University of Southern Denmark

Introduction to Robotics and Computer Vision

Vision – Mandatory Exercise 1
Image restoration

by:
Keerthikan Ratnarajah
kerat12@student.sdu.dk
Jes Grydholdt Jepsen
jejep12@student.sdu.dk

Dato: November 2, 2015

Contents

1 **Introduction** 2

2 **Image 1** 2

3 **Image 2 – Salt and Pepper noise** 2

4 **Image 3** 4

5 **Image 4** 4

6 **Image 5** 4

1 Introduction

In the real world images are sometimes affected by noise in a way that makes working with them (or looking at them) difficult. The purpose of this project is therefore look at image restoration on a set of images that are affected by different kinds of defects. The overall task is to minimize the impact of the noise and thereby to improve the quality of the image.

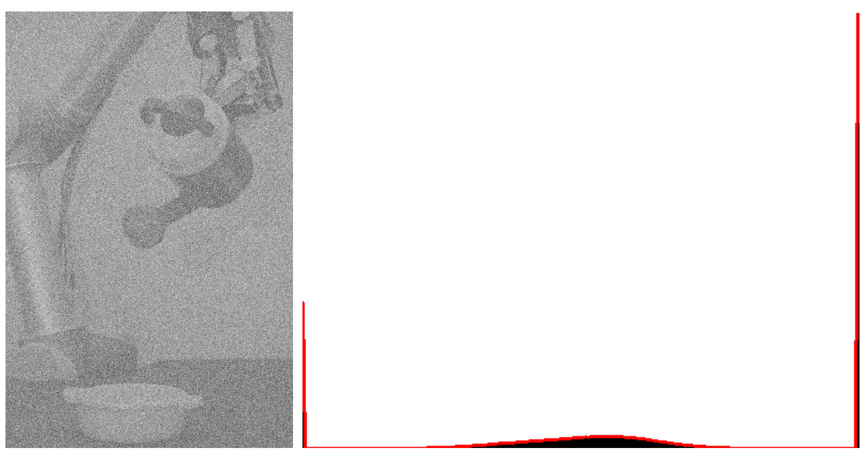
To come up with an analysis and a way to remove (or weaken) the defect, the following considerations are done for each image:

- Investigate the image and identify the defect, for example by using the histogram and/or the frequency spectrum of the image.
- Design a solution that removes or weakens the impact of the defect and investigate the properties of the solution.
- Investigate different solution possibilities.
- Implement and apply the solution(s)

2 Image 1

3 Image 2 – Salt and Pepper noise

Figure 1a shows the original image, which is filled with black and white pixels. This kind of noise is known as salt–and–pepper noise, and from the image's histogram (figure 1b) can the amount of salt–and–pepper noise be seen; salt noise on the right and pepper noise on the left.



(A) The original image (B) Histogram of the original image

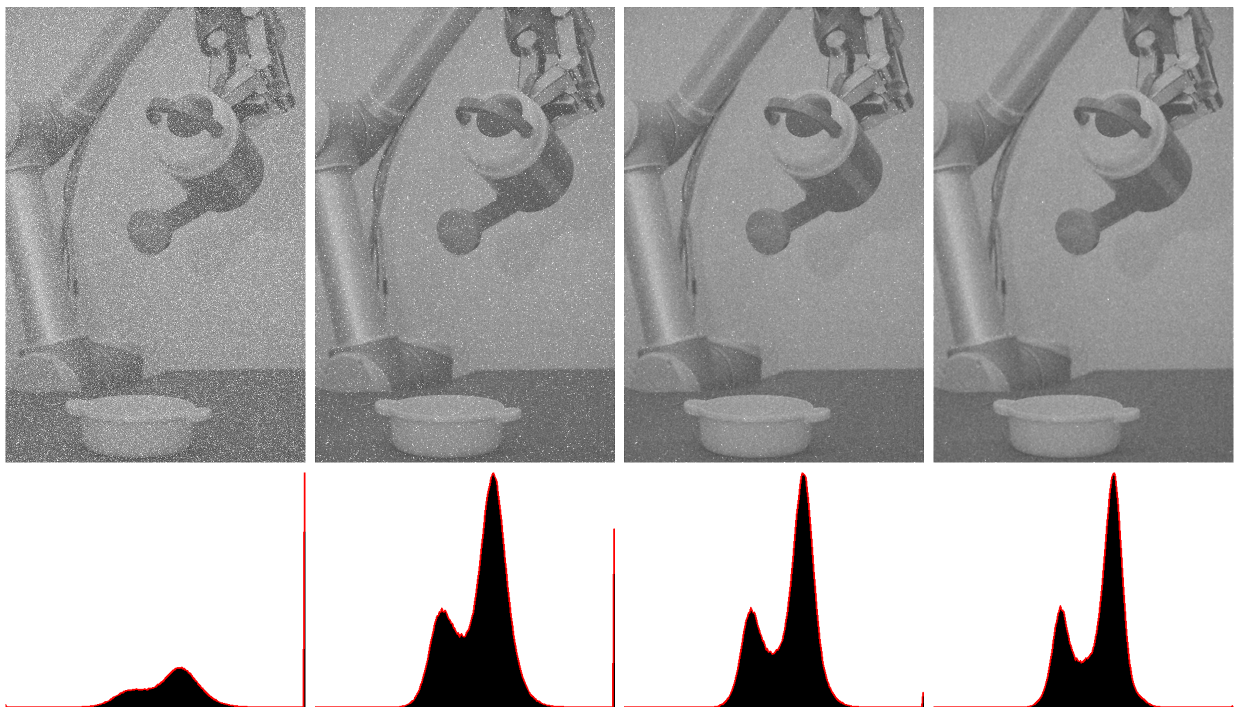
Figure 1: Analysis of image 2

Approximately are there three times more salt noise compared to the pepper noise. The values in the middle of the histogram is what is left of the original image, and in order to restore as much as possible, and median filter is applied on the image. The median filter is chosen because it is very effective against salt-and-pepper noise in the images, and the OpenCV function

```
void medianBlur(InputArray src, OutputArray dst, int ksize)
```
[1]

is used for applying the filter to the image. Basically the image blurred by using the median filter, and depending on the size of `ksize`, the more blurred will the image become.

The `ksize` is the size of the kernel filter applied to each pixel in the image, and therefore must the value of `ksize` be odd and greater than 1, so in order to test if the noise is removed, a kernel of `3` is applied on the image 2 and checked if all the noise is removed. If not all the noise is removed, the kernel size is increased with two, and checked again, and so on.

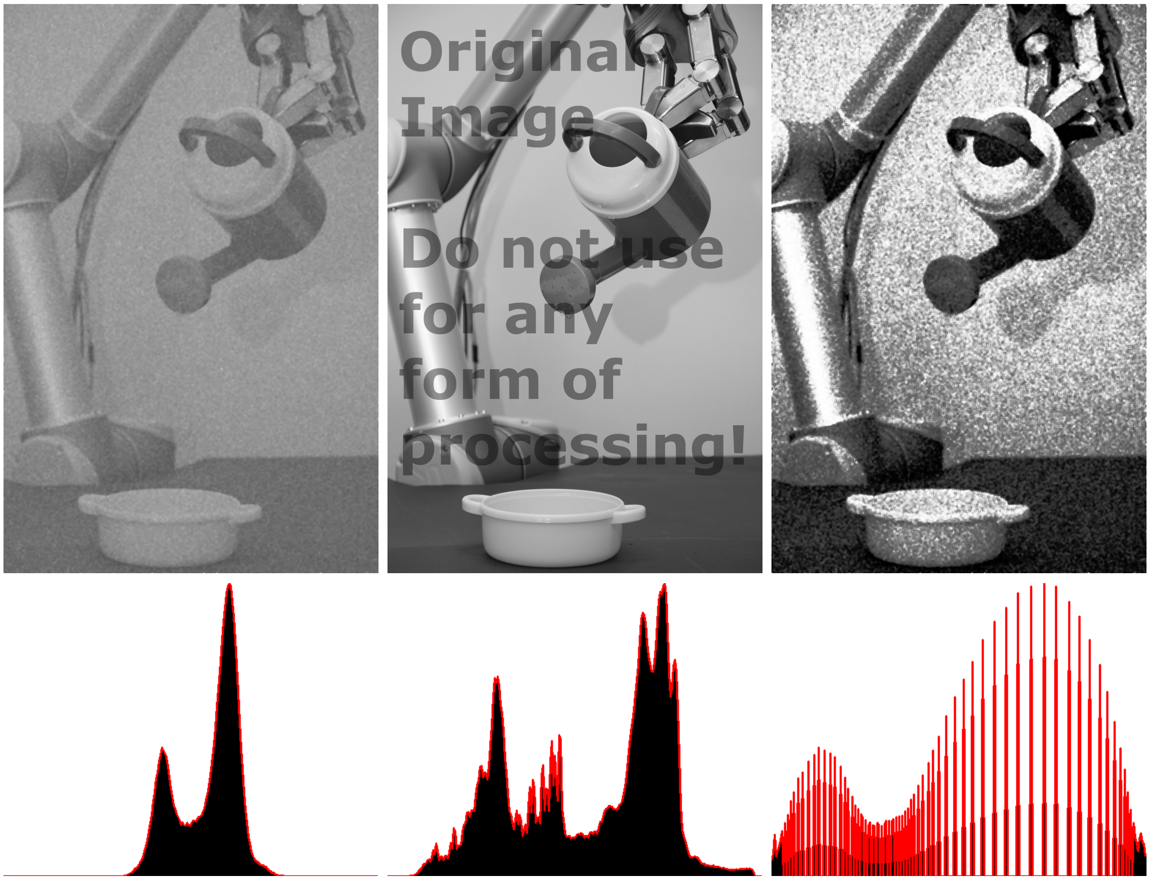


(A) KSIZE = 3 (B) KSIZE = 5 (C) KSIZE = 7 (D) KSIZE = 9

FIGURE 2: ANALYSIS OF IMAGE 2

Figure 2a, 2b, 2c and 2d shows the process of finding the right `ksize`, but all four kernel sizes still does not remove all the noise, especially the salt noise. All the salt–and–pepper noise is removed with a `ksize=11`, but the disadvantage of this approach is that the details in the image are reduced. For restoring as much as possible of the these details, a histogram equalization is applied on the noise reduced image. Histogram equalization restores the details by improves the contrast in the image by stretching out the intensity range of the image. On the left and right side of image 3a's histogram are there underpopulated intensities, which is why an histogram equalization is an optimal choice. The result is shown on figure 3c.

[1] `http://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html#voidmedianBlur(InputArraysrc, OutputArraydst,intksize)`



(a) `ksize = 11` (b) Equalized (c) Equalized

Figure 3: Analysis of image 2

Which one is best?

4 Image 3

5 Image 4

6 Image 5