UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

DAVI PADILHA MESQUITA

Reaction-Diffusion Woodcuts

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Advisor: Prof. Dr. Marcelo Walter

Porto Alegre
March 2019

CIP — CATALOGING-IN-PUBLICATION

Mesquita, Davi Padilha

Reaction-Diffusion Woodcuts / Davi Padilha Mesquita. – Porto Alegre: PPGC da UFRGS, 2019.

80 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2019. Advisor: Marcelo Walter.

1. Reaction-Diffusion. 2. Woodcuts. 3. Expressive Rendering. I. Walter, Marcelo. II. Título.

ABSTRACT

Woodcuts are a traditional form of engraving, where paint is rolled over the surface of a carved wood block which will be used as a printing surface over a sheet of paper, so only the non-carved parts will be printed in the paper. In this work, we present an approach for computer simulated woodcuts using reaction-diffusion as the underlying mechanism. First, we preprocess the segmented input image to generate a parameter map, containing values for each pixel of the image. This parameter map will be used as an input to control the reaction-diffusion processing, allowing different areas of the image to have distinct appearances, such as spots or stripes with varied size or direction, or areas with plain black or white color. Reaction-diffusion is then performed resulting in the raw appearance of the final image. After reaction-diffusion, we apply a thresholding filter to generate the final woodcut black and white appearance. To better validate our work, we performed a qualitative evaluation of our results. Our results show that the final images look qualitatively similar to some styles of woodcuts, and add yet another possibility of computer generated artistic expression, which demonstrates the potential of reaction-diffusion to tasks in the field of non-photorealistic rendering.

Keywords: Reaction-Diffusion. Woodcuts. Expressive Rendering.

Xilogravuras via Reação-Difusão

RESUMO

A xilogravura é uma forma tradicional de arte onde passa-se tinta na superfície de um bloco de madeira entalhado que será usado como superfície de impressão sobre uma folha de papel, de modo que apenas as partes que não foram entalhadas serão impressas no papel. Neste trabalho, nós apresentamos uma nova abordagem para a síntese computacional de xilogravuras usando reação-difusão como mecanismo fundamental. Primeiro, nós pré-processamos uma imagem de entrada segmentada para gerar um mapa de parâmetros contendo valores para cada pixel da imagem. Este mapa de parâmetros será usado para controlar o processo de reação-difusão, permitindo que diferentes áreas da imagem apresentem um aspecto distinto, como listras e pintas em diferentes tamanhos e direções, ou áreas monocromáticas pretas e brancas. Executamos então o processo de reação-difusão resultando em uma prévia da aparência final. Após a reação-difusão, nós aplicamos um filtro de limiarização para gerar como resultado uma xilogravura em preto e branco. Para melhor validar o nosso trabalho, nós realizamos uma avaliação qualitativa dos nossos resultados. Nossos resultados mostram que as imagens finais parecem qualitativamente similares a alguns estilos de xilogravuras, e adicionam mais uma possibilidade para a expressão artística gerada por computador, o que demonstra o potencial da reação-difusão para trabalhos na área de renderização expressiva.

Palavras-chave: Reação-Difusão, Xilogravuras, Renderização Expressiva.

LIST OF ABBREVIATIONS AND ACRONYMS

RD	Reaction-Diffusion
NPR	Non-Photorealistic Rendering

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

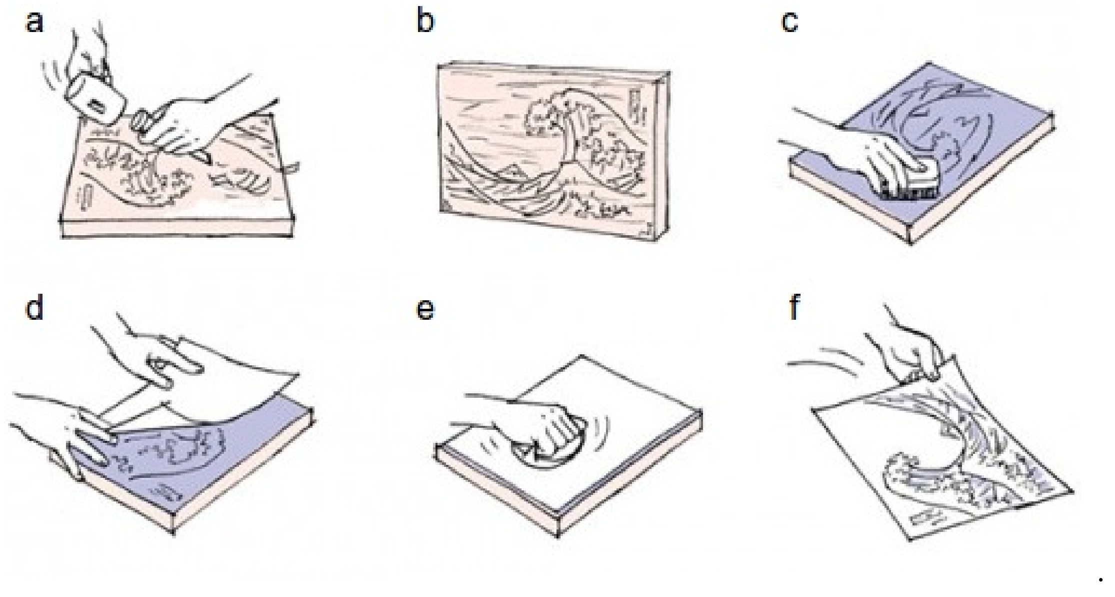
Traditionally, computer graphics (CG) aims to represent real objects and scenes as accurate as possible, or, in other words, to achieve photorealism, the same degree of visual similarity with the real world a photograph has. However, non-photorealistic rendering (NPR) (STROTHOTTE; SCHLECHTWEG, 2002), also known as *Expressive Rendering*, is a CG area which works not on photorealistic renditions, but with the simulation of expressive artistic style. NPR always looked for inspiration in the many forms of traditional artistic expression, trying to simulate the visual results of manual techniques, such as paintings, comics or animation.

Since the early work of Haeberli (HAEBERLI, 1990), with the computational simulation of a painting, this area of research has seen major advances, being able to convincingly simulate natural media effects such as pencil drawing (LU; XU; JIA, 2012), oil painting in real-time and on mobile devices (STUYCK et al., 2017), watercolors (DIVERDI et al., 2013) (WANG et al., 2014), among many others. Hegde, Gatzidis, and Tian present a review of these artistic inspired results in (HEGDE; GATZIDIS; TIAN, 2013).

Among the many relief printing and artistic techniques, such as etching or engraving, woodcuts have a distinct look due to their nature. A block of wood is used as the substrate where the artist draws the scene by carefully carving the wood with specialized tools. A thin layer of usually black paint is rolled over the wood substrate, so only the non-carved areas will receive the paint. The image is formed by evenly pressing a usually white paper over the wood block. The carved areas will pass their paint to the paper, forming the black contours of the drawing, while the non-carved areas, which do not have paint, will stay with the same color of the paper. The result is a black-and-white image where the woodgrain is visible and contributes to the appeal of the result. In Fig. 1.1 we present a schematic image with the main steps in woodcut making.

In this work, we present an approach to synthesize virtual woodcuts from natural images. We propose the use of reaction-diffusion (RD) systems as the underlying mechanism to achieve the “feel and look” of woodcuts. Our main contribution is the introduction of RD addressing an artistic technique not yet explored, with visually pleasing results. This thesis is organized as follows: Chapter 2 presents related work found in literature or commercial applications, while in the next chapter we present our methodology, i.e., how our system was implemented. Chapter 4 introduces the results we obtained, followed by the conclusions and future work.

Figure 1.1: Illustrative image showing the pipeline to generate a real woodcut, specifically a Japanese Ukiyo-e. Adapted from: <https://www.nippon.com/en/views/b02306/>. First, a block of wood is carved with the desired image lines (a), producing an engraved woodblock (b). Then paint is rolled on the block (c), in order that only the non-carved areas will receive the paint. After that, a sheet of paper (d) is pressed over the block (e), so the regions with paint (i.e., the non-carved ones) will pass the paint to the paper, generating the final image (f).



.

2 RELATED WORK

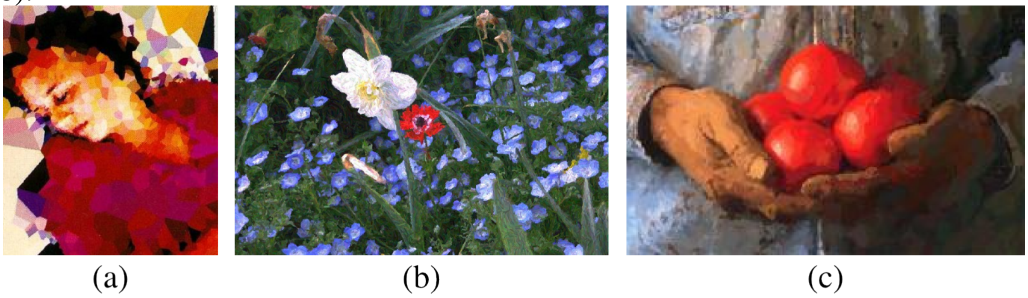
There has been countless approaches to render 2D input into artistic oriented renderings. The pioneer work in the NPR area was the already mentioned Haeberli's work (HAEBERLI, 1990), with a system to semi-automate the generation of a impressionist painting from a photograph (Fig. 2.1a). In this work, a canvas with the same dimensions as the base image can be clicked by the user. Each click corresponds to a brush stroke, in analogy with the brush strokes of a painter in a real canvas, and the image is defined as a collection of brushes, with the following properties: location, color, size, direction and shape. The system automates the selection of color, which comes from the equivalent position of the base image, and direction, calculated with an edge detector over the color gradients of the base image so the strokes are aligned tangentially to its edges. However, his system was not fully automated, needing user input to set the location, size and shape of brushes. This work marked the first NPR technique, the Stroke-Based Rendering or SBR (a review of these methods can be found in (VANDERHAEGHE; COLLOMOSSE, 2013)).

The first fully automated NPR work was Litwinowicz's one in 1997 (LITWINOWICZ, 1997), where pixel locations are randomly sampled at regular intervals instead of placed by the user (Fig. 2.1b). Two additional aspect of his work are to clip strokes against edges in the image to prevent loss of details, and the usage of interpolation for the gradient direction in regions with few or no color variations among pixels, to avoid problematic values for the gradient. A year later, Hertzmann (HERTZMANN, 1998) proposed the use of different brush stroke sizes in accord with the desired detail level of a region - large strokes for less detailed regions, as the background, and small for the areas with more detail, to focus the viewer on the latter (Fig. 2.1c). He did this by comparing successive levels of blur of the original image. Regions where the blurred images did not differ so much in comparison to less blurred versions or the original one are considered less detailed, and so the brush strokes have a larger size. On the other hand, areas where the blur yields a bigger change in the coloration of pixels are more detailed, and receive the smaller brush strokes to paint the details.

The survey by Kyprianidis and colleagues (KYPRIANIDIS et al., 2013) presents a taxonomy of Image-Based Artistic Renderings, as they name this class of techniques. Their focus was on the underlying methods and techniques being used rather than the artistic effect being simulated. They classified these techniques in four main groups:

Stroke-Based Rendering (as the two techniques described above, and comprising most NPR works), Region-based Techniques (techniques based on region segmentation), Example-based Techniques (which use machine learning or other artificial intelligence approaches to generate textures or colors from example images), and Image Processing and Filtering (the usage of filters as the bilateral or the difference of Gaussians to style the images). The chapter by Lai and Rosin (LAI; ROSIN, 2013) presents yet another survey with focus only on techniques working with reduced color palettes, typical of some artistic expressions such as comics, paper-cuts, and woodcuts.

Figure 2.1: Some results from NPR works. (a) Haeberli's the semi-automated painting method, which inaugurated the NPR area (HAEBERLI, 1990). (b) Litwinowicz work, the first fully automated NPR work from (LITWINOWICZ, 1997). (c) Hertzmann's work with different brush stroke sizes according to the required detail level (HERTZMANN, 1998).



Virtual woodcuts have not attracted much attention from graphics researchers. Perhaps because it is a less popular artistic technique than oil and watercolor paintings, or due to the need of simulate the aspect of woodgrain which usually appear in woodcut final results, for instance.

In 1998, Mizuno and colleagues implemented a virtual analog of real woodcuts in a series of works (MIZUNO; OKADA; TORIWAKI, 1998). Their approach is to allow virtual sculpting in a 3D model representing the woodblock, by the user-commanded addition or removal of shapes over its surface, as a direct analog of the carving process in real woodcuts, and later by the application of virtual ink over the woodcut surface (Fig. 2.2a). Their work is perhaps the first to bring the attention to woodcuts in the NPR field. However, their system is not fully automated, needing user input by the virtual carving process, consequently requiring artistic abilities to obtain a convincing visual result.

In a later work (MIZUNO et al., 2000), they improved their system to automate the carving process by using feature extraction over a grayscale image. It is also possible to partially automate this process by using the "computer aided carving", which uses the grayscale image as a basis where the user can select different areas of the image to use specific carving methods, and then the system automatically carves this area based on the user choice. This work also allowed multicolor woodcuts, instead of just grayscale ones.

Later works focused on the generation of *Ukiyo-e*, a traditional Japanese woodcut style (MIZUNO et al., 2002) (MIZUNO et al., 2002). In (MIZUNO et al., 2006), with the use of a pressure sensitive pen and a tablet, their system allowed the user the feeling of carving real wood which could then be virtually printed (Fig. 2.2b).

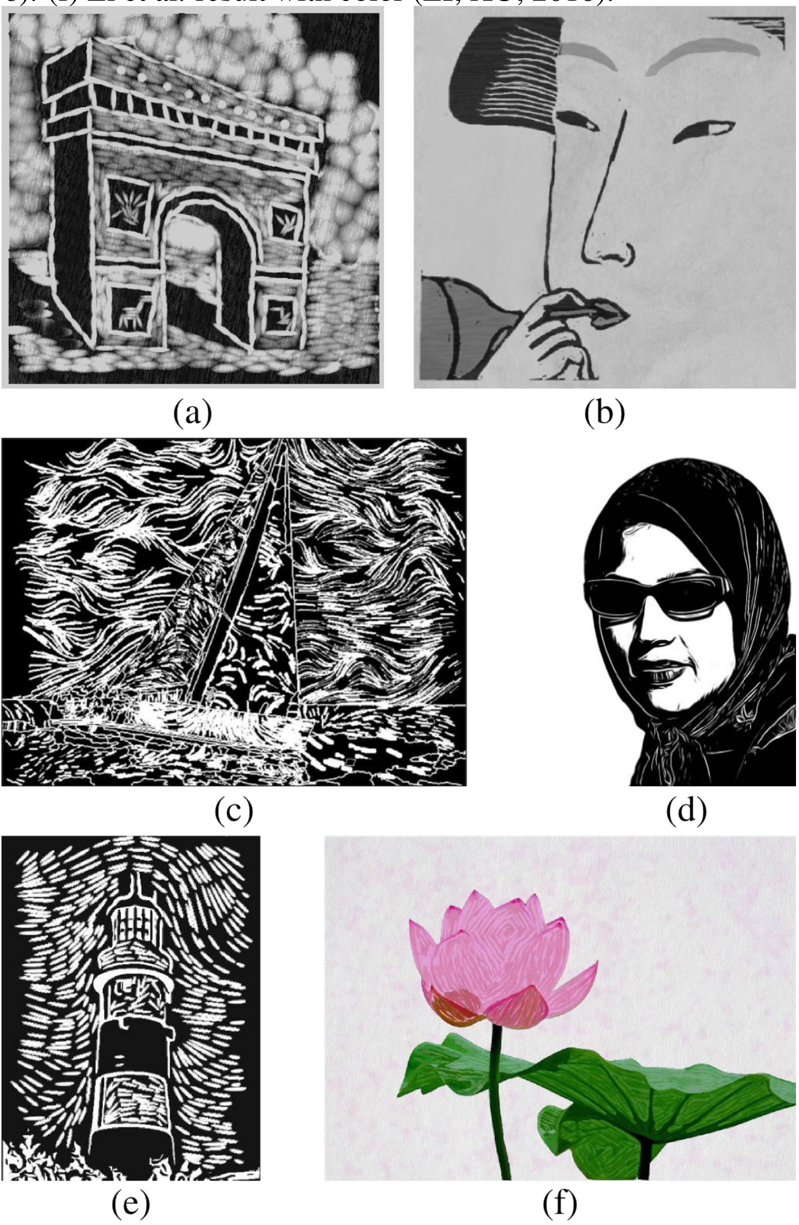
A year later, Mello, Jung, and Walter (MELLO; JUNG; WALTER, 2007) introduced the first approach to synthesize woodcuts from natural images (Fig. 2.2c). Their approach applied a sequence of image processing operations to turn any image into a woodcut, which can be basically divided in four steps: image segmentation (getting a collection of regions corresponding to separated areas of the image), detection of direction fields (the calculation of carving orientation inside each region), generation of woodcuts (distribution of strokes corresponding to the carvings according to the orientation previously calculated), and rendering of woodcuts (the application of an optional smoothing filter to the image). Their results showed some similarity with real woodcuts, but there is few variation on the strokes styles, which resembled short lines.

In 2011 Winnemoller (WINNEMÖLLER, 2011) presented an extension on the Difference-of-Gaussians operator allowing new results for artistic renderings from images (Fig. 2.2d). Although his work did not particularly address woodcuts, he present one result mentioned as similar to woodcuts.

Just recently we have seen new approaches to virtual woodcuts. In 2015 a short paper (LI; XU, 2015) addressed a particular style of color woodcuts, known as *Yunnan* woodcuts. These are color woodcuts that use only a single wood block that is recarved for each color. A year later the same researchers (LI; XU, 2016) extended their previous work, paying particular attention to the color mixing intrinsic to the technique (Fig. 2.2e) and (Fig. 2.2f). In their work, the carving process is simulated by the placement of textures simulating the scores caused by carving the wood. These textures follow the orientation calculated in a manner similar to (MELLO; JUNG; WALTER, 2007), and can have different sizes and orientations.

Outside academic works, there are some tools to synthesize woodcuts from images, mostly commercial ones. Mello (MELLO; JUNG; WALTER, 2007) mentioned Photoshop plugins in their paper, most notably the Xylograph plugin by AmphiSoft (AmphiSoft, 2007), which produces a halftone pattern (Fig. 2.3a). More recently, the Photoshop plugin Simplify 3 Preset List by Topaz Labs (Topaz Labs, 2018) has an option for Wood Carving (Fig. 2.3d). Prisma (Prisma Labs, 2016) is an app for Android and iOS

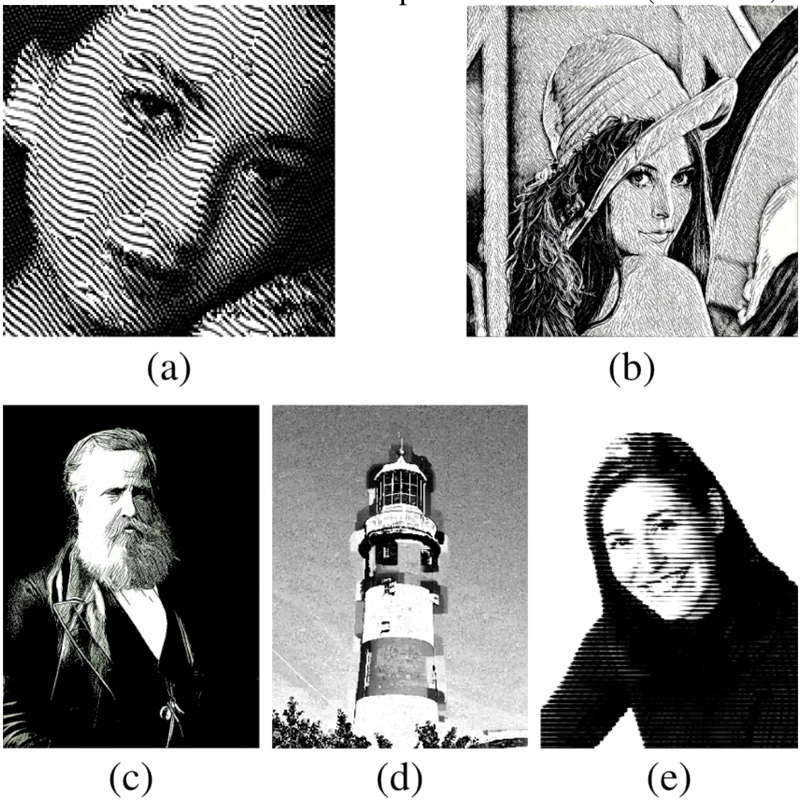
Figure 2.2: Results from previous woodcut synthesizing works. (a) Mizuno's work, the first one to deal with generating of woodcuts (MIZUNO; OKADA; TORIWAKI, 1998). (b) Another result from Mizuno and colleagues, now with the usage of pressure sensitive pen and a tablet (MIZUNO et al., 2006). (c) A result from Mello et al., the first work which automatically synthesize a woodcut from an input image (MELLO; JUNG; WALTER, 2007). (d) Winnemoller woodcut-like result using an extension of Difference-of-Gaussians (WINNEMÖLLER, 2011). (e) Li et al. simulation of woodcuts from Yunnan (LI; XU, 2016). (f) Li et al. result with color (LI; XU, 2016).



(a) (b) (c) (d) (e) (f)

which edits input images in several artistic styles. Two of its styles, Light Summer Reading (Fig. 2.3b) and Heisenberg (Fig. 2.3c), bears some similarity with woodcuts. Lastly, a series of manual operations in the open source software GIMP (WELCH, 2018) promised to generate a woodcut-like image (Fig. 2.3e). However, the resemblance between actual woodcuts and the results of these tools is limited, with traces which did not appear like woodcut strokes. Also for the commercial plugins there is no proper information about the details of its implementation and internal operation.

Figure 2.3: Results from non-academic tools to generate woodcuts. (a) Xylograph Photoshop plugin by AmphiSoft (AmphiSoft, 2007). (b) Prisma app with the Light Summer Reading filter (Prisma Labs, 2016). (c) Prisma app with the Heisenberg filter (Prisma Labs, 2016). (d) Simplify 3 Preset List Photoshop plugin by Topaz Labs (Topaz Labs, 2018). (e) The result of a series of manual operations in GIMP (WELCH, 2018).



Reaction Diffusion was introduced by Alan Turing (TURING, 1952) as a mechanism to explain biological patterns, such as the stripes in a zebra, the spots in a leopard, for instance. Basically, a set of chemical substances react between them, so these substances are synthesized or decomposed, and diffuse in a surface, until they reach a stationary state. Mapping their concentrations to colors can generate patterns of stripes, spots, and labyrinthines.

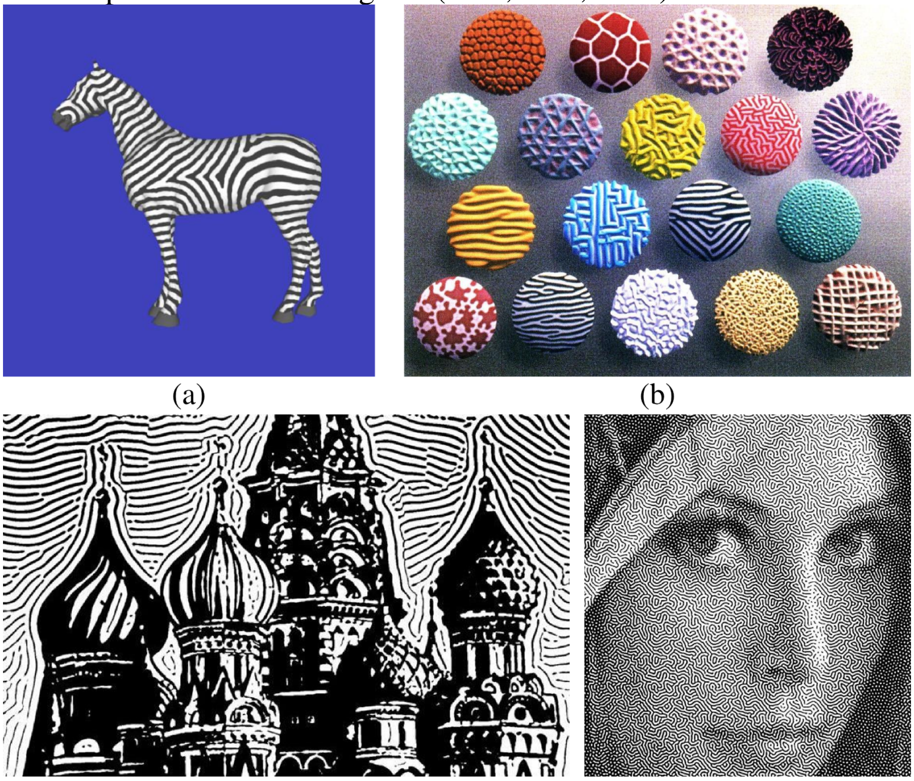
RD has been explored by graphics researchers in many ways. One of the first works in CG with RD was Turk's one (TURK, 1991), where RD was used to model natural markings for mammal coat patterns (Fig. 2.4a). In the same year, Witkin and Kass (WITKIN; KASS, 1991) used RD as a generic pattern generation mechanism (Fig. 2.4b). An interesting aspect of their work was the usage of a rotation matrix to direct the RD stripes. The usage of a rotation matrix was expanded by Sanderson et al (SANDERSON et al., 2006). Kider, Raja, and Badler (JR.; RAJA; BADLER, 2011) successfully applied RD to simulate decaying processes in fruits and vegetables. More recently, RD was the mechanism to simulate skin pigmentation disorders in Sales and Walter work (BARROS; WALTER, 2017).

Although it has been used in many graphics tasks before, just recently RD has been explored in NPR approaches. Chi and colleagues (CHI; LIU; HSU, 2016) explored these patterns for image stylization, achieving many visually interesting results, by the usage of anisotropic RD with an orientation field to guide the pattern formation and shape deformation on the spots, turning them into, for example, triangles and water drops (Fig. 2.4c). The orientation field can be obtained from an input image, allowing the final RD result to retain similarity with the original.

A year later, Jho and Lee (JHO; LEE, 2017) used the RD patterns in a new method of tonal depiction (Fig. 2.4d). In their work, they mapped a parameter from a RD system to color intensity, generating patterns visually lighter or darker according to it. They also added a mask of previously generated RD to allow real-time processing of an image in order to get its tonal depiction. Their results have an interesting resemblance to other methods of tonal depiction, such as halftoning and hedcut. Both these two NPR works using RD have not addressed woodcuts.

In this work, we introduce a new method for synthesizing virtual woodcuts from natural images adapting a RD system to the needs of rendering woodcut-like results. Our visual results compare favorably both with previous work and with commercial image manipulation software.

Figure 2.4: The usage of RD in CG and NPR. (a) Turk's simulation of zebra stripes (TURK, 1991). (b) The patterns obtained by Witkin and Kass (WITKIN; KASS, 1991). (c) Chi and colleagues NPR work with anisotropic RD (CHI; LIU; HSU, 2016). (d) Jho and Lee tonal depiction method using RD (JHO; LEE, 2017).



(a)

(b)

(c)

(d)

3 METHODOLOGY

In this chapter we present the methodology used to generate our woodcuts. Our method is subdivided into three steps: preprocessing, processing (RD) and post-processing (Fig. 3.1), as will be described below.

Briefly speaking, the preprocessing step receives as input an image and returns a parameter map, where each pixel receives a specific value for several parameters used in the processing step. The processing step runs the RD mechanism over the parameter map, setting the overall appearance of the final rendering. Finally, the post-processing step filters the RD result in order to enhance the similarity with a real woodcut image. We describe these steps below.

3.1 Preprocessing

The preprocessing step consists in a Python (PYTHON, 1991) script which uses the OpenCV2 (OPENCV, 2000), Scikit-Image (SCIKIT-IMAGE, 2009) and NumPy (NUMPY, 2006) libraries for image processing functionalities. This script receives a settings file as input, containing all user-defined parameters and the path for all input image files that will be needed in this step (Table 3.1). In the end, the preprocessing step yields a parameter map with specific values for each pixel, to control the RD mechanism. In Table 3.2 we present the parameters used in the preprocessing step, which will be explained throughout the text.

Table 3.1: Types of images which can be loaded by the system in the pre-processing step. First, we have the basic image which will be used as the basis for the final woodcut. This is the only obligatory image. The second image is the segmentation image, where each region has a specific color to separate them from each other. While it is possible to generate a woodcut without segmentation, some options require it, so these will not be available in this case. The other two images, the black and white regions map and the orientation map, are optional images to allow manual setting of, respectively, regions with plain black and white color and the orientation angle of strokes. For more details about how these two images are used, consult the sections 3.1.1 and 3.1.3, respectively

Image	Required
Basic Image	Yes
Segmentation	No
Black and White regions map	No
Orientation Map	No

.

reunir com a figura de workflow

Our algorithm receives as input both an image to be used as the basis for our final

Figure 3.1: Pipeline of our methodology. First, we submit an image and its segmentation to the preprocessing step, to generate a parameter map. This map is then used as basis for the RD process, generating a raw woodcut-like image. Last, the RD result is filtered in a post-processing step in order to improve its appearance, generating the final woodcut image.

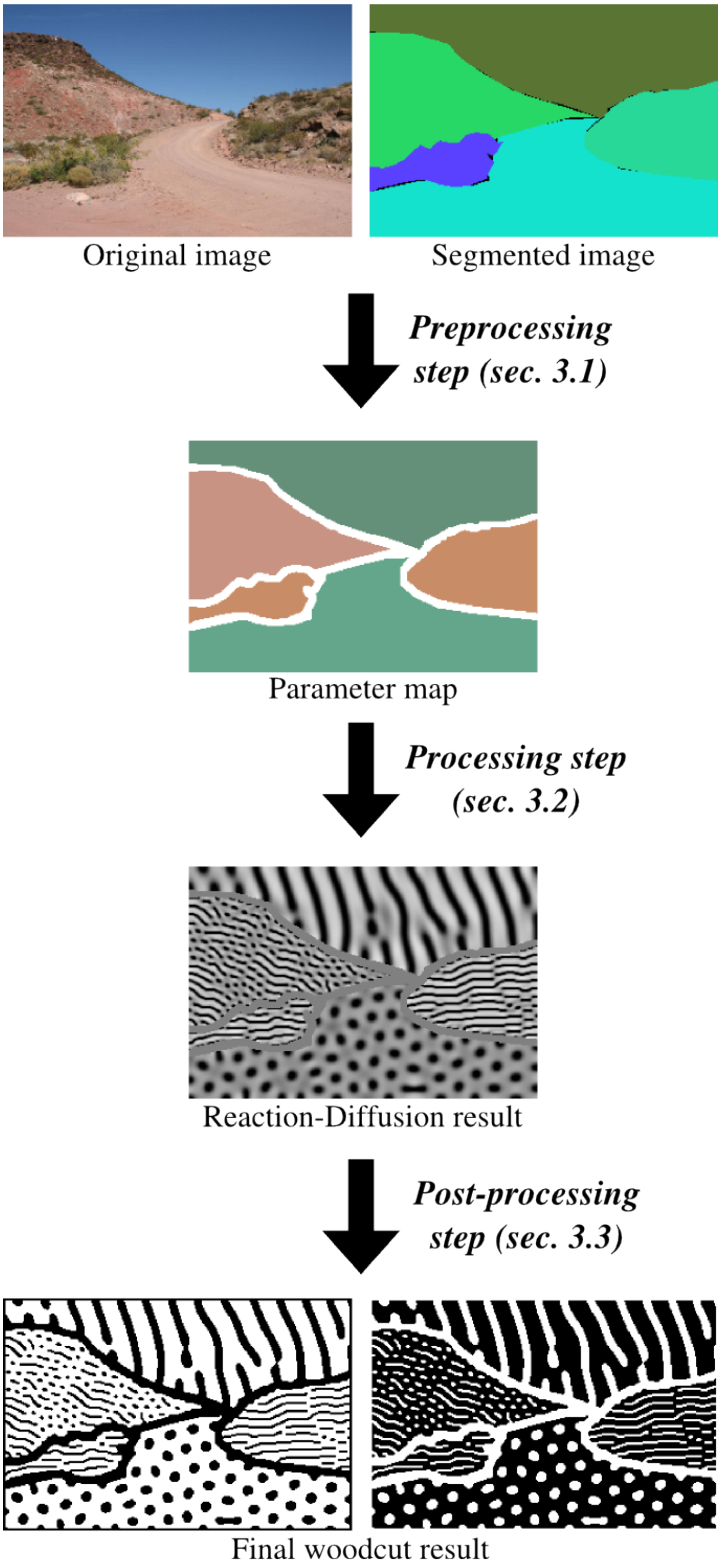

Original image
Segmented image
Preprocessing step (sec. 3.1)
Parameter map
Processing step (sec. 3.2)
Reaction-Diffusion result
Post-processing step (sec. 3.3)
Final woodcut result

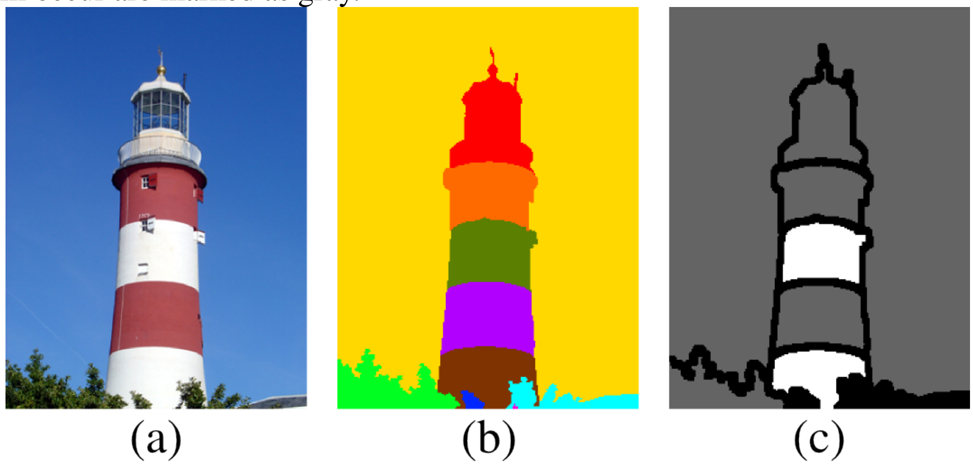
Table 3.2: User-defined parameters for the preprocessing step. Details about each parameter are given in the text.

Parameter	Default Value
Border Color	0
n_{dilate}	1
T_{low}	25
T_{high}	210
T_{size}	500 pixels
k	{1, 5, 9, 13, 17}
T_{diff}	5.0
s_{θ}	0.5

woodcut, through the reading of a second image where each segmented region is colored with a different color. This segmented input image will be defined as a collection of regions. Alternatively, it is possible to use the input image without segmentation. In this case, some steps will be skipped, as will be detailed later. This section presupposes that segmentation is used. The input image is converted to gray scale for all calculations used in our work.

For most of our simulations we used the dataset from ADE20k (ZHOU et al., 2017). This dataset contains $22,210$ images, from landscapes to indoor scenes, being visually adequate for our purposes. Besides, each image has a semantic segmentation. To properly compare our work with previous methods for woodcut generation, we ran simulations using a lighthouse image used in (MELLO; JUNG; WALTER, 2007) and (LI; XU, 2015) (Fig. 3.2a). In this case, we manually generated the segmentation of this image (Fig. 3.2b).

Figure 3.2: Lighthouse image used in our tests. (a) Original image from (MELLO; JUNG; WALTER, 2007). (b) Manual segmentation of this image. (c) Borders and areas without RD, calculated in the preprocessing step, already painted as white and black. Areas where RD will occur are marked as gray.



3.1.1 Black and White Regions

In most real woodcuts, there are many regions without any discernible stroke pattern, being either plain white or plain black. The first step is to discriminate the parts of the image where no RD pattern will be generated, and instead a plain black or white color will be applied. First, from observation of real woodcuts, we define the borders of the segmented regions as black. These borders define the scene and so they should appear in the final woodcut as black strokes. To generate the borders, we use the *mark_boundaries* function from the scikit-image library over the segmented image. This function detects and labels the boundaries between regions, painting these pixels with a specific color to highlight them. It should be noted that our system allows to set the border color as white if desired, through the *Border Color* parameter.

The borders can be thickened to the desired size by a dilate operation (using the dilate function from the OpenCV library), where, for every pixel in a binarized version of the image (where borders are black and other pixels are white), a kernel with dimensions 5×5 is used: if any pixel inside this kernel is black, then the pixel in question is marked as black; otherwise it is white. This operation can be repeated n_{dilate} times. Larger kernels and/or larger values of n_{dilate} result in thicker borders, with smaller regions being absorbed into them.

After the border operations, we set some regions to be either plain white or plain black. To simulate this, our system allow two options: automatic or manual setting of white or black regions. The automatic option follows the logic of (MELLO; JUNG; WALTER, 2007): in real woodcuts the brightest regions are white, while the darkest ones are black, in both cases without any discernible stroke pattern. To reproduce this appearance, we use the same mechanism of their work, where two thresholds, T_{low} and T_{high}, are used. We then mark as white and black any regions whose average gray-scale color intensity is, respectively, higher than T_{high} and lower than T_{low}. The manual option can be used if the user wants more control to the color of each region. In this option, the system reads a gray-scale image where each segmented region is coded as black, white, or gray. Black and white regions will be marked as the respective plain color, while in gray areas RD is allowed. The colors in this image are manually defined by the user, allowing manual choice of plain black and white regions.

If the automatic option is being used, the next step is to merge small regions into the borders, to reduce unnecessary noise. For that, we first need to count the pixels inside each region after border dilation, so we can compare the actual size that that region would have in the final image.

We could have simply counted the number of pixels of each color using the OpenCV histogram function, but by doing that we would be ignoring the cases when a thickening of the borders would separate a segmented region into two new regions (Fig. 3.3a and 3.3b). Thus, we need to detect and label each region, task done by the Algorithm 1, which receives as input an image map where each border pixel (calculated after dilation in the previous steps) is white and other pixels are black, and returns another image map where each region separated from other regions by borders have a specific color, and borders continue to be color-coded as white.

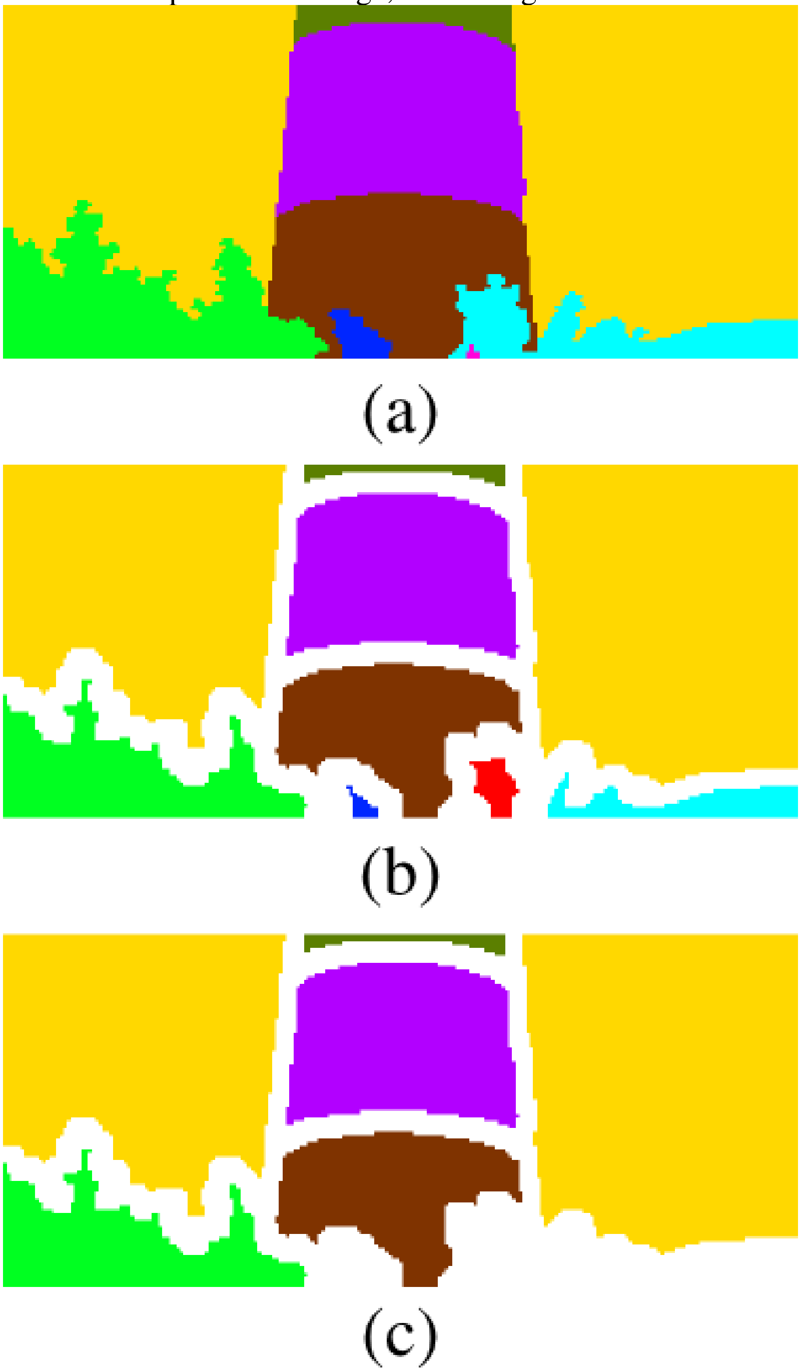
After running Algorithm 1, we have an image map where each region separated by borders have a specific color (Fig. 3.3c). We now simply apply a color histogram to get the number of pixels of each region (using the Numpy function *bincount* (NUMPY, 2006)) and, finally, we merge the smallest regions, whose area (i.e., the number of pixels) is smaller than a given threshold, T_{size}, into the borders, by marking these regions as white (the same color of borders in these image maps). This image map will now be applied into the black and white regions calculated from the previous step, so the white pixels in this map will be marked as the color defined by the *Border Color* parameter, making the small regions merge with the border.

After all these calculations, the program saves the image with the black and white regions color-coded as such, and regions which will receive RD painted as gray. Fig. 3.2c shows an example of this image. It will be used in the post-processing step.

3.1.2 Detail Level and Size of Strokes

To allow a variation of detail levels for different parts of the image, we compute, for each pixel, a value for a RD parameter, S, which is related to the size of structures generated by this mechanism (BARD; LAUDER, 1974) (MESQUITA; WALTER, 2017). We adapted the method from Hertzmann (HERTZMANN, 1998) for this task. In that work, different detail levels in a painting are simulated by using distinct brush sizes in sequential steps, such that regions where fine detail is needed are painted with smaller brushes than other parts of the image, drawing more attention toward these regions. In our work, more detailed areas receive a higher value of S, resulting in smaller structures and thus increased preservation of details.

Figure 3.3: Behavior of small regions (using Fig. 3.2 as example). (a) Detail of manual segmentation. (b) Detail after applying border and dilating it. Dilating alone was able to absorb the small purple region at the bottom of the (a) image. The cyan region was divided into two regions by the border thickening, marked as cyan and red in (b). (c) Detail after the operation of small region merge. The regions marked in blue, red and cyan, at the lower bottom part of the image, were merged into the border.



(a)

(b)

(c)

Algorithm 1 Merging of Small Regions Into Borders

Input: Image M, defined as a bidimensional array with $w \times h$ dimensions, where w is its width and h its height. In this image, each border pixel is white ($M_{i,j} = 255$) and each non-border pixel is black ($M_{i,j} = 0$).

Output: Image I with the same dimensions as M where each region is labeled by a different color, and borders are labeled as white ($I_{i,j} = 255$).

```
procedure MERGESMALLREGIONSINTOBORDERS($M_{w \times h} \rightarrow I_{w \times h}$)

    $I \leftarrow M$                                    ▷ //Copy $M$ as $I$
    for $i \leftarrow 0$ to $h$ do
        for $j \leftarrow 0$ to $w$ do
            if $I_{i,j} \neq 255$ then                  ▷ //$I_{i,j}$ is not a border pixel
                if $i > 0 \wedge I_{i-1,j} \neq 255$ then    ▷ //The top neighbour $I_{i-1,j}$ is inside image limits and is not a border pixel
                    $I_{i,j} \leftarrow I_{i-1,j}$      ▷ //Set $I_{i,j}$ with the same color as its top neighbour
                    if $j > 0 \wedge I_{i,j-1} \neq 255 \wedge I_{i,j-1} \neq I_{i,j}$ then    ▷ //Left neighbour $I_{i,j-1}$ is inside image limits, is not a border pixel, and has a different color code than $I_{i,j}$
                        ▷ //Merge the regions with the same color as the pixel $(i, j)$ and its top neighbour into one

                        for all $pinIwherep = I_{i,j-1}$ do
                            $p \leftarrow I_{i,j}$▷ //Mark all pixels with the same color of the pixel $(i, j-1)$ as the color of the $(i, j)$ pixel. In other words, all these pixels will have the same label and so belong to the same region.
                        end for
                    end if
                else if $j > 0 \wedge I_{i,j-1} \neq 255$ then    ▷ //The left neighbour $I_{i,j-1}$ is inside image limits and is not a border pixel
                    $I_{i,j} \leftarrow I_{i,j-1}$      ▷ //Set $I_{i,j}$ with the same color as its left neighbour
                else
                    $I_{i,j} \leftarrow generateNewColor()$    ▷ //Set the pixel $(i, j)$ with a color which is not present in the image
                end if
            end if
        end for
    end for
    return $I$
end procedure
```

To implement this feature, we defined the following parameters: a list of integers k, and an error threshold T_{diff}. Due to the form of communication between the preprocessing and the processing parts of our pipeline (the preprocessing yields an image where data is encoded in the color of pixels, and the processing reads this image and translate it to parameters usable to control the RD process), and to allow the reuse of the same preprocessing map to tests with different RD parameters, in this step we only calculate the *relative* S between different regions. So 1 correspond to S_{max} and 0 to S_{min}, as will be defined in the processing step, and each pixel will have a value between 0 and 1 which we denote as s. In other words, for this step we only need to know how each region differs one from another. In comparison with Hertzmann's method, we do not need to define the exact size of each brush size, only the proportions between them.

First, we mark all pixels as 1 (which correspond to S_{max}). Then, for each integer k_i in the list of integers k, starting from the smallest one, we generate a reference image by performing a Gaussian blur using a $k_i \times k_i$ kernel over the original input image (noting that larger values for k_i result in more blurred images, and so more different than the original). This is done using the OpenCV GaussianBlur function with $sigmaX = 0$ (OPENCV, 2000). For every pixel, we compute the average pixel-to-pixel difference between the reference image and the original one for the entire neighborhood of this pixel, in a window with the same dimensions as the blur kernel. If the area difference is smaller than T_{diff}, we consider that the area around this pixel did not change too much between the blurred image and the original one, being consequently less detailed. We mark this pixel as the value of s corresponding to this kernel, calculated as:

$$S_i = 1 + m * (k_i - \min k)$$

$$m = \frac{1}{\min k - \max k}$$

After computing s for every pixel, we set a unique s for each region by computing the average s inside that region , if a segmentation image was provided. For that, we use the NumPy mean function (NUMPY, 2006) over all pixels which share the same label (color). By doing that, we ensure that inside a region all strokes will have the same size, improving structure size coherence for the result.

The parameter T_{diff} affects the average level of detail and size of strokes: smaller values for T_{diff} reduce the chance of every step to set a smaller value for s in the comparison between the original image and the blurred one, resulting on larger values for s and consequently smaller strokes, and a more detailed final result.

3.1.3 Orientation of Strokes

After that, we need to calculate the orientation of strokes. In our work, we have four options: a) orientation by region, b) orientation by pixel, c) adaptive orientation, and d) orientation by image.

In the orientation by region, every pixel inside a given region will have the same orientation, so strokes will be parallel to each other inside the same region. As (MELLO; JUNG; WALTER, 2007) observed, the wood carvings tend to initiate in the brighter parts of a region, so for every region we will start the strokes from the brighter point near region borders. First, for each border pixel in the image, we verify the region of each of its immediate neighbors (the top, bottom, left and right pixels) by their color label in the segmented image, generating for each region a list of pixels near to the region border. Then, for each of these pixels, we test the average luminance for a 5×5 window centered on it (i.e., the average color intensity, from the original input image, among all pixels inside this window). If this window surpass the image limits, we simply test for all pixels available, ignoring positions outside the image. For each region, we retrieve the pixel with the highest average luminance inside its corresponding window. Then, we compute the center pixel of every region as the average of the horizontal and vertical positions of every pixel in that region. To find the orientation angle of a region we compute the slope of the line formed by these two pixels: the border pixel with the largest average luminance in its surroundings and the central pixel, as follows:

$$\theta_{reg} = \arctan \frac{d_y - c_y}{d_x - c_x}$$

where θ_{reg} is the orientation of strokes for a given region, d is the boundary pixel and c is the central pixel of a given region.

When calculating orientation by pixel, it is possible to set different pixel-specific values for θ, which allows the contour of small details to appear in the resulting image. Our method for such calculation is based on local orientation of luminance as used by (MELLO; JUNG; WALTER, 2007) and (LI; XU, 2016). First, for each pixel (i, j) in the image, we compute the local average orientation of luminance Θ in a 5×5 window W centered in that pixel as follows:

$$\Theta_{i,j} = \frac{1}{2}\tan^{-1}\left(\frac{V_x(i,j)}{V_y(i,j)}\right)$$

where

$$V_x(i,j) = \sum_{(u,v)\in W} 2G_x(u,v)G_y(u,v)$$

$$V_y(i,j) = \sum_{(u,v)\in W} (G_x(u,v)^2 - G_y(u,v)^2)$$

and G_x and G_y are the image gradients in the x and y directions, calculated using the Sobel operator (obtained from the Sobel function of the OpenCV library (OPENCV, 2000) with first order derivatives). If the window surpass the image limits, we only add the available pixels inside the image, as we did in the orientation by region option.

To orient the strokes preserving the contour details of the original image, their orientation should be perpendicular to the local average orientation of luminance Θ. This happens because Θ follows the direction of color gradients, i.e., it goes from black areas to white areas or vice-versa, being thus perpendicular to any existing line in the image (since the lines have more or less the same color). So, the stroke orientation θ for a pixel is given by the following formula:

$$\theta_{i,j} = \frac{\pi}{2} + \Theta_{i,j}$$

While more complex objects as an human body or decorated buildings would need a finer level of detail, in background areas such as the sky a single orientation works better. This way the user would focus on the foreground instead of the background. The adaptive orientation method allows us to use both methods, orientation by region or by pixel, for different areas of the image, depending on the detail level required by that region. This detail level was encoded by the value of S previously calculated for that region. For a given region with S value equals to S_{region}, if $S_{region} > S_\theta$ (an user-defined threshold), then that region will use orientation by pixel; otherwise, the region will use orientation by region. Thus, regions with higher values for S, and consequently more details in the original image, will have these details preserved in the final woodcut, using a pixel-specific orientation. Less detailed regions will have a single orientation, enhancing their status as background regions.

If the user wants to manually control the orientation of strokes in the image, then the orientation by image option allows the loading of an image map, where the color of each pixel corresponds to an orientation angle. This option should be used together with one of the other three, since the image map allows the setting of parts of image to use another method, making it possible to have both manual and automatic orientation calculation in the same image. The image is converted to the HSV color space, which will be used in our calculations. For each pixel, if saturation is at its maximum value (1.0), then we read the hue property (which goes from 0 to 1) and translate it to an angle between 0 and π, using the following linear formula:

$$\theta_{i,j} = \pi h_{i,j}$$

where $h_{i,j}$ is the hue property for the pixel (i, j). Pixels with saturation lower than 1.0 will have their orientation calculated through one of the other methods, as defined by the user. It should be noted that for our system two opposite angles (i.e., θ and $\theta + \pi$) result in the same direction for the strokes, so it's not necessary to consider angles larger than π when reading the image map.

3.1.4 Stripes versus Spots

Many real woodcuts show, besides lines and flat color areas, dots or other similar structures. In our RD system, it is possible to control which regions will have stripes or spots. We simulate this pattern diversity by setting different values of a parameter called δ for every region. We considered brighter areas to have spots and darker ones to have stripes, using the average color of each region to compute this parameter, by the following formula:

$$\delta_{region} = a_{region}$$

where a_{region} is the average gray-scale intensity for that region. Both values vary from 0 to 1. We will describe in the next section how this parameter works, when we will detail the main engine of our system, the RD process.

3.1.5 The Parameter Map

At the end of the preprocessing step, we save an image with the same dimensions of the original to be used as input for the RD process, where different color channels are used to set the local value of different parameters calculated in this step as follows: red for s, green for δ, and blue for the orientation angle θ. Also, the value corresponding to plain white (255 for all channels) is interpreted as absence of RD for that pixel. To encode these parameters into a RGB image, we translate them to integers inside a given interval corresponding to a color channel. For red and green, this interval is $[100, 200]$, and the conversion is done by linearly translating the domain of the corresponding values, i.e., $[0, 1]$, to this interval, which is done by the following formulas:

$$R_{i,j} = 100 + 100 s_{i,j}$$

$$G_{i,j} = 100 + 100 \delta_{i,j}$$

The orientation angle θ will be converted to the blue channel integer B inside the interval $[100, 172]$ by the following formula:

$$B_{i,j} = 100 + 36\left(\frac{\theta_{i,j}}{\pi} + 1\right)$$

In this formula, each increment in B correspond to an increase of 5 degrees in the orientation angle. This image is saved in the PPM (Portable Pixmap Format) format (POSKANZER, 1988), one of the Netpbm image formats, which stores the image as a text file. PPM allows images with the three RGB channels. The details of how the RD process reads this parameter map into usable values will be treated in the next session.

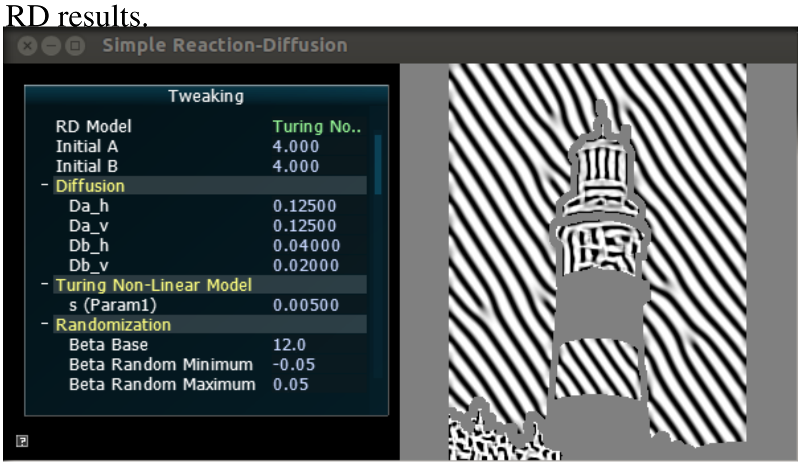
Alternatively, it is possible to use an image without segmentation, for cases where a segmentation is not available. In this case, there will be no borders or regions without RD, θ will necessarily be defined by pixel or by image map reading (orientation by region or adaptive are not available), and each pixel will have a different value for S, since there will be no region to compute the average value for this parameter. Also, δ will be defined by the pixel color, and not by the region average color.

Não poderia segmentar no pre-processamento?

3.2 Processing

In the processing step we generate the pattern of strokes using RD. Our RD simulator was developed by expanding a previously simulator developed in Prof. Marcelo Walter's laboratory, the *Simple Reaction-Diffusion*, which was modified to suit the needs of this work. This simulator was developed in C (C, 2018) with the following libraries: (OPENGL, 1992) for the rendering of the results, and (ANTTWEAKBAR, 2005) to add a graphical interface with options to manage our system (Fig. 3.4).

Figure 3.4: Graphical interface of the RD processing program (without the preprocessing and post-processing steps). To the left, there is a menu where the user can set the value for the processing parameters and options. This menu also contains options to operate the system (these options are also accessible by keyboard buttons). To the right is the area to visualize the RD results.



```
RD Model                    Turing No..
Initial A                   4.000
Initial B                   4.000
- Diffusion
   Da_h                     0.12500
   Da_v                     0.12500
   Db_h                     0.04000
   Db_v                     0.02000
- Turing Non-Linear Model
   s (Param1)               0.00500
- Randomization
   Beta Base                12.0
   Beta Random Minimum      -0.05
   Beta Random Maximum      0.05
```

The RD system used in our work is the non-linear system defined by (TURING, 1952) and later used by (BARD, 1981) (BARD; LAUDER, 1974). In Table 3.3 we present the parameters defined in the processing step. All of these parameters can be set by the user using the interface during execution. This system is expressed by the set of partial differential equations below:

$$\frac{\partial a}{\partial t} = S(16.0 - ab) + D_a \nabla^2 a$$

$$\frac{\partial b}{\partial t} = S(ab - b - \beta) + D_b \nabla^2 b$$

Here, a and b represent the concentration of morphogens (the substances responsible for generate patterns in Turing's theory), $\partial a / \partial t$ and $\partial b / \partial t$ the respective rates of variation in time, $\nabla^2 a$ and $\nabla^2 b$ the Laplacian representing the spatial variation of the morphogens (so

generating

they can diffuse from areas of higher concentration to areas of lower concentration). D_a and D_b the diffusion coefficients (i.e., the speed at which diffusion takes place), and S a constant representing the speed of chemical reactions (and, as already mentioned, related to the size of structures generated by the RD process). β is added to generate the initial spatial heterogeneity in order to start the pattern formation process, as we will explain below.

Table 3.3: User-defined parameters for the processing step. Details about each parameter are given in the text

Parameter	Default Value
D_{ah}	0.125
D_{av}	0.125
D_{bh}	0.030
D_{bv}	0.025
t	16,000
a_0	4.0
b_0	4.0
S_{min}	0.005
S_{max}	0.015
δ_{min}	0.5
δ_{max}	1.5
M_{white}	8.0
M_{black}	0.0

To solve this system numerically, we discretized it spatially using a grid of pixels, and temporally with discrete steps. The changes of a and b for a specific pixel in a single step are given by the following equations:

$$\Delta a_{i,j} = S(16.0 - a_{i,j}b_{i,j}) + D_a\gamma_a$$

$$\Delta b_{i,j} = S(a_{i,j}b_{i,j} - b_{i,j} - \beta_{i,j}) + D_b\gamma_b$$

$$\gamma_a = a_{i-1,j} + a_{i+1,j} + a_{i,j-1} + a_{i,j+1} - 4a_{i,j}$$

$$\gamma_b = b_{i-1,j} + b_{i+1,j} + b_{i,j-1} + b_{i,j+1} - 4b_{i,j}$$

$$a \geq 0, b \geq 0$$

where i and j represent respectively the vertical and the horizontal position of the pixel in question. γ_a and γ_b represent the diffusion component, already discretized, and the other terms of the equations represent the reaction component.

Each pixel has a different value for β, generated randomly during initialization by a uniform distribution which vary between 11.95 and 12.05. This uniform deviate receives a seed to the random number generation, so we can reproduce a given pattern obtained. The value of β doesn't suffer any alteration during the processing.

The total number of steps can be defined by the user as the parameter t. In the initialization, all pixels receive the same values for the morphogens a and b, a_0 and b_0 respectively. The diffusion component for a morphogen can be written as a kernel. In this case, the matrix corresponding to this kernel is the following:

$$D = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

As we described before, a parameter map, computed in the preprocessing step, is loaded into the RD system to allow different behaviors for each part of the image, particularly regarding the size of stripes and spots (S), distinction between spots and stripes (δ), orientation of stripes (θ), and absence of RD (morphogens will not be generated neither travel from or to a given pixel).

To load the parameter map, our program reads the PPM file generated in the previous step, and interprets the color of each pixel to parameters used in our system. Since we defined the minimum and maximum ranges for S and δ as 100 and 200, we translate these numbers by using the corresponding minimum and maximum parameters, as defined by the user (Table 3.3), through the following formulas:

$$S_{i,j} = S_{min} + \frac{(S_{max} - S_{min})(R_{i,j} 100)}{100}$$

$$\delta_{i,j} = \delta_{min} + \frac{(\delta_{max} - \delta_{min})(G_{i,j} 100)}{100}$$

where $S_{i,j}$ and $\delta_{i,j}$ are the values of S for the pixel (i, j), and $R_{i,j}$ and $G_{i,j}$ the RGB values for the red and green channel for this pixel. So δ will vary between δ_{min} and δ_{max}, and S between S_{min} and S_{max}.

As it was stated in Sec. 3.1, the orientation angle θ was converted to the interval between 100 and 172, in intervals of 5 degrees. So, to retrieve its value, we use the following formula:

$$\theta_{i,j} = 5(B_{i,j} - 100)$$

where $\theta_{i,j}$ is the orientation angle in degrees for the pixel (i, j) and $B_{i,j}$ the RGB value for the blue channel for this pixel.

By setting a specific value of S for each pixel, we can have areas with different sizes for the patterns. So when computing a step of the RD simulation for a given pixel (i, j), we consider the value of S as equal to $S_{i,j}$, i.e., the specific value for that pixel.

The RD system described above is isotropic regarding the diffusion (diffusion occurs in all directions with the same intensity). To properly control the direction of stripes and thus orient the resulting pattern, we modified the diffusion term as in (WITKIN; KASS, 1991), using the parameter map for θ.

Não entendi se o RD é usado como caixa-preta ou se tudo foi implementado por ti.

In that work, anisotropy is added by first setting two different diffusion coefficients for each axis in the bidimensional space, which allows the pattern to be elongated in the vertical or horizontal direction. So we have D_{ah} and D_{av}, respectively the horizontal and the vertical diffusion coefficient for a, and D_{bh} and D_{bv}, the horizontal and the vertical diffusion coefficient for b. Particularly, if the diffusion coefficient for the morphogen b is sufficiently larger in an axis than in another, parallel stripes are formed, a pattern common in many wood carving images. These stripes are parallel to the axis where the diffusion coefficient is larger, so if $D_{bh} > D_{bv}$, we get horizontal stripes, otherwise we get vertical stripes.

To allow this stretching in any direction, the diffusion directions are rotated by an angle. If we set D_{bh} to be higher than D_{bv}, then stripes will be parallel to the horizontal axis. Rotating this system by an angle will also rotate the stripes by the same angle, so it is possible to control the orientation of stripes.

For the morphogen a in a given pixel with coordinates (i, j), the rotated diffusion matrix to be used as the diffusion kernel is the following:

$$D_{ij} = \begin{bmatrix} -f_{12} & 2f_{22} & f_{12} \\ 2f_{11} & -4f_{11} - 4f_{22} & 2f_{11} \\ f_{12} & 2f_{22} & -f_{12} \end{bmatrix}$$

$$f_{11} = D_{ah} \cos^2 \theta_{i,j} + D_{av} \sin^2 \theta_{i,j}$$

$$f_{12} = (D_{av} - D_{ah}) \cos \theta_{i,j} \sin \theta_{i,j}$$

$$f_{22} = D_{av} \cos^2 \theta_{i,j} + D_{ah} \sin^2 \theta_{i,j}$$

D_{ah} and D_{av} are the diffusion coefficients of a in the horizontal and vertical direction, respectively, and $\theta_{i,j}$ is the orientation angle for the pixel (i, j), provided by the parameter map calculated in the preprocessing step. An analog matrix is used for the morphogen b. For more details regarding the mathematical formulation of this anisotropic RD system, consult (WITKIN; KASS, 1991).

To form both spots and stripes in the same image, we use the parameter map of δ. If the difference between the diffusion coefficients of b in a given direction is not so large, the stripes are broken after an interval. Reducing even more this difference can result in spots slightly stretched in the direction of the larger diffusion coefficient. Consequently, by allowing different areas of the image to have a relatively larger or smaller value for this difference, it is possible to change the pattern obtained for each area. For each pixel, this information is encoded in the input image as the parameter δ, which is interpreted by the system as a floating point number, varying between δ_{min} and δ_{max}, to be multiplied by D_{bv} (the vertical diffusion coefficient of b) for that pixel. Since in our simulations we always have $D_{bh} > D_{bv}$ (so the stripes are parallel to the orientation angle), values larger than 1 for δ tend to form spots or broken stripes, while values smaller than 1 result in continuous, parallel lines. An increase in the difference between δ_{min} and δ_{max} allows the regions to show more varied types of patterns, while a small difference makes all regions to have the same kind of structure. It should be taken into account that if $\delta D_{bv} > D_{bh}$ the direction of patterns will be inverted (so horizontally-aligned spots and stripes became vertically-aligned and vice-versa). To avoid that, we limit $\delta D_{bv} > D_{bh}$ to have a maximum value equal to D_{bh}.

After these alterations, the equations 3.1 which govern the variation of a and b for each step in a given pixel (i, j), became the following:

$$\Delta a_{i,j} = \begin{cases} S_{i,j}(16.0 - a_{i,j}b_{i,j}) + \gamma_a & \text{if } n_{i,j} \neq 0 \\ 0 & \text{if } n_{i,j} = 0 \end{cases}$$

$$\Delta b_{i,j} = \begin{cases} S_{i,j}(a_{i,j}b_{i,j} - b_{i,j} - \beta_{i,j}) + \gamma_b & \text{if } n_{i,j} \neq 0 \\ 0 & \text{if } n_{i,j} = 0 \end{cases}$$

where $n_{i,j}$ is equal to 1 for pixels with RD, and 0 to pixels without it, and γ_a and γ_b represent the diffusion component, which is calculated through the following equations:

$$\gamma_a = \sum_{k=1}^{3}\sum_{l=1}^{3} \begin{cases} d_{a_{k,l}}^{i+k-2,j+l-2} a_{i+k-2,j+l-2} - d_{a_{2,2}}^{i,j} a_{i,j} & \text{if } n_{i+k-2,j+l-2} \neq 0 \\ 0 & \text{if } n_{i+k-2,j+l-2} = 0 \end{cases}$$

$$\gamma_b = \sum_{k=1}^{3}\sum_{l=1}^{3} \begin{cases} d_{b_{k,l}}^{i+k-2,j+l-2} b_{i+k-2,j+l-2} - d_{b_{2,2}}^{i,j} b_{i,j} & \text{if } n_{i+k-2,j+l-2} \neq 0 \\ 0 & \text{if } n_{i+k-2,j+l-2} = 0 \end{cases}$$

where, for a given pixel (i^*, j^*), $d_{a_{k,l}}^{i^*,j^*}$ is the element in the k-th row and the l-th column of the following matrix:

$$D_a^{i^*,j^*} = \begin{bmatrix} -f_{a_{12}} & 2f_{a_{22}} & f_{a_{12}} \\ 2f_{a_{11}} & -4f_{a_{11}} - 4f_{a_{22}} & 2f_{a_{11}} \\ f_{a_{12}} & 2f_{a_{22}} & -f_{a_{12}} \end{bmatrix}$$

$$D_b^{i^*,j^*} = \begin{bmatrix} -f_{b_{12}} & 2f_{b_{22}} & f_{b_{12}} \\ 2f_{b_{11}} & -4f_{b_{11}} - 4f_{b_{22}} & 2f_{b_{11}} \\ f_{b_{12}} & 2f_{b_{22}} & -f_{b_{12}} \end{bmatrix}$$

with the following values:

$$f_{a_{11}} = D_{ah}\cos^2\theta_{i^*,j^*} + D_{av}\sin^2\theta_{i^*,j^*}$$

$$f_{a_{12}} = (D_{av} - D_{ah})\cos\theta_{i^*,j^*}\sin\theta_{i^*,j^*}$$

$$f_{a_{22}} = D_{av}\cos^2\theta_{i^*,j^*} + D_{ah}\sin^2\theta_{i^*,j^*}$$

$$f_{b_{11}} = D_{bh}\cos^2\theta_{i^*,j^*} + \delta_{i,j}D_{bv}\sin^2\theta_{i^*,j^*}$$

$$f_{b_{12}} = (\delta_{i,j}D_{bv} - D_{bh})\cos\theta_{i^*,j^*}\sin\theta_{i^*,j^*}$$

$$f_{b_{22}} = \delta_{i,j}D_{bv}\cos^2\theta_{i^*,j^*} + D_{bh}\sin^2\theta_{i^*,j^*}$$

and the following restrictions, which are applied to the values of a and b for all pixels in our image:

$$a \geq 0$$

$$b \geq 0$$

$$\delta_{i,j}D_{bv} \leq D_{bh}$$

These equations are a summary of the alterations described in the text. First, there is no variation of a and b in pixels without RD (i.e., pixels where $n_{i,j} = 0$). Note that in the altered equations each pixel (i, j) has its own value for S, denoted by $S_{i,j}$.

γ_a and γ_b, which represent the diffusion, are a sum of the individual diffusion between the pixel (i, j) and all of its neighbors $(i+k-2, j+l-2)$, including diagonal ones. Since k and l go from 1 to 3, the neighbors are the pixels inside the intervals $[i-1, i+1]$ and $[j-1, j+1]$, forming a 3x3 kernel centered in the pixel (i, j). Diffusion to diagonal neighbors occurs due to the rotation of the horizontal and vertical axes. There is also no diffusion from or to pixels (i^*, j^*) with n_{i^*,j^*}, so neighbouring pixels with RD just ignore these during their calculation of the diffusion term. If these positions are outside the image limits, they are ignored too (there is no diffusion outside the image or circular boundary condition in our system).

The diffusion involving the pixel (i, j) and each of its neighbors comes from Witkin and Kass diffusion kernel (Eq. 3.2), with one alteration: we multiply D_{bv} by $\delta_{i,j}$ to change the relative diffusion rate between the horizontal and vertical axes, to obtain stripes or spots in different points, as we explained before when explaining the parameter δ.

To save the result of the RD simulation into an image, we need to translate the morphogen concentrations into a color intensity. Our system allows two options for that: dynamic and static. In the dynamic option, the maximum concentration for the entire grid is treated as plain white (255 in RGB), while the minimum concentration is considered as plain black (0 in RGB). Intermediate values are linearly interpolated from these extremes, generating shades of gray. The static option introduces two limits, M_{white} and M_{black}, so any pixel with a morphogen concentration above M_{white} will be translated as white, while the ones with a concentration below M_{black} will be painted as black, and values between will be linearly interpolated. This option is useful in certain simulations where some points have a very high morphogen concentration, leading most of the image to be saved as dark gray, reducing the contrast and disallowing the formation of a pattern during post-processing binarization (see Sec. 3.3 below).

It is also possible to choose which morphogen will be converted to the color intensities, or even a mathematical combination of their values. Our program has the following options: a, b, $|b-a|$ and $a+b$. By default, our system translates the value of b to color.

The resulting image will be saved as a PNG image file, which will be submitted to the post-processing operations.

3.3 Post-Processing

To enhance the similarity with real woodcuts, we post-process the resulting image from the processing step. As preprocessing, post-processing is done by a Python (PYTHON, 1991) script which uses functions from the OpenCV (OPENCV, 2000) and NumPy (NUMPY, 2006) libraries. It receives as input a PNG image from the processing step. The list of parameters for the post-processing step is given in the Table 3.4.

Table 3.4: User-defined parameters for the post-processing step. Details about each parameter are given in the text

Parameter	Default Value
σ_{Gauss}	50
$\sigma_{Poisson}$	6
$\lambda_{Poisson}$	5
$N_{Contoli}$	3,000
$Color_{Contoli}$	40
$l_{Contoli_{min}}$	2
$l_{Contoli_{max}}$	8

First, we recover from the preprocessing step the information about plain black and plain white regions, where RD did not occur, painting these areas accordingly. This is done by reading the image with black and white regions generated after calculation borders, plain color regions and merging of small regions (Fig. 3.2c), and, for each pixel in this image, if the pixel is black or white, its corresponding pixel will be painted as black or white. Gray pixels in the black and white image map will maintain its color from the RD processing.

To reduce noise, we apply a median filter with a 3×3 kernel over the image. For each pixel in the image, we retrieve all of its neighbors, including the diagonal ones, forming a 3×3 kernel centered on it. We then apply the NumPy median function (NUMPY, 2006)to get the median value among the colors of all pixels in this kernel, which is applied to the pixel. Pixels in the image limit (and so unable to have a complete 3×3 kernel) are instead painted as black, similar to a picture frame.

After that, to get the aspect of an actual woodcut with white background and black strokes, the median filter result is submitted to thresholding using Otsu's technique (OTSU, 1979), resulting in a black and white binary image. OpenCV threshold method

(OPENCV, 2000) is used with the *THRESH_OTSU* option.

To generate a white-line woodcut, with a dark background and white strokes, we simply invert the black and white pixels generated by the Otsu binarization (i.e. the negative of the image), using the OpenCV *bitwise_not* function (OPENCV, 2000). which inverts every bit in the image.

After these last steps, our woodcut is done. However, we have some optional filters which can improve similarity with real woodcuts for certain cases.

According to Mello et al. (MELLO; JUNG; WALTER, 2007), real woodcuts are not a binary, black and white image, but contain some tones of gray, particularly in the borders of strokes, due to imperfections occurred in the carving and the printing process. To simulate this, an optional filter to smooth the resulting image, as introduced in Mello's et al. work (MELLO; JUNG; WALTER, 2007), can be applied to the result of Otsu binarization. This filter was implemented by using the OpenCV filter2D function (OPENCV, 2000). The kernel used is the weight average kernel:

$$M = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 10 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

To better simulate the aspect of wood, noise can be applied to the Otsu binarization result. Our system allows three types of Noise: Gaussian, Poisson, and Contoli. Gaussian and Poisson noise yields a noise with the appearance of points, while Contoli noise generates noise in the format of lines.

We consider in our work that the paper is white and the paint is black (both for the standard woodcut and the white-line one), and this noise corresponds to imperfections in the wood which appear in the woodcut result as points where the paint is not transferred to the paper. So we apply noise only to the black pixels, leaving the white ones without noise.

Gaussian and Poisson noise are generated using a similar method. First, we assign to each black pixel a sample from the corresponding distribution (Gaussian or Poisson). The Gaussian distribution is centered at its mean or expected value μ (in our simulations, $\mu = 0$) and has a standard deviation equals to σ_{Gauss}, while the Poisson distribution has standard deviation equals to $\sigma_{Poisson}$ and an expected number of occurrences equals to $\lambda_{Poisson}$. For the Gaussian case, which generates a real number, we convert the sample to an integer (Poisson distributions already yield an integer value). The sample obtained for each pixel will be applied as its color. To avoid any overflow problem, values above 255 and below 0 are treated as 255 and 0, respectively.

Contoli noise uses a method adapted from (CONTOLI, 2015). Briefly speaking, Contoli noise corresponds to add to random positions in the image an object with a high transparency level, such as a rectangle or a line. If this is done for a large amount of times, a noisy texture will be formed. In our method, we added $N_{Contoli}$ times an horizontal line with a width of 1 pixel and a random length $2l+1$ varying from $2l_{Contoli_{max}}+1$ to $2l_{Contoli_{min}}+1$, where l is the length of each "arm" of the line. This line has a color equals to $Color_{Contoli}$, which is added to the already existing color of the pixels where it is placed. The placement is done by randomly choosing $N_{Contoli}$ positions inside the image and putting the central pixel of the line on it, and growing the line pixel-by-pixel to the left and to the right until each side reaches l pixels (so the total length of the line is $2l+1$). As with the Gaussian and Poisson noise, we do not change the color of already white pixels in our woodcut, and any values above 255 are converted to 255.

4 RESULTS

In this chapter we present our synthesized woodcuts using the defined methodology, together with the validation of our work and discussion.

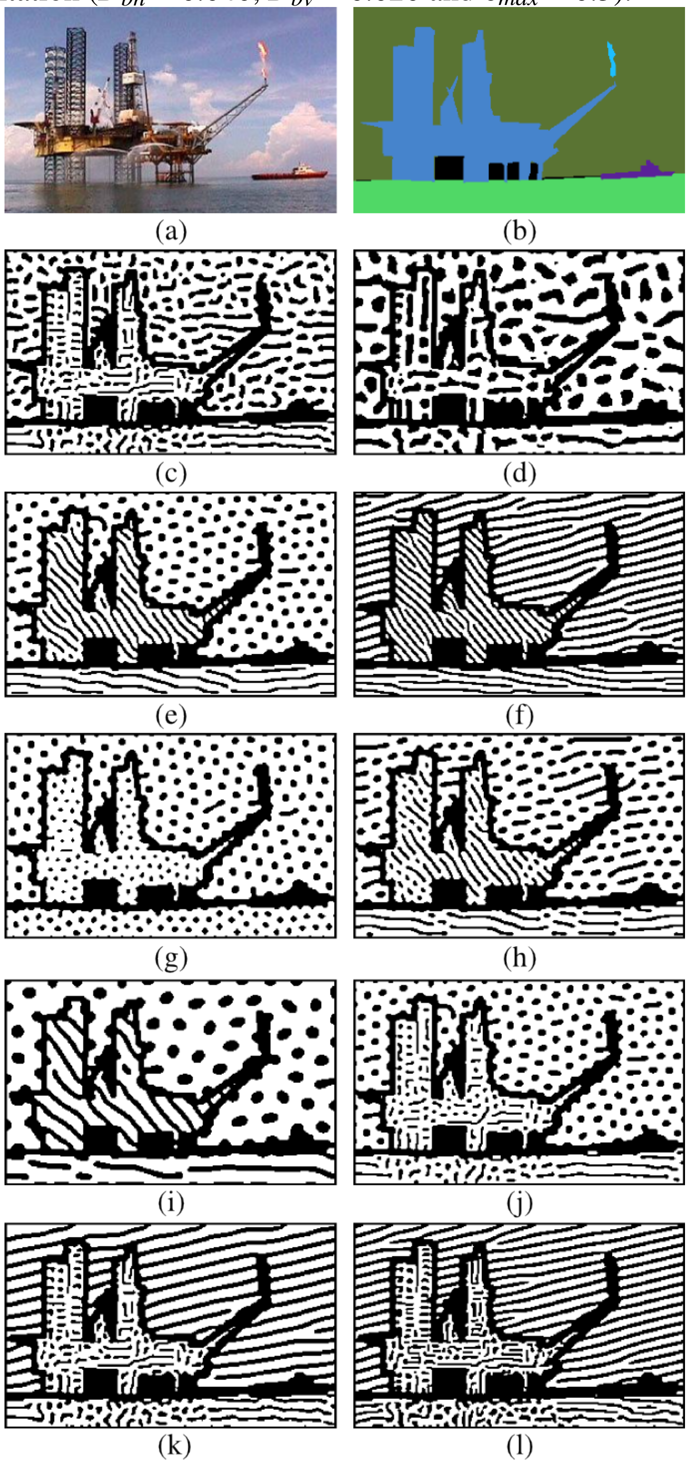
4.1 Experimental Results

In this section we show the results of our work. All images were made with the default values for the parameters, with θ calculated by region, using the static visualization option over the morphogen b and without the optional post-processing features (smoothing filter and noise), unless stated otherwise. Some of our user-defined parameters did not yield interesting results, so we keep them fixed for all our simulations, such as T_{low}, T_{high}, t, a_0, b_0, M_{white} and M_{black}.

For our simulations, we used an Ubuntu PC with 4 GB of RAM memory and an Intel Core 2 Duo processor (2.80 GHz x2). Our results are not realtime. For Fig. 4.1e, for instance, our system took a total of 223 seconds, being 19s for the preprocessing (mainly for the computation of S values), 199s for the RD simulation, and 5s for post-processing. It should be taken into account that this work is a proof of concept, so our focus was not on code optimization. Also, for many cases, the pattern was settled before $t = 16,000$ iterations. To better analyze the process, our prototype shows the image being generated in real time, which increases the computation burden.

Fig. 4.1 illustrates the flexibility of our system. For the same input image, many woodcut variations are possible by adjusting the parameters. Different regions can present distinct patterns, which eases the identification of image parts (Figs. 4.1e and 4.3). Figs. 4.1f and 4.1g show homogeneous patterns (stripes and spots, respectively) due to setting an unique value for δ in the entire image. By setting a smaller difference between δ_{min} and δ_{max} the pattern difference is more nuanced, although still present, as some regions show longer stripes than others (Fig. 4.1h). Reducing the values for S makes the stripes and spots bigger (Fig. 4.1i), while increasing the difference between S_{min} and S_{max} allows finer strokes for more detailed regions and a simpler appearance for background areas (Fig. 4.4). When allowing each pixel to have their own stroke orientation θ, it is possible to preserve details of the original image from inside the regions (Figs. 4.1c, 4.1d, 4.5, 4.7 and 4.8). Adaptive orientation can only be used on specific parts of the image, allowing detail preservation for the most detailed regions while other regions have a more consistent

Figure 4.1: A series of results over the same oil platform image showing the effect of parameter variation. (a) Original image from the ADE20k dataset (ZHOU et al., 2017). (b) Segmentation of the image from the ADE20k dataset. (c) Result for θ calculated by pixel (D_{bh} = 0.040, D_{bv} = 0.020). (d) Result for θ calculated by pixel (D_{bh} = 0.040, D_{bv} = 0.020, S_{min} = 0.002, S_{max} = 0.005). (e) Result using the default value for parameters. (f) Result for δ_{max} = 0.5. (g) Result for δ_{min} = 1.2 and δ_{max} = 1.2. (h) Result for δ_{min} = 0.8 and δ_{max} = 1.2. (i) Result for S_{min} = 0.002 and S_{max} = 0.005. (j) Result for adaptive orientation. (k) Result for adaptive orientation (D_{bh} = 0.040 and D_{bv} = 0.020). (l) Result for adaptive orientation (D_{bh} = 0.040, D_{bv} = 0.020 and δ_{max} = 0.5).



(a) (b)

(c) (d)

(e) (f)

(g) (h)

(i) (j)

(k) (l)

single pattern (Figs. 4.1j, 4.1k, 4.1l, 4.2b and 4.6). The effect of a plain color background can be seen in Figs. 4.7 and 4.8.

Fig. 4.2 shows the result for different border widths, calculated in the start of the preprocessing step. The orientation angle for regions where θ was calculated by region suffered some alterations among these results, due to the dilating process changing which is the brightest pixel from the region limit.

Figure 4.2: Comparison between different border widths, obtained by varying the value of n_{dilate} (for Fig. 3.2a with $k = \{1, 11, 21\}$, $T_{diff} = 10.0$, D_{bh} = 0.040, D_{bv} = 0.020 and adaptive orientation). (a) Result for $n_{dilate} = 0$. (b) Result for $n_{dilate} = 1$. (c) Result for $n_{dilate} = 2$.

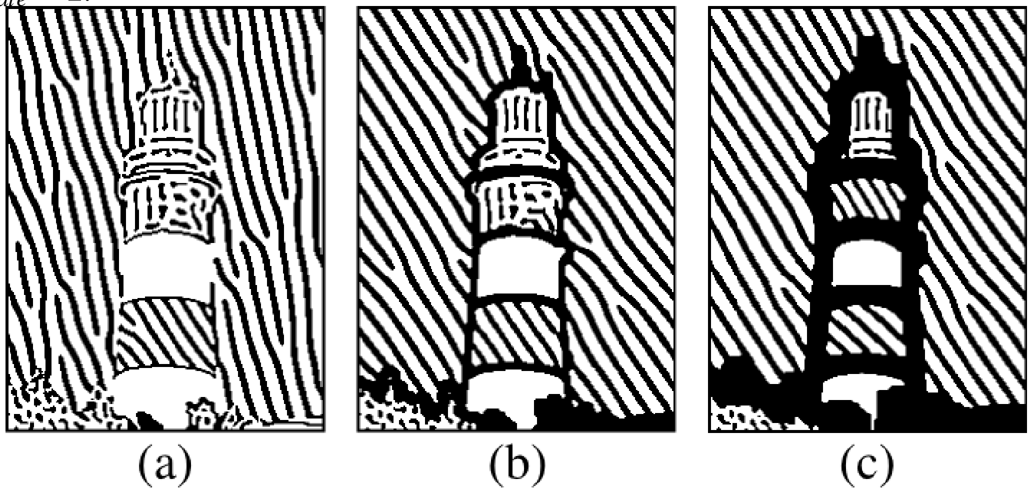
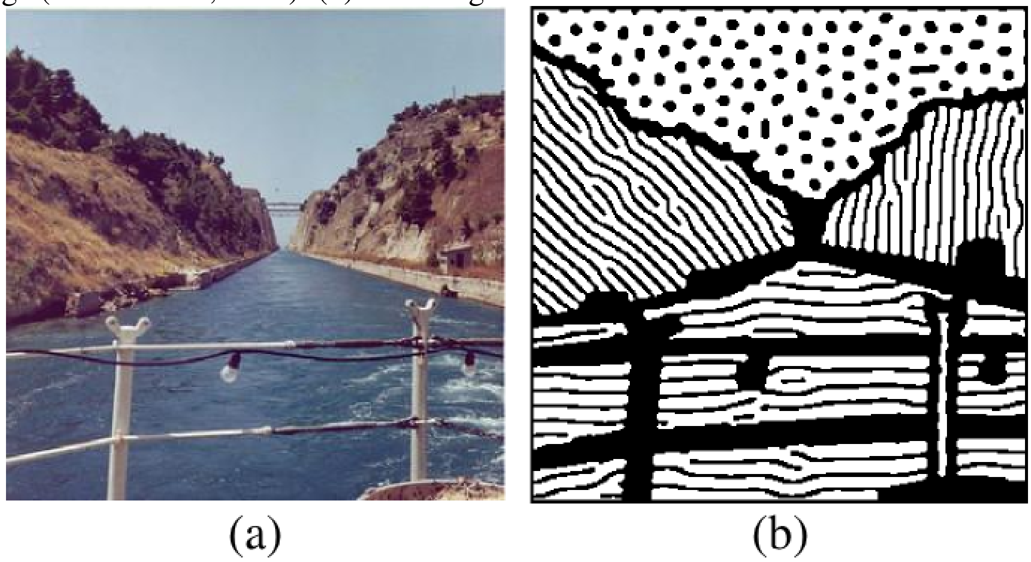


Figure 4.3: Results for a channel image from the ADE20K dataset, presenting regions with different styles and orientation for the strokes, as the sky with spots and the river and hills with stripes whose orientation reinforce their status as different regions. (a) Original image (ZHOU et al., 2017). (b) Resulting woodcut.



We tested the effect of variations in the coefficient diffusion of a, i.e., D_a, by setting non-default values for D_{ah} and D_{av} (Fig. 4.9). The images in the first row use a

Figure 4.4: Results for a desert road image from the ADE20K dataset, showing the effect of larger difference between S_{min} and S_{max} which allows finer details in the hills and a more simplified appearance for background regions as the sky and the road. (a) Original image (ZHOU et al., 2017). (b) Resulting woodcut (with $S_{min} = 0.002$ and $S_{max} = 0.018$).

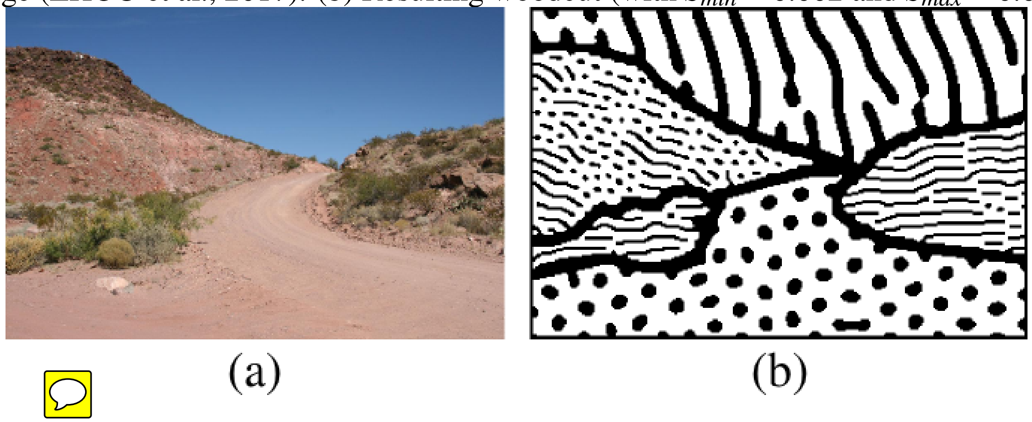


Figure 4.5: Results using Lenna, demonstrating how details and traces from the original image can be transmitted to the final woodcut even without segmentation. (a) Original image. (b) Resulting woodcut (without segmentation, and with S and θ calculated by pixel).

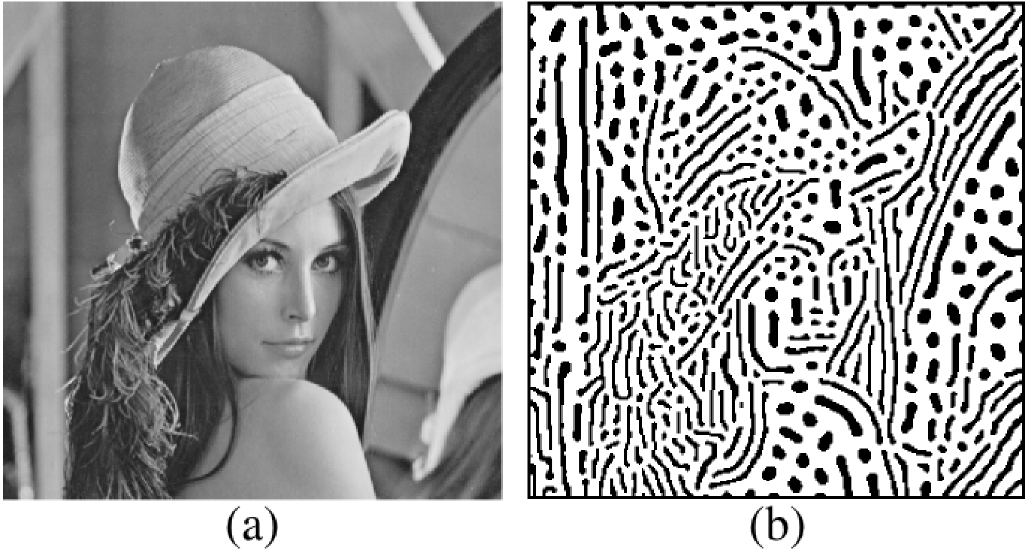


Figure 4.6: Results for a delicatessen image from the ADE20K dataset using adaptive orientation, which preserves the details of foreground regions as the table and reducing focus on background regions as the wall in the upper left corner. (a) Original image (ZHOU et al., 2017). (b) Resulting woodcut (with $T_{size} = 300$, adaptive orientation, $T_{diff} = 10.0$, $s_{theta} = 0.2$, $D_{bh} = 0.040$ and $D_{bv} = 0.020$).

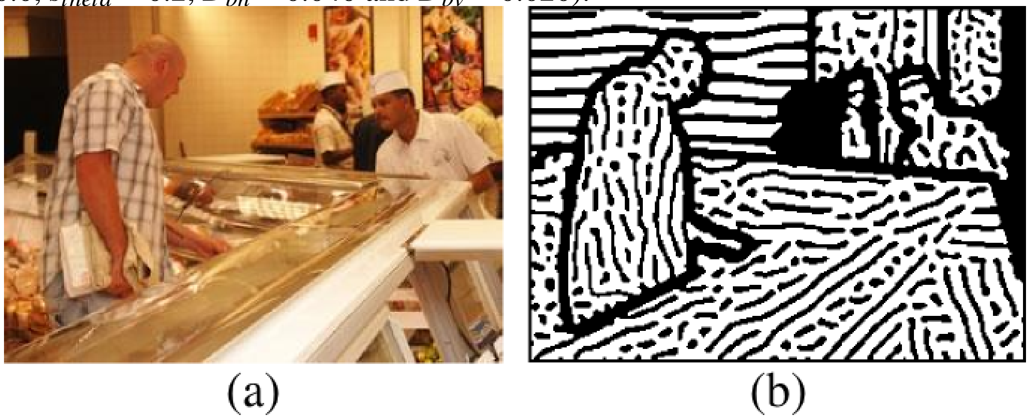
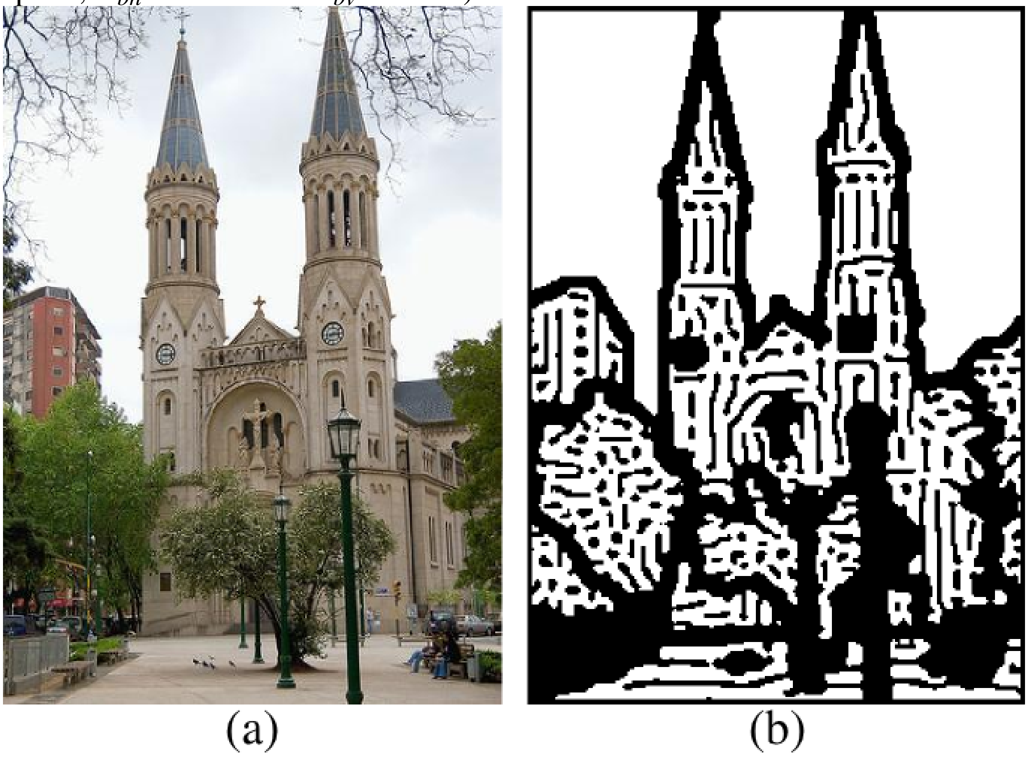
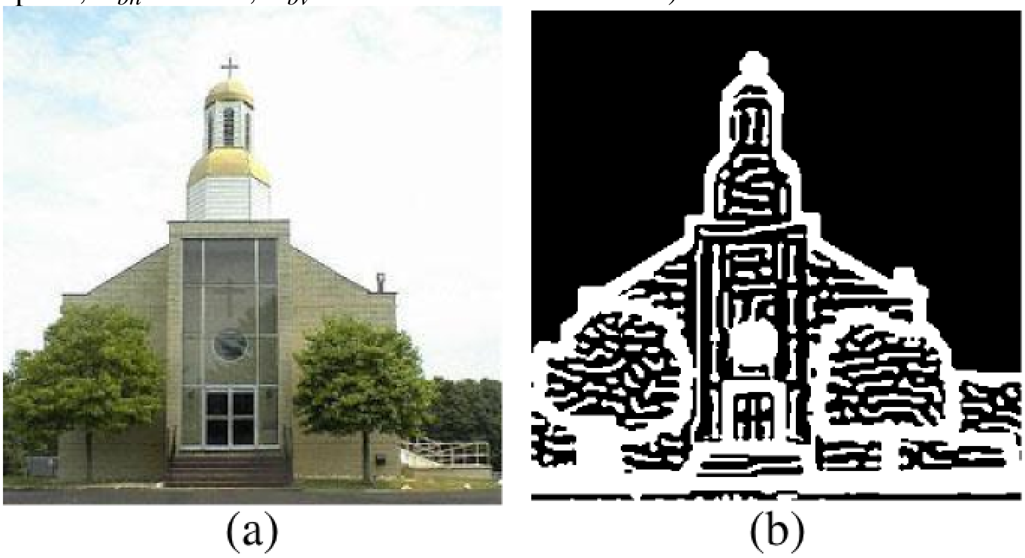


Figure 4.7: Results for a church image from the ADE20K dataset, showing preservation of details inside the regions due to the orientation by pixel mode and the effect of plain color background. (a) Original image (ZHOU et al., 2017). (b) Resulting woodcut (with θ by pixel, $D_{bh} = 0.040$ and $D_{bv} = 0.020$).



(a) (b)

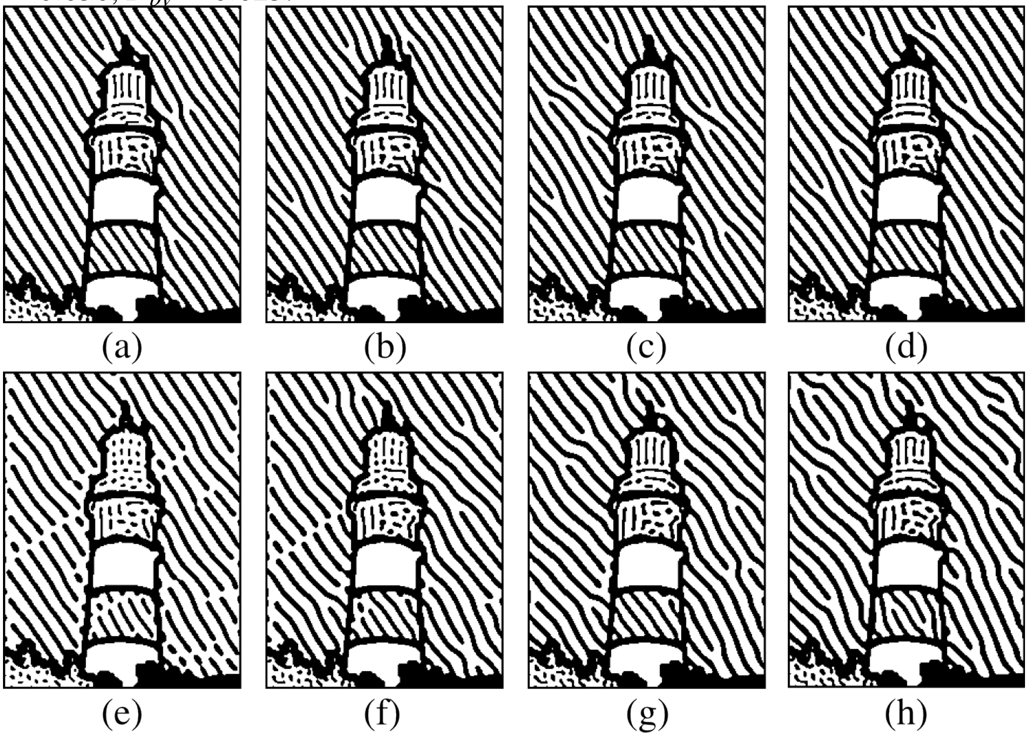
Figure 4.8: Results for a church image from the ADE20K dataset, showing preservation of details inside the regions due to the orientation by pixel mode and the effect of plain color background. (a) Original image (ZHOU et al., 2017). (b) Resulting woodcut (with θ by pixel, $D_{bh} = 0.040$, $D_{bv} = 0.020$ and inverted colors).



(a) (b)

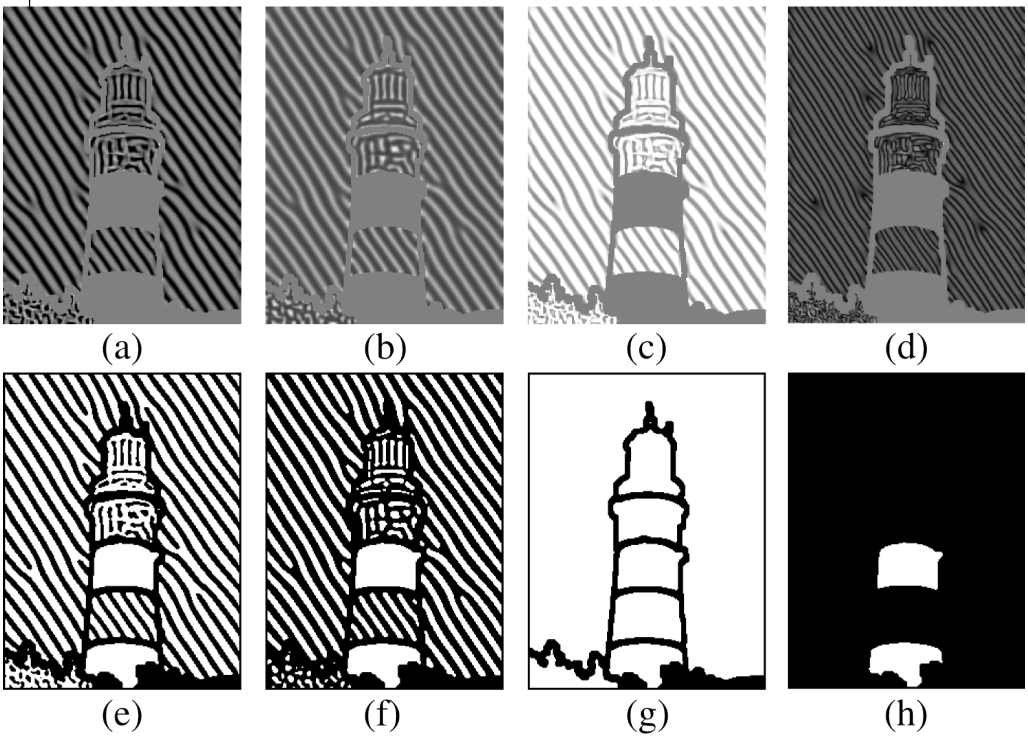
larger difference between D_{bh} and D_{bv}, producing longer, straight stripes. In this situation, enlarging D_a does not produce a huge effect, although bifurcations become more common. In the second row variations of D_a are performed for a smaller difference between D_{bh} and D_{bv}. In this case, the effect of D_a variations is more noticeable: broken stripes in the background and spots in the lighthouse top, shown when $D_a = 0.100$, turn into labyrinthine stripes when D_a increases. Setting $D_a = 0.075$ results in lack of pattern for the background.

Figure 4.9: Tests with variation in the coefficient diffusion of the morphogen a. All the tests used the same parameters as Fig. 4.14i unless stated as another value. (a) $D_{ah} = D_{av} = 0.100$. (b) $D_{ah} = D_{av} = 0.125$. (c) $D_{ah} = D_{av} = 0.150$. (d) $D_{ah} = D_{av} = 0.175$. (e) $D_{ah} = D_{av} = 0.100$, $D_{bh} = 0.030$, $D_{bv} = 0.025$. (f) $D_{ah} = D_{av} = 0.125$, $D_{bh} = 0.030$, $D_{bv} = 0.025$. (g) $D_{ah} = D_{av} = 0.150$, $D_{bh} = 0.030$, $D_{bv} = 0.025$. (h) $D_{ah} = D_{av} = 0.175$, $D_{bh} = 0.030$, $D_{bv} = 0.025$.



The different options to translate the concentrations of a and b into color are shown in Fig. 4.10. The first row shows the raw RD result, before the post-processing step. It is interesting to note that a and b seems almost complementary: a pixel with a high value for a has a low value for b and vice-versa. Fig. 4.10d (where color is obtained from $|b-a|$) shows an interesting pattern of empty line contours. However, after Otsu the results for morphogen combinations did not resulted in an interesting pattern, due to being either too bright (Figs. 4.10c and 4.10g) or too dark (Figs. 4.10d and 4.10h). On other hand, using a produced a slightly darker woodcut (Fig. 4.10f), with black and white stripes having approximately the same width.

Figure 4.10: Translating into color different combinations of a and b concentrations. All these simulations have the same parameters as Fig. 4.14i except for dynamic visualization. (a) Result before post-processing for b. (b) Result before post-processing for a. (c) Result before post-processing for $a+b$. (d) Result before post-processing for $|b-a|$. (e) Final result for b. (f) Final result for a. (g) Final result for $a+b$. (h) Final result for $|b-a|$.



In most of our tests, the difference between static and dynamic visualization was either almost none or only one of these option was able to generate a final pattern after Otsu's method. Fig. 4.11 shows one of the few results where both options generated images with valid patterns and a certain degree of difference between them.

Figure 4.11: Comparison of static and dynamic visualization. Both simulations have the same parameters as Fig. 4.1f. (a) Static visualization. (b) Dynamic visualization.

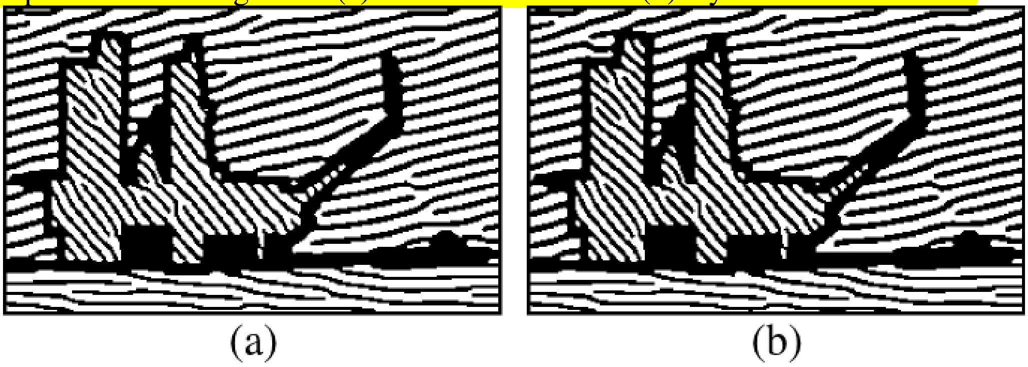
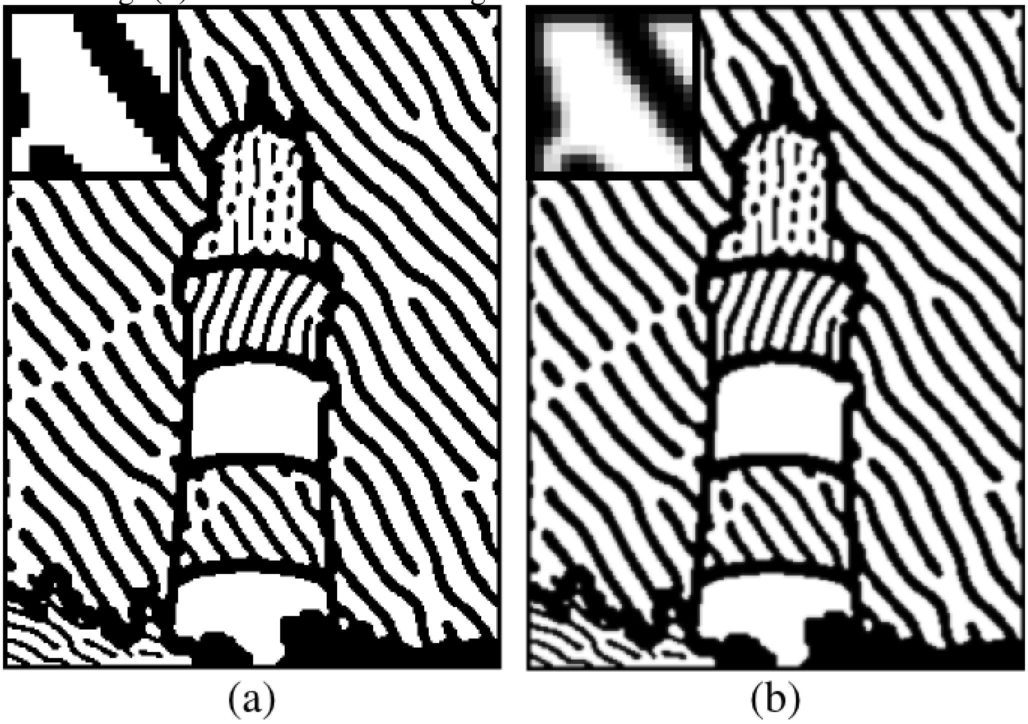


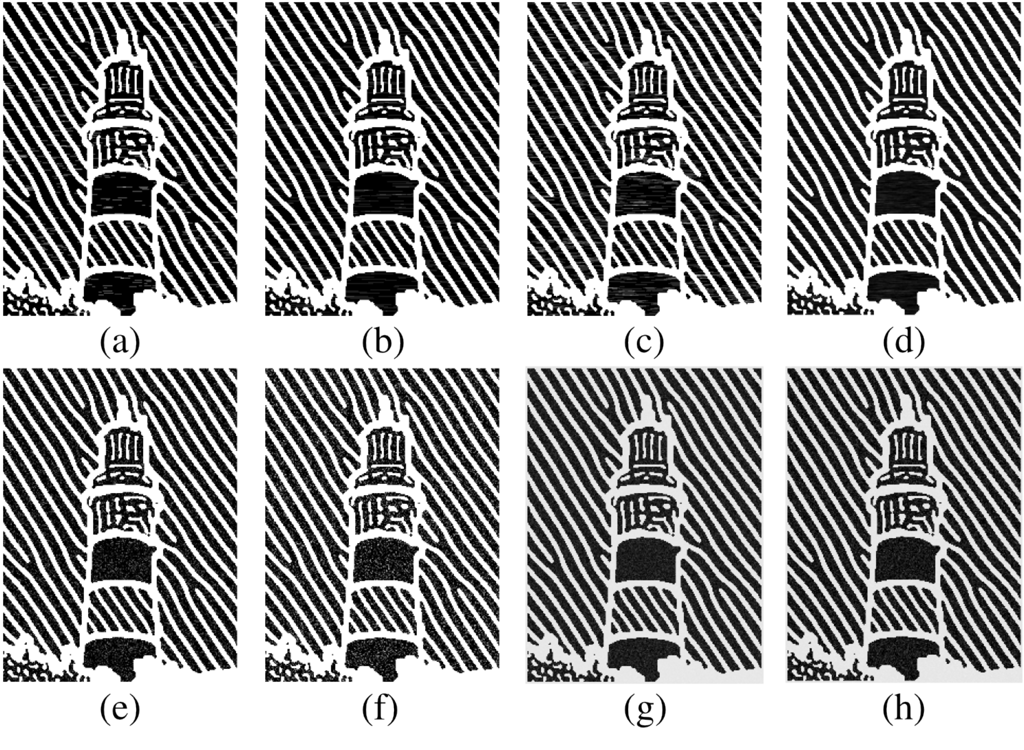
Fig. 4.12 compares the result with and without the optional smoothing filter. On one hand, smoothing filter reduces the aliasing, but the image seems a bit blurred.

Figure 4.12: Example of the post-processing smoothing optional step (with $k = \{1, 11, 21\}$ and $T_{diff} = 10.0$). In the top left, a close-up of the result. (a) Result without smoothing. (b) Result with smoothing.



The second optional feature from the post-processing step, addition of noise, can be seen in Fig. 4.13. Contoli noise was able to generate noise resembling short horizontal cuts in the wood (Fig. 4.13a to 4.13d). In fig. 4.13 the noise seems somewhat artificial, with few color variation and a clear difference between a noise element and the black background. Figs. 4.13b and 4.13c, with an increase in $N_{Contoli}$ (the amount of noise elements added) but a reduction on the color intensity of the noise elements, represented by $Color_{Contoli}$, show a more distributed noise. Raising $N_{Contoli}$ even more while reducing $Color_{Contoli}$ yields a picture where noise is just a small alteration in the black background color (Fig. 4.13d). Both Gaussian and Poisson noise resulted in similar point-like noise which resembles the traditional appearance of noise in TV transmissions, instead of the kind of irregularities we would expect to see in wood. Figs. 4.13e and 4.13f show Gaussian noise, while Figs. 4.13g and 4.13h have Poisson noise. We noted that our implementation of Poisson noise is not able to produces noise elements with a large intensity difference as the background color (i.e., white points in the black background), as Gaussian noise did. Anyway, both line-like Contoli noise and point-like Gaussian and Poisson noises produced some similarity with the irregularities in some real woodcuts (Fig. 4.17).

Figure 4.13: Addition of noise in the post-processing step. Besides the post-processing parameters and inverted colors in post-processing, all simulations have the same ones as Fig. 4.14i. (a) Contoli Noise with $N_{Contoli} = 1000$, $l_{Contoli_{min}} = 1$, $l_{Contoli_{max}} = 5$ and $Color_{Contoli} = 80$. (b) Contoli Noise with $N_{Contoli} = 1000$, $l_{Contoli_{min}} = 2$, $l_{Contoli_{max}} = 15$ and $Color_{Contoli} = 30$. (c) Contoli Noise with $N_{Contoli} = 3000$, $l_{Contoli_{min}} = 2$, $l_{Contoli_{max}} = 8$ and $Color_{Contoli} = 40$. (d) Contoli Noise with $N_{Contoli} = 30000$, $l_{Contoli_{min}} = 2$, $l_{Contoli_{max}} = 8$ and $Color_{Contoli} = 4$. (e) Gaussian Noise with $\sigma_{Gauss} = 50$. (f) Gaussian Noise with $\sigma_{Gauss} = 75$. (g) Poisson Noise with $\sigma_{Poisson} = 6$, $\lambda_{Poisson} = 5$. (h) Poisson Noise with $\sigma_{Poisson} = 12$, $\lambda_{Poisson} = 2$.



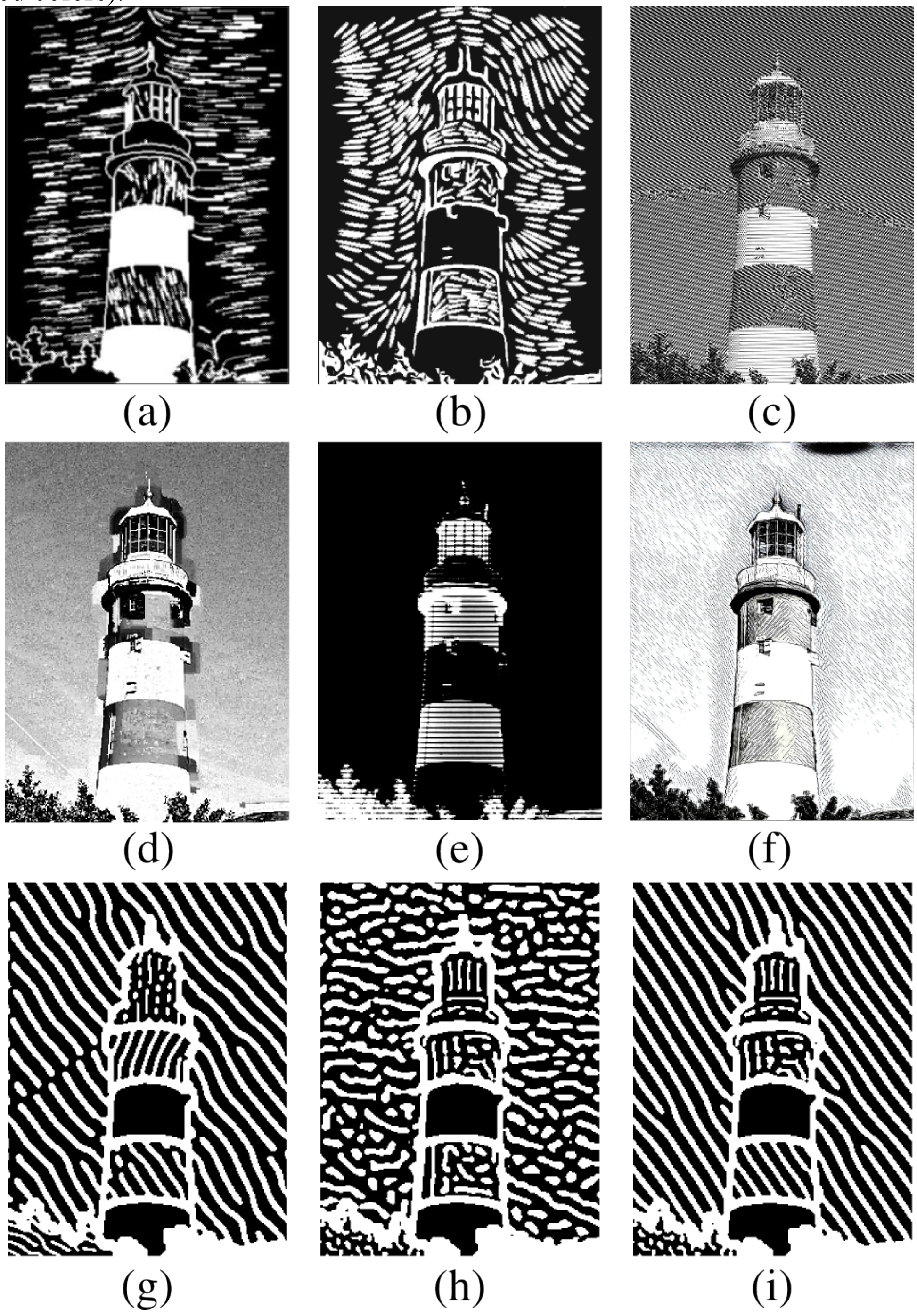
4.2 Validation

Validation is a difficult task in the general area of NPR (ISENBERG, 2013). A quantitative evaluation is hard since there are no specific attributes that can be measured and compared. We perform a qualitative comparison against the previous results addressing woodcuts known in the literature, using a lighthouse image used in (MELLO; JUNG; WALTER, 2007) and (LI; XU, 2015) (Fig. 3.2a). To test this image, we did a manual segmentation over it (Fig. 3.2b). Comparing with Mello's work (Fig. 4.14a), our results (Fig. 4.14g, 4.14h and 4.14i) have a different appearance, with directional stripes resembling actual strokes, expanding the range of styles which can be generated. On the other hand, Li's work (LI; XU, 2016) is also aesthetically appealing (Fig. 4.14b), representing a different woodcut style than ours (since it intends to simulate the Yunnan Out-of-Print woodcut, a tradition Chinese style). We also tested some image processing tools such as an Adobe Photoshop plugin (Fig. 4.14d), GIMP (Fig. 4.14e) and Prisma (Fig. 4.14f), which did not yield images resembling actual wood carvings. Mello et al. (MELLO; JUNG; WALTER, 2007) also compared another Adobe Photoshop plugin, which was not able to produce a woodcut-like result.

To assess the ability of our system to replicate actual woodcuts, we asked an artist, Guilherme León Berno de Jesus, to make a real woodcut based on the lighthouse image (Fig. 4.15a), in order to try to generate a similar image (JESUS, 2018). We did not specified the details of our system neither showed any previous results to avoid bias, but we informed him about the general kinds of patterns RD generates, i.e., stripes and spots, so the woodcut could be in a style that our system is able to replicate. We manually set the black and white regions and the orientation angle through the image loading option for these parameters (Fig. 4.15b). The final result for this simulation is shown in Fig. 4.15c.

Finally, we performed a qualitative validation through an online form, asking users to assess images presented in random order. Before presenting the images, we described the woodcut process, as illustrated in Fig. 1.1. For each image we asked the participants if they consider plausible that the image was produced by this process. They answered by rating the images with a number from "1" (totally disagree) to "5" (totally agree). We show a series of 23 images, containing our results (7 images), results from other woodcut synthesizing methods (3 images), other NPR results using RD (2 images), other general black-and-white NPR results (2 images), real woodcuts in general styles (3 images), real woodcuts from Brazilian Northeast (3 images), and other real black-and-white artworks

Figure 4.14: Comparison between different methods for generating woodcuts (for Fig. 3.2a). (a) Mello's result (MELLO; JUNG; WALTER, 2007). (b) Li's result (LI; XU, 2016). (c) Xylograph plugin (AmphiSoft, 2007) (source: (MELLO; JUNG; WALTER, 2007)). (d) Simplify 3 Preset List (Photoshop plugin) result for the Wood Carving option (Topaz Labs, 2018). (e) Result of a GIMP pipeline to generate a woodcut-like image (WELCH, 2018). (f) Prisma app, Light Summer Reading filter with sharpen +100, contrast +100 and gamma -100 (Prisma Labs, 2016). (g) Our result (with $k = \{1, 11, 21\}$, $T_{diff} = 10.0$ and inverted colors). (h) Our result for θ calculated by pixel (with $k = \{1, 11, 21\}$, $T_{diff} = 10.0$, $D_{bh} = 0.040$, $D_{bv} = 0.020$, and inverted colors). (i) Our result for adaptive orientation (with $k = \{1, 11, 21\}$, $T_{diff} = 10.0$, $D_{bh} = 0.040$, $D_{bv} = 0.020$, and inverted colors).

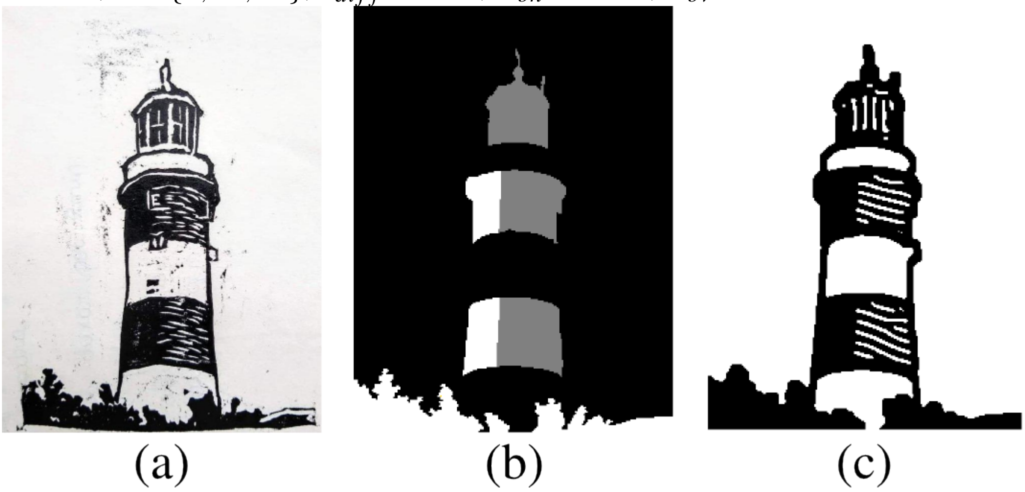


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 4.15: Comparison of our work with an actual woodcut work over the lighthouse image. (a) Real woodcut artwork by Guilherme León Berno de Jesus (JESUS, 2018). (b) Our segmentation of black and white parts. (c) Our results using manual setting of black and white parts, manual orientation, Border Color = 255 (i.e., white border color), $T_size = 600$, $k = \{1, 11, 21\}$, $T_{diff} = 10.0$, $D_{bh} = 0.040$, $D_{bv} = 0.020$ and inverted colors.



(3 images). After assessing the images, we allowed the participant to write reasons why they gave low ratings for the images they did not considered plausible to be generated by the woodcut process.

A total of 72 people answered the questionnaire, being 58 males and 14 females. From these people, only 2 had worked before with woodcuts, while 51 have not worked but already had heard about it, and 19 had never heard about woodcuts. About NPR, we had 3 people who had worked with it, 25 who had heard about the subject, and 44 who did not know about NPR.

On average, images were rated as 3.67 in our scale. Our results were rated a bit higher, at 3.86, while other computer generated woodcuts received a grade of 3.55. However, real woodcuts were rated as 3.64, slightly below the total average and below the grade for other real artworks (3.74), which could reflect the lack of familiarity with woodcuts from the participants. The two participants who have previous experience with woodcuts rated our images with a lower grade, 2.93, but they did not gave a good rating for actual woodcuts (only 3.00) or other woodcut generating methods (2.17). Due to the small number of participants in this category, it is hard to reach a conclusion about it. Among the participants who had heard about woodcuts but not worked with it, average grade for all images is 3.63, and our results a grade of 3.82, while the results of other woodcut methods were rated as 3.53. These participants gave actual woodcuts a rate of only 3.67. The grades in this category are not so different than the total results. Lastly, the three participants who have experience in NPR gave to our results a grade of 2.90

but a grade of 3.78 to other woodcut synthesis methods. They also were able to detect properly the actual woodcuts, giving them a grade of 4.17. The three participants with NPR experience gave as reasons for low ratings the presence of details difficult to make by hand, too much photorealistic images which appear to be digitally treated, and the presence of aliasing which denotes an artificial image.

The list of images and the complete result tables are in appendixes A and B.

4.3 Discussion

In this section, we analyze some aspects from our results as its similarity with certain styles of woodcuts, and the limitations of our technique.

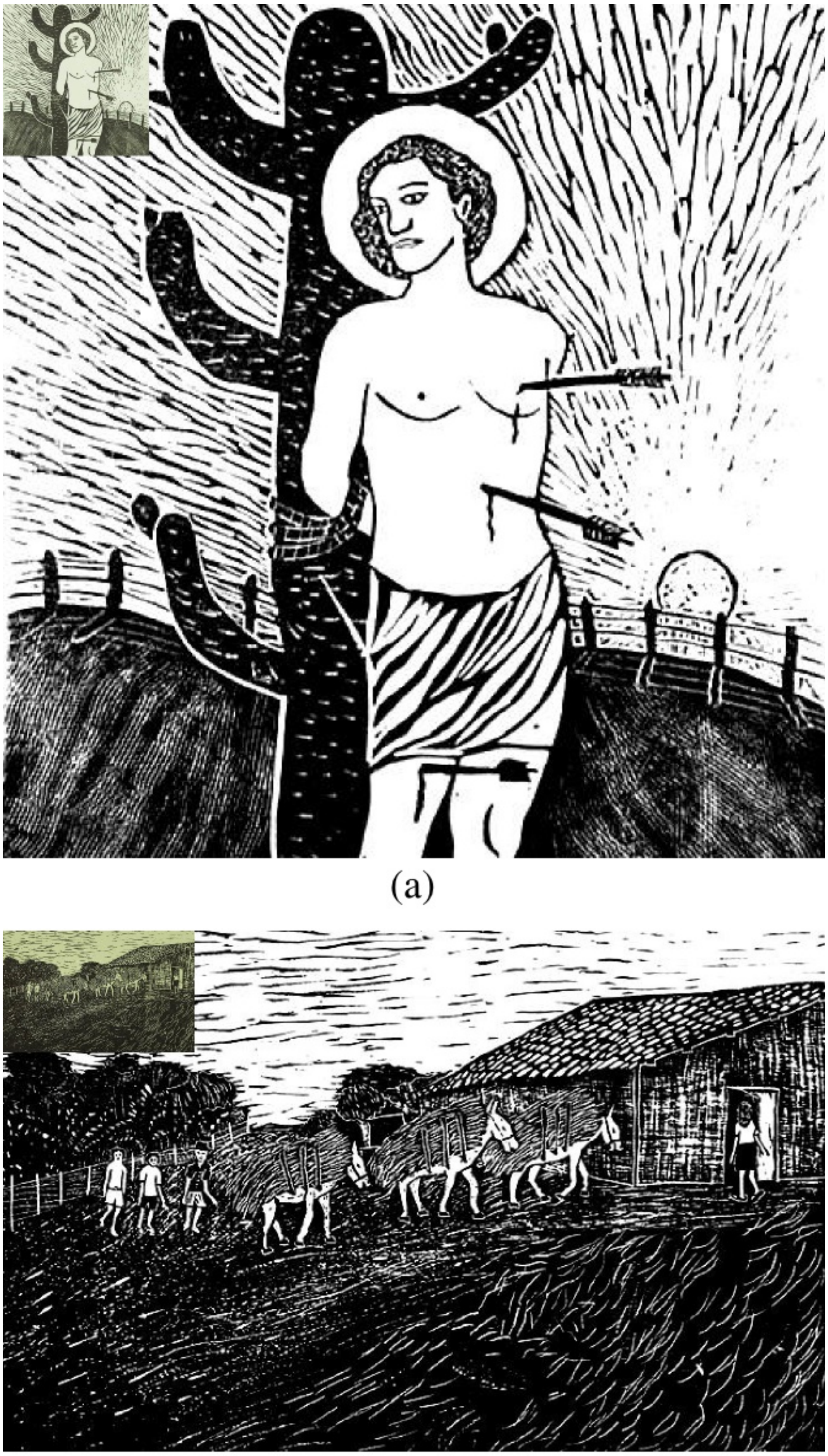
During the development of this work, we noticed that many of our woodcuts have an appearance similar to typical Brazilian Northeast woodcuts (as in Fig. 4.8, for example), which are characterized by an abstract form, a black-and-white coloration (including areas with flat black and white color), and the usage of lines and points to express different textures (Fig. 4.16, for example). As we mentioned before, some styles of irregularities encountered in these woodcuts are similar to the noise we implemented (Fig. 4.17).

In Fig. 4.15 we produced an image similar to an actual woodcut. On one hand, our method is able to simulate the overall aspect of the real artwork, but on the other hand some fine details such as the specific contour or the lighthouse top.

A limitation of our work is that, due to the nature of RD processes as the distribution of chemical substances forming waves of larger or smaller concentration, white and black stripes have more or less the same relative width proportion, so it is not possible to vary the spacing of strokes. However, by setting a as the morphogen to translate to color we obtained an image where black and white stripes have similar width, so theoretically it is possible to overcome this limitation. Another limitation is the inability to have lines crossing each others (as in the walls of the house in Fig. 4.16b). In actual woodcuts, strokes are usually more regular than our wave-like stripes, and tend to be longer, without breaking in the middle as sometimes the reaction-diffusion stripes do.

Our validation study shows that most viewers considered plausible that our results could be generated by the woodcut making process, but viewers with more experience with woodcut or NPR on overall disagreed with this opinion. Given the small sample of more experienced users and the fact they rated actual woodcuts with low grades, this result should be taken with caution, but they indicated the need of more improvement in order to increase similarity with the real artworks.

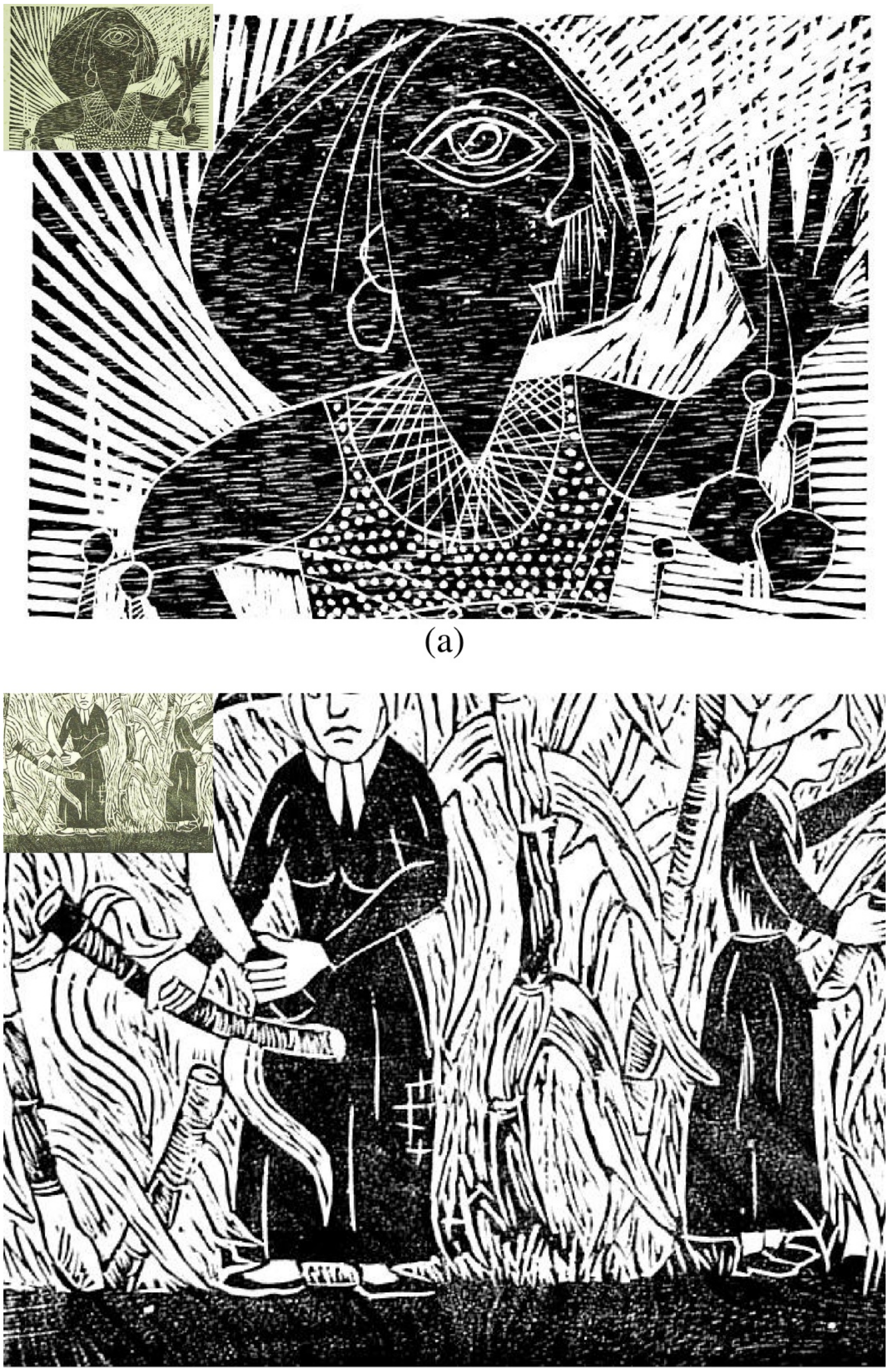
Figure 4.16: Actual woodcuts from the Brazilian Northeast Region, digitally treated to pure black and white. Original woodcuts with green-stained paper are shown as a miniature in the upper left corner of the image. (a) (FRANCORLI, 2018), (b) (GONZAGA, 2018).



(a)

(b)

Figure 4.17: Actual woodcuts from the Brazilian Northeast Region showing noise-like irregularities in the wood, digitally treated to pure black and white. Original woodcuts with green-stained paper are shown as a miniature in the upper left corner of the image. (a) (FERNANDES, 2018) showing line-like irregularities similar to our Contoli noise, (b) detail of (SILVA, 2018) showing point-like irregularities like our Gaussian or Poisson noise.



(a)

(b)

5 CONCLUSIONS

We presented a method to synthesize a woodcut image by using Reaction-Diffusion (RD). Our method first applies a preprocessing step on the input image where we compute information regarding regions with pure black and white coloration, the detail level and correspondent size of strokes, the tendency to form stripes or spots, and the orientation of strokes, generating a parameter map. Then, we run a RD system over this parameter map, generating the raw image with the pattern of strokes. Last, a post-processing step is done in order to remove noise and binarize the image, resulting into a black and white woodcut.

Our main contribution is the creation of another tool to synthesize woodcuts, an artwork style with few previous attempts to generate, by the unprecedented usage of RD for this task, a mechanism which was previously used in other NPR works. Our method yields woodcuts with a different style than the existing tools. Our results hold similarity to some styles of actual woodcuts. Qualitative validation shows that most participants agreed that our results could plausibly be generated by the woodcut process. However, since the few participants with experience in woodcutting gave a lower rating to both our images and actual woodcuts than the ones without experience, and we had only 2 of these participants, validation should be taken with caution.

Finally, we can conclude that our method is able to generate images which resemble woodcuts through the usage of an adapted RD system. This shows that RD has a good potential for artistic ends, adding the possibility of generating woodcuts with a different aspect than other tools with the same goal, increasing the range of styles NPR mechanisms can simulate.

5.1 Future Work

To improve our woodcut generating system, we intend to allow more than one type of noise in the same image, such as using Gaussian point-like noise together with line-like Contoli noise. Computational efficiency can be improved by setting a stabilization condition for the RD system, so the system will not need to run the entire t steps after being stabilized into a pattern.

We also intend to improve the user interface to make its usage easier for artists and users without programming abilities. For that, all the steps - preprocessing, processing and post-processing - would be united in the same program, and an option to preset the value of parameters to simulate specific styles.

Finally, to enlarge the range of woodcuts our system can generate, we intend to explore more values for parameters, as for instance to run more simulations using *a* instead of *b* as the morphogen whose concentration will be translated into color, since our preliminary tests showed that this option allows black and white stripes with the same width in the final result.

REFERENCES

AmphiSoft. **Xylograph Plugin for Photoshop**. 2007. URL: <http://photoshop.msk.ru/as/ngrave.html>. [Online; accessed February 7th 2019].

ANTTWEAKBAR. 2005. <http://anttweakbar.sourceforge.net/doc/>. [Online; accessed January 28th 2019].

BARD, J. A model for generating aspects of zebra and other mammalian coat patterns. **Journal of Theoretical Biology**, v. 93, n. 2, p. 363–385, 1981.

BARD, J.; LAUDER, I. How well does turing's theory of morphogenesis work? **Journal of Theoretical Biology**, v. 45, n. 2, p. 501–531, 1974.

BARROS, R. S.; WALTER, M. Synthesis of human skin pigmentation disorders. In: WILEY ONLINE LIBRARY. **Computer Graphics Forum**. [S.l.], 2017. v. 36, n. 1, p. 330–344.

BHUMIYA. **A stippled drawing of Jorge Luis Borges**. 2006. Available at Wikimedia Commons: <https://commons.wikimedia.org/wiki/File:Stippled_Borges.PNG>. [Online; accessed February 17th 2019].

BRAZELTON, K. **Musica Orbis instruments on stage with MO banner behind**. 1975. Available at Wikimedia Commons: <https://commons.wikimedia.org/wiki/File:Woodcut_of_Musica_Orbis_instruments.gif>. [Online; accessed February 17th 2019].

C. 2018. <http://www.open-std.org/jtc1/sc22/wg14/>. [Online; accessed January 28th 2019].

CHI, M.-T.; LIU, W.-C.; HSU, S.-H. Image stylization using anisotropic reaction diffusion. **The Visual Computer**, Springer, v. 32, n. 12, p. 1549–1561, 2016.

CONTOLI, A. **Procedural seamless noise texture generator**. 2015. URL: <https://www.codeproject.com/Articles/838511/Procedural-seamless-noise-texture-generator>. [Online; accessed January 31th 2018].

DEAK, A. **Aranha, Louise Joséphine Bourgeois**. 2011. Available at Wikimedia Commons: <https://commons.wikimedia.org/wiki/File:Aranha,_Louise_Jos%C3%A9phine_Bourgeois_(5877468973).jpg>. [Online; accessed February 17th 2019].

DIVERDI, S. et al. Painting with polygons: A procedural watercolor engine. **IEEE Transactions on Visualization and Computer Graphics**, v. 19, n. 5, p. 723–735, May 2013. ISSN 1077-2626.

FERNANDES, C. **A Rendeira**. 2018. Available at the Centro Nacional de Folclore e Cultura Popular (CNFCP) site: <http://www.cnfcp.gov.br/interna.php?ID_Secao=64>. [Online; accessed February 14th 2019].

FORESMAN, S. **Line art drawing of a girl with bangs**. 2008. Available at Wikimedia Commons: <https://commons.wikimedia.org/wiki/File:Bangs_(PSF).png>. [Online; accessed February 17th 2019].

FRANCORLI. **São Sebastião**. 2018. Available at the Centro Nacional de Folclore e Cultura Popular (CNFCP) site: <http://www.cnfcp.gov.br/interna.php?ID_Secao=64>. [Online; accessed December 10th 2018].

GONZAGA, J. L. **Meninos Cambiteiros**. 2018. Available at the Centro Nacional de Folclore e Cultura Popular (CNFCP) site: <http://www.cnfcp.gov.br/interna.php?ID_Secao=64>. [Online; accessed December 10th 2018].

HAEBERLI, P. Paint by numbers: Abstract image representations. In: ACM. **ACM SIGGRAPH computer graphics**. [S.l.], 1990. v. 24, n. 4, p. 207–214.

HEGDE, S.; GATZIDIS, C.; TIAN, F. Painterly rendering techniques: a state-of-the-art review of current approaches. **Journal of Visualization and Computer Animation**, v. 24, p. 43–64, 2013.

HERTZMANN, A. Painterly rendering with curved brush strokes of multiple sizes. In: ACM. **Proceedings of the 25th annual conference on Computer graphics and interactive techniques**. [S.l.], 1998. p. 453–460.

ISENBERG, T. Evaluating and validating non-photorealistic and illustrative rendering. In: **Image and video-based artistic stylisation**. [S.l.]: Springer, 2013. p. 311–331.

JESUS, G. L. B. de. **Lighthouse**. 2018. Artwork generated to this project.

JHO, C.-W.; LEE, W.-H. Real-time tonal depiction method by reaction–diffusion mask. **Journal of Real-Time Image Processing**, Springer, v. 13, n. 3, p. 591–598, 2017.

JR., J. T. K.; RAJA, S.; BADLER, N. I. Fruit senescence and decay simulation. **Computer Graphics Forum**, v. 30, n. 2, p. 257–266, abr. 2011.

KIPLING, R. **Original woodcut illustration for The Just So story "The crab that played with the sea" by Rudyard Kipling**. 2006. Available at Wikimedia Commons: <https://en.wikipedia.org/wiki/File:Justso_crabplay.jpg>. [Online; accessed February 17th 2019].

KROGH, C. **Snorre Sturluson - Illustration for Heimskringla 1899-edition**. 1890. Available at Wikimedia Commons: <https://en.wikipedia.org/wiki/File:Snorre_Sturluson-Christian_Krohg.jpg>. [Online; accessed February 17th 2019].

KYPRIANIDIS, J. E. et al. State of the "art": A taxonomy of artistic stylization techniques for images and video. **IEEE transactions on visualization and computer graphics**, IEEE, v. 19, n. 5, p. 866–885, 2013.

LAI, Y.-K.; ROSIN, P. L. Non-photorealistic rendering with reduced colour palettes. In: **Image and Video-Based Artistic Stylisation**. [S.l.]: Springer, 2013. p. 211–236.

LI, J.; XU, D. A scores based rendering for yunnan out-of-print woodcut. In: IEEE. **Computer-Aided Design and Computer Graphics (CAD/Graphics), 2015 14th International Conference on**. [S.l.], 2015. p. 214–215.

LI, J.; XU, D. Image stylization for yunnan out-of-print woodcut through virtual carving and printing. In: SPRINGER. **International Conference on Technologies for E-Learning and Digital Entertainment**. [S.l.], 2016. p. 212–223.

LITWINOWICZ, P. Processing Images and Video for an Impressionist Effect. In: WHITTED, T. (Ed.). **SIGGRAPH97**. New York: ACM Press, 1997. p. 407–414.

LU, C.; XU, L.; JIA, J. Combining sketch and tone for pencil drawing production. In: **Proceedings of the Symposium on Non-Photorealistic Animation and Rendering**. Goslar Germany, Germany: Eurographics Association, 2012. (NPAR '12), p. 65–73. ISBN 978-3-905673-90-6. Available from Internet: <http://dl.acm.org/citation.cfm?id=2330147.2330161>.

MELLO, V.; JUNG, C. R.; WALTER, M. Virtual woodcuts from images. In: ACM. **Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia**. [S.l.], 2007. p. 103–109.

MESQUITA, D.; WALTER, M. A model for generating aspects of zebra and other mammalian coat patterns. **Revista de InformÃątica TeÃşrica e Aplicada**, v. 24, n. 2, p. 28–46, 2017.

MIZUNO, S. et al. Automatic Generation of Virtual Woodblocks and Multicolor Woodblock Printing. **Computer Graphics Forum**, Blackwell Publishers Ltd and the Eurographics Association, 2000. ISSN 1467-8659.

MIZUNO, S. et al. Creating a virtual wooden sculpture and a woodblock print with a pressure sensitive pen and a tablet. **Forma**, v. 21, n. 1, p. 49–65, 2006.

MIZUNO, S. et al. Improvement of the virtual printing scheme for synthesizing ukiyo-e. In: . [S.l.: s.n.], 2002. v. 3, p. 1043–.

MIZUNO, S.; OKADA, M.; TORIWAKI, J. ichiro. Virtual sculpting and virtual woodcut printing. **The Visual Computer**, v. 14, p. 39–51, 1998.

MIZUNO, S. et al. Japanese traditional printing âĂIJ ukiyo-e âĂİ in a virtual space. In: . [S.l.: s.n.], 2002.

MOULD, D. A stained glass image filter. In: **Proceedings of the 14th Eurographics Workshop on Rendering**. Eurographics Association, 2003. (EGRW '03), p. 20–25. ISBN 3-905673-03-7. Available from Internet: <http://dl.acm.org/citation.cfm?id=882404.882407>.

NUMPY. 2006. <http://www.numpy.org/>. [Online; accessed January 13th 2019].

OPENCV. 2000. <https://opencv.org/>. [Online; accessed January 7th 2019].

OPENGL. 1992. <http://www.opengl.org/>. [Online; accessed January 28th 2019].

OTSU, N. A threshold selection method from gray-level histograms. **IEEE transactions on systems, man, and cybernetics**, IEEE, v. 9, n. 1, p. 62–66, 1979.

POSKANZER, J. **PPM Format Specification**. 1988. <http://netpbm.sourceforge.net/doc/ppm.html>. [Online; accessed January 23th 2019].

Prisma Labs. **Prisma Photo Editor**. 2016. URL: <https://prisma-ai.com/>. [Online; accessed February 7th 2019].

PYTHON. 1991. <https://www.python.org/>. [Online; accessed January 7th 2019].

SANDERSON, A. R. et al. Advanced reaction-diffusion models for texture synthesis. **Journal of Graphics, GPU, and Game Tools**, v. 11, n. 3, p. 47–71, 2006.

SCIKIT-IMAGE. 2009. <https://scikit-image.org/>. [Online; accessed January 7th 2019].

SILVA, E. F. da. **Os bóias frias**. 2018. Available at the Centro Nacional de Folclore e Cultura Popular (CNFCP) site: <http://www.cnfcp.gov.br/interna.php?ID_Secao=64>. [Online; accessed February 14th 2019].

STROTHOTTE, T.; SCHLECHTWEG, S. **Non-photorealistic computer graphics: modeling, rendering, and animation**. [S.l.]: Morgan Kaufmann, 2002.

STUYCK, T. et al. Real-time oil painting on mobile hardware. In: WILEY ONLINE LIBRARY. **Computer Graphics Forum**. [S.l.], 2017. v. 36, n. 8, p. 69–79.

Topaz Labs. **Simplify Plugin for Photoshop**. 2018. URL: <https://topazlabs.com/simplify/>. [Online; accessed September 23th 2018].

TURING, A. M. The chemical basis of morphogenesis. **Phil. Trans. R. Soc. Lond. B**, The Royal Society, v. 237, n. 641, p. 37–72, 1952.

TURK, G. Generating textures for arbitrary surfaces using reaction-diffusion. In: **Computer Graphics (Proceedings of SIGGRAPH 91)**. [S.l.: s.n.], 1991. p. 289–298.

VANDERHAEGHE, D.; COLLOMOSSE, J. Stroke based painterly rendering. In: ____. **Image and Video-Based Artistic Stylisation**. London: Springer London, 2013. p. 3–21. ISBN 978-1-4471-4519-6. Available from Internet: <https://doi.org/10.1007/978-1-4471-4519-6_1>.

WANG, M. et al. Towards photo watercolorization with artistic verisimilitude. **IEEE Transactions on Visualization and Computer Graphics**, v. 20, n. 10, p. 1451–1460, Oct 2014. ISSN 1077-2626.

WELCH, M. **Gimp Tips - Woodcut**. 2018. URL: <http://www.squaregear.net/gimptips/wood.shtml>. [Online; accessed September 23th 2018].

WINNEMÖLLER, H. Xdog: advanced image stylization with extended difference-of-gaussians. In: ACM. **Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Non-Photorealistic Animation and Rendering**. [S.l.], 2011. p. 147–156.

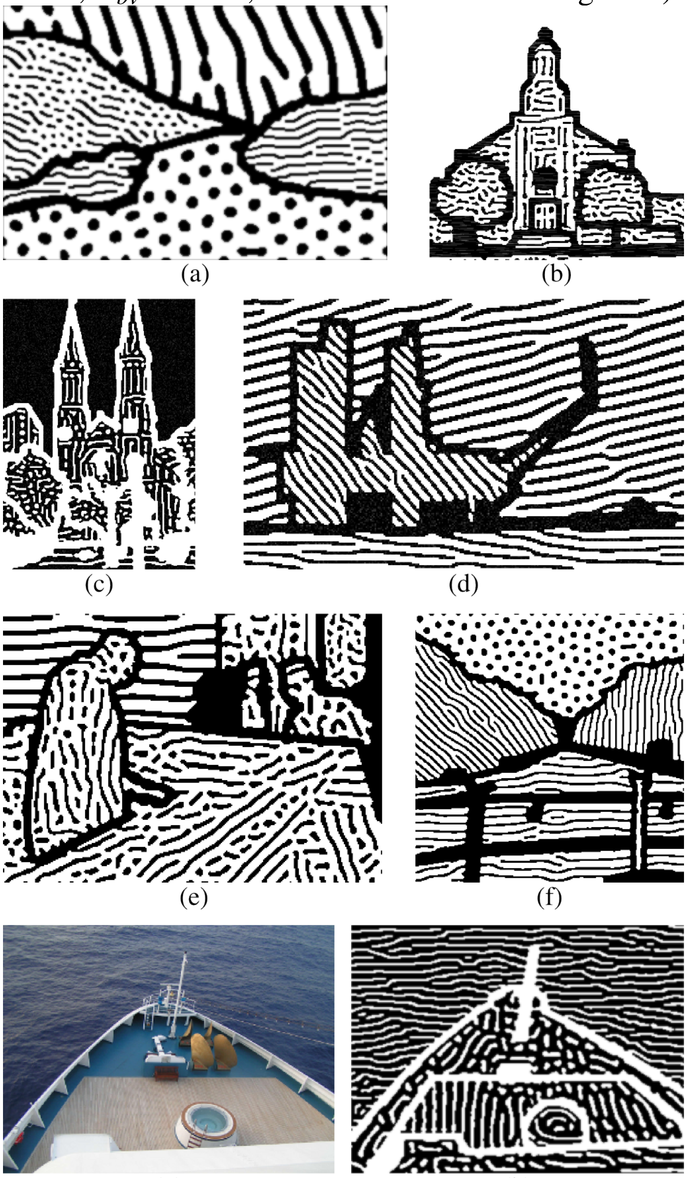
WITKIN, A.; KASS, M. Reaction-diffusion textures. In: **Computer Graphics (Proceedings of SIGGRAPH 91)**. [S.l.: s.n.], 1991. p. 299–308.

XU, J.; KAPLAN, C. S. Artistic thresholding. In: ACM. **Proceedings of the 6th international symposium on Non-photorealistic animation and rendering**. [S.l.], 2008. p. 39–47.

ZHOU, B. et al. Scene parsing through ade20k dataset. In: IEEE. **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2017. v. 1, n. 2, p. 4.

APPENDIX A — IMAGES USED IN VALIDATION

Figure A.1: Images used in validation which are generated by our methodology. Unless stated otherwise, all these images are from the Ade20k dataset (ZHOU et al., 2017) and have default values for the parameters, θ calculated by region, static visualization over the morphogen b and no optional post-processing features (smoothing filter and noise). (a) Desert road image (with S_{min} = 0.002, S_{max} = 0.018 and smoothing filter). (b) Church image (with θ by pixel, D_{bh} = 0.040, D_{bv} = 0.020 and Contoli Noise with $N_{Contoli}$ = 2,500 and $Color_{Contoli}$ = 30). (c) Church image (with θ by pixel, D_{bh} = 0.040, D_{bv} = 0.020, inverted colors and Gaussian Noise with $\sigma_{Contoli}$ = 20). (d) Oilrig image (with δ_{max} = 0.5 and Gaussian Noise with $\sigma_{Contoli}$ = 20). (e) Store image (with T_{size} = 300, adaptive orientation, T_{diff} = 10.0, s_{theta} = 0.2, D_{bh} = 0.040 and D_{bv} = 0.020). (f) River channel image. (g) Boat deck original image from the Ade20k dataset. (h) Boat deck image (with θ by pixel, D_{bh} = 0.040, D_{bv} = 0.020, inverted colors smoothing filter).



(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure A.2: Images used in validation which are generated by other woodcut synthesis methods. (a) Image from Mello et al. work (MELLO; JUNG; WALTER, 2007). (b) Image from Winnemoller's work (WINNEMÖLLER, 2011). (c) Image from Li et al. work (LI; XU, 2016).

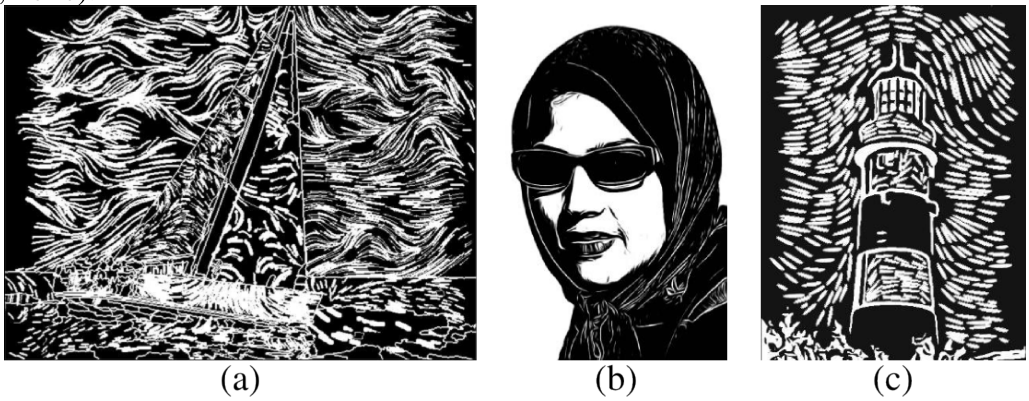


Figure A.3: Images used in validation which are generated by other NPR works with RD. (a) Image from Chi et al. (CHI; LIU; HSU, 2016). (b) Image from Chi et al. (CHI; LIU; HSU, 2016).

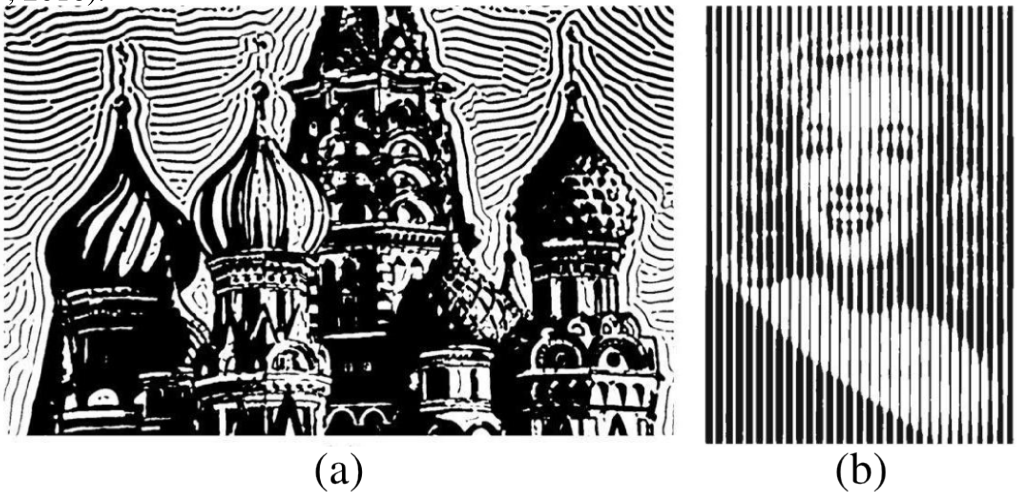


Figure A.4: Images used in validation which are generated by other NPR works with black-and-white renderings. (a) Image from Xu and Kaplan method (XU; KAPLAN, 2008) (source: (LAI; ROSIN, 2013)). (b) Imagem from Mould's method (MOULD, 2003) (source: (LAI; ROSIN, 2013)).

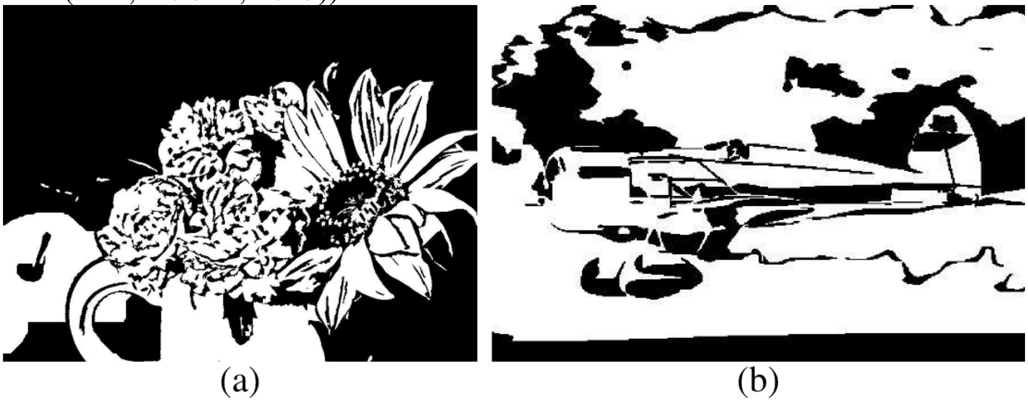
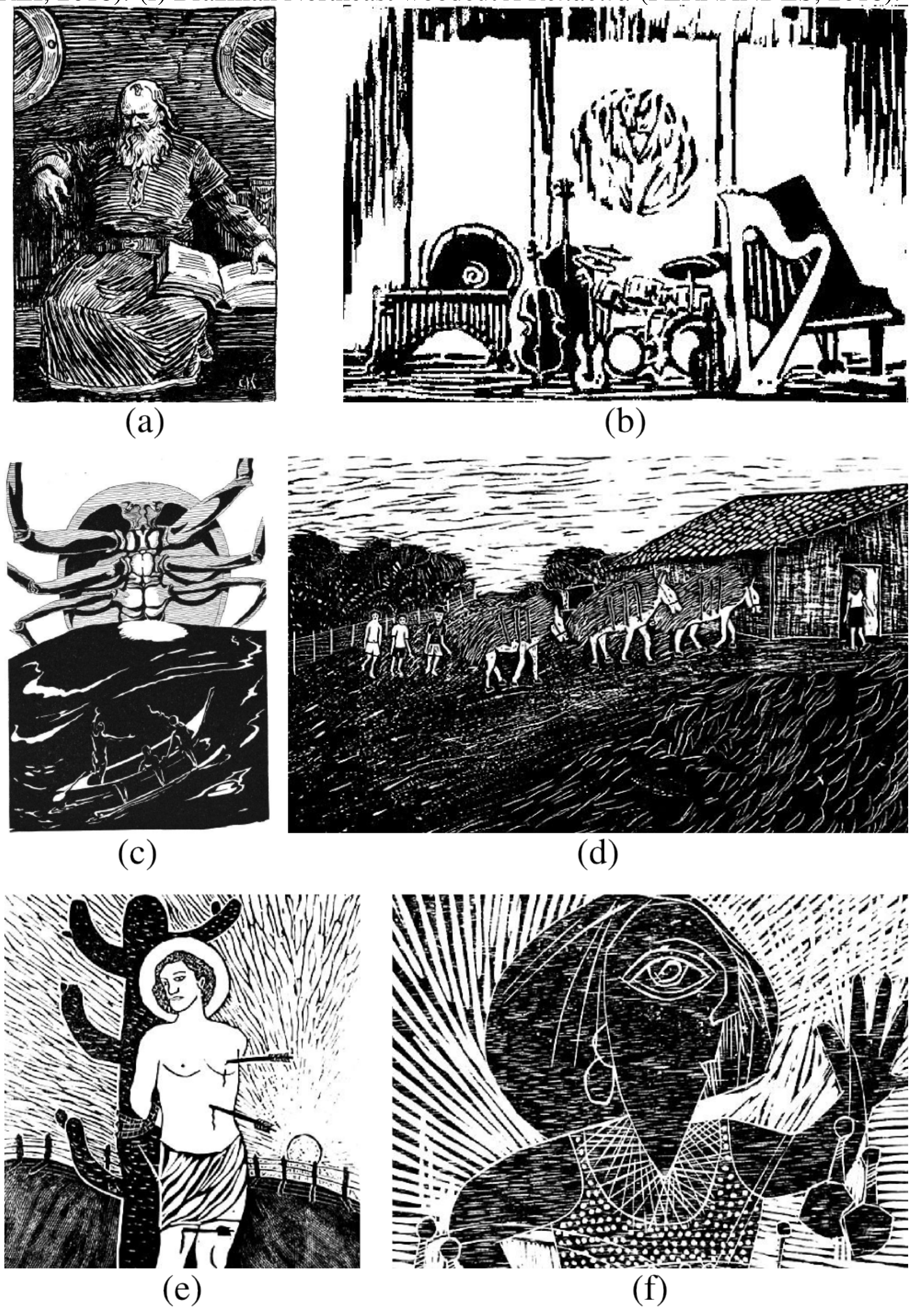


Figure A.5: Real woodcut images used in validation. Brazilian Northeast woodcuts were treated to pure black and white to be in the same color palette as other images. (a) Medieval-like woodcut of a Viking (KROGH, 1890). (b) Contemporaneous woodcut of musical instruments (detail) (BRAZELTON, 1975). (c) Modern woodcut for book illustrations of a crab (KIPLING, 2006). (d) Brazilian Northeast woodcut *Meninos Cambiteiros* (GONZAGA, 2018). (e) Brazilian Northeast woodcut *São Sebastião* (FRANCORLI, 2018). (f) Brazilian Northeast woodcut *A Rendeira* (FERNANDES, 2018).

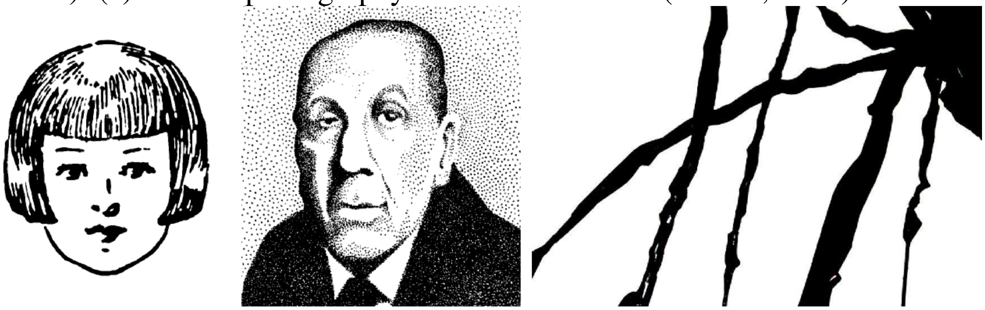


(a) (b)

(c) (d)

(e) (f)

Figure A.6: Other real black-and-white artwork images used in validation. (a) Line drawing of a girl (FORESMAN, 2008). (b) Stippled drawing of Jorge Luis Borges (BHUMIYA, 2006). (c) Treated photography to black-and-white (DEAK, 2011).



(a) (b) (c)

APPENDIX B — RESULTS OF THE VALIDATION PROCESS

Table B.1: Age ranges of participants in the validation step.

Age	Number of Participants
1-19	10
20-29	51
30-39	6
40-49	1
50-59	1
60+	3

Table B.2: Gender of participants in the validation step.

Gender	Number of Participants
Feminine	14
Masculine	58

Table B.3: Educational Background of participants in the validation step. Since the participants are Brazilian, the categories relate to the levels in the Brazilian educational system.

Educational Background	Number of Participants
Elementary school (Ensino Fundamental)	1
High school (Ensino Médio)	19
Technical education (without University Degree)	8
Graduation	36
Master’s Degree	6
Doctor’s Degree	2

Table B.4: Experience with Woodcuts.

Experience with Woodcuts	Number of Participants
Worked with it	2
Heard about it, but never worked with it	51
Never heard about it	19

Table B.5: Experience with NPR.

Experience with NPR	Number of Participants
Worked with it	3
Heard about it, but never worked with it	25
Never heard about it	44

Table B.6: Results for Fig. A.1a. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	5
2	10
3	9
4	17
5	31
Mean by Category	**Grade**
Mean	3.8194
Mean (Experience with woodcuts)	3.0000
Mean (Only heard about woodcuts)	3.7843
Mean (Did not know about woodcuts)	4.0000
Mean (Experience with NPR)	3.3333
Mean (Only heard about NPR)	3.4800
Mean (Did not know about NPR)	4.0455

Table B.7: Results for Fig. A.1b. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	3
2	10
3	11
4	17
5	31
Mean by Category	**Grade**
Mean	3.8750
Mean (Experience with woodcuts)	3.5000
Mean (Only heard about woodcuts)	3.7843
Mean (Did not know about woodcuts)	4.1579
Mean (Experience with NPR)	3.3333
Mean (Only heard about NPR)	3.7600
Mean (Did not know about NPR)	3.9773

Table B.8: Results for Fig. A.1c. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	4
2	9
3	13
4	15
5	31
Mean by Category	**Grade**
Mean	3.8333
Mean (Experience with woodcuts)	4.0000
Mean (Only heard about woodcuts)	3.8039
Mean (Did not know about woodcuts)	3.8947
Mean (Experience with NPR)	3.0000
Mean (Only heard about NPR)	3.5200
Mean (Did not know about NPR)	4.0682

Table B.9: Results for Fig. A.1d. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	6
2	10
3	8
4	17
5	31
Mean by Category	**Grade**
Mean	3.7917
Mean (Experience with woodcuts)	1.0000
Mean (Only heard about woodcuts)	3.7647
Mean (Did not know about woodcuts)	4.1579
Mean (Experience with NPR)	2.0000
Mean (Only heard about NPR)	3.7200
Mean (Did not know about NPR)	3.9545

Table B.10: Results for Fig. A.1e. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	5
2	6
3	8
4	20
5	33
Mean by Category	**Grade**
Mean	3.9722
Mean (Experience with woodcuts)	2.0000
Mean (Only heard about woodcuts)	3.9608
Mean (Did not know about woodcuts)	4.2105
Mean (Experience with NPR)	2.6667
Mean (Only heard about NPR)	3.7600
Mean (Did not know about NPR)	4.1818

Table B.11: Results for Fig. A.1f. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	4
2	8
3	11
4	21
5	28
Mean by Category	**Grade**
Mean	3.8472
Mean (Experience with woodcuts)	3.5000
Mean (Only heard about woodcuts)	3.8235
Mean (Did not know about woodcuts)	3.9474
Mean (Experience with NPR)	3.0000
Mean (Only heard about NPR)	3.6400
Mean (Did not know about NPR)	4.0227

Table B.12: Results for Fig. A.1h. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	3
2	12
3	6
4	22
5	29
Mean by Category	**Grade**
Mean	3.8611
Mean (Experience with woodcuts)	3.5000
Mean (Only heard about woodcuts)	3.7843
Mean (Did not know about woodcuts)	4.1053
Mean (Experience with NPR)	3.0000
Mean (Only heard about NPR)	3.6400
Mean (Did not know about NPR)	4.0455

Table B.13: Results for Fig. A.2a. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	5
2	12
3	11
4	23
5	21
Mean by Category	**Grade**
Mean	3.5972
Mean (Experience with woodcuts)	2.5000
Mean (Only heard about woodcuts)	3.5686
Mean (Did not know about woodcuts)	3.7895
Mean (Experience with NPR)	3.3333
Mean (Only heard about NPR)	3.8800
Mean (Did not know about NPR)	3.4545

Table B.14: Results for Fig. A.2b. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	12
2	19
3	19
4	12
5	10
Mean by Category	**Grade**
Mean	2.8472
Mean (Experience with woodcuts)	2.0000
Mean (Only heard about woodcuts)	2.8039
Mean (Did not know about woodcuts)	3.0526
Mean (Experience with NPR)	4.0000
Mean (Only heard about NPR)	2.5200
Mean (Did not know about NPR)	2.9545

Table B.15: Results for Fig. A.2c. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	0
2	7
3	9
4	19
5	37
Mean by Category	**Grade**
Mean	4.1944
Mean (Experience with woodcuts)	2.0000
Mean (Only heard about woodcuts)	4.2157
Mean (Did not know about woodcuts)	4.3684
Mean (Experience with NPR)	4.0000
Mean (Only heard about NPR)	4.1200
Mean (Did not know about NPR)	4.2500

Table B.16: Results for Fig. A.3a. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	4
2	12
3	15
4	25
5	16
Mean by Category	**Grade**
Mean	3.5139
Mean (Experience with woodcuts)	3.0000
Mean (Only heard about woodcuts)	3.4902
Mean (Did not know about woodcuts)	3.6316
Mean (Experience with NPR)	3.6667
Mean (Only heard about NPR)	3.6000
Mean (Did not know about NPR)	3.4545

Table B.17: Results for Fig. A.3b. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	14
2	19
3	13
4	15
5	11
Mean by Category	**Grade**
Mean	2.8611
Mean (Experience with woodcuts)	4.5000
Mean (Only heard about woodcuts)	2.7647
Mean (Did not know about woodcuts)	2.9474
Mean (Experience with NPR)	3.6667
Mean (Only heard about NPR)	2.9200
Mean (Did not know about NPR)	2.7727

Table B.18: Results for Fig. A.4a. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	4
2	16
3	12
4	16
5	24
Mean by Category	**Grade**
Mean	3.5556
Mean (Experience with woodcuts)	3.5000
Mean (Only heard about woodcuts)	3.4902
Mean (Did not know about woodcuts)	3.7368
Mean (Experience with NPR)	3.6667
Mean (Only heard about NPR)	3.5600
Mean (Did not know about NPR)	3.5455

Table B.19: Results for Fig. A.4b. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	3
2	13
3	10
4	22
5	24
Mean by Category	**Grade**
Mean	3.7083
Mean (Experience with woodcuts)	3.5000
Mean (Only heard about woodcuts)	3.5294
Mean (Did not know about woodcuts)	4.2105
Mean (Experience with NPR)	3.6667
Mean (Only heard about NPR)	3.4800
Mean (Did not know about NPR)	3.8409

Table B.20: Results for Fig. A.5a. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	4
2	9
3	8
4	19
5	32
Mean by Category	**Grade**
Mean	3.9167
Mean (Experience with woodcuts)	2.5000
Mean (Only heard about woodcuts)	4.0196
Mean (Did not know about woodcuts)	3.7895
Mean (Experience with NPR)	4.0000
Mean (Only heard about NPR)	4.3200
Mean (Did not know about NPR)	3.6818

Table B.21: Results for Fig. A.5b. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	1
2	6
3	12
4	22
5	31
Mean by Category	**Grade**
Mean	4.0556
Mean (Experience with woodcuts)	3.0000
Mean (Only heard about woodcuts)	4.0000
Mean (Did not know about woodcuts)	4.3158
Mean (Experience with NPR)	4.6667
Mean (Only heard about NPR)	3.8800
Mean (Did not know about NPR)	4.1136

Table B.22: Results for Fig. A.5c. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	18
2	16
3	10
4	17
5	11
Mean by Category	**Grade**
Mean	2.8194
Mean (Experience with woodcuts)	4.0000
Mean (Only heard about woodcuts)	2.8824
Mean (Did not know about woodcuts)	2.5263
Mean (Experience with NPR)	4.0000
Mean (Only heard about NPR)	3.0000
Mean (Did not know about NPR)	2.6364

Table B.23: Results for Fig. A.5d. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	7
2	11
3	11
4	17
5	26
Mean by Category	**Grade**
Mean	3.6111
Mean (Experience with woodcuts)	2.5000
Mean (Only heard about woodcuts)	3.7059
Mean (Did not know about woodcuts)	3.4737
Mean (Experience with NPR)	3.3333
Mean (Only heard about NPR)	4.0000
Mean (Did not know about NPR)	3.4091

Table B.24: Results for Fig. A.5e. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	6
2	11
3	8
4	18
5	29
Mean by Category	**Grade**
Mean	3.7361
Mean (Experience with woodcuts)	3.0000
Mean (Only heard about woodcuts)	3.7451
Mean (Did not know about woodcuts)	3.7895
Mean (Experience with NPR)	4.3333
Mean (Only heard about NPR)	3.9200
Mean (Did not know about NPR)	3.5909

Table B.25: Results for Fig. A.5f. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	11
2	4
3	11
4	17
5	29
Mean by Category	**Grade**
Mean	3.6806
Mean (Experience with woodcuts)	3.0000
Mean (Only heard about woodcuts)	3.6667
Mean (Did not know about woodcuts)	3.7895
Mean (Experience with NPR)	4.6667
Mean (Only heard about NPR)	3.8800
Mean (Did not know about NPR)	3.5000

Table B.26: Results for Fig. A.6a. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	1
2	6
3	3
4	29
5	33
Mean by Category	**Grade**
Mean	4.2083
Mean (Experience with woodcuts)	3.0000
Mean (Only heard about woodcuts)	4.2157
Mean (Did not know about woodcuts)	4.3158
Mean (Experience with NPR)	4.6667
Mean (Only heard about NPR)	4.0400
Mean (Did not know about NPR)	4.2727

Table B.27: Results for Fig. A.6b. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	15
2	14
3	16
4	14
5	13
Mean by Category	**Grade**
Mean	2.9444
Mean (Experience with woodcuts)	3.5000
Mean (Only heard about woodcuts)	2.8824
Mean (Did not know about woodcuts)	3.0526
Mean (Experience with NPR)	4.0000
Mean (Only heard about NPR)	2.6800
Mean (Did not know about NPR)	3.0227

Table B.28: Results for Fig. A.6c. The first rows show, for each grade (first column), the number of participants who gave that grade for the image (second column). The last columns show, in the second column, the mean grade of that image and the mean grade given by each category of participants regarding their level of experience in woodcuts or NPR.

Grade	**Number of Participants**
1	4
2	8
3	6
4	16
5	38
Mean by Category	**Grade**
Mean	4.0556
Mean (Experience with woodcuts)	4.0000
Mean (Only heard about woodcuts)	3.8627
Mean (Did not know about woodcuts)	4.5789
Mean (Experience with NPR)	4.0000
Mean (Only heard about NPR)	3.6000
Mean (Did not know about NPR)	4.3182

Table B.29: Mean grade by category of participant regarding their level of experience in woodcuts or NPR for all images used in the qualitative validation.

Mean by Category	**Grade**
Mean	3.6655
Mean (Experience with woodcuts)	3.0435
Mean (Only heard about woodcuts)	3.6326
Mean (Did not know about woodcuts)	3.8192
Mean (Experience with NPR)	3.6522
Mean (Only heard about NPR)	3.6052
Mean (Did not know about NPR)	3.7006

Table B.30: Mean grade by category of participant regarding their level of experience in woodcuts or NPR for all images produced by our methodology.

Mean by Category	**Grade**
Mean	3.8571
Mean (Experience with woodcuts)	2.9286
Mean (Only heard about woodcuts)	3.8151
Mean (Did not know about woodcuts)	4.0677
Mean (Experience with NPR)	2.9048
Mean (Only heard about NPR)	3.6457
Mean (Did not know about NPR)	4.0422

Table B.31: Mean grade by category of participant regarding their level of experience in woodcuts or NPR for all images produced by other computational methods to synthesize woodcuts.

Mean by Category	Grade
Mean	3.5463
Mean (Experience with woodcuts)	2.1667
Mean (Only heard about woodcuts)	3.5294
Mean (Did not know about woodcuts)	3.7368
Mean (Experience with NPR)	3.7778
Mean (Only heard about NPR)	3.5067
Mean (Did not know about NPR)	3.5530

Table B.32: Mean grade by category of participant regarding their level of experience in woodcuts or NPR for other NPR works using RD.

Mean by Category	Grade
Mean	3.1875
Mean (Experience with woodcuts)	3.7500
Mean (Only heard about woodcuts)	3.1275
Mean (Did not know about woodcuts)	3.2895
Mean (Experience with NPR)	3.6667
Mean (Only heard about NPR)	3.2600
Mean (Did not know about NPR)	3.1136

Table B.33: Mean grade by category of participant regarding their level of experience in woodcuts or NPR for other NPR works with black-and-white renderings.

Mean by Category	Grade
Mean	3.6319
Mean (Experience with woodcuts)	3.5000
Mean (Only heard about woodcuts)	3.5098
Mean (Did not know about woodcuts)	3.9737
Mean (Experience with NPR)	3.6667
Mean (Only heard about NPR)	3.5200
Mean (Did not know about NPR)	3.6932

Table B.34: Mean grade by category of participant regarding their level of experience in woodcuts or NPR for real woodcuts which are not from Brazilian Northeast.

Mean by Category	Grade
Mean	3.5972
Mean (Experience with woodcuts)	3.1667
Mean (Only heard about woodcuts)	3.6340
Mean (Did not know about woodcuts)	3.5439
Mean (Experience with NPR)	4.2222
Mean (Only heard about NPR)	3.7333
Mean (Did not know about NPR)	3.4773

Table B.35: Mean grade by category of participant regarding their level of experience in woodcuts or NPR for real woodcuts which are from Brazilian Northeast.

Mean by Category	**Grade**
Mean	3.6759
Mean (Experience with woodcuts)	2.8333
Mean (Only heard about woodcuts)	3.7059
Mean (Did not know about woodcuts)	3.6842
Mean (Experience with NPR)	4.1111
Mean (Only heard about NPR)	3.9333
Mean (Did not know about NPR)	3.5000

Table B.36: Mean grade by category of participant regarding their level of experience in woodcuts or NPR for all real woodcuts images.

Mean by Category	**Grade**
Mean	3.6366
Mean (Experience with woodcuts)	3.0000
Mean (Only heard about woodcuts)	3.6699
Mean (Did not know about woodcuts)	3.6140
Mean (Experience with NPR)	4.1667
Mean (Only heard about NPR)	3.8333
Mean (Did not know about NPR)	3.4886

Table B.37: Mean grade by category of participant regarding their level of experience in woodcuts or NPR for other real black-and-white artworks.

Mean by Category	**Grade**
Mean	3.7361
Mean (Experience with woodcuts)	3.5000
Mean (Only heard about woodcuts)	3.6536
Mean (Did not know about woodcuts)	3.9825
Mean (Experience with NPR)	4.2222
Mean (Only heard about NPR)	3.4400
Mean (Did not know about NPR)	3.8712