André Pires Alves

Licenciado em Engenharia Informática

Best Photo Selection

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática

Orientador:	Fernando P. Birra, Prof. Auxiliar, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa
Co-orientador:	João M. Lourenço, Prof. Auxiliar, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa

Júri

Presidente:	Prof. Artur Miguel Dias, FCT UNL
Arguente:	Prof. Rui Nóbrega, FEUP
Vogal:	Prof. Fernando Birra, FCT UNL

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

December, 2015

Best Photo Selection

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template “unlthesis” [1] desenvolvido no Dep. Informática da FCT-NOVA [2]. [1] https://github.com/joaomlourenco/unlthesis [2] http://www.di.fct.unl.pt

Aos meus pais e avô.

Acknowledgements

First and foremost, I would like to express my gratitude to my advisers, Prof. Fernando Birra and Prof. João Lourenço, for the opportunity of letting me work and explore this project and also for their patience, support and guidance through its course.

I would also like to express my appreciation to Departamento de Informática da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa (DI - FCT/UNL) for being like a second home during my academic journey and also for allowing me to become a better person, not only professionally but also as a human being.

To my family, particularly my parents and grandfather, for always being there for me. Their endless encouragement, support and love gave me the strength to go through this hard path.

Last, but not least, to my friends with who I shared the same successes and adversities, day in and day out, through these years.

Abstract

The rising of digital photography underlies a clear change on the paradigm of the photography management process by amateur photographers. Nowadays, taking one more photo comes for free, thus it is usual for amateurs to take several photos of the same subject hoping that one of them will match the quality standards of the photographer, namely in terms of illumination, focus and framing. Assuming that the framing issue is easily solved by cropping the photo, there is still the need to select which of the well framed photos, technically similar in terms of illumination and focus, are going to be kept (and in opposition which photos are going to be discarded). The process of visual observation, on a computer screen, in order to select the best photo is inaccurate and thus becomes a generator of insecurity feelings that may lead to no discarding of photos at all. In this work, we propose to address the issue of how to help the amateur photographer to select the best photo from a set of similar photos by analysing them in technical terms. The result is a novel workflow supported by a software package, guided by user input, which will allow the sorting of the similar photos accordingly to their technical characteristics (illumination and focus) and the user requirements. As a result, we expect that the process of choosing the best photo, and discarding of the remaining, becomes reliable and more comfortable.

Keywords: Signal Processing, Image Processing, Decisions Support Systems, Digital Photography

Resumo

O aparecimento da fotografia digital está na base de uma clara mudança de paradigma no processo de gestão da fotografia por amadores. Porque tirar mais uma fotografia agora não representa qualquer custo adicional, é habitual tirarem-se múltiplas fotografias ao mesmo sujeito, na expectativa de que uma delas corresponda aos padrões de qualidade desejados, em termos de iluminação, foco e enquadramento. Assumindo que a questão do enquadramento se resolve facilmente recorrendo ao recorte (*crop*) da fotografia, tem-se ainda assim que selecionar qual das várias fotografias bem enquadradas, tecnicamente parecidas em termos de iluminação e foco, vamos guardar (e por oposição quais vamos descartar). A escolha da melhor fotografia com base na observação visual em ecrã de computador é um processo muito pouco preciso e, portanto, gerador de sensações de insegurança que resultam, muitas vezes, na opção de não descartar nenhuma das várias fotografias semelhantes. Neste trabalho propomo-nos endereçar a questão de como ajudar um fotógrafo amador a selecionar a melhor fotografia, de um conjunto, analisando-as em termos técnicos. Propomos assim uma solução inovadora para este problema específico, baseada num processo (*workflow*) suportado por um pacote de *software*, que com alguma ajuda do utilizador permite ordenar um conjunto de fotografias semelhantes de acordo com as suas características técnicas (iluminação e foco), permitindo assim escolher aquela que melhor corresponde às expectativas e dando segurança e conforto na eliminação das restantes.

Palavras-chave: Processamento de Sinal, Processamento de Imagem, Sistema de Apoio à Decisão, Fotografia Digital

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

1.1 Context & Motivation

Photography can be defined as the art and science of recording images by means of capturing light on a light-sensitive medium [Per08]. The earliest known permanent photography, recognized as the *View from the Window at Le Gras* (c. 1826), was taken by Nicéphore Niépce (1765-1833) having an approximate 8 hour exposure [Ros97].

After the Niépce's discovery, photography has evolved greatly culminating into digital photography, created by digital cameras, which awards photographers with novel ways to explore and master new light capturing techniques.

The ever growing research in fields such as computer vision and image processing promote the union between the world of computer science and the world of photography, leading to numerous image editing techniques and software products that aim at providing assistance to the photographer's task of manipulating their images.

It was estimated that in the year of 2012 alone, over 15 million digital single lens reflex (DSLR) cameras were sold [Gri]. Also, by 2010 Adobe predicted that there were over 10 million Photoshop users worldwide [Ado]. Such facts give the ideal motivation for the creation or reinvention of workflows that explore the world of photography and thus simplify the photographer's life.

1.2 Problem Description & Objectives

Light is a vital component within the art of photography since it makes the image registration by the camera sensor possible. Registering the light of a scene with the sensor of a DSLR camera may lead to very different results, varying on factors such as sharp focused regions, blur, noise or colour contrast. Digital photography unlike its predecessor, the film photography, may be stored in reusable memory cards with advantages, for instance, in cost terms. Consequently, to increase the probability of achieving the desired result it is a common practice among amateur photographers to take several shots of the same subject, in automatic mode, hopping that one of them will match their quality requirements. Good lighting, clear center of interest, no shaking, absence of noise or a good composition can be among the desired requirements. Combining such requirements may be hard to achieve in a single photo as a series of unsought effects may occur due to a wrong use, or misinterpretation, of the camera settings.

1.2.1 Improper Focus

While taking photos, the photographer may choose to capture an image where all the details are acceptably focused, for instance a landscape photo, or may opt to focus a specific subject leaving it sharply focused with the remainder of the image blurred.

However, the process of focusing a specific subject may be not always successful. An improper focus may occur due to photographer's mistakes such as bad focus points adjustment, leaving the focus points away from the subject, or a wrong definition of the range of distance (depth of field) that is in sharp focus. On the other hand, the photographer may be trying to capture an image where the subject is slightly moving, like a flower in a windy day or a restless child making it difficult to set the points of focus on the desired area.

The focused area is the one that attracts the most attention from the viewer, a photo in which the center of interest is not sharply focused becomes undesirable. Therefore, the focusing process may be considered one of the most relevant at the image capturing moment.

1.2.2 Camera & Motion Blur

When photographers capture images focusing on a specific subject, the remaining details appear blurred thereby emphasizing the main subject. Nevertheless, blur can also occur in other situations that bring a negative impact to the photo.

The first case, and the most critical one, happens when the camera moves during the image capturing time frame. The phenomenon of camera blur is due to simple factors such as trembling of hands or breathing of the photographer or even the simple pressing of the camera button, being specially sensitive for longer exposure times. As a result, the photo contains an image that shows general blur, with no clear center of interest, being certainly short from the expected result.

The second case occurs when there is movement in the scene. The photographer may leverage on this and take a photo that gives the impression of existing movement, either in the foreground or the background, depending on whether the camera is frozen or moving along the direction of motion. For instance, a moving train or a giant wheel in an amusement park are good subjects to a motion blur image. Motion blur effects require some expertise and technical skills from the photographer, and are very hard to achieve by the common amateur photographer. A wrong definition of the exposure time can result in a photo where the subject of motion is sharp when it should be blurred or vice-versa, once again producing results different from the expected.

1.2.3 Noise

Digital noise can be defined as a set of pixels whose colour and brightness are unrelated to the subject, spread randomly along the photo, which degrade the image quality. It is generally seen as alienated dots scattered through the photo.

Typically, the digital noise appearance is an image sensor size consequence. The largest the image sensor, the lesser digital noise. Larger sensors have a higher signal-to-noise ratio, resulting in photos with less noise [San13]. However even cameras with large image sensors, such as DSLR cameras, are susceptible to capture noisy images. For instance, a low light room requires a higher sensitivity from the camera sensor to capture the existing light quickly. Cameras increase the sensitivity by amplifying the signal of the image sensor. The higher the sensitivity increases, the higher the noise in the captured image, as increasing the sensitivity amplifies the captured signal, but also amplifies the background noise captured along with it [Cur04].

Therefore, if situations such as the aforementioned happen, the outcome will be a noisy image wherein, usually a clear photo is preferable over a grainy one.

1.2.4 Uneven Exposure

Colour plays an integral part not only in the visual perception but also in the emotions that a photo creates. Digital cameras have the great virtue of offering the opportunity for photographers to create their own vision of the world around them with the ability to adjust or even to change colours [Rut06]. Hence, knowing how to properly use colour becomes one of the photographer's main challenges.

Lightness refers to the amplitude of colour or its proximity to the white or black end of the tonal scale [Cur04]. To get the ideal photo, the right amount of light must be let in through the camera sensor. Such is accomplished by defining the camera settings correctly for each scene. However, small variations of the camera settings have large effects on the resulting image. If the quantity of light registered by the camera sensor is too low then the resulting photo may appear dark (underexposed) where the details are lost in the shadows. On the other hand, if the light quantity is higher than it should be, the result may be a too bright (overexposed) photo where the details are lost in the highlights.

Usually photographers aim at taking photos that have a balanced lightness of colour, in opposition to underexposed or overexposed photos.

1.2.5 Unrealistic Colour Cast

Although the human brain has the ability to perceive the white colour in scenes under different types of illumination, digital cameras tend to capture the light colour as it is. Digital cameras offer the *White Balance* feature that may erase or attenuate unrealistic colour casts. However, an inadequate white balance setting may lead to an undesired photo. For instance, the warm colours in the end of a summer day may mislead the automatic white balance feature of a digital camera and consequently a bluish and insipid image is captured [San13].

Thus, a photo whose colours are well balanced, creating an appealing contrast is, most of the times, more captivating than one that is too *cold* or too *warm*.

1.2.6 Best Photo Selection

After taking the photos, the photographer transfers them to the computer so that he can choose which one best suits his taste. This work addresses precisely the process of choosing the best photo from a series of similar photos. Most DSLR cameras capture images whose resolution is higher than a regular computer screen resolution. For instance, a *Canon 7D Mark II* has a resolution of 5472x3648 pixels [Incb], which is nearly 20 megapixels, while a regular computer screen has a resolution of 1 megapixel and thus the photographer can, at most, see 5% of the image at a time. Zooming out, i.e. image shrinking, implies geometrical transformations where original pixels are lost. A reduction by one-half means that every other row and column is lost creating a new version of the image with a quarter of the original version pixels [GW01]. Therefore image shrinking is not always a good option since it changes some of the image's primitive features. So, if the photographer wants to make a detailed analysis he must work on the original resolution.

Hereupon when the photographer wants to compare the set of similar images he may, for instance, open various windows, each containing an image, and analyse them side by side. However he can only analyse a certain area of an image which can lead to unpleasant situations such as the decontextualisation of the photographer in the scope of the image. Also, given that the number of photos taken can vary from a couple to many, the process of choosing the best one may create the feeling of insecurity and frustration, in addition to being time consuming, tedious and error prone, including having to switch forwards and backwards on the candidate photos.

There are several software products, such as Adobe Photoshop or Adobe Lightroom, which are heavily used by photographers to manipulate their images. However, none of these software products really aid the process of choosing the best one from a set of similar photos. Our goal is to fill this gap by creating a workflow that receives the set of similar photos returning an analysis based on specific parameters for each photo. Moreover, combining this analysis with some user specifications may lead to a reduction of the initial sample of photos to a smaller one, sorted by their technical characteristics, for the purpose of simplifying the process of choosing the best photo.

1.3 Research Question & Approach

Given a set of similar photos, it is possible to help the photographer in the process of sorting those photos according to their technical characteristics and in the detection of the weakest ones. To this purpose, we propose a phased workflow and a support software package, where the phases are prioritized by their technical relevance and in each phase the elimination of the technically less accurate photos is proposed.

The solution is defined by a workflow, supported by a software package, which aims at helping the user in the best photo selection process. To do so, an analysis over the aforementioned problems (improper focus, camera and motion blur, noise and colour) is made in separate phases. In each phase of the workflow, the user can reject the photos that do not match his/her quality standards, hopefully leading to a significantly smaller set of photos when compared to the original one.

The presented solution consists in six fundamental procedures:

1. The first procedure involves the detection and matching of image's characteristic points, which will allow the correlation of the set of similar photos in order to match scenes among photos, enabling their comparative analysis on the following phases.

2. The second procedure addresses the detection of the image's focused areas, which may be local or global. This procedure will allow the comparison of interest regions in terms of focus.

3. The following procedure is the detection of motion blur. As referred in Section 1.2, the blur may be due to camera movement or due to subject movement in the scene. Both types of blur can be analysed by the same procedure as the camera blur has the same characteristics of motion blur, apart from the fact that camera blur covers all the photo surface and motion blur does not.

4. The forth procedure is an estimation of the level of noise in each image. The set of similar photos serves as input wherein the output will be an evaluation of the level of noise existing in each image. Thereafter, it will be possible to sort the set of the remaining similar photos in terms of noise.

5. The fifth procedure analyses the exposure level of each photo. An underexposed image is characterized by having dark colour tones while an overexposed image is mainly composed by bright colour tones. In principle, photographers aim at photos covering the full tonal spectrum and again it is possible to sort the candidate photos according to this criteria.

6. The sixth, and last procedure, analyses the colour temperature on images aiming at differentiating those that are well colour balanced from those considered *cold* or *warm*.

To achieve the aforementioned solution, a set of computer vision algorithms provided by the Open Source Computer Vision (OpenCV) [Its] library were evaluated and applied, using the C++ programming language.

1.4 Contributions

The main contributions of this work are:

- *A photo selection workflow*: The proposed workflow leads to what we believe to be a unique combination of a set of image processing techniques, thus yielding a novel and useful software product with the sole purpose of helping the user to choose the best photo from a set of similar photos.
- *Decision support system over a set of similar photos*: Naturally, the art of photography is highly subjective. The best photo for one person may not be the best photo for someone else. Thus, by using the software product described in this work, we hope that the user quality standards are **indeed** taken into account, giving safety and comfort to the process of choosing the best photo from a set of similar photos.

1.5 Publications

An article describing a preliminary version of the proposed approach [Alv+15] was published and presented at the INForum Symposium 2015. The final version of this article can be found in: `http://docentes.fct.unl.pt/sites/default/files/joao-lourenco/files/inforum15-photo.pdf`.

1.6 Document Organization

This document is structured in four chapters: Introduction, Background & Related Work, System Description & Features and Conclusions. The first chapter, *Introduction*, contextualizes and introduces the reader to the problem at hand and our proposed approach to address it. The second chapter, *Background & Related Work*, presents the fundamental concepts of photography in order to understand the origins of the problems stated in Section 1.2. This is followed by a few sections of background study on how to address each problem from a technical point of view and a section describing methodologies that are partly similar to the one present in this document. The third chapter, *System Description & Features*, addresses the fundamental concepts of the implementation, namely the workflow, the evaluation and the general structure. At last, in *Conclusions* is held a self-criticism by commenting the final results of our work as well as possible future improvements.

Chapter 2

Background & Related Work

This chapter presents some background concepts as well as an aggregate of techniques and algorithms that are relevant in the context of this work.

Section 2.1 presents some fundamental concepts of photography and their influence in the process of capturing an image. In Section 2.2, image correlation algorithms are approached, giving a brief description of each of the studied algorithms. These algorithms aim at matching scenes among different images. Then, Section 2.3 presents some related work on focus detection in an image. Following comes Section 2.4, which introduces methods for motion blur detection. In Section 2.5, previous work on noise detection is addressed. Section 2.6 presents some background study aiming at analysing the colour information of an image. Section 2.7 presents two related works that combine similar algorithms to those presented from Section 2.3 to Section 2.6 in order to select the best photos from an album. In Section 2.8 are presented a few recognized image processing applications that analyse images from a technical point of view. Finally, in Section 2.9 OpenCV is introduced as a specialized image processing library.

2.1 Fundamental Concepts of Photography

The word "photograph" was formed by the concatenation of two ancient Greek words: *photos*, which means "light", and *graphos* that meant "drawing, painting, writing". So, in ancient Greek, the word "photograph" means "drawing with light" [Per08]. Thus, it is fair to say that photography is all about light. In this section some of the most important concepts about photography are explained, along with their influence in the final result.

2.1.1 Light

The same way as writing on a paper requires ink, photography requires light [San13]. Light has characteristic features that have great influence in photography: brightness and colour. Brightness of light determines, for instance, the quantity of available light and thus it may influence the exposure variables (see Section 2.1.2). In turn, colour of light influences the way a photography can be interpreted by the human eye being that its spectrum is much wider than what the human eye can recognise.

In order to take a photograph, one or more light sources must be present. Such sources can be divided into natural and artificial [Fre08; San13]. Sun emits natural light that may be captured in different ways depending on factors such as the station of the year, weather condition, time of day and location. Artificial light is emitted by multiple types of illumination like incandescent lamps, fluorescent lamps or even the camera flash.

Digital cameras bring an unprecedented exactness to measuring light by recording the light falling on the camera sensor as an electrical charge, in proportion to the light intensity [Fre08]. This unprecedented exactness happens due to the freedom offered to the photographer to adjust the light capturing mode by combining different camera settings such as the aperture, shutter speed and ISO (see Sections 2.1.4, 2.1.6 and 2.1.7, respectively). Consequently, the different combinations of the camera settings result in different exposures that may vary on factors such as focus, blur, noise or colour balance.

2.1.2 Exposure

Exposure is the amount of light that reaches the camera sensor and it depends on three variables. Pairing the aperture (see Section 2.1.4) with the shutter speed (see Section 2.1.6) will let in the right amount of light and therefore it will create the desired exposure [San13]. Another important variable is ISO (see Section 2.1.7), as it defines the sensibility of camera sensor towards the existing light.

2.1.3 Focus Points

Focus is a key feature in photography as it indicates which parts of the image are sharpest [Cur04]. Naturally, those sharp areas are the ones that draw the most attention from the viewer. It is up to the photographer to decide whether to have a large set of focus points, leading to a globally focused photo or to focus a smaller region in order to emphasize a specific subject in the image. Most photographs have a specific location in the scene which represents their subject. That subject must be where the focus point(s) lie. The focus points of an image are an essential concept in photography for they can change the way an image is perceived by the viewer. For instance, in Figure 2.1 two images targeting the same subject but differing on the focus points are shown, wherein the final result is remarkably different.

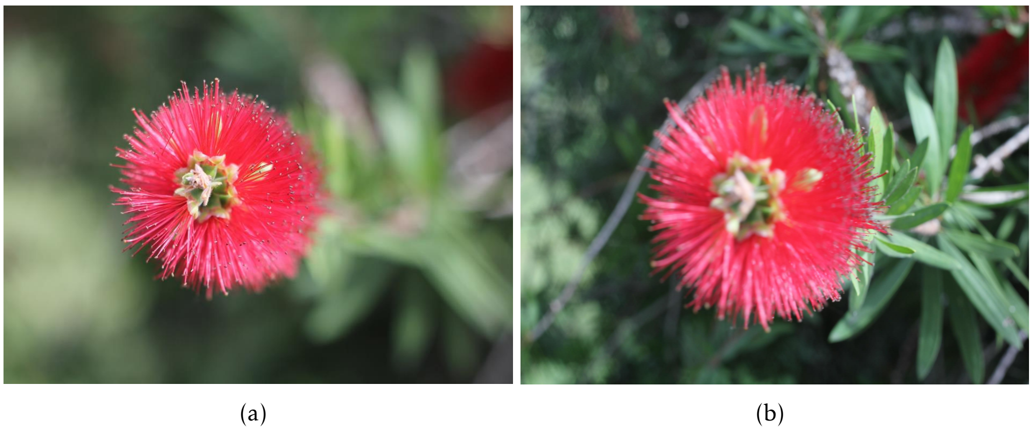


Figure 2.1: Point of focus on the flower petals (a) and point of focus on the flower stalk (b).

2.1.4 Aperture

The aperture is the hole through which light enters the camera [San13]. The wider the aperture the more light is let in. It is measured in f-stops, which is a ratio between the camera focal length and the diameter of the aperture. The f-stop value is conversely opposite to the aperture value, given that the wider the aperture the smaller is the f-stop. Large f-stops are preferable when taking photos where a large depth of field is required such as landscapes. Small f-stops are more suited to take photos where a small depth of field is needed, such as portraits.

2.1.5 Depth of Field

The photographer may control which areas of the image will be focused. Depth of field represents a range of distances at which the objects present in the scene will appear acceptably sharp in the resulting photo [San13]. It depends mainly on the aperture (see Section 2.1.4) and focus length of the camera. Wider apertures (small f-stops) and closer focusing distances origin a shallower depth of field resulting in a local focus effect. Smaller apertures (larger f-stops) and increased focusing distances result in a deeper depth of field originating a global focus effect, as shown in Figure 2.2.

Regions of objects within the depth of field become less sharp the farther they are from the plane of critical focus [Cur04], as shown in the left side of Figure 2.2. The transition from the sharp region to the unsharp region does not create an abrupt change. The pixels outside the depth of field are not all equally blurred. This phenomenon is commonly referred to as the circle of confusion.

2.1.6 Shutter Speed

Also known as exposure time, shutter speed refers to the amount of time the shutter is opened while the exposure happens [San13], i.e., the amount of time required for the

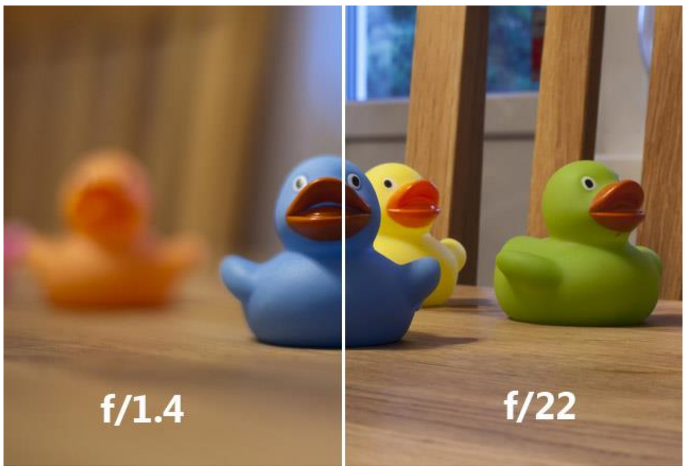


Figure 2.2: Local focus (left side) and global focus (right side)[1].

camera sensor to capture the image.

Faster shutter speeds imply smaller exposure times and thus they usually freeze the action in the image. In opposition, slower shutter speeds mean larger exposure times and consequently more light is captured by the camera sensor. A slow shutter speed can create the effect of motion blur that happens when any object moving along the direction of relative motion appears blurry, as it it shown in Figure 2.3.

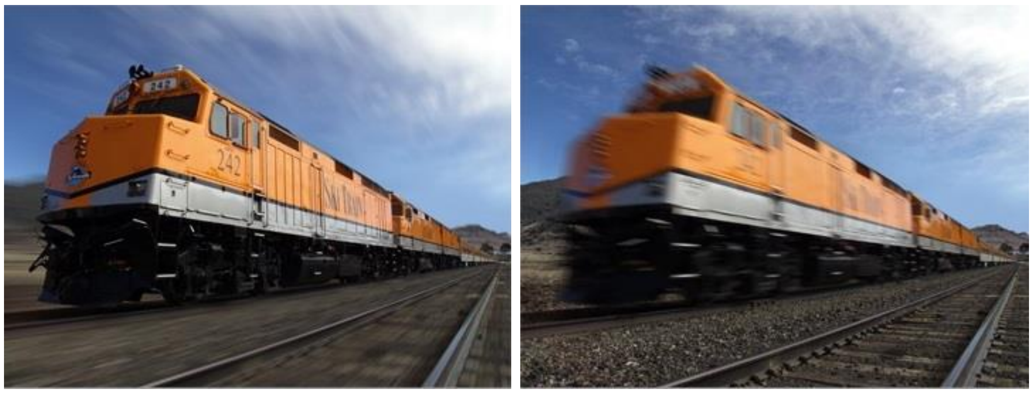


Figure 2.3: Slow shutter speed combined with panning the camera(a); Slow shutter speed but no panning of the camera(b)[2].

A negative effect of using a slow shutter speed is to capture an image where most details are blurred due to camera shake. The slightest move during the exposure, such as trembling of hands or breathing of the photographer or even the simple press of the camera shutter button, may result in a blurry image, even if the subject stood still. Figure 2.4

[1]Source: http://cameralensadvice.com/dslr-basics/
[2]Source: http://www.mblpro.com/overview.htm

(a) shows the result of an image captured when there was camera motion while (b) shows the expected result.

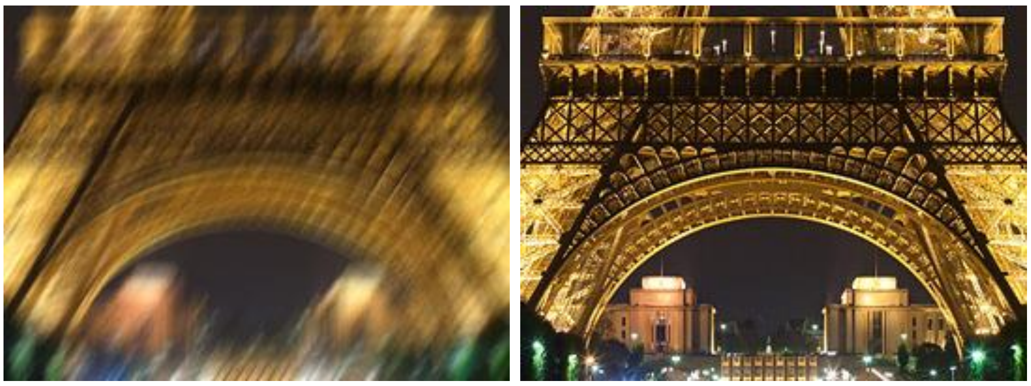

(a) (b)

Figure 2.4: Blurry image due to camera shake(a); expected result (b)[3].

2.1.7 ISO

ISO refers to the level of sensitivity of the camera sensor towards the light by using a numerical scale. The more sensitivity the less time is required to capture the image although sometimes at the expense of grainier images. A lower ISO requires more light in order to create a good exposure. This can be achieved by using a wider aperture or a slower shutter speed. High ISO is preferable when the existing light is scarce. As counterpart, a high ISO will capture a noisy image. The noise influence is shown in Figure 2.5.

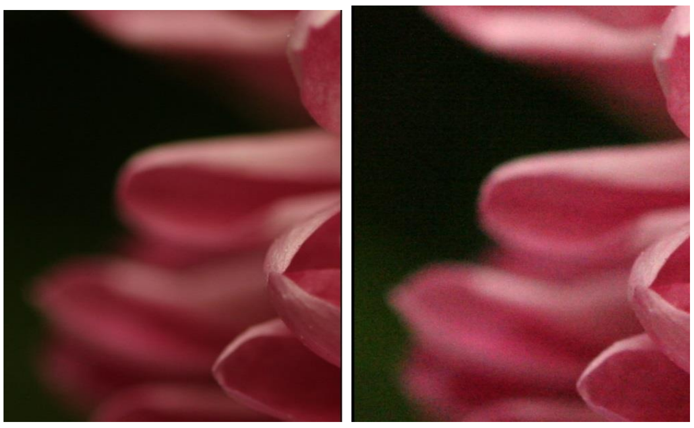

(a) (b)

Figure 2.5: Difference between an image with ISO value of 100 (a) and 3200 (b)[4].

[3]Source: http://www.cambridgeincolour.com/tutorials/camera-shake.htm
[4]Source: http://digital-photography-school.com/iso-settings/

Noise is originated because cameras increase the sensitivity by amplifying the signal of the image sensor. The higher the sensitivity increases, the higher the noise in the captured image, as increasing the sensitivity amplifies the captured signal, but also amplifies the background noise captured along with it [Cur04].

2.1.8 Level of Exposure

The concepts introduced in the sections above, namely the aperture, the shutter speed and the ISO, define the level of exposure in a photo. These three variables are interdependent given that, to keep the same level of exposure, when one of them changes, at least one of the remainder must change too [San13]. The combination of these variables may lead to different types of exposure such as shown in Figure 2.6, where the difference between an underexposed, a well exposed and an overexposed image is displayed. An underexposed photo is characterized by dark colour tones, an overexposed photo is dominated by bright colour tones while a well exposed photo has balanced lightness in its colours.

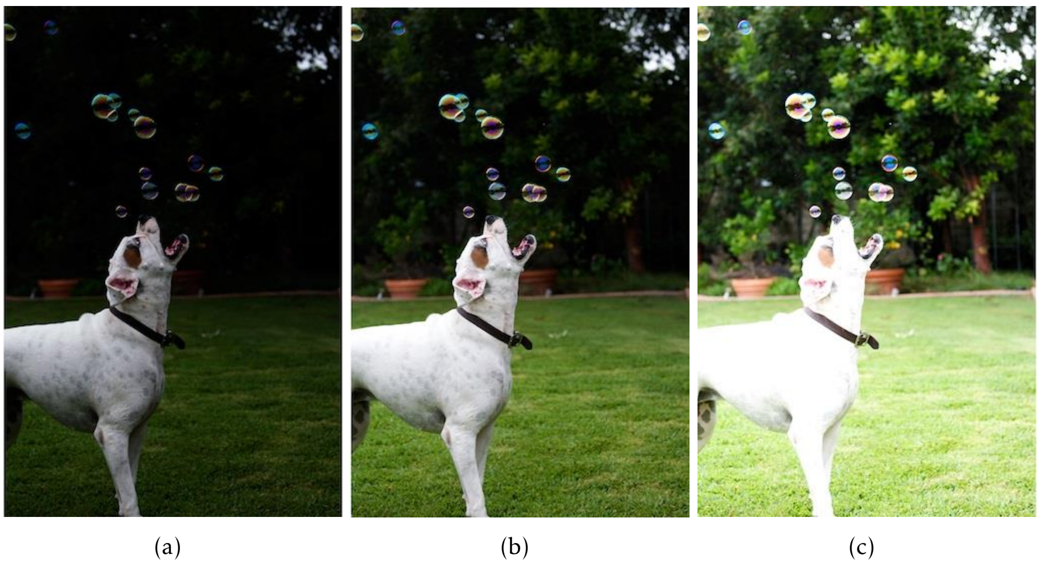


Figure 2.6: Difference between underexposed image (a), the ideal exposure (b) and an overexposed image (c)[5].

2.1.9 White Balance

Looking at the colour spectrum and temperature, it is clear that the colour of light can vary significantly [Fre08]. The human brain can interpret the variance in white light produced by different sources of light making it look *normal*. However a camera sensor captures the light as it is. Thus it is necessary to use white balance in order to match the captured ambient light the way our brain would read it. Figure 2.7 shows an example of an image taken with different tonalities.

[5]Source: http://whiteonricecouple.com/photography-tips/what-is-photography-exposure/

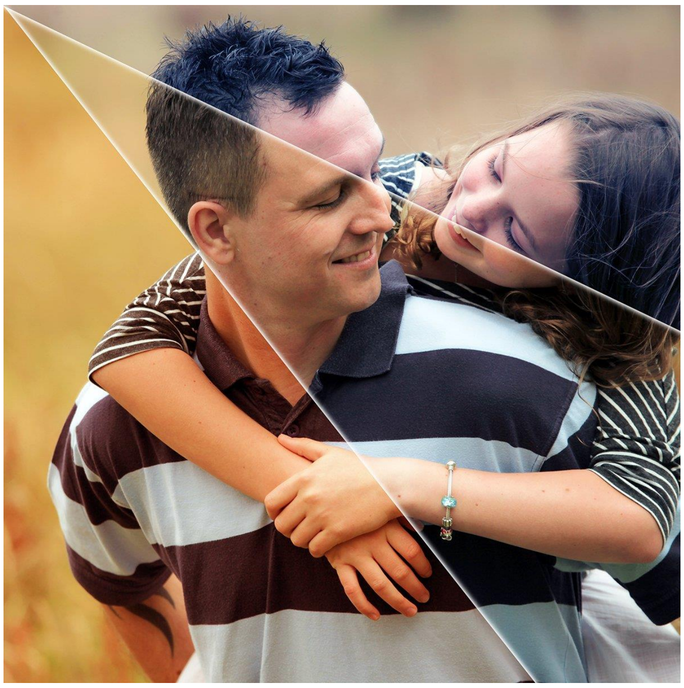


Figure 2.7: Top portion is *cold*. Bottom section is *warm*. Middle section shows a correct white balance setting for the image[6].

2.2 Image Correlation Algorithms

Image correlation refers to the process of analysing data sets from different images and to match them in order to enable their comparative analysis. This section is divided in two parts: the first (see Section 2.2.1) approaches the detection of keypoints in images, which are identifiable and peculiar points in the image scope; the second (see Section 2.2.2) refers to how those keypoints, from different images, can be correlated.

2.2.1 Keypoint Detector

Keypoints provide a large amount of information about an image. An important advantage of keypoints is that they permit matching even in the presence of clutter (occlusion) and scale or orientation changes. There are multiple feature detector algorithms. In this section the ones considered to provide best results are described: FAST [RD06], SIFT [Low04] and SURF [Bay+08].

2.2.1.1 FAST

Features from Accelerated Segment Test (FAST) [RD06] is a corner detector algorithm best suited to process image features in real-time frame-rate applications due to its performance driven implementation. To detect a potential corner an analysis for each pixel

[6]Source: http://www.danoah.com/2012/09/learning-white-balance-the-easy-way.html

p in the image is done. Pixel p is a corner if there is a set of n contiguous pixels in its neighbourhood whose intensities are higher than $I(p) + t$ or smaller than $I(p) - t$, where $I(p)$ represents the intensity value of pixel p and t defines an appropriate threshold value. Synthesizing, a potential corner is detected if its intensity is distinguishable among the intensities of n points in its neighbourhood. Although FAST is faster than both SIFT (see Section 2.2.1.2) and SURF (see Section 2.2.1.3), it may not perform as good as desirable when images are subjected to variations on scale and illumination.

2.2.1.2 SIFT

Scale-Invariant Feature Transform (SIFT) [Low04] allows the extraction of features that are invariant to image scale and rotation and partially invariant to a substantial range of affine distortions, changes in the 3D viewport, addition of noise and changes in illumination. It consists of four elemental procedures: keypoint detection, keypoint localization, orientation assignment and keypoint descriptor. Potential keypoints are detected, by a multiscale approach where the image is rescaled, using a Difference-of-Gaussian function. Afterwards, ambiguous keypoints, i.e., low contrast points, which are not robust to noise, and those that are poorly localized along edges, are eliminated leaving only the stablest keypoints. In the third step, an orientation is assigned to each keypoint based on a histogram computed by the gradient orientation of a sample of points around the keypoint. Finally, the standard keypoint descriptor is created by sampling the magnitudes and orientations of the image gradient in the patch around the keypoint and building a smoothed orientation histogram to capture important aspects of the patch.

2.2.1.3 SURF

While SIFT (see Section 2.2.1.2) builds an image pyramid with progressively downsampled images, Speed-Up Robust Features (SURF) [Bay+08] uses images with the same resolution but different scaled box filters convolved with the integral image. Potential keypoints are detected with a Hessian matrix that, according to the authors, presents advantages in terms of speed and accuracy. The dominant orientation is obtained through the Gaussian weighted Haar wavelet response within a circular neighbourhood around the keypoint. Thereafter, a 4x4 oriented quadratic grid is laid over the interest point. For each square, the wavelet responses are computed from 5x5 samples extracting the descriptor from it.

2.2.2 Keypoint Matcher

After obtaining the keypoints of the images, a next logical step is to match them in the different images. Basically, a keypoint matching algorithm will take on the characteristic properties, such as a keypoint descriptor, detected in images and compare them in order to find similarities. By doing so, it is possible to identify the same subjects in different images and thus to make its comparative analysis.

2.2.2.1 FLANN

The algorithms described in Section 2.2.1 output a set of keypoints in images. Each keypoint may vary on factors such as the scale where the keypoint was detected and its orientation. Fast Approximate Nearest Neighbor Search (FLANN) [ML09] is a library for processing fast approximate neighbour searches in high dimensional spaces. The main goal of the FLANN library is to analyse data sets, which in this case is a set of keypoints, between different images and then match corresponding elements between different data sets. Based on the characteristics of the data sets, the most appropriate nearest neighbour searches algorithm is automatically chosen. Beyond the choice of the most appropriate algorithm, the optimal values for the algorithm parameters are also the computed. As for the collection of algorithms to be used, the authors identified algorithms that use hierarchical k-means trees or multiple randomized kd-trees to be the ones that provide the best result in nearest neighbour searches.

2.2.2.2 RANSAC

The result of FLANN [ML09] is a set of matches, i.e. pairs (k_1, k_2) of nearest neighbours, where k_1 and k_2 belong to different images. However, the appearance of outliers may occur. An outlier is a false-positive match, where k_1 and k_2 are matched as being the same point in different images but in reality they are not. Random Sample Consensus (RANSAC) [FB81] is an iterative method that analyses a mathematical model, in this case a set of keypoint matches, which may contain outliers. It is a resampling technique that generates candidate solutions by using the minimum number of iterations required to estimate the underlying model parameters [Der10]. For a specific number of iterations, a sample of n matches is selected and a homography H is computed from those n matches. Each match is then classified as inlier or outlier depending on its concurrence with H. After all the iterations are done, the iteration that contained the largest number of inliers is selected. Then, H can be recomputed from all the matches that were considered as inliers in that iteration [Dub09].

2.3 Focus Detection Algorithms

Digital images may be analysed in terms of focus. There are different approaches to focus detection such as analysis based on the spatial domain or a study over the frequency domain. In this section focus detection techniques are presented, as well as some background study for it to be possible.

2.3.1 Fourier Transform

Image processing can be done both in its spatial and frequency domains. The term spatial domain refers to the aggregate of pixels composing an image. Thus, when working in an image spatial domain the procedures are realized directly on the pixels.

Jean-Baptist Joseph Fourier (1768-1830) in his book, *The Analytic Theory of Heat* [FF78], stated that any function can be expressed as the sum of sines and/or cosines of different frequencies. Such discovery is important and useful in the branch of image processing as it allows the conversion between the spatial and frequency domains of an image.

The Fourier Transform decomposes an image into sine and cosine components representing the image frequency domain. When applied on a discrete function, such as digital image, it is commonly referred as the Discrete Fourier Transform (DFT) [GW01]. For an image of size MxN, the two-dimensional DFT, which converts the spatial domain into the frequency one, is given by the mathematical expression:

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})}$$

for $u = 0, 1, 2, ..., M-1$ and $v = 0, 1, 2, ..., N-1$ where $f(x,y)$ represents the image in the spatial domain and j is a complex number of value $\sqrt{-1}$.

2.3.2 Wavelet Transform

The Fourier transform can be a powerful tool to analyse the frequency components of a signal, such as a digital image. However the temporal information is lost in the transformation process.

The wavelet transform takes the temporal information into account making it easier to compress, transmit and analyse images. Unlike the Fourier transform, whose functions are based on sinusoids and thus do not have location, the wavelet transform are based on small waves, called wavelets, of varying frequency and limited duration [GW01]. Thus, it is able to capture both frequency and location (in time) information.

Wavelets became the foundation to an approach called the multiresolution theory. As the name implies, this theory consists in the representation of a function signal in multiple resolutions. Thus, features that cannot be detected in a determined scale may be detected in a different one.

2.3.3 Focus Quality Classification

An appropriate classification of an image focus quality enables the detection of focused regions and allow their distinction from blurred ones. Such analysis can be based on peculiar characteristics of an image [Liu+08] or, another possible approach, is to compare similar images [LY08; Lu+07], namely by their frequency spectra.

Liu et al. [Liu+08] proposed a method for the detection and classification of blur based on image features such as colour, gradient and spectrum information. This method suitably adapted can also be used to detect focused regions. However, methods based on an image alone, as input, tend to be ambiguous. For instance, a low contrast image have a wide set of common features with a blurry image, such as low frequency variations, and thus can be computationally interpreted as blurry.

When several similar images are to be processed, there is the advantage of having the ability to compare their frequency spectra. Theoretically, a focused image have higher frequency variations than a blurry one.

Lu et al. [Lu+07] proposed an image fusion method in which they analyse the focus quality of each image pixel by comparing wavelets coefficients of the input image and a blurred version of itself and then compare the result with the respective pixel result of another image. Supposing that the images to be analysed are f_1 and f_2, and their smoothed versions are f_1' and f_2'. Df_1, Df_2, Df_1' and Df_2' represent the high frequency variations of f_1, f_2, f_1' and f_2', respectively, in their neighbourhood. They conclude the following:

If a pixel is focused in f_1 and blurry in f_2, then it is more blurry in f_2', i.e

$$\begin{cases} (|Df_1 - Df_2'| - |Df_1 - Df_2|) \geq T \\ (|Df_1 - Df_2| - |Df_1' - Df_2|) \geq T \end{cases}$$

If a pixel is focused in f_2 and blurry in f_1, then it is more blurry in f_1', i.e

$$\begin{cases} (|Df_1' - Df_2| - |Df_1 - Df_2|) \geq T \\ (|Df_1 - Df_2| - |Df_1 - Df_2'|) \geq T \end{cases}$$

If a pixel is focused or blurred both in f_1 and f_2 , then

$$\begin{cases} (|Df_1 - Df_2| - |Df_1 - Df_2'|) < T \\ (|Df_1 - Df_2| - |Df_1' - Df_2|) < T \end{cases}$$

Where T is determined through a genetic algorithm.

Another image fusion method, in which the focus is detected, was proposed by Li and Yang [LY08]. Contrary to the aforementioned method, the method proposed by Li and Yang analyses the image spatial frequency based on regions instead of individual pixels. The first step is the fusion of the input images by pixel-by-pixel averaging. Then the fused image is segmented using a normalized cut technique. By doing so it is possible to segment the input images equally according to the result of the fused image segmentation. After the segmentation process, each region of the image is analysed in terms of spatial frequency, which indicates the overall active level in that region. Considering an image of size MxN, where M represents the number of rows and N represents number of columns, the row (RF) and column (CF) frequency variation may be calculated through:

$$RF = \sqrt{\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=1}^{N-1}[F(m,n) - F(m,n-1)]^2}$$

$$CF = \sqrt{\frac{1}{MN}\sum_{n=0}^{N-1}\sum_{m=1}^{M-1}[F(m,n) - F(m-1,n)]^2}$$

where $F(m,n)$ is the grayscale pixel value at position (m,n) of the image F.

Hereupon the total spatial frequency (SF) of the image can be defined as:

$$SF = \sqrt{(RF)^2 + (CF)^2}$$

By applying the previous computations to each region it is possible to do the comparison between respective regions given that the higher is the spatial frequency value the more focused is that region.

2.4 Motion Blur Detection Algorithms

An image may have focused regions and blurred regions. Blur may occur due to an object or an entire region being out of the image depth of field, i.e., out of the volume in which the image is acceptably sharp or it may occur when there is movement in a photographed scene, originating the so called motion blur. This section presents techniques for blur detection and classification as well as relevant background knowledge to that matter.

2.4.1 Image Gradient

The image gradient is a very useful property as it defines directional changes of intensity or colour in an image.

Mathematically, the gradient of a two variable function defines, for each point, a 2D vector taken from the derivatives in the horizontal and vertical directions. For the intensity case, the 2D vector points in the direction of the strongest variation of intensity values wherein the length of the vector corresponds to the rate of variation in that direction.

Computationally, digital images are defined by discrete points so an approximation of the image gradient is computed by convolving the image with a specific kernel. Sobel, Prewitt and Scharr operators are just a few examples of kernels that can compute an image gradient.

2.4.1.1 Sobel Operator

Sobel derivatives have an interesting property since they can be defined to kernels of any size, being constructed quickly and iteratively [BK08].

The Sobel operator computes an approximation of the gradient of an image intensity function combining Gaussian smoothing and differentiation. The directional changes are computed with the following kernels:

$$\mathrm{Gx} = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \qquad \mathrm{Gy} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

Where Gx and Gy represent the horizontal and vertical changes respectively.

2.4.1.2 Prewitt Operator

Prewitt presented another operator to compute the gradient of the image intensity function. This operator is composed by the convolution of two filters with the image [BK08].

Unlike Sobel operator, Prewitt does not empathize the pixels that are closer to the center of the mask. The directional changes are computed with the following kernels:

$$Gx = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} \qquad Gy = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix}$$

Where Gx and Gy represent the horizontal and vertical changes respectively.

As refered by Shrivakshan [Shr12] the biggest disadvantage of Sobel (see Section 2.4.1.1) and Prewitt operators is their sensibility to noise, which may lead to a degradation of the gradient vector computation and therefore the appearing of inaccurate results.

2.4.1.3 Scharr Operator

The Scharr operator aims to improve the Sobel (see Section 2.4.1.1) and Prewitt (see Section 2.4.1.2) operators inaccuracy by minimizing errors that occur in the Fourier domain [BK08]. Convolving the image with small filters, such as a 3x3 filter, may result in output images where the inaccuracy is more notable. Scharr operator tries to surpass this problem by giving more emphasis to the pixels that are closer to the center of the mask.

The directional changes are computed with the following kernels:

$$Gx = \begin{bmatrix} -3 & 0 & +3 \\ -10 & 0 & +10 \\ -3 & 0 & +3 \end{bmatrix} \qquad Gy = \begin{bmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ +3 & +10 & +3 \end{bmatrix}$$

Where Gx and Gy represent the horizontal and vertical changes respectively.

2.4.2 Blur Detection & Classification

A possible approach to blur detection was proposed by Tong et al. [Ton+04]. Their method is based on an analysis over the edge type and sharpness, using Haar Wavelet Transform. According to the authors, when blur occurs both the edge type and its sharpness change. They classify edges into Dirac-Structure, Roof-Structure, Astep-Structure and Gstep-Structure. A graphical description of the different types of edges can be seen in Figure 2.8.

Roof-Structure and Gstep-Structure have a parameter α, $(0 < \alpha < \pi/2)$, which indicates the sharpness of the edge: the larger α is, the sharper the edge is. As the authors state, most natural images are likely to contain all types of edges. When blur occurs, the Dirac-Structure and Astep-Structure will disappear while Roof-Structure and Gstep-Structure tend to lose their sharpness, i.e., the α value becomes smaller.

Their proposed algorithm starts by performing Haar Wavelet Transform to the original image with a decomposition level of three. The result is a hierarchical pyramid structure. Then they construct the edge map for each scale of the pyramid in order to find the

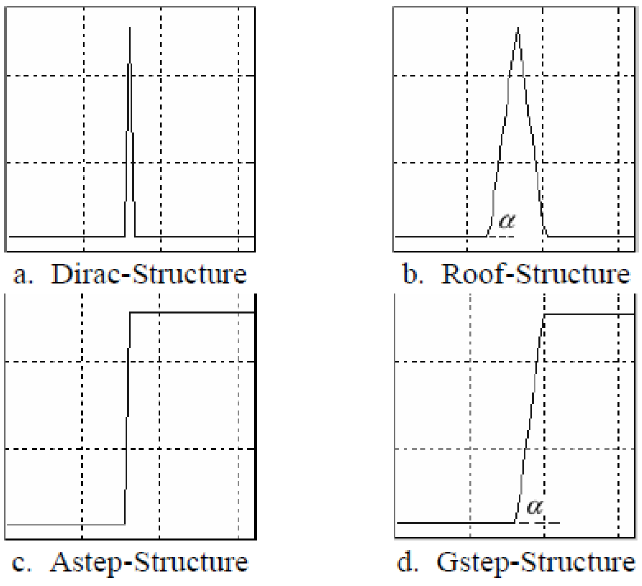


Figure 2.8: Different edge types [Ton+04].

local maxima ($Emax_i$) in each scale. $Emax_i$ represents the intensity of the edge, where $i = (1,2,3)$. For a given threshold, if $Emax_i(k,l) > threshold$, then (k,l) is labelled as an edge point. An important property of Haar Wavelet Transform is its ability to recover the sharpness of blurred edges: when observed in small scales, the Roof-Structure and Gstep-Structure tend to recover their sharpness. Thus, based on the $Emax_i$ value the determination of whether an image is blurred or not is possible. For any edge point (k,l), this determination is based on five rules:

1. If $Emax_1(k,l) > threshold$ or $Emax_2(k,l) > threshold$ or $Emax_3(k,l) > threshold$, then (k,l) is an edge point.
2. If $Emax_1(k,l) > Emax_2(k,l) > Emax_3(k,l)$, then (k,l) is Dirac-Structure or Astep-Structure.
3. If $Emax_1(k,l) < Emax_2(k,l) < Emax_3(k,l)$, then (k,l) is Roof-Structure or Gstep-Structure.
4. If $Emax_2(k,l) > Emax_1(k,l)$ and $Emax_2(k,l) > Emax_3(k,l)$, then (k,l) is a Roof-Structure.
5. For any Gstep-Structure or Roof-Structure, if $Emax_1(k,l) < threshold$, then (k,l) is more likely to be in a blurred image.

Based on the aforementioned rules, the calculation of: the number of Dirac-Structure and Astep-Structure edge points (N_{da}), the number of Roof-Structure and Gstep-Structure edge points (N_{rg}) and the number of Roof-Structure and Gstep-Structure edge points that have lost their sharpness (N_{brg}) is possible. Then they compute $Per = N_{da}/N_{edges}$, where

N_{edges} represents the total number of edge points. If $Per > MinZero$, where $MinZero$ is a positive number close to zero, then the image if assumed to be focused. Otherwise the equation $BE = N_{brg}/N_{rg}$ outputs the blur confidence coefficient for the image.

Other examples of possible approaches to blur detection and classification are based on the image gradient as it offers a great source of information not only in blur detection but specially in its classification.

Liu et al. [Liu+08] proposed a method for partially blurred images classification and analysis modelled by image colour, gradient and spectrum information. Their system starts by detecting blur and in a second step they classify it. The detection of blurred regions is based on three assumptions:

1. The amplitude spectrum slope of a blurred region tends to be steeper than that of a focused one.

2. Blurred regions rarely contain sharp edges, which result in smaller gradient magnitudes.

3. Focused regions are likely to have more vivid colours than blurred ones so the maximum saturation in blurred regions is expected to be smaller than that in focused regions.

Hereupon by analysing the aforementioned features the detection of blurred regions is supposedly accomplished. Their next step is to classify the detected blurred regions. To achieve that they create a smoothed version of the input image, through a Gaussian function, given that:

1. If an image patch is mostly directional-motion blurred, the edges with gradient perpendicular to the blur direction will not be smoothed, leaving strong gradient only along one direction.

2. If an image patch is approximately focal blurred, the gradient magnitudes along all directions are attenuated.

Based on these statements, they construct a directional response histogram for all image patches, where each bin represents a specific direction and the value of that bin denotes the number of pixels which have that direction. Thus, when in face of a motion blurred patch, the histogram presents a distinctive peak in the direction of motion.

Thus, the differentiation between blur is accomplished and it can be classified as motion blur or out-of-focus blur.

Su et al. [Su+11] proposed another approach to blur classification. Their method is based on a two-layer image composition model, in which each pixel I in an input image is viewed as a linear combination of a foreground colour, F, and a background colour, B, using an alpha mate value. It is defined as follows:

$$I = \alpha F + (1 - \alpha)B$$

where α can be any value in $[0,1]$. In focused images most α values are either zero or one. If $\alpha = 1$, pixel I is considered foreground, otherwise it is denoted as background. However, in a blurred image, foreground and background tend to mix together, namely on its boundaries, so α values become decimal as certain pixels are composed by both foreground and background colours.

Hereupon they analyse the alpha channel model constraint, defined by $\nabla\alpha.b \in \{-1,1\}$, where b is a vector that denotes the blur extent in horizontal and vertical direction. The blur kernel b of motion blurred images is usually directional, so $\nabla\alpha$ will be lines while in an out-of-focus image the $\nabla\alpha$ will be spread out at every direction, in a radial manner. Thus by the analysis of the $\nabla\alpha$ circularity pattern, the distinction between motion blur and out-of-focus blur is achieved.

2.5 Noise Detection Algorithms

Digital noise is usually seen as alienated dots scattered through the image. It is one of the primary factors for image degradation. There are several approaches to noise detection. This section presents filtering-based detection and block-based detection methods.

2.5.1 Filtering-based Detection

The filtering-based approaches, such as the one proposed by Nguyen and Hong [NH12], are done by filtering the noisy input image with a low-pass filter in order to obtain a smoothed version of it. Then, the standard deviation of the difference between the noisy image and its smoothed version is computed. The process schematic is shown in Figure 2.9.

The insight of this method lies in the fact that low pass filters tend to attenuate the existing noise in an image and so the difference between the noisy image and its smoothed version will accentuate the noise.

2.5.2 Block-based Detection

Another possible approach to noise detection is a block-based one. Lee and Hopel [LH89] proposed a block-based algorithm in which the smallest standard deviation of intensity in an image block is assumed to be equal to that of the additive white Gaussian noise. A block-based approach starts by tessellating the input image in MxN sized blocks. For each block the standard deviation of intensity is computed and then the resulting standard deviations are sorted. Given that the intensity variation of a smooth block may be due to noise then the standard deviation of the smoothest block is considered close to that of the Gaussian noise added. Thus having intensity variation of the smoothest block as a reference it is possible to estimate the level of noise in an image.

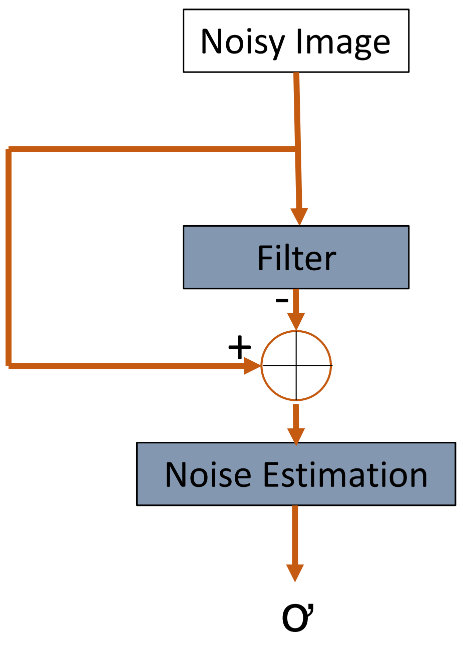


Figure 2.9: Filtering-based noise estimation schematic.

2.6 Colour Analysis

Colour plays an integral part in image processing as it can provide a powerful descriptor about the information contained in an image. It is known that humans can discern thousands of colour shades and intensities. However they can only differentiate two dozen shades of gray [GW01]. Such facts are taken into account by the image processing community so that the best results possible can be achieved. This section addresses some important concepts of colour and how they can be analysed, namely by the use of histograms.

2.6.1 Colour Space

Colour space is a specific organization of colours. The RGB is an additive colour space based on the colours of **R**ed, **G**reen and **B**lue. A colour in the RGB space is described by how much of each of these components it includes. This is one of the most well known colour spaces as it is used in electronic devices such as televisions, computer screens and digital camera sensors.

Another important colour space is the HSL, which was created to match the human visual system. It stands for **H**ue, **S**aturation and **L**uminance. Hue refers to the nature of colour having designations such as *red*, *blue* or *yellow*. Saturation designates the colour intensity being that an intensive colour is seen as lively while a low intensity colour is a faint one [San13]. Lastly, luminance refers to the brightness that colour presents.

2.6.2 Colour Temperature

Each light source has its own colour, or *temperature*. Cool colours like blue or violet, defined by smaller wavelengths (400nm-500nm), usually have temperatures between 7000 kelvin to 10000 kelvin. On the other hand, warm colours like red or orange have temperatures from 1000 to 2000 kelvin. In table 2.1, the difference between common light sources can be seen, in terms of temperature [San13].

Table 2.1: Light sources temperature.

Temperature (Kelvin)	**Light Source**
1000-2000	Candle light
2500-3500	Homemade tungsten lamp
3000-4000	Sunset and sunrise in a clear sky
4000-5000	Fluorescent lamp
5000-5500	Camera flash
5000-6500	Sunlight in a clear sky at noon
6500-8000	Sunlight in a cloudy sky
9000-10000	Sunlight in a heavily cloudy sky

2.6.3 Histogram Processing

A histogram is a graphical representation of how some data values are distributed. In image processing, this is a valuable tool for image evaluation since it allows the analysis of certain image parameters behaviour. For instance, an analysis done on a grayscale image, showing its brightness distribution, can be a hint to distinguish a well exposed image from an underexposed or overexposed one. That is possible because a well exposed image tends to contain a larger range of colours than the other two.

2.7 Photo Quality Estimation

This section presents two similar works [Chu+07; Pot+09] to that presented in this document. Both works aim at estimating the quality of photos, not necessarily similar, in an album. This estimation is based on technical characteristics such as focus and illumination. Low quality photos are detected and automatically removed from prior processing.

2.7.1 Automatic Photo Selection for Media and Entertainment Applications

Potapova et al. [Pot+09] proposed an algorithm for media and entertainment applications like photobook and slideshow. Their technique comprises three main steps: detection of low quality photos, adaptive quantization of the remaining and automatic selection of the most appealing photos. For time optimization, the algorithm works on downsized versions of each image.

In a first step, low quality photos, such as images containing an uneven exposure, blurriness and JPEG artefacts, are detected using machine learning techniques and are not taken into further processing.

Photos affected by compression artefacts, which as stated by the authors looks as noise, are detected through a filter for deblocking and deriging method [Foi+06]. For adjustment of the filter parameters the top left corner of the square of $3 * 3$ quantization table q of the brightness channel is analysed:

$$k = 1/9 \sum_{i,j=1}^{3} q_i, j$$

According to the authors experiments, if k is higher than 6.5 then the photo is considered to be strongly affected by compression artefacts.

Low contrast photos, namely the ones affected by backlighting, are identified from an analysis over the brightness histogram. The ratio of tones in shadows to midtones and the ratio of the histogram maximum in shadows and the global histogram maximum are two of the values that serve as input to the machine learning algorithm AdaBoost [FS95] in order to detect low contrast photos.

For the identification of blurred photos, each image of the album is convolved with high-pass filters of different sizes. Then an analysis over the variation of the edges is done over a histogram. The entropy of the histogram characterizes the flatness or peakdness of each histogram. In order for this method to be more effective, the variation between each image and its smoothed version is also computed. If the variation is high, then the original image is in sharp focus, otherwise it is already blurred. Once again the computations are analysed by the AdaBoost algorithm in order to detect blurred photos.

As a result of the aforementioned approaches, low quality images are removed from further processing. In order to cluster the remaining images, the authors took into consideration the moment when the photo was taken and also the camera used to take the photo. That information is obtained through the Exchangeable Image File Format [JEI02] (EXIF). Afterwards, in order to select the best photo from each cluster, they assumed that the most appealing and relevant photo is the most notable one, i.e., the most salient one. Thus, a saliency map is constructed based on intensity, orientation and colour maps being a specific weight applied to each map.

2.7.2 Tiling Slideshow

Chu et al. [Chu+07] proposed a system that automatically generates audiovisual slideshows, in which multiple photos with similar characteristics are displayed at the same frame. Although their work has not the end purpose of selecting the best photo from the album, one of the procedures consists in estimating the quality of each image of the album. They consider blurred, underexposed and overexposed photos to have low quality and thus must not be taken into consideration for the slideshow presentation. For the detection of blurred photos, they adopted a wavelet-based method [Ton+04] that analyses edge characteristics at different resolutions, which was already referenced in Section 2.4.2. Underexposed and overexposed images are detected by comparing the number of dark pixels and bright pixels to a determined threshold. If one of the values surpasses the threshold value then its respective image is considered to underexposed (dark pixels) or overexposed (bright pixels). At this point they claim to have detected the low quality images, so their next step is to group them based on timestamps and content.

2.8 Image Processing Applications

With the advances in photography and technology, the number of image processing applications are constantly increasing. Applications that analyse images from an exposure point of view are no exception. In this section are briefly presented some of those applications, ending with a discussion comparing them and relating them with our work.

2.8.1 Adobe Bridge

Adobe Bridge is a digital asset management application, developed by Adobe Systems[Incc], which may work as a pre-processing unit before editing the photos in Adobe Photoshop. This application offers the option of organizing sets of images in different collections wherein the user can quickly preview them using a slideshow and also filter them according to metadata parameters such as the ISO speed, exposure time, focal distance and white balance.

2.8.2 Adobe Lightroom

Adobe Lightroom, developed by Adobe Systems [Incc], lets the user import, organize and create collections of photos and also offering insightful metadata information, such as histograms. For a comparison between images, the user can choose the compare and survey views, where it is possible to compare pairs of images in more detail, namely by allowing to zoom respective parts of different images and make a visual analysis. However as the images are not correlated there is still the necessity to navigate in their scope in order to compare respective regions of interest.

A key strength of both Adobe Bridge and Adobe Lightroom is how quickly a set of photos is imported into the application. This process is fast (less than one second) even when in face of a large set of photos. By analysing these software products, it seems that the set of photos is uploaded in a low resolution in order to be fast. After the initial import, background operations are applied in order to import each photo in its original resolution.

2.8.3 Capture One

Capture One [One] is an image processing and managing application. This software product provides a feature entitled *Session*. Each *Session* contains a set of photos, uploaded by the user. When a *Session* is created, four folders are also physically created: Capture, Selects, Output and Trash. The Capture folder contains the set of photos uploaded by the user. The user can then visually analyse the set of photos and select those of interest into the Selects folder. The unwanted photos can be placed into the Trash folder. The selected images can then be adjusted according to parameters such as exposure, contrast, high dynamic range or white balance. Thereafter, these photos can be put into the Output folder, being considered the best photos of the album.

2.8.4 DxO Optics Pro

DxO Optics Pro [Laba] is a standalone image processing application that can adapt to any host application. For any photo, the exposure settings can be consulted through the EXIF specification [JEI02]. As stated by the authors, this software product's main advantage is the ability to enhance low quality images. Uneven exposures can be treated by changing the brightness of the image. The authors also state that the denoising operation can be made in a lossless way, maintaining the image sharpness. The highlights and shadows details can also be recovered in order to enhance the image quality. A compare view, where two or more images can be compared is also available. However, similar images are not correlated and thus a comparative analysis over a region of interest is not possible.

Unlike Adobe Bridge and Adobe Lightroom, the importation of the set of similar photos is quite slow, which worsens the user experience. We considered the lossless noise reduction operation to be DxO Optics Pro's strongest feature. According to what is explained in [Labb], the denoising process in based on a PRIME (Probabilistic RAW Image Enhancement) technology. This technology is based on an algorithm that searches for similarities among pixels, whose values are then averaged. For each pixel, a region of interest around that pixel, containing one thousand pixels, is computed. If the pixels inside the region of interest are similar then their values are averaged. Based on the average value, the noisy pixels can be detected as their values tend to be farthest from the average value. Comprehensibly, the noise reduction algorithm requires heavy processing, thus the denoising process for a high resolution image may take several minutes.

2.8.5 Google+

Google's social network, Google+ [Goo], took into consideration the fact that the user can upload a large set of photos and thus offers some automatic features to simplify the user task of choosing the best photos. In the context of this work becomes important to empathize the *Auto Highlight* feature, which can be used to highlight the best photos, tossing duplicates as well as improperly exposed ones. Besides, automatic corrections are also applied, namely in terms of brightness, contrast, saturation, noise, among other parameters.

2.8.6 Photo Mechanic

Photo Mechanic is an image processing application, created by the Camera Bits company [Inca], available for the Mac OS X and Windows operative systems. This software product does not involve complex image processing algorithms. Its main goal is to provide a quick way for the user to visually analyse a set of (similar) photos. The option of a selective speeded up slideshow of images is offered, wherein thereafter the user can choose the best ones from the sequence. Furthermore, the user is able to sort the images based on metadata such as the ISO speed, exposure time, focal distance or size.

The process of visual observation, on a computer screen, in order to select the best photo is inaccurate and thus becomes a generator of insecurity feelings that may lead to selecting a photo that in reality is not the best photo from the set of similar photos. Thus, using a software product like Photo Mechanic may not be the best option for a thorough selection of the best photo from a set of similar photos.

2.8.7 Discussion

From the aforementioned applications, it may be considered that Adobe Bridge [Incc] and Photo Mechanic [Inca] are very similar in terms of goals and functionalities. These software products do not use complex image processing algorithms to analyse the image. They seek to help the users organize their sets of photos, offering access to metadata information. Adobe Lightroom [Incc] offers a wider set of functionalities, namely a more detailed comparison between each pair of photos, which makes the process of choosing the best one more reliable. Google+ [Goo] took a more venturesome approach by automatically choosing the best photos from each set. Blurriness, underexposure and overexposure are some of the factors that define low quality photos as well as photos where people is affected by red eyes. In Google+ these photos are detected and removed, remaining only the ones deemed to have good quality.

None of mentioned applications are exclusively dedicated to the process of choosing the best photo from a set of similar photos, giving enough safety and reliability to the process and, above all, sets of similar images are not correlated and thus an automatic comparative analysis between similar images is not possible. The goal of the work presented in this document is to fill this gap by helping the user to choose, in a flexible, quick and precise way, the best photo from a set of similar photos.

2.9 OpenCV

Open Source Computer Vision (OpenCV) [Its] is an open source computer vision library widely used as an infrastructure for computer vision applications given its large set of algorithms.

OpenCV has a modular organization that includes components such as Image Processing, Video Analysis, Camera Calibration and 3D Reconstruction, Machine Learning, a basic graphical user interface and a few more others.

Due to the context of this work, it is significant to emphasize the Image Processing module. Having this module as basis, it is possible to do operations like image filtering, image segmentation, histogram computation and processing, detection of image features among other operations.

OpenCV was written in optimized C/C++ having C, C++, Python, Java and MATLAB interfaces. Besides, it runs under the operating systems of Windows, Linux, Android and MacOS.

Chapter 3

System Description & Features

This chapter describes the most relevant aspects of the developed system such as the concept, the workflow and the general structure. For each phase of the workflow, the functionalities are explained in detail, namely their implementation and evaluation. All experiments have been carried out on a PC Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz 2.40GHz with 8.00 GB RAM and running under a 64 bits Windows 8.1 operating system.

3.1 System Description

As referred in Section 1.2 a thorough best photo selection, from a set of similar photos, may turn out to be a hard task for any photographer. Not only is there technical details that may escape a visual analysis but also the time consumed and consequently the demotivation may lead the photographer to choose a photo that in reality is not the best photo according to his/her quality standards. This section describes in greater detail the solution that we present to the photographer, in particular the concept of the proposed workflow, the goal of each of its phases and the challenges that they offer.

3.1.1 Concept

Photographies can be analysed in terms of their exposure. The focused regions attract the viewer attention immediately, the blurred regions may be used to embellish the photograph, namely by demoting the background or by the creation of the motion blur effect. There are situations where noise can be originated, for instance a low-light room requires high values for both ISO and shutter speed contrasting with a smaller value for aperture, thus the existing light is captured quicker, having as consequence the possible appearance of noise. Furthermore, the different registration of light by the camera sensor can lead to different levels of exposure and colour cast.

As mentioned in Chapter 2, there are different approaches to classify an image in terms of focus, blur, noise and colour. However, to our best knowledge, such approaches have never been combined into a unique workflow that aims at analysing similar images from an exposure point of view.

Taking the aforementioned fact into account, the idea of the Best-Photo Selection software tool emerges. Our objective is to create a software product that aims at helping its user to choose the best photo among a set of similar photos. The process of choosing the best photo is based on a workflow. The user is guided through each phase where he/she can selectively eliminate the photos that do not match his/her quality standards. The expected result is a much smaller set of photos, when compared to the original one, thus facilitating the process of choosing the best photo.

3.1.2 Workflow

The workflow describing the procedure of selecting the best photo, from the user point of view, can be seen in Figure 3.1.

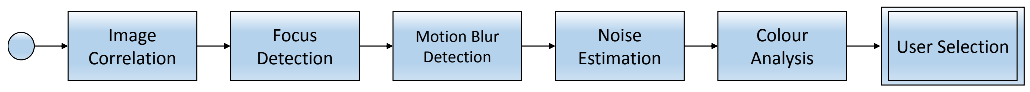


Figure 3.1: Workflow schematic.

The workflow ordering reflects the relevance of the problems that each phase presents. For instance, an improper focus is an irreversible effect and so becomes an eliminatory factor. Hereupon focus detection is the first phase to be analysed in the workflow (only preceded by image correlation as it represents a pre-requisite for some of the remaining phases). The attenuation of noise can be made using specific software products, however some information is lost in the process. Thus, the noise estimation becomes the third most relevant problem. At last, there are several techniques for changing the level of exposure and temperature of an image, having a reduced impact on the photo quality. Thus, the colour analysis is the last procedure in the presented workflow.

Note: In Section 1.3, the level of exposure and colour temperature were separately introduced for the sake of a better understanding of the presented solution. However, these procedures are based on the image colours. Thus, in the presented workflow schematic they are included into a single procedure, called Colour Analysis. In the context of this work, colour will be analysed in order to define an image level of exposure and also its colour cast.

Image Correlation

In order to do a comparative analysis between similar images, there is the need to match scenes among different photos. The result of the correlation process allows the automatic identification of points, or areas, between two similar images. This procedure does not require any user input. Its main purpose is to enable a comparative analysis between similar images (when needed) in the following procedures of the workflow. Given that the input images may vary significantly on technical characteristics, such as focus, depth of field, luminance or orientation, the main challenge that this procedure presents is to find a methodology that is invariant to these changes.

Focus Detection
This procedure aims at presenting, to the user, a mask over the sharp focused regions of each image. Thereafter, the user can reject the ones that do not have a properly focused center of interest.

Motion Blur Detection
As in the previous procedure, this procedure also aims at presenting a mask over the regions of the image that are affected by motion blur. It will then be the user to decide which photos to reject. The distinction between out-of-focus blur and motion blur is the main challenge in this procedure.

Noise Estimation
The set of similar images are sorted according to their level of noise in this procedure. By principal, photographers aim at taking photos containing the less noise possible. Thus the user will then be able to reject the images containing a different level of noise from the intended.

Colour Analysis
At last, this procedure aims at sorting the set of similar images according to their exposure level (from darker to brighter). After the removal of those containing an undesired exposure level, the user will be presented with the images that are considered to be *cold*, *warm* and those that contain a correct white balance.

User Selection
After the completion of the procedures involving a computational analysis, the set of similar photos is expected to be much smaller when compared to the original one. At this point, if there is more than one photo remaining, the user will have to choose the best one according to different parameters, such as framing or composition.

3.2 Features

This section describes in detail the functionalities as well as their implementation, the experiments performed and the final result. Each phase of the workflow is considered to be a functionality, even though the first one (image correlation) works as a pre-requisite for the remaining.

3.2.1 Image Correlation

The procedure of image correlation is divided into two phases. Initially, there is the necessity to detect keypoints in each image of the set of similar images. Keypoints are distinguishable points in the image scope, such as corners. In a second phase the correlation of respective keypoints between different images is needed. The proceeding overview can be seen in Figure 3.2.

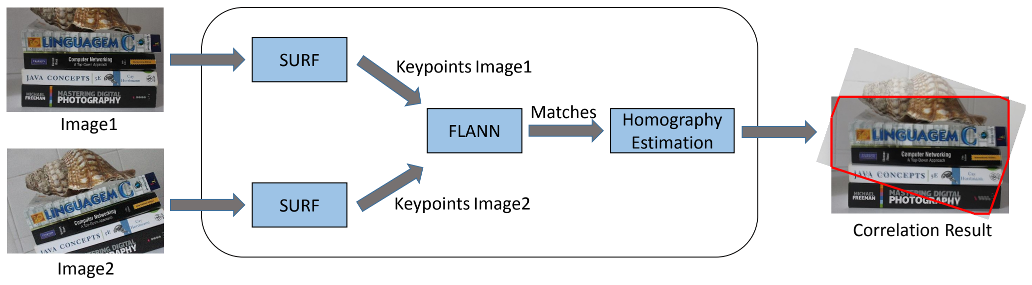


Figure 3.2: Proceeding overview for image correlation.

The algorithms studied for keypoint detection, SIFT [Low04] and SURF [Bay+08] (examined in greater detail in Section 3.2.1.2), claim to be invariant to scale and rotation changes. However, these algorithms are only partially invariant to illumination changes (as proved in Section 3.2.1.1). Thus our first step was to normalize the colour tones of images using the histogram equalization operation.

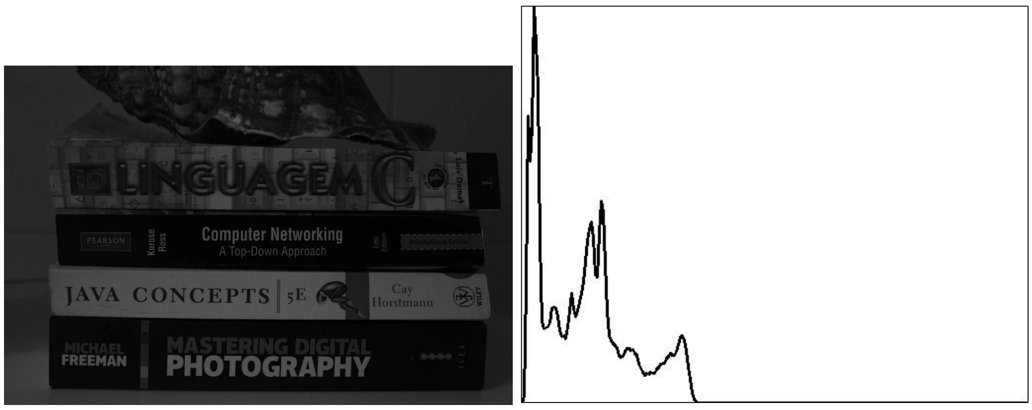
3.2.1.1 Histogram Equalization

The operation of histogram equalization can be used to improve the contrast and brightness of an image by stretching out the intensity range [Bag+12]. Given that the effectiveness of the correlation process rises as the similarity between images rises, the use of the histogram equalization operation becomes insightful in the context of this work. With particular emphasis on the fact that the input images may vary significantly in illumination matters, our first step was to normalize the intensity of the image colours in order to make them even more similar and thus make the correlation process more effective. To do so, the histogram equalization operation, available in the OpenCV library [Its], was used:

```
void equalizeHist(InputArray src, OutputArray dst)
```

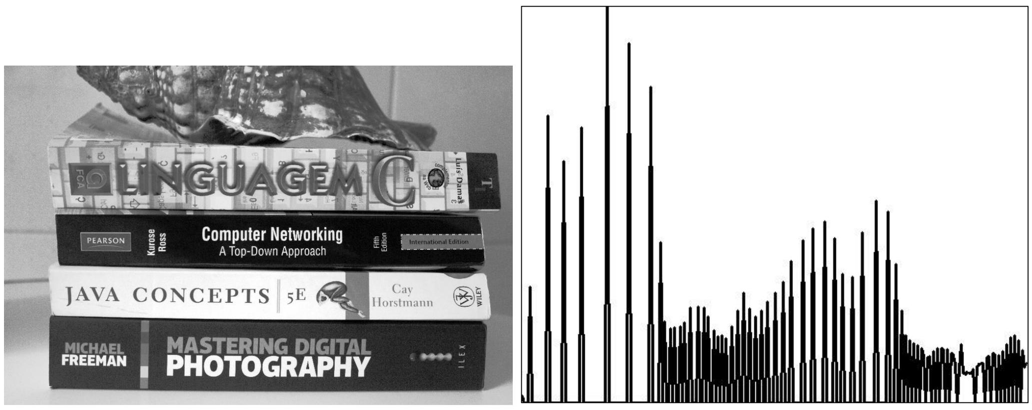
Where `src` represents the input image and `dst` represents the output image. Figure 3.3 represents an underexposed image and its corresponding histogram of colour intensities. Figure 3.4 represents the result of the histogram equalization.

The graphic shown in Figure 3.5 presents the impact of the histogram equalization operation applied over an underexposed image (Figure 3.3(a)) at different resolutions. The number of keypoints detected is clearly superior after this operation was applied.



(a) (b)

Figure 3.3: Underexposed image before histogram equalization (a) and respective histogram of colour intensities (from 0 to 255) (b).



(a) (b)

Figure 3.4: Underexposed image after histogram equalization (a) and respective histogram of colour intensities (from 0 to 255) (b).

Also the number of keypoints detected comes closer to those detected in a reference image (blue graphic), which contains a balanced exposure. The values composing the graphics present in the aforementioned figures can be seen in Tables A.1, A.2 and A.3 of Appendix A.

3.2.1.2 Keypoint Detection

Hereupon the next step is to detect keypoints in each image. SIFT [Low04] and SURF [Bay+08] descriptors are the most promising due to good performance and have now been used in many applications. In Section 2.2.1, the FAST [RD06] algorithm was also referenced for keypoint detection. However FAST always considers the same number of neighbour pixels to detect a keypoint. This fact can lead to undesired results when there are scale changes or blur variations between similar images as the pixels to be processed,

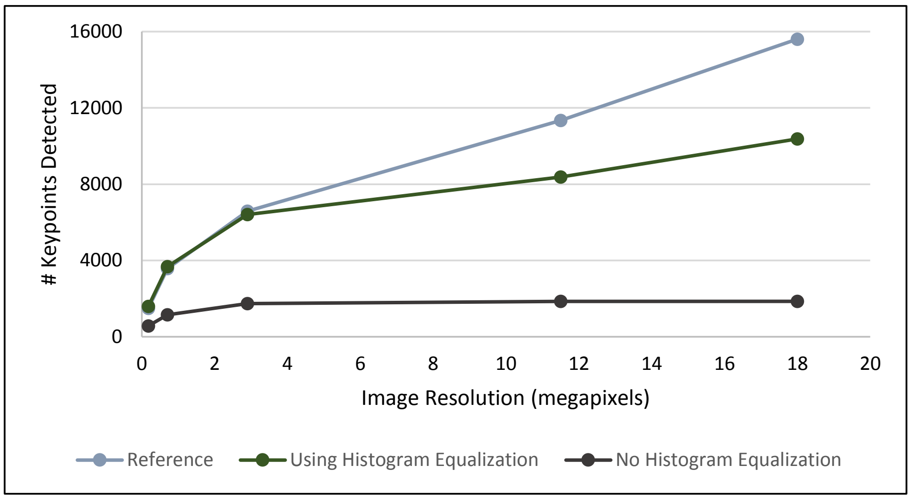


Figure 3.5: Histogram equalization impact on keypoint detection using SIFT.

in order to detect a keypoint, may not be the same. Also illumination changes, which may result from an underexposure or an overexposure, may imply that the neighbourhood pixels become more alike and therefore, once again, potential keypoints may not be detected. Given that, in the context of our work, there may be a wide variety of transformations between similar images, we suspect that FAST may not be the best solution and consequently it was discarded. Thus, SIFT and SURF were tested as they claim to be invariant, above all, to rotation and scale changes. In our experiments, both of the algorithms had similar results in terms of number of keypoints detected. The graphic shown in Figure 3.6 proves that the number of keypoints detected in one image, at different resolutions, tends to have the same order of magnitude when using SIFT and SURF. The values composing this graphic are available in Table A.4 of Appendix A.

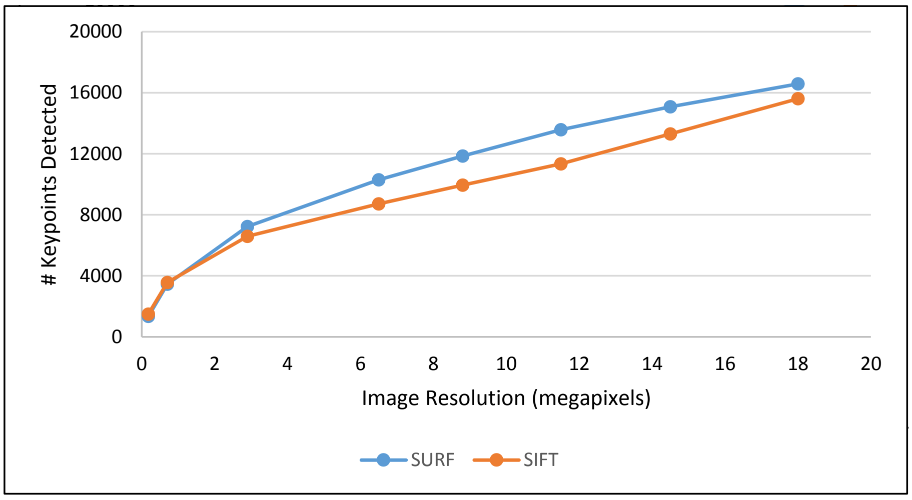


Figure 3.6: Comparison of SIFT and SURF in keypoint detection.

In Figure 3.7 is shown a graphic where SURF proves to be faster than SIFT, wherein the difference becomes significant as the image resolution raises. According to what is explained in the papers related to the algorithms in question [Bay+08; Low04], we believe that the main reason for SURF to be faster than SIFT relates to the fact that SURF always processes the input image at the same resolution, varying only the size of the box filter applied to the integral image. SIFT takes a different approach: each image is processed at several scales and consequently has higher processing times. Given that our objective is to build an interactive tool the processing times must be as low as possible, thus for the sake of efficiency and user satisfaction we opted to use the SURF algorithm, available on OpenCV [Its]. The values composing the graphic in Figure 3.7 can be found in Table A.5 of Appendix A.

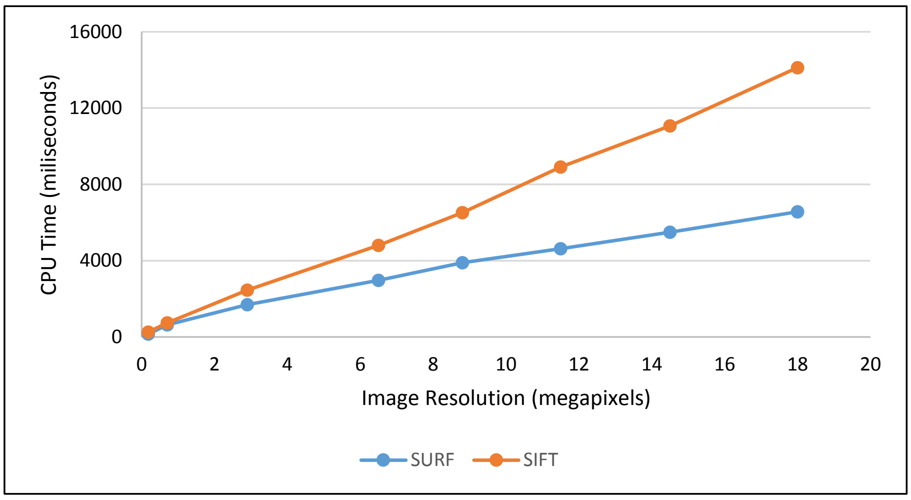


Figure 3.7: Comparison of SIFT and SURF in processing times.

Figure 3.8 presents the result of applying the SURF algorithm to an image. The diameter of each circle is proportional to the scale where the keypoint was detected. The lines going out from the center of the keypoint onto the circumference represent the orientation of the keypoint, which is computed based on the image gradient. As result, for each keypoint is built a descriptor containing information about the scale at which the keypoint was detected and also its orientation.

3.2.1.3 Keypoint Matching

Having computed the keypoints for each image, the next logical step is to match them. In order to match correspondent descriptors, the method `match` and the method `knnMatch`, available on the FLANN library [ML09], described in Section 2.2.2.1,were tested. The constructor of the `match` is defined as:

```
match(const Mat& queryDescriptors, const Mat& trainDescriptors, vector<DMatch>& matches,
    const Mat& mask=Mat())
```

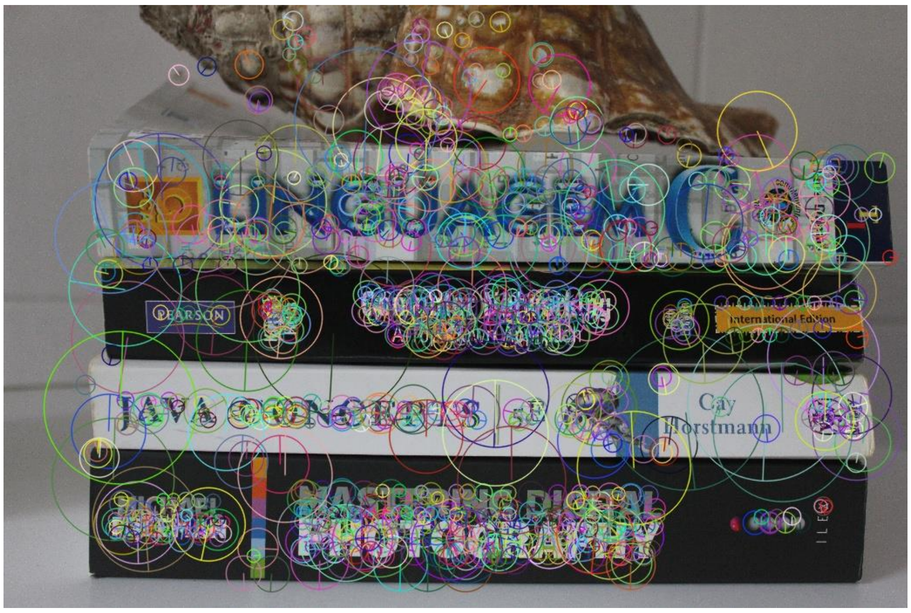


Figure 3.8: Result of SURF algorithm.

The constructor of the `knnMatch` is defined as:

```
knnMatch(const Mat& queryDescriptors, const Mat& trainDescriptors,
    vector<vector<DMatch>>& matches, int k, const Mat& mask=Mat(), bool
    compactResult=false)
```

Where `queryDescriptors` and `trainDescriptors` represent the set of the keypoints descriptors from the two images to be matched.

The method `match` correlates the descriptor of a keypoint from one image with the nearest match from another image. The `knnMatch` method finds the `k` nearest matches for each descriptor.

The biggest challenge in the matching procedure is to minimize the false-positives, i.e., matches where the descriptor of a keypoint is mistakenly matched with that of another image. Using the method `match` a possible approach is to define a global threshold. For all matches found, the minimum distance (`min_dist`) is computed. Afterwards, if the match distance (`m_dist`) between descriptors falls bellow a determined threshold (for instance `m_dist < 2 x min_dist`), the match is considered to be positive, otherwise it is considered a false-positive. However using a global threshold for the filtering process may not have an effective performance as some descriptors are much more discriminatory than others. A more effective measure was proposed by Lowe [Low04] using a ratio test. This measure is obtained by comparing the distance of the closest neighbour to that of the second-closest neighbour, performing well because correct matches need to have the closest neighbour significantly closer than the closest incorrect match. Thus using the `knnMatch` with `k=2`, the ratio test can be used to filter out false-positive matches.

Figure 3.9 shows the difference between the two methods in terms of matches computed while Figure 3.10 shows the difference between the two methods in processing time terms. The values composing the graphics aforementioned were taken from Table A.8 and Table A.9 present in Appendix A.

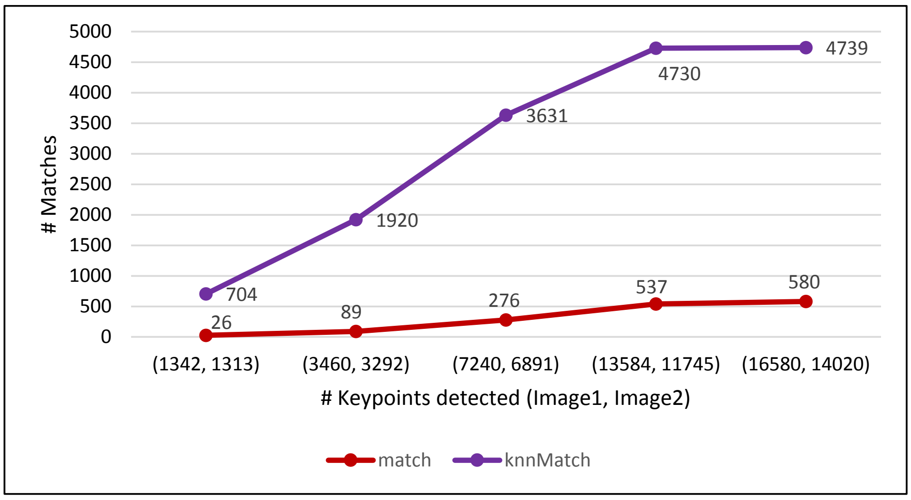


Figure 3.9: FLANN: `match` and `knnMatch` comparison on the matching process.

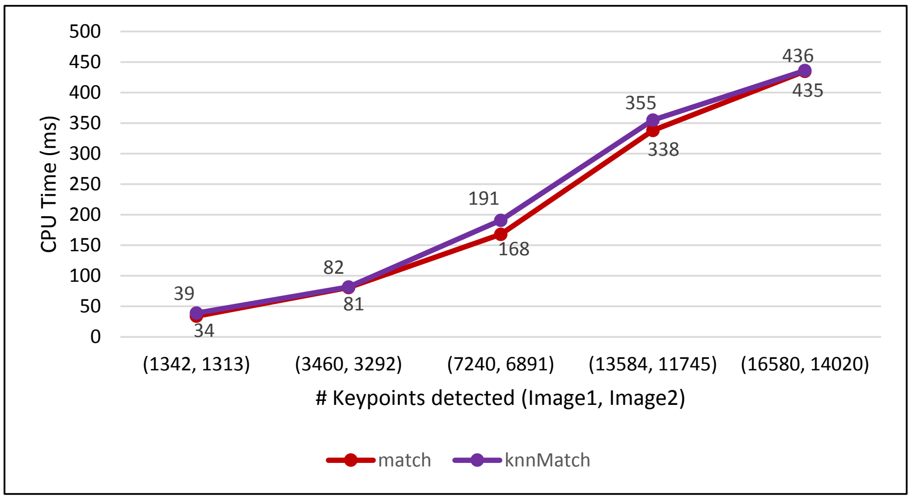


Figure 3.10: FLANN: `match` and `knnMatch` comparison on processing time.

Analysing the graphics, our conclusion is that both methods present similar and satisfactory processing times (varying from 34 to 436 milliseconds). However the `knnMatch` method presents significantly better results in the number of matches computed, specially by non removing positive matches.

Figures 3.11 and 3.12 show a visual illustrative result of the correlation process using the method `match` and `knnMatch` respectively.

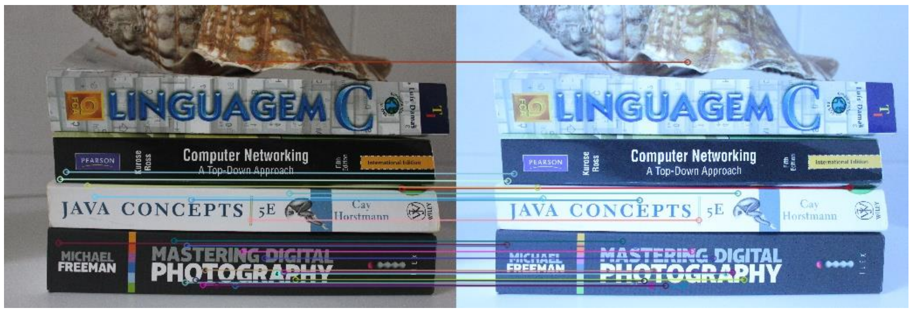


Figure 3.11: Method `match` using $2 * min\_dist$ as a global threshold.

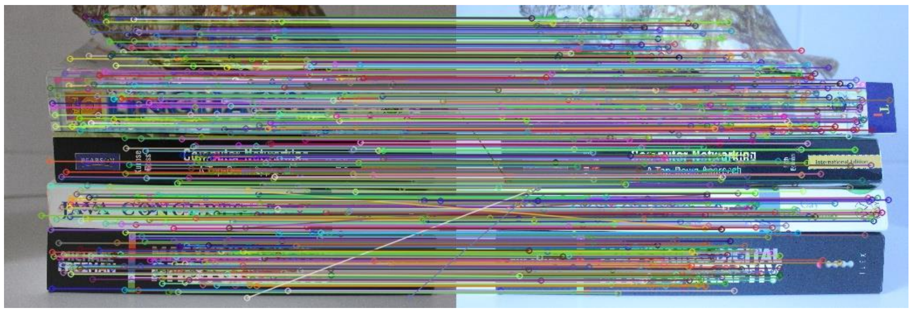


Figure 3.12: Method `knnMatch` using 1/1.5 as a local threshold.

3.2.1.4 Homography Estimation

In Figure 3.12 can be seen (at least) three visible false matches (outliers), marked in red in Figure 3.13. So in order to remove the remaining outliers and thus make the process of image correlation more effective, the homography matrix was estimated using the OpenCV function:

```
Mat findHomography(InputArray srcPoints, InputArray dstPoints, int method=0, double
    ransacReprojThreshold=3, OutputArray mask=noArray())
```

The parameter `method` was set to `CV_RANSAC` in order to use the RANSAC method (see Section 2.2.2.2) for the homography estimation. The output is a 3x3 matrix describing the homography:

$$M = \begin{bmatrix} R11 & R12 & T1 \\ R21 & R22 & T2 \\ P & P & 1 \end{bmatrix}$$

where R represents the rotation matrix, T represents a translation and P represents a perspective transform. By removing the outliers and keeping the inliers, the correlation process becomes more effective.

The final result of this procedure can be seen in Figures 3.14 and 3.15.

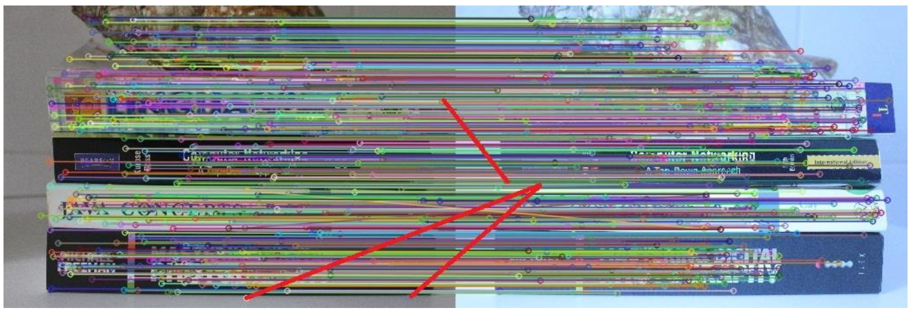


Figure 3.13: FLANN: Three false matches marked in red colour.

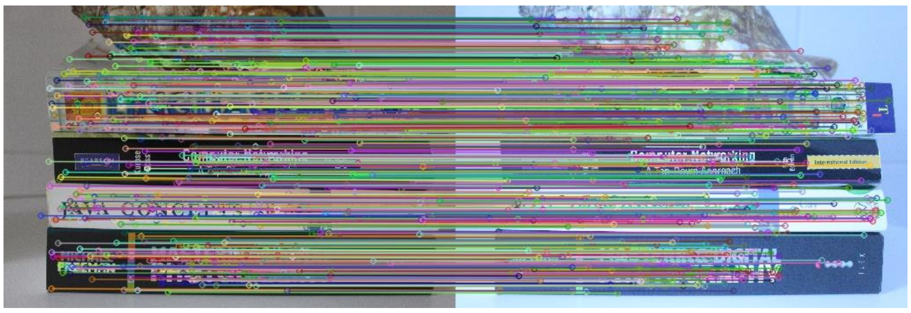


Figure 3.14: Image correlation after the homography estimation (1).

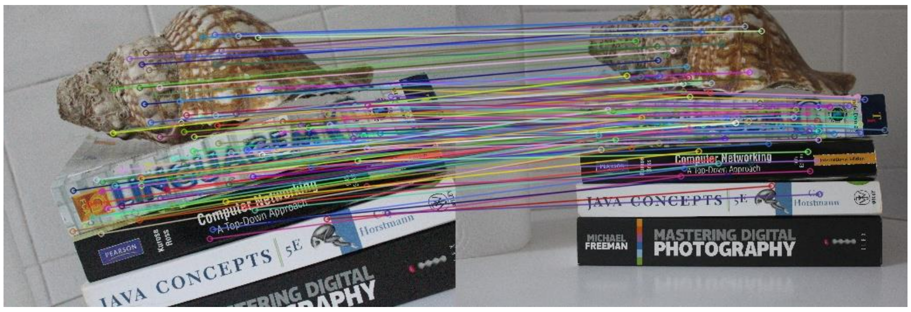


Figure 3.15: Image correlation after the homography estimation (2).

The processing times for homography estimation, according to the number of matches to be processed can be seen in Figure 3.16.

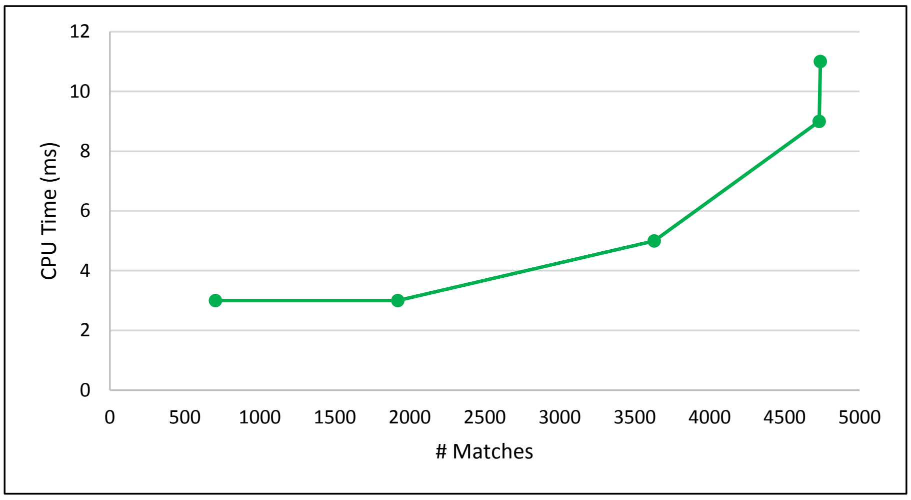


Figure 3.16: Processing times (ms) for homography estimation.

Even for a high number of matches (≈ 5000) the homography estimation is fast, taking approximately eleven milliseconds to be estimated.

3.2.1.5 Correlation Strategy

In order to correlate the universe of similar images, two different approaches were considered: the first one would be the correlation of every image with each other; the second approach has a reference image, which is correlated with the remaining. Once again, for the sake of efficiency and user satisfaction, we opted to implement the second approach, illustratively represented in Figure 3.17.

The illustrative example shown in Figure 3.17 requires nine correlations. However, if the first approach was implemented then 45 correlations would be needed.

3.2.1.6 Discussion & Evaluation

Using the followed methodology for the procedure of image correlation it was possible to overcome the challenges that this procedure initially presented. Namely, an effective correlation of images that vary in terms of illumination, orientation, noise and partially in terms of focus. However, according to our experiments two limitations were found:

1. **Correlation of blurry images**: Images that are completely disfigured, with no clear center of interest, present a scarce number of keypoints detected. Thus, the process of correlating these images with the remaining is not possible. A possible solution is to detect these images in a prior non-reference method, leaving them off the correlation procedure.

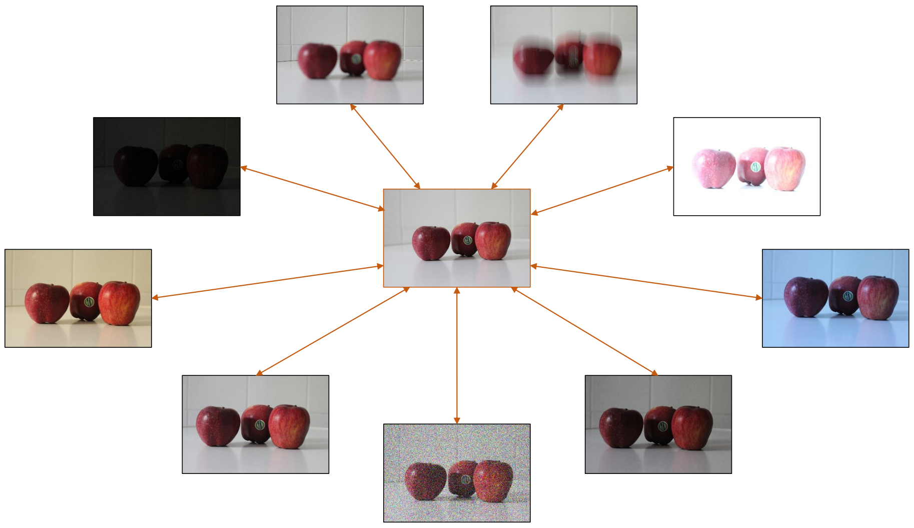


Figure 3.17: Strategy used to correlate multiple images.

2. **Processing times in keypoint detection**: By the graphic shown in Figure 3.7, SURF proved to be faster than SIFT. However even when using SURF, the processing times are prohibitive for an interactive tool, specially for high resolution images. A possible approach to enhance the image correlation times is to detect keypoints on a downsized version of the input images. The detected keypoints are then grouped into clusters. The regions of the downsized image that contain each cluster are then computed on the original image and only those regions are considered for the keypoint detection process. If the clusters are properly chosen, from the downsized image, the keypoints detected in the original image will be spread all over the original image, hopefully resulting in lower processing times.

3.2.2 Focus Detection

The second procedure of the proposed workflow consists in an analysis of the image focus. Two different approaches were implemented in this procedure. The first one allows an analysis of focus over the whole image. In order to raise the criterion, an analysis over a region-of-interest (ROI) is also possible.

The image gradient can be a powerful tool for focus detection as the colour intensity of a focused image tends to be greater than that of an unfocused one and consequently its gradient has greater magnitude values. Thus the Sobel algorithm (see Section 2.4.1.1) was applied in order to compute the gradient of each image. This algorithm receives as input a normalized grayscale version of each image in order to make this procedure invariant to lightness variations. The normalization was performed using the histogram equalization operation (see Section 3.2.1.1).

The Sobel algorithm was performed using the OpenCV function:

```
void Sobel(InputArray src, OutputArray dst, int ddepth, int dx, int dy, int ksize=3,
    double scale=1, double delta=0, int borderType=BORDER_DEFAULT)
```

3.2.2.1 Global Analysis of Focus

This approach aims at presenting a quick and intuitive way to detect the focused regions of each image. Such is achieved based on the image gradient and some user input.

The approach overview can be seen in Figure 3.18.

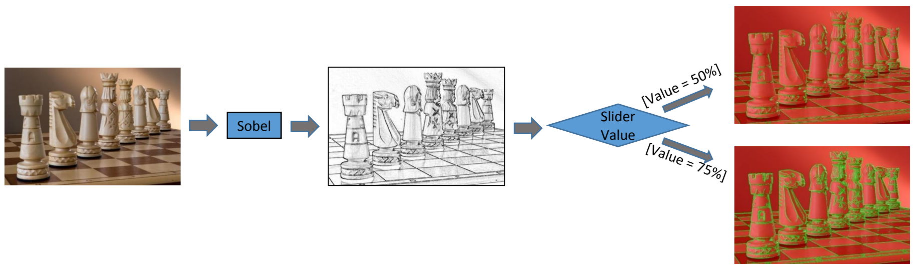


Figure 3.18: Approach overview in focus detection.

In Figure 3.19 can be seen two differently focused images. Figure 3.19(a) has a deep depth of field, while Figure 3.19(b) has a shallow depth of field. They represent the illustrative input images being the Sobel algorithm applied to each of them.

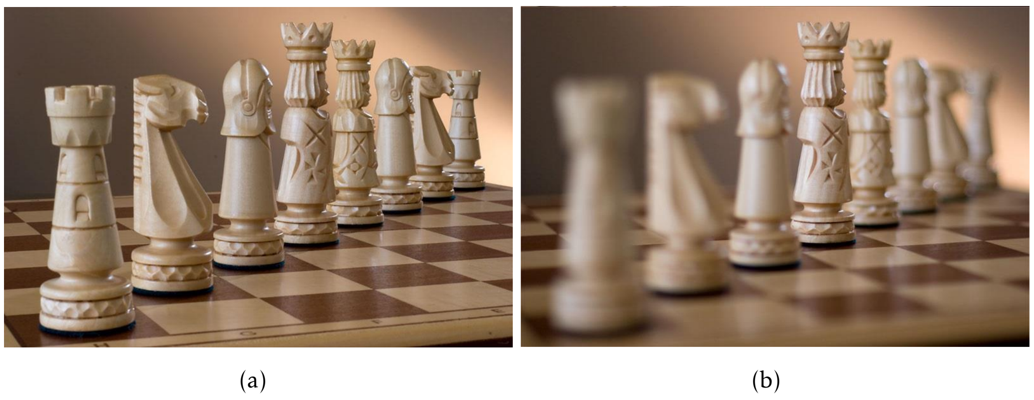


Figure 3.19: Differently focused images[1].

The output of the Sobel algorithm is a grayscale image where regions of notorious intensity changes, namely edges, are detected. The result can be seen in Figure 3.20. Figure 3.20(a) and 3.20(b) represent the output of the Sobel algorithm when applied to Figure 3.19(a) and 3.19(b), respectively.

The intensity of edges in focused regions are clearly stronger than those in unfocused ones. Having this, a slider will be offered to the user, having values that vary between 0

[1]Source: http://teachers.sduhsd.net/delliott/Files-Photo/deep%20vs%20shallow%20DOFchess%20pieces.JPG

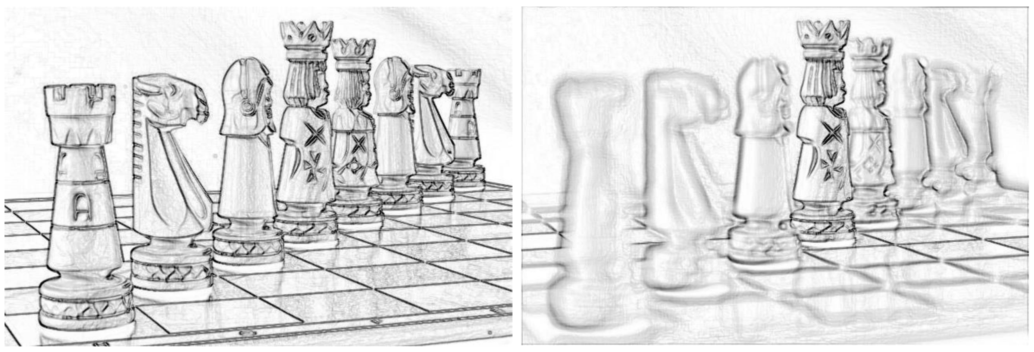

(a) (b)

Figure 3.20: Sobel output (inverted colours).

and 255. A green mask is put over the pixels whose intensities are greater than the value of the slider. Pixels whose intensity fall bellow the slider value have a red mask on. The effect can be seen if Figure 3.21.

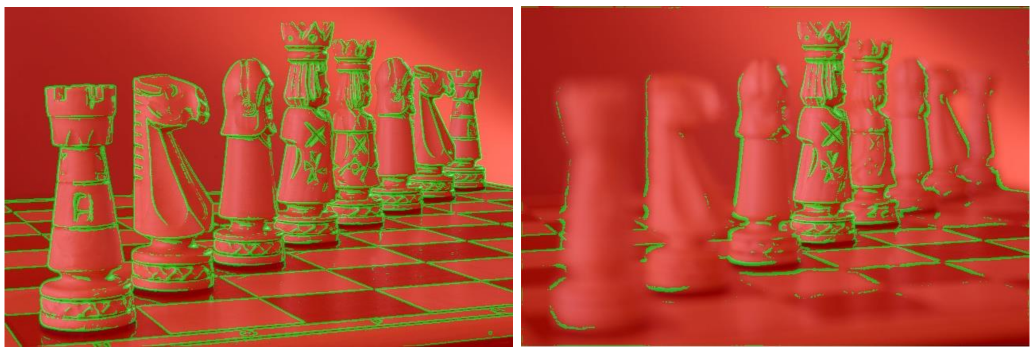

(a) (b)

Figure 3.21: Proceeding result for different slider values.

3.2.2.2 Analysis Over a Region of Interest

It may occur that the images to be tested are very similar in term of focus, which makes the aforementioned approach ineffective. Thus, the user may also choose a polygonal ROI, in the reference image, which is automatically computed in the remaining via homography. The output is the image in which the selected ROI is sharper. Figure 3.22 shows an illustrative example of ROI selection. The top image is the reference image. The selected ROI is represented by a red polygon in each image. For each ROI in each image the bounding box is computed and then the Sobel algorithm is applied to that bounding box as shown in Figure 3.23.

In order to analyse only the pixels that lie inside the polygon, the OpenCV function `pointPolygonTest` was applied:

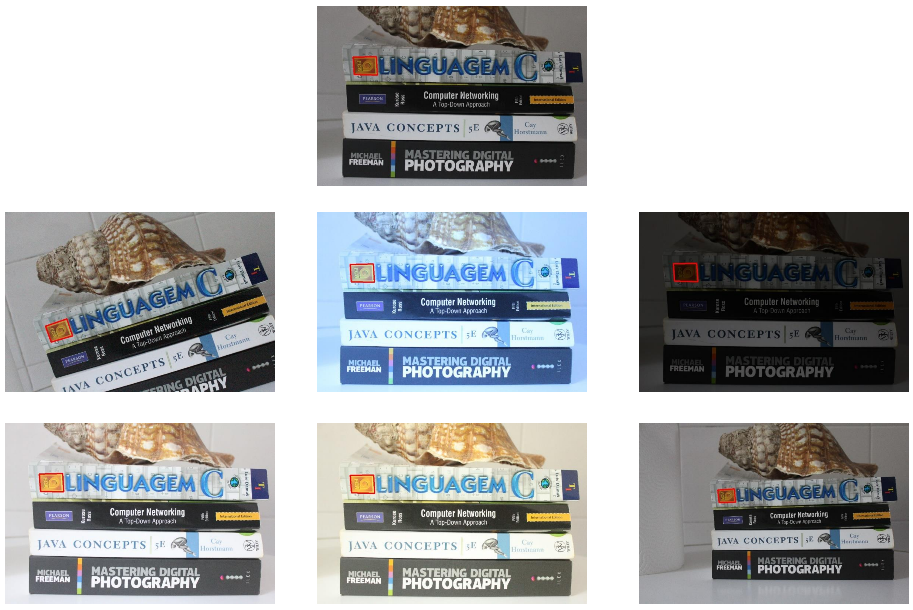


Figure 3.22: ROI selection.

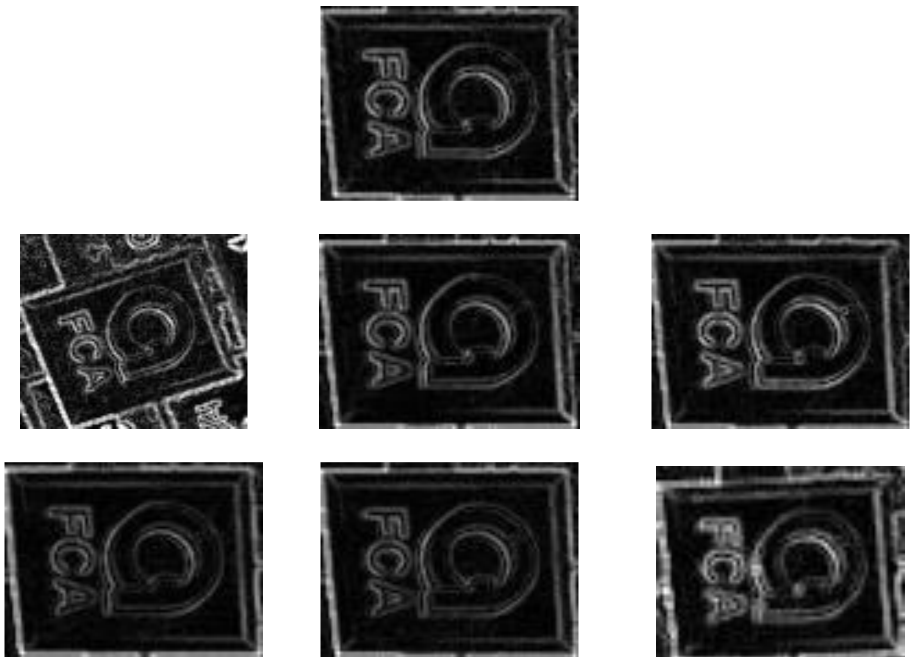


Figure 3.23: ROIs obtained.

```
double pointPolygonTest(InputArray contour, Point2f pt, bool measureDist)
```

This function determines whether the point pt is inside a contour, outside, or lies on an edge (or coincides with a vertex). It returns positive (inside), negative (outside), or zero (on an edge) value, correspondingly [Its].

The segmentation result can be seen in Figure 3.24, where the inside pixels are represented in white. For each ROI the intensity mean value of the inside pixels is computed. The larger is the value, the sharper is the respective ROI.

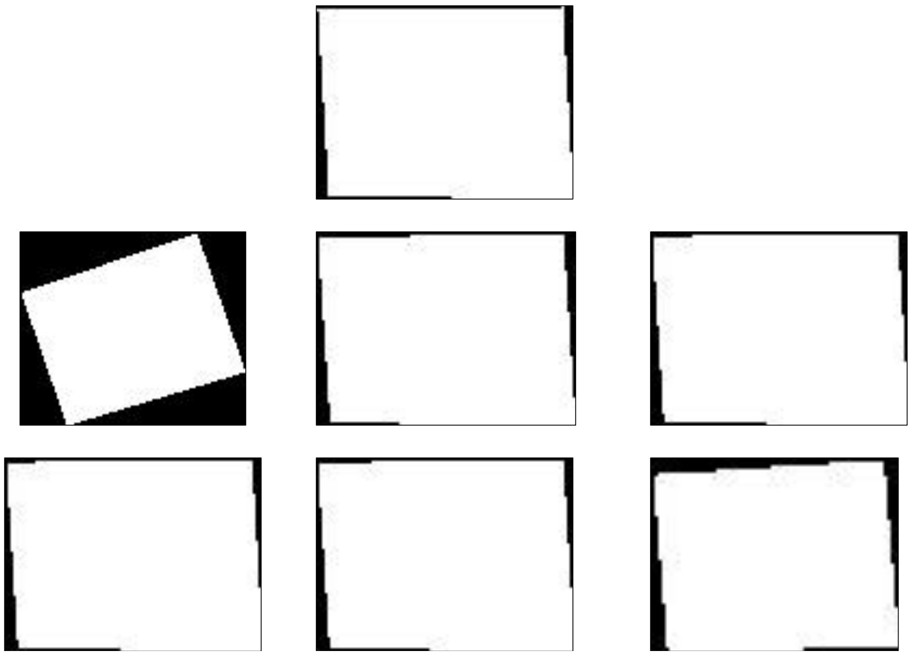


Figure 3.24: Segmentation result: pixels inside the polygons are represented in white colour.

3.2.2.3 Discussion & Evaluation

We expect that the two implemented approaches to focus analysis will give enough safety and comfort for the user to reject the images whose focus does not satisfy his/her quality standards.

As stated in the beginning of Section 3.2.2, the focus analysis is based on the image gradient. Given that the input images may vary on factors such as luminance, noise and temperature, several tests were conducted in order to verify the effect of those parameters on the image gradient. Figure 3.25 shows a graphic containing the result of this analysis. The horizontal axis represents the gradient value. The vertical axis represents the number of megapixels whose intensities are higher than the gradient value. The values composing this graphic can be seen in Table B.1 of Appendix B.

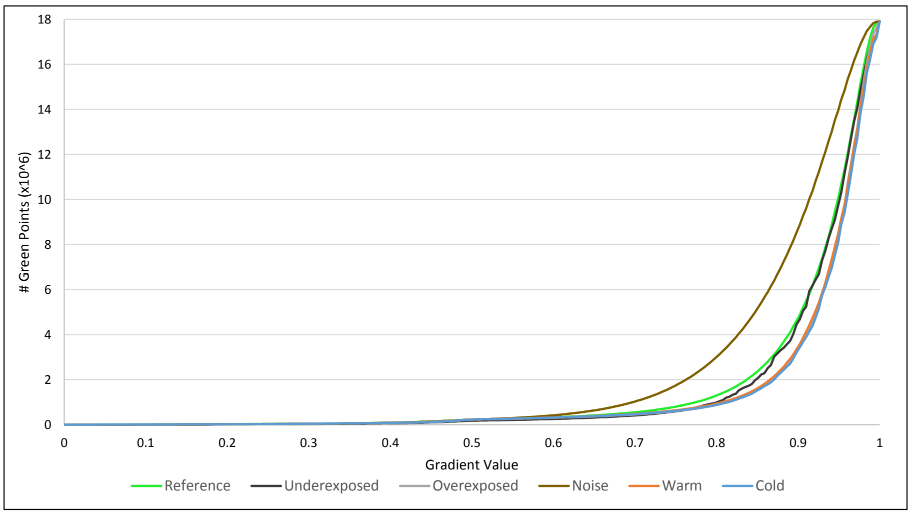


Figure 3.25: Impact of the image technical characteristics on the gradient.

Images that are affected by an uneven exposure (underexposed or overexposed) or an unrealistic colour cast (*warm* or *cold*) have a similar behaviour to the reference image, i.e., the image containing the *ideal* exposure. However noise proves to affect the image gradient significantly. That is because the colour of noisy pixels is unrelated to the remaining pixels of the image, causing a greater variation of colours and thus greater gradient magnitudes. In practice a noisy image can be detected as being the most focused one when in reality is not. Therefore, it will be necessary to create a mathematical model that takes into account the level of noise in each image (possibly using the result of step three of the workflow) and use this mathematical model as a leveller to provide better results when in face of noisy images.

The execution time of the focus detection procedure equals the execution time of the algorithm Sobel. In order to analyse the execution time of Sobel, several experiments were conducted over four different images at different resolutions. These images can be found in Figure B.2 of Appendix B. The execution times are shown in the graphic present in Figure 3.26.

For each resolution, the execution time is similar regardless of the image characteristics and its resolution. The values composing the graphic are available in Table B.2 of Appendix B, varying from 15 to 330 milliseconds, which we considered to be satisfactory and rewarding for the user experience.

3.2.3 Motion Blur Detection

The third procedure of the proposed workflow consists in the detection of motion blur. This procedure is not fully implemented. Our approach is based on the fact that motion blur creates smoothness on the direction of motion, originating lines/edges in that direction. An illustrative example of the motion blur effect can be seen in Figure 3.27. The bus

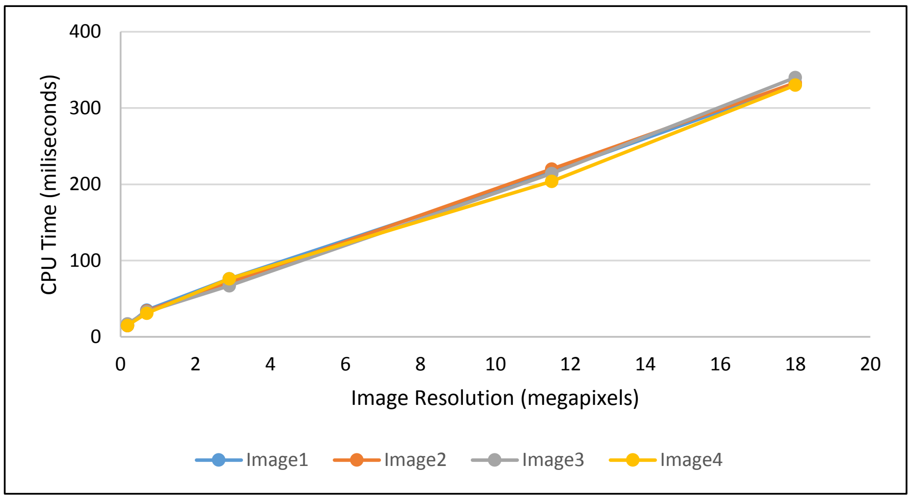


Figure 3.26: Sobel execution time (milliseconds) on different image resolutions.

movement created lines/edges in the direction of motion.

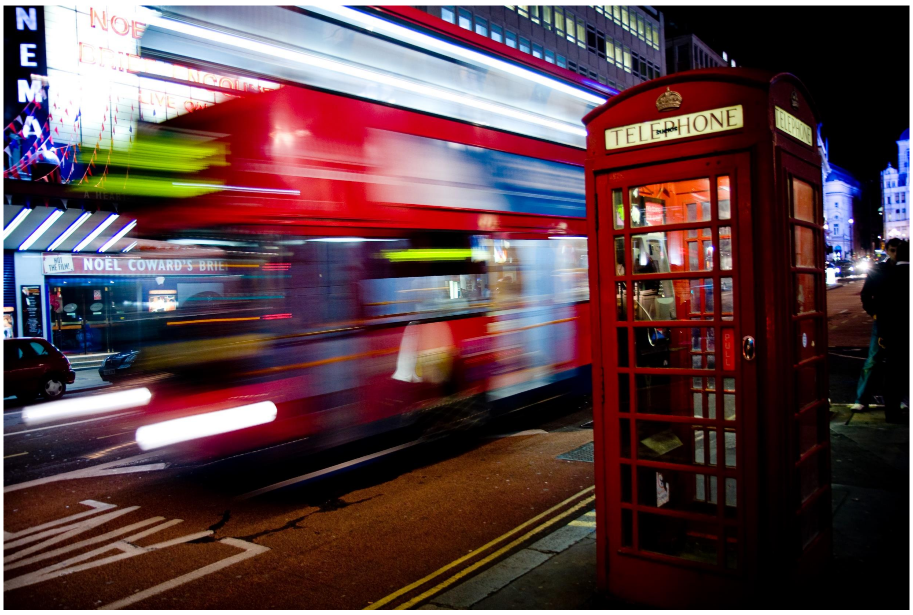


Figure 3.27: Motion blur example[2].

Thus our approach consisted in obtaining the image edges (see Figure 3.28) and then analysing the direction of each edge detected. If a large number of edges contain the same direction, there is a high probability that the image is affected by motion blur.

Given that motion blur may be local, only a specific number of regions may be affected by motion blur. Thus each image is divided into NxN regions. The result is shown in

[2]Source: `https://upload.wikimedia.org/wikipedia/commons/2/26/London_bus_and_telephone_box_on_Haymarket.jpg`

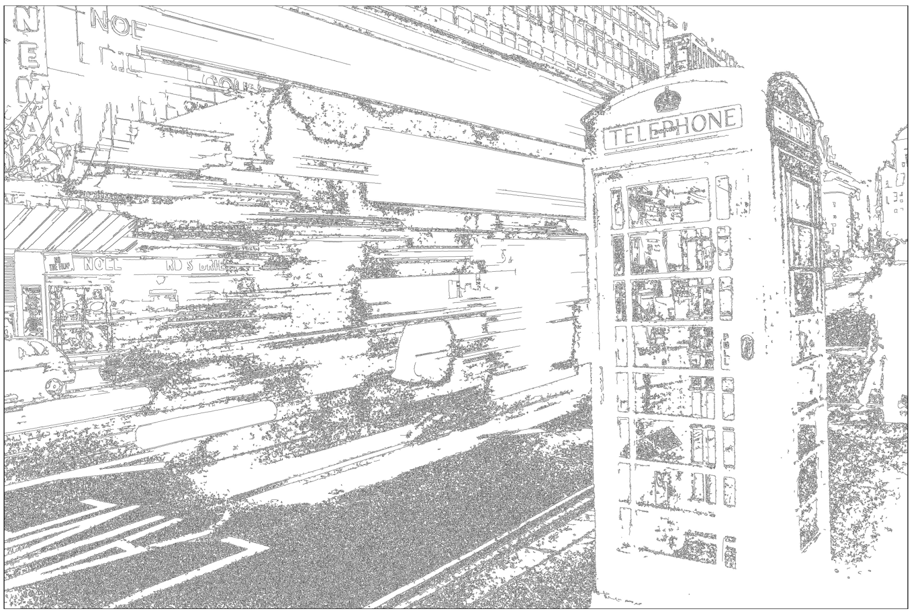


Figure 3.28: Edges detected in Figure 3.27 (inverted colours).

Figure 3.29.

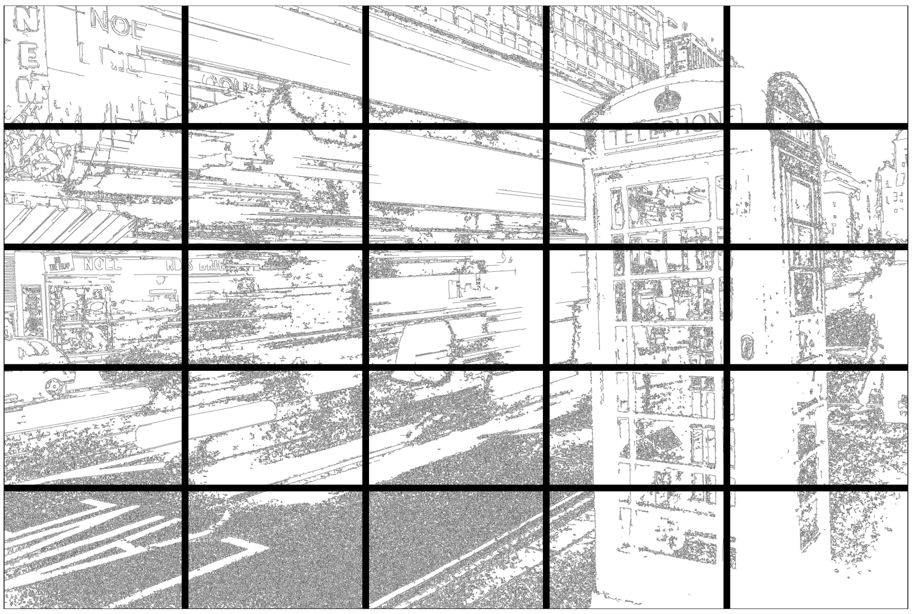
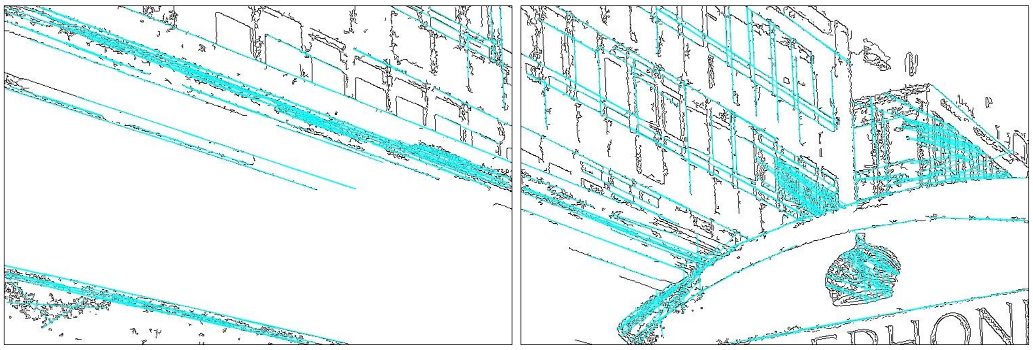


Figure 3.29: Image divided into NxN regions.

The Hough Transform [VC62] is widely used for the detection of specific shapes in an image, such as lines or circles. Therefore, it was used to detect lines caused by motion blur and computed its directions. The implemented version of the Hough Transform, implemented on Opencv [Its] is defined by the following method:

```
void HoughLines(InputArray image, OutputArray lines, double rho, double theta, int
    threshold, double srn=0, double stn=0)
```

where *lines* is a vector containing the detected lines. Each line is represent in polar coordinates by a two element vector (ρ, θ) where ρ is the line distance from the coordinate origin (0,0) and θ is the line rotation angle in radians [Its]. Thus the Hough Transform is applied to each of the computed image regions. The result of Hough Transform, for two different regions, can be seen in Figure 3.31, where the detected lines are marked in blue.



(a) Motion blur. (b) No motion blur.

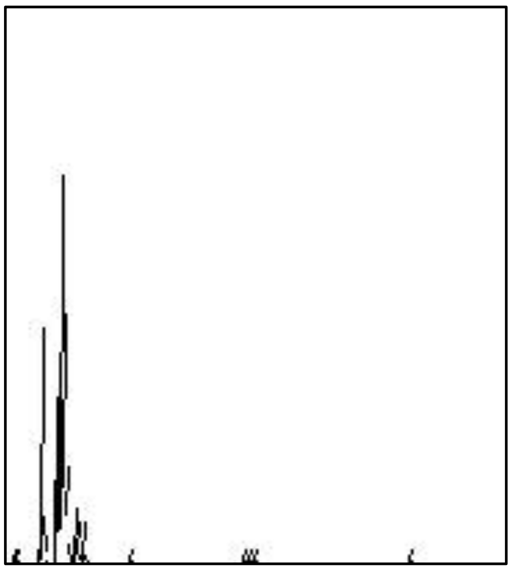
Figure 3.30: Result of the Hough Transform (inverted colours).

The region of the image shown in Figure 3.30(a) is affected by motion blur and thus a high number of the detected lines have the same orientation. Figure 3.30(b) is not affected by motion blur thus the variation of the orientation of the detected lines varies more than the ones detected in Figure 3.30(a). Hereupon based on the values of θ, i.e., the orientation of each detected line, is built a directional histogram. If a specific region contains motion blur, it will have a distinctive peak in the direction of motion. The directional histograms for Figures 3.30(a) and 3.30(b) can be seen in Figures 3.31(a) and 3.31(b), respectively.

3.2.3.1 Discussion

Due to time limitations, it was not possible to continue the implementation of this method. The next logical step is to analyse the distribution of the histograms in order to find those that contain a distinctive peak in a specific direction. In principle, a distinctive peak in a specific direction will be an indicator of the presence of motion blur.

One possible improvement to what has been implemented is to detect, in a previous procedure, which regions of the image are blurred. Thus, only the blurred regions are taken into account for the detection of motion blur. The detection of blurred regions will turn out to be insightful because, for instance, if a region of an image is sharply focused, containing a high number of equally oriented edges, there is the possibility that that region will be computed as containing motion blur when in reality it does not.


(a) Directional histogram of 3.30(a).

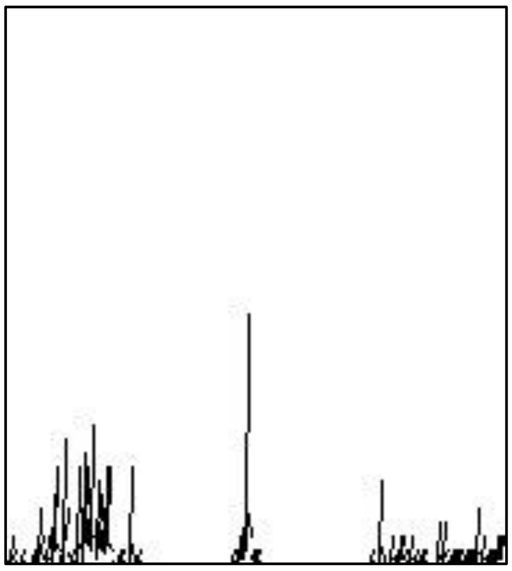

(b) Directional histogram of 3.30(b).

Figure 3.31: Directional histograms of Figures 3.30(a) and 3.30(b).

3.2.4 Noise Estimation

In order to estimate the level of noise in each image, two different approaches were implemented and evaluated. The first one is the filtering based method (see Section 2.5.1), the second one is based on the calculation of the entropy in the hue colour channel. Both approaches are described below.

3.2.4.1 Filtering-based Method

As referred in Section 2.5.1, the filtering-based method [NH12] consists of computing a smooth version of the original image, through a low-pass filter, followed by the calculation of the difference between the two versions. Given that the low-pass filters tend to attenuate or remove the noise, the standard deviation of intensity of the difference between images can be seen as an estimate of the level of noise.

Hereupon in order to implement the filtering-based approach, we used three functions available on Opencv [Its]: to obtain the smoothed version of the image, a mean filter was used through the function `blur`; then the difference between the two versions was computing by using the function `subtract`; for the calculation of the standard deviation of intensity, the function `meanStdDev` was used.

3.2.4.2 Entropy of the Hue Colour Channel

Our second approach consisted in the calculation of the entropy in the hue colour channel. Entropy can be understood as a measure of disorder in a system [DH13]. Noise can be seen as a set of pixels whose colour and brightness are unrelated to the subject, thus it is a variable of disorder on the image colours. Calculating the entropy of the hue colour channel per se does not present a real estimation of the level of noise in an image. For instance, a high contrast image, containing several colours, will present a high entropy in the hue colour channel. However as the level of noise raises so does the entropy, given that noise adds a greater variation of colours. Thus the initial goal of sorting the images according to their level of noise can be achieved.

The entropy equation can be defined as:

$$entropy = -\sum_{i=0}^{180} h_i * \log(h_i)$$

where h is a histogram containing the distribution of the hue colour component and i varies from 0 to 180.

It is important to refer that only the common regions between similar images must be analysed. For instance, if two photos are taken with a different focal distance, the appearance of different colours may occur, leading to a higher entropy value. As shown in Figure 3.23 of Section 3.2.2.2, the common regions between similar images may be obtained through the result of the image correlation procedure, i.e., the homography matrix.

3.2.4.3 Discussion & Evaluation

Figures 3.32(a), 3.32(b) and 3.32(c) show an illustrative example of three images that contain different levels of noise. These images were captured with the same exposure value ($EV = 0$), varying only the ISO value. Figures 3.32(d), 3.32(e) and 3.32(f) show the result of the subtraction when using the filtering-based method while Figures 3.32(g), 3.32(h) and 3.32(i) show the respective hue colour channel for each of the original images. As expected, both the standard deviation of intensity and the entropy of the hue colour channel are proportional to the value of the ISO.

In order to properly validate the proposed approaches to noise estimation and choose which one to use in this project, four sets of similar photos were taken in a total of 320 photos [And]. Naturally as the ISO value increases, the resulting noise must also increase. Figures 3.33(a) and 3.33(b) present the average results for the filtering-based and entropy-based methods, respectively.

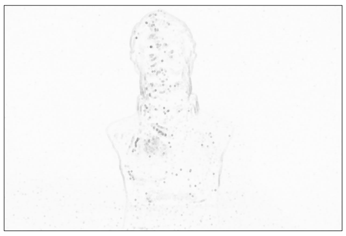
The values composing these graphics can be found in the tables of Appendix C. By analysing the graphics, we concluded that the entropy-based method is more reliable than the filtering-based method. This statement is based on the fact that, according to our experiments, the noise estimations are always proportional to the ISO values when using the entropy-based method. The results of the filtering-based method were proportional to the ISO values in 81.25% of the cases.

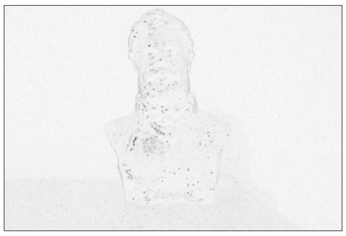
The previous evaluation was done over a set of natural images [And], taken with a digital single-lens reflex camera. As already stated, the entropy-based method proved

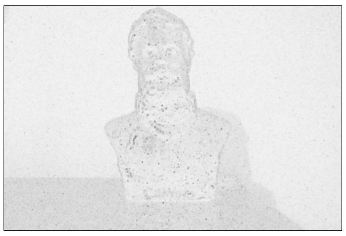
(a) ISO = 100.

(b) ISO = 3200.

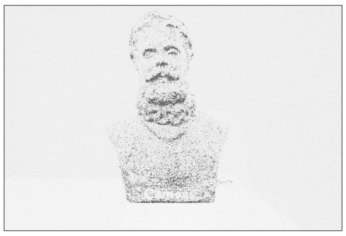
(c) ISO = 12800.


(d) Standard deviation = 1.147.


(e) Standard deviation = 2.471.


(f) Standard deviation = 4.47.

(g) Entropy = 2.967.


(h) Entropy = 6.99.

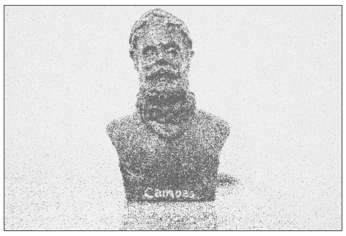
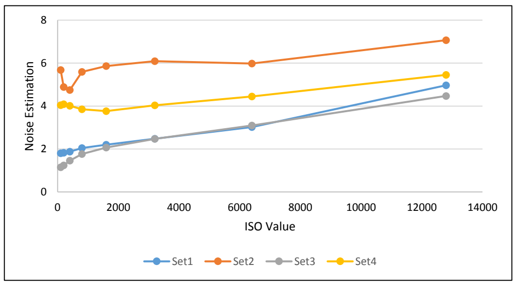
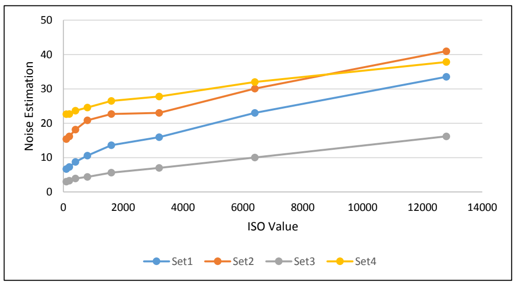

(i) Entropy = 16.234.

Figure 3.32: Three images taken with different ISO values and respective hue colour channels.



(a) Filtering-based method.



(b) Entropy-based method.

Figure 3.33: Noise estimation results.

to be a better option for noise estimation. However, if the image for testing contains a high number of colours, the existence of noise may lower the entropy of the hue colour channel. The reason being the fact that the colour of the noisy pixels may already exist in the image and thus the presence of noise makes the image colours more homogeneous. Figure 3.34 shows an artificial image, which contains 16.7 million colours.

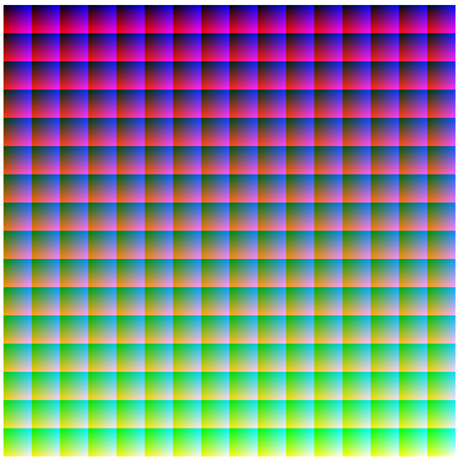
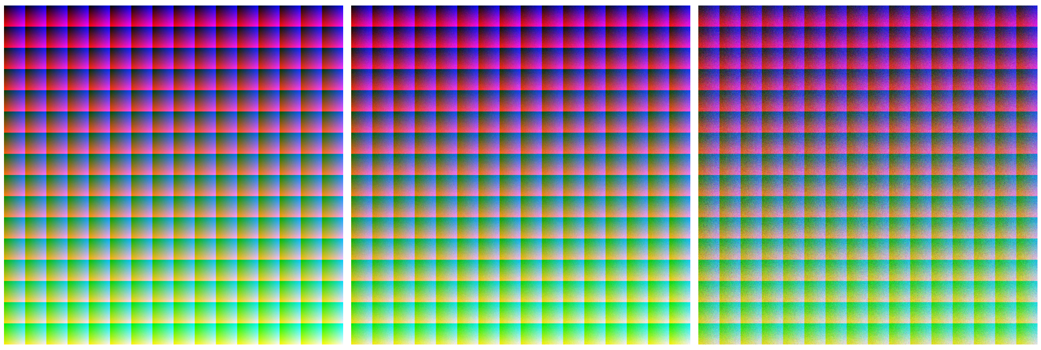


Figure 3.34: Image containing 16.7 million colours[3].

Synthetic noise, having different intensity values, was added to this image as shown in Figure 3.35.



(a) Intensity = 20%. (b) Intensity = 40%. (c) Intensity = 60%.

Figure 3.35: Synthetic noise added to Figure 3.34.

The entropy-based method had an unrealistic result for this test, as shown in Table 3.1.

As the level of noise raised, the noise estimation did not. This proves that the entropy-based method may fail when in face of images that contain a high number of colours. Even so, it is expected that in most cases, natural images does not contain such a high number

[3]Source: `http://morris-photographics.com/photoshop/articles/images/png-format/16million-pschmidt.png`

Table 3.1: Entropy-based method results for Figure 3.34.

Image	Noise Estimation
Figure 3.34	57.64
Figure 3.35(a)	54.75
Figure 3.35(b)	59.07
Figure 3.35(c)	44.95

of colours. Thus we consider that the entropy-based method is still the best option for noise estimation.

Figure 3.36 shows a graphic containing the processing times for the entropy-based and the filtering-based methods when applied to different image resolutions.

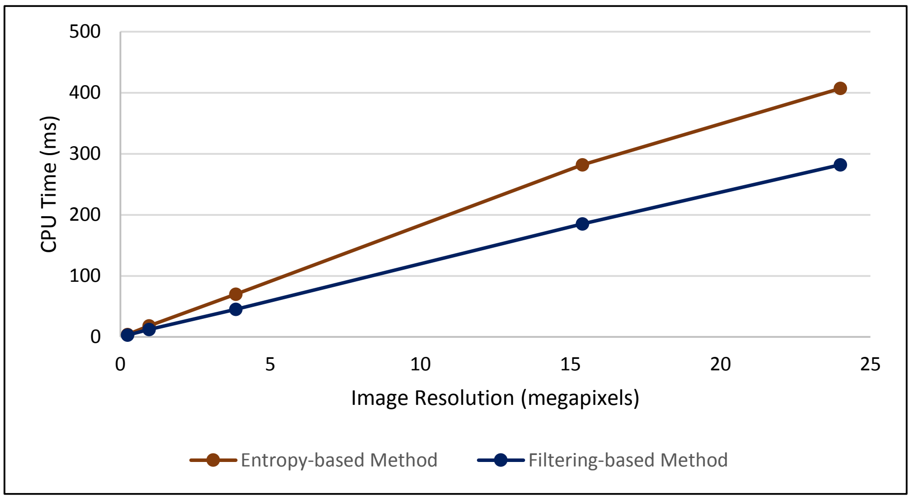


Figure 3.36: Processing times (ms) for entropy-based and filtering-based methods.

As the image resolution raises so does the processing times required for noise estimation in an image. The filtering-based method presents times that go from 3 to 282 milliseconds. The entropy-based method presents slightly higher processing times, varying from 4 to 407 milliseconds. Even for high resolution images, both methods present satisfactory processing times for an interactive tool. Thus, given that the entropy-based method presents better results in terms of noise estimation, we decided to use the entropy-based method for the procedure of noise estimation in this project.

3.2.5 Colour Analysis

The colours present in an image may offer insightful information from a technical point of view, i.e., information relating to its exposure. An underexposed image tends to have darker colour tones while an overexposed one tends to have brighter colour tones. On the issue of image temperature, those that contain an inadequate white balance (explained in

Section 2.1.9) are normally dominated by *warm* colours, like red or orange, or *cold* colours, such as blue or violet.

Section 3.2.5.1 presents our approach to sort the image according to their exposure level (from darker to brighter). The distinction between *warm* and *cold* images is still unimplemented. Nevertheless, in Section 3.2.5.2 is presented a possible approach to differentiate images according to their temperature.

3.2.5.1 Exposure Level

Based on the principle that underexposed images tend to have darker colour tones and overexposed images tend to have brighter colour tones, the sorting of the images by their exposure level became relatively easy. As shown in Figure 3.37, the V component of the HSV (**H**ue, **S**aturation, **V**alue) colour space, defines the brightness of colours in an image.

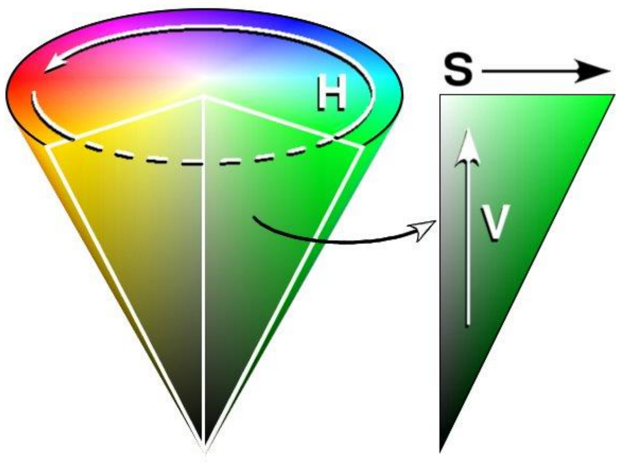


Figure 3.37: HSV cone[4].

Our approach to sort the images by their level of exposure is based on the average brightness of the image. The mean value of the V component, i.e., the average brightness of the image defines its exposure level. In the version of the HSV color space, available in OpenCV [Its], the V component reaches values from 0 to 255, where 0 represents an totally dark image and 255 represents an image entirely bright.

Figure 3.38 shows an illustrative example of the result of this method. Figures 3.38(a), 3.38(b) and 3.38(c) contain an underexposure, a balanced exposure and an overexposure, respectively. As expected the average brightness raises as the image becomes brighter.

[4]Source: `https://upload.wikimedia.org/wikipedia/commons/f/f1/HSV_cone.jpg`

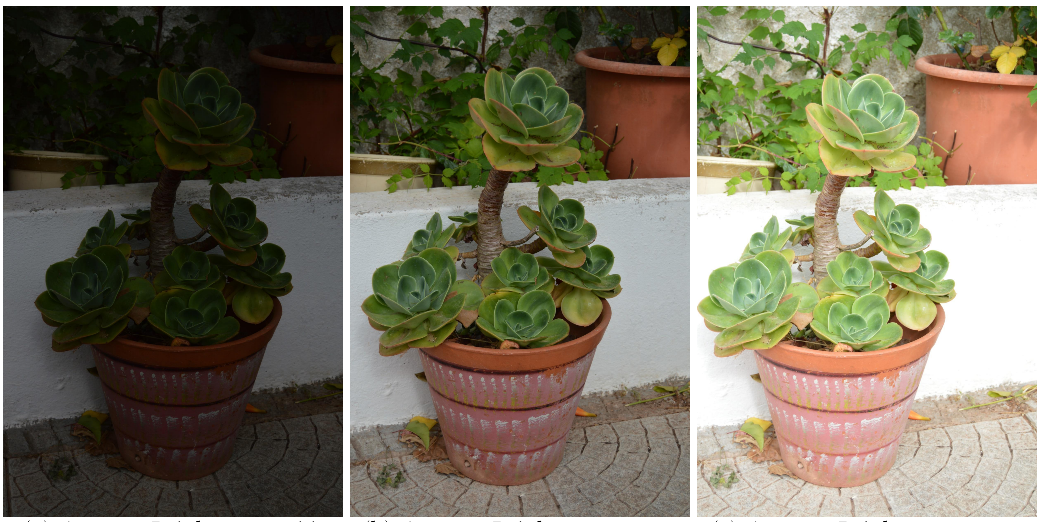

(a) Average Brightness = 44. (b) Average Brightness = 113. (c) Average Brightness = 198.

Figure 3.38: Brightness values for different level of exposure.

3.2.5.2 Image Colour Cast

The last procedure of the workflow consists in an analysis over the colour cast of the images. As stated in Section 1.2.5 the human brain has the ability to perceive white colour in scenes under different types of illumination, however digital cameras tend to capture the primitive characteristics of light. Thus *warm* and *cold* images may be captured. Figure 3.39 shows three similar images containing different white balance settings. Figure 3.39(a) is *cold*, Figure 3.39(b) presents a correct white balance while Figure 3.39(c) is *warm*.

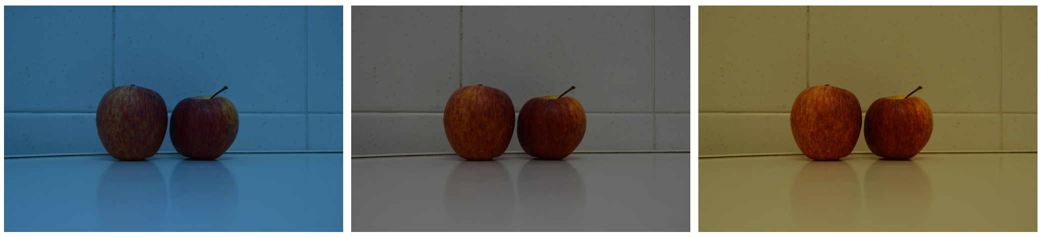

(a) *Cold* image. (b) Correct white balance. (c) *Warm* image.

Figure 3.39: Similar images containing different white balance settings.

Our approach to detect images that contain an unrealistic colour cast passes by asking the user to select a white point in one of the images from the set of similar images. According to the result of the image correlation procedure, the corresponding point is automatically detected in the remaining images, as exemplified in Figure 3.40.

White points are characterized for having low saturation values (≈ 0) as shown in Figure 3.37. However, *cold* and *warm* colours, such as blue and red respectively, tend to have larger saturation values. If the saturation value of the point selected by the user is above a determined threshold then the image is considered to have an unrealistic colour

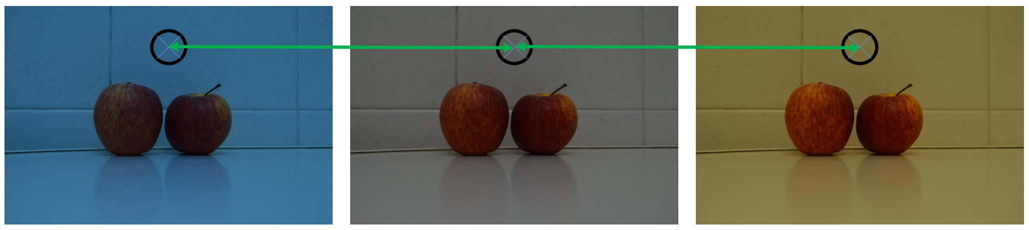


Figure 3.40: Respective points automatically detected in different images.

cast. Thus the next logical step is to identify whether the image is *cold* or *warm*. As shown in Figure 3.41, *cold* and *warm* colours are contiguous in the circle of the hue component.

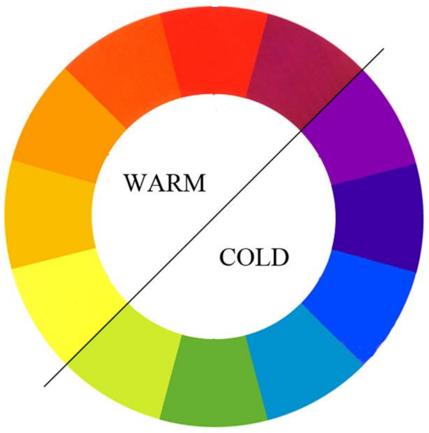


Figure 3.41: *Cold* and *warm* colours[5].

Therefore by analysing the hue component of the selected point, the distinction between *cold* and *warm* images will be achieved. The values for the saturation and hue colour components, referring the illustrative examples shown in Figure 3.39 can be seen in Table 3.2.

Table 3.2: Saturation and hue values for different white balance settings.

Image	Saturation (0-255)	Hue (0º-360º)
Figure 3.39(a) (*cold*)	121	144
Figure 3.39(b)	5	54
Figure 3.39(c) (*warm*)	85	33

As expected the saturation value of the *cold* and *warm* images is significantly higher than that of the image containing a correct white balance. The hue valued 144 corresponds to a *cold* colour, next to blue, while the hue valued 33 corresponds to a *warm* colour next to yellow.

[5]Source: `https://houseplansllcblog.files.wordpress.com/2013/06/color-wheel-warm-cold1.`

3.2.5.3 Discussion & Evaluation

The procedure that analyses each image colour cast was not implemented. However, it is the final phase of the workflow, thus the image correlation and the hue colour space of each image are already computed. Consequently, this procedure is expected to have low processing times. Even so, further evaluation must be done in order to validate the proposed approach.

As for the procedure of exposure level analysis, our approach is based on the average brightness value of the images colours. By using this approach we expected the processing times to be low. The graphic shown in Figure 3.42 proves that the processing times go as up as 170 milliseconds for a 24 megapixels image.

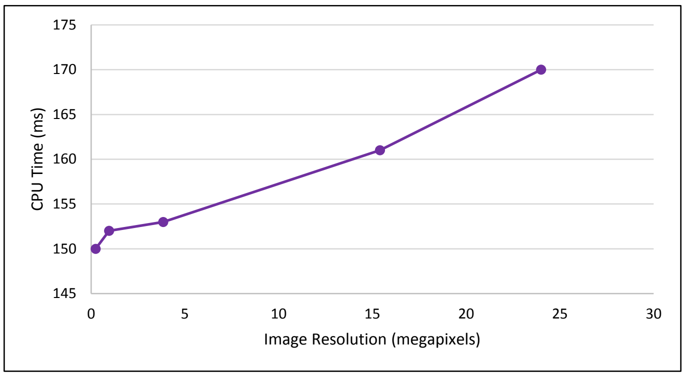


Figure 3.42: Processing times (ms) for exposure level estimation.

Once again we found the processing times of this procedure to be satisfactory for an interactive tool. The values composing the graphic shown in Figure 3.42 can be found in Table D.1 of Appendix D.

3.3 Discussion

Throughout this section the methodology used to implement each functionality of the workflow was described. For the image correlation procedure we followed an approach that proved to be invariant to scale, orientation and illumination changes. However if there is a high variation in terms of focus between the images contained in the set, this approach becomes less effective. The second weakness is related to the processing times as they tend to be prohibitive for an interactive tool, specially when in face of high resolution images. The reason being the fact that SURF [Bay+08] requires heavy processing in order to be invariant to scale and rotation changes.

For focus analysis, a simple approach, based on an analysis over the edges of the image, was implemented. The higher is their intensity the more focused is the image. The detection of the image edges proved to be fast, which enhances the user experience. In

png

Section 2.3, a few algorithms on focus detection were described. However, we suspect that those algorithms require a heavier processing than the one implemented in this dissertation. Consequently they were not taken into further consideration.

The procedure of motion blur detection is not fully implemented. Its fully implementation and evaluation will be done in the future. If the proposed approach proves not to be ideal for an interactive software tool, namely in terms of processing times and accurate results, two alternative approaches [Liu+08; Su+11] are described in Section 2.4.2.

In order to sort the images according to their level of noise, the entropy-based method proved to be more effective than the filtering-based method [NH12]. It is important to refer that the entropy-based approach cannot be used as a non-reference method for noise estimation as it is performed over the hue colour channel. However when in face of a set of similar images, the same colours are expected to be in all images. Thus the increasing of the entropy value, between images, is caused by noise.

The average of the image brightness was computed in order to sort the images according to their exposure level. A possible improvement to this method would be to threshold the set of average values in order to suggest which images are underexposed and overexposed.

The programming code related to the implemented algorithms can be found in Appendix E.

Chapter 4

Conclusions & Future Work

4.1 Conclusions

This dissertation introduced a novel workflow that aims at helping the photographer to choose the best photo from a set of similar photos. The implemented algorithms will be used and combined into a unique interactive software tool.

According to our experiments, the implemented algorithms will be able to successfully analyse and order the set of similar images according to their technical characteristics (focus, noise, exposure level). In a world where everything can be photographed, it would be unwise to state that the implemented algorithms are "bulletproof". There may be unexpected exposures that may lead to unexpected results. As this work evolves so will the evaluation of each algorithm, not only by the authors but also by the community of image processing, as these algorithms are widely used. Therefore the need to improve them may come up. Our main concern, above all, was the user satisfaction. Apart from the image correlation procedure, all other procedures present low and satisfactory processing times. Thus selecting the best photo will turn out to be fast and hopefully effective. All the challenges that this process imposes will be easily overcome. A possible approach to lower the processing times of the image correlation procedure is proposed in Section 3.2.1.6.

It is important to refer that the final decision always belongs to the user. It is up to the user to know, in each phase of the workflow, when to keep the photos that are *ideal* from a technical point of view or when to break the rules and make a daring selection.

There are still many improvements that could be implemented, which is the subject of the following section.

4.2 Future Work

For future improvements, the initial step will be to implement and evaluate the remaining algorithms. Namely the algorithm for motion blur detection and the one that analyses the image colour cast. A possible approach detect *warm* and *cold* images is described in Section 3.2.5.2. Thus the core of the interactive software tool will be implemented.

The next logical step is to build the user interface. OpenCV [Its] was used for image processing. However the implementation of a graphical user interface is not possible using OpenCV solely. OpenFrameworks [Lie+] and QT [Com] are widely used frameworks for developing software applications. These frameworks have the main advantage of being configurable alongside with OpenCV.

Afterwards, at a later stage, applying machine learning techniques will also be insightful. Such techniques will learn the preferences of the user and thus an automatic suggestion of the best photo will be possible.

In a final note, in order to make this software tool more embracing, the possibility for the user to upload an entire album will also be interesting. The results of the image correlation procedure and possible timestamp information, which can be obtained through the EXIF [JEI02] specification, may be the base to automatically detect sets of similar photos in an album.

Bibliography

[Ado] Adobe Systems Inc. *Adobe Photoshop Hits Twenty*. URL: https://goo.gl/1s3JWA (visited on 12/14/2014).

[Alv+15] A. Alves, F. P. Birra, and J. M. Lourenço. “Como separar o trigo do joio? Ou: Como selecionar a melhor fotografia de um conjunto de fotografias semelhantes”. In: *Proceedings of INForum Simpósio de Informática*. INForum 2015. Covilhã, Portugal, 2015.

[And] André Alves. *Noise Tests*. URL: https://goo.gl/6SJx9A (visited on 09/23/2015).

[Bag+12] D. L. Baggio, S. Emami, D. M. Escriva, K. Ievgen, N. Mahmood, J. Saragih, and R. Shilkrot. *Mastering OpenCV with Practical Computer Vision Projects*. Packt Publishing, Limited, 2012. ISBN: 9781849517829.

[Bay+08] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. “Speeded-Up Robust Features (SURF)”. In: *Comput. Vis. Image Underst.* 110.3 (June 2008), pp. 346–359. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2007.09.014.

[BK08] D. G. R. Bradski and A. Kaehler. *Learning Opencv, 1st Edition*. First. O’Reilly Media, Inc., 2008. ISBN: 9780596516130.

[Chu+07] W.-T. Chu, J.-C. Chen, and J.-L. Wu. “Tiling Slideshow: An Audiovisual Presentation Method for Consumer Photos”. In: *MultiMedia, IEEE* 14.3 (2007), pp. 36–45. ISSN: 1070-986X. DOI: 10.1109/MMUL.2007.66.

[Com] Q. Company. *Qt*. URL: http://www.qt.io/ (visited on 09/20/2015).

[Cur04] D. Curtin. *The Textbook of Digital Photography*. ShortCourses.com, 2004. ISBN: 9781928873600. URL: http://books.google.pt/books?id=UNDIAAAACAAJ.

[DH13] J. W. David Halliday Robert Resnick. *Fundamentals of Physics Extended*. 10th Edition. Wiley, 2013. ISBN: 9781118230725.

[Der10] K. G. Derpanis. “Overview of the RANSAC Algorithm”. In: *York University, Toronto, Canada* (2010).

[Dub09] E. Dubrofsky. *Homography Estimation*. 2009.

[FB81] M. A. Fischler and R. C. Bolles. “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”. In: *Commun. ACM* 24.6 (June 1981), pp. 381–395. ISSN: 0001-0782. DOI: 10.1145/358669.358692.

[Foi+06] A. Foi, V. Katkovnik, and K. Egiazarian. "Pointwise shape-adaptive DCT for high-quality deblocking of compressed color images". In: *European Signal Processing Conference* 16.5 (2006), pp. 1–17. ISSN: 22195491.

[FF78] J. Fourier and A. Freeman. *The Analytical Theory of Heat*. The University Press, 1878. URL: http://books.google.pt/books?id=No8IAAAAMAAJ.

[Fre08] M. Freeman. *Mastering Digital Photography*. First. Alastair Campbell, 2008. ISBN: 978-981-245-713-4.

[FS95] Y. Freund and R. E. Schapire. *A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting*. 1995.

[GW01] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. 2nd. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2001. ISBN: 0201180758.

[Goo] Google. *Google+*. URL: https://plus.google.com/ (visited on 06/24/2015).

[Gri] A. Griffith. *Crunching the Numbers: Four Insights We Can Glean from Camera Sales Data*. URL: http://petapixel.com/2013/12/18/crunching-numbers-4-insights-camera-sales-data/ (visited on 12/14/2014).

[Inca] C. B. Inc. *Camera Bits*. URL: http://www.camerabits.com/ (visited on 06/24/2015).

[Incb] C. U. Inc. *Cannon EOS 7D Mark II*. URL: http://www.usa.canon.com/cusa/consumer/products/cameras/slr_cameras/eos_7d_mark_ii (visited on 01/05/2015).

[Incc] A. S. Incorporated. *Adobe Systems*. URL: http://www.adobe.com (visited on 06/24/2015).

[Its] Itseez. *OpenCV*. URL: http://opencv.org/ (visited on 12/15/2014).

[JEI02] JEITA. *Exchangeable image file format for digital still cameras*. 2002. URL: http://www.sno.phy.queensu.ca/~phil/exiftool/TagNames/EXIF.html.

[Laba] D. Labs. *DxO Optics Pro*. URL: http://www.dxo.com/us/photography/photo-software/dxo-opticspro (visited on 10/09/2015).

[Labb] D. Labs. *Exclusive PRIME technology*. URL: http://www.dxo.com/us/photography/dxo-optics-pro/features/denoising (visited on 11/15/2015).

[LH89] J. Lee and K. Hoppel. "Noise Modeling and Estimation of Remotely-Sensed Images". In: *Geoscience and Remote Sensing Symposium, 1989. IGARSS'89. 12th Canadian Symposium on Remote Sensing., 1989 International*. Vol. 2. 1989, pp. 1005–1008. DOI: 10.1109/IGARSS.1989.579061.

[LY08] S. Li and B. Yang. "Multifocus Image Fusion Using Region Segmentation and Spatial Frequency". In: *Image Vision Comput.* 26.7 (July 2008), pp. 971–979. ISSN: 0262-8856. DOI: 10.1016/j.imavis.2007.10.012. URL: http://dx.doi.org/10.1016/j.imavis.2007.10.012.

[Lie+] Z. Lieberman, T. Watson, and A. Castro. *OpenFrameworks*. URL: http://openframeworks.cc/ (visited on 09/20/2015).

[Liu+08] R. Liu, Z. Li, and J. Jia. "Image partial blur detection and classification". In: *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), 24-26 June 2008, Anchorage, Alaska, USA*. 2008. DOI: 10.1109/CVPR.2008.4587465. URL: http://dx.doi.org/10.1109/CVPR.2008.4587465.

[Low04] D. G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In: *Int. J. Comput. Vision* 60.2 (Nov. 2004), pp. 91–110. ISSN: 0920-5691. DOI: 10.1023/B:VISI.0000029664.99615.94. URL: http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94.

[Lu+07] Y. Lu, X. Feng, J. Zhang, R. Wang, K. Zheng, and J. Kong. "A Multi-focus Image Fusion Based on Wavelet and Region Detection". In: 2007, pp. 294–298. DOI: 10.1109/EURCON.2007.4400459.

[ML09] M. Muja and D. G. Lowe. "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration". In: *International Conference on Computer Vision Theory and Application VISSAPP'09)*. INSTICC Press, 2009, pp. 331–340.

[NH12] T.-A. Nguyen and M.-C. Hong. "Filtering-based Noise Estimation for Denoising the Image Degraded by Gaussian Noise". In: *Proceedings of the 5th Pacific Rim Conference on Advances in Image and Video Technology - Volume Part II*. PSIVT'11. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 157–167. ISBN: 978-3-642-25345-4. DOI: 10.1007/978-3-642-25346-1_15. URL: http://dx.doi.org/10.1007/978-3-642-25346-1_15.

[One] P. One. *Capture One*. URL: https://www.phaseone.com/en/Imaging-Software/Capture-One.aspx? (visited on 10/09/2015).

[Per08] M. Peres. *The Concise Focal Encyclopedia of Photography: From the First Photo on Paper to the Digital Revolution*. Elsevier, 2008. ISBN: 9780240809984. URL: http://books.google.pt/books?id=vmZtFrHO7doC.

[Pot+09] E. Potapova, M. Egorova, and I. Safonov. "Automatic Photo Selection for Media and Entertainment Applications". In: *GRAPHICON-2009* (2009), pp. 117–124.

[Ros97] N. Rosenblum. *A world history of photography*. Abbeville Press, 1997. ISBN: 9780789203298. URL: http://books.google.pt/books?id=0I7rAAAAMAAJ.

[RD06] E. Rosten and T. Drummond. "Machine Learning for High-speed Corner Detection". In: *Proceedings of the 9th European Conference on Computer Vision - Volume Part I*. ECCV'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 430–443. ISBN: 3-540-33832-2, 978-3-540-33832-1. DOI: 10.1007/11744023_34.

[Rut06] C. Rutter. *The Essential Color Manual for Photographers*. A RotoVision book. RotoVision SA, 2006. ISBN: 9782940378111. URL: http://books.google.pt/books?id=QL5AAAAACAAJ.

[San13] J. Santos. *Fotografia: Luz, Exposição, Composição, Equipamento*. Seventh. Centro Atlântico, 2013. ISBN: 978-989-615-099-0.

[Shr12] G. T. Shrivakshan. "A Comparison of various Edge Detection Techniques used in Image Processing". In: 9.5 (2012), pp. 269–276.

[Su+11] B. Su, S. Lu, and C. L. Tan. "Blurred Image Region Detection and Classification". In: *Proceedings of the 19th ACM International Conference on Multimedia*. MM '11. New York, NY, USA: ACM, 2011, pp. 1397–1400. ISBN: 978-1-4503-0616-4. DOI: 10.1145/2072298.2072024. URL: http://doi.acm.org/10.1145/2072298.2072024.

[Ton+04] H. Tong, M. Li, H. Zhang, and C. Zhang. "Blur detection for digital images using wavelet transform". In: *Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on*. Vol. 1. IEEE. 2004, pp. 17–20.

[VC62] H. P. VC. *Method and means for recognizing complex patterns*. US Patent 3,069,654. 1962.

Appendix A

Image Correlation Results

This appendix presents the results obtained during the evaluation of the image correlation procedure. Namely, for the process of histogram equalization, keypoint detection and keypoint matching.

A.1 Histogram Equalization

The number of keypoints detected at several image resolutions for an *ideal* exposure can be seen in Table A.1. Table A.2 presents the same results for an underexposed image. The results after the operation of histogram equalization can be seen in Table A.3.

Table A.1: Results for a balanced exposure.

Image Resolution (megapixels)	# Keypoints Detected
0.18	1495
0.7	3575
2.9	6584
11.5	11340
18	15607

Table A.2: Results before histogram equalization (underexposed image).

Image Resolution (megapixels)	# Keypoints Detected
0.18	572
0.7	1153
2.9	1743
11.5	1855
18	1853

Table A.3: Results after histogram equalization (underexposed image).

Image Resolution (megapixels)	# Keypoints Detected
0.18	1599
0.7	3671
2.9	6414
11.5	8380
18	10374

The number of keypoints detected on the underexposed image (Table A.2) is significantly inferior when compared to the original image (Table A.1). The insight of the histogram equalization is to improve the contrast and brightness of the image by stretching out the intensity range. Thus the number of keypoints detected comes closer to those detected in a reference image, which can be proven by comparing Tables A.1 and A.3.

A.2 SIFT & SURF Comparison

The comparison between SIFT [Low04] and SURF [Bay+08] in terms of keypoints detected at several image resolutions can be seen in Table A.4. The comparison between SIFT and SURF in terms of processing times can be seen in Table A.5. These values are graphically shown and discussed in Section 3.2.1.2.

Table A.4: Comparison of SIFT and SURF in keypoint detection.

Image Resolution (megapixels)	# Keypoints using SIFT	# Keypoints using SURF
0.18	1495	1342
0.7	3575	3460
2.9	6584	7240
6.5	8724	10300
8.8	9944	11861
11.5	11340	13584
14.5	13302	15081
18	15607	16580

The implemented versions of SIFT and SURF in OpenCV [Its] are defined by the following constructors:

```
SIFT(int nfeatures=0, int nOctaveLayers=3, double contrastThreshold=0.04, double
    edgeThreshold=10, double sigma=1.6)
```

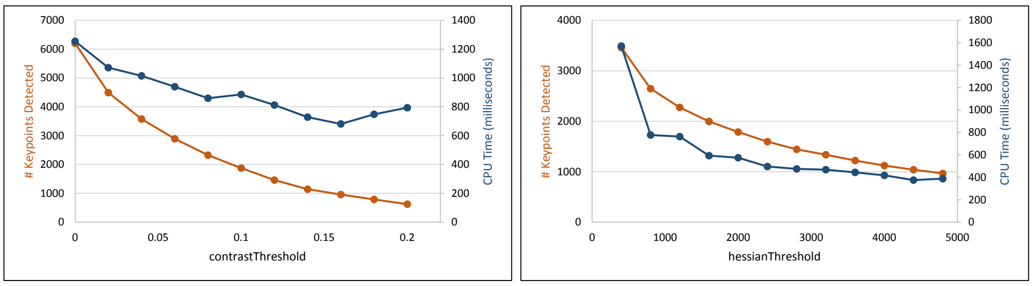
```
SURF(double hessianThreshold, int nOctaves=4, int nOctaveLayers=2, bool extended=true,
    bool upright=false)
```

The threshold values of SIFT and SURF, *contrastThreshold* and *hessianThreshold* respectively, are used, for instance, to filter out dubious points in low contrast regions. The

Table A.5: Comparison of SIFT and SURF in processing times.

Image Resolution (megapixels)	CPU Time using SIFT (ms)	CPU Time using SURF (ms)
0.18	259	150
0.7	745	644
2.9	2451	1697
6.5	4801	2968
8.8	6520	3891
11.5	8912	4630
14.5	11061	5500
18	14125	6564

graphic shown in Figure A.1 present the *contrastThreshold* and *hessianThreshold* impact in terms of number of keypoints detected and processing times (milliseconds). The values composing these graphics can be found in Tables A.6 and A.7.



(a) (b)

Figure A.1: Impact of *contrastThreshold* (SIFT) and *hessianThreshold* (SURF) on processing time (ms) and keypoint detection.

Naturally, as the threshold values increase fewer keypoints are detected. However SIFT proves to have higher processing times. Thus, as already referred in Section 3.2.1.2, SURF proves to be a better option for keypoint detection in an interactive software tool.

Nevertheless, even SURF proves to have high processing times (> 400 *milliseconds*). As described in Section 3.2.1.6, a possible approach to lower the processing times is to do a hierarchical keypoint detection. Our commitment is to find enough keypoints, spread out along the image, as fast as possible. Finding an *hessianThreshold* optical value may be a hard task as some image regions may be more homogeneous than others. A recursive call of the SURF algorithm, varying the *hessianThreshold* value, until enough keypoints are detected may be a possible approach to enhance the keypoint detection procedure.

A.3 FLANN

The results of *match* and *knnMatch* methods, available in FLANN [ML09] library, can be seen in Tables A.8 and A.9. These results are graphically shown and discussed in Section

Table A.6: SIFT: Impact of *contrastThreshold* on processing times (ms) and keypoint detection.

contrastThreshold	CPU Time (ms)	# Keypoints Detected
0	1254	6202
0.02	1072	4497
0.04	1015	3575
0.06	939	2894
0.08	860	2326
0.1	886	1876
0.12	813	1459
0.14	728	1145
0.16	681	955
0.18	748	789
0.2	794	624

Table A.7: SURF: Impact of *hessianThreshold* on processing times (ms) and keypoint detection.

hessianThreshold	CPU Time (ms)	# Keypoints Detected
400	3460	1571
800	2648	778
1200	2274	764
1600	1996	594
2000	1787	575
2400	1598	496
2800	1444	475
3200	1339	468
3600	1222	445
4000	1121	418
4400	1041	376
4800	967	388

3.2.1.3.

Table A.8: FLANN results using method `match`.

# Keypoints Img1	# Keypoints Img2	# Correct Matches	# False Matches	# CPU Time (ms)
1342	1313	26	1316	34
3460	3292	89	3371	81
7240	6891	276	6964	168
13584	11745	537	13047	338
16580	14020	580	16000	435

The processing times are similar for both methods (varying from 34 to 436 milliseconds). However the `knnMatch` method presents significantly better results in the number

Table A.9: FLANN results using method `knnmatch`.

# Keypoints Img1	# Keypoints Img2	# Correct Matches	# False Matches	# CPU Time (ms)
1342	1313	704	638	39
3460	3292	1920	1540	82
7240	6891	3631	3609	191
13584	11745	4730	8854	355
16580	14020	4739	11841	436

of matches computed as a higher number of matches was computed at very similar processing times.

Appendix B

Focus Detection Results

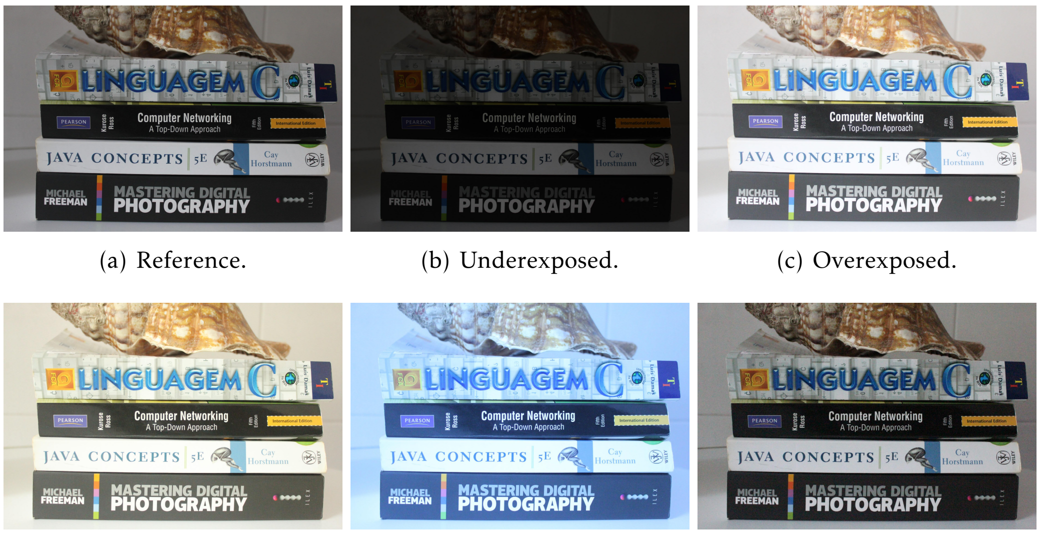
This appendix presents the results of the evaluation performed over the focus detection procedure.

Table B.1 presents the impact of technical characteristics (exposure level, noise and temperature) on the image gradient. The values under the last six columns represent the number of megapixels whose gradient value is higher the reference value (first column). The images used for testing the impact of the image technical characteristics on the gradient can be seen in Figure B.1.

Table B.1: Impact of the image technical characteristics on the gradient.

Gradient value	Reference	Underexposed	Overexposed	Noisy	Warm	Cold
1	17.915	17.915	17.915	17.915	17.915	17.915
0.9	4.866	4.672	3.455	8.868	3.573	3.433
0.8	1.297	0.989	0.875	3.007	0.943	0.889
0.7	0.561	0.422	0.462	1.05	0.492	0.480
0.6	0.330	0.258	0.309	0.421	0.322	0.320
0.5	0.233	0.182	0.228	0.231	0.236	0.235
0.4	0.094	0.067	0.080	0.089	0.083	0.082
0.3	0.049	0.035	0.043	0.040	0.045	0.045
0.2	0.026	0.017	0.021	0.018	0.023	0.023
0.1	0.013	0.007	0.010	0.007	0.011	0.011
0	0.006	0.002	0.003	0.002	0.004	0.004

For this particular experiment, it is notorious that as the gradient value reaches 0.6 the number of megapixels, in the images for testing, whose gradient value is higher than 0.6 increases exponentially (see Figure 3.25). However, this behaviour is especially notorious in the noisy image as the gradient value is significantly higher than that of the remaining images. That is because the colour of noisy pixels is unrelated to the remaining pixels of the image, causing a greater variation of colours and thus greater gradient magnitudes. In practice a noisy image can be detected as being the most focused one when in reality is not. Therefore, it will be necessary to create a mathematical model that takes into account the level of noise in each image (possibly using the result of step three of the workflow) and use this mathematical model as a leveller to provide better results when in face of noisy images.



(a) Reference. (b) Underexposed. (c) Overexposed.

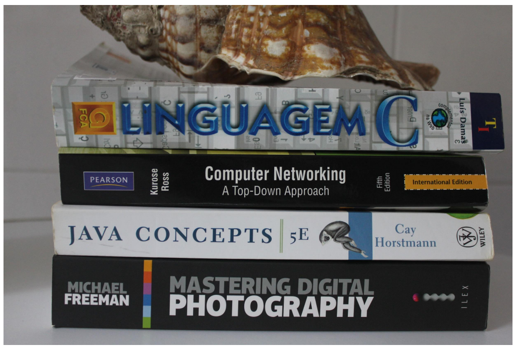
(d) Warm. (e) Cold. (f) Noisy.

Figure B.1: Differently exposed images used for gradient testing.

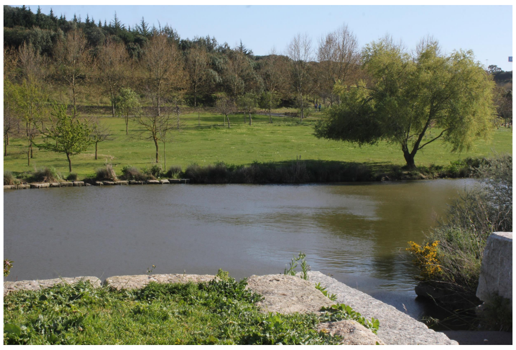
Table B.2 presents the Sobel algorithm processing times for four images, shown in Figure B.2, at different resolutions.

Table B.2: Sobel execution time (milliseconds) on different image resolutions (megapixels).

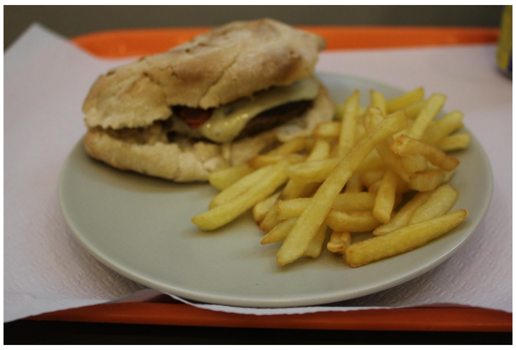
Image	0.18	0.7	2.9	11.5	18
Figure B.2(a)	15 ms	35 ms	76 ms	216 ms	331 ms
Figure B.2(b)	15 ms	34 ms	71 ms	220 ms	333 ms
Figure B.2(c)	17 ms	33 ms	67 ms	214 ms	340 ms
Figure B.2(d)	15 ms	31 ms	76 ms	204 ms	330 ms
Average	15.5 ms	33.25 ms	72.5 ms	213.5 ms	333.5 ms



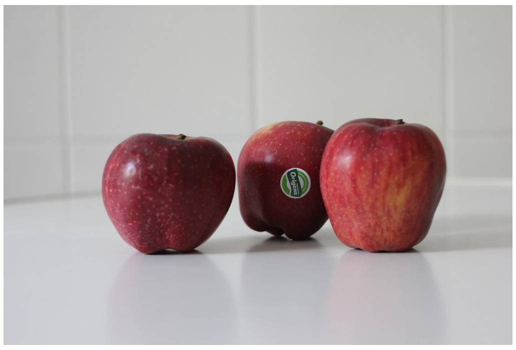
(a) Image1.



(b) Image2.



(c) Image3.



(d) Image4.

Figure B.2: Images used to test the processing times of Sobel algorithm.

Appendix C

Noise Estimation Results

This appendix presents the results of the evaluation performed over the noise estimation procedure. Table C.1 presents the comparison between the filtering-based [NH12] and the entropy-based methods in terms of processing times.

Table C.1: Comparison of entropy-based and filtering-based methods in processing times.

Image Resolution (megapixels)	Entropy-based (ms)	Filtering-based (ms)
0.24	4	3
0.96	18	12
3.85	70	45
15.4	282	185
24	407	282

From Table C.2 to Table C.9 is shown an exhaustive analysis over the filtering-based and the entropy-based method in terms of noise estimation. Four sets of similar photos were taken, each containing eight photos at standard ISO value (100, 200, 400, 800, 1600, 3200, 6400 and 12800) having the same exposure level (EV = 0), in a total of 320 photos. The average results of both methods are shown in the aforementioned tables. The unexpected results, i.e., those that are not proportional to the ISO value, are marked in red.

C.1 Filtering-based Method

Table C.2: Noise estimation results for Set 1 (filtering-based method).

ISO Value	Exp.1	Exp.2	Exp.3	Exp.4	Exp.5	Exp.6	Exp.7	Exp.8	Exp.9	Exp.10	μ
100	1.805	1.710	1.788	1.679	1.769	1.752	1.511	1.731	1.669	1.739	1.715
200	1.825	1.823	1.784	1.714	1.850	1.692	1.844	1.787	1.701	1.799	1.782
400	1.872	1.795	1.789	1.882	1.883	1.896	1.893	1.895	1.900	1.918	1.872
800	2.044	2.060	2.047	2.055	2.052	2.067	2.045	2.068	2.067	2.049	2.055
1600	2.196	2.213	2.200	2.217	2.23	2.147	2.230	2.225	2.239	2.247	2.215
3200	2.480	2.492	2.494	2.506	2.501	2.508	2.520	2.521	2.522	2.558	2.510
6400	3.021	3.012	3.009	3.040	3.072	3.073	3.080	3.084	3.110	3.089	3.059
12800	4.965	4.900	4.884	4.925	5.034	5.020	5.011	5.027	5.110	5.102	4.998

Table C.3: Noise estimation results for Set 2 (filtering-based method).

ISO Value	Exp.1	Exp.2	Exp.3	Exp.4	Exp.5	Exp.6	Exp.7	Exp.8	Exp.9	Exp.10	μ
100	5.675	5.095	3.786	5.184	5.407	5.519	5.584	5.557	5.723	5.800	5.333
200	4.887	5.165	4.709	4.746	4.574	4.663	4.675	4.631	4.582	4.725	4.736
400	4.743	4.842	4.514	4.493	4.482	4.463	4.476	4.393	4.613	4.829	4.585
800	5.592	5.652	5.527	5.441	5.677	5.693	5.565	5.502	5.519	5.671	5.584
1600	5.861	5.881	5.835	5.700	5.677	5.653	5.528	5.857	5.505	5.637	5.713
3200	6.088	5.920	5.770	5.541	5.797	5.873	5.842	5.918	5.897	5.821	5.847
6400	5.981	5.995	6.063	5.778	6.035	5.992	6.043	5.689	6.033	6.061	5.967
12800	7.067	7.011	7.207	6.907	7.019	7.100	7.045	6.978	7.031	7.171	7.053

Table C.4: Noise estimation results for Set 3 (filtering-based method).

ISO Value	Exp.1	Exp.2	Exp.3	Exp.4	Exp.5	Exp.6	Exp.7	Exp.8	Exp.9	Exp.10	μ
100	1.147	0.971	0.926	0.884	1.189	1.000	0.95	1.12	1.02	1.07	1.03
200	1.238	1.368	1.100	1.117	1.279	1.147	1.371	1.232	1.185	1.210	1.225
400	1.460	1.543	1.400	1.351	1.504	1.482	1.415	1.399	1.518	1.368	1.444
800	1.762	1.741	1.746	1.755	1.835	1.656	1.696	1.685	1.697	1.853	1.743
1600	2.064	2.058	2.040	2.056	1.980	1.971	2.053	2.027	2.047	2.017	2.031
3200	2.471	2.476	2.469	2.473	2.474	2.429	2.407	2.450	2.484	2.477	2.461
6400	3.092	3.096	3.022	3.082	3.082	3.104	3.110	3.129	3.149	3.117	3.098
12800	4.470	4.492	4.436	4.500	4.422	4.514	4.558	4.533	4.540	4.562	4.503

Table C.5: Noise estimation results for Set 4 (filtering-based method).

ISO Value	Exp.1	Exp.2	Exp.3	Exp.4	Exp.5	Exp.6	Exp.7	Exp.8	Exp.9	Exp.10	μ
100	4.323	4.122	4.026	4.109	4.166	4.222	4.027	4.016	3.672	3.758	4.044
200	4.252	4.315	4.041	4.189	4.123	3.995	3.940	3.983	4.021	3.959	4.082
400	4.416	4.283	3.914	4.125	4.030	3.981	3.914	3.850	3.695	3.887	4.009
800	4.044	4.173	3.776	3.934	3.956	3.912	3.760	3.653	3.579	3.743	3.853
1600	3.939	3.945	3.836	3.763	3.733	3.642	3.846	3.682	3.710	3.547	3.764
3200	4.267	4.132	4.148	4.104	4.091	4.018	3.963	3.875	3.829	3.883	4.031
6400	4.484	4.582	4.561	4.471	4.464	4.423	4.345	4.265	4.413	4.428	4.443
12800	5.543	5.489	5.446	5.433	5.406	5.439	5.354	5.458	5.468	5.478	5.451

C.2 Entropy-based Method

Table C.6: Noise estimation results for Set 1 (entropy-based method).

ISO Value	Exp.1	Exp.2	Exp.3	Exp.4	Exp.5	Exp.6	Exp.7	Exp.8	Exp.9	Exp.10	μ
100	6.232	7.088	6.364	7.391	6.739	6.727	6.834	6.505	6.704	6.353	6.694
200	7.156	7.079	7.131	7.057	7.340	7.528	7.592	7.166	7.344	7.191	7.259
400	8.561	8.578	8.630	8.872	8.99	8.810	8.677	8.677	8.703	8.661	8.751
800	10.386	10.485	10.495	10.348	10.864	10.712	10.469	10.656	10.541	10.672	10.563
1600	13.404	13.439	13.473	13.327	13.945	13.369	13.638	13.833	13.782	13.750	13.623
3200	15.710	15.980	16.152	15.826	15.897	15.898	16.067	15.843	15.862	16.115	15.935
6400	22.711	22.301	22.704	23.086	22.682	23.392	23.239	23.298	23.410	23.497	23.032
12800	33.352	33.103	32.917	33.176	33.650	33.671	33.570	33.813	33.879	33.979	33.511

Table C.7: Noise estimation results for Set 2 (entropy-based method).

ISO Value	Exp.1	Exp.2	Exp.3	Exp.4	Exp.5	Exp.6	Exp.7	Exp.8	Exp.9	Exp.10	μ
100	15.983	15.400	15.690	16.625	14.846	15.127	15.142	15.029	14.958	14.915	15.372
200	16.943	16.450	16.450	16.228	15.973	16.034	16.216	15.971	16.142	15.449	16.186
400	18.830	18.892	18.468	18.290	17.771	17.938	17.900	18.039	17.673	17.469	18.127
800	22.321	21.554	21.250	21.578	20.712	20.935	19.968	20.652	20.152	19.621	20.874
1600	23.162	22.626	22.779	23.936	21.907	22.846	22.799	21.082	23.033	22.723	22.689
3200	22.961	23.285	23.442	22.928	23.177	23.179	23.002	22.620	22.734	22.893	23.024
6400	29.920	29.636	29.969	31.199	29.673	30.119	30.018	29.981	29.945	30.185	30.058
12800	40.972	40.454	39.947	40.127	40.775	40.352	41.159	41.535	41.362	40.565	40.972

Table C.8: Noise estimation results for Set 3 (entropy-based method).

ISO Value	Exp.1	Exp.2	Exp.3	Exp.4	Exp.5	Exp.6	Exp.7	Exp.8	Exp.9	Exp.10	μ
100	2.967	3.107	2.767	3.044	2.958	3.079	3.031	3.095	2.910	3.079	3.004
200	3.243	3.507	3.006	3.145	3.240	3.231	3.284	3.445	3.210	3.414	2.372
400	3.845	3.675	3.675	3.974	3.907	3.983	4.057	4.117	3.975	4.137	3.934
800	4.373	4.282	4.325	4.375	4.522	4.438	4.378	4.399	4.371	4.411	4.819
1600	5.331	5.496	5.283	5.367	5.839	5.664	6.038	5.642	5.777	5.646	5.608
3200	6.990	6.243	6.981	7.055	6.956	6.960	6.887	7.513	7.234	7.135	6.995
6400	9.541	10.160	11.345	9.418	10.220	9.919	9.773	10.062	10.167	9.713	10.029
12800	16.234	16.626	16.313	15.777	16.183	17.425	16.083	15.920	16.122	16.159	16.184

Table C.9: Noise estimation results for Set 4 (entropy-based method).

ISO Value	Exp.1	Exp.2	Exp.3	Exp.4	Exp.5	Exp.6	Exp.7	Exp.8	Exp.9	Exp.10	μ
100	20.776	21.534	22.150	22.533	23.233	23.180	23.003	23.141	23.682	23.249	22.648
200	20.750	20.616	21.189	22.291	23.461	23.660	23.810	23.963	23.479	23.876	22.710
400	20.834	21.651	22.294	23.212	23.849	24.390	24.895	25.347	24.841	24.983	23.629
800	23.403	22.544	23.463	23.908	24.548	25.077	25.648	25.887	25.634	25.548	24.566
1600	25.128	25.131	25.982	25.974	26.869	27.188	26.328	27.428	27.162	27.718	26.491
3200	27.883	27.630	27.396	27.517	27.534	27.987	27.769	28.267	28.429	27.349	27.776
6400	32.308	30.844	31.287	31.189	32.150	31.937	32.652	33.354	32.057	32.271	32.005
12800	36.787	37.237	37.932	37.622	39.094	39.041	40.176	36.738	36.841	36.726	37.819

Appendix D

Exposure Level Results

This appendix presents the results of the evaluation performed over the exposure level procedure. Table D.1 presents the processing times required for the computation of the image average brightness.

Table D.1: Processing times for the estimation of the image exposure level.

Image Resolution (megapixels)	CPU Time (ms)
0.24	150
0.96	152
3.85	153
15.4	161
24	170

Appendix

E

Code Snippets

This appendix presents the programming code, using the Opencv [Its] library, for the implemented procedures.

E.1 Image Correlation

Listing E.1 presents the programming code for keypoint detection in an image. According to the evaluation described in Section 3.2.1.2, SURF [Bay+08] proved to be the best option to keypoint detection in the context of this project. Thus, the code shown in Listing E.1 refers to the SURF algorithm.

E.1.1 Keypoint Detection

Listagem E.1: Code for keypoint detection using SURF algorithm

```
std::vector<KeyPoint> KeypointDetection::getKeypointsSURF(double thresholdValue){
  SurfFeatureDetector detector(thresholdValue, 4, 2, true, false);
  std::vector<KeyPoint> keypoints;
  detector.detect(img, keypoints);
  return keypoints;
}
```

The code related to keypoint description, using SURF, is shown in Listing E.2.

E.1.2 Keypoint Matching

Listing E.3 presents the programming code for keypoint matching. As described in Section 3.2.1.3, the FLANN [ML09] library was chosen for keypoint matching between two different images. The output of this method is a set of matches (*good_matches*). In case

Listagem E.2: Code for keypoint description using SURF algorithm

```
cv::Mat KeypointDetection::getDescriptorSURF(std::vector<KeyPoint> keypoints){
  SurfDescriptorExtractor extractor;
  cv::Mat descriptor;
  extractor.compute(img, keypoints, descriptor);
  return descriptor;
}
```

of need for future evaluation, a data structure (*bad_matches*) was also created. This data structure contains the set of matches that are considered to be outliers, i.e., false matches between keypoints of different images.

Listagem E.3: Code for keypoint matching using FLANN library

```
std::vector<DMatch> KeypointMatching::flann(){
  cv::Mat descriptorImg1 = image1.getDescriptors();
  cv::Mat descriptorImg2 = image2.getDescriptors();

  FlannBasedMatcher matcher;
  std::vector<std::vector<DMatch>> matches;
  matcher.knnMatch(descriptorImg1, descriptorImg2, matches, 2);

  std::vector< DMatch > good_matches, bad_matches;

  const float minRatio = 1.f / 1.5f;
  for (size_t i=0; i<matches.size(); i++){
    cv::DMatch bestMatch = matches[i][0];
    cv::DMatch betterMatch = matches[i][1];

    float distanceRatio = bestMatch.distance/betterMatch.distance;

    if (distanceRatio < minRatio)
      good_matches.push_back(bestMatch);
    else
      bad_matches.push_back(bestMatch);
  }

  return good_matches;
}
```

Listing E.4 presents the code related to the removing of the outliers that FLANN did not detect. The method $findHomography$ takes as input a set of matches, between keypoints of two images, returning as output the estimated homography matrix between those two images.

E.2 Focus Detection

Listing E.5 shows the code for focus analysis. Namely, the gradient computation obtained through the Sobel algorithm. Sobel computes the gradient in the horizontal and vertical

Listagem E.4: Code for outlier removal using RANSAC method

```
  cv::Mat KeypointMatching::ransac(std::vector<DMatch> matches, float
      reprojectionThreshold){
  std::vector<KeyPoint> keypointsImg1, keypointsImg2;
  keypointsImg1 = image1.getKeypoints();
  keypointsImg2 = image2.getKeypoints();

  std::vector<Point2f> srcPoints(matches.size());
  std::vector<Point2f> dstPoints(matches.size());
  for (size_t i = 0; i < matches.size(); i++){
    srcPoints[i] = keypointsImg2[matches[i].trainIdx].pt;
    dstPoints[i] = keypointsImg1[matches[i].queryIdx].pt;
  }

  std::vector<unsigned char> inliersMask(srcPoints.size());
  cv::Mat h = findHomography(srcPoints, dstPoints, CV_FM_RANSAC, reprojectionThreshold,
      inliersMask);

  return h;
}
```

directions, wherein its output is an approximate weighted sum of the gradient in both directions (see Sections 3.2.2 for more details).

Listagem E.5: Code for focus analysis

```
cv::Mat FocusDetection::applySobel(cv::Mat image){
    int scale = 1;
    int delta = 0;
    int ddepth = CV_64F;
    cv::Mat grad_x, grad_y;
    cv::Mat abs_grad_x, abs_grad_y;
    cv::Mat grad;

    // Gradient X
    cv::Sobel(image, grad_x, ddepth, 1, 0, 3, scale, delta, cv::BORDER_DEFAULT);
    cv::convertScaleAbs(grad_x, abs_grad_x);

    // Gradient Y
    cv::Sobel(image, grad_y, ddepth, 0, 1, 3, scale, delta, cv::BORDER_DEFAULT);
    cv::convertScaleAbs(grad_y, abs_grad_y);

    // Total Gradient (approximate)
    cv::addWeighted(abs_grad_x, 0.5, abs_grad_y, 0.5, 0, grad);

    return grad;
}
```

The code presented in E.6 may be useful in the future as a possible approach to apply a mask over the focused regions of an image.

Listagem E.6: Code for applying a mask over the image focused regions

```
void FocusDetection::applyMask(cv::Mat image, cv::Mat image1, cv::Mat gradient, int
    value){
   image = image1.clone();
   for(int i=0; i<image.cols; i++){
         for(int j=0; j<image.rows; j++){
            if(gradient.at<uchar>(cv::Point(i,j)) < value){
               image.at<cv::Vec3b>(cv::Point(i,j))[0] = 0;
               image.at<cv::Vec3b>(cv::Point(i,j))[1] = 0;
               image.at<cv::Vec3b>(cv::Point(i,j))[2] = 255;
            }
            if(gradient.at<uchar>(cv::Point(i,j)) >= value){
               image.at<cv::Vec3b>(cv::Point(i,j))[0] = 0;
               image.at<cv::Vec3b>(cv::Point(i,j))[1] = 255;
               image.at<cv::Vec3b>(cv::Point(i,j))[2] = 0;
            }
         }
      }
   cv::Mat dst1;
   cv::addWeighted(image1, 0.5, image, 0.5, 0.0, dst1);
}
```

E.3 Motion Blur

Listing E.7 shows the code related to motion blur detection (not fully implemented). In a first step the image is divided into NxN regions. In a second step, the Hough Transform [VC62] is applied to each region in order to detect equally oriented lines (see Section 3.2.3 for more details).

E.4 Noise Estimation

E.4.1 Filtering-based Method

Listing E.8 presents the code for noise estimation using the filtering-based method (see Section 2.5.1 and 3.2.4.1 for more details).

E.4.2 Entropy-based Method

Listing E.9 shows the code for noise estimation based on the entropy of the hue colour channel. The image colour space is converted from RGB to HSV in order to access the hue component. The histogram containing the distribution of the hue component is then computed in order to be used for the calculation of the entropy (see Section 3.2.4.2 for more details).

Listagem E.7: Code for motion blur detection

```
void MotionBlurDetection::applyHoughTransform(cv::Mat image){
   cv::Mat dst;
   cv::Canny(image, dst, 40, 120, 3);
   divideImage(dst, 5);
}

void divideImage(cv::Mat img, int N){
   for (int r = 0; r < img.rows; r += img.rows/N){
      for (int c = 0; c < img.cols; c += img.cols/N){
         cv::Mat tileCopy = img(cv::Range(r, cv::min(r + img.rows/N, img.rows)),
                   cv::Range(c, cv::min(c + img.cols/N, img.cols))).clone();
         if(tileCopy.rows > 20)
            applyHoughTransformToROI(tileCopy);
      }
   }
}

void applyHoughTransformToROI(cv::Mat roi){
   std::vector<cv::Vec4i> lines;
   int directions[180] = {0};
   int maxDirection = 0;
   int houghThreshold = roi.rows/8;
   cv::HoughLinesP(roi, lines, 1, CV_PI/180, houghThreshold, roi.rows/8, 10);
}
```

Listagem E.8: Code for noise estimation using filtering-based method

```
double NoiseEstimation::filteringBasedValue(cv::Mat grayImage){
   cv::Mat filteredImage;
   cv::blur(grayImage, filteredImage, cv::Size(11,11));
   cv::Mat result;
   cv::subtract(grayImage, filteredImage, result);
   cv::Scalar meanSrc, stdSrc;
   cv::meanStdDev(result, meanSrc, stdSrc);
   double stdSrcValue = stdSrc.val[0];
   return stdSrcValue;
}
```

E.5 Level of exposure

Listing E.10 shows the code that sorts the images, from the set of similar images, according to their average brightness. It is important to refer that in this Listing there is another operation of conversion from the RGB to the HSV colour space. However when all the referenced algorithms are combined in a unique software tool, it is expected that the HSV is already computed at the procedure of noise estimation.

Listagem E.9: Code for noise estimation using entropy-based method

```cpp
double NoiseEstimation::getHueEntropy(cv::Mat rgbImage){
   cv::Mat hsvImage;
   cv::cvtColor(rgbImage, hsvImage, CV_BGR2HSV);
   int h_bins = 180;
   int histSize[] = {h_bins};
   float h_ranges[] = {0, 180};
   const float* ranges[] = {h_ranges};
   int channels[] = {0};
   cv::MatND hist_input;
   cv::calcHist(&hsvImage, 1, channels, cv::Mat(), hist_input, 1, histSize, ranges,
       true, false);
   cv::normalize(hist_input, hist_input, 0, 1, cv::NORM_MINMAX, -1, cv::Mat());
   cv::Mat logP;
   cv::log(hist_input, logP);
   double entropy = -sum(hist_input.mul(logP)).val[0];

   return entropy;
}
```

Listagem E.10: Code for exposure level analysis

```cpp
double ExposureLevel::getLightness(cv::Mat image){
   cv::Mat hsvImg;
   cv::cvtColor(image, hsvImg, CV_BGR2HSV);
   std::vector<cv::Mat> channels;
   cv::split(hsvImg, channels);
   cv::Scalar mean = cv::mean(channels[2]);
   return mean[0];
}
```