5
Análise experimental e resultados

Neste capítulo, são apresentados os experimentos realizados neste trabalho, com o propósito de avaliar o desempenho do algoritmo proposto.

Os experimentos descritos neste capítulo foram divididos em três grupos. O primeiro tem como foco a avaliação do impacto da substituição dos atributos de forma previstos no algoritmo original de Baatz e Schäpe (2002) pelos novos atributos propostos na seção 4.4.

O segundo grupo tem o intuito de demonstrar que eventuais variações que se observam nos resultados produzidos pela versão paralela em relação à versão seqüencial do algoritmo são exclusivamente causadas pela alteração da ordem de visitação dos segmentos, e que, portanto as versões paralelas e seqüenciais são funcionalmente equivalentes.

O terceiro grupo de experimentos visa estimar o ganho de desempenho (*speedup*) em função do tempo de processamento das versões paralelas em relação às seqüenciais.

5.1. Impacto da substituição de atributos de forma

O objetivo destes experimentos é averiguar se a alteração dos atributos que compõem a medida de heterogeneidade, conforme proposto na seção 4.4 não afeta substancialmente o resultado final da segmentação. Para tanto, é necessário avaliar a qualidade dos resultados da segmentação.

Dois atributos serão considerados "similares" se for possível obter resultados similares de segmentação a partir de funções de heterogeneidade formuladas em termos de um ou do outro.

Cabe ressaltar que, ao alterar os atributos que compõem a função de heterogeneidade, é necessário reajustar os parâmetros do algoritmo de segmentação para que este possa produzir resultados semelhantes.

Nesta dissertação, a análise de semelhança é auxiliada pela utilização de uma função de disparidade que representa a diferença entre o resultado obtido

pela segmentação e uma dada imagem de referência. Além disso, optou-se pelo emprego de algoritmos genéticos para realizar o ajuste dos parâmetros de segmentação.

Para execução destes experimentos foi adotado o seguinte procedimento:

1. Selecionam-se conjuntos de segmentos de referência em uma imagem de teste.
2. Os parâmetros do algoritmo de segmentação são ajustados de modo a maximizar a coerência entre o resultado do algoritmo e os segmentos de referência.
3. Os passos 1 e 2 são repetidos para os dois grupos de atributos
4. Mede-se finalmente a similaridade entre as segmentações produzidas no passo 2 para cada grupo de atributos

5.1.1.Base de dados

Foi utilizado um recorte de uma imagem QuickBird de coordenadas (S 23º 13’ 33”; W 45º 53’ 06”) e (S 23º 13’ 22”; W 45º 53’ 14”) apresentado na Figura 5.1. As cinco bandas originais foram fundidas em três aplicando a técnica de análise dos componentes principais.

Figura 5.1. Recorte utilizado para avaliação dos atributos de forma.

Três diferentes conjuntos de referência foram definidos nesta imagem. O primeiro contém objetos considerados espectralmente homogêneos, o segundo

apresenta objetos espectralmente heterogêneos e o terceiro é composto por uma mescla dos conjuntos anteriores. A Figura 5.2 expõe as imagens de referência.

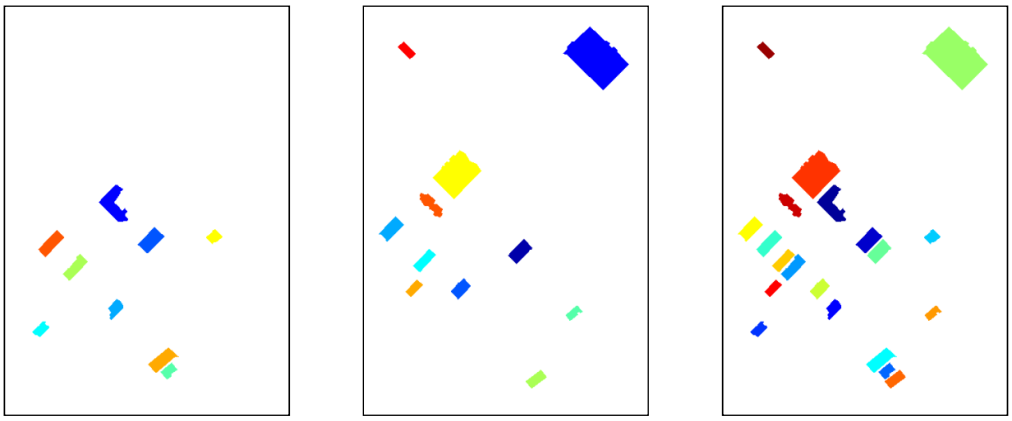


Figura 5.2. Conjuntos de referências: homogêneos (esq.), heterogêneos (centro) e mesclados (dir.)

5.1.2. Ferramenta de avaliação

Para a execução desses experimentos foi utilizada uma ferramenta, denominada SPT (*Segmentation Parameters Tuner*) (SPT, 2010), que realiza o ajuste automático dos parâmetros de segmentação através de algoritmo genético.

Neste experimento, a função de disparidade escolhida foi a RBSB (*Reference Bounded Segments Booster*) (Feitosa et al., 2006; Costa et al., 2008), que já se encontra disponível neste *software*.

5.1.3. Resultados e discussões

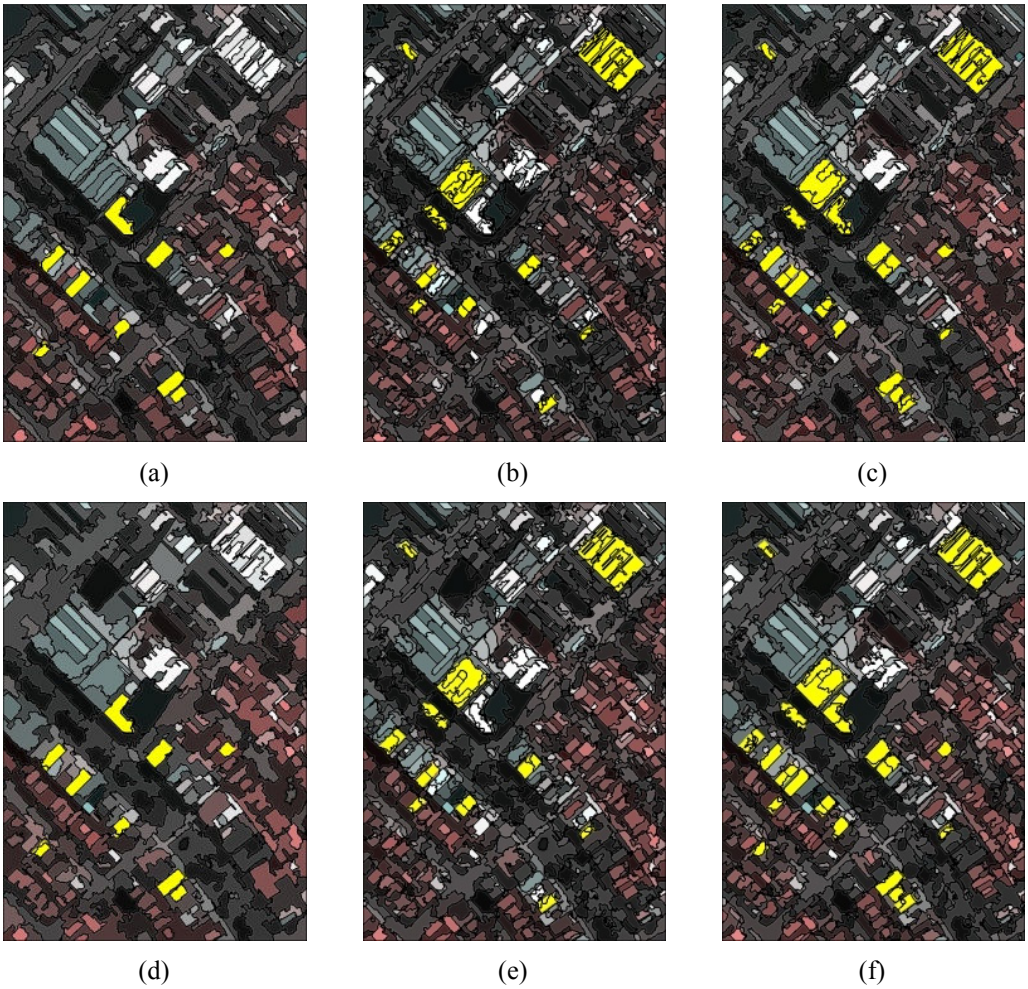
A Tabela 5.1 exprime o melhor valor obtido pela função de disparidade para cada par de atributos de acordo com os tipos de objetos da referência. Pode-se observar que para os três grupos de objetos, a diferença em relação aos dois pares de atributos é pequena. Isto demonstra que é possível ajustar a função de heterogeneidade baseada nos novos atributos propostos de modo a produzir um resultado semelhante ao que se obtém aplicando a versão do algoritmo baseada nos atributos originais.

A Figura 5.3 permite avaliar visualmente a semelhança entre resultados da segmentação obtidos para cada grupo de objetos. As ilustrações (a), (b) e (c) se referem respectivamente à utilização dos parâmetros de compacidade e suavidade

para grupos homogêneos, heterogêneos e mesclados. Já as imagens (d), (e) e (f) são relativas à utilização dos novos parâmetros para objetos homogêneos, heterogêneos e mesclados, respectivamente. As referências são apresentadas na cor amarela.

Parâmetros utilizados	Valor de disparidade por tipo de objetos		
	Homogêneos	Heterogêneos	Mesclados
$Comp$ e Svd	0,14	0,56	0,46
$Comp_{Russ}$ e Sol_{Russ}	0,16	0,57	0,40

Tabela 5.1. Valor da disparidade por parâmetros utilizados e tipos de objetos



(a) (b) (c)

(d) (e) (f)

Figura 5.3. Resultado da segmentação utilizando os parâmetros $\boldsymbol{Comp}$ e $\boldsymbol{Svd}$ para objetos homogêneos (a), heterogêneos (b) e mesclados (c) e utilizando os parâmetros $\boldsymbol{Comp_{Russ}}$ e $\boldsymbol{Sol_{Russ}}$ também para objetos homogêneos (d), heterogêneos (e) e mesclados (f).

Nota-se claramente a semelhança entre (a) e (d) e ao mesmo tempo verificar que, em ambas, os objetos de referência estão bem representados pelos segmentos da imagem. Observa-se também certa similaridade entre as representações (b) e (e) que resultam em segmentos menores, mas que podem ser aglomerados para compor os objetos de referência. O mesmo ocorre para (c) e (f), que se assemelham entre si.

Estes resultados corroboram a expectativa de que os atributos de suavidade e de compacidade podem ser substituídos pelo atributo de solidez e por outra definição do atributo de compacidade sem alteração substancial nos resultados produzidos pelo algoritmo de segmentação.

5.2. Comparação das segmentações seqüenciais e paralelas

Outro aspecto a ser avaliado é a coerência entre os resultados da segmentação obtidos pela versão paralela e pela versão seqüencial. Este experimento tem como objetivo ponderar acerca da alteração dos resultados da segmentação e evidenciar que eventuais diferenças ocorrem devido à mudança na ordem de visitação dos segmentos.

5.2.1.Base de dados

Foi utilizado um recorte de uma imagem IKONOS, chamada Refinaria, conforme ilustrado na Figura 5.4.

Figura 5.4. Imagem denominada Refinaria

5.2.2. Ferramentas de avaliação

Para determinar o grau de discrepância provocado pela alteração na ordem de visitação em algoritmos de crescimento de região foram executados dois segmentadores amplamente utilizados pela comunidade de sensoriamento remoto: o disponível no Definiens (Definiens, 2008) e o SPRING (SPRING, 2011).

Primeiramente houve uma segmentação da imagem Refinaria. Em seguida, foi realizado um rebatimento horizontal desta imagem (Figura 5.5) e foi aplicado o segmentador novamente com os mesmos parâmetros. Este processo foi realizado para ambos os segmentadores.

A seguir, foram então executadas a versão seqüencial e a paralela do algoritmo descrito nesta dissertação com a heurística de melhor vizinho.

Figura 5.5. Imagem Refinaria após rebatimento horizontal

5.2.3.Resultados e discussões

A Figura 5.6 exprime o resultado da segmentação para a imagem original e a imagem rebatida utilizando o Definiens. Já a Figura 5.7 apresenta os resultados gerados pelo SPRING. Os retângulos definidos em azul enfocam 10 áreas cuja segmentação apresenta diferença.

Desta maneira, é possível observar, em ambos os casos, as diferenças entre a segmentação da imagem original e a segmentação da imagem rebatida. Portanto, por mais que se trate da mesma imagem, dos mesmos segmentadores e dos mesmos parâmetros, a mudança na ordem de visitação dos segmentos afetou o resultado.

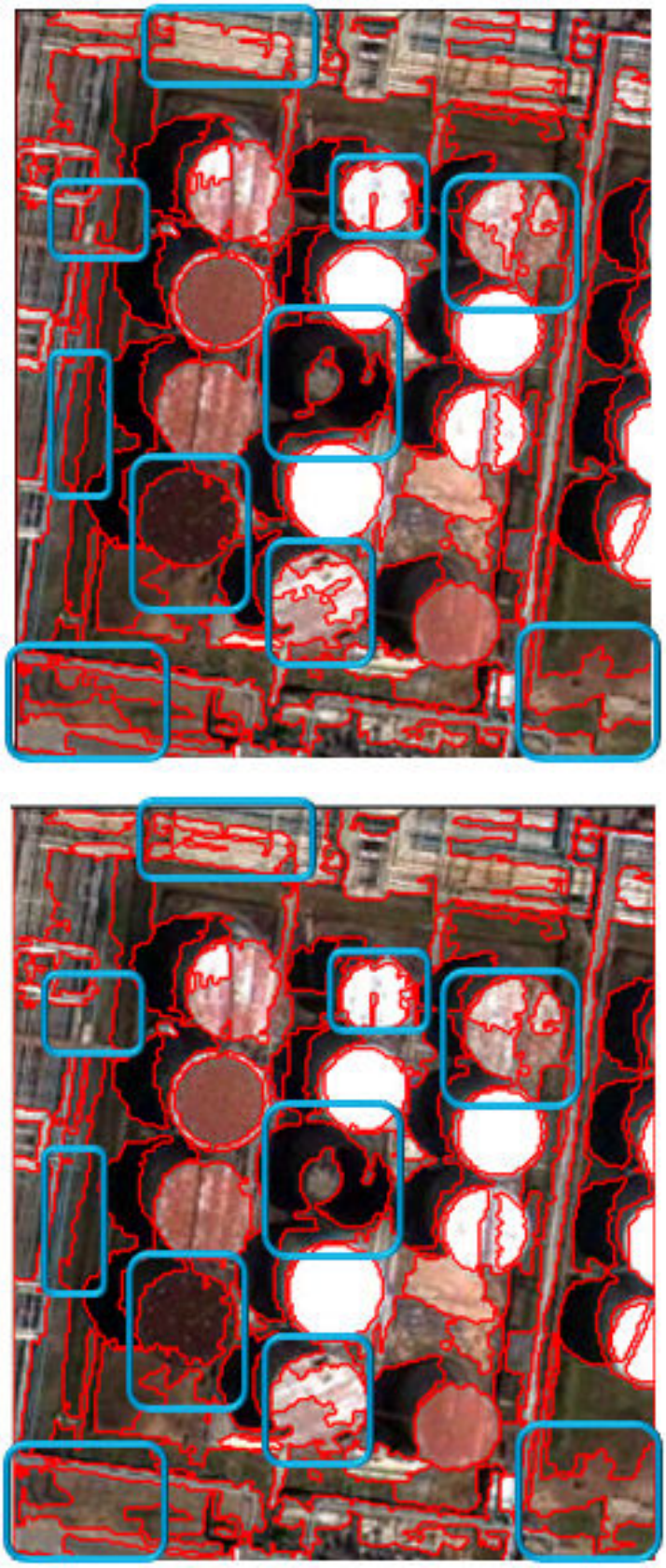


Figura 5.6. Resultado da segmentação no Definiens para imagem original (cima) e rebatida (baixo). Retângulos em azul representam áreas com diferenças.

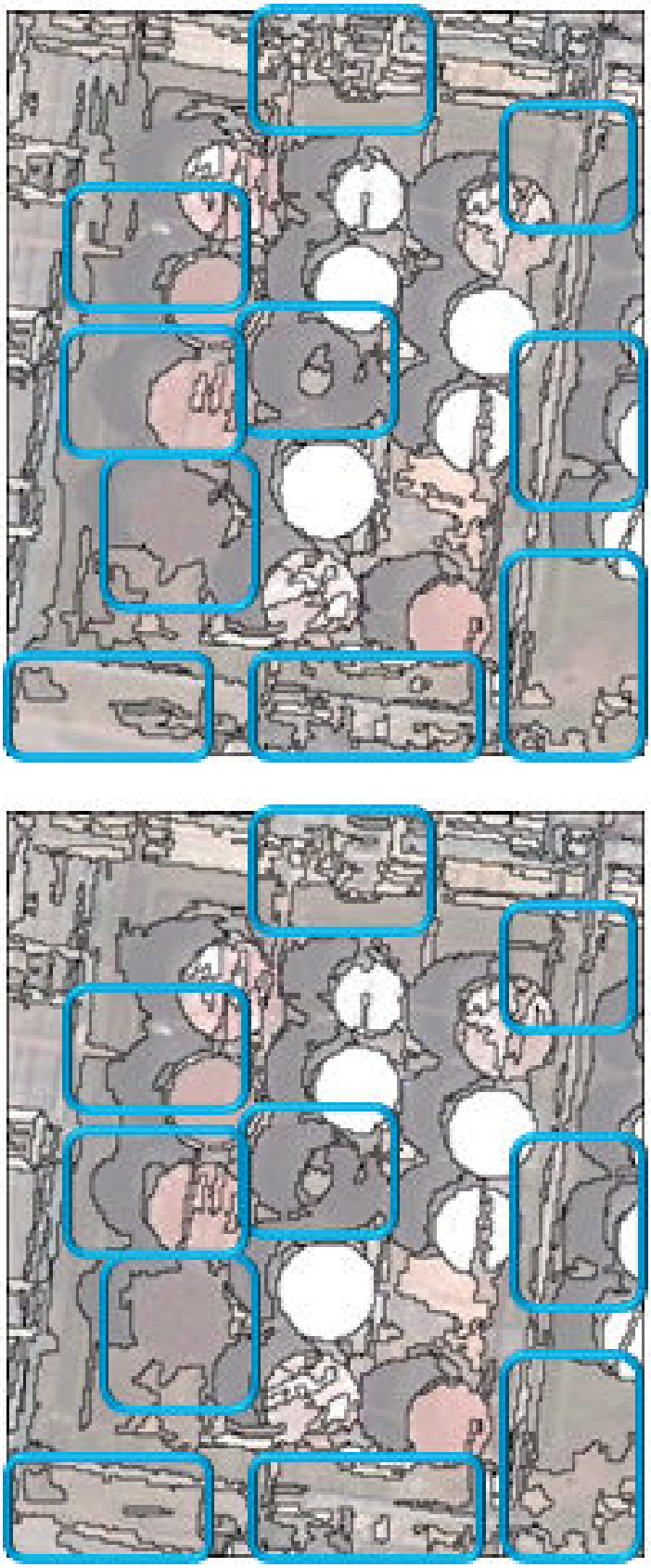


Figura 5.7. Resultado da segmentação no SPRING para imagem original (cima) e rebatida (baixo). Retângulos em azul representam áreas com diferenças.

A Figura 5.8 ilustra diferenças entre a versão seqüencial do algoritmo e a versão paralela por melhor vizinho. As discrepâncias observadas decorrem do processamento independente das *threads*, que pode acarretar em variação na ordem de visitação dos segmentos.

Os experimentos demonstraram, portanto, que as diferenças entre a versão seqüencial e paralela são semelhantes às discrepâncias que ocorrem em versões seqüenciais quando se altera a ordem de visitação.

Figura 5.8. Resultado da segmentação no segmentador proposto: versão seqüencial (cima) e paralela (baixo). Retângulos em azul representam áreas com diferenças.

5.3. Aceleração das versões paralelas em relação às seqüenciais

Estes experimentos têm por objetivo avaliar o ganho de desempenho computacional das duas versões paralelas do algoritmo (PBF e PLMBF), executadas na GPU em relação às respectivas versões seqüenciais.

5.3.1. Ambientes de teste

No intuito de realizar uma análise mais completa, os experimentos foram executados em dois ambientes distintos com diferentes processadores, GPUs e sistemas operacionais.

A idéia contempla, portanto, uma configuração básica de GPU de baixo custo e uma configuração mais avançada de um custo mais elevado.

A primeira configuração utilizada, referida deste ponto em diante no texto como Win_9600, possui uma GPU NVIDIA GeForce 9600 GT com 64 núcleos e 1GB de memória para o processamento paralelo. O Sistema Operacional em execução é o Windows XP e o processador é um Core 2 6300 operando a 1.86Ghz e 3.25GB de RAM.

A segunda configuração, denominada Lnx_Tesla, possui uma GPU NVIDIA TESLA C1060 com 240 núcleos e 4GB de memória. O sistema operacional é GNU/Linux e o processador é um Intel Xeon 2.5Ghz e 7.8GB de RAM.

5.3.2.Base de dados

No primeiro teste utilizou-se um recorte de uma imagem aérea rotulada, nesta dissertação, de Resende_4300 e disposta na Figura 5.9. Este recorte compreende uma área do bairro Jardim Tropical situado na cidade de Resende. A partir deste, foram extraídos outros recortes com a finalidade de gerar imagens de diferentes tamanhos conforme exposto na Tabela 5.2. Cabe informar que os centros dos recortes referem-se à mesma coordenada geográfica em todos os casos.

Figura 5.9. Imagem de teste Resende_4300

Sigla	Colunas	Linhas	Total de *Pixels*
Resende_500	500	500	250000
Resende_1000	1000	1000	1000000
Resende_1500	1500	1500	2250000
Resende_2000	2000	2000	4000000
Resende_2500	2500	2500	6250000
Resende_2800	2800	2800	7840000
Resende_3500	3500	3500	12250000
Resende_3900	3900	3900	15210000
Resende_4300	4300	4300	18490000

Tabela 5.2. Recortes da imagem Resende e seus respectivos tamanhos

Para um maior volume de experimentos, foram utilizadas ainda outras cinco imagens: Quadra, Serra, Morros, Maracanã e Rio das Pedras respectivamente apresentadas nas Figuras 5.10, 5.11, 5.12, 5.13 e 5.14. A tabela 5.3 indica o tamanho das imagens em questão.

Sigla	**Colunas**	**Linhas**	**Total de *Pixels***
Quadra	1000	1000	1000000
Serra	1600	1600	2560000
Morros	2311	2086	4820746
Maracanã	2732	2456	6709792
Rio das Pedras	4200	4200	17640000

Tabela 5.3. Imagens de teste e seus respectivos tamanhos

Quadra é um recorte de uma imagem QuickBird. Serra e Morros são recortes de imagens IKONOS. Maracanã e Rio das Pedras são imagens aéreas analógicas pancromáticas que foram digitalizadas respectivamente a 300 DPI e a 600 DPI.

Quadra representa uma área no bairro Brooklin Novo na cidade de São Paulo. Morros é uma área entre os bairros de Santa Tereza e Laranjeiras na cidade do Rio de Janeiro. Maracanã é uma área que compreende o bairro Maracanã na cidade do Rio de Janeiro. Rio das Pedras representa uma área do bairro homônimo na cidade do Rio de Janeiro.

Cabe ressaltar que para todas as imagens de teste foram consideradas apenas três bandas espectrais.

Figura 5.10. Imagem de teste Quadra

Figura 5.11. Imagem de teste Serra

Figura 5.12. Imagem de teste Morros

Figura 5.13. Imagem de teste Maracanã

Figura 5.14. Imagem de teste Rio das Pedras

5.3.3. Configuração de *threads* por blocos

A escolha da configuração de execução dos *kernels* da versão paralela é crucial em termos de desempenho, pois irá determinar o escalonamento dos conjuntos de *threads*. Desta forma, deve-se selecionar adequadamente o número de *threads* por bloco para cada *kernel* existente. Não existe, porém, um procedimento teórico que determine a configuração ótima. Destarte, a abordagem adotada foi essencialmente empírica.

O tempo de processamento para cada *kernel* foi medido para 4, 8, 16, 32, 64 e 128 *threads* por bloco para a imagem Morros. Esta análise foi realizada para ambas as versões paralelas nos dois ambientes de teste. Os resultados desta

avaliação são apresentados na Tabela 5.4 (PBF em Win_9600), 5.5 (PLMBF em Win_9600), 5.6 (PBF em Lnx_Tesla) e 5.7 (PLMBF em Lnx_Tesla).

Kernel	Tempo de Execução em (ms) para PBF em Win_9600					
	4	8	16	32	64	128
Inicializar Segmentos	167,63	157,40	140,48	**137,29**	137,40	148,21
Encontrar Vizinhos	27274,49	17061,62	10698,65	7798,20	**6094,20**	7992,56
Realizar Fusões	5282,41	3206,70	1970,27	1353,82	**1075,54**	1139,47
Redefinir Segmentos	631,11	402,45	328,06	329,82	**324,54**	326,65
Recalcular Bordas	2074,13	1159,75	649,76	411,41	**353,74**	355,07
Escrever Imagem	54,18	42,50	34,61	33,06	**31,75**	31,78

Tabela 5.4. Tempo de Execução por número de *threads* por bloco para PBF em Win_9600.

Kernel	Tempo de Execução em (ms) para PLMBF em Win_9600					
	4	8	16	32	64	128
Inicializar Segmentos	174,35	168,48	148,40	148,69	**147,24**	149,28
Encontrar Vizinhos	134376,70	85516,48	56785,30	41294,74	**32708,44**	44056,37
Realizar Fusões	14989,30	8901,10	5328,93	3424,54	**2432,36**	2752,65
Redefinir Segmentos	2915,10	2297,70	2227,11	**2218,91**	2238,43	2266,80
Recalcular Bordas	9417,41	5034,59	2814,35	1752,30	**1532,19**	1520,94
Escrever Imagem	52,70	42,36	34,81	33,37	**31,71**	32,51

Tabela 5.5. Tempo de Execução por número de *threads* por bloco para PLMBF em Win_9600.

Kernel	Tempo de Execução em (ms) para PBF em Lnx_Tesla					
	4	8	16	32	64	128
Inicializar Segmentos	82,84	**64,77**	68,34	69,35	74,50	76,26
Encontrar Vizinhos	9935,38	5735,02	4023,45	3278,03	**3263,16**	3562,89
Realizar Fusões	2530,58	1469,33	974,00	805,30	**720,18**	741,40
Redefinir Segmentos	303,55	**180,01**	185,19	196,56	211,86	227,61
Recalcular Bordas	1506,99	761,30	471,28	**282,87**	309,44	316,04
Escrever Imagem	16,18	8,79	**7,11**	7,780	8,23	8,19

Tabela 5.6. Tempo de Execução por número de *threads* por bloco para PBF em Lnx_Tesla.

Kernel	Tempo de Execução em (ms) para PLMBF em Lnx_Tesla					
	4	8	16	32	64	128
Inicializar Segmentos	85,63	**71,11**	77,08	77,28	81,85	82,04
Encontrar Vizinhos	59410,22	37990,61	25140,31	19741,69	**18739,00**	23075,93
Realizar Fusões	7249,88	4222,49	2541,66	1798,04	**1502,30**	1553,88
Redefinir Segmentos	1647,06	**1534,01**	1563,92	1655,17	1770,61	1809,74
Recalcular Bordas	7107,65	4093,45	2403,83	**1405,79**	1532,84	1547,39
Escrever Imagem	16,39	8,96	**7,47**	8,00	8,45	8,24

Tabela 5.7. Tempo de Execução por número de *threads* por bloco para PLMBF em Lnx_Tesla.

A partir destes resultados é possível definir a melhor configuração para cada versão em cada ambiente. Esta configuração foi então escolhida para ser utilizada

na avaliação do ganho de desempenho das versões paralelas em relação às versões seqüenciais. A tabela 5.8 apresenta esta configuração.

Kernel	Número de *threads* por bloco por ambiente e método			
	Win_9600		Lnx_Tesla	
	PBF	PLMBF	PBF	PLMBF
Inicializar Segmentos	32	64	8	8
Encontrar Vizinhos	64	64	64	64
Realizar Fusões	64	64	64	64
Redefinir Segmentos	64	32	8	8
Recalcular Bordas	64	64	32	32
Escrever Imagem	64	64	16	16

Tabela 5.8. Número de *threads* por bloco por ambiente e método para cada *kernel.*

5.3.4. Parâmetros de segmentação

Os segmentadores possuem parâmetros através dos quais é possível ajustar o resultado produzido aos objetivos da aplicação. No entanto, para este grupo de experimentos optou-se por manter uma parametrização constante de modo a não criar outra variável independente que pudesse influenciar nos resultados. Esta lista de parâmetros é definida pela Tabela 5.9 e foi selecionada empiricamente.

Parâmetro	Valor
Parâmetro de escala	70
Peso da cor em relação à forma	0.80
Peso do atributo $Comp_{Russ}$ em relação ao Sol_{Russ}	0.50
Peso da banda 1 em relação ao total	0.33
Peso da banda 2 em relação ao total	0.33
Peso da banda 3 em relação ao total	0.34

Tabela 5.9. Valores utilizados para os parâmetros de segmentação.

5.3.5. Resultados e discussões

Os resultados obtidos nestes experimentos são apresentados em gráficos que indicam a razão entre o tempo de processamento de uma versão seqüencial em relação ao tempo de processamento de sua respectiva versão paralela. Esta razão, conhecida como *speedup*, corresponde à aceleração e serve para avaliar o ganho de desempenho adquirido pelo processamento paralelo.

Ressalta-se que os valores apresentados neste trabalho são relativos à média dos resultados de cinco experimentos para cada teste realizado.

Primeiramente, foi utilizado o grupo de recortes da imagem Resende para verificar a aceleração alcançada pela paralelização do segmentador. O resultado destes experimentos no ambiente Win_9600 é mostrado na Figura 5.15 para a versão com heurística de melhor vizinho (PBF) e na Figura 5.16 para a de melhor vizinho mútuo (PLMBF). Já as Figuras 5.17 e 5.18 apresentam as acelerações obtidas na Lnx_Tesla para PBF e PLMBF respectivamente.

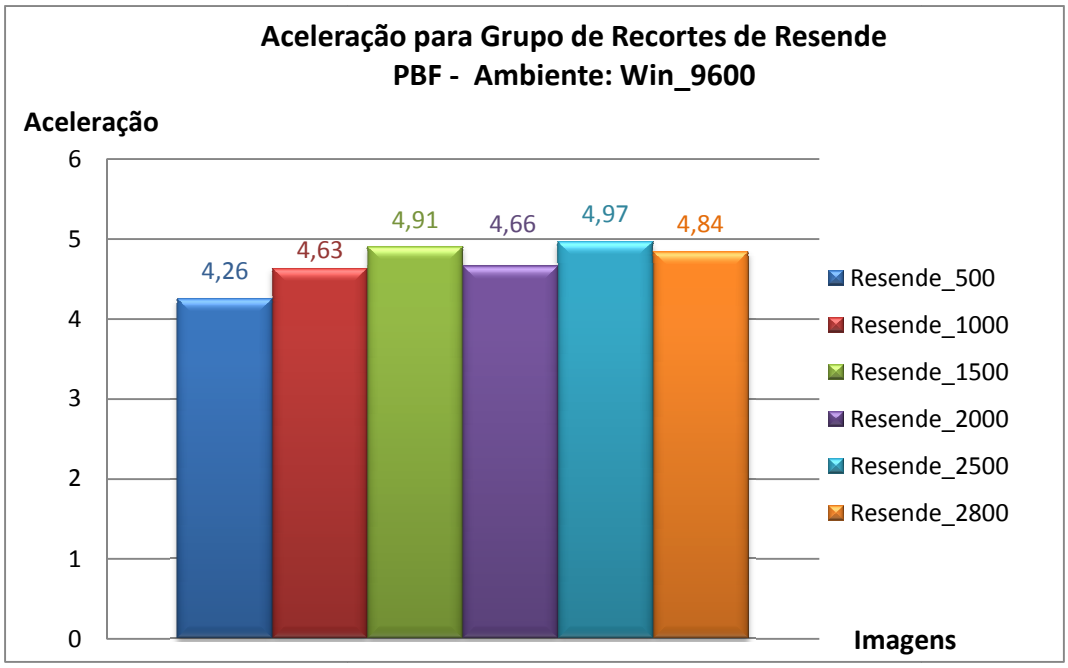


Figura 5.15. Aceleração do grupo de recortes da imagem Resende para o ambiente Win_9600 utilizando Melhor Vizinho.

Na Figura 5.15, nota-se que as acelerações obtidas pela versão paralela foram superiores a 4, chegando no melhor caso próxima a 5. Os dados sugerem

que há uma tendência de aumento da aceleração à medida que o tamanho da imagem aumenta, porém para Resende_2000 e Resende_2800 essa tendência não se confirmou.

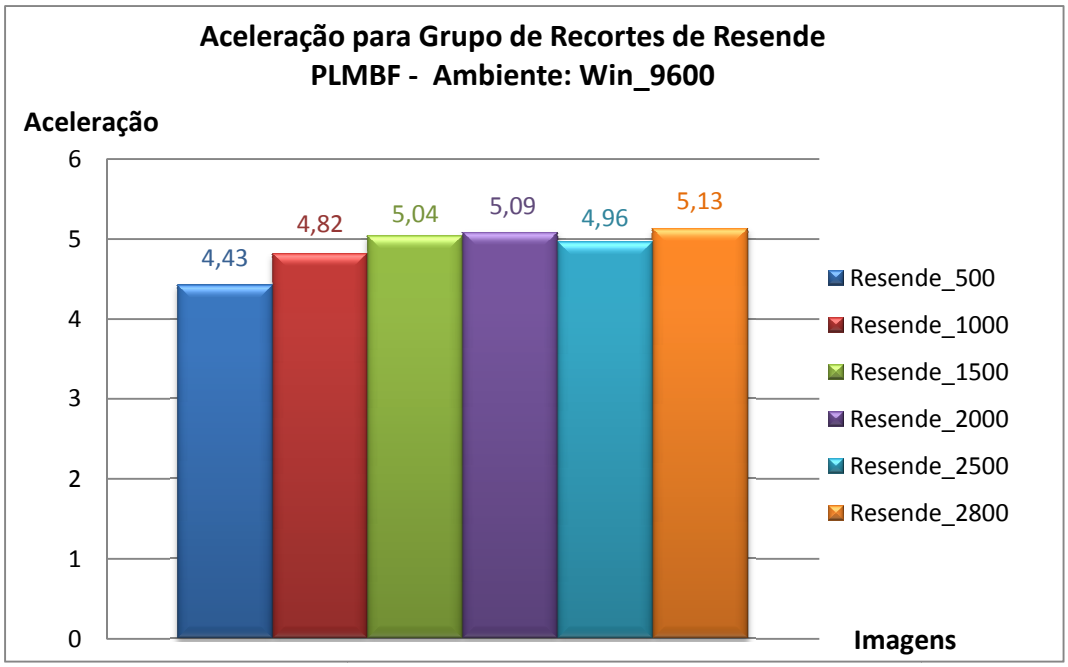


Figura 5.16. Aceleração do grupo de recortes da imagem Resende para o ambiente Win_9600 utilizando Melhor Vizinho Mútuo.

Nota-se igualmente na Figura 5.16 uma tendência no aumento da aceleração com o tamanho do recorte com exceção do resultado de Resende_2500 que apresentou uma pequena queda em relação aos resultados de Resende_1500 e Resende_2000. Foram alcançadas, neste caso, acelerações em torno de 5, atingindo o máximo de 5,13.

A tendência de um ganho mais alto para imagens de maiores tamanhos tem relação com o fato de haver um maior número de *pixels* para processar, o que causa a expectativa de haver mais processamento a ser paralelizado. Porém, cada imagem contém regiões diferentes o que faz com que a segmentação seja executada de uma maneira singular para cada uma delas e por conseqüência que seus tamanhos não sejam o único fator determinante para obtenção de um maior desempenho.

No ambiente Lnx_Tesla, a aceleração obtida para as imagens Resende_500 e Resende_1000 foram relativamente baixas, conforme se pode observar nas

Figuras 5.17 e 5.18. Este fato tem a ver com a subutilização de uma GPU mais potente para um menor processamento. Desta forma, no caso destas menores imagens, o custo relativo da transferência de dados da CPU para esta GPU e dos acessos à sua memória é comparativamente maior do que o processamento.

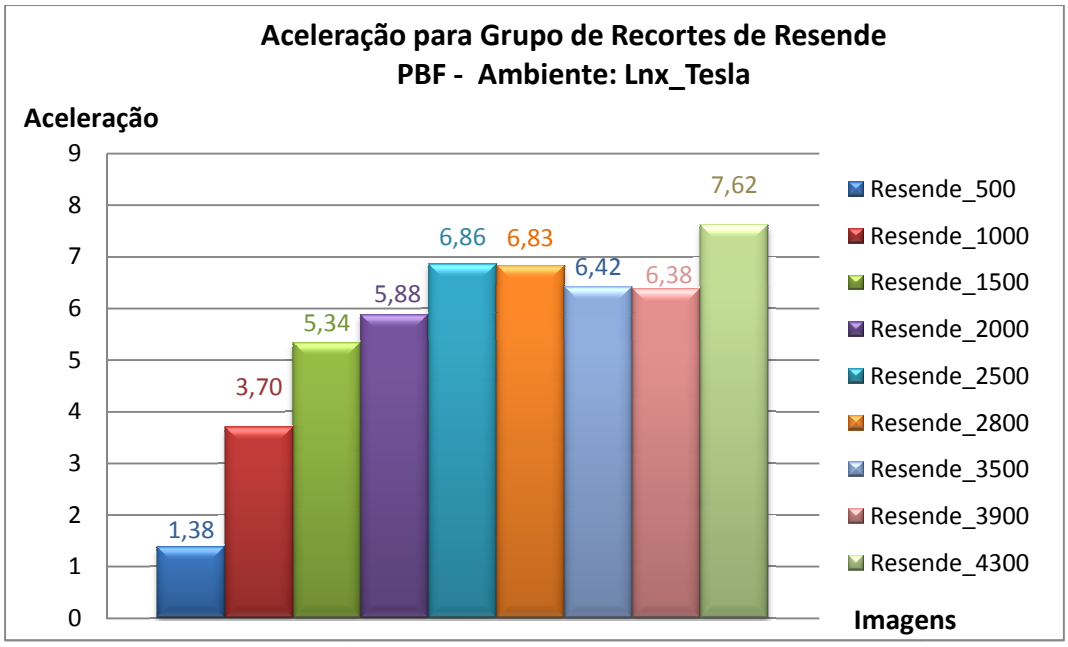


Figura 5.17. Aceleração do grupo de recortes da imagem Resende para o ambiente Lnx_Tesla utilizando Melhor Vizinho.

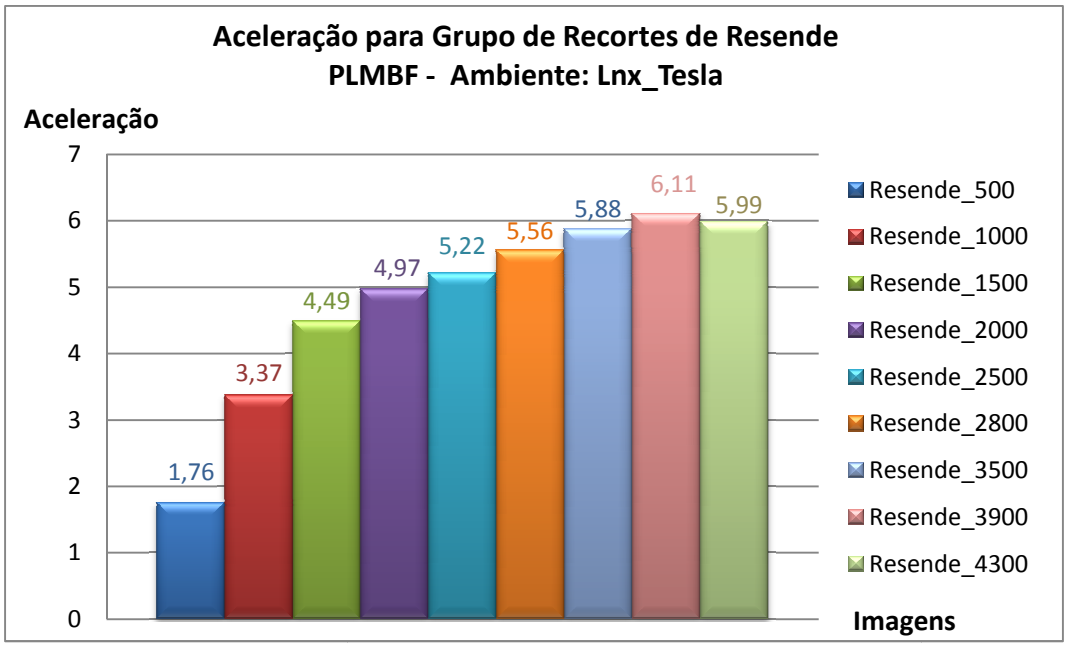


Figura 5.18. Aceleração do grupo de recortes da imagem Resende para o ambiente Lnx_Tesla utilizando Melhor Vizinho Mútuo.

Na versão PBF (Figura 5.17), a partir de Resende_1500, as acelerações são superiores a 5, superando em alguns casos 6, e atingindo o máximo de 7,62 para Resende_4300.

Na Figura 5.18, conforme o tamanho do recorte aumenta a aceleração também se torna maior. A exceção é a imagem Resende_4300 neste caso. A aceleração máxima obtida foi de 6,11 para a imagem Resende_3900.

Mais uma vez observa-se a tendência do aumento da aceleração de acordo com o crescimento do tamanho das imagens. A explicação deste fato neste ambiente é a mesma para o ambiente Win_9600 e de maneira análoga, as exceções são justificadas.

Outra bateria de experimentos foi realizada utilizando o grupo de diferentes imagens de teste. As Figuras 5.19 e 5.20 expressam, respectivamente, as acelerações obtidas utilizando as versões PBF e PLMBF no ambiente Win_9600. As acelerações atingidas em Lnx_Tesla com a heurística de Melhor Vizinho são exibidos na Figura 5.21. A Figura 5.22 expõe as acelerações alcançadas com a heurística de Melhor Vizinho Mútuo no ambiente Lnx_Tesla.

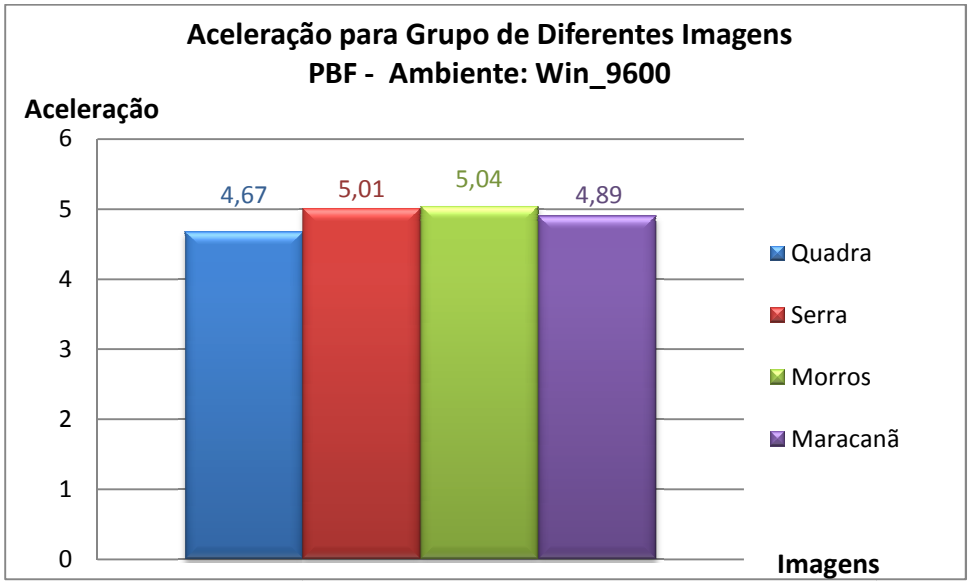


Figura 5.19. Aceleração do grupo de diferentes imagens para o ambiente Win_9600 utilizando Melhor Vizinho

Os resultados obtidos pelo Melhor Vizinho em Win_9600 (Figura 5.19) variam entre 4,67 e 5,04, o que é um resultado próximo do observado nos outros experimentos deste ambiente.

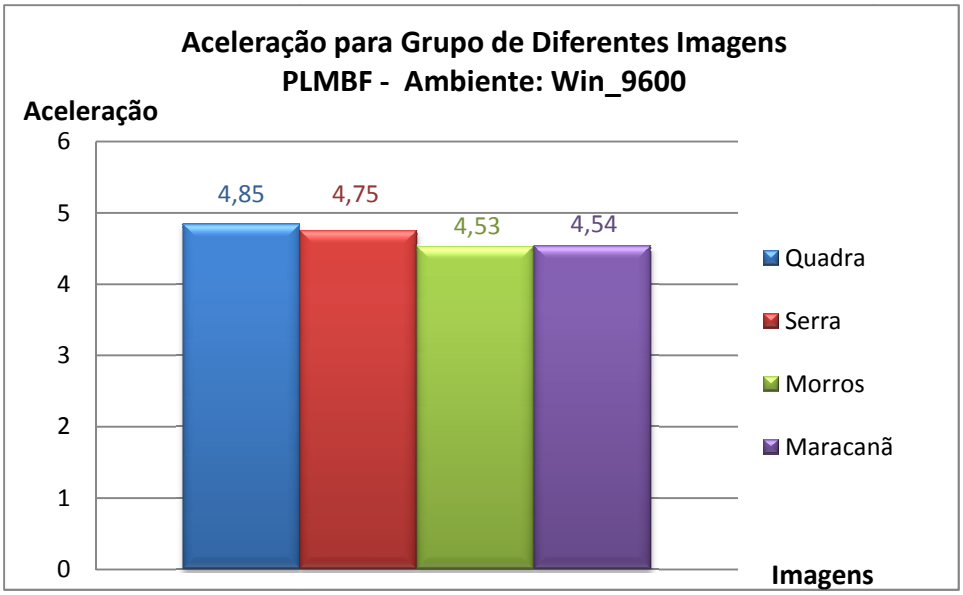


Figura 5.20. Aceleração do grupo de diferentes imagens para o ambiente Win_9600 utilizando Melhor Vizinho Mútuo

Na Figura 5.20, nota-se certo declínio da aceleração à medida que o tamanho das imagens aumenta. Este comportamento difere daquele verificado nos experimentos anteriores. De qualquer forma, este declínio é pequeno em termos relativos. Além disso, para todas as imagens houve acelerações superiores a 4,5 com o máximo em 4,85, o que é um resultado compatível com os outros obtidos nos experimentos realizados no ambiente Win_9600.

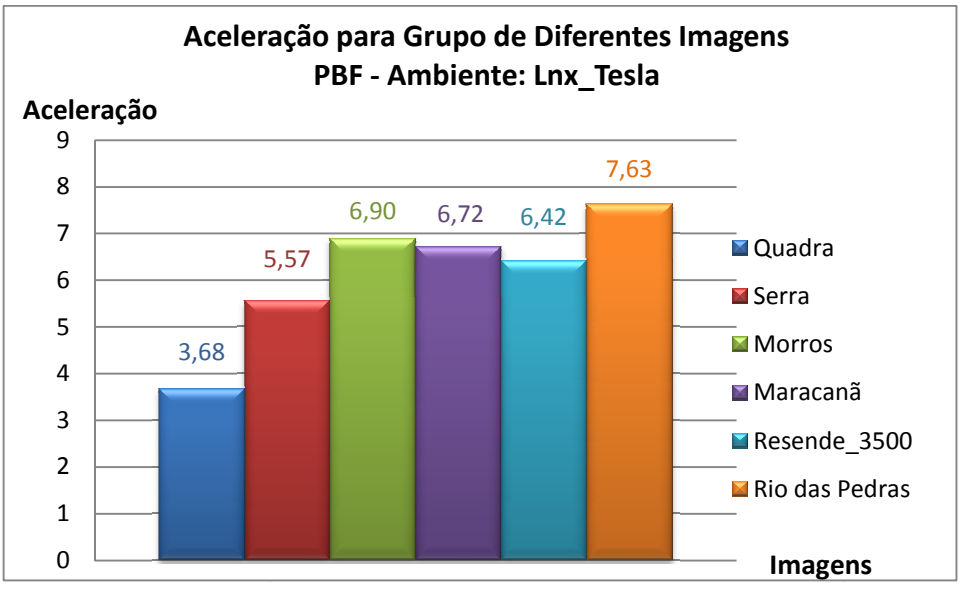


Figura 5.21. Aceleração do grupo de diferentes imagens para o ambiente Lnx_Tesla utilizando Melhor Vizinho

A Figura 5.21 apresenta uma aceleração relativamente menor para Quadra. A explicação está novamente relacionada ao processamento de uma menor quantidade de dados e a subutilização da GPU TESLA, da mesma maneira que ocorreu para o grupo de recortes de Resende.

Para as outras imagens deste grupo de teste, a Figura 5.21 apresenta acelerações consideravelmente mais altas, com valores próximos de 7 para Morros e Maracanã e atingindo 7,63 para a imagem Rio das Pedras.

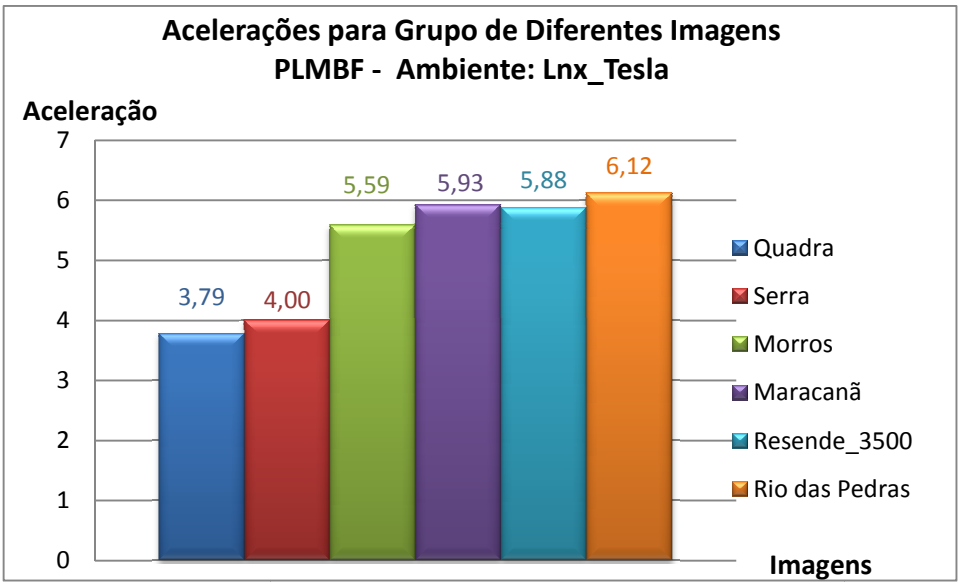


Figura 5.22. Aceleração do grupo de diferentes imagens para o ambiente Lnx_Tesla utilizando Melhor Vizinho Mútuo

A Figura 5.22 também exibe uma menor aceleração para as imagens Quadra e Serra devido aos custos de se utilizar a TESLA para um menor processamento, conforme explicado anteriormente.

Contudo, ao se utilizar imagens de teste maiores, são obtidos maiores ganhos de desempenho computacional. As acelerações foram superiores a 5,59 atingindo a máxima de 6,12 para a imagem Rio das Pedras.

Em geral, as acelerações obtidas para a versão PBF foram semelhantes às obtidas para a versão PLMBF, apesar de serem em média um pouco maiores. Isto demonstra que para ambas as heurísticas de decisão a paralelização resultou em um ganho de desempenho similar.

Cabe ainda ressaltar que apesar de alcançar acelerações de até 7,63 vezes em relação à versão seqüencial, estas ficaram bem abaixo do número de núcleos utilizados nas GPUs. Isto ocorre devido às características do algoritmo de segmentação que possui uma forte dependência entre as *threads* e necessita de acesso constante a memória global.