# Verificação de Modelos Aplicada aos Filtros Espaciais em Processamento Digital de Imagens

**Manoel S. Lima, Endrews S. Silva, Lucas C. Cordeiro**

Programa de Pós Graduação em Engenharia Elétrica – Universidade Federal do Amazonas (UFAM)
Caixa Postal 69077-000– MANAUS – AM – Brazil

mslneto@gmail.com, endrews.eng@gmail, lucascordeiro@ufam.edu.br

***Abstract.*** *Spatial filtering is one of the main methods used in image processing, which aims to selectively highlight the features of high, medium, or low frequency that make up the images. To determine the efficiency of a filter, several parameters must be considered, e.g., preserving edges, noise, or even eliminating time complexity. The aim of this study is to compare two existing filters: Average Filter and Minimum Filter, from a formal verification perspective, based on image quality metrics using the Efficient SMT-Based Context-Bounded Model Checking tool.*

***Resumo.*** *Filtragem espacial é um dos principais métodos utilizados em processamento de imagens, o qual tem como objetivo realçar seletivamente as feições de alta, média ou baixa frequência que compõem as imagens. Para determinar a eficiência de um filtro, alguns parâmetros devem ser analisados como: preservação de bordas, eliminação do ruído ou até mesmo a complexidade de tempo. A partir do exposto, o objetivo deste trabalho é comparar dois filtros existentes: Filtro de Média e Filtro de Mínimo, através da ferramenta de verificação formal Efficient SMT-Based Context-Bounded Model Checking, baseando-se em métricas de qualidade de imagens.*

## 1 Introdução

A restauração de imagem é o processo de obtenção da imagem original a partir da imagem degradada, conhecendo-se os fatores degradantes. Restauração de imagem digital é o campo da engenharia que estuda os métodos utilizados para recuperar um cenário original, a partir da imagem degradada e observações, Khare e Nagwanshi (2012). Há uma variedade de fatores que podem degradar uma imagem como borramento, movimento e ruído, Andrews e Hunt (1977).

Filtros espaciais e espectrais são utilizados para manipular padrões de dados em 1D, 2D e 3D ou mais dimensões. Exceto alguns raros padrões espectrais, Kaur *et al*. (2012). Eles são projetados para serem utilizados em certo tipo de degradação e modelo de ruído, como exemplo, existe o filtro de média que pode ser usado para combater os ruídos gaussiano, uniforme e *Erlang*. Os filtros são essencialmente classificados em dois tipos: filtros no Domínio Espacial e filtros no Domínio de Transformação, Aboshosha *et al.* (2009).

Em Sharma e Singh (*2013*), foram utilizadas as métricas de relação sinal ruído de pico e correlação cruzadas 2D, para determinar a eficiência de diversos filtros no domínio espacial, aplicados aos diferentes modelos de ruídos em um conjunto de *benchmark* de imagens. De acordo com os autores, para os ruídos Gaussiano, *salt* and *peper*, *Poisson* e

*Gama*, o melhor filtro foi o de média. Os ruídos Uniforme e *Rayleigh* foram melhores corrigidos pelo filtro de mínimo e para o ruído exponencial, o melhor filtro foi o harmônico. No trabalho de Sharma e Singh não foi atribuído um limiar para definir o valor aceitável para cada métrica na análise dos filtros. Dessa forma, foi feita uma comparação entre todos os filtros. Os que tinham os maiores valores para todas as métricas foram considerados os melhores filtros.

Cordeiro *et al.* (2012) integraram os solucionadores CVC3, Boolector, e Z3 com o *front-end* do CBMC (C *Bounded Model Checker*). Os autores avaliariam os solucionadores usando *benchmarks* padrão para a verificação de modelos e aplicações embarcadas em telecomunicações, controle de sistemas e dispositivos médicos. Em seus experimentos, os autores verificaram que o solucionador Z3 é o mais eficiente e eficaz no processo de verificação, exceto para alguns programas ANSI-C que lidam com cópia de string e soma de vetores.

A partir do exposto, este trabalho propõe a utilização do verificador de modelos *Efficient SMT-Based Context-Bounded Model Checking* (ESBMC), Cordeiro *et al.* (2012), baseado nas técnicas *Bounded Model Checking* (BMC) e *Satisfiability Modulo Theories* (SMT), para determinar a eficiência dos filtros de Média e de Mínimo através de assertivas, que definem um limiar de aceitabilidade de suas métricas de desempenho, para diferentes níveis de ruído *salt* aplicados as imagens em nível de cinza.

A partir de um limiar para o valor de uma determinada métrica de qualidade (*e.g.*, erro médio absoluto, erro médio quadrático e relação sinal-ruído), e com o auxílio da ferramenta de verificação ESBMC, a contribuição mais importante deste trabalho é proporcionar uma nova metodologia para avaliar os resultados da aplicação de filtros espaciais, em imagens em escala de cinza corrompidas. A partir de um estudo realizado, este é o primeiro trabalho que utiliza a abordagem de verificação de modelos em processamento digital de imagens usando a técnica SMT-*based* BMC para determinar a eficiência de filtros.

## 2 Filtros Espaciais

No domínio do espaço, os filtros são classificados em três grupos: filtros de média, filtros de ordem estatística e filtros adaptativos. Nesta seção serão discutidos os filtros de média (média aritmética) e de ordem estatística (*Min* e *Max*).

### 2.1 Filtros de Média Aritmética

Filtros de média aritmética é um filtro de suavização que reduz a intensidade das variações entre *pixels*, Andrews e Hunt (1977). Também chamado de filtro linear ou filtro de média, opera com uma máscara *m x n* calculando a média de todos os *pixels* que estão nesta janela, substituindo o *pixel* central na imagem final com o resultado obtido $\hat{f}(x, y)$, causando certo grau de borramento na imagem, Marques (2011). O filtro de média aritmética pode ser definido pela seguinte equação:

$$\hat{f}(x, y) = \frac{1}{mn} \sum_{(r,c) \in W} g(r, c) \tag{1}$$

Em que *x e y* representam as coordenadas do *pixel* central da máscara na imagem original; *w* representa o conjunto de todos os *pixels* da máscara; $g(r, c)$ representa cada um dos

*pixels* da máscara localizados em um eixo arbitrário *r, c; m e n* representam o número de linhas e colunas da máscara do filtro, respectivamente.

### 2.2 Filtros de *Min* e *Max*

Em vez de substituir o valor do *pixel* de referência pelo valor médio do conjunto ordenado, filtro de mediana, o filtro *Min* substitui o *pixel* de referência pelo menor valor, Gonzalez (2009). Similarmente o filtro *Max* substitui o *pixel* de referência pelo valor mais alto. Os filtros de *Min* e *Max* podem ser definidos pelas seguintes equações:

$$\hat{f}(x,y) = min\{g(r,c)|(r,c) \in W\} \tag{2}$$

$$\hat{f}(x,y) = max\{g(r,c)|(r,c) \in W\} \tag{3}$$

Em que $\hat{f}(x,y)$ representa o resultado obtido para o *pixel* de referência localizado na imagem original; *w* representa o conjunto de todos os *pixels* da máscara; $g(r,c)$ representa cada um dos *pixels* da máscara localizados em um eixo arbitrário r, c.

## 3 Métricas de Qualidade de Imagem

As próximas subseções descrevem os critérios utilizados para definir qual filtro é melhor para a eliminação do ruído *salt*.

### 3.1 Erro Médio Absoluto

O erro médio absoluto (MAE, do inglês, *Mean Absolute Error*) é a soma da diferença absoluta de cada ponto da imagem original, $f(x,y)$ e da imagem aproximada, $g(x,y)$ dividido pela multiplicação das dimensões da imagem, Kassam (1977). Esse valor é expresso como:

$$MAE = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}|f(x,y) - g(x,y)| \tag{4}$$

Em que *M* e *N* representam as dimensões da imagem; *x* e *y* representam as coordenadas dos *pixels* das duas imagens.

### 3.2 Erro Médio Quadrático

O erro médio quadrático (MSE, do inglês, *Mean Square Error*) é a soma do quadrado das diferenças de cada ponto da imagem original, $f(x,y)$ e da imagem aproximada, $g(x,y)$ dividido pela multiplicação das dimensões da imagem, Murray; Rodriguez e Pattichis (2007). Esse valor é expresso como:

$$MSE = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}(f(x,y) - g(x,y))^2 \tag{5}$$

Em que *M* e *N* representam as dimensões da imagem; *x* e *y* representam as coordenadas dos *pixels* das duas imagens.

### 3.3 Relação Sinal-Ruído (SNR)

SNR é definido como a razão entre a potência média do sinal e a potência média do ruído, Damera-Venkata *et al.*(2000). O seu valor é expresso como:

$$SNR = 10log_{10} \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left(f(x,y)\right)^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (f(x,y) - g(x,y))^2} \quad (6)$$

Em que *M* e *N* representam as dimensões da imagem; $f(x,y)$ e $g(x,y)$ representam os *pixels* da imagem original e com ruído, respectivamente.

## 4 Verificação de Modelos usando o ESBMC

ESBMC é um verificador de modelos de contexto limitado para programas C/C++, o qual verificar estouro de limites de vetores, divisão por zero e assertivas definidas pelo usuário em programas sequências e multitarefas Cordeiro *et al.* (2011).

A ferramenta ESBMC faz uso dos componentes do C *Bounded Model Checker* (CBMC[1]) que é um verificador de modelos que utiliza solucionadores de *Boolean Satisfiability* (SAT). O ESBMC é capaz de modelar, em um sistema de transição de estados, o programa a ser analisado, Cordeiro *et al.* (2012) e Januário *et al.* (2013). Dado um sistema de transição de estados *M*, uma profundidade *k* e uma propriedade $\phi$, o ESBMC desdobra o sistema de transição *k* vezes e o traduz em uma condição de verificação $\psi$ tal que $\psi$ é satisfatível se e somente se $\phi$ possuir um contraexemplo de profundidade *k* ou menor. O ESBMC também permite estender essa abordagem de verificação de violação de propriedade para programas complexos, que possuam muitas iterações, Cordeiro *et al.* (2012). Na figura a seguir mostra-se a arquitetura do ESBMC.

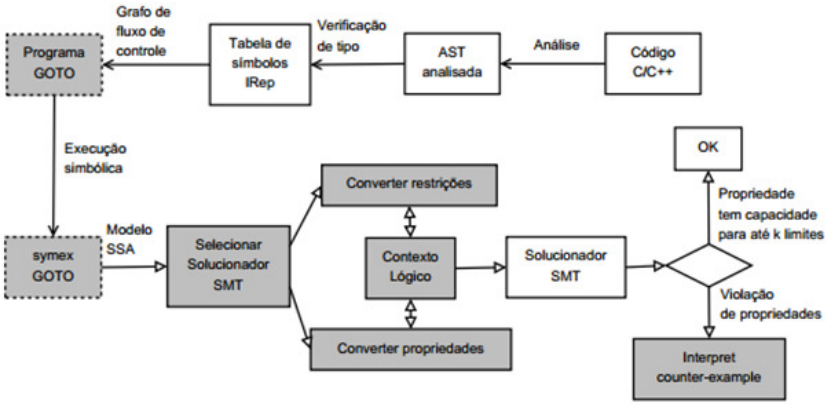


**Figura *3.* Arquitetura do ESBMC, Cordeiro *et al.* (2012).**

O ESBMC converte um programa ANSI-C/C++ em um programa *GOTO*, ou seja, transformando expressões como *switch* e *while* em instruções *goto*, que é então simulado simbolicamente pelo compoenente *symex GOTO*. Então, um modelo em *Single Static Assignment* (SSA) é gerado, com a atribuição de valores estáticos as propriedades, para ser verificado por um solucionador SMT adequado.

[1] Disponível em: http://www.cprover.org/cbmc/

## 5 Metódo de verificação formal para comparação de filtros

Nesta seção será apresentada a metodologia utilizada para a realização deste trabalho, sendo essa descrita em duas subseções: obtenção das imagens com ruído e aplicação dos filtros e a análise de desempenho dos filtros.

### 5.1 Obtenções das imagens com ruído e aplicação dos filtros

Para a análise de filtros espaciais é necessário a obtenção de imagens com ruídos. A obtenção dessas imagens é feita de maneira simples. A metodologia consiste em aplicar o ruído *salt* em uma matriz *NxN,* representando uma imagem. O ruído é aplicado em posições não-determinísticas da matriz. Ao fim desse processamento obtém-se a imagem com ruído, "*Matrix_Noise*".

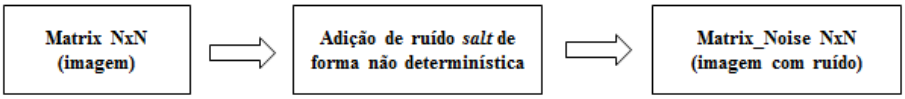


**Figura 5. Diagrama da obtenção das imagens com ruído.**

A Figura 6 apresenta o pseudocódigo para a tarefa de aplicação de ruído em uma imagem ou matriz *NxN*:

```
unsigned int nondet_uint();
int N;

int Matrix_Noise[N][N],noise_applied,
noise_settled;

int line, column, position, size;

while (noise_applied <= noise_settled)
{
    size<-N*N;

    position <- nondet_uint() mod size;

    line <- position/N;

    column <- position mod N;

    if(Matrix_Noise[line][column] <> 255)
    {
        Matrix_Noise[line][column] <- 255;
        noise_applied++;
    }
}
```

**Figura 6. Procedimento para a obtenção de imagens corrompidas em posições não determinísticas.**

O código acima apresenta o método de aplicação dos ruídos. Sabe-se que uma imagem em escala de cinza é bi-dimensional, porém, para que o ruído pudesse ser aplicado na imagem utilizando valores não-determinísticos, foi necessário transformar a matriz em um vetor unidimensional, aplicando o valor de 255, ruído *salt*, em *pixels* da imagem, separando o valor não-determinístico em linha e coluna.


**Figura 7. Diagrama da aplicação do filtro em imagens corrompidas**

Após a obtenção da imagem com ruído, tem-se a fase de pré-processamento da imagem que pode ser descrita, nesse caso, como uma preparação para a aplicação do filtro espacial. Nessa fase é realizada a aplicação de zeros nas bordas, conhecida como *padding*,

logo após o pré-processamento, inicia-se a fase de restauração da imagem, ou seja, remoção de ruído, caracterizada pela aplicação do filtro na imagem.

### 5.2 Análise de desempenho dos filtros espaciais

Depois da obtenção das imagens corrompidas, escolhem-se as métricas para análise de qualidade de imagens. Através dessas métricas compara-se a eficiência entre os filtros quando aplicados em imagens corrompidas com o ruído *salt*. O teste das métricas é realizado pelo ESBMC, o *software* realiza todos os possíveis testes de aplicação de ruído, verificando se as métricas estão no limiar definido pra cada uma delas. O processo de um caso de teste é mostrado no diagrama a seguir:

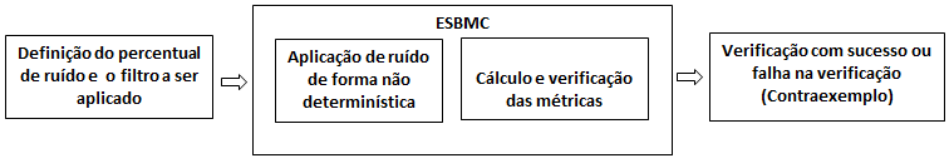


**Figura 8. Esquema de verificação formal proposta neste trabalho.**

O resultado de uma verificação de propriedade, de acordo com a figura 8, poderá ser verificação com sucesso, ou falha na verificação (contraexemplo), esses resultados dependem da relação entre a imagem filtrada e a métrica, que pode ser proporcional ou inversamente proporcional.

## 6 Avaliação Experimental

Esta seção descreve o projeto e execução dos experimentos para a análise do método de verificação formal proposto. O ambiente e ferramentas utilizadas para executar os experimentos deste trabalho foram: computador Intel Core i7 3.4 GHz, com 24 GB de RAM; Plataforma Linux *Ubuntu* 11.10 *Meerkat* 64-bits; ESBMC v1.23 com solucionador z3 v4.0; *MatLab* R2009b – 64 bits; 10 Imagens em escala de cinza. Os filtros a serem comparados são os filtros de média e de mínimo.

**Figura 4. Imagens utilizadas neste experimento.**

### 6.1 Objetivos dos Experimentos

Os experimentos realizados nesse trabalho tem o objetivo de investigar a eficácia do verificador de códigos para filtros espaciais. As características analisadas são:

**C1**: A quantidade de casos de teste que o verificador é capaz de executar em função das imagens de entrada, sem atingir o *timeout* de 1 hora de processamento;

**C2**: A capacidade do verificador comparar a qualidade dos filtros espaciais.

Para a obtenção de **C2**, o limiar escolhido para a verificação de propriedade de um filtro, será a média aritmética dos *pixels* das imagens. As métricas escolhidas para

este trabalho são: Erro Médio Quadrático, Erro Médio Absoluto, e a Relação Sinal Ruído. Após essas definições, pode-se realizar os casos de teste, utilizando a seguinte função:

1) *assert*(Métrica <= Limiar_da_Imagem) para as métricas MSE e o MAE;

2) *assert*(Métrica >= Limiar_da_Imagem ) para a métrica SNR.

Em que *assert* é a função das assertivas utilizada nos casos de teste deste trabalho; Métrica representa o valor calculado para cada uma das métricas após aplicação de um filtro. Para a obtenção do custo deste experimento, **C1**, em relação ao processo de verificação de propriedade, após a aplicação do ruído e do filtro, algumas considerações devem ser feitas: (1) tem-se *NxN* combinações possíveis para inserir ruídos na matriz; (2) tem-se um laço com um limite superior determinado por *noise_settled*. O Quadro a seguir apresenta o número de casos de teste para verificar as propriedades no melhor e no pior caso, respectivamente:

**Tabela 1. Custo dos testes.**

| Casos | Quantidade de Testes |
|---|---|
| Melhor Caso (Filtro de Média) | $C_{noise\_settled}^{NXN} = \frac{64!}{(64-3)\,!\,3!} = 41664$ |
| Pior Caso (Filtro de Mínimo) | $C_{noise\_settled}^{NXN} = \frac{64!}{(64-32)\,!\,32!} = 1{,}832 * 10^{18}$ |

Em que $noise\_settled = 3$ e $noise\_settled = 32$ representam as quantidades mínima (5%) e máxima (50%) de ruído aplicado em uma imagem, respectivamente; $N = 8$ representa as dimensões da imagem quadrada.

### 6.2 Limitações dos Experimentos

O presente trabalho consiste de uma abordagem inicial para a verificação de códigos de filtros espaciais e para a comparação entre eles. Neste trabalho apenas dois filtros foram comparados e um tipo de ruído foi utilizado para corromper as imagens do *benchmark*. Uma versão mais robusta poderá incluir mais filtros e mais ruídos, além de realizar testes empíricos com imagens de maior dimensão, pois para a realização dos experimentos deste trabalho foi necessário redimensionar as imagens.

## 7 Resultados e Discussões

Neste experimento, as imagens do *benchmark* foram corrompidas com o ruído *salt* em 5% a 50% de suas posições. Desta forma, o tempo de verificação de propriedade poderá ser avaliado de acordo a quantidade de ruído de cada imagem e o número de iterações dos *loops* do algoritmo. As Tabelas 2-4 mostram os resultados obtidos para a comparação dos filtros de média e de mínimo.

A primeira coluna *R* apresenta a quantidade de ruído aplicada nas imagens de teste, a segunda coluna *LC* exibe o número de linhas de código, a terceira coluna *LI* mostra o limite de iterações do *loop* realizadas. *P* indica a quantidade de propriedades verificadas pelo programa ANSI-C, que para este trabalho será a avaliação das métricas de qualidade de imagens. A avaliação consiste em verificar se, no caso dos erros Médio Absoluto e Médio Quadrático, o valor calculado é menor ou igual à média dos *pixels* da imagem, pois quanto menor o valor dessas métricas melhor a qualidade da imagem. Para a Relação

Sinal Ruído, o valor calculado deverá ser maior ou igual à média dos *pixels* da imagem, pois quanto maior o valor dessa métrica, mais a imagem com o filtro se assemelha da original. O tempo *TE* fornece o tempo médio para checar as propriedades do programa ANSI-C.

**Tabela 2. Resultados Experimentais – Média Geral (10 imagens) – Métrica: Erro Médio Absoluto, (a) filtro de média e (b) filtro de Mínimo.**

**(a)**

| Filtro | R | LC | LI | P | TE (seg) |
|---|---|---|---|---|---|
| Média | 5% | 101 | 10 | 1 | <1 |
| | 10% | 101 | 10 | 1 | <1 |
| | 15% | 101 | 11 | 1 | <1 |
| | 20% | 101 | 14 | 1 | <1 |
| | 25% | 101 | 17 | 1 | <1 |
| | 30% | 101 | 20 | 1 | <1 |
| | 35% | 101 | 23 | 1 | <1 |
| | 40% | 101 | 27 | 1 | <1 |
| | 45% | 101 | 30 | 1 | <1 |
| | 50% | 101 | 33 | 1 | <1 |

**(b)**

| Filtro | R | LC | LI | P | TE (min) |
|---|---|---|---|---|---|
| Mínimo | 5% | 104 | 10 | 1 | 2 |
| | 10% | 104 | 10 | 1 | 2 |
| | 15% | 104 | 11 | 1 | 2 |
| | 20% | 104 | 14 | 1 | 2 |
| | 25% | 104 | 17 | 1 | <1 |
| | 30% | 104 | 20 | 1 | <1 |
| | 35% | 104 | 23 | 1 | <1 |
| | 40% | 104 | 27 | 1 | <1 |
| | 45% | 104 | 30 | 1 | <1 |
| | 50% | 104 | 33 | 1 | <1 |

De acordo com as Tabelas 2(a) e 2(b), verifica-se que o ESBMC precisou de menos de 1 segundo para a verificação das métricas do filtro de média, e de no máximo 2 minutos para o filtro de mínimo. A verificação das métricas do filtro de mínimo é um processo computacionalmente mais custoso do que o filtro de média, devido ao procedimento de comparação entre *pixels* existente neste filtro, que resulta em mais iterações dos *loops* do algoritmo. Para o filtro de mínimo percebe-se também, que para percentuais de ruído acima de 25%, o tempo de verificação é menor que para percentuais menores. Isso acontece devido à distribuição do ruído não ser uniforme na imagem.

**Tabela 3. Resultados Experimentais – Média Geral (10 imagens) – Métrica: Erro Médio Quadrático, (a) filtro de média e (b) filtro de Mínimo.**

**(a)**

| Filtro | R | LC | LI | P | TE (seg) |
|---|---|---|---|---|---|
| Média | 5% | 97 | 10 | 1 | <1 |
| | 10% | 97 | 10 | 1 | <1 |
| | 15% | 97 | 11 | 1 | <1 |
| | 20% | 97 | 14 | 1 | <1 |
| | 25% | 97 | 17 | 1 | <1 |
| | 30% | 97 | 20 | 1 | <1 |
| | 35% | 97 | 23 | 1 | <1 |
| | 40% | 97 | 27 | 1 | <1 |
| | 45% | 97 | 30 | 1 | <1 |
| | 50% | 97 | 33 | 1 | <1 |

**(b)**

| Filtro | R | LC | LI | P | TE (min) |
|---|---|---|---|---|---|
| Mínimo | 5% | 100 | 10 | 1 | 2 |
| | 10% | 100 | 10 | 1 | 2 |
| | 15% | 100 | 11 | 1 | 2 |
| | 20% | 100 | 14 | 1 | 2 |
| | 25% | 100 | 17 | 1 | 5 |
| | 30% | 100 | 20 | 1 | 5 |
| | 35% | 100 | 23 | 1 | 17 |
| | 40% | 100 | 27 | 1 | 26 |
| | 45% | 100 | 30 | 1 | 36 |
| | 50% | 100 | 33 | 1 | 38 |

De acordo com as Tabelas 3(a) e 3(b), verifica-se que o ESBMC precisou de menos de 1 segundo para a verificação das métricas do filtro de média, e de no máximo 38 minutos para o filtro de mínimo. Para o filtro de mínimo, percebe-se também, que o maior tempo de verificação da propriedade deste filtro foi para 50% de ruído aplicado.

**Tabela 4. Resultados Experimentais – Média Geral (10 imagens) – Métrica: Relação Sinal Ruído, (a) filtro de média e (b) filtro de Mínimo.**

**(a)**

| Filtro | R | LC | LI | P | TE (seg) |
|---|---|---|---|---|---|
| Média | 5% | 98 | 10 | 1 | <1 |
| | 10% | 98 | 10 | 1 | <1 |
| | 15% | 98 | 11 | 1 | <1 |
| | 20% | 98 | 14 | 1 | <1 |
| | 25% | 98 | 17 | 1 | <1 |
| | 30% | 98 | 20 | 1 | <1 |
| | 35% | 98 | 23 | 1 | <1 |
| | 40% | 98 | 27 | 1 | <1 |
| | 45% | 98 | 30 | 1 | <1 |
| | 50% | 98 | 33 | 1 | <1 |

**(b)**

| Filtro | R | LC | LI | P | TE (min) |
|---|---|---|---|---|---|
| Mínimo | 5% | 101 | 10 | 1 | 2 |
| | 10% | 101 | 10 | 1 | 2 |
| | 15% | 101 | 11 | 1 | 2 |
| | 20% | 101 | 14 | 1 | 2 |
| | 25% | 101 | 17 | 1 | 5 |
| | 30% | 101 | 20 | 1 | 5 |
| | 35% | 101 | 23 | 1 | 8 |
| | 40% | 101 | 27 | 1 | 10 |
| | 45% | 101 | 30 | 1 | 11 |
| | 50% | 101 | 33 | 1 | 11 |

De acordo com as Tabelas 4(a) e 4(b), verifica-se que o ESBMC precisou de menos de 1 segundo para a verificação das métricas do filtro de média, e de no máximo 11 minutos para o filtro de mínimo. Para o filtro de mínimo percebe-se também, que para percentuais de ruído de 45% e 50% o tempo de verificação é o maior.

A partir dos resultados obtidos, observa-se que independente da quantidade de ruído aplicado nas imagens, o tempo de resposta do ESBMC é rápido para o filtro de média. Em especial, este filtro atendeu a propriedade para as três métricas. Observa-se também, que o tempo de resposta do ESBMC aumentou com a quantidade de ruído aplicada e o número de iterações necessárias para o filtro de mínimo. Este filtro não atendeu a propriedade para as três métricas. Com os experimentos acima, foi possível comprovar **C2**, pois, o filtro de média não violou a propriedade para as três métricas, comprovando ser mais eficiente que o filtro de mínimo, para recuperação de imagens corrompidas com o ruído *salt*. Os experimentos foram realizados em menos de 1 hora de processamento, comprovando **C1**. Em imagens com dimensões *16x16 pixels,* o tempo de resposta ESBMC atingiu *timeout*.

## 8 Conclusões e Trabalhos Futuros

Pode-se comprovar que, fazendo uso da verificação de modelos, é possível realizar a comparação de dois ou mais filtros, que fazem parte do conjunto de *benchmarks* na área de processamento digital de imagens. No domínio espacial, identifica-se que o filtro de média possui melhores resultados no caso da retirada de ruído *salt*, em um conjunto de imagens em nível de cinza , do que o filtro de mínimo, levando em consideração as assertivas inseridas no verificador de modelos ESBMC.Para situações de falha, o tempo de resposta do ESBMC mostrou-se proporcional à quantidade de ruído aplicado, enquanto os testes que obtiveram sucesso foram verificados rapidamente.

Como trabalhos futuros, sugere-se a realização dos testes com imagens maiores, fazendo uso de programação paralela, além da utilização de outros filtros, como o filtro de máximo e o de mediana, além de outras métricas de qualidade.

## 9 Referências


Khare, C. and Kumar Nagwanshi, K.(2012) “Image Restoration Technique with Non Linear Filter”, *In: International Journal of advanced Science and Technology*, vol. 39, pp. 67-74.

Andrews, H.C. and Hunt,B.R. (1977).“Digital Image restoration”, Englewood cliffs, NJ, Prentice Hall.

Kaur, J.; Kaur, M.; Kaur, P.; Kaur, M. (2012). “Comparative Analysis of Image Denoising Techniques”, *In: International Journal of Emerging Technology and Advanced Engineering*, vol. 2, pp. 160-167.

Aboshosha, A.; Hassan, M.; Ashour, M.; Mashade, M. El (2009). “Image denoising based on spatial filters, an analytical study”, In: *International Conference on Computer Engineering & Systems*, pp. 245-250.

Marques, O. (2011).Practical Image and Video Processing Using MATLAB, John Wiley & Sons.

Gonzalez, R. C., (2009). Digital Image Processing, Pearson Education India.

Damera-Venkata, N.; Kite, T.D.; Geisler, W.S.; Evans, B.L.; Bovik, AC., (2000) “Image quality assessment based on a degradation model”, *Image Processing, IEEE Transactions on*, vol.9, no.4, pp. 636-650.Murray, V.; Rodriguez, P.V.; Pattichis, M.S. (2007)."Robust Multiscale AM-FM Demodulation of Digital Images", *Image Processing, 2007.ICIP 2007. IEEE International Conference on*, vol.1, no., pp. 465–468.

Kassam, S.A, (1977) “The mean-absolute-error criterion for quantization”, *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '77*, vol.2, no., pp. 632- 635.

Januário, F.; Cordeiro, Lucas C.; e de Lima Filho, E. B.(2013) “Verificacão de códigos Lua Utilizando BMCLua”, *XXXI Simpósio brasileiro de telecomunicações, pp.1-6.*

Cordeiro, L.C.; Fischer, B.; Marques-Silva, J. (2012) “SMT-Based Bounded Model Checking for Embedded ANSI-C Software”. *In: IEEE Trans. Software Eng., v. 38, n. 4, pp. 957–974.*

Cordeiro, L. and Fischer, B (2011). “Verifying Multi-threaded Software using SMT-based Context-Bounded Model Checking”. *In: International Conference on Software Engineering (ICSE)*, pp. 331-340.

Sharma, A.; Singh, J. (*2013*) “Image denoising using spatial domain filters: A quantitative study”, *Image and Signal Processing (CISP), In: 6th International Congress on*, vol.01, pp. 293-298.