

Visão por Computador

Tutorial 1

João Pimentel (a80874) Carolina Cunha (a80142)

Novembro de 2019

Universidade do Minho
Mestrado Integrado em Engenharia Informática

Resumo

O presente trabalho baseou-se no desenvolvimento de funções capazes de aplicar filtros Passa-Baixo a imagens, quer no domínio espacial, quer no domínio das frequências, bem como implementar o método de *Canny* para deteção de *edges*. Para tal, recorreu-se à linguagem de programação *Matlab*, a qual possui uma vasta biblioteca de ferramentas para processamento de imagem.

A elaboração deste projeto permitiu aumentar a capacidade de desenvolvimento e compreensão dos algoritmos referentes a processamento de imagem e, consequentemente, retirar conclusões sobre o impacto causado pelos diversos parâmetros inerentes a cada filtro.

Conteúdo

1	Introdução	1
2	Análise e Especificação	2
2.1	Descrição do Projeto	2
2.2	Especificação de Requisitos	2
3	Concepção da Resolução	4
3.1	Suavização de Imagens Ruidosas	4
3.1.1	Aplicação de Ruído	4
3.1.2	Filtros Espaciais	5
3.1.2.1	Filtro Gaussiano	5
3.1.2.2	Filtro Média	6
3.1.2.3	Filtro Mediana	6
3.1.3	Filtros no Domínio da Frequência	7
3.1.3.1	<i>Padding</i> da Imagem	7
3.1.3.2	Centrar a Imagem	8
3.1.3.3	Transformada Discreta de Fourier	8
3.1.3.4	Filtro	9
3.1.3.4.1	Filtro Gaussiano	9
3.1.3.4.2	Filtro Butterworth	10
3.1.3.5	Inversa da Transformada de Fourier	10
3.1.3.6	Descentralização da Imagem	11
3.1.3.7	<i>Crop</i> da Imagem	11
3.1.4	Filtros Espaciais <i>vs</i> Filtros de Frequência	12
3.2	Deteção de Arestas com o Método <i>Canny</i>	12
3.2.1	Suavização com um Filtro Gaussiano	12
3.2.2	Gradiente	12
3.2.3	<i>Non-Max Suppression</i>	13
3.2.4	<i>Threshold</i> Duplo	13
3.2.5	<i>Threshold</i> de Histerese	14
3.2.6	Influência da Suavização	15
4	Conclusões e Trabalho Futuro	16

Lista de Figuras

1	Figura Original - <i>Baboon</i>	4
2	Ruído <i>Salt and Pepper</i> e Ruído Gaussiano	4
3	Ruído <i>Salt and Pepper</i> com Densidade 20%	5
4	Resultados de Aplicação de Filtros Gaussianos com Desvio 3, 5 e 10, Respetivamente	6
5	Resultados de Aplicação de Filtros Média com Dimensão 5, 10 e 20, Respetivamente	6
6	Resultados de Aplicação de Filtros Mediana com Dimensão 5, 10 e 20, Respetivamente	7
7	<i>Zero Padding</i> da Imagem	8
8	Centralização da Imagem	8
9	Transformada Discreta de Fourier da Imagem	9
10	Representação Gráfica do Filtro Gaussiano	9
11	Aplicação de Filtros Gaussianos com Desvio Padrão 1, 3, 7 e 10, Respetivamente . .	10
12	Efeito da Ordem do Filtro em Artefactos de <i>Ringing</i>	10
13	Aumento do Diâmetro e Suavização em Filtros Butterworth	10
14	Inversa da Transformada de Fourier	11
15	Descentralização da Imagem	11
16	Resultado da Aplicação de um Filtro no Domínio da Frequência	11
17	Suavização com um Filtro Gaussiano	12
18	Magnitude do Gradiente da Imagem	13
19	Aplicação de <i>Non-Max Suppression</i> à Imagem	13
20	Aplicação de um <i>Threshold</i> Duplo à Imagem	14
21	Conceito de <i>Threshold</i> de Histerese com <i>Blobs</i>	14
22	Resultado Final da Aplicação do Método de Canny com Dimensão 5 e Desvio Padrão 3	14
23	Resultados Finais do Método de Canny com Desvios de 1, 3, 7 e 10, Respetivamente	15

Lista de Extratos

1	Extrato 1 - Código <i>Matlab</i> para Filtro Gaussiano.	5
2	Extrato 2 - Código <i>Matlab</i> para Filtro Média.	6
3	Extrato 3 - Código <i>Matlab</i> para Filtro Mediana.	7

1 Introdução

O presente relatório é o resultado da resolução do primeiro trabalho prático da unidade curricular de Visão por Computador.

O foco deste trabalho passou por implementar e analisar o efeito da aplicação de filtros Passa-Baixo e o método de deteção de *edges*, *Canny*. Para tal, teve-se por base a utilização das ferramentas de processamento de imagem fornecidas pela linguagem *Matlab*, bem como os algoritmos aprendidos nas aulas lecionadas.

Este projeto baseou-se em definir duas grandes funções, que deveriam aplicar um determinado algoritmo consoante o objetivo em questão. É de destacar que cada função tem os seus *inputs* específicos, bem como *outputs*, sendo que estes variam dependendo dos parâmetros passados à função.

Dito isto, o grande objetivo deste projeto foi o aumento da experiência dos alunos no âmbito do processamento de imagem. Quer isto dizer que se pretendia conseguir o desenvolvimento de ferramentas que implementassem, passo a passo, algoritmos bastante utilizados para suavização de imagens, como é o caso de um filtro Gaussiano, e para a deteção de *edges*, como o método *Canny*.

Nos capítulos seguintes serão demonstrados os problemas, formas de resolução e testes efetuados de modo a obter os resultados ideais para o problema proposto.

2 Análise e Especificação

2.1 Descrição do Projeto

O projeto em questão consistiu na realização de duas grandes tarefas. A primeira consistia no desenvolvimento de uma função que permitisse efetuar a aplicação de um filtro Passa-Baixo sobre uma imagem, podendo este ser no domínio espacial ou das frequências. Note-se que, à imagem de *input*, deve ser adicionado ruído, ficando o tipo do mesmo ao critério do utilizador. Já a segunda consistia na realização de uma função que implementasse o método *Canny*, ou seja, que apliquesse um conjunto de passos pré-definidos, resultando numa imagem a preto e branco, em que se realçam as *edges* da imagem.

2.2 Especificação de Requisitos

Anteriormente foram mencionados, de uma forma muito geral, os requisitos que descrevem o projeto. Em seguida, serão mencionados detalhadamente os mesmos.

Para a questão 1 tem-se:

1. Definir uma função que receba como parâmetros uma imagem em tons de cinzento, tipo de ruído, parâmetros de ruído, domínio do filtro, tipo de *smoothing* e parâmetros do filtro;
2. Tipo de ruído: Gaussiano ou *salt and pepper*;
3. Parâmetros de ruído: variação, percentagem;
4. Domínio do filtro: Espacial ou Frequência;
5. Tipo de *smoothing*:
 - (a) Espacial: Mediana, Gaussiano, Média;
 - (b) Frequência: Gaussiano, Butterworth;
6. Parâmetros do filtro:
 - (a) Dimensão do filtro;
 - (b) Gaussiano: Desvio Padrão;
 - (c) Butterworth: Ordem do filtro;
7. Definir uma função para introduzir ruído na imagem;
8. Aplicar um filtro no domínio espacial, sendo que o processo deve ser repetido para máscaras com dimensão 5, 10 e 20, bem como várias variâncias;
9. Determinar a Transformada Discreta de Fourier para a imagem original e com ruído, comparando;
10. Aplicar filtros no domínio das frequências, para vários parâmetros, analisando a presença de *ringing*;
11. Comparar os filtros de frequência com espaciais, em condições semelhantes, analisando eficiência e precisão.

Para a questão 2 tem-se:

1. Definir um função que aplique o método *Canny* e receba uma imagem com ruído;
2. Efetuar o *smoothing* da imagem através da aplicação de um filtro gaussiano no domínio espacial, comparando o impacto do valor do desvio padrão;
3. Calcular os gradientes vertical e horizontal da imagem. Calcular, ainda, a magnitude e orientação do gradiente;

4. Encontrar a melhor orientação para cada *edge*, entre os valores 0, 45, 90 ou 135 graus, tendo em conta a orientação do gradiente na *edge* em questão;
5. Melhorar as *edges* aplicando:

$$E_m(i, j) = \begin{cases} 0, & \text{se } E_s(i, j) \text{ é menor que um dos dois vizinhos,} \\ & \text{tendo em conta a orientação da } edge; \\ E_s(i, j), & \text{caso contrário.} \end{cases}$$

6. Aplicar um *threshold* duplo;
7. Aplicar um *threshold* de histerese.

3 Concepção da Resolução

De modo a executar e descrever de forma simples e direta todo o processo de resolução do enunciado, foi definido que se explicaria todo este, passo a passo, demonstrando o impacto da alteração de parâmetros, bem como os resultados obtidos.

3.1 Suavização de Imagens Ruidosas

3.1.1 Aplicação de Ruído

De modo a aplicar ruído a uma imagem, foi utilizada uma função fornecida pela linguagem, em que esta recebe o tipo de ruído, bem como os parâmetros a este associados. Assim, através da utilização da função *imnoise*, é facilmente obtida uma imagem ruídososa a partir de uma imagem em tons de cinzento.

Na Figura 1 é possível observar a figura original, antes de ser inserido qualquer ruído.

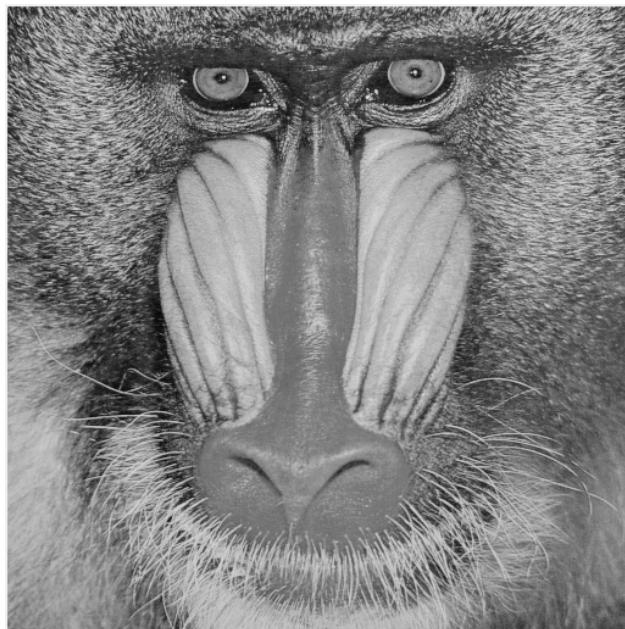


Figura 1 - Figura Original - *Baboon*.

Em seguida, tenha-se a Figura 2 onde, na imagem da esquerda, foi aplicado ruído do tipo *salt and pepper* com densidade de 0.02 e, na da direita, ruído gaussiano, onde a média e a variância tomam valores de 0.02 e 0.01, respectivamente. Como se pode ver, o primeiro ruído coloca píxeis pretos e brancos em posições aleatórias, ao invés do ruído gaussiano, que segue uma função, aplicando ruído a toda a imagem.

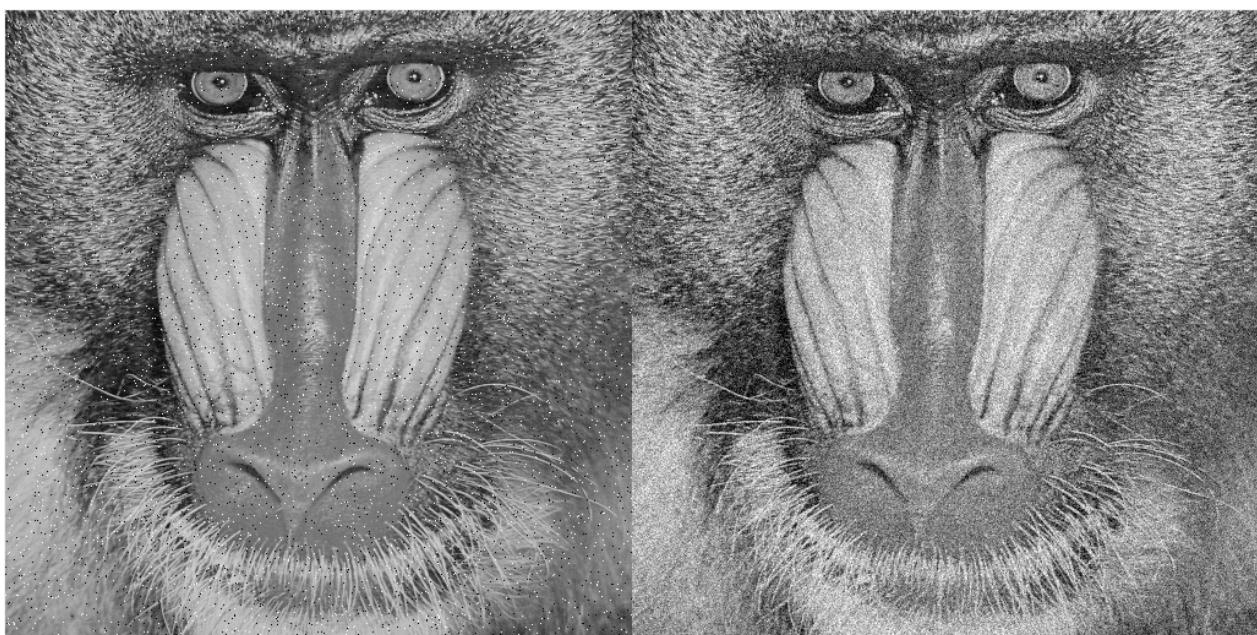


Figura 2 - Ruído *Salt and Pepper* e Ruído Gaussiano.

Aumentando, por exemplo, a densidade de ruído *salt and pepper*, a imagem perde mais qualidade, sendo possível observar uma maior quantidade de píxeis ruídosos, como se vê na Figura 3.

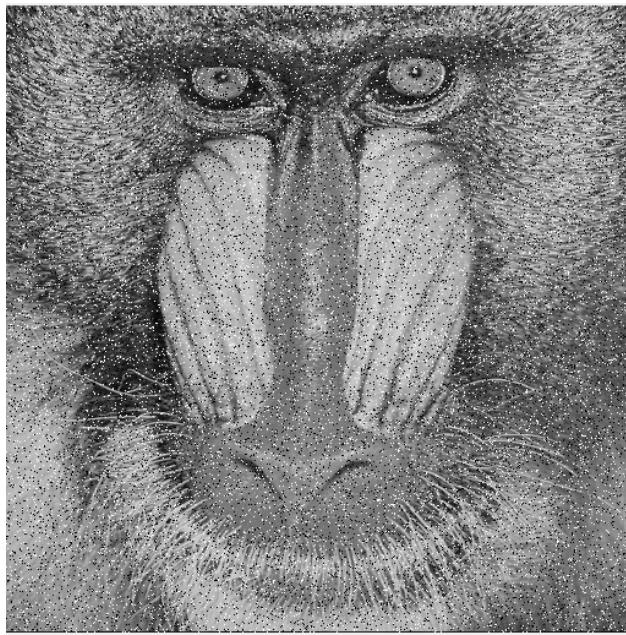


Figura 3 - Ruído *Salt and Pepper* com Densidade 20%.

3.1.2 Filtros Espaciais

Um filtro Passa-Baixo do tipo *Box-Filter* é um filtro que, após a sua aplicação a uma imagem, permite obter uma imagem mais suavizada, ou seja, uma imagem menos nítida em que transições abruptas de intensidade são reduzidas (componentes de alta frequência). Deste modo, permite, então, reduzir o ruído numa imagem. Além disso, é de destacar que todos os pesos na máscara são positivos e a sua soma é igual a 1. Tendo isto em conta, são filtros que assumem píxeis semelhantes à sua vizinhança e ruído independente de píxel para píxel. Neste caso, os filtros deste tipo que são retratados são o gaussiano e o filtro média.

No entanto, o enunciado permite um tipo de filtro espacial extra, o filtro mediana. Este é um filtro não linear, tendo, por isso, características distintas que serão mencionadas no seu sub-capítulo.

Tirando proveito das funcionalidades da linguagem, a utilização da função *imfilter* permite filtrar uma imagem, através da aplicação da operação correlação de uma máscara fornecida. Assim, tenham-se em seguida as explicações de forma detalhada.

3.1.2.1 Filtro Gaussiano

Uma máscara gaussiana é caracterizada por ser uma matriz de dimensão N , tendo por base um certo desvio padrão. Dito isto, a ideia passa por gerar um filtro Passa-Baixo, ou seja, uma máscara, que será aplicada a uma determinada imagem. Para isso, utiliza-se a função *fspecial*, que, recebendo a dimensão, tipo de filtro e desvio padrão (no caso do filtro gaussiano), retorna a matriz pretendida. Note-se, ainda, que os píxeis mais próximos do centro possuem um peso mais elevado na operação de filtragem.

Após a aplicação desta função, apenas é necessário efetuar a operação que percorre toda a matriz, sendo, neste caso, a correlação, como se pode ver no Extrato 1.

```
% Gaussian
kernel = fspecial(type_of_smoothing, [filter_parameters.width_of_filter filter_parameters.
width_of_filter], filter_parameters.standard_deviation);
smoothed_image = imfilter(image_with_noise, kernel);
```

Extrato 1 - Código *Matlab* para Filtro Gaussiano.

Sendo assim, o passo seguinte é compreender de que forma cada um destes componentes, o desvio padrão (σ) e a dimensão da máscara, afetam a imagem final. Note-se que a dimensão da máscara deve ser igual a 2σ ou 3σ , segundo a regra do polegar.

Como se pode ver pela Figura 4, o aumento do desvio padrão resulta em imagens mais suavizadas, ou seja, implica uma redução maior no detalhe.

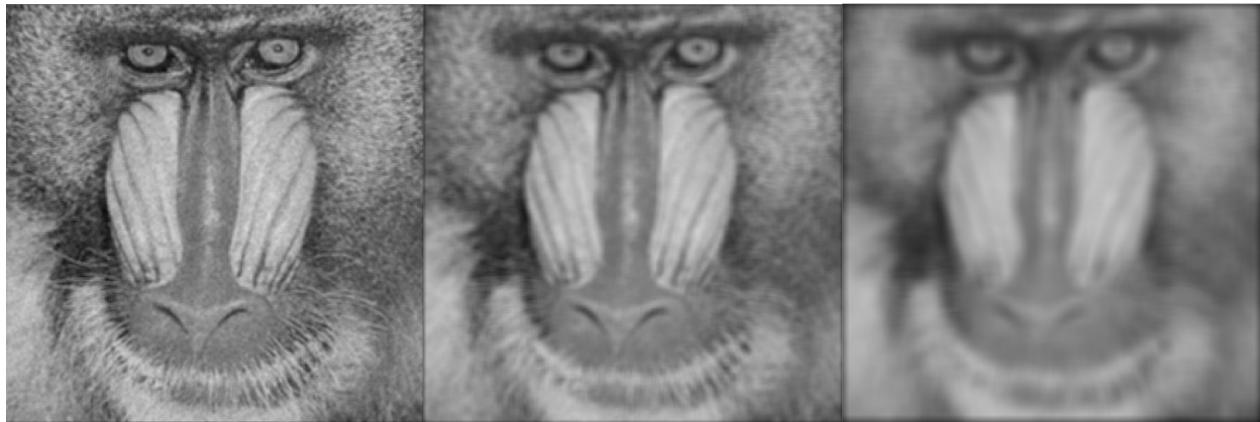


Figura 4 - Resultados de Aplicação de Filtros Gaussianos com Desvio 3, 5 e 10, Respetivamente.

3.1.2.2 Filtro Média

Por sua vez, o filtro média resulta numa imagem com mudanças bruscas de intensidade reduzidas, ou seja, reduz o ruído aleatório. No entanto, pode provocar artefactos de *ringing*.

No que toca ao algoritmo, a ideia passa por efetuar uma correlação em que todos os elementos da máscara possuem o mesmo peso e a sua soma é igual a 1, substituindo o píxel em análise pela média da vizinhança. Veja-se o Extrato 2, onde a definição desta máscara é visível, bem como a sua aplicação.

```
% Average
kernel = ones(filter_parameters.width_of_filter, filter_parameters.width_of_filter) / (
    filter_parameters.width_of_filter * filter_parameters.width_of_filter);
smoothed_image = imfilter(image_with_noise, kernel);
```

Extrato 2 - Código Matlab para Filtro Média.

Observe-se agora a Figura 5, onde a imagem mais à esquerda tem dimensão de 5 píxeis, a do meio de 10 e a da direita de 20. Assim, pode-se constatar que o aumento no tamanho da máscara leva a uma suavização bastante mais elevada. Porém, com o seu aumento, surgem também bordas negras na imagem devido à inclusão de uma quantidade superior de píxeis resultantes da aplicação do *zero padding*.

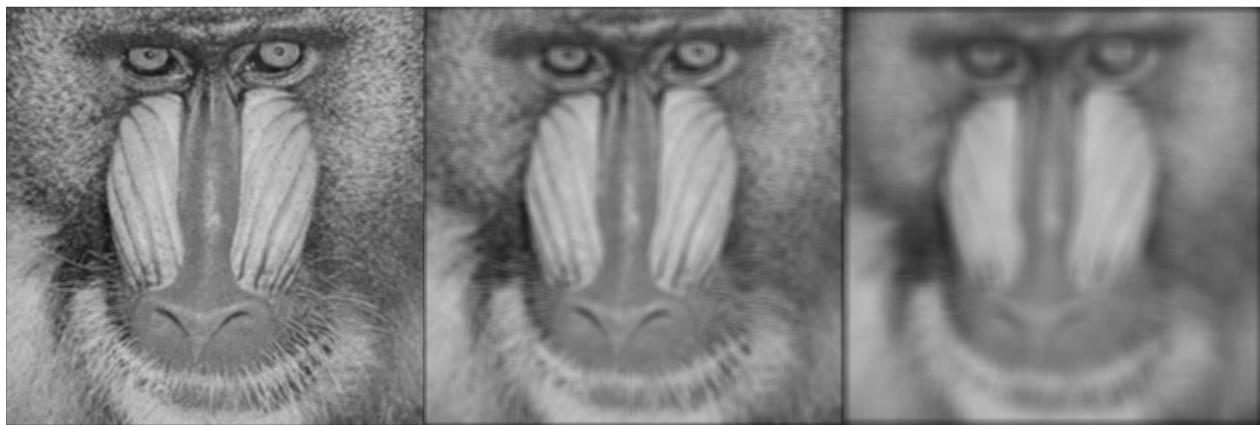


Figura 5 - Resultados de Aplicação de Filtros Média com Dimensão 5, 10 e 20, Respetivamente.

3.1.2.3 Filtro Mediana

Como mencionado anteriormente, este tipo de filtro é bastante diferente dos supra-mencionados. O algoritmo baseia-se em ordenar os píxeis dentro da área de vizinhança, por ordem crescente/-decrescente de intensidade. O píxel na posição central da máscara toma o valor da mediana da lista ordenada. Este é um filtro bastante utilizado para ruído de impulso, ou seja, não contínuo, pulsos irregulares de grande intensidade, e ruído *salt and pepper*, isto é, descontinuidades abruptas e

isoladas. É, ainda, caracterizado por não introduzir novos valores de intensidade de píxeis, remover picos de intensidade e preservar as bordas.

Como se pode ver pelo Extrato 3, a forma de implementar este tipo de filtro em *Matlab* é bastante diferente dos outros tipos de filtros até agora mencionados.

% Median

```
smoothed_image = medfilt2(image_with_noise, [filter_parameters.width_of_filter
filter_parameters.width_of_filter]);
```

Extrato 3 - Código *Matlab* para Filtro Mediana.

Tenha-se a Figura 6 como exemplo. Nesta, é possível observar que o aumento da dimensão da máscara leva a uma suavização maior da imagem, como seria de esperar, visto que a intensidade do píxel depende mais valores da sua vizinhança. Mais uma vez, verifica-se a geração de bordas negras na imagem, com o aumento da máscara.



Figura 6 - Resultados de Aplicação de Filtros Mediana com Dimensão 5, 10 e 20, Respetivamente.

3.1.3 Filtros no Domínio da Frequência

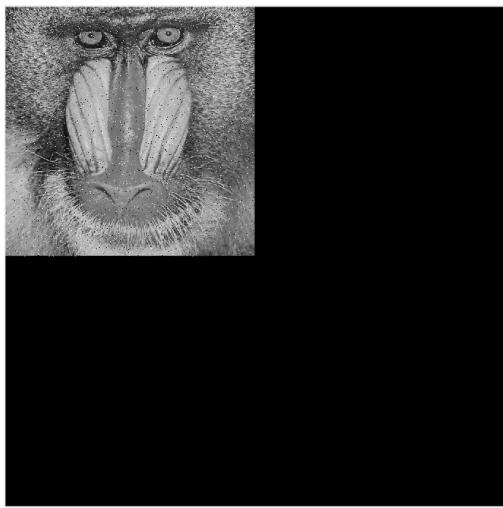
Este tipo de filtros atua sobre a Transformada de Fourier de uma imagem, dando uso ao facto de que uma multiplicação no domínio da frequência corresponde a uma convolução espacial.

Além disso, é mais intuitivo utilizar conceitos como filtros Passa-Baixo, Passa-Alto, entre outros, no que toca a frequência. Assim, é de realçar que componentes de alta frequência representam transições abruptas de intensidade e componentes de baixa frequência representam os componentes que variam de forma lenta. Dito isto, o ruído pode ser caracterizado por ser um componente de alta frequência.

Tenham-se em conta, de seguidas os passos para a aplicação destes filtros a uma imagem.

3.1.3.1 Padding da Imagem

O primeiro passo é efetuar o *padding* da imagem, ou seja, adiciona uma borda à imagem. Neste caso foi aplicado *zero padding*, ou seja, adicionar uma borda de píxeis pretos à imagem, como se pode ver na Figura 7. Note-se que a nova dimensão deve ser maior ou igual a $2M - 1$ por $2N - 1$, sendo M o número de linhas e N o número de colunas. Neste caso, foi definido que a nova dimensão da imagem seria $2M$ por $2N$.

Figura 7 - *Zero Padding* da Imagem.

3.1.3.2 Centrar a Imagem

O passo seguinte é efetuar a centralização, de modo a simpificar as operações com o filtro. Para isso, recorreu-se à Equação 1.

$$f_p(x, y) \times (-1)^{(x+y)} \quad (1)$$

Através da aplicação desta função obtém-se algo como a Figura 8.

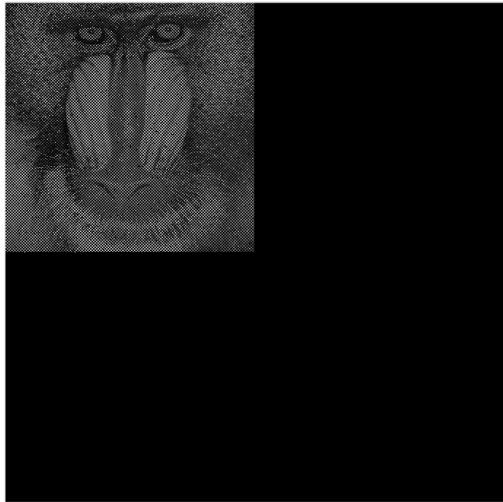


Figura 8 - Centralização da Imagem.

3.1.3.3 Transformada Discreta de Fourier

Efetuada a centralização, segue-se o cálculo da Transformada Discreta de Fourier, sendo que esta é caracterizada por possuir uma parte real e uma parte imaginária, bem como uma magnitude e um ângulo de fase. Esta operação é crucial, pois muda o domínio espacial para o domínio das frequências. Note-se que, graficamente, a Transformada não fornece muita informação, como se pode ver na Figura 9.

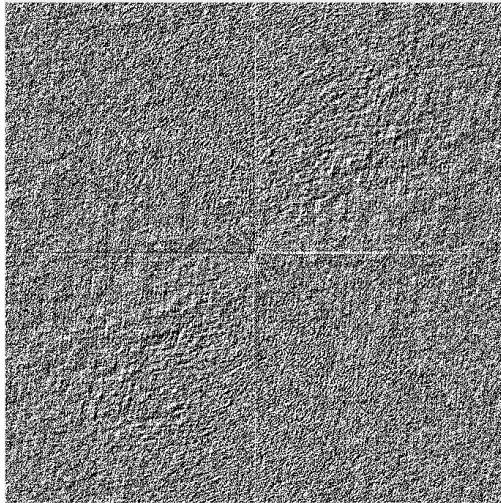


Figura 9 - Transformada Discreta de Fourier da Imagem.

3.1.3.4 Filtro

O filtro deve possuir dimensão igual à imagem após o *padding* (P, Q), estando centrado em $(\frac{P}{2}, \frac{Q}{2})$.

3.1.3.4.1 Filtro Gaussiano

O filtro Gaussiano é caracterizado pela Equação 2 para cada píxel, em que D_0 (frequência de corte) toma o valor do desvio padrão. Note-se que a Inversa da Transformada Discreta de Fourier é, também, uma gaussiana espacial, não possuindo efeito de *ringing*. É de realçar ainda que $D(u, v)$ é a norma da distância ao centro do filtro em cada ponto.

$$H(u, v) = e^{-\frac{D^2(u, v)}{2D_0^2}} \quad (2)$$

O filtro fica com um aspeto semelhante ao da Figura 10, dependendo, claro, dos seus parâmetros.



Figura 10 - Representação Gráfica do Filtro Gaussiano.

Este é um filtro bastante utilizado para o preenchimento de espaços em *broken characters* e na indústria de impressão e publicação. Além disso, como se pode ver pela Figura 11, o aumento do raio do filtro (desvio padrão) leva a uma menor atenuação, já que o cada vez menos componentes são removidos, resultando em menor suavização.

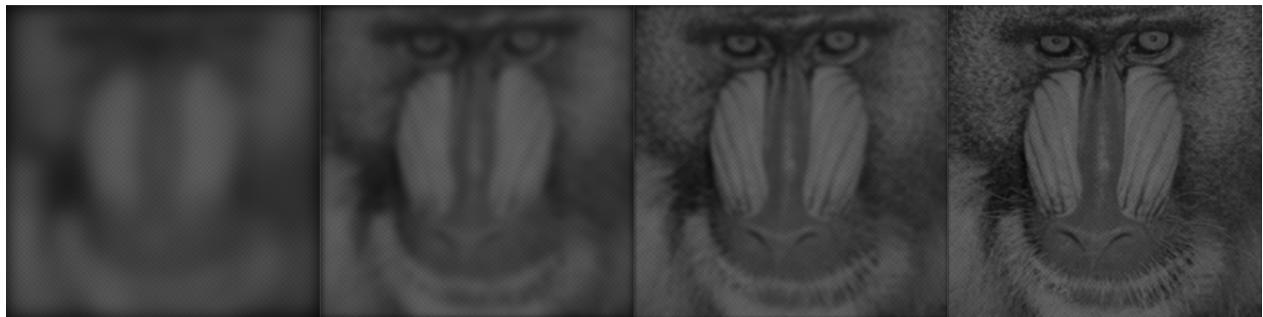


Figura 11 - Aplicação de Filtros Gaussianos com Desvio Padrão 1, 3, 7 e 10, Respetivamente.

3.1.3.4.2 Filtro Butterworth

Já no caso do filtro de Butterworth, a frequência de corte toma o valor do diâmetro do filtro, sendo que N representa a ordem do mesmo. Assim, tenha-se a Equação 3, que representa o valor para cada item da matriz.

$$H(u, v) = \frac{1}{1 + \left(\frac{D(u, v)}{D_0}\right)^{2N}} \quad (3)$$

Este tipo de filtro não possui uma descontinuidade acentuada entre a frequência cortada e a restante. No entanto, possui efeito de *ringing*, sendo que este aumenta com o incremento da ordem do filtro, como se vê na Figura 12. Nesta, as ordens são de 20, 100, 200 e 1000, respectivamente, para um diâmetro de 100 unidades.

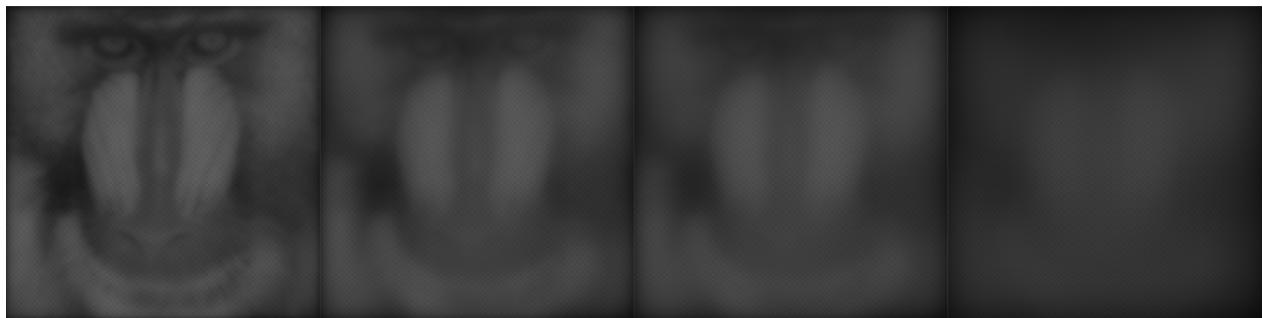


Figura 12 - Efeito da Ordem do Filtro em Artefactos de *Ringing*.

Por sua vez, o aumento do diâmetro, ou seja, o valor do raio do filtro, leva a uma menor atenuação, tal como no caso do filtro gaussiano aplicado no domínio das frequências, como se vê na Figura 13.

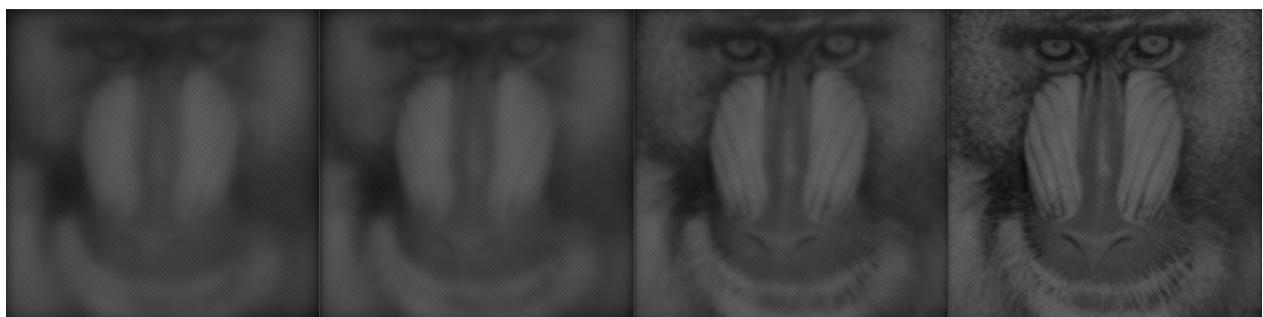


Figura 13 - Aumento do Diâmetro e Suavização em Filtros Butterworth.

3.1.3.5 Inversa da Transformada de Fourier

A aplicação da inversa muda o domínio onde estão a ser aplicadas as operações para o domínio espacial. O seu resultado é visível na Figura 14. Note-se que apenas a parte real é importante para o problema em questão.



Figura 14 - Inversa da Transformada de Fourier.

3.1.3.6 Descentralização da Imagem

De modo a reverter a centralização efetuada anteriormente, é necessário multiplicar todos os píxeis da matriz por $(-1)^{x+y}$, obtendo-se algo como a Figura 15.

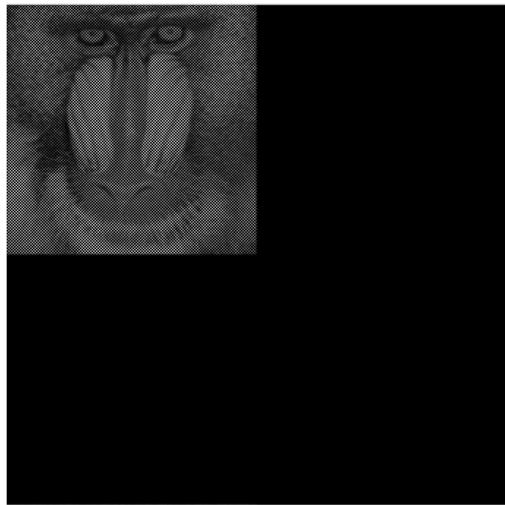


Figura 15 - Descentralização da Imagem.

3.1.3.7 *Crop* da Imagem

Por fim, falta extrair o conteúdo útil da imagem, visto que necessário aplicar um *padding* à mesma. Para isso, apenas é necessário observar o quadrante superior esquerdo da imagem, ficando-se com o resultado final. Assim, tenha-se a Figura 16.

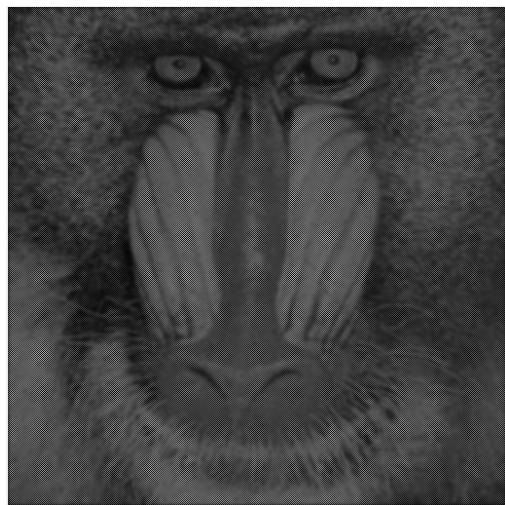


Figura 16 - Resultado da Aplicação de um Filtro no Domínio da Frequência.

3.1.4 Filtros Espaciais vs Filtros de Frequência

Como já mencionado, existem diversas diferenças entre os filtros no domínio das frequências e os filtros espaciais no que toca ao processamento de imagens. Os algoritmos espaciais manipulam diretamente os píxeis, enquanto que os de frequência trabalham sobre a transformada de Fourier da imagem. Além disso, o domínio espacial pode ser mais simples de compreender, devido ao tratamento dos próprios píxeis, apesar do conceito de filtros ser mais direcionado às frequências.

Dito isto, no que toca a gastos computacionais, o domínio espacial é menos pesado, sendo mais rápido a executar, não tendo que efetuar mudanças de domínio. Note-se que no caso de a imagem e a máscara serem ambas de dimensões elevadas, a filtragem no domínio das frequências revela ser mais rápida.

No que toca a resultados, ambos os métodos são viáveis para a suavização de imagens, como se pode ver pelos capítulos anteriores, mostrando resultados bastante semelhantes, dependendo dos parâmetros associados aos filtros.

3.2 Deteção de Arestas com o Método *Canny*

Este é o método mais usado no que toca à deteção de *edges* em imagens, em que se baseia na suposição de que as *step-edges* da imagem foram corrompidas por ruído gaussiano. O algoritmo começa por filtrar a imagem com um filtro gaussiano, sendo calculada, posteriormente, a magnitude e orientação do gradiente (aproximação finita). O passo seguinte baseia-se em aplicar *non-max suppression* à magnitude do gradiente e, por fim, aplicar um *threshold* duplo de histerese.

3.2.1 Suavização com um Filtro Gaussiano

Tendo por base o exercício anterior, de modo a suavizar a imagem, a ideia passou por efetuar uma filtração espacial à imagem, através de uma correlação. Observe-se a Figura 17. Note-se que, quanto maior a suavização, ou seja, o desvio padrão associado ao gaussiano, menor será o detalhe obtido no final.



Figura 17 - Suavização com um Filtro Gaussiano.

3.2.2 Gradiente

O cálculo do gradiente tem por base o operador de Sobel (Equação 4), em que é utilizado para convoluir a imagem por um filtro vertical e outro horizontal.

$$G_x = \begin{vmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{vmatrix} \quad G_y = \begin{vmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{vmatrix} \quad (4)$$

Obtidos os gradientes com as *edges* verticais ($\frac{\partial f}{\partial x}$) e horizontais ($\frac{\partial f}{\partial y}$), é possível calcular a orientação do gradiente. Para isso, é utilizada a função *atan2*, a qual recebe como parâmetros o

gradiente horizontal e vertical. Note-se que é necessário garantir que todas as orientações são positivas, pelo que, caso a orientação num ponto seja negativa, é necessário atribuir-lhe o valor de $360 + orientation(i, j)$. Já a magnitude é dada por $\sqrt{(G_x^2) + (G_y^2)}$, obtendo-se uma imagem semelhante à Figura 18.

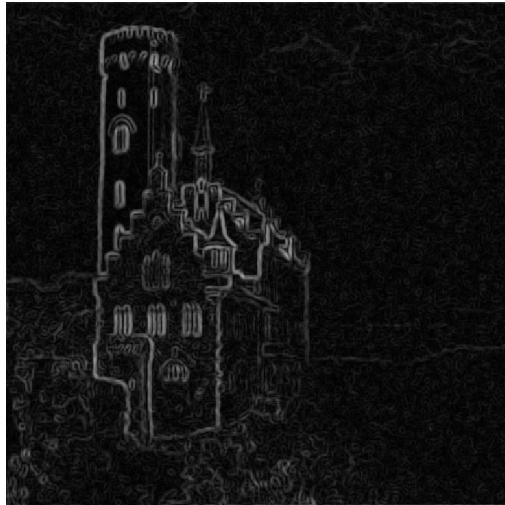


Figura 18 - Magnitude do Gradiente da Imagem.

3.2.3 Non-Max Suppression

Este passo consiste em estreitar as *edges* e refinar a localização, preservando todos os máximos locais no gradiente e apagando o resto. Dito isto, por cada píxel do gradiente é necessário saber a direção da *edge* e o número de orientações discretas para o vetor gradiente (por exemplo, Dimensão = 3 \Rightarrow 4 orientações (0, 45, 90 e 135 graus)). Assim, a ideia passa por comparar a intensidade do píxel em questão com os vizinhos na direção pretendida, em que o valor da sua intensidade se mantém caso seja maior que os vizinhos, ou passe a zero em caso contrário. Assim, tenha-se como exemplo a Figura 19.

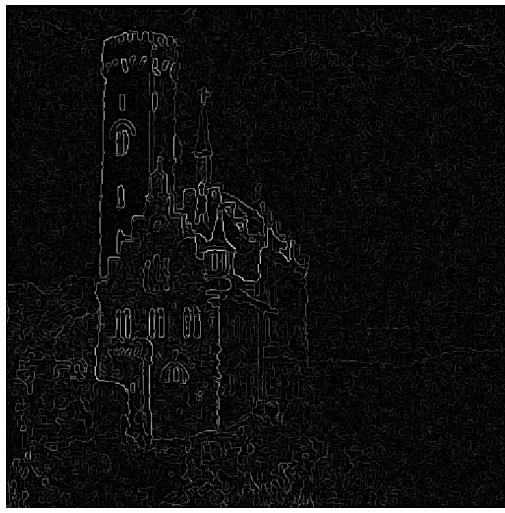
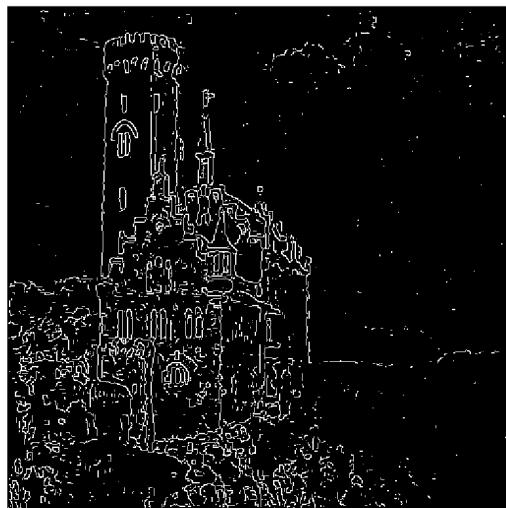


Figura 19 - Aplicação de *Non-Max Suppression* à Imagem.

3.2.4 Threshold Duplo

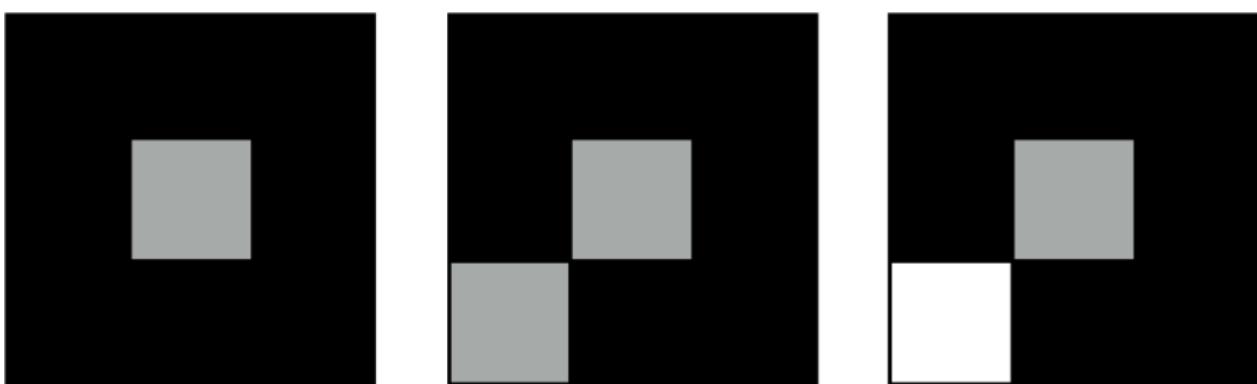
Note-se que a imagem resultante ainda contém máximos locais gerados por ruído, sendo que a solução para este problema passa por utilizar um *threshold* duplo. A ideia passa por atribuir o valor zero aos píxeis cuja intensidade é inferior ao *threshold* mínimo, 255 aos píxeis com intensidade superior ao *threshold* máximo e os restantes devem manter a sua intensidade, pois esta será necessária para o próximo passo, o *threshold* de histerese. O valor mínimo continua a permitir que os máximos do ruído sejam aceites, enquanto que o valor máximo pode fazer com que os máximos reais variem acima e abaixo do *threshold*, fragmentando a *edge*.

Como se pode ver pela Figura 20, existe uma grande diferença no resultado obtido em comparação com a secção anterior.

Figura 20 - Aplicação de um *Threshold* Duplo à Imagem.

3.2.5 *Threshold* de Histerese

Este passo tem por base a ideia de que o *threshold* alto começa uma cadeia, e o baixo é utilizado para encontrar *edges* fracas e continuar a cadeia. Por norma, uma *edge* fraca real está conectada a uma *edge* forte, enquanto que o ruído está desconectado. Dito isto, os píxeis da *edge* são divididos em *blobs* usando 8 píxeis vizinhos conectados. Os *blobs* que contém, pelo menos, um píxel forte (intensidade máxima após o *threshold* duplo) são preservados, sendo que os restantes são suprimidos, como se pode ver na Figura 21, onde apenas se preserva a intensidade do píxel contido no *blob* mais à direita.

Figura 21 - Conceito de *Threshold* de Histerese com *Blobs*.

Deste modo, observe-se agora a Figura 22, onde se pode observar o resultado da aplicação deste passo.

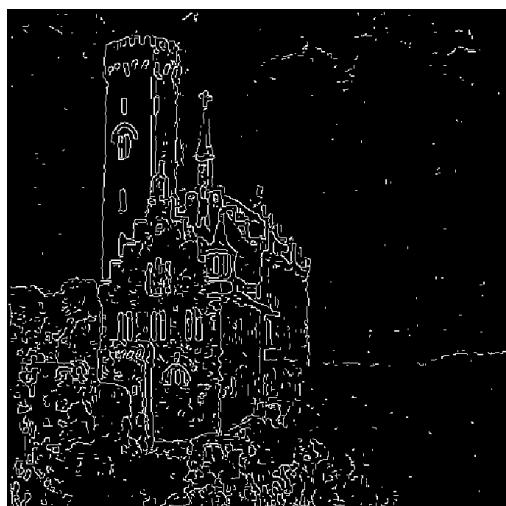


Figura 22 - Resultado Final da Aplicação do Método de Canny com Dimensão 5 e Desvio Padrão 3.

3.2.6 Influência da Suavização

É, ainda, merecedor de destaque o facto de que o valor de desvio associado à suavização tem uma influência grande no resultado final. Tendo em conta que a suavização implica uma perda de componentes de alta frequência, ou seja, *edges*, a escolha do valor do desvio tem um peso grande no resultado. Como mencionado anteriormente, quanto maior o desvio, maior a suavização. Além disso, um desvio pequeno implica a deteção de detalhes finos, ao contrário de um desvio alto que deteta *edges* de larga escala. Isto pode ser visto na Figura 23, onde o desvio começa por ser 1, passando a 3, 7 e, por fim, 10. Como se pode constatar, a figura mais à esquerda possui mais *edges*. No entanto, se o objetivo for a deteção do castelo, a imagem mais à direita é mais eficiente, não incluindo detalhes existentes no fundo da imagem.

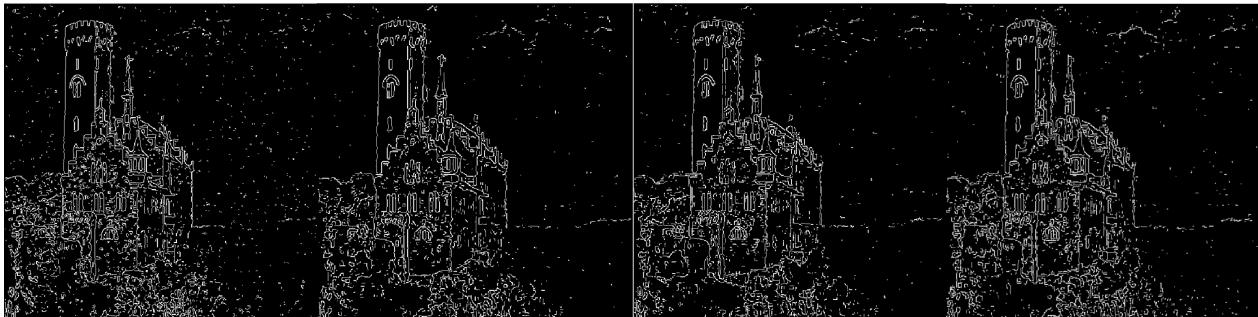


Figura 23 - Resultados Finais do Método de Canny com Desvios de 1, 3, 7 e 10, Respetivamente.

Tendo em atenção os resultados, a escolha do desvio padrão deve ter em conta o objetivo na deteção do detalhe da imagem, como já mencionado.

4 Conclusões e Trabalho Futuro

Ao longo do presente projeto encontra-se representado o primeiro tutorial da unidade curricular de Visão por Computador. Neste contexto foram abordados os diferentes tipos de filtros (nos domínios espacial e de frequência) que se podem aplicar a uma imagem, permitindo verificar o efeito que estes provocam na mesma, bem como as diferenças entre estes filtros. Foi, ainda, temática deste projeto, o método de deteção de arestas *Canny*, o qual se revelou de extrema importância, uma vez que a deteção de arestas de um objeto possibilita a resolução de inúmeras questões associadas ao processamento de imagem.

No que diz respeito aos filtros, o objetivo passou pela suavização de imagens através da redução de ruído na mesma. Para este efeito, recorreu-se a filtros no domínio espacial, que atuam diretamente nos píxeis da imagem, e a filtros no domínio de frequência, que atuam na Transformada Discreta de Fourier da imagem, onde posteriormente é computada a sua inversa, a fim de retomar a mudança do domínio de atuação para o espacial.

Os filtros no domínio espacial caracterizam-se por manipularem os valores de uma vizinhança com o intuito de modificar a imagem digital, usando para isso operações de convolução ou correlação com base em máscaras distintas, consoante o método a implementar.

Por sua vez os filtros no domínio da frequência revelam-se igualmente úteis, apesar de as imagens não possuirem a sua informação codificada neste domínio. No entanto, os conceitos de filtragem são mais intuitivos neste domínio, pelo que as frequências altas correspondem a transições mais abruptas e mudanças de intensidade mais rápidas na imagem. Contrariamente, as frequências baixas dizem respeito à variação lenta da intensidade dos componentes de uma imagem.

Em suma, a realização deste trabalho exigiu a aplicação de todos os conhecimentos lecionados em contexto de aula, permitindo que o grupo cumprisse todos os objetivos propostos no enunciado, conciliando assim a teoria e a prática.