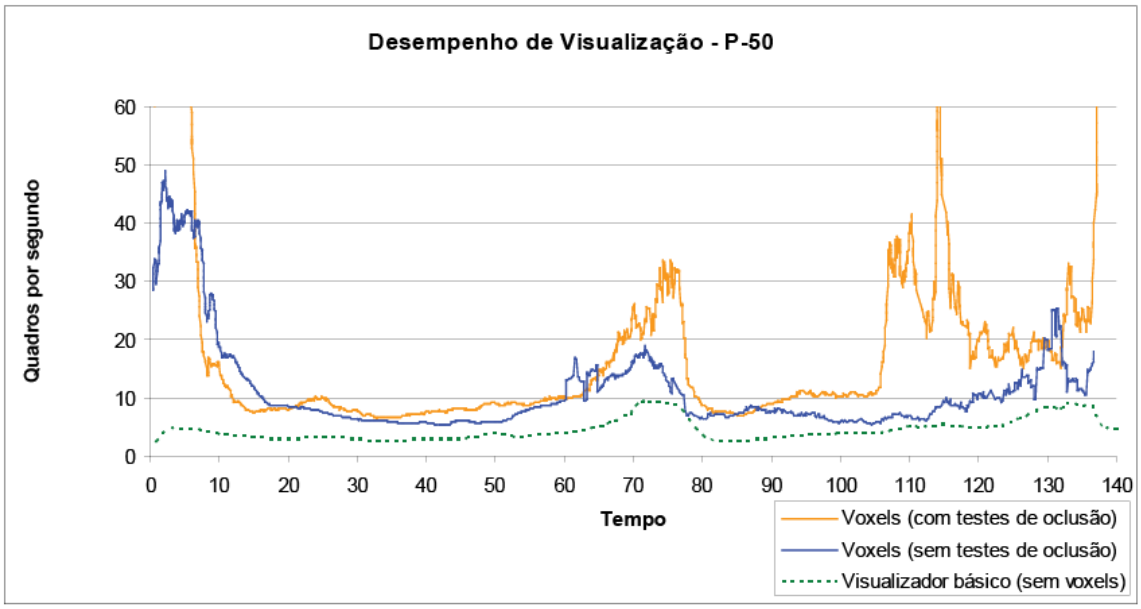

Figura 5.16 - Resultado em quadros por segundo dos testes de desempenho na P-50. Os valores de tempo se relacionam com as posições indicadas na Figura 5.14 e 5.15

O resultado apresentado mostra que, para esse modelo, o algoritmo de Voxels Distantes trouxe um bom ganho de desempenho. Os testes de oclusão realizados contribuíram para melhorar ainda mais o desempenho, principalmente nos trechos em que a navegação ocorreu nas partes internas do modelo, entre os tempos 100 e 120.

Os próximos testes realizados visam identificar em quais etapas da cadeia de renderização ocorreram gargalos durante a navegação nesse modelo. Os resultados são apresentados nas Figuras 5.17 e 5.18 e são analisados a seguir.

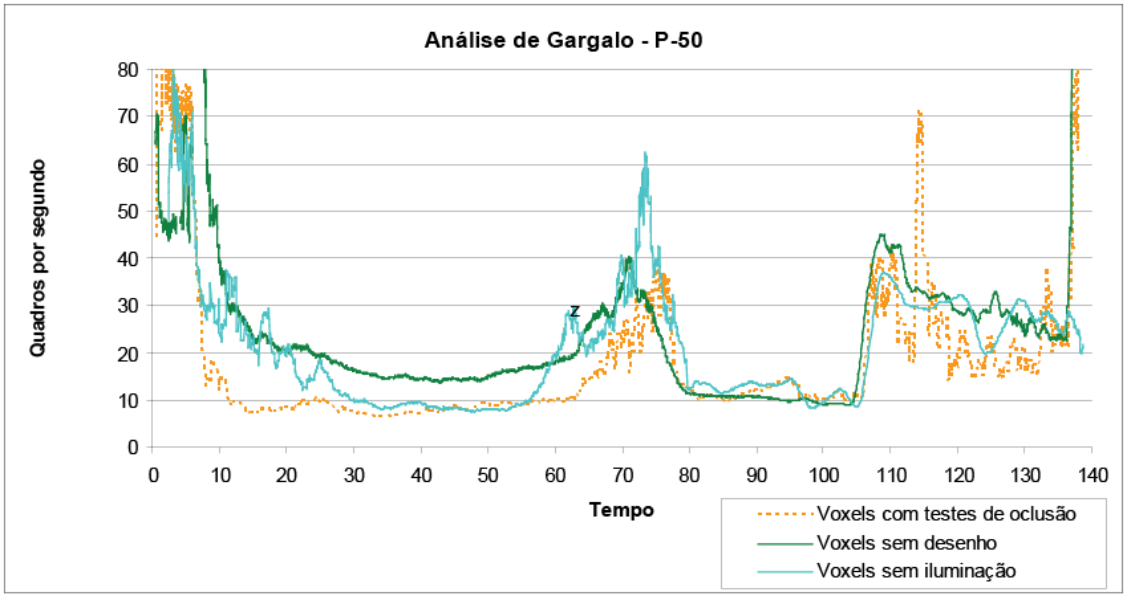

Figura 5.17 – Primeiro gráfico com resultados da análise realizada para localizar o gargalo na navegação da P-50. Esse gráfico compara os resultados obtidos na visualização normal, os resultados obtidos sem que nenhuma primitiva fosse enviada à placa e os resultados obtidos com o estágio de vértice simplificado

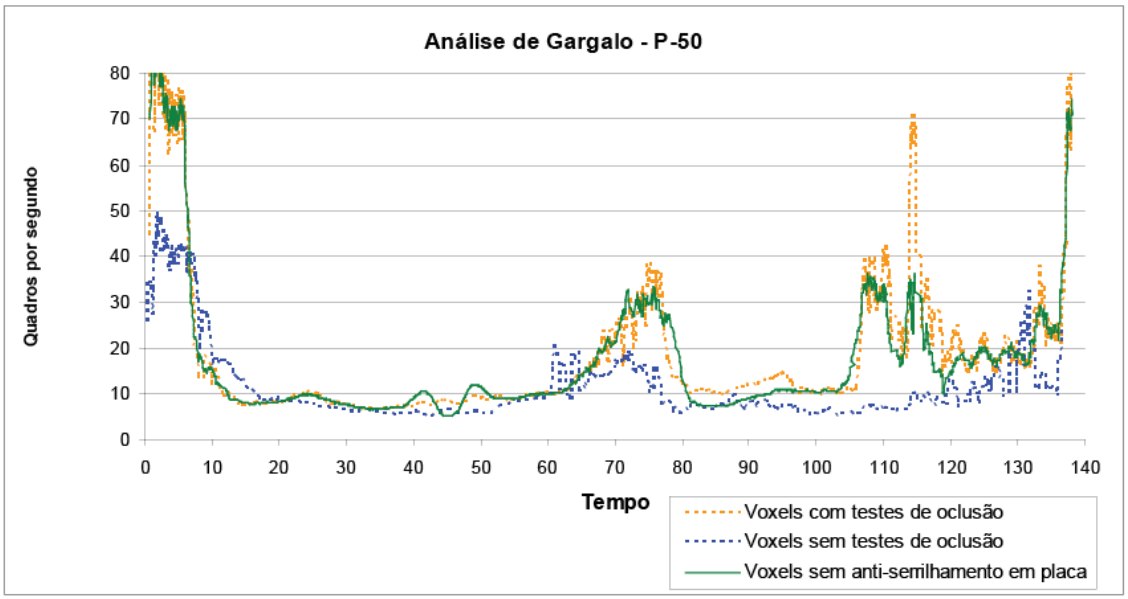

Figura 5.18 – Segundo gráfico com resultados da análise realizada para localizar o gargalo na navegação da P-50. Esse gráfico compara os resultados obtidos na visualização normal com os resultados obtidos sem o uso de anti-serrilhamento em hardware

O gráfico apresentado na Figura 5.17 mostra que na maior parte da navegação, o desempenho não estava sendo limitado pela CPU. Apenas nos instantes entre os tempos 80 e 105, desabilitar o desenho de primitivas não melhorou o desempenho da visualização. Podemos dizer que, nesse trecho, a

aplicação estava sendo limitada pela CPU.

O gráfico apresentado na Figura 5.18 mostra que não houve melhora no desempenho com o anti-serrilhamento da placa gráfica desligado. Isso indica que durante a navegação nesse modelo, a etapa de rasterização da placa gráfica não limitou a visualização.

A navegação realizada com o estágio de transformação de vértices simplificado apresentado na Figura 5.17 confirmou que o gargalo está na CPU no trecho entre os tempos 80 e 105, já que nesse teste, não houve ganho de desempenho. Um resultado inesperado foi encontrado no trecho entre os tempos 30 e 55. Nesse trecho, o visualizador não estava limitado pela CPU, nem pelo estágio de rasterização, e também não houve melhora significativa no desempenho quando o estágio de vértices foi simplificado. Esse resultado pode indicar que, nesse trecho, o desempenho está sendo limitado por algum outro elemento da cadeia de renderização, como por exemplo, a transferência de objetos recém carregados para a placa gráfica.

5.3 Testes de desempenho para os voxels com filtro de anti-serrilhamento

Na seção 3.5, foi apresentado o filtro que descreve um novo tipo de voxel que foi criado para contornar defeitos visuais causados por características encontradas nos modelos usados nesses testes.

Como os voxels criados usam primitivas transparentes e precisam ser renderizados após toda a geometria opaca da cena, é possível que eles tornem a visualização dos modelos que os usem mais lenta. Os testes apresentados nos gráficos das Figuras 5.19, 5.20 e 5.21 comparam o desempenho durante um percurso realizado nos modelos da P-38, P-40 e P-50, respectivamente, usando esse novo tipo de voxels e sem usá-los.

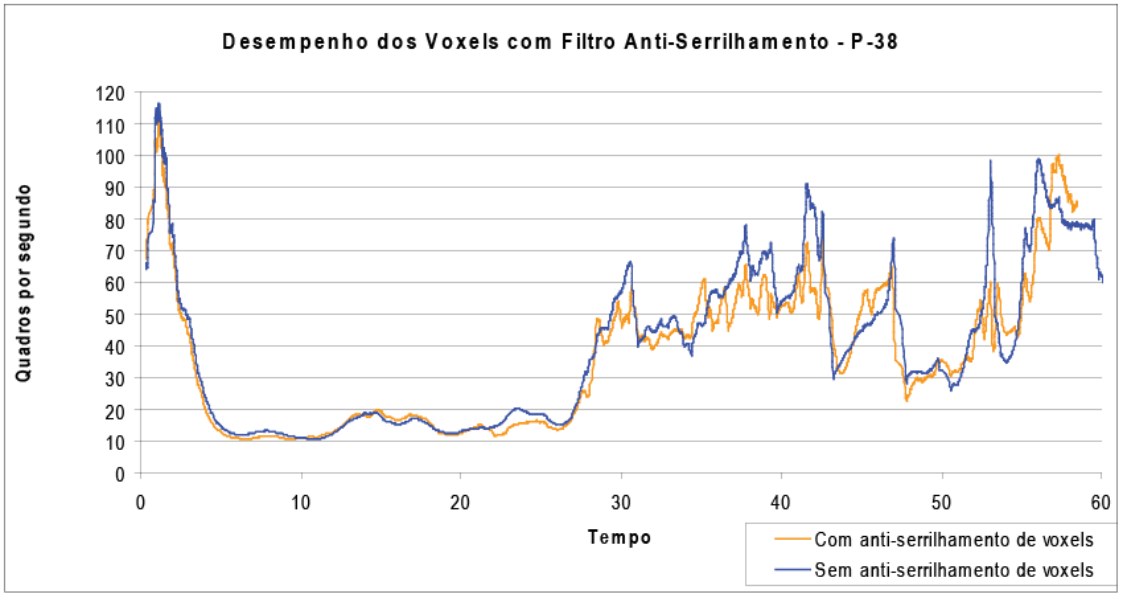

Figura 5.19 – Gráfico comparativo do desempenho usando voxels com o filtro anti-serrilhamento desenvolvido e sem usar esse filtro no modelo da P-38

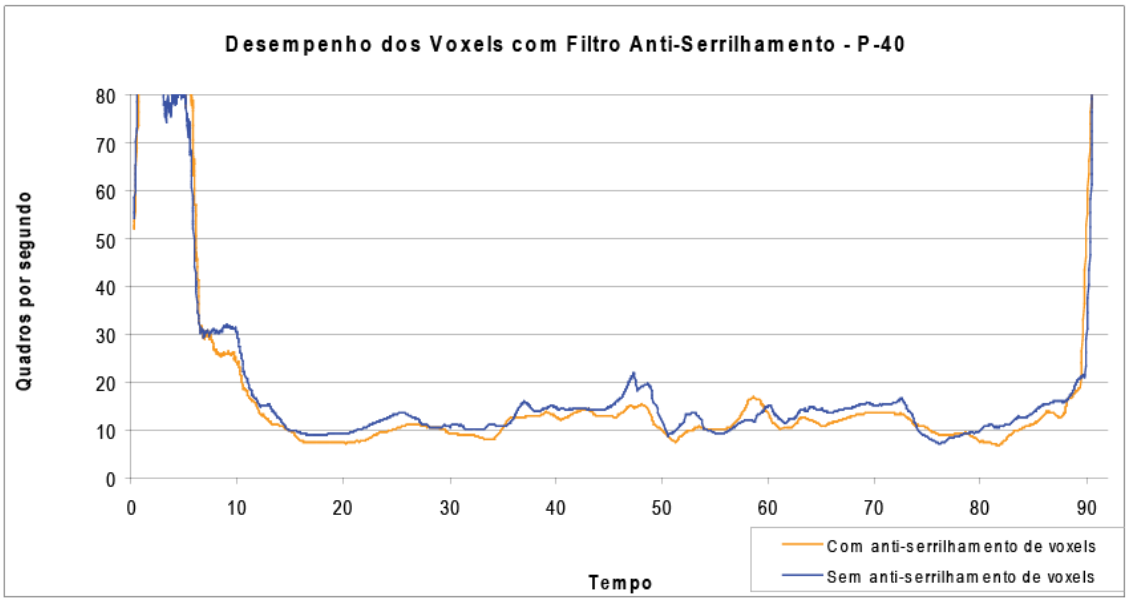

Figura 5.20 – Gráfico comparativo do desempenho usando voxels com o filtro anti-serrilhamento desenvolvido e sem usar esse filtro no modelo da P-40

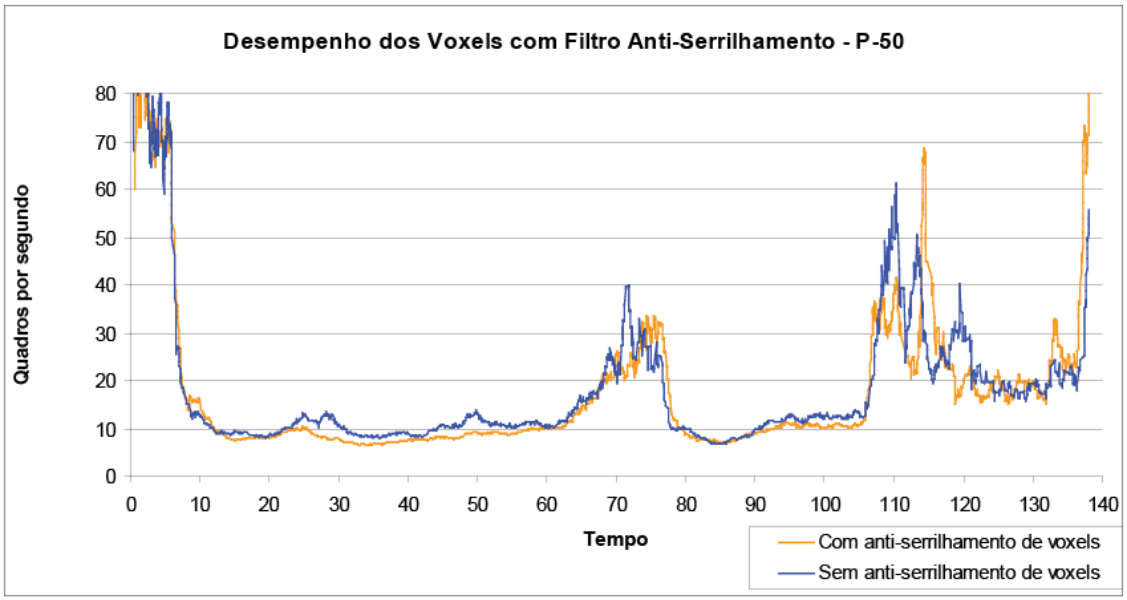

Figura 5.21 – Gráfico comparativo do desempenho usando voxels com o filtro anti-serrilhamento desenvolvido e sem usar esse filtro no modelo da P-50

Os resultados mostram que a perda de desempenho resultante do uso desses voxels é bem pequena, o que torna o seu uso perfeitamente viável na visualização dos modelos usados nesses testes.

5.4 Testes de desempenho com cópias dos modelos

O último teste realizado tem como objetivo verificar a escalabilidade do visualizador para modelos maiores. A existência de um modelo maior foi simulada com a criação de diversas cópias dos modelos existentes. Dessa forma, a complexidade total do modelo é aumentada, mas a densidade de triângulos em cada região do modelo continua a mesma. Um bom visualizador de modelos massivos deve conseguir manter um bom desempenho quando a complexidade do modelo cresce dessa forma.

No primeiro teste, foram criadas 25 cópias da P-50, o nosso modelo mais pesado, posicionadas como ilustrado na Figura 5.22. O caminho percorrido durante a navegação foi realizado em torno da P-50 posicionada no meio da cena, seguindo o caminho usado nos testes de desempenho para uma única P-50, que foi apresentado na seção 5.4. O resultado desse teste é apresentado na Figura 5.23.

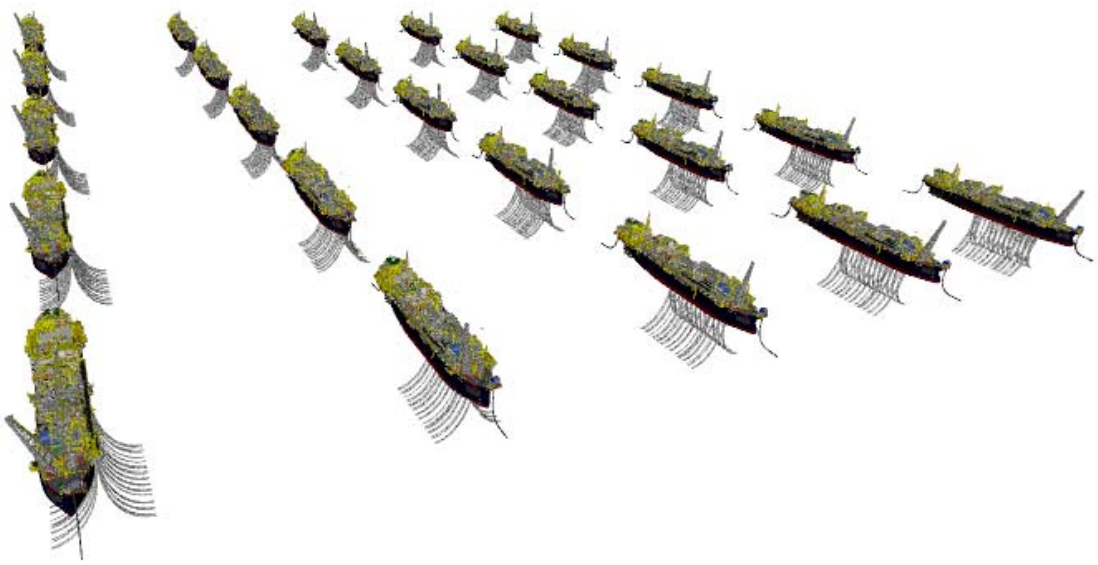

Figura 5.22 – 25 Cópias da P-50 posicionados lado-a-lado

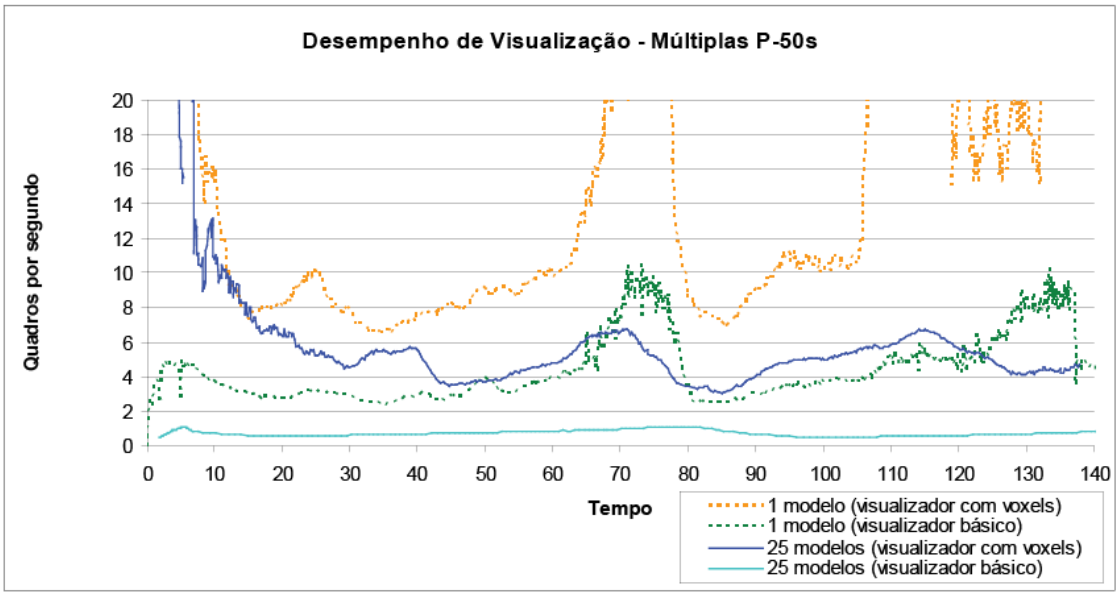

Figura 5.23 – Desempenho obtido com as 25 cópias da P-50

Nesse teste, podemos notar que houve alguma perda de desempenho devido à existência de várias plataformas, mas esta não ocorreu de forma proporcional ao número de cópias criadas. No visualizador básico, por outro lado, essa perda de desempenho foi bem mais acentuada.

No segundo teste realizado, foram criadas 9 cópias de cada uma dos 3 modelos usados nos testes de trabalho, totalizando 27 cópias. A disposição dessas cópias é apresentada na Figura 5.24.

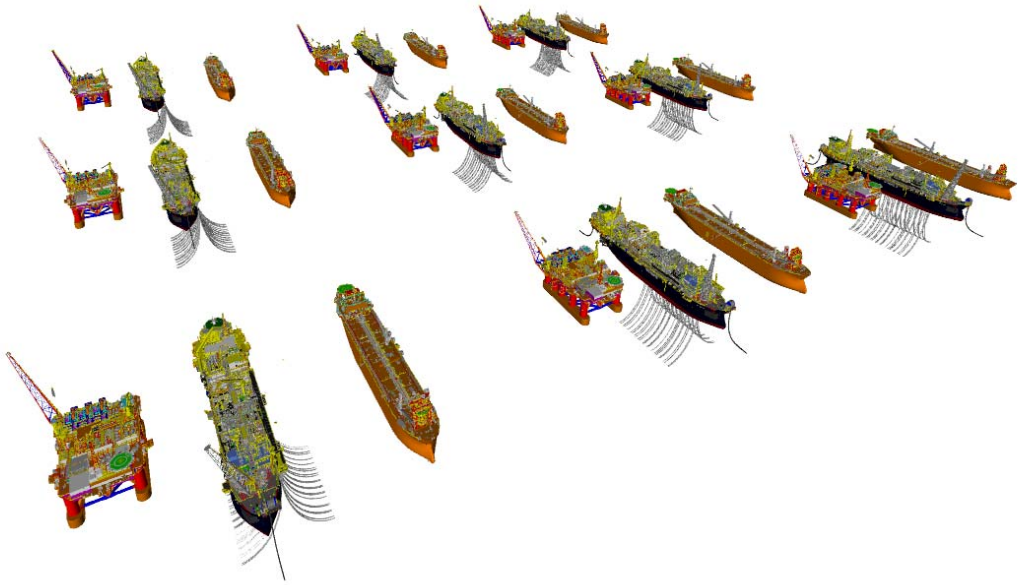

Figura 5.24 – 9 Cópias de cada um dos 3 modelos de teste posicionados lado-a-lado, totalizando 27 modelos

A navegação realizada nesse teste de desempenho também seguiu o mesmo caminho realizado no teste de desempenho com apenas uma P-50. Os modelos foram posicionados de forma que a navegação fosse realizada em torno da P-50 que está localizada no centro de todos os outros modelos. Na Figura 5.25, mostramos os resultados desse teste comparados com o resultado obtido com apenas um modelo da P-50. Como a versão básica do visualizador não é capaz de paginar dados do disco, não foi possível usá-lo nesse teste com diversos modelos diferentes.

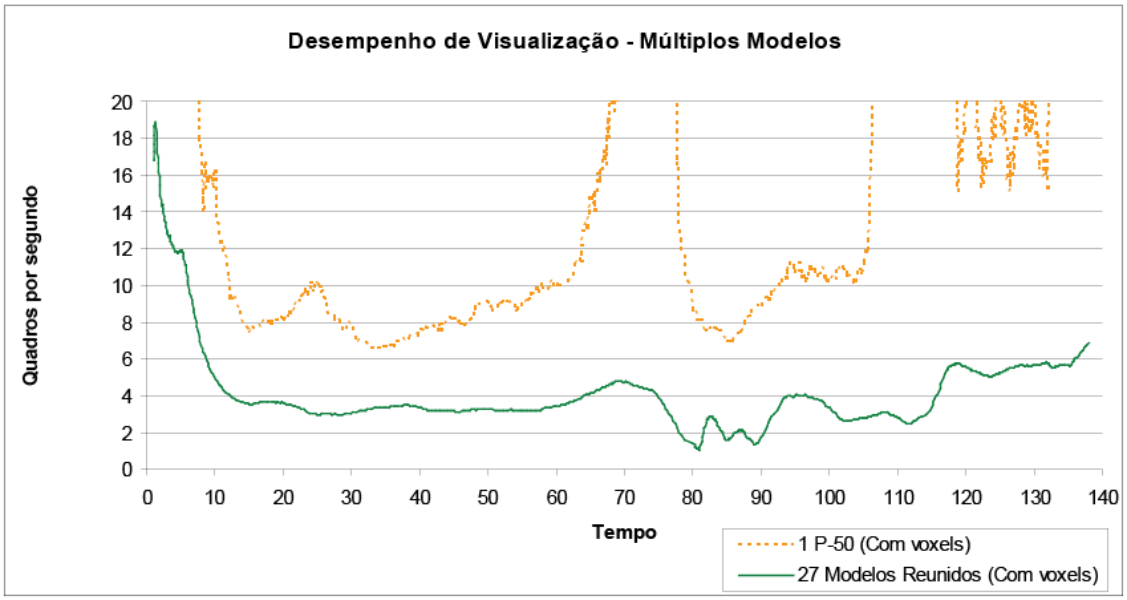

Figura 5.25 – Desempenho obtido com os 27 modelos.

Pelo gráfico da Figura 5.25, podemos notar que há alguma perda de desempenho quando comparamos essa navegação com a navegação realizada com apenas um modelo da P-50. Por outro lado, essa perda foi pequena se considerarmos o aumento de complexidade total da cena.