4
Otimizando o *NLM*

Este capítulo detalha uma proposta para otimizar o *NLM* utilizando não apenas um enfoque matemático, mas considerando várias propriedades das imagens e características do sistema humano de visão (*HVS*) que foram omitidas da proposta original do *NLM*. O capítulo também apresenta uma revisão de diferentes enfoques para otimizar o *NLM*.

4.1.
Predizendo Resultados de Semelhança

A complexidade alta, em tempo, do *NLM* origina-se da computação da distância ponderada L^2 para todos os pares de pixels, ou pelo menos entre cada pixel e todos os pixels na sua janela de busca. Se pudéssemos ter conhecimento a priori das comparações que terão similaridade tão baixa que a sua contribuição à média ponderada do pixel será negligível, poderíamos omitir a computação de todos estes pares de pixels. Portanto, necessitamos de um teste simples que possa predizer os resultados da computação da distância ponderada L^2.

Um esquema similar já foi utilizado em outros algoritmos para processamento de imagens. Este trabalho tenta utilizar o conhecimento acumulado neste sentido para acelerar o *NLM*. Codificação de imagens por *quantificação vetorial* (*vector quantization VQ*) trata do casamento de vetores de pixels similares, que podem ser interpretados como vizinhanças, e é rápido o suficiente para ser utilizada em compressão de vídeo. As seções seguintes recordam alguns métodos para acelerar *VQ* e tratam de adaptações para o ambiente do *NLM*.

4.1.1.
Quantificação Vetorial

VQ é um esquema de compressão de imagens no qual uma imagem é codificada por um número limitado de vetores. Desta forma podemos criar uma descrição compacta da imagem baseada em um dicionário de vetores de pixels e

substituindo a imagem por apontadores para o dicionário de códigos em lugar de manter a imagem na sua resolução original. Isto é possível pois uma imagem natural é composta de características contendo predominantemente gradientes suaves. Pixels adjacentes a estas características tem valores similares e a quantização destes valores em códigos produz um erro pequeno, que na maioria dos casos (dependendo do nível de quantização) é invisível ao olho humano. *VQ* é assimétrico por natureza. A compressão exige bastante computação, pois está baseada em um código e em encontrar a melhor aproximação do pixel com uma entrada no vetor. A descompressão no entanto usa uma pesquisa simples no vetor e é portanto rápida. O que se segue revisa a base matemática para *VQ*.

Podemos definir *VQ* como um mapeamento *Q* do espaço Euclidiano *k*-dimensional R^k em um conjunto finito de códigos *C* em R^k. Assim, $Q: R^k \rightarrow C$ onde $C = \{C_i : i = 1,2,...,N\}$ é um conjunto de vetores chamado de ***código*** de tamanho *N*. Cada vetor *k*-dimensional $x = (x_1, x_2, ..., x_k)$, representando uma imagem, é comparado com todos os códigos no código de acordo com a distância Euclidiana quadrática (L^2) já definida na Equação 2, só que aplicada a um vetor de pixels X e um código C_i

$$d^2(X, C_i) = \sum_{j=1}^{k} (x_i - c_{ij})^2 .$$

O mapeamento *Q* casa *X* com a palavra C_{bm} que melhor satisfaz a condição:

$$d^2(X, C_{bm}) = \min_{i=1,..,N} d^2(X, C_i).$$

Nosso interesse em *VQ* vem do segundo estágio de compressão – encontrar o melhor código para um vetor de entrada. Como pode ser visto, *VQ* requer pesquisa exaustiva para encontrar o código que melhor se casa com cada vetor, enquanto a decodificação de *VQ* pode ser implementada como uma simples pesquisa em uma tabela. Para acelerar o processo de codificação, vários algoritmos foram propostos para acelerar o processo de casamento de códigos. Todos os métodos investigados neste Trabalho implementam pesquisa exaustiva com um estágio preliminar antes de computar L^2. O estágio de avaliação usa "parâmetros de seleção" que podem predizer rapidamente um limite inferior para a computação de L^2. Ao considerar os parâmetros de seleção, podemos selecionar

códigos apropriados para serem comparados com cada vetor, e rejeitar códigos inapropriados sem computar a distância L^2.

4.1.1.1. Pesquisa do vizinho mais próximo com média igual (*ENNS*)

Este método foi proposto por Ra e Kim [8] é e comum em codificação *VQ*. usa o fato de que a distância Euclidiana quadrática entre 2 vetores é usualmente próxima da distância entre a média quadrática destes vetores. Em termos simples, se dois vetores tem uma distância pequena entre eles então as suas médias tem comportamento similar. Esta técnica usa o valor médio de um vetor como parâmetro de seleção para rejeitar códigos que não são similares ao vetor de entrada e que portanto, reduz o custo de computar a similaridade consideravelmente.

Quando k é a dimensão dos vetores, X é o vetor de pixels de entrada e C é uma coleção de códigos, podemos escrever a seguinte desigualdade [8]:

Equação 6. $$d^2(X, C_i) \geq k(\overline{X} - \overline{C_i})^2$$

A distância L^2 será maior ou igual à dimensão do vetor multiplicada pela média quadrática das diferenças. A pesquisa do código será feita computando-se a distância ao primeiro código, tomando-o como mínimo e computando a distância L^2 apenas para os códigos que tem uma distância menor de acordo com a desigualdade média.

Resultados experimentais ([11], [12]) mostram uma redução de 81-95% no número de operações como conseqüência da alta taxa de rejeição no processo de seleção. Isto leva a uma redução no tempo de compressão por um fator de 4 a 15.

4.1.1.2. Pesquisa do vizinho mais próximo com média e variância iguais (*EENNS*)

Este método foi proposto por Lee e Chen [9] e também é bastante difundido para codificação *VQ*. Dois vetores com a mesma média podem ter uma distância grande entre eles. Levando-se em consideração a variância como um parâmetro secundário de seleção, podemos rejeitar muitos códigos com média similar e

distância grande. Em conjunto com o critério *ENNS*, usamos o seguinte critério [9] para rejeitar candidatos que passaram no primeiro filtro.

Equação 7. $$d^2\left(X,\overline{C}_i\right) \geq k\left(V_x - V_{C_i}\right)^2$$

onde V_x^2 representa a variância dos vetores, definida como:

$$V_x^2 = \frac{1}{k}\sum_{j=1}^{k}\left(x_j - \overline{x}\right)^2$$

A distância L^2 será maior ou igual à dimensão do vetor multiplicada pelo quadrado do desvio padrão das diferenças. A pesquisa pelo código trabalha computando a distância ao primeiro código, tomando-o como mínimo e computando a distância L^2 apenas para os códigos que tem menor distância de acordo com as desigualdades *ENNS* e *EENNS*.

Note que enquanto todas as operações do *ENNS* e *VQ* são relativamente simples, *EENNS* requer a computação também de raízes quadradas. Mesmo assim, esta computação adicional é interessante pois ajuda a atingir taxas mais altas de rejeição e menos cálculos da distância L^2 para os códigos.

Resultados experimentais ([11], [12]) mostram uma redução de 89-98% no número de computações como conseqüência da taxa mais alta de rejeição do processo de seleção. Isto leva a uma redução do tempo de compressão por um fator de 6 a 25.

4.1.1.3.Pesquisa do vizinho mais próximo com média e variância iguais melhorada (*IEENNS*)

Além das duas desigualdades presentes no *ENNS* e *EENNS*, uma terceira for proposta por Baek, Jeon e Sung [10]:

Equação 8. $$d^2\left(X,C_i\right) \geq k\left(\overline{X} - \overline{C}_i\right)^2 + k\left(V_x - V_{C_i}\right)^2$$

A notação é semelhante ao *ENNS* e *EENNS*, usando medidas da média e do desvio padrão como parâmetros de seleção para computar rapidamente o limite inferior para a distância L^2. Esta modificação usa a mesma informação que o *EENNS* sem necessitar memória adicional e combina as duas medidas estatísticas da média e do desvio padrão apresentadas nas Equação 6 e Equação 7 em uma única desigualdade elegante. A soma dos limites da distância é naturalmente

maior do que cada distância computada pelos métodos anteriores e portanto, uma taxa maior de rejeição ocorre.

Resultados experimentais [11], [12]) mostram uma redução de 92-98% no número de computações dado a alta taxa de rejeição do processo de seleção. Isto leva a uma redução do tempo de compressão por um fator de 8 a 29.

4.1.1.4. Métodos adicionais de *VQ* e ambientes com ruído

Técnicas adicionais de aceleração para *VQ* forma consideradas na tentativa de acelerar o *NLM*, entre as quais encontramos aquelas sugeridas por Pan, Lu e Sun [11] e Wang e Tang [12]. Estes algoritmos baseiam-se na tentativa de aumentar a taxa de rejeição dividindo o vetor em sub-vetores e aplicando os métodos já explicados (*IENNS* etc.) a cada sub-vetor.

O uso destas técnicas na aceleração do *NLM* pode ser investigado em mais detalhe, mas está fora do escopo deste trabalho.

4.1.2. Uso do IEENNS como acelerador para NLM

A similaridade entre as medidas de semelhança do *VQ* e do *NLM* torna os métodos de aceleração do *VQ* candidatos óbvios para aceleração do *NLM*. É possível reduzir o número de computações da distância ponderada L^2 rejeitando vetores com pequena contribuição ponderada para um pixel específico. As maiores diferenças entre os algoritmos de VQ e de NLM são:

1. *VQ* usa a distância L^2 ao passo que o *NLM* usa a distância ***ponderada*** L^2.
2. *VQ* compara o vetor com um conjunto de códigos, enquanto que o *NLM* compara **todas** as combinações de dois vetores dentro de um conjunto.

Para utilizar os métodos de aceleração desenvolvidos para *VQ* no contexto do *NLM*, precisamos desenvolver novas desigualdades para tratar a distância ponderada L^2 e novos parâmetros de seleção para esta métrica. Em seguida descrevemos uma adaptação da métrica L^2 usada no *IEENNS* para a métrica ponderada L^2 usada no *NLM*.

4.1.2.1. Definição das Métricas

Quando $x = (x_1, x_2, ..., x_k)$ e $y = (y_1, y_2, ..., y_k)$ são vetores em R^k a distância L^2 e dada pela Equação 2. $d^2(x,y) = \sum_{j=1}^{k} (x_j - y_j)^2$.

A média e a variância são definidas pela Equação 9 e Equação 10:

Equação 9. $$\bar{x} = \frac{1}{k}\sum_{j=1}^{k} x_j$$

Equação 10. $$V_x^2 = \frac{1}{k}\sum_{j=1}^{k} (x_j - \bar{x})^2$$

e a distância ***ponderada*** L^2 já foi definida na Equação 3. $d_a^2(x,y) = \sum_{j=1}^{k} a_j (x_j - y_j)^2$ *e* $a_j \geq 0$.

Recorde que o algoritmo *NLM* restringe a soma dos coeficientes a ser 1 ($\sum_{j=1}^{k} a_j = 1$). Esta restrição não é relevante, porém, para esta métrica.

4.1.2.2. Adaptação do *IEENNS* para a distância ponderada L^2

Nesta seção, procuraremos parâmetros para o processo de seleção semelhantes aos parâmetros de média e desvio padrão usados pelo *IEENNS*. Estes parâmetros nos fornecerão uma forma rápida de avaliar as possibilidades de computação da distância L^2 através de pesos relevantes para o processo de computação da média.

Da definição da distância ponderada L^2 (Equação 3) podemos voltar facilmente à distância L^2 (Equação 2).

$$d_a^2(x,y) \equiv \sum_{j=1}^{k} a_j (x_j - y_j)^2 = \sum_{j=1}^{k} \left(\sqrt{a_j}\, x_j - \sqrt{a_j}\, y_j\right)^2 \equiv d^2(x_a, y_a)$$

definindo $x_a = (\sqrt{a_1}x_1, \sqrt{a_2}x_2, ..., \sqrt{a_k}x_k)$ e $y_a = (\sqrt{a_1}y_1, \sqrt{a_2}y_2, ..., \sqrt{a_k}y_k)$ como os "*root weighted vectors*".

Observamos que a distância ponderada L^2 entre dois vetores é igual à distância ***não ponderada*** L^2 entre os vetores ponderados pela raiz quadrada dos pesos. Podemos então usar as desigualdades derivadas das acelerações para *VQ* para a métrica ponderada L^2. Tudo o que precisamos fazer é encontrar os parâmetros de seleção correspondentes (média e desvio padrão) para os vetores ponderados pela raiz quadrada. Neste espaço o "*root weighted mean*", de acordo como a Equação 9 será:

Equação 11. $$\bar{x}_a = \frac{1}{k}\sum_{j=1}^{k} x_{aj} = \frac{1}{k}\sum_{j=1}^{k} \sqrt{a_j}\, x_j$$

e o *"root weighted variance"*, de acordo com a Equação 10, será:

Equação 12. $$V_{x_a}^2 = \frac{1}{k}\sum_{j=1}^{k} \left(x_{aj} - \bar{x}_a\right)^2 = \frac{1}{k}\sum_{j=1}^{k} \left(\sqrt{a_j}\, x_j - \bar{x}_a\right)^2$$

e a desigualdade do *IEENNS* (Equação 8) para a métrica ponderada será:

$$d_a^2(x,y) = d^2(x_a, y_a) \geq k\left(\bar{x}_a - \bar{y}_a\right)^2 + k\left(V_{x_a} - V_{y_a}\right)^2$$

logo, a distância ponderada L^2 entre 2 vetores não pode ser menor do que:

Equação 13. $$d_{a\,\min}^2(x,y) = k\left(\bar{x}_a - \bar{y}_a\right)^2 + k\left(V_{x_a} - V_{y_a}\right)^2$$

Da Equação 11, Equação 12 e Equação 13 obtemos:

Equação 14. $$d_{a\,\min}^2(x,y) = \left(x' - y'\right)^2 + \left(V_x' - V_y'\right)^2$$

definindo os parâmetros do processo de seleção como:

Equação 15. $$x' = \frac{1}{\sqrt{k}}\sum_{j=1}^{k} \sqrt{a_j}\, x_j$$

Equação 16. $$V_x' = \sqrt{\sum_{j=1}^{k} \left(\sqrt{a_j}\, x_j - \frac{1}{\sqrt{k}} x'\right)^2}$$

A distância ponderada L^2 mínima será então dada pela distância L^2 entre dois pontos correspondentes no espaço $\left(x', V_x'\right)$ de parâmetros de seleção. As equações para o mínimo da distância ponderada L^2 e para os parâmetros de seleção (Equação 14, Equação 15 e Equação 16) nos acompanharão por todo o trabalho como a base para o "filtro de distância mínima" (4.1.2.3) e para o esquema de agrupamento sugerido (4.2.1).

4.1.2.3. Filtro de distância mínima

Temos agora uma métrica que pode ser usada para definir um limite inferior para a distância ponderada L^2 entre duas vizinhanças. Isto naturalmente define um limite superior para a contribuição ponderada dos pixels na sua filtragem mútua.

Calculando e armazenando os parâmetros $(\bar{x}_a, V_{xa})$ para cada um dos pixels da imagem, podemos determinar o limite inferior da distância entre os pixels da imagem. Da Equação 6, podemos encontrar a contribuição ponderada máxima esperada:

$$w_{\max}(x,y) = e^{(-1)\frac{d^2_{a\min}(x,y)}{h^2}}$$

ou

$$w_{\max} = e^{\frac{-d^2_{a\min}}{h^2}}$$

Podemos então definir um limite para as comparações. Se w_{TH} é o menor peso aceitável e desejamos investir na computação de L^2, devemos computar L^2 apenas se:

Equação 17. $$d^2_{a\min} = (x'-y')^2 + (V'_x - V'_y)^2 < -h^2 \ln(w_{TH})$$

Lembre-se de que a distância quadrática esperada entre duas vizinhanças sem ruído e idênticas é $2\sigma_n^2$. seguindo esta lógica, devemos pesquisar um limitador proporcional a σ_n^2. devemos lembrar que a distância ponderada esperada L^2 pode ser muito maior do que o limite inferior encontrado com a Equação 17, de tal forma que definindo o limite baixo o suficiente podemos eliminar a maioria das computações que não são produtivas, sem bloquear computações que terão alguma contribuição mesmo que sejam maiores do que o limitar imposto.

4.1.3. Filtro de Gradiente

Procuremos entender primeiro a função gradiente no contexto de imagens. Se $I(x,y)$ é uma função do nível de intensidade de uma imagem, então o gradiente de $I(x,y)$, denotado por $\nabla I(x,y)$, fornece-nos informação sobre a

declividade da função $I(x,y)$ em um ponto (x,y). A direção do declive mais acentuado de cada ponto $I(x,y)$ é dada pelo ângulo do gradiente, enquanto a magnitude do gradiente indica o quanto inclinado o declive é. O gradiente é calculado usando as máscaras dos filtros passa-alto horizontal e vertical. Por exemplo, os kerneis horizontal e vertical para a computação de um gradiente 5x5 são:

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix}, \quad V = \begin{bmatrix} 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \end{bmatrix}.$$

A magnitude do gradiente é computada nas direções horizontal e vertical usando a operação de convolução com os kerneis acima. Se os resultados das convoluções forem $V(x,y) = \frac{\partial I(x,y)}{\partial x}$ e $H(x,y) = \frac{\partial I(x,y)}{\partial y}$ então a magnitude do gradiente é

$$\|\nabla I(x,y)\| = \sqrt{H(x,y)^2 + V(x,y)^2}$$

e a orientação é:

$$\angle \nabla I = \arctan \frac{H(x,y)}{V(x,y)}$$

A Figura 11 mostra um exemplo de um gradiente de uma imagem. À direita está a imagem, ao centro uma representação da magnitude do gradiente da imagem e à direita a orientação do gradiente da imagem. Gradientes baixos são mais escuros que os declives acentuados e o ângulo é quantizado em 4 níveis de cinza mostrando gradientes apontando pata a esquerda, direita, para cima e para baixo. Note que as áreas planas possuem magnitudes do gradiente baixas (escuras) e orientação instável, enquanto que gradientes fortes são definidos por linhas brancas na imagem representando a magnitude e possuem orientação bem definida.

Figura 11 Exemplo do gradiente de uma imagem

Explicado o gradiente, passamos a discutir como podemos usar os gradientes das vizinhanças de um pixel para rejeitar computações desnecessárias da distância ponderada L^2 . Embora uma equação exata para o limite inferior pode ser apresentada neste caso (como no caso do filtro de distância mínima), é intuitivo que duas vizinhanças com magnitudes significativas dos gradientes não devem produzir pesos significativos se o ângulo entre as orientações dos seus gradientes for muito. Um filtro baseado nesta observação foi usado por Mahmoudi e Sapiro [5], que calcula a magnitude média e a orientação média do gradiente para a vizinhança de cada pixel. Quando σ_∇ é o limite das magnitudes dos gradientes e σ_θ é o limite do ângulo entre as orientações dos gradientes, o filtro que utilizaram para rejeitar comparações dos pixels *i* e *j* é dado por:

Se $\|\nabla v(i)\| > \sigma_\nabla$ *and* $\|\nabla v(j)\| > \sigma_\nabla$ *and* $\theta(i,j) > \sigma_\theta$ então rejeite as comparações entre as vizinhanças de *i* e *j*.

O filtro de gradiente usado neste Trabalho utiliza valores do gradiente local em lugar de valores médios, utilizados em [5]. Observamos que o gradiente local é mais relevante para o resultado da distância ponderada L^2 do que o gradiente médio. Este é um filtro complementar ao filtro de distância média que permite eliminar computações de L^2 entre vizinhanças com média e variância ponderadas quadráticas (*root weighted mean and variance*) semelhantes, mas estão rotacionados por um certo ângulo e não devem produzir um peso significativo

4.2. Técnicas de agrupamento

Outra forma de eliminar computações redundantes consiste em agrupar a vizinhança de um pixel em grupos de pixels já detectados como semelhantes. Sugere-se que o esquema de agrupamento seja executado como um passo preliminar à computação da semelhança. Semelhança será computada apenas dentro de um agrupamento, economizando computações irrelevantes. Isto é similar ao esquema sugerido por Mahmoudi e Sapiro em [5]. Eles sugerem agrupar pela média e pelo gradiente das vizinhanças e testar semelhança dentro e nos blocos adjacentes. O método proposto aqui será utilizado para agrupar em blocos, mas fornecerá também limites exatos para a contribuição esperada de um pixel dentro de um bloco.

Ao usar agrupamento precisamos considerar os seguintes fatores:

1. O processo de agrupamento deve ter baixa complexidade de tempo de tal forma a não penalizar o algoritmo final.
2. O processo de agrupamento deve criar uma vizinhança para cada pixel na qual o algoritmo *NLM* original irá operar.
3. A vizinhança gerada pelo agrupamento deve ter muitos pixels semelhantes.

4.2.1. Esquema de agrupamento

O esquema de agrupamento classifica a vizinhança de cada pixel de acordo com parâmetros que podem ser usados para predizer o resultado da computação da distância ponderada L^2. Os dois parâmetros escolhidos foram aqueles definidos na seção “4.1.2.2”.

A Figura 12 mostra uma classificação das vizinhanças da imagem com ruído da “*lena*” de acordo com os parâmetros x' e V_x'. O eixo horizontal representa a média ponderada quadrática (x') enquanto que o eixo vertical representa o desvio padrão ponderado quadrático (V_x'). Cada ponto branco representa a localização de uma ou mais vizinhanças da imagem. Cada retângulo representa uma coleção de vetores de vizinhança. Uma linha vertical indica a localização de ”zonas planas”,

áreas em que a variância do nível de intensidade local (não a variância ponderada) dentro da vizinhança é baixa. Como a "lena" é uma imagem natural com muitas zonas planas, podemos ver que a maioria das vizinhanças da imagem está abaixo desta linha e a densidade das coleções de vetores cai à medida que cresce a distância para esta linha.

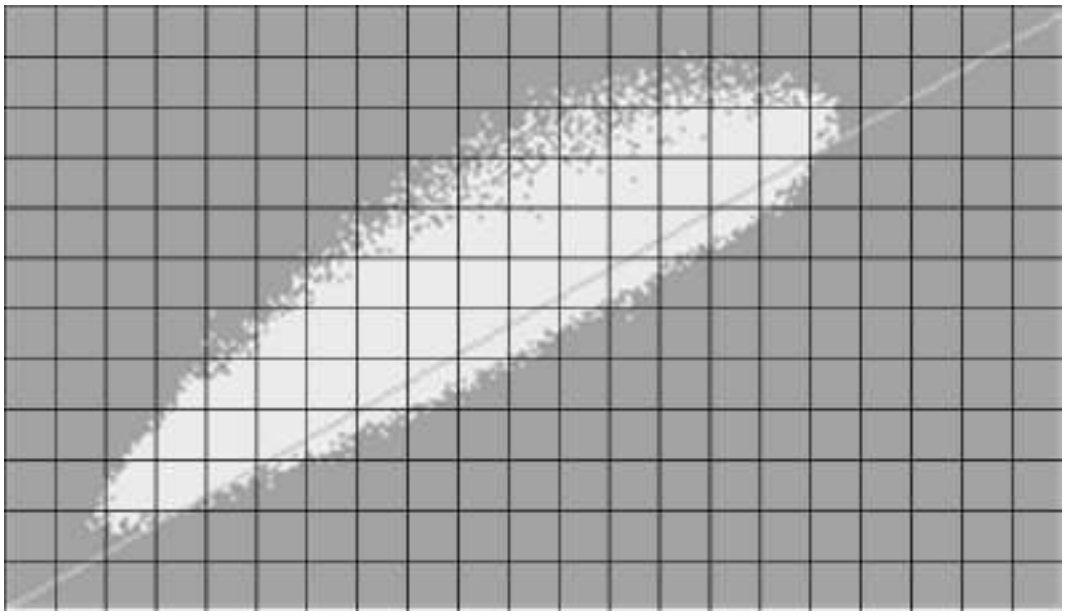

Figura 12 Classificação por média quadrática ponderada e desvio padrão ponderado

Podemos agora computar um limite inferior para a distância ponderada L^2 entre quaisquer duas vizinhanças de pixels na imagem da "lena" computando a distância L^2 entre os pontos correspondentes na Figura 12. Para tal, devemos ordenar os vetores da imagem em coleções. Cada coleção contendo apenas pixels com média quadrática ponderada (x') e desvio padrão quadrático ponderado (V_x'). A organização proposta é sob forma de uma tabela bidimensional apontando para listas encadeadas de pixels semelhantes. As entradas das tabelas são os valores quantizados x' e V_x'. Dada uma lista de pixels com o mesmo valor de (x', V_x'), o acesso a listas com pixels com distância dentro de um dado limite pode ser feito em tempo $O(1)$. Popular a tabela pode ser feito em um estágio preliminar cuja complexidade de tempo é linear com respeito ao número de pixels da imagem.

Uma pesquisa por vizinhanças semelhantes pode ser feita na tabela em lugar de na imagem original. Isto reduz a complexidade original de quadrática no número de pixels na imagem para quadrática no número de pixels nas listas

relacionadas. Isto é claro pode ter complexidade igual à original (no caso de uma imagem totalmente plana por exemplo), mas tipicamente no caso de imagens naturais será provavelmente bem menor. Uma complexidade linear pode ser forçada fixando-se o tamanho máximo da janela de busca no domínio da tabela de agrupamento, da mesma forma que foi feito para o enfoque de janela de busca para o *NLM*.

Note que uma pesquisa no domínio da tabela de agrupamento nos trás para muito perto da idéia original de pesquisar características semelhantes na imagem completa. Este princípio foi sacrificado na versão usando janela de busca do *NLM* original para assegurar complexidade adequada. Agrupamento pode combinar muitas áreas detalhadas da imagem na mesma coleção ou em coleções vizinhas, mesmo que a distância entre as áreas seja maior do que o tamanho da janela de busca original.

4.3. Propriedades de Imagens Naturais

Imagens naturais tendem a ter grandes áreas uniformes. Estas são áreas com gradiente pequeno e variância entre as vizinhanças dos pixels também pequena. No esquema de classificação proposto, os pixels destas áreas produzirão valores similares para x' e V_x'. Como estas áreas são tipicamente grandes, espera-se que a distribuição da tabela de classificação tenha densidade muito maior para áreas uniformes do que para áreas com grande contraste. Isto tanto pode ser um problema quanto uma oportunidade. Por um lado, muitos pixels participam na média de uma área com ruído, por outro lado seria preferível que as coleções fossem pequenas.

No que tange a áreas uniformes, uma pesquisa por vizinhanças semelhantes a uma distância pequena pode na maioria dos casos ter melhor resultado do que uma pesquisa na tabela pré-classificada. Neste caso, o enfoque que usa uma janela de busca no *NLM* original pode ser melhor. Nas áreas de alto contraste, o esquema de agrupamento pode ser mais vantajoso. Esta conclusão pode ser alcançada também se analisando os resultados em [5], que reportam melhor desempenho em áreas uniforme, quando comparado com o algoritmo *NLM* original, e melhor desempenho que o agrupamento por gradiente/média em áreas de alto contraste.

Por esta razão, propomos manter pesquisa geométrica para áreas uniformes e usar pesquisa baseada em agrupamento para áreas de alto contraste.

Há outras formas de usar as propriedades enumeradas acima, desde que seja considerado o sistema humano de visão. A combinação destes enfoques com os parâmetros do *NLM* possui um potencial para melhorar tanto o desempenho quanto a qualidade percebida do resultado do algoritmo *NLM*.

4.4. Considerações sobre o sistema humano de visão

Dois fatores dominam a qualidade da percepção de uma imagem natural pelo olho humano: embaçamento (falta de nitidez) e ruído. Os dois devem ser levados em consideração em qualquer tentativa para melhorar a qualidade da imagem. Em muitos casos, os dois fatores estão inter-relacionados e melhorando um, prejudica-se o outro. Por exemplo, em muitas técnicas para redução de ruído, substituir um pixel pela média da vizinhança reduz ruído, mas tem uma forte tendência a embaçar a imagem, o que degrada a qualidade da imagem. por outro lado, um filtro de convolução passa-alto que reduz o embaçamento aumenta o ruído.

No que se segue, recordaremos alguns pontos relativos à sensibilidade do sistema humano de visão (*Human Vision System – HVS*) a ruído e embaçamento que deve ser considerado e oferece oportunidades para acelerar o *NLM*.

4.4.1. Sensibilidade do HVS a ruído

A sensibilidade do sistema humano de visão (*Human Vision System - HVS*) a ruído é altamente dependente da riqueza da imagem. Uma imagem suave é muito mais sensível a ruído do que uma imagem com muitos detalhes. Winkler e Süsstrunk [13] demonstraram um relacionamento linear entre o desvio padrão do ruído e o desvio padrão mínimo da imagem necessário para o observador padrão perceber o ruído. Em termos simples, o mesmo nível de ruído perceptível em uma imagem uniforme pode ser mascarado por detalhes em imagem de alto contraste.

Figura 13 Ruído em áreas uniformes e de alto contraste. Esquerda – imagem original; Direita – imagem com ruído adicionado com $\sigma_n = 20$

No exemplo da Figura 13, a atenção do observador é dirigida para a face com ruído, embora o mesmo nível de ruído esteja presente no tecido que cobre a cabeça, que requer mais atenção do observador para ser percebido.

O conceito de "ruído perceptível" é usado na adição de marcas d'água digitais, uma tecnologia que adiciona dados visíveis ou não a imagens. Um exemplo é dado na Figura 14.

Figura 14 Marca d'água digital

A Figura 14 mostra um exemplo de marca d'água visível. Neste caso, é essencial assegurar que a marca seja visível sobre a área escolhida. No caso de marca d'água invisível, é importante manter a marca mascarada pela imagem.

Na criação de marcas d'água é comum o uso de um parâmetro chamado de *Noise Visibility Function* (*NVF*) que usa um modelo Gaussiano para estimar quanta textura existe em qualquer área da imagem. Esta função, sugerida por Voloshynovskiy et al. em [14] é definida por

$$NVF(i) = \frac{1}{1+\theta\sigma_x^2(i)}$$

onde $\sigma_x^2(i)$ é a variância local da imagem em torno do pixel *i*, e θ é um parâmetro de ajuste definido como $\theta = \frac{D}{\sigma_{x\max}^2}$ onde $\sigma_{x\max}^2$ é a variância máxima da imagem $\sigma_{x\max}^2 = \max_{i\in I}\sigma_x^2(i)$ e *D* é um parâmetro experimental entre 50 e 1000. Para áreas da imagem de alto contraste, caracterizadas por uma variância alta, o *NVF* é perto de 0, enquanto que para áreas uniformes, o *NVF* é perto de 1.

4.4.2. Sensibilidade do HVS ao embaçamento

A sensibilidade do *HVS* ao contraste depende da freqüência especial. A Figura 15, por Izumi Ohzawa [15], mostra uma imagem de linhas se alternando em freqüências espaciais que aumentam da esquerda para a direita, e em contraste do topo para a parte inferior.

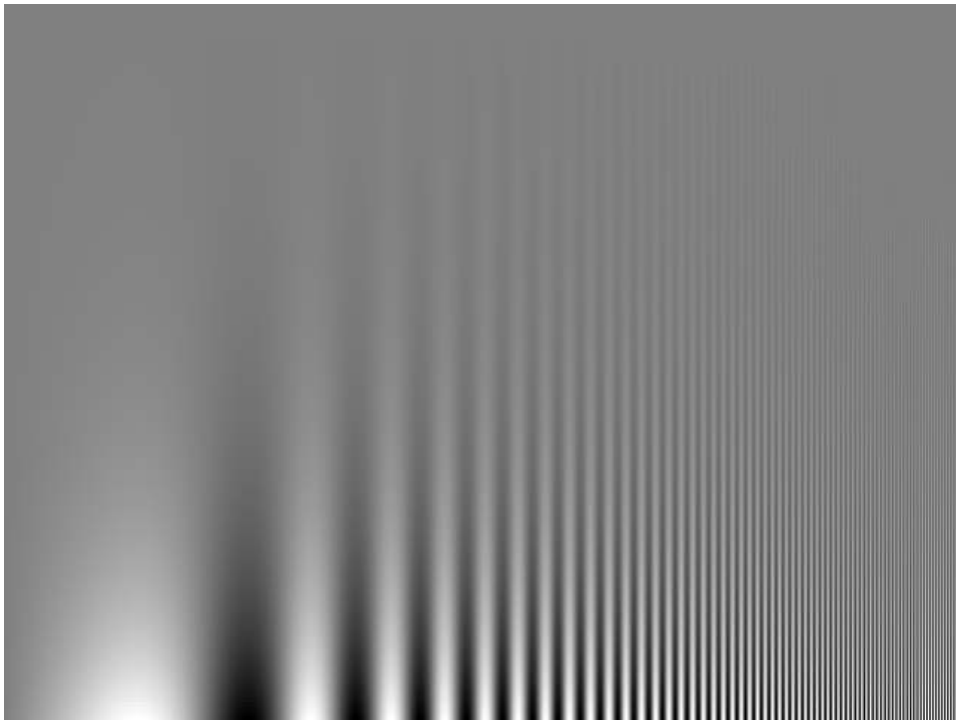

Figura 15 Sensibilidade do *HVS* ao contraste

Podemos ver que as linhas são mais fáceis de identificar ao longo da seção do meio ao alto. Este fato levou Campbell e Robson [16] a definir uma função de sensibilidade ao contraste que pode variar de observador para observador mas que tem a forma geral mostrada na Figura 16.

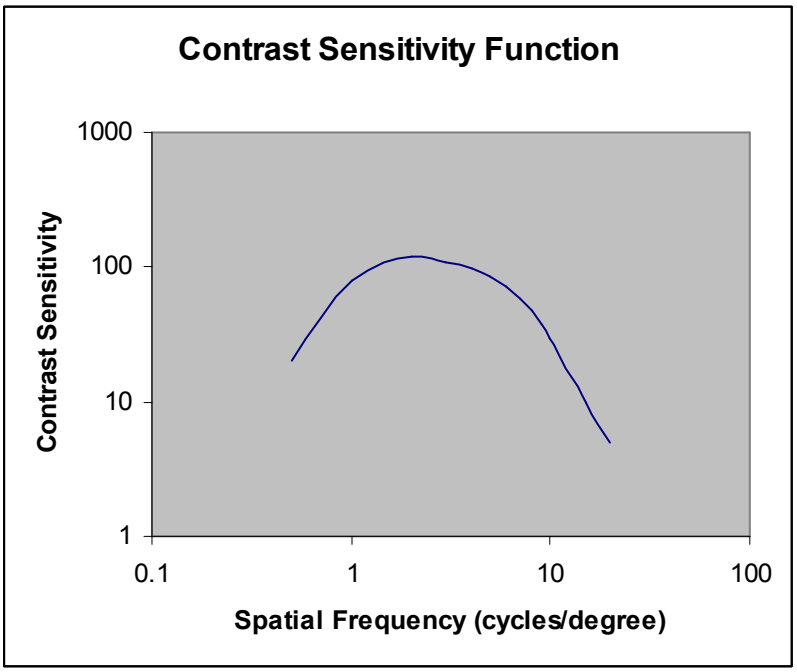

Figura 16 Função de sensibilidade ao contraste

Um efeito colateral de tomar a média dos pixels para reduzir ruído é reduzir o contraste. Como pode ser visto da função de sensibilidade ao contraste, em certas freqüências espaciais, isto pode causar perda de detalhe.

Johnson e Fairchild [17] realizaram um grande número de experimentos psico-físicos para estudar a percepção de nitidez (*sharpness*) do *HVS*. A nitidez é um critério subjetivo de qualidade de imagens. Os autores descobriram que a resolução espacial da imagem é o principal fator de nitidez de uma imagem. Uma descoberta interessante é que a adição de baixos níveis de ruído aumenta a percepção de nitidez da imagem.

É importante que durante o processo de filtragem, a percepção de nitidez da imagem não seja muito danificada. Também é importante que se mantenha os detalhes de pequeno tamanho intactos. Isto é na verdade uma das principais razões para utilizar o *NLM*, já que a forma de filtrar pixels similares mantém os valores próximos dos valores originais, embora a adoção de um fator de filtragem alto em áreas de alta resolução possa degradar a percepção da nitidez.

4.4.3. Alguns comentários sobre cor

Ao lidar com cor, toda a discussão anterior sobre ruído e nitidez continua valendo, mas com níveis distintos para os distintos canais de cor da imagem, dependendo do esquema de cor utilizado. O canal de luminância do espaço *CIELab* é muito mais sensível a ruído e embaçamento do que os canais cromáticos. O canal verde do *RGB* é muito mais sensível do que os canais do vermelho e do azul. O canal amarelo do *CMYK* é muito menos sensível do que os canais do preto, magenta e cyan.

Com esta informação, fica claro que para operar o *NLM* em uma única dimensão de cor devemos levar em consideração a sensibilidade específica de cada canal de cor. Possíveis otimizações na filtragem de imagens coloridas podem ser alcançadas adotando-se técnicas de outros algoritmos para processamento de cores, como o JPEG. Analisar semelhança no canal de luminescência, por exemplo, e aplicar os resultados aos outros canais pode ser uma estratégia adequada.

Este trabalho não procurará trazer avanços na área de cor, baseando-se na suposição de que o que puder ser alcançado para imagens em tons de cinza pode ser aproveitado para imagens coloridas.

4.5. Ajustando os parâmetros do *NLM*

Algumas sugestões para melhorar o *NLM* estão relacionadas ao controle dos parâmetros do *NLM*. A primeira refere-se ao fator de decaimento já definido para o *NLM*. Os outros parâmetros serão introduzidos nas próximas sub-seções.

4.5.1. Fator de decaimento

O importante papel do *fator de decaimento* (h) pode ser melhor apreciado depois de rever a sensibilidade do *HVS* a ruído e nitidez. Conforme visto na seção 3.1, os pesos dados pela Equação 4 podem ser escritos como:

$$w(i,j) = \frac{1}{Z(i)} e^{(-1)\frac{d_a^2(N_i,N_j)}{h^2}}$$

Para uma distância ponderada fixa L^2 entre duas vizinhanças, h^2 determinará o peso do pixel j na média do pixel i. Um valor alto para h^2 causará um nível alto de filtragem ao forçar valores altos no cálculo da média, e um valor baixo de h^2 melhor preservará detalhes ao forçar valores baixos no cálculo das médias. Um bom compromisso deve ser almejado para se ter o nível certo de filtragem. Um nível baixo não removerá ruído suficientemente e um nível alto causará perda de nitidez dos detalhes. Os autores de [1] recomendam o uso de um fator de filtragem da ordem do desvio padrão do ruído. No exemplo utilizado, um valor de $10\sigma_n$ foi utilizado.O relacionamento entre o nível de ruído e o nível de filtragem é óbvio, mas não há compensação pela perda de detalhe com um nível alto de filtragem. O efeito do *fator de decaimento* está resumido na Tabela 1.

h	Noise reduction	Image blur
Low	bad	good
High	good	bad

Tabela 1 Influência do fator de decaimento

A Tabela 2 resume a sensibilidade do *HVS* ao ruído e à perda de nitidez em áreas uniformes e de alto contraste.

HVS sensitivity	Flat area	High contrast area
Noise	high	low
Blur	low	high

Tabela 2 Sensibilidade do *HVS* ao ruído e à falta de nitidez

A conclusão óbvia é que ao adaptar o fator de decaimento ao nível de contraste de cada pixel, podemos otimizar a qualidade do *NLM*. A primeira proposta deste trabalho relativa aos parâmetros do *NLM* consiste em usar fator de filtragem variável em lugar de um fixo. O fator de filtragem variável é linear com relação ao desvio padrão do ruído conforme sugerido em [1], mas deve também ser uma função do contraste da vizinhança do pixel. Em geral: $h_i^2 = \sigma_n^2 f\left(\sigma_{Ni}^2\right)$ quando σ_n^2 é a variância do ruído e σ_{Ni}^2 é a variância da vizinhança do pixel filtrado.

Neste ponto, é essencial entender que a variância da imagem com ruído é a soma da variância da imagem original com a variância do ruído:

$$\sigma_i^2(v) = \sigma_i^2(u) + \sigma_n^2$$

Isto deve ser levado em consideração e ajustado antes de usar σ_{Ni}^2 em qualquer cálculo.

4.5.2. Objetivo ponderado

Conforme explicado, quando agrupamos pixels em blocos, esperamos que os blocos que contém vizinhanças uniformes estejam bastante populados e que blocos que contêm vizinhanças com alto contraste estejam poucos populados.

Por outro lado, isto é exatamente o que precisamos em termos de redução de ruído, um grande número de pixels por área uniforme. Mas, como a complexidade do algoritmo é linear em relação ao número de pixels em um bloco, a execução de cada um dos pixel sem um bloco pode levar mais tempo e deve ser limitada de alguma forma.

O mecanismo proposto para parada do algoritmo de filtragem é filtrar cada pixel até o ponto em que um certo peso for atingido. Retornando à sensibilidade ao ruído do *HVS*, este nível deve ser também uma função do contraste da vizinhança filtrada do pixel $w_{THi} = f\left(\sigma_{Ni}^2\right)$.

Áreas de baixa freqüência serão filtradas com um peso alto e um fator de decaimento baixo permitindo níveis de filtragem mais altos em áreas sensíveis a ruído e menos sensíveis a perda de detalhe. Áreas de alta freqüência devem ser filtradas com um peso baixo e fator de decaimento alto.

4.5.2.1. Uso de objetivo para pesos no domínio de imagens

O algoritmo *NLM* original limita a pesquisa em vizinhanças de pixels semelhantes através de uma janela de busca fixa. A ordem de busca não importa já que todos os pixels dentro da janela de busca contribuirão com seus pesos para o cálculo da média. Se utilizarmos um objetivo para o peso como limite além da janela de busca então deveremos cobrir todos os pixels na janela de busca e a busca deve começar pelos pixels com mais probabilidade de serem similares ao pixels em questão. Sugere-se conduzir a busca dentro de círculos com raio crescente centrados no pixel em questão. Assim, a busca tocará primeiro os pixels com maior probabilidade de serem úteis (à filtragem).

				10				
	9	8	7	6	7	8	9	
	8	5	4	3	4	5	8	
	7	4	2	1	2	4	7	
	6	3	1	A	1	3	6	B
	7	4	2	1	2	4	7	
	8	5	4	3	4	5	8	
	9	8	7	6	7	8	9	

Figura 17 Busca com raio crescente

A busca começa no pixel central (ponto A) e continua através dos pixels em distância crescente do pixel central, espiralando em torno dele até que os pesos

acumulados atinjam o objetivo do peso (ponto B). A janela de busca original pode ser utilizada como limite, no pior caso, caso o objetivo de peso não seja alcançado.

4.5.3. Contribuição mínima para o peso

Ao pesquisar pixels semelhantes em um domínio de agrupamento, um terceiro parâmetro deve ser usado para computar os blocos válidos que participam na filtragem de um dado pixel. Para estes pixels, já sabemos o fator de decaimento de tal forma que para decidir a distância dos blocos considerados na filtragem devemos estimar a contribuição máxima para o peso dos pixels nestes grupos. Se os blocos forem divididos usando a distância quadrática ponderada quantizada d_m, e o desvio padrão quadrático quantizado de d_v então, pelo *IEENNS*, n_m, a distância quadrática ponderada mínima do bloco, e n_v, o desvio padrão quadrático ponderado mínimo são dados por:

$$d^2_{\max} = k\left(\left((n_m - 1)d_m\right)^2 + \left((n_v - 1)d_v\right)^2\right)$$

e o peso máximo que poderá construir será

$$w(i,j) = \frac{1}{Z(i)} e^{(-1)\frac{d^2_{\max}}{h_i^2}}$$

Como $Z(i)$ é apenas uma constante de normalização, um pequeno valor de $e^{-\frac{d^2_{\max}}{h_i^2}}$ é considerado insignificante para o processo de filtragem. A contribuição mínima ao $w_{\min}$ permitirá a computação em um raio de pesquisa dado por

$$r^2 = d^2_{\max} = -h_i^2 \ln(w_{\min})$$

4.5.4. Distância ponderada L^2 em áreas uniformes

É possível que o limite inferior dado por *IEENNS* pode ser substituído pela computação da distância ponderada L^2 como uma "medida de similaridade" em áreas bastante uniformes sem nenhum efeito visível. Se este for o caso, aceleração significativa poderá ser atingida.

4.6. Simetria

Ao computar a distância ponderada L^2 entre duas vizinhanças de pixels, podemos rotacionar a vizinhança em torno do pixel central por um ângulo que minimize a distância ponderada L^2 entre duas vizinhanças. Isto não é feito no algoritmo *NLM* original ou em [5]. Assim uma taxa de acerto maior poderá ser alcançada nas comparações e o tamanho da janela de busca poderá ser reduzido.

Encontrar o ângulo certo pode ser feito computando-se o gradiente local da vizinhança do pixel central $\nabla v(i)$. e computando-se a sua orientação. Isto deve ser feito apenas para vizinhanças com uma magnitude alta do gradiente, pois de outra forma o gradiente pode ser resultado apenas de ruído e portanto não ser confiável.

4.7. Métodos adicionais

Uma descrição do algoritmo *NLM* em pseudo-código pelos autores originais, divulgada por Buades [18], revela detalhes que escapam das publicações originais do *NLM*. De acordo como algoritmo original, cada pixel contribui com o seu próprio peso para a sua filtragem. O peso próprio produz $w(x,x)=1$, e pode ser muito maior do que os pesos fornecidos por outros pixels, reduzindo assim o nível de filtragem do pixel e deixando um alto nível de ruído. O pseudo-código evita este problema reduzindo o peso próprio de cada pixel ao nível dos outros pixels. Mais precisamente, em lugar de tomar $w(x,x)=1$, o peso de cada pixel na sua própria filtragem será igual ao peso máximo dos pixels que participam na sua filtragem:

$$w(x,x) = \max w(x,y) \mid y \neq x, y \in search\_window(x) .$$

Isto deve ser útil na suavização de pixels raros que não possuem pixels similares a serem utilizados na sua filtragem. Embora não seja parte da nossa estratégia, acreditamos que haja necessidade de testar a sua influência nos resultados já que esta questão não é discutida nas publicações originais do *NLM*.

4.8. Resumo dos métodos propostos

Até o momento, este trabalho enumerou as seguintes sugestões para acelerar a execução do *NLM:*

1. *Pré-classificação das vizinhanças* dos pixels de acordo com a média e o desvio padrão quadrático ponderado para rejeitar pixels com alta distância ponderada L^2 ao pixel a ser filtrado.
2. *Fator de decaimento adaptativo* como função da variância local da imagem.
3. Uso de um *objetivo do peso* além da *janela de busca* como condição de parada para a filtragem de um pixel.
4. Rotação das vizinhanças a serem comparadas até os seus gradientes ficarem alinhados na mesma direção para aumentar a taxa de acerto de pixels similares.
5. Agrupar imagens de acordo com parâmetros de classificação e realizar a busca por semelhança na dimensão do agrupamento.
6. Para pixels em áreas uniformes da imagem, realizar buscas por pixels semelhantes na própria imagem; e, para pixels em áreas de alto contraste, realizar buscas por pixels semelhantes na tabela de agrupamentos.

Note que, exceto pela proposição do agrupamento (5,6), todas as outras proposições podem ser usadas com o *NLM* original e podem trazer benefícios tanto na qualidade da imagem quando na aceleração como será mostrado no capítulo de resultados experimentais. Em alguns casos, o método não pode ser testado plenamente para o *NLM* original. Por exemplo, o uso de um fator de decaimento muito baixo para vizinhanças com alta variância deverá falhar com o *NLM* original pois a probabilidade de encontrar vizinhanças semelhantes na janela de pesquisa é pequena. O desacoplamento dos métodos propostos do método de agrupamento permite testar os vários métodos independentemente como opções para modificar o algoritmo *NLM* original.

4.9. Algoritmo proposto

Inicialmente, a imagem é percorrida e a vizinhança de cada pixel é pré-processada para extrair os seguintes parâmetros:

1. Variância local
2. Média ponderada quadrática local
3. Desvio padrão ponderado quadrático local
4. Gradiente local

Este é o processo de classificação de pixels que já foi demonstrado visualmente na seção (4.10). A imagem é então segmentada em blocos definidos pela média ponderada quadrática local e pelo desvio padrão ponderado quadrático local. Cada pixel é adicionado a uma lista encadeada armazenando pixels com a mesma média e desvio padrão quantizados. A tabela é então processada lista a lista da seguinte forma:

1. O pixel a ser filtrado *i* é rotacionado de tal forma que o seu gradiente aponte para o norte.
2. A variância σ_i^2 de *i* é computada extraindo-se:
   a. O fator de decaimento $h_i^2 = f(\sigma_i^2)$
   b. O objetivo do peso $w_{TH}(i) = f(\sigma_i^2)$
   c. A distância máxima $d_{\max}^2 = f(h_i^2)$
3. De acordo com a variância local da imagem, decide-se se a busca por pixels semelhantes será feita na imagem ou na tabela.
4. Enquanto nenhuma das condições de parado for alcançada, continua-se a busca por pixels a serem utilizados na filtragem nas vizinhanças da média e do desvio padrão ponderados quadráticos. Se a condição *IEENNS* $d_{\max}^2 \geq (i'-j')^2 + (V_i'-V_j')^2$ for atingida para qualquer pixel *j* então a distância real $d_a^2(i,j)$ é computada ao rotacionar o pixel *j* até seu gradiente apontar para o norte e o peso do pixel filtrado w_i e o valor de $u(i)$ são incrementados de acordo com o resultado da comparação.
5. Depois que a condição de parado foi alcançada, o *i* é atualizado para o valor filtrado $u(i)$.

4.10. Classificação dos pixels

O esquema de classificação dos pixels calcula uma série de parâmetros para cada pixel da imagem que serão posteriormente utilizados pelo algoritmo. Os parâmetros computados são:

1. Média ponderada quadrática
2. Desvio padrão ponderado quadrático
3. Média
4. Variância
5. Magnitude do gradiente
6. Orientação (ângulo) do gradiente

Segue-se um exemplo de classificação de pixels de uma imagem. A imagem original é a "lena" e ruído branco adicionado possuía um desvio padrão $\sigma_n = 20$. O kernel usado na filtragem e na classificação foi um kernel Gaussiano de 7x7. Cada parâmetro é apresentado sob forma de uma imagem. Em todas as imagens, áreas brancas representam valores altos e áreas escuras, valores baixos. As imagens, da esquerda para a direita e do topo para a parte de baixo são:

1. Imagem original.
2. Imagem com ruído ($\sigma_n = 20$).
3. Imagem representando a média ponderada quadrática.
4. Imagem representando o desvio padrão ponderado quadrático.
5. Imagem representando a variância, depois de removida a variância do ruído e mapeando-se todos os níveis acima de 255 para branco.
6. Imagem representando a média.
7. Imagem representando a magnitude do gradiente. As áreas escuras representam magnitudes baixas e as áreas brancas, magnitudes altas.
8. Imagem representando a orientação do gradiente. Para fins de visualização, há quatro níveis de cinza na imagem, indicando orientação para cima, para a direita, para baixo e para a esquerda.

Figura 18 Classificação dos pixels

4.11. Calibrando as expectativas

Há um limite inerente à aceleração que pode ser alcançada para o algoritmo proposto. Como o algoritmo usa a mesma medida de similaridade que o algoritmo original, devemos esperar o mesmo alto número de computações por comparação de pixels. A aceleração deve vir diminuindo-se o número de comparações envolvendo pixels e não através da aceleração de cada comparação. Os resultados devem ser comparados com o enfoque por "janela de busca" e não com o algoritmo por pesquisa exaustiva da imagem já que o último é impraticável.

Os mesmos parâmetros dos experimentos originais com o *NLM* [3] serão usados como referência: uma janela de busca de 21 x 21 pixels, um quadrado de semelhança de 7 x 7 pixels e um fator de decaimento de $h^2 = 10\sigma_n$. Com este esquema, o algoritmo original faz $21 * 21 - 1 = 440$ comparações por pixel.

No algoritmo proposto, o número de comparações depende das características do pixel filtrado. Para pixels em áreas uniformes, um número mínimo de comparações deve ser realizado para computar a média de cada pixel. Não devemos esperar que este número seja melhor do que outro algoritmo "simples" de filtragem. Se tomarmos como referência uma suavização Gaussiana com kernel 5 x 5, devemos esperar que cada pixel seja filtrado levando em consideração pelo menos 25 outros pixels. Mesmo se a pesquisa por cada um destes pixels for mais simples em áreas uniformes, a melhor aceleração possível é de cerca de 20 vezes.

Para áreas ativas, o número de comparações deve ser muito menor do que no algoritmo original já que o número de pixels semelhantes é menor do que o tamanho da janela de busca, e o número de pixels necessários para áreas ativas é normalmente menos do que em áreas uniformes levando em consideração a dificuldade do *HVS* em detectar variações em tais áreas. Isto mostra que o algoritmo proposto deverá ter um desempenho melhor para imagens muito detalhadas com poucas áreas uniformes. Como a maioria dos pixels de imagens naturais tende a pertencer a áreas uniformes, nossa expectativa para a aceleração diminui consideravelmente. Estimando-se que cerca de 50% dos pixels pertencem a áreas uniformes, uma aceleração de 30 vezes parece-nos ser o limite superior do que pode ser alcançado para imagens naturais com a nossa proposta.