Digital Image Processing: 2nd assignment

Tiago Moreira Trocoli da Cunha
RA: 226078
t226078@dac.unicamp.br

May 2019

1 Introduction

The second assignment informs that three different algorithms must be developed and employed to turn a grayscale images into binary ones made up of black and white colors only as well as intentionally noising them, technique known as dithering. The first operation converts every pixel's intensity into values between 0 to 9 for half-toning and 0 to 16 for the Bayer filter. The last algorithm used is the Floyd-Steinberg dithering that adds the residual quantization error of a pixel onto its neighboring pixels, it also can scan images from left to right and the opposite direction making a zigzag movement during its operation. The assignment also informs that input images must be in pgm format and output ones in pbm.

2 Program Specification

The code is written using Python 2.7 in Ubuntu 18.04. I used the following libraries:

- **scipy**: used to save images.
- **opencv**: just used it to open an image.
- **matplotlib**: used to plot images.
- **numpy**: used for matrix operations.

The program receives images AndrewHepbur.pgm (figure 11), forest.pgm (figure 1) and winter.pgm (figure 6) as input and produces .pbm images as output. The output images is created after applying half-toning, the Bayer filter, the Floyd-Steinberg with and without zigzag movements totalizing 12 images. The zip file contains homework2.pdf (this file), homework2.ipynb, homework2.html and all the aforementioned images.

3 Decisions and Code

The half-toning has 10 patterns while the Bayer filter 17 in total. I could have manually produced them, but it would take some time, besides, comparing each pixel's intensity of an image to each value of the mask during dithering process would require unnecessary computational calculation, so I developed an algorithm to automatically create that patterns prior to dithering technique, it called create_pattern.

The half-toning and Bayer filter require normalizing the image to scale 0 to 9 (base 9) and 0 to 16 (base 16), respectively. To do that I develped a simple function *convert_baseN* that relies on the following calculations:

$$I^{'} = \left[\frac{I}{(256/n)} \right],$$

which I is the intensity in base 256, $I^{'}$ in base n and $[\cdot]$ is the nearest integer function. Doing that for $n = 16$, produces $I^{'} = 0$ for I between 0 to 15, $I^{'} = 1$ for values 16 to 31 an so on.

The next section is about the analysis of each image and their respective results after dithering. However, to better organize this paper, I show the images on the Appendix section.

3.1 Analysis of Forest Image

I don't see any difference regarding the quality of images between Floyd-Steinberg with and without zigzag, as we can see in figures 4 and 5. In half-toning (figure 2) and Bayer (figure 3) filter we can clearly see small squares if we zoom, however, the Bayer also prints small white dots equidistant to each other. The Floyd-Steinberg filters turn the image more black then the other ones, we can clearly see at the road. I prefer the Bayer filter since we have the illusion of seeing the image in more tonality even knowing that there are only black and white colors and also is less black than the others.

3.2 Analysis of Winter Image

We can clearly see the equidistant squares in images produced by Bayer (figure 8) and half-toning (figure 7), although the last one is more prominent as if the image became more discrete. Of course, it is the result of the squared mask. The results from Floyd-Steinberg (figures 9 and 10) produced more noise, we can notice many white dots throughout images, also, the reflect of water became all black.

3.3 Analysis of Andrew Hepburn Image

The big difference between filters relies on the oval background light and the letter. Both Bayer (figure 13) and half-toning (figure 12) are very similar, producing small equidistant squares, however, the Bayer produces the illusion of

more level of tonality in the background light, as if it had 7 levels, while the other had 5. Interesting to see that the original one the tonality appears to be continuous, not discrete. I didn't like the results of Floyd-Steinberg (figures 14 and 15) because the letters became "melted" and background light became weird, increasing and decreasing the tonality.

4 Comments

In general, the Floyd-Steinberg makes much more noise them the other two because of its propriety of spreading the error of each pixel, while half-toning and the Bayer filter make the image similar to the original one as well as turning the image's appearance more discrete. I developed the entire code in my computer with Ubuntu 18.4. I was surprised to see that Fedora shows the images in better quality than Ubuntu. Moreover, the notebook shows low-quality images, so it is better to visualize them in this text or in a image viewer.

5 Appendix

Figure 1: Original Forest Image.

Figure 2: Forest image after Half-Toning.

Figure 3: Forest image after Bayer Filter.

Figure 4: Forest image after Floyd Steinberg without zigzag.

Figure 5: Forest image after Floyd Steinberg with zigzag.

Figure 6: Original Winter Image.

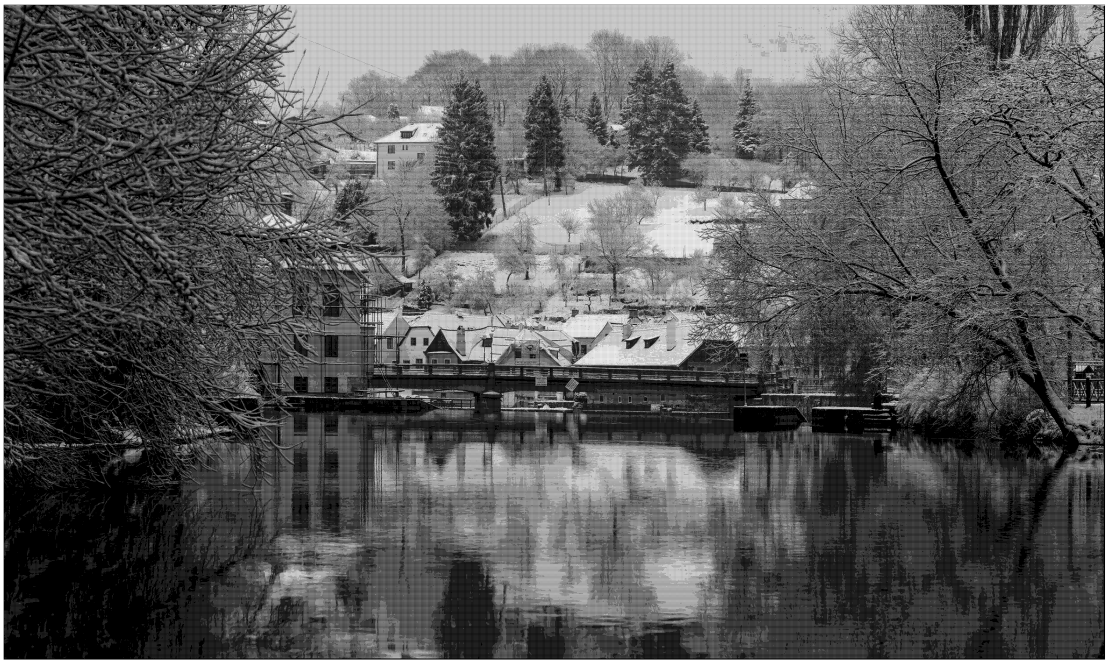

Figure 7: Winter image after Half-Toning.

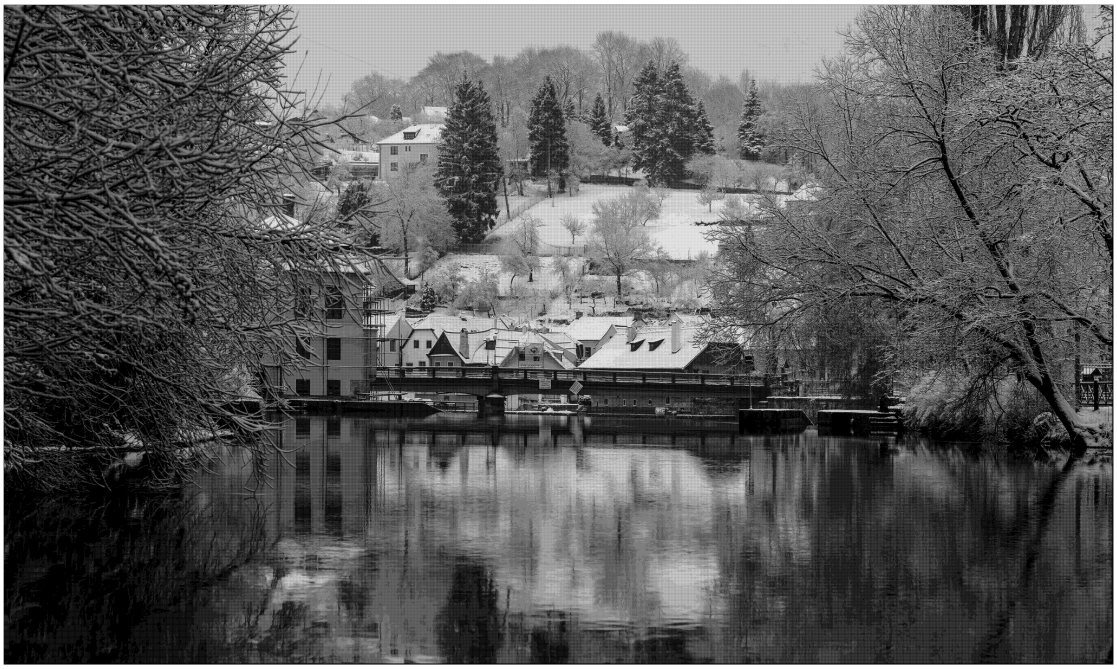

Figure 8: Winter image after Bayer Filter.

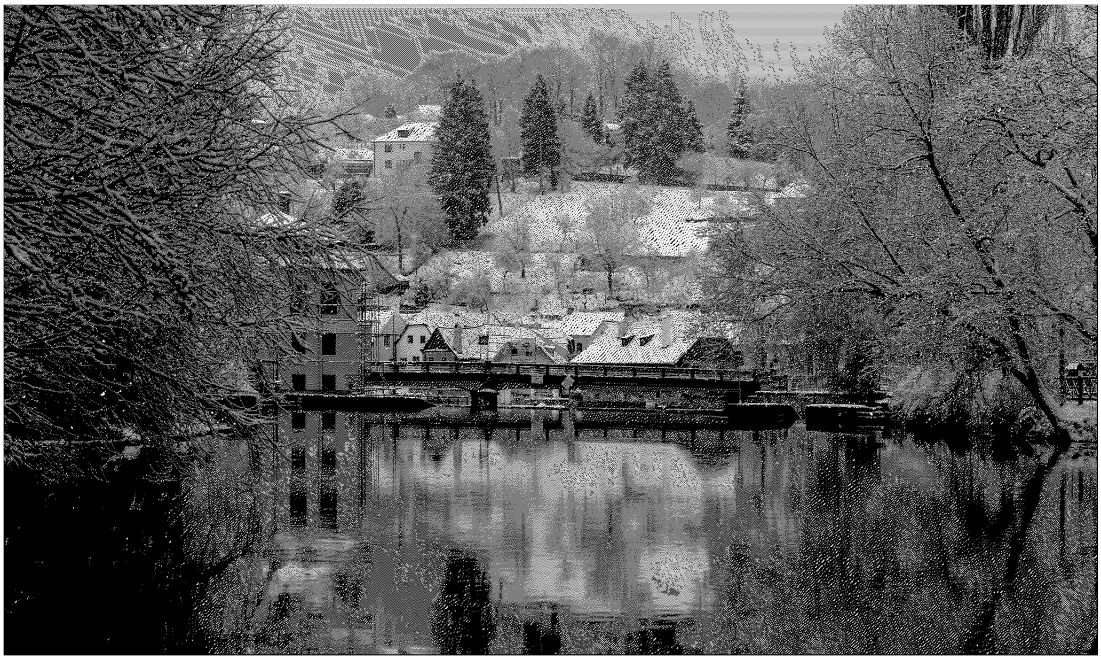

Figure 9: Winter image after Floyd Steinberg without zigzag.

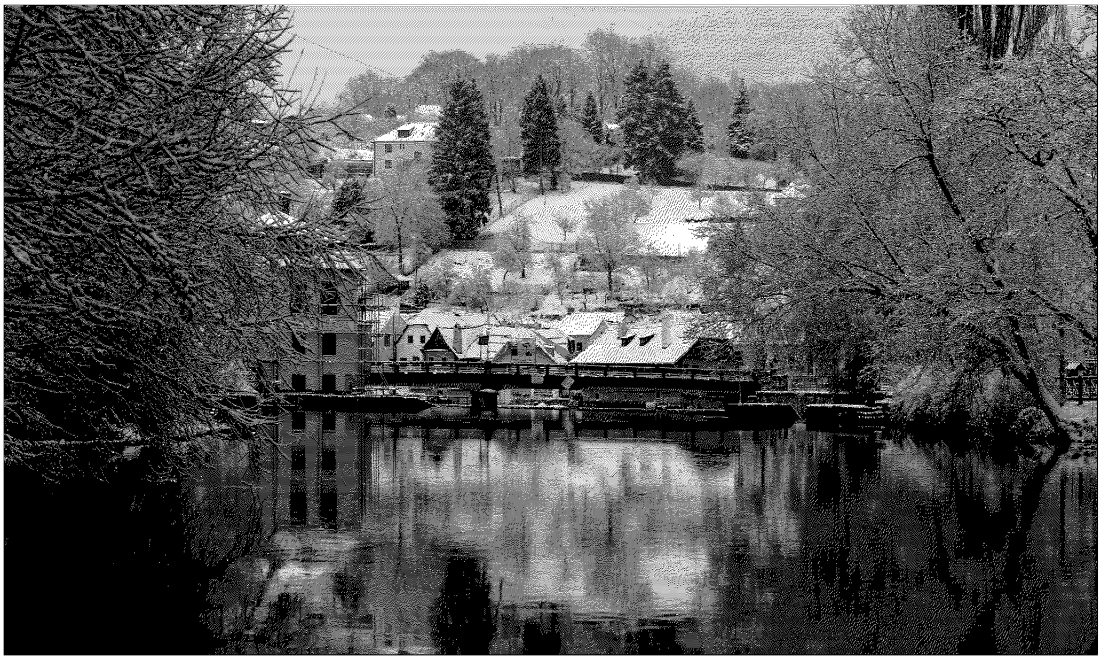

Figure 10: Winter image after Floyd Steinberg with zigzag.

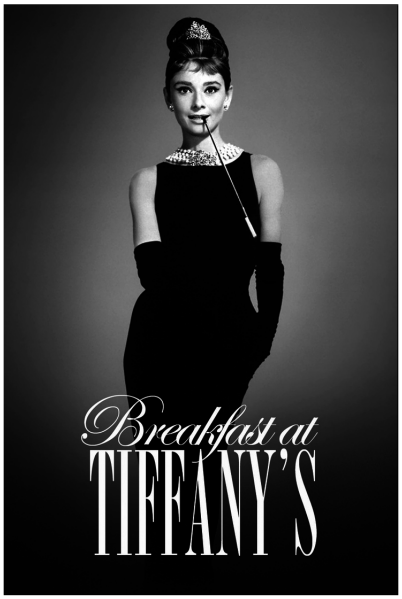

Figure 11: Original Andrew Hepburn Image.

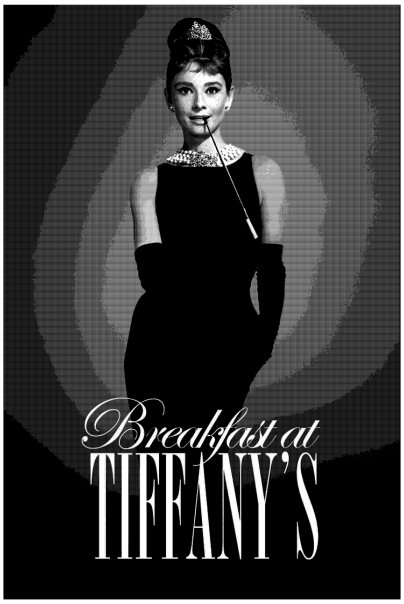

Figure 12: Andrew Hepburn image after Half-Toning.

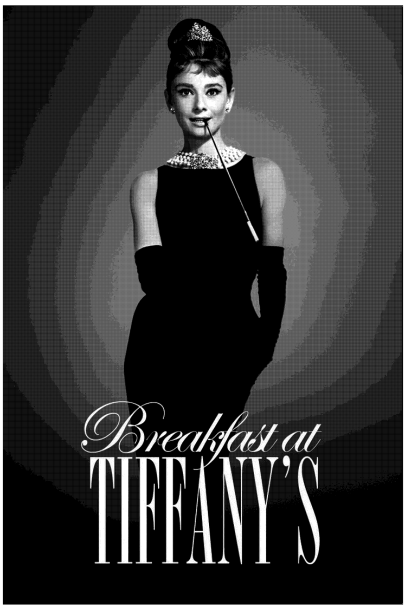

Figure 13: Andrew Hepburn image after Bayer Filter.

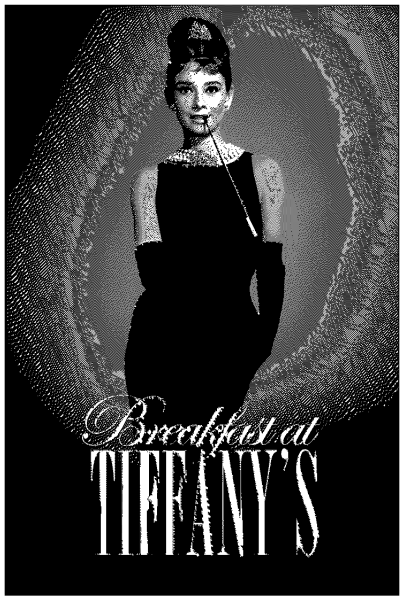

Figure 14: Andrew Hepburn image after Floyd Steinberg without zigzag.

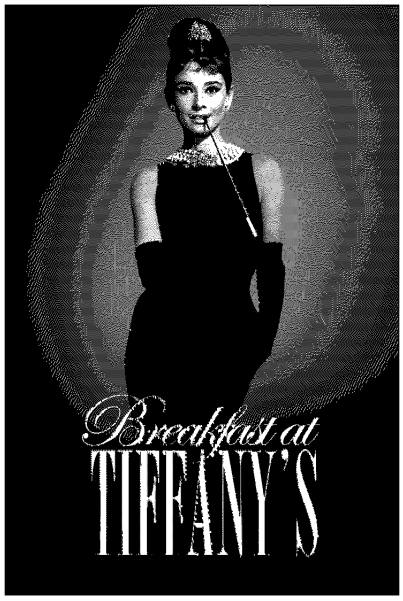

Figure 15: Andrew Hepburn image after Floyd Steinberg with zigzag.