Universität Zürich[UZH]

Graffiti Stencils

Automated creation of graffiti stencils

Joël Rüttimann

16-940-892

Bachelor's Thesis BSc. Informatics

Department of Informatics

University of Zurich

Supervisors:

Prof Dr. Renato Pajarola

Lars Zawallich

25.10.2022

Abstract

Graffiti are omnipresent in today's urban areas. A special form of graffiti are stencils which are mostly two-tone and easy to replicate. The research objective of the present thesis is to evaluate whether it is possible to automatically create stencils from an arbitrary image. While research on image abstraction is present in relevant literature, automatic stencil creation is understudied in scholarly work. In order to automatically create a stencil, a known algorithm is implemented and extended to create artistically pleasing stencils. The created stencils are then assessed using guidelines driven by research and expert input. Consequently, this work shows that the combination of already existing algorithms with carefully chosen parameters leads to the production of objectively well-made stencils.

Contents

List of Tables

List of Figures

1 Introduction

Modern-style graffitis, a visual communication tool in public space, are omnipresent in today's society. However, there exist multiple different styles of graffitis in the modern era. Whether those are tags, which simply are the artist's signature sprayed in one colour, or stylistic symbols or phrases in different colours. However, a specific type of graffiti are stencils. Made out of cardboard or paper, stencils allow to replicate an image multiple times. Using a stencil allows to transfer the image on a surface by using paint. Unlike traditional graffiti which most people perceive as colourful and eye-catching, stencils are mostly two-coloured to be easily transferable. The most famous stencil artist worldwide known is Banksy whose graffiti are mainly black and white.

In order to produce stencils, images have to be simplified and reduced to the essential parts that still leave the image recognisable. Hence, an abstract image has to be produced. The goal of this thesis is not only to produce abstract images, the created images should also be aesthetically pleasing. This raises two questions: First, how is this possible, and second, how can the result be judged? In order to answer the first question, the goal is to create a stencil that can be sprayed on a wall or any other type of surface. But how do we get from a picture to a black and white abstraction, such that the original picture is still recognisable? In an image with colour, the colour plays a huge role as without the colour a lot of detail would be lost. If we now look at the transition from one colour to another colour in an image, there is a visible change. This can be considered as an edge and is further explained in section 2.2. But what can be already said is that these changes, or edges, are important. Thus, it becomes clear that the transition from one colour to another colour has to be preserved during the abstraction process. Furthermore, the main feature of the images should still be visible. But what is a main feature? This is not easy to answer and strongly relies on the viewer. For example, if we look at a portrait, the eyes should still be somewhat visible in the abstracted picture as those are a defining feature of a face. Or in an image of a car, the wheels should be round and not just a black square. The goal of this thesis is to create or implement a program which generates a stencil from an arbitrary input image including an analysis of the result. Thus, apart from computationally producing abstract images, the measurement of the results is a key challenge.

Judging the result objectively is hard if not even impossible. It is more subjective what one would classify as a nice picture while someone else will say the opposite. There exist no exact guidelines or scientific papers on what a good-looking stencil must include. In order to solve this problem the idea is to include and talk tho experts in the art/graffiti world. With the help of experts, a guideline will be developed and then used to judge the results.

To give a better visualization what this thesis intends to achieve, we take a look at an example of a graffiti stencil in figure 4. In the area of stencil art, aforementioned Banksy is the figurehead of the scene. Banksy has presented his work at exhibitions all over the wold and his art is being sold for millions of US-Dollars while remaining anonymous (Ellsworth-Jones 2012). Hence, some of Banksy's artwork is used as a guideline to compare the achieved results to his art. This does not mean to goal is to create exactly the same artwork as that of Banksy as this would be impossible but to analyse whether the results of the implementation go into the preferred direction.

1.1 Graffiti

Since it is the goal to create a specific type of graffiti, the question arises what a graffiti exactly is apart from the brief description given in section 1. The singular of graffiti is graffito which is Italian and means "to engrave". The definition of graffito is "a drawing scratched on a wall" (Campbell 2003).

The first examples of what can be considered as a graffiti dates back to the pharaohs in Egypt (Frood and Ragazzoli 2013). Graffiti are usually placed at highly frequented places, such that they can be seen by a wide variate of spectators (Lohmann 2020).

(a) Graffito, Kom Ombo Temple, Egypt

(b) Graffito, Ancient Graffiti Museum Marsilly, France

Figure 1: Ancient Graffiti (Rémih 2009; Helfer 2006)

By looking at the graffiti in figure 1 and the pure definition of the word graffiti/graffito given before, quite a lot can be judged in a graffiti. Usually, if somebody thinks about graffiti, they do not have an ancient image scratched to a wall in mind but instead the colourful, bold letters that are sprayed on bridges or walls. What most people think of graffiti can be referred to as the modern day graffiti introduced above. Modern day graffiti mostly originated from New York and there are large varieties of different styles of graffiti (Dimitri Ehrlich 2006). Additional pictures and information on these graffiti can be found in appendix C.

(a) Heavily tagged subway car of the NYC subway graffiti

(b) 1970's NYC Subway Graffiti

Figure 2: Modern Graffiti (Calonius 1973; alphabetcityblog 2008)

In figure 2 picture b) an example of a modern day graffiti from the New York subway is displayed.

1.2 Graffiti stencils

While the previously given description and definition focused on graffiti in general, the following part introduces stencils which are a specific type of graffiti.

1.2.1 Definition

A piece of cardboard, paper or another medium is normally taken and a cut-out template with the help of which outlines, patterns, characters or else is produced. This is called stencil, a template, which is then used to spray the outline onto a wall or any other surface (Truman 2010). Hence, a graffiti stencil refers to the method of producing the graffiti and the specific sub-type of graffiti simultaneously.

(a) Stencil example

(b) Stencil in Lagos, Portugal

Figure 3: Example of stencils, a) from Put (2020) b) photographed by Rüttimann 2022

There are other techniques to display the graffiti stencil as well but spraying is the most common one. This is done in two tones most of the time, but some stencils do have more colours (e.g. the coloured heart in figure 3 a)). Hereby, the difficult step is to get from a picture to a point where the picture can be transferred into two tones while the original picture is still recognisable. Stencils are not only special graffiti which can be quickly replicated. A lot of the artwork of stencil artists is often politically influenced and communicates ideas or emotions on specific topics (Philipps 2015).

Figure 4 is an artwork from the previous mentioned artist Banksy. A lot of Banksy's artwork do communicate a political opinion or current societal problems and challenges (D'Cruz et al. 2010).

1.3 Abstraction

One definition of "abstraction" is "the action of removing something from something else; the process of being removed from something else" (Dictionaries 2022). Keeping this definition in mind, how do we get from an image to a stencil? Looking at figure 4, which a stencil Banksy sprayed to a wall in Beit Sahour in the West Bank, the interesting question is how it was created.

Figure 4: Flower Thrower by Banksy in the West Bank (Levinger 2005)

Except for the flower bouquet, the stencil is two-toned. Black paint is used to spray the stencil while the other defining colour of the stencil is defined by the surface, the background, it was sprayed upon. In this case it is a white wall. Apart from these two tones, it is clearly visible that the stencil shows a man throwing a flower bouquet. What can be seen as well is that the main features of the person are still intact. Such as the eyes or the cap of the flower thrower. Looking at the definition of "abstraction", there have clearly been parts or features that have been taken away from the original picture such as the colour or more refined details of the person. With that, it can be said that this picture is an abstraction of the original coloured picture and that parts have been

removed whilst the picture is still recognisable. Consequently, this is what is meant when talking about abstraction and important features.

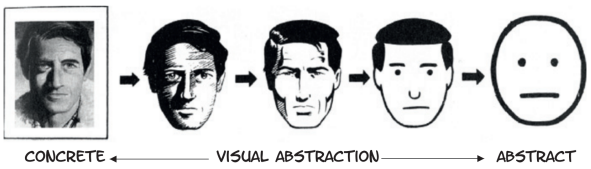


Figure 5: Abstraction example (Winnemöller et al. 2012)

As can bee seen in figure 5, the most right picture still resembles a face. It is still recognisable what has been there in the original image: A face. For sure, the more right you go, the less you can recognise the the man on the left. But the essence of the picture, it displaying a man, can still be seen in the most right, abstracted, picture. What is to say about the "main feature" of on image, the feature differs from image to image and defining what actually should be preserved is objective and has to be judged from picture to picture. The main challenge is to automate this abstraction while still keeping the individual features of each specific original image.

1.4 Guidelines to judge stencils

To judge the result of this thesis, two experts have been asked about the features of a good-looking stencil and what characteristics it must have. With the help of the experts a guideline was developed to judge the automatically produced stencils. The first expert I talked to is Noah Hertzog, a student of visual communication in Basel. The other expert is Michael Müller an artist with experience in the creation of graffiti stencils (Müller 2022).

With these two experts the following guideline was developed. First, the original image still needs to be recognisable. Second, the stencil has to be an abstraction of the original image, while the main features are preserved. Noah pointed to figure 6 to illustrate his thoughts on abstraction. Third, the different parts of the original image have to be distinguishable, for example the face should be separated from the body. Fourth, the level of detail has to be considered. The stencil should have less details than the original image. Small details are harder to spray and if the stencil is appraised from further away, small details disappear. Less detail is most of the time better. Fifth, larger black and white areas should be present such that a contrast is visible.

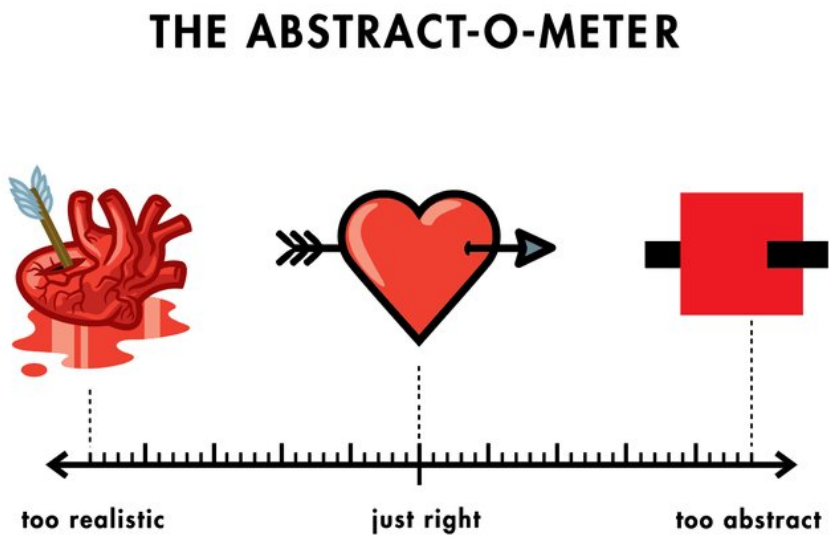


Figure 6: Level of abstraction and detail (Niemann n.a.)

2 Techniques

As mentioned in the introduction, the main objective is "choosing or designing an algorithm which generates stencils from an arbitrary image, considering the literature". Often designing an algorithm by oneself can be more interesting than just choosing one which already exists. Therefore, it is decided to design and develop an algorithm. As this is a big challenge, the process of designing and developing an algorithm is described in detail below.

In the following section some techniques which could be useful to create an abstracted image are presented. The idea was to combine and adjust them in such a way that in the end a stencil can be extracted from an image. As mentioned in section 1.3, details of the original image have to be extracted, altered, some preserved and simplified. Most of the techniques focus on edge detection whereby the concept is described in section 2.2.

2.1 Canny edge detector

The canny edge detector was developed by John F. Canny in 1986 (Canny 1986). The canny edge detector follows these subsequent steps (Ding and Goshtasby 2001). First, reduce noise. Second, determine gradient magnitude and gradient direction at each pixel. Third, non-maximum suppression. Fourth double threshold and fifth edge tracking by hysteresis. Figure 7 is an example result of the canny edge detector by Sahir (2019*a*). One could argue that the canny edge detectors is an algorithm itself and does not belong into the technique section and the description given is rather short. Although this is true, it is argued that the canny edge detector still belongs in this section because it was the

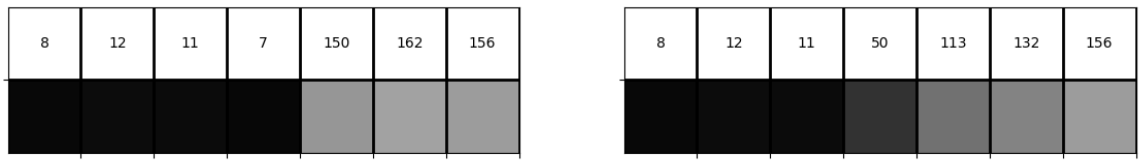
(a) Canny edge detection (b) Canny edge detector 2

Figure 7: Canny edge detection by Sahir (2019*a*)

gateway and inspiration to look deeper into edges, how they work, and edges detectors in general. More specific, the explanation on how the canny edge detector works by Sahir (2019*a*) delivered the inspiration. The canny edge detector will be used in the end to compare the implementation to see if and what improvements have been made during the time of this thesis.

2.2 Edge

In order to apply edge detection, it is crucial to know what an edge is. According to Jain (1989), an edge is a line which divides an image into different parts, or the image brightness having discontinuities at this point. To find these lines, the different intensity variations occurring in different points on the image can be used. An edge is where a discontinuity appears (Dharampal 2015).



(a) Clear edge (b) Difficult edge

Figure 8: Example of edge in image, Rüttimann 2022

First we take a look at figure 8 a): example of a discontinuity which can be judged as an edge. By looking at the difference values in square 4 and 5. The visual difference can be seen clearly. With the definition from above this would account as an edge. But it is

no always that obvious. Looking at figure 8 b) regarding the values it could be seen as two edges between square three and four and the second edge between square four and five. But visually it can easily be seen as one. And since the computer has no eyes, the computer has to look at the values and judge from there whether it is one or two edges. As seen with this example, it as not always obvious where an edge is located.

Edges in computer vision are thin, mostly 2x2 pixels or sometimes even smaller. The edges are also as accurate as possible (Winnemöller et al. 2012). But this kind of edge is not the right fit for graffiti stencils. For graffiti, wider and bigger edges which have artistic touches are needed (Winnemöller et al. 2012). It is also important to have coherent edges. Therefore, in the following, "artistic edges" is referred to when suited for a graffiti stencil and "edges" for computer vision-like edges.

2.3 Gradient of an image

The gradient by itself is not a technique. But it is used in the final algorithm and is the central part from which all the following steps are influenced. For an image, the gradient is defined as a change in the intensity or colour. Looking at one pixel, the gradient is given by two derivatives, one in the horizontal and one in the vertical axis. At the given pixel, the gradient vector points in the direction of the biggest change as can be seen in figure 9 (Shrivakshan and Chandrasekar 2012). A more technical description is that the gradient of image is the vector of its partials (equation 1) (Gonzalez 2008).

$$\nabla f = \begin{pmatrix} g_x \\ g_y \end{pmatrix} = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix} \tag{1}$$

In equation 1, $\frac{\partial f}{\partial x}$ is the derivative in x direction and $\frac{\partial f}{\partial y}$ the derivative in y direction (Gonzalez 2008).

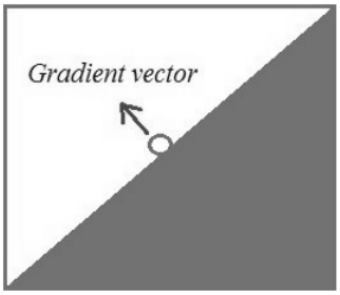


Figure 9: Vector in gradient direction (Shrivakshan and Chandrasekar 2012)

2.4 Gaussian Blur

As well as the gradient, the Gaussian Blur is not an edge technique by itself but it is used extensively in the final algorithm. What happens when an image is blurred is that

each pixel is mixed with its surrounding pixel. More precisely, the blurring is achieved by convolving the image with a low pass filter. This has the effect of removing noise from the image (Haddad et al. 1991). These filters work with convolution and a kernel which will be elaborated upon in the following section.

2.4.1 Convolution

Image convolution is defined by the following equation:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x-dx, y-dy) \tag{2}$$

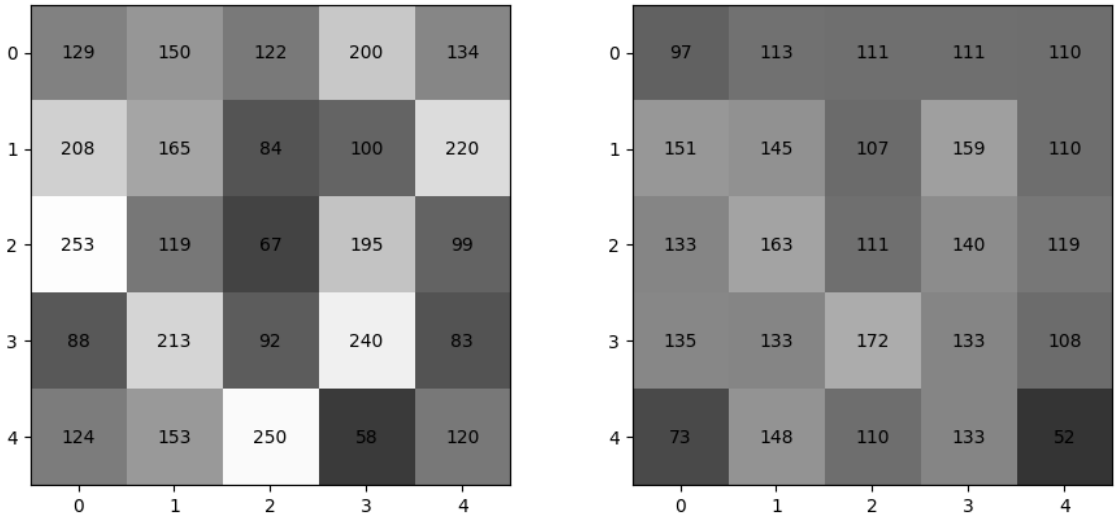
with $g(x,y)$ in equation 2 being the filtered image, $f(x,y)$ being the original image and ω the kernel and a half the kernel width (Akgün and Erdoğmuş 2015). Convolution can be used as filter effect for images. To achieve this a matrix is applied to every pixel in the image. The new pixel values are determined by the applied kernel and the neighbours of the pixel. According to Ludwig "a convolution is done by multiplying a pixel's and its neighboring pixels color value by a matrix" (Ludwig 2013: p. 3). The kernel is a matrix of numbers. The example in table 1 shows how convolution works by taking pixel $1,1$ from figure 10 a) and the kernel in table 1.

0	1	0
1	1	1
0	1	0

*1/5 convolved pixel $p(1,1) = (150+165+208+84+119)/5 = 726/5 = 145.2$

Table 1: Simple Kernel

Figure 10 shows a larger example with the kernel from table 1.



(a) Original Image (b) Convolved Image

Figure 10: Convolution example, Rüttimann 2022

Instead of just using a simple kernel like in table 1, the Gaussian can be used to compute the values in the matrix. The Gaussian function in one dimension is defined as displayed in equation 3 (Lindeberg 1994):

$$G_\sigma(x) = \frac{1}{2\pi\sigma^2} \exp(-\frac{x^2}{2\sigma^2}) \tag{3}$$

In two dimensions it is the product of two Gaussian functions in each direction as displayed in equation 4 (Shapiro et al. 2001):

$$G_\sigma(x) = \frac{1}{2\pi\sigma^2} \exp(-\frac{x^2 + y^2}{2\sigma^2}) \tag{4}$$

Here x and y are the distance to the origin. To clarify further, let p be the current pixel for which the Gaussian blur is calculated and let p_x and p_y be the x and y position in the image. Then let w be a pixel in the image and in the kernel range of p. Then x is the distance from p_x to w_x, similar for y. What might not be obvious from looking at the Gaussian equation is that the further away w is from p, the less influence w has on p. This can be seen by letting the distance between p and w get to infinity. This would mean that x and y go to infinity.

$$\lim x \to \infty, \lim y \to \infty \quad \frac{1}{2\pi\sigma^2} \exp(-\frac{x^2 + y2}{2\sigma^2}) \to \frac{1}{2\pi\sigma^2} \exp(-\inf) = 0 \tag{5}$$

Example of a Gaussian kernel with $\sigma = 2$ is shown in table 2.

0.1553	0.1760	0.1553
0.1760	0.1995	0.1760
0.1553	0.1760	0.1553

Table 2: 3x3 Gaussian Kernel

Two parameters exist for the Gaussian blur, the kernel size and σ. The influence of the kernel size and σ is shown in figure 11. The larger the kernel is and the higher σ, the more blurred is the resulting image.

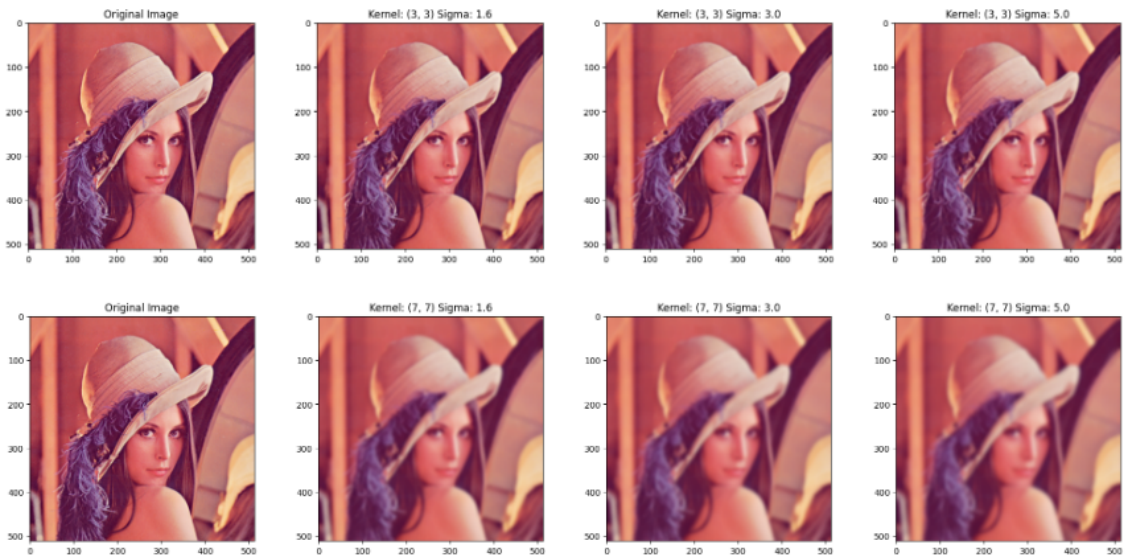


Figure 11: Influence of different kernel sizes and σ for Gaussian blur, Rüttimann 2022, original picture from Kyprianidis and Döllner (2008)

2.5 Difference of Gaussian

For the difference of Gaussian, a Gaussian blurred image is subtracted from a less blurred version. In the following, the difference of Gaussian is referred to as DoG. Actually, DoG is not an edge detector by itself and cannot be compared directly to other standard computer vision techniques (Winnemöller et al. 2012). Nevertheless, the DoG is quite important since the final implementation relies on an enhanced DoG. The DoG algorithm removes high-frequency spatial components which represents noise and enhances edges (Misra et al. 2019). To explain the DoG more simply, one can look at it from a signal processing point of view. Looking at the Gaussian, it can be considered as a low pass filter. This means that low spatial frequencies can pass, while higher spatial frequencies are eliminated. Looking at the difference of two Gaussian's with different *sigma*, they create a band pass filter (Winnemöller et al. 2012). A band pass filter is a filter which only lets signals pass in a frequency band. The frequency ranges below and above the band are blocked or significantly attenuated (Shenoi 2005).

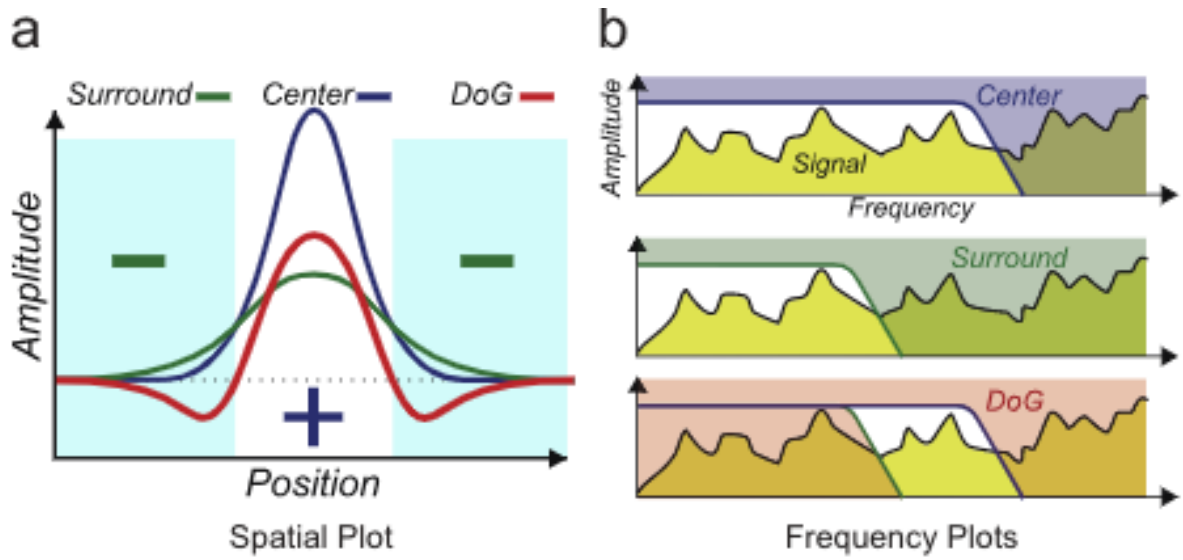


Figure 12: Band Pass Filter explained (Winnemöller et al. 2012)

In figure 12 in picture a, the green is a surrounding Gaussian and blue is the inner Gaussian. The surrounding Gaussian is subtracted from the center Gaussian to produce the DoG. In figure 12 picture b, the center is a low pass filter cutting off all frequencies above a threshold. Same goes for the surrounding Gaussian. When the difference between the Gaussian is taken the DoG is created. The frequencies which "survive" this process are the ones in bright yellow in the last plot in figure 12 picture b. The DoG will then extract the image features within this characteristic band and those features tend to correspond to edges (Pallás-Areny and Webster 1999). The DoG is usually applied to a greyscale image (Wang et al. 2012). The Dog for one pixel x, y is given by equation 6 with $v = (x, y)$ (Winnemöller et al. 2012).

$$G_\sigma(v) - G_{\sigma_r}(v) \tag{6}$$

Most of the time, $r = 1.6$ is used to approximate the Laplacian of Gaussian filter. Also, $r = 1.6$ is a good compromise between accurate approximation and reasonable sensitivity (Marr and Hildreth 1980). Kernel size and *sigma* are the two parameters for the DoG, the influence of them is illustrated in figure 13.

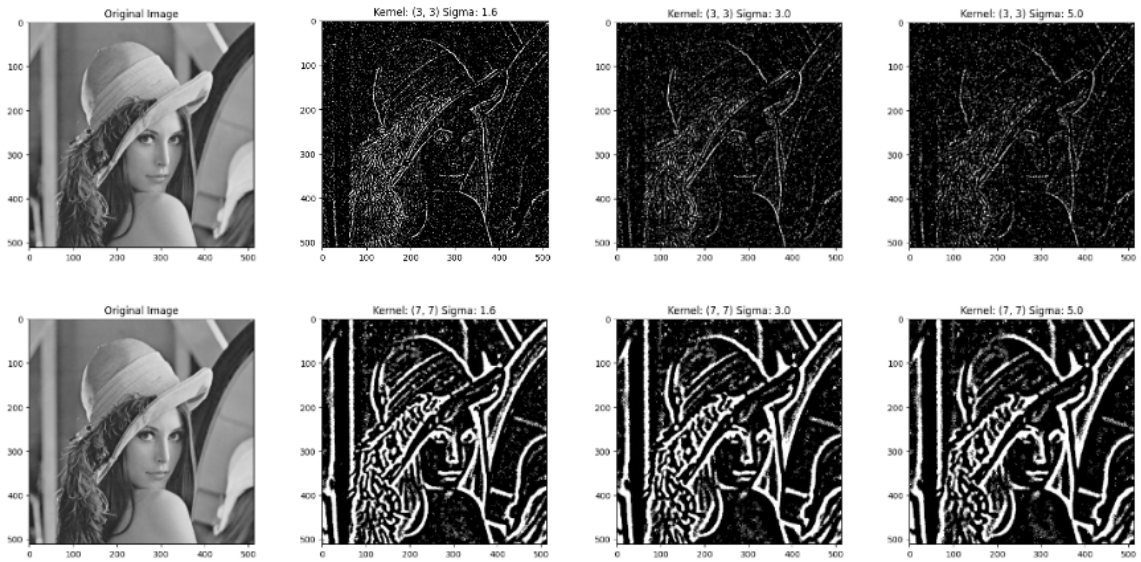


Figure 13: Influence of different kernel sizes and σ and r=1.6, Rüttimann 2022

2.6 Sobel Operator

A Sobel Operator is an algorithm which emphasises edges in images. It is a discrete differentiation operator computing an approximation of the gradient intensity function. The algorithm is based on convolving the image with a small filter in x and y direction of the image (Sobel and Feldman 2015). Therefore, the computations are restively inexpensive. But there is also a down side, the gradient approximation is rather crude, particularly for high-frequency variations of the image (Sobel and Feldman 2015). Table 3 and 4 is an example if $3x3$ kernel which is used for the Sobel filter.

+1	0	-1
+2	0	-2
+1	0	-1

Table 3: Sobel filter in x direction

+1	+2	+1
0	0	0
-1	-2	-1

Table 4: Sobel filter in y direction

For a pixel $p(x, x)$ in the image I, the filter is calculated by the following equation 7 (Shrivakshan and Chandrasekar 2012).

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * I \quad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * I \tag{7}$$

Here I denotes the input picture and $*$ the convolution operation. The final result,

the gradient magnitude, is given by equation 8 (Kanopoulos et al. 1988).

$$G = \sqrt{G_x^2 + G_y^2} \tag{8}$$

Figure 14 is an example of the Sobel Operator applied to example picture.

(a) Sobel filter (b) Sobel filter 2

Figure 14: Sobel Operator examples, Rüttimann 2022

2.7 K-Means clustering

Edge detection is not the only approach for an image abstraction. Another idea was to divide the image into different areas. Furthermore, the different areas should all have the same colours. To achieve this, a k-means clustering algorithm could have been used. However, a downfall of this approach is that the number of clusters k should have been known before starting the algorithm. The sketch of the k means clustering would involve the following steps. First, choosing the number of clusters, k. Second, randomly assign the data points to any of the k clusters. Third, calculate the center of the clusters. Fourth, calculate the distance of the data points from the centers of each of the clusters. Fifth, depending on the distance of each data point from the cluster, reassign the data points to the nearest clusters. Sixth, calculate the new cluster center. Seventh, repeat steps 4, 5 and 6 until data points do not change the clusters or until the assigned number of iterations is reached (Timbers et al. 2022). The results of this technique are not promising by itself since the abstraction level is not high enough. However, the idea was to combine it with a technique mentioned before. Figure 15 is an example of an image with different number of clusters.

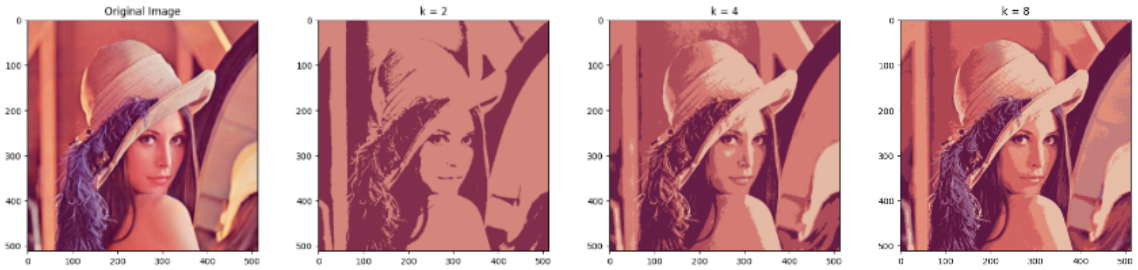


Figure 15: k-Means clustering, Rüttimann 2022

2.8 Seam Carving

Seam Carving is used to resize and enlarge images. The process considers geometric constraints and the image content. The image can be reduced and expanded. A seam is a 8-connected optimal path of a pixel from top to bottom or left to right. What is considered an optimal path is defined by the energy function (Avidan and Shamir 2007). The algorithm consists of the following steps. First, assign an energy to every pixel. Second, find the optimal path. Third, delete all the pixels in the path and then repeat the previous steps. The energy for the image I is calculated as displayed in the following equation 9 (Avidan and Shamir 2007).

$$e(I) = |\frac{\partial}{\partial x}I| + |\frac{\partial}{\partial y}I| \tag{9}$$

The partial deviation are computed by $3x3$ Sobel filters. Figure 16 is an example of an image and its energy. The seam with the least energy is calculated by dynamic programming shown in equation 10 (Avidan and Shamir 2007):

$$M = e(i,j) + min(M(i-1,j-1), M(i-1,j), M(i-1,j+1)) \tag{10}$$

This means, that $M[i,j]$ contains the smallest seam at that point in the image considering all the possible seams to this point. The minimum seam can then be found in the last row of this matrix. To find the path, backtracking is applied from the point with the minimum energy (Avidan and Shamir 2007).

In figure 17 a) is an example of the first seam which will be removed in b) is the originally image from 16 a) with 0.75 width of the original image. What can be seen, is that the image is now way smaller but almost all of the details are still there.

2.9 Technique summary

As stated in section 2.2, the goal is to get artistic edges. By looking at the results of the different techniques in the previous sections, the resulting edges are not looking

(a) Original image

(b) Energy of image

Figure 16: Image with energy map, picture taken by Rüttimann 2022

(a) Original image

(b) Image reduced width with seam carving

Figure 17: Seam carving applied, Rüttimann 2022

nice and nowhere near the stencils from Banksy. They are just normal edges and neither in any way artistically pleasing nor do they produce artistic edges. After adjusting the parameters for the techniques and joining different techniques the conclusion is made, that it is not possible to design an algorithm in a reasonable time to create well-made stencils. Therefore, an already existing algorithm with the focus on image abstraction is used.

3 Related work

There are different papers which employ an artistic abstraction of an image. But, there is no paper which does a stencil creation. Some papers which do an image abstraction include steps which could be used to create Banksy-like stencils. In the following section an overview of interesting and related papers are presented. For each paper, the result which could be used for a stencil creation is shown.

3.1 Image abstraction papers

3.1.1 Image abstraction by structure adaptive filtering

This paper, written by Kyprianidis and Döllner (2008), is later discussed in more detail, therefore just some results of this paper are presented here. The baseline of the paper is that they use an enhanced DoG filter which follows the edges. The results of the steps which could be used in the present work are displayed in figure 18

Figure 18: Kyprianidis and Döllner (2008) result

3.1.2 Flow-based image abstraction

Like Kyprianidis and Döllner (2008), Kang et al. (2008) create a flow-based difference of Gaussian. First, an Edge Tangent Flow (ETF) is calculated. This is a flow field which will guide the filter. Then, the DoG filter is steered along the ETF field to enhance the line qualities. Different to a normal DoG, the resulting edges are sharper and have more detail. The FDoG can be applied iteratively to get better results. After this step, a flow-based bilateral filter is applied. Figure 19 shows the results the authors achieved.

3.1.3 An eXtended difference-of-Gaussians compendium including advanced image stylization

Similar as Kyprianidis and Döllner (2008), this paper written by Winnemöller et al. (2012) is later discussed in detail. The important key takeaway of their research is the

Figure 19: Kang et al. (2008) result

suggestion of further enhancement to Dog or FDog. The results of the steps which could used for the present work are shown in figure 20

Figure 20: Winnemöller et al. (2012) result

3.1.4 Tangent-based binary image abstraction

Wu et al. (2017) use a similar approach as Kyprianidis and Döllner (2008). First, the structure tensor is computed the same way as done by Kyprianidis and Döllner (2008). Then, a line integral convolution is applied which follows the vector field from the structure tensor. Afterwards, an enhanced DoG is applied to the line integral convolution result, however not a flow-based DoG. As a last step a binarization with a threshold is applied. The results of the steps which could be used can be seen in figure 21

Figure 21: Wu et al. (2017) result

3.1.5 Stylized black and white images from photographs

Mould and Grant (2008) propose three different algorithms. The first two use energy minimization, using loopy belief propagation and graph cuts to produce binary images. The third algorithm computes a base layer which consist of large flat-coloured regions. This is done by energy minimization. Then, a detail layer containing small high contrast details is computed by adaptive thresholding. The final labeling is done by removing small components and smoothing the region boundaries. The approaches from Mould and Grant (2008) are the only ones not including a Difference of Gaussian. Figure 22 shows the results of their approaches.

Figure 22: Mould and Grant (2008) result

3.2 Choosing the right algorithm

As already stated in the technique summary, an existing algorithm has to be used. Considering all the papers in section 3, Winnemöller et al. (2012) has the best results with respect to a stencil creation. The quality of how the paper is written and the amount of citations are convincing as well. Winnemöller et al. (2012) focuse on an artistic extraction of edges. And this is exactly what is needed for the stencil creation process. But would the XDoG implementation from Winnemöller et al. (2012) be enough to create a stencil? No. The reason is that for a stencil to be cut out, all what should not be cut out has to be coherent.

(a) Holes (b) Example areas which needs attention

Figure 23: Problematic areas, Rüttimann 2022, original image from Winnemöller et al. (2012)

Figure 23 a) is an example result from Winnemöller et al. (2012). When taking a closer look, there are quite a lot of white areas completely surrounded by black areas. If for a stencil the black areas are cut away, the surrounded white areas will be cut away as well. Figure 23 b) is an example of a white areas surrounded by black pixels. The two white areas would fall away if the black is cut out. Therefore, this problem has to be fixed. In order to do this, there are two possibilities: closing small holes or connecting the white areas with each other. Closing the smaller holes can be achieved by comparing the hole area to the image area. If the ratio is below a certain threshold, the hole could be forced to be turned black. Connecting the white areas is more difficult. From where to where should the connection be, how should the the connection line be chosen and how is this possible in a reasonable time are some of the questions to be answered in the following.

Figure 24 a) and b) is an example of a straight connection versus a more natural connection. Clearly the straight connection is the worse fit. In order to speed up the

(a) Straight Connection (b) Natural Connection (c) Border Pixels to Consider

Figure 24: Connection Styles, Rüttimann 2022

process of connecting two areas, not all the pixels in a white area have to be considered.

One can only look the the pixel which are direct next to an edge as shown in figure 24 c) with the red line. From here, there are different approaches thinkable. The first approach is to select the most left, right, top and bottom red pixel and follow the gradient until a white pixel is found, then select the shortest connection. The second approach would be to follow the gradient from each red pixel until a white pixel is found and then use the shortest connection. Or the third approach is to search for the nearest neighbour in two different white areas and connect them.

The idea is, following the gradient would lead to a more natural connection like in figure 24 b). Just connecting the nearest neighbour would lead to straight connections like in figure 24 a). For sure the natural connection would be better suited for a stencil. But just following the gradient did almost never lead to a connection to another white area, and when a connection was found, the result was highly questionable regarding aesthetics. Therefore, the decision was made to use the nearest neighbour approach even if it is the worse fit. But in comparison to following the gradient, the areas are at least connected in the end.

4 Implementation

As stated in section 3, the best fit is the XDoG by Winnemöller et al. (2012). This means the goal is to implement the XDoG and then extend it with closing and connecting the white area problem.

4.1 Parameterisation

The final algorithm has ten parameters. This is more than the original XDoG implementation. To get better looking results each of the $\sigma's$ which control the different Gaussian functions can be adjusted.

Ten parameters are quite a lot of parameters to adjust. On one side, the algorithm can be fine-tuned really nicely. But on the other hand, finding the right parameter settings is also extremely difficult. Winnemöller et al. (2012) did provide the original parameter settings which can be found in table 6. This is a good guideline but the implementation does slightly differ from the implementation done by Winnemöller et al. (2012). The difference of the approach will be clarified in the following section. Winnemöller et al. (2012) use only one σ for the bilateral filter step and one for the gradient direction DoG step and then use the value 1.6 to calculate the second σ. These σ control the Gaussian functions. In the implementation for this thesis, the factor 1.6 has been replaced by two more σ parameters to have a better control of the edge details. In order to make the parameterisation more comprehensible and clearer, they are divided into to steps which will be described in a step-by-step walk-through below. The proposed values have been empirically proved to be a good starting point.

Local orientation estimation

The higher the values, the more coherent are gradient and tangent vector, starting with $\sigma_a = 4$

- σ_a: This value controls the Gaussian for the coherent orientations.

Bilateral filter

The higher the values, the more blurred is the image. $\sigma_d = 3, \sigma_r = 4.25$.

- σ_d: This value controls the first Gaussian.
- σ_r: This value controls the second Gaussian.

DoG in gradient direction

Higher σ values lead to less detailed $\sigma_e = 2, \sigma_f = 1.6\sigma_e, p = 20$ is a good starting point.

- σ_e: This value controls the first Gaussian.
- σ_f: This value controls the second Gaussian.

- p: Sharpening and blurring

Smoothing along tangent flow field
Higher values will produce smoother pictures with more coherent edges. But high values need a lot more of computation time. How big the influence is depends also on the other parameters. Starting with $\sigma = 2$ is usually fine.

- σ_m: This value controls the Gaussian.

Thresholding Higher values turn more of the image black. Starting with $\epsilon = 70, \sigma_z = 2, area = 0.01$ is fine most of the time.

- σ_z: Final smoothing with Gaussian blur.

- ϵ: Controls the threshold for the binary conversation.

Closing and Connecting Holes

- $area$: The percent of an area which will be turned black.

4.2 XDoG implementation

The goal is to implement a version of the XDoG described in Winnemöller et al. (2012) and as the name already says, it is an extended difference of Gaussian. Winnemöller et al. (2012) proposed a re-parameterisation of the DoG and then apply a thresholding as displayed in equations 12 and 13. The re-parameterisation for the DoG is shown in equation 11 and has the effect of sharpening the image.

$$S_\sigma, k, p(x) = (1 + p) * G_\sigma(x) - p * G_{\sigma k}(x) \tag{11}$$

$$T\epsilon(S_\sigma, k, p(x)) \tag{12}$$

$$T\epsilon(u) = \begin{cases} 1 & u \geq \epsilon \\ 0 & otherwise \end{cases} \tag{13}$$

There is another threshold function for the "otherwise" case. This would lead to greyscale images and this is not important for the stencil creation. What is more interesting is that Winnemöller et al. (2012) explained that the XDoG is a stand-alone extension and can be applied to a normal DoG or a FDoG. When looking at the results, the FDoG combined with XDoG gives the best results. Therefore, an FDoG implementation had to be done as well. Winnemöller et al. (2012) proposed two papers which do an FDog implementation: Kyprianidis and Döllner (2008) and Kang et al. (2008). After comparing the two papers, the first idea was to implement Kang et al. (2008) but after some discussions the switch was made to Kyprianidis and Döllner (2008). Figure 25 shows the

overview of how the different steps of the FDoG interact with each other. The last step, "colour quintization", is not of interest for this thesis since this step is not needed for the stencil creation. After each step, the result is passed on to the next step. All the steps are explained in detail in the following part, section 4.3. The equations, referenced with the $*$ symbol have been provided by Kyprianidis and Döllner (2008).

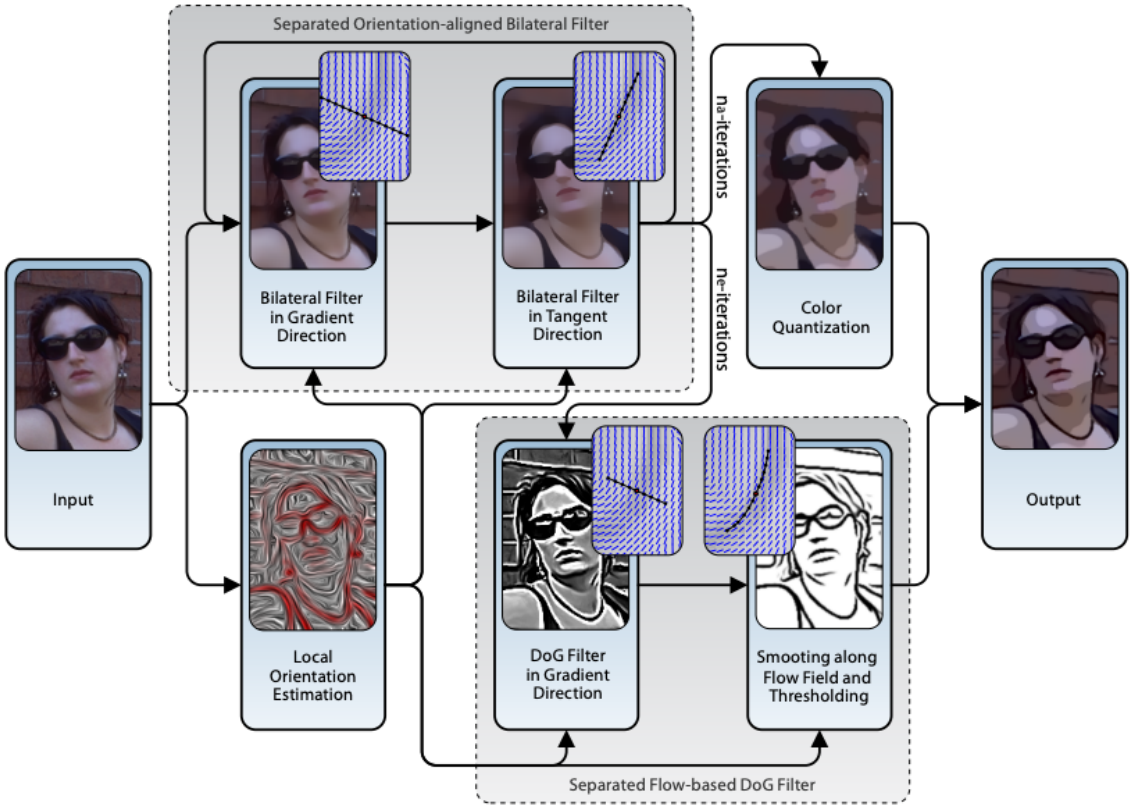


Figure 25: Framework visualization (Kyprianidis and Döllner 2008)

4.3 FDoG implementation

4.3.1 Local orientation estimation

The structure tensor is used to approximate the local orientation, the gradient and tangent directions. These are the directions in which the different filters will be applied. The orientations are based on the eigenvalues of the structure tensors. More precisely, a Principal Component Analysis is done here (Winnemöller et al. 2012). The PCA is applied to g_{ij} which is a two dimensional symmetric matrix, therefore two eigenvectors are obtained. The first one pointing in the direction of the greatest change, the gradient direction, the second one is perpendicular to it and called tangent direction. To compute the structure tensor, a Sobel filter in the x and y direction is applied. This gives the derivations in x and y direction. R, G and B are representing the three colour channels

from the image as displayed in equation 14*.

$$\frac{\partial f}{\partial x} = \begin{pmatrix} \frac{\partial R}{\partial x} & \frac{\partial G}{\partial x} & \frac{\partial B}{\partial x} \end{pmatrix}^t, \quad \frac{\partial f}{\partial y} = \begin{pmatrix} \frac{\partial R}{\partial y} & \frac{\partial G}{\partial y} & \frac{\partial B}{\partial y} \end{pmatrix}^t \tag{14}$$

Then the structure tensor is given by equation 15*,

$$J = \begin{pmatrix} \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \end{pmatrix} \quad g_{ij} = J^t J = \begin{pmatrix} E & F \\ F & G \end{pmatrix} \tag{15}$$

whereby g_{ij} is representing one pixel in the image.

After this step, Gaussian smoothing is applied with $sigma_a$. This is done by applying it to each E, F and G. E, F, G are values for each pixel. Looking for example at all the E from all the pixels, they represent a matrix with the dimension of the input image. To this matrix, the Gaussian filter can be applied. Kyprianidis and Döllner (2008) proposed to use an $9x9$ filter. The Gaussian filter has to be applied in order to get connected edges. Then, for each pixel in the image, two eigenvectors $v1$, the gradient direction and $v2$, the tangent, are computed. All the $v1$ or $v2$ introduce a discrete vector field.

In the following, the notation $v(c_0)$ means the vector v at position c_0. v is either $v1$ or $v2$ and c_0 a vector with x and y coordinates corresponding to a pixel in the image.

The gradient $v1$ and tangent $v2$ vector are computed as shown below. The computation of the eigenvector is given by equations 16* and 17*.

$$\lambda_{1,2} = \frac{E + G \pm \sqrt{(E-G)^2 + 4F^2}}{2} \quad D = \sqrt{(E-G)^2 + 4F^2} \tag{16}$$

$$v1 = \begin{pmatrix} F \\ \lambda_1 - E \end{pmatrix} \quad v2 = \begin{pmatrix} \lambda_2 - G \\ F \end{pmatrix} \tag{17}$$

Looking at equations 16* and 17*, it can be seen that the vectors are orthogonal to each other (equation 18).

$$\langle v1, v2 \rangle = \left\langle \begin{pmatrix} F \\ \frac{E+G+D}{2} - E \end{pmatrix}, \begin{pmatrix} \frac{E+G-D}{2} - E \\ F \end{pmatrix} \right\rangle = F(E + G - E - G) = 0 \tag{18}$$

The same can be achieved be looking at g_{ij} which is a symmetric matrix. By definition, the eigenvector of an symmetric matrix are orthogonal to each other (Horn and Johnson 2012). By looking at the doctoral thesis from Kyprianidis (2013), we can see that only $v1$ is calculated and $v2$ is derived from $v1$ in the following way (equation 19) (Kyprianidis 2013).

$$v2_x = v1_y \quad v2_y = -v1_x \tag{19}$$

Therefore, it is sufficient to only calculate $v1$ and save some computation time. With this, the computation of $v2$ is unnecessary. The eigenvectors are then normalised. This step was not described in Kyprianidis and Döllner (2008), but is needed in order to make the tangent smoothing step work. Another hint that this is right is given by Kang et al. (2008) which have a similar approach to normalise the direction vector they use. Applying the Sobel filter in x and y direction, and calculating the E, F and G values can lead to large positive or negative numbers due to the convolution from the Sobel filter. Without normalisation one would end up with pixels which are not in the image anymore in a later step of the implementation.

4.3.2 Bilateral filter

A bilateral filter is a non-linear filter to blur images while still leaving the edges intact. The colours and the difference in the colours are taken into account from the neighbouring pixels to calculate the new pixel value (Tomasi and Manduchi 1998).

The filter is first applied in gradient direction, then in tangent direction. The only difference is that we use v_1 for gradient or v_2 as the tangent direction. The filter can be applied multiple times, Kyprianidis and Döllner (2008) proposed to use three iterations. On iteration consists of one pass in the gradient and tangent direction. The more often it is applied, the more abstract the images becomes. For each pixel $c0 = (x, y)$ the following is computed (equation 20*), for the computation, the RGB input image is converted to the cielab colour space.

$$\delta(v) \begin{cases} (1, \frac{v_x}{v_y}) & |v_x| \geq |v_y| \\ (\frac{v_y}{v_x}, 1) & |v_x| < |v_y| \end{cases} \tag{20}$$

t_0 is the step direction. The idea is that we start at $c_0 = (x, y)$ and from there N steps are taken with and against the direction from $v1$ or $v2$. Each of the steps have the same length. t_0 the step direction at c_0 is given by equation 21*:

$$t_0 = \delta(v1(c_0) \text{ or } t_0 = \delta(v2(c_0) \tag{21}$$

The pixel values at c_0 are actually a vector with three entries corresponding to the three colour channels. The new value of a pixel at $c0$ is given by equation 22* and 23*:

$$c^{\pm} = c_0 \pm it_0 \tag{22}$$

$$f(c_0) = \frac{1}{k}\Bigg[f(c_0) + \sum_{i=1}^{N} G_{\sigma_d}(i\|t_0\|)\Big[G_{\sigma_r}(\|f(c_i^+) - f(c_0)\|)f(c_i^+) + G_{\sigma_r}(i\|f(c_i^-) - f(c_0)\|)f(c_i^-)\Big]\Bigg] \tag{23}$$

Figure 26 visualizes the aforementioned process.

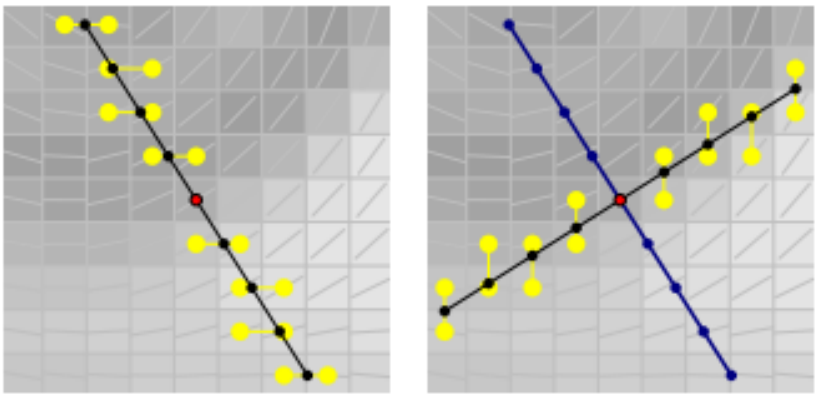
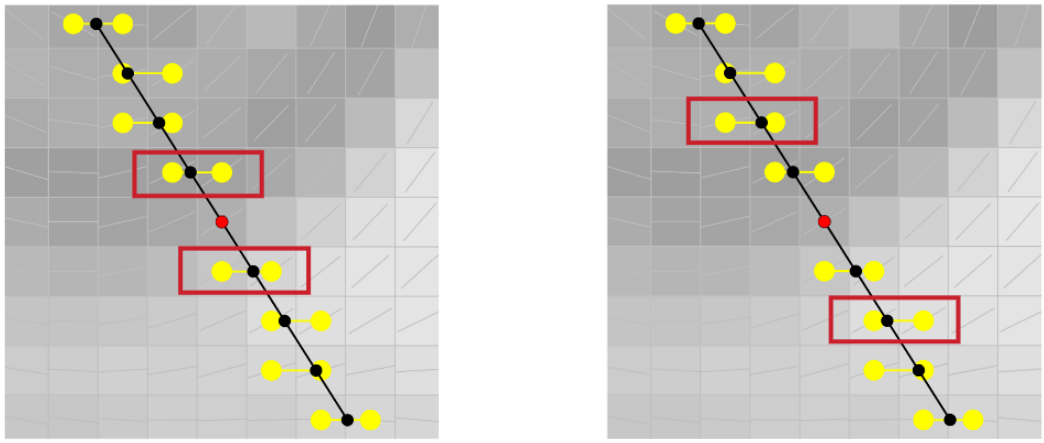


Figure 26: Explanation of Bilateral Filter (Kyprianidis and Döllner 2008)

On the left side of figure 26, the filter in gradient direction is shown and on the right side of figure 26, the filter in tangent direction is displayed. The procedure for the filter for both directions is the same, therefore the filter is explained from the first picture in gradient direction. In reference to equation 23, the red point in the middle corresponds to $f(c_0)$ at position x, y. Starting from the value f_0 at position c_0, we go one step in the gradient direction, this is t_i, i is the current step. The step is done with and against the direction of the gradient, as shown in figure 27 a) the first two black points next to the middle point in the red boxes. Then from each of the points, the interpolation (yellow points) is taken. The interpolation is needed since with adding the step the sample point will not be exactly in the center of one pixel. The interpolated points are $f(c^+)$ and $f(c^-)$. Then, the Gaussian is calculated of the interpolated points and normalised with k.



(a) Gradient step one (b) Gradient step two

Figure 27: Gradient steps (Kyprianidis and Döllner 2008)

This is repeated N times. Figure 27 b) shows the second step $i = 2$. K, the normalization factor, is given by equation 24.

$$k = 1 + \sum_{i=1}^{N} G_{\sigma_r}(i\|t_0\|) \left[G_{\sigma_d}(\|f(c_i^+) - f_0\|) + G_{\sigma_r}(\|f(c_i^-) - f_0\|)\right] \tag{24}$$

N is given by equation 25*.

$$N = 2\sigma_r / \|t_0\| \tag{25}$$

4.3.3 DoG in gradient direction

"To evaluate the one-dimensional DoG filter in gradient direction we use the same approach as described in the previous section" (Kyprianidis and Döllner 2008: p. 6). There were no exact equations or more explanations given on how to calculate the Difference of Gaussian. The DoG in gradient direction is applied to the luminance channel of the picture. Different to a normal DoG, the flow based version has no kernel. The kernel is replaced by steps along the gradient direction. What might be the equation considering the previous step is shown in equation 26. In equation 26 the proposed reformulation from Winnemöller et al. (2012), shown in equation 11, is already considered.

$$\begin{aligned} f(c_0) = \frac{p+1}{k_1} &\left[f(c_0) + \sum_{i=1}^{N} (G_{\sigma_e}(\|i * i\|)(f(c_i^+) + f(c_i^-)) \right] \\ -\frac{p}{k_2} &\left[(f(c_0) + \sum_{i=1}^{N} G_{\sigma_f}(\|i * i\|)(f(c_i^+) + f(c_i^-)) \right] \end{aligned} \tag{26}$$

$$k_1 = \sum_{i=1}^{N} (G_{\sigma_e}(\|i * i\|) * 2, \quad k_2 = \sum_{i=1}^{N} (G_{\sigma_f}(\|i * i\|) * 2 \tag{27}$$

Equation 26 and 27 were partly reverse-engineered from the code in the PhD thesis by Kyprianidis (2013). He uses a CUDA implementation, therefore equation 26 might not be completely right. In terms of how it is applied, it is the same as in the bilateral filter in gradient direction. Only the calculation of the equation for the image value $f(c0)$ is different.

4.3.4 Smoothing along tangent flow field

This step is used to average out the calculated filter response. A one dimensional Gaussian filter is applied along the flow field induced by the tangent directions. In contrast to the gradient direction, we do not sample across a straight line and the sampling points are not uniformly distributed. After each of the N steps, t_i, the direction, and l_i, the step length, is newly calculated leading to curved line as seen in figure 28 on the right side (Kyprianidis and Döllner 2008). Calculation steps for each pixel: c_0 the starting point

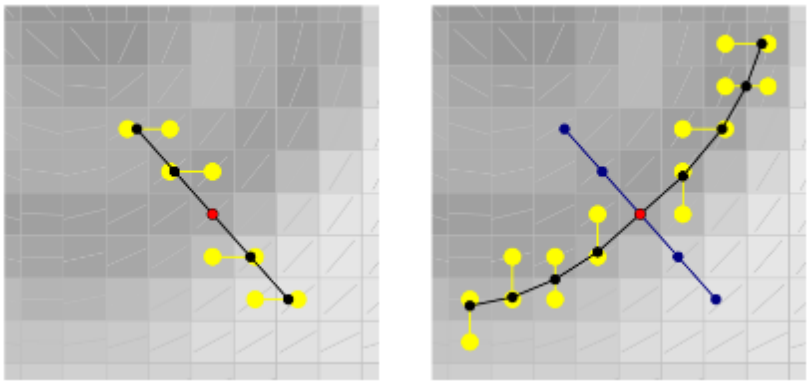


Figure 28: Left gradient direction, right tangent (Kyprianidis and Döllner 2008)

with t_0 the starting step length (equation 28*).

$$t_0^+ = v2(c0) \quad t_0^- = -t_0^+ \tag{28}$$

As a step direction the orientation of the tangent $v_2(c_{i-1})$ is used, the tangent does not define a particular direction therefore the direction with the smaller change is taken (equations 29* and 30*).

$$t_i = sgn\langle v_2(c_{i-1}^+), t_{i-1}^+ \rangle * v_2(c_{i-1}^+) \tag{29}$$

$$c_i = c_{i_i} + l_i t_i, \quad N = 2\sigma_m \tag{30}$$

The step length is chosen in such a way that the new sampling position is aligned to the center for the pixel in either x or y axis (Kyprianidis and Döllner 2008) (equation 31*).

$$l_i = \begin{cases} \left| \frac{c_{i-1,x} - \lfloor c_{i-1,x} \rfloor 1/2 sgn(t_{i,x})}{t_{i,x}} \right| & |t_{i,x}| \geq |t_{i,y}| \\ \left| \frac{c_{i-1,y} - \lfloor c_{i-1,y} \rfloor 1/2 sgn(t_{i,y})}{t_{i,y}} \right| & |t_{i,x}| < |t_{i,y}| \end{cases} \qquad L_i = \sum_{j=1}^{N} l_j \tag{31}$$

The result of smoothing along the flow curve is shown in equation 32*.

$$\begin{aligned} f(c_0) = \frac{1}{k} \Bigg[& \frac{\|t_0^+\|}{2} f(c0) + \sum_{i=1}^{N} \frac{\|t_i^+\| + \|t_{i-1}^+\|}{2} G_{\sigma_m}(L_i^+) f(c_i^+) + \\ & \frac{\|t_0^-\|}{2} f(c0) + \sum_{i=1}^{N} \frac{\|t_i^-\| + \|t_{i-1}^-\|}{2} G_{\sigma_m}(L_i^-) f(c_i^-) \Bigg] \end{aligned} \tag{32}$$

$c0$ being any pixel in the picture. t^+ or t^- do not matter as this is only the direction for the sampling position as can be see in figure 28 on the right side. Therefore, t_i is referred to instead of t_i^+ or t_i^-. Taking a closer look at t_i equation 29, t_i is $+1$ or -1 times $v_2(c_{i-1})$ at an interpolated point. This means ti is an interpolation of normalized vectors, therefore $\|t_i\| = 1$ and the equation can be rewritten as:

$$\frac{1}{k} \left[f(c0) + \sum_{i=1}^{N} G_{\sigma_m}(L_i^+) f(c_i^+) + \sum_{i=1}^{N} G_{\sigma_m}(L_i^-) f(c_i^-) \right] \tag{33}$$

$$k = \sum_{i=1}^{N} \left(G_{\sigma_e}(\|L_i^+ * L_i^+\|) + G_{\sigma_e}(\|L_i^- * L_i^-\|) \right) \tag{34}$$

4.3.5 Interpolation

In the previous section often c_0 or $f(c_0)$ is used. The position c_0 can be anywhere in the pixel plan and is mostly not an integer-valued vector. This means, $c_{0_x} = x$ and $c_{0_y} = y$ do not necessarily have an integer value. But only the pixel values for integer x and y are known. To solve this problem, Kyprianidis and Döllner (2008) proposed to use linear interpolation. This function checks as well if the values are still in the image as it handles the edge cases. To implement linear interpolation, the following function was used and applied whenever $f(c_0)$ was used in an equation. x and y are the non-discrete position and $img(x, y)$ the pixel values at position x, y.

$$\begin{aligned} xInt = \lfloor x \rfloor, \quad yInt = \lfloor y \rfloor, \quad x0 = xInt, \quad x1 = xInt + 1, \quad y0 = yInt, \\ y1 = yInt + 1 \quad a = x - xInt, \quad c = y - yInt \\ interpolation = (img(x0, y0) * (1 - a) + img(x1, y0) * a) * (1 - c) \\ + img(x0, y1) * (1 - a) + img(x1, y1) * a) * (1 - c) \end{aligned} \tag{35}$$

4.3.6 Threshold and connecting areas

First, a Gaussian Blur with a kernel size of $3x3$ and $sigma_z$ is applied. Then the threshold equation 13 is used, this determines which pixel is turned black. Afterwards, all the pixel on the edges of the picture are turned white. This has the effect of connecting areas which are on the edge of the image. The next step is to connect or to get rid of the white areas, which are surrounded by black pixels as seen in figure 23. This is done by extracting the different segments of the images. If the area of the segment is white and below the parameter *area*, the whole segment is made black. If the segment is larger, all the edges pixel (the white pixels which are next to a black pixel see figure 24 c)) are extracted within the area and outside the area. Then, the nearest white edge pixel is searched for each of the edge pixel. This is done by searching the shortest distance from any edge pixel in the area and any edge pixel outside the area. Then, the shortest distance within the area and outside area is connected with a straight line. The further away the areas to connect the more recognisable is the connection. But the results with

(a) nearest neighbour in different ares (b) straight connection (c) highest energy seam connection

Figure 29: Improved connections, Rüttimann 2022

this approach are just not satisfying enough. To improve the connections, the idea is to use the seams from section 2.8 to connect two areas. Like before, the nearest neighbour was searched. With the orientation of the vector connecting the two points, figure 29 picture a), the image was turned into the direction of the connecting vector. This had to be done to use the top-to-bottom approach from the seam carving algorithm. To speed up the process only the surroundings from the two areas are considered and not the whole picture. Then, the highest energy seam going through the pixel was taken. For this, equation 10 was adjusted to extract the maximum instead of the minimum. The pixel through which the seam goes is the one nearest neighbour pixel from the area which is further down the y axis. Afterwards, it is checked whether the connection was successful, if not, the straight connection is taken. The highest energy seam is taken since it follows the pixel with the biggest change and that path can resemble an edge since it has a lot of

discontinuities with a high energy. This leads to more natural connections as can be seen in figure 29 b), straight connection, and c) highest energy seam connection. The seam connection was only checked when the length of the connection is higher than five pixel. There are further improvements thinkable: Instead of searching the nearest neighbour, the most extreme, the corner points from an area are extracted. Then, from each of this points, the highest energy seam is followed in all four directions, top, bottom, left, right. Then the shortest connection to another ares is selected.

4.4 Problems

Winnemöller et al. (2012) pointed to Kyprianidis and Döllner (2008) and Kang et al. (2008) for the details of the FDog implementation, both describe similar approaches but do not use the exactly same steps. After an recommendation by my supervisor, the decision to follow the approach by Kang et al. (2008) is taken. Looking deeper into Kyprianidis and Döllner (2008), the following problems appeared.

K is described as the normalization factor, and no further information was given. Hence, I had to figure out myself what k should be. At some point a normalization of the structure tensor had to be done in order to get usable numbers for the tangent step, since there the length of the vector is used. But a normalization has never been mentioned by Kyprianidis and Döllner (2008). Also at a first glance, $f(c_0)$ being the same as f_0 was not obvious. There are even more smaller inconsistencies which made it hard to know what exactly they tried to describe. Later, they described the DoG in gradient direction as "To evaluate the one-dimensional DoG filter in gradient direction we use the same approach as described in the previous section"(Kyprianidis and Döllner 2008: p. 6).

Finding all these missing pieces and combining them to a working algorithm is extremely frustrating and time-consuming. Alone getting an idea how the flow DoG in gradient direction could probably work took multiple days. Figuring out what the equations or factors exactly are is one thing. What made it really challenging is that there are no values to check if the results from a calculation are right. For example, after the computation of the structure tensor it was not possible to compare if the values are accurate. At the end, one can see whether the result is good or not. But figuring out where the actual bug could be if the result is not good is almost impossible. As already mentioned, Winnemöller et al. (2012) provided the exact parameters for their test images. But without knowing what exact equations they used, it is not possible to use the exact parameters they used. However, they can act as a guideline.

4.5 Getting to know the algorithm

The algorithm is split into three parts. The bilateral filter, extracting the edges and finally connection the areas. The ideas is to first calculate the bilateral filter and saving the

result. From there, the parameters for extracting the edges are altered until as pleasant result is found. The parameters for this step have way the higher impact then the bilateral filter in terms of how the result will look like. After extracting the edges, connecting the areas can be done. The reason for this is that connecting the areas can get extremely expensive. Therefore, not too many holes should be present when starting this process.

The implementation was one step but with ten parameters the algorithm is extremely hard to control. Since for each pass the algorithm takes several minutes, checking the result of different parameters is time-consuming. To get more ore less good results, the parameters have to be fine-tuned for each picture. The best approach to do this is to give the algorithm a wide spread of parameters and then select the best result. From there, one adjusts the parameters slightly and compares the result. This procedure has to be done multiple times. In table 6 are the exact parameters from Winnemöller et al. (2012) displayed. What can bee seen is the wide variety and how specific the parameters are. With this example it can be seen that it is hard to find the perfect parameters. The resolution of the image is also important. If the resolution is higher, details are kept better but a higher resolution means longer computation times.

4.6 Performance

The first implementation was done such that it is as understandable as possible and without any means of optimisation. This allows to find bugs faster while this obviously leads to longer run times. This is comprehensible since for-loop after for-loop is used to calculate all the values. And this is done in each step. After the implementation was run, it was time to apply some optimisations. The most effective one being vectorisation and introducing a maximal resolution of $500x500$ pixel. With vectorisation the idea is instead of looking at each pixel individually to calculate all the values for the image at once. This was first applied for the local orientation estimation. With this, the run time for the local orientation estimation was cut down to a fraction to what it was before. Unfortunately, it was (for me) not possible to apply vectorisation to other steps. And all the these steps combined used up the majority of the computation time. Especially the bilateral filter which is applied multiple times takes a certain time to run. Another attempt to speed up the process was replacing library calls, like calculating the norm with “numpy”, with in-place calculation. This reduced the run time by around 20 percent. To speed up the calculation time even more, an improvement step would be to write the algorithm in C++.

4.6.1 Big O notation

In order to make the big O notation easier, we assume that the image is a square and has n pixel in length and width.

Step	Run time
Bilateral filter	$O(3 * n * n * 2\sigma_r/\|\|t_0\|\|) = O(n^2)$
DoG in gradient direction	$O(n * n * 2\sigma_e/\|\|t_0\|\|) = O(n^2)$
Smoothing along tangent flow	$O(n * n * 2\sigma_m) = O(n^2)$
Threshold	$O(n * n)$
XDoG (all steps combined)	$O(4 * n^2) = O(n^2)$

Table 5: Run time approximation for the XDoG implementation

As we can see in table 5, the XDoG is dependent on the resolution. The run time for the connection of the holes is more complicated. Assuming the picture has $l << n * n$ holes, each hole has $i << n * n$ edge pixel and $k << n * n$ are all the edge pixels not in the hole. Then, each i is compared to each k for each hole l. This leads to a maximal run time of $\Theta(n^6)$. Imperial measurements have shown that connecting the holes is less costly than the bilateral filter and xDoG computation as can be seen in table 7. The faster run time works only if there are not too many areas which need connecting. Connecting the areas has the potential of extreme high run times. In table 7, the connection time for figure 37 b) is high. Th reason for that is that 46 areas need connections. The parameter *area* was intentionally left low to show the problematic run time.

5 Results

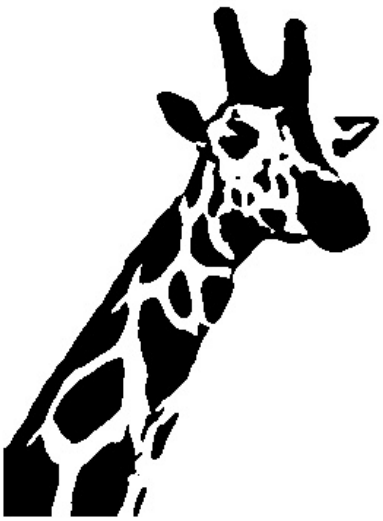


Figure 30: Example stencil one

Figure 31: Example stencil two

While the choice of images for testing the algorithm may seem arbitrary, there are reasons for this diverse selection. The giraffe was chosen since it is my favourite animal. The image of former German chancellor Angela Merkel was used because my supervisor showed me an abstracted example of her at the start of this thesis. The two actors, Bradley Cooper and Emma Stone, have been used in the canny edge detector project by Sahir (2019*a*). As it was the first project looked into, the same images are used to be able to make a comparison between the obtained results and to track what progress has been made in this thesis.

Looking at the results in figures 30, 31, 32 and 33, the pictures are two-tone, abstract and all the white areas are connected. Hence it can be concluded that the implementation works. The parameters can be found in table 8. Taking now into consideration the guidelines from section 1.4, the original picture in appendix B, figure 38, 39, 40 and 41 have to be considered. In all these examples, the original image is still recognisable and has larger black areas. This means that the main feature preservation works and the images are more abstracted than the original ones. In figure 30, 31 and 32 the different parts from the image are distinguishable from each other. In figure 33 an artefact above the right shoulder can be detected, however this can be blamed on the input image. More problematic are the hole connections in figure 31 as they look artificial. Creating physical stencils with this would be challenging due to the thin connections. In all four pictures

Figure 32: Example stencil three

Figure 33: Example stencil four

there are larger black and white areas which leads to a good contrast.

Seeing the results on "paper" is one thing, but how good are they transferable to a physical medium? Figure 34 shows figure 30 and 32 sprayed onto canvas. With this, the last point from section 1.4 "details have to be recognisable after the stencil is sprayed" can be checked and it can be concluded that the details are still recognisable after the stencil is sprayed.

When comparing the results to the canny edge detector presented in section 2.1 and displayed in figure 7, the results differ a lot. The results in figure 7 do not have coherent, artistic edges and the edges are mostly to thin. The original image is still recognisable and the different parts of the image are separated. The level of detail is too high and not abstract enough. The black and white ares not big enough to create a good contrast. Therefore, it can be concluded that the implementation this thesis uses does a significantly better job of producing stencils compared to the canny edge detector by Sahir (2019*a*).

Figure 35 a) and b) have been created by the online stencil creator from Chris (2022). Comparing the giraffe in figure 30 and figure 35 a) the results are similar. But the level of detail from Chris (2022) a) is a bit too high. The edges from the the result in figure 30 are smoother and the black areas are more coherent. The level of detail from 35 b) is not at the right level. The original is still recognisable but the face misses details and the right arm is not visible enough. On the same level are the connections in both figure 31 and figure 35 b). In some points they look good and sometimes artificial. Generally, it

(a) Example stencil one sprayed on canvas

(b) Example stencil three sprayed on canvas

Figure 34: Physical stencils

can be said that the online stencil creator by Chris (2022) works well but the algorithm developed in this thesis works better.

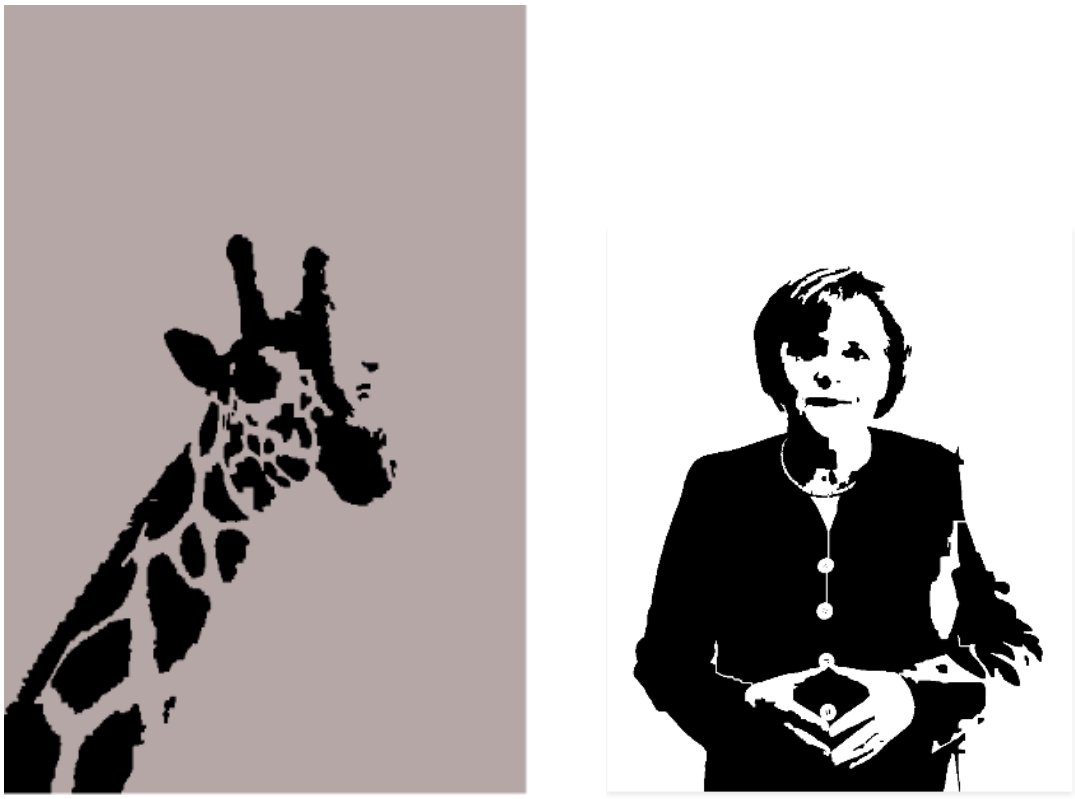
5.1 Direct comparison with used sources

5.1.1 Bilateral Filter

The first step leading to a result which is comparable is the Bilateral Filter from Kyprianidis and Döllner (2008).

Figure 36: Bilateral Filter comparison between (Kyprianidis and Döllner 2008) and my implementation with same parameters

In figure 36, the left picture is the original image. The result from Kyprianidis and Döllner (2008) is displayed in the middle and on the right side the implementation of this



(a) Example stencil one (b) Example stencil

Figure 35: Stencil created with stencil creator from Chris (2022)

thesis is displayed with the same parameters. The results are not the same but go into a similar direction. What can be seen is that the result from Kyprianidis and Döllner (2008) is more blurred and abstract. Why the results differ is hard to tell, it might be the resolution, slightly different implementation or a small bug. But the result goes clearly in the right direction. In figure 45 on top is the original result from Kyprianidis and Döllner (2008) and on the bottom the implementation results with the number of iteration shown. One can see that with higher iterations the two picture are getting more similar. But there is always a small difference. In figure 44 are more results for the Bilateral Filter from the implementation displayed.

5.1.2 XDoG

The second comparable result is the XDoG from Winnemöller et al. (2012). For the only two-tone picture for which the original was in the paper used by Winnemöller et al. (2012), they did not provide the parameters. The other examples cannot by used since they use the greyscale threshold function. Comparing figure 37, a) the original algorithm from 20 and b) the implementation shown in this thesis, there is a clear difference. The result from Winnemöller et al. (2012) has smoother and more coherent edges. Another big difference is the right side of the face, where in the implementation of thesis shows larger black ares. Generally speaking the result from Winnemöller et al. (2012) looks better but

(a) Original (b) My implementation

Figure 37: Comparison XDoG (Winnemöller et al. 2012)

the reproduced results goes in the right direction. The difference might come from not the exact parameters, another resolution, different implementation, or simply just a bug.

6 Conclusion

In most fields of IT, or at least in the areas where I have worked so far, the work is done digitally and often the result cannot be directly seen or touched. At the end, one cannot tell why all the work had to be done and where all the time was spent. In this thesis it is refreshingly different.

This research aimed to automatically create abstracted images, so-called stencils, and evaluate their looking. Based on a thorough analysis of present algorithms used for image abstraction as well as its components, and a general overview on graffiti and stencils particularly, it can be concluded that the extended algorithm is able to automatically produce stencils from input images. In addition, the stencils are physically touchable and usable as they have been printed and sprayed on canvas. Connecting back to Banksy, the produced stencils are comparable to this type of art judging by the level of abstraction and level of appearance. Hence, this thesis shows how to develop and adapt a data visualisation task from a technical perspective, while at the same time it demonstrates how the output can be used in the urban sphere as art.

Big improvements have been made throughout this thesis, beginning with the canny edge detector and then the final results. The key part of the algorithm is the XDoG combined with the FDoG. The combination of these two algorithms leads to an artistic abstract image. Extending this combination with connecting the white areas then creates a stencil. Although is was not always easy, as especially implementing the papers with the missing information was time-consuming and frustrating, the present work satisfies with its results and with some fiddling, the right parameters can be found. Certainly, there are some downfalls such as the run time which could be better so as a higher resolution could be used when the algorithm is faster. In addition, with higher resolution, the algorithm could be better adjusted for the different pictures. Furthermore, the connection of the white area needs some improvement. The results would look better if the connection would always follow the natural lines present in the original image. Overall it can be said that the automated creation of stencils works.

Based on these elaborations, some recommendations can be made to further develop and improve on the issue of image abstraction. In order to speed up the algorithm, the algorithm could be implemented in C++. Another method would be to use parallelisation to compute different pixels at the same time. Relating back to the problem of the white areas, the connection of the white areas could be improved by adjusting the maximum seam approach. Not only the nearest neighbours could be considered for connecting, every pixel at the edge of an white area could be checked. This could lead to more natural looking connection. Another interesting avenue would be to find the parameters automatically. However, the implementation of this certainly poses a certain problem.

7 Bibliography

Akgün, D. and Erdoğmuş, P. (2015), 'Gpu accelerated training of image convolution filter weights using genetic algorithms', *Applied Soft Computing* **30**, 585–594.

alphabetcityblog (2008), 'Heavily tagged subway car of the nyc subway', Online. [Accessed August 5, 2022].
URL: *https://4.bp.blogspot.com/_GeUoqt1lLXY/STxVGtrbLlI /AAAAAAAAAElY/j2YwwwWK1Bs/s1600h/happyholidaygraff.jpg*

Avidan, S. and Shamir, A. (2007), Seam carving for content-aware image resizing, *in* 'ACM SIGGRAPH 2007 papers', pp. 10–es.

Calonius, E. (1973), 'Heavily tagged subway car of the nyc subway', Online. [Accessed August 3, 2022].
URL: *https://en.wikipedia.org/wiki/File:Heavily_tagged_subway_car_in_NY.jpg*

Campbell, G. (2003), 'The oxford dictionary of the renaissance', Online. [Accessed August 18, 2022].
URL: *https://www.oxfordreference.com/view/10.1093/acref/9780198601753.001.0001/ acref-9780198601753-e-1701*

Canny, J. (1986), 'A computational approach to edge detection', *IEEE Transactions on pattern analysis and machine intelligence* (6), 679–698.

Chris (2022), 'Super stencil maker', Online. [Accessed October 13, 2022].
URL: *https://superstencilmaker.com/*

Dharampal, V. M. (2015), 'Methods of image edge detection: A review', *Journal of Electrical & Electronic Systems* **4**(2), 5.

Dictionaries, O. L. (2022), 'Oxford learners dictionaries', Online. [Accessed August 18, 2022].
URL: *https://www.oxfordlearnersdictionaries.com/definition/english/abstraction*

Dimitri Ehrlich, G. E. (2006), 'Graffiti in its own words', Online. [Accessed April 29, 2022].
URL: *https://nymag.com/guides/summer/17406/*

Ding, L. and Goshtasby, A. (2001), 'On the canny edge detector', *Pattern recognition* **34**(3), 721–725.

D'Cruz, J., Cushing, H. and Gay-Reese, J. (2010), *Exit Through the Gift Shop*, United Kingdom: Universal Studios.

Ellsworth-Jones, W. (2012), *Banksy: The man behind the wall*, Aurum.

Frood, E. and Ragazzoli, C. (2013), 'Writing on the wall: two graffiti projects in luxor'.

Gonzalez, W. (2008), *Digital Image Processing*, Prentice-Hall, Inc.

Haddad, R. A., Akansu, A. N. et al. (1991), 'A class of fast gaussian binomial filters for speech and image processing', *IEEE Transactions on Signal Processing* **39**(3), 723–727.

Helfer, D. (2006), 'Graffiti ancien représentant un bateau. musée du graffiti ancien à marsilly (charente-maritime)', Online. [Accessed July 3, 2022].
URL: *https://en.wikipedia.org/wiki/File:Graffiti_2.JPG*

Horn, R. A. and Johnson, C. R. (2012), *Matrix analysis*, Cambridge university press.

Jain, A. K. (1989), *Fundamentals of digital image processing*, Prentice-Hall, Inc.

Kang, H., Lee, S. and Chui, C. K. (2008), 'Flow-based image abstraction', *IEEE transactions on visualization and computer graphics* **15**(1), 62–76.

Kanopoulos, N., Vasanthavada, N. and Baker, R. L. (1988), 'Design of an image edge detection filter using the sobel operator', *IEEE Journal of solid-state circuits* **23**(2), 358–367.

Kyprianidis, J. E. (2013), Structure adaptive stylization of images and video, PhD thesis, Universität Potsdam.

Kyprianidis, J. E. and Döllner, J. (2008), Image abstraction by structure adaptive filtering., *in* 'TPCG', pp. 51–58.

Levinger, J. (2005), 'Love is in the air, flower thrower, 2005', Online. [Accessed August 5, 2022].
URL: *https://publicdelivery.org/banksy-flower-thrower/*

Lindeberg, T. (1994), 'Scale-space theory: A basic tool for analyzing structures at different scales', *Journal of applied statistics* **21**(1-2), 225–270.

Linnartz, A. (2010), 'Angela merkel, kanzlerin der bundesrepublik deutschland, vorsitzende der cdu', Online. [Accessed September 2, 2022].
URL: *https://de.wikipedia.org/wiki/Datei Angela_Merkel_Juli_2010_-_3zu4.jpg*

Lohmann, P. (2020), 'Historical graffiti: The state of the art', *Journal of Early Modern Studies* **9**, 37–57.

Ludwig, J. (2013), 'Image convolution', *Portland State University* p. 3.

Marr, D. and Hildreth, E. (1980), 'Theory of edge detection', *Proceedings of the Royal Society of London. Series B. Biological Sciences* **207**(1167), 187–217.

Misra, S., Li, H. and He, J. (2019), *Machine learning for subsurface characterization*, Gulf Professional Publishing.

Mould, D. and Grant, K. (2008), Stylized black and white images from photographs, *in* 'Proceedings of the 6th international symposium on Non-photorealistic animation and rendering', pp. 49–58.

Müller, M. (2022), 'About me', Online. [Accessed September 15, 2022].
URL: *https://michaelmueller.me*

Niemann, C. (n.a.), 'The abstract-o-meter', Online. [Accessed September 10, 2022].
URL: *https://www.christophniemann.com/detail/my-life-in-illustration/*

Pallás-Areny, R. and Webster, J. G. (1999), *Analog signal processing*, John Wiley & Sons.

Papari, G., Petkov, N. and Campisi, P. (2007), 'Artistic edge and corner enhancing smoothing', *IEEE Transactions on Image Processing* **16**(10), 2449–2462.

Philipps, A. (2015), 'Defining visual street art: In contrast to political stencils', *Visual anthropology* **28**(1), 51–66.

Put, Z. (2020), 'Graffiti, 15 podgórska street, kazimierz, krakow, poland', Online. [Accessed July 10, 2022].
URL: *https://en.wikipedia.org/wiki/File:Graffiti_15_Podg%C3%B3rska_street,_Kazimierz,_Krakow,_Poland.jpg*

Rémih (2009), 'Graffiti, kom ombo temple, egypt', Online. [Accessed July 5, 2022].
URL: *https://en.wikipedia.org/wiki/File:Graffiti_Kom_Ombo.JPG*

Sahir, S. (2019*a*), 'Canny edge detection step by step in python — computer vision', Online. [Accessed April 29, 2022].
URL: *https://medium.com/towards-data-science/canny-edge-detection-step-by-step-in-python-computer-vision-b49c3a2d8123*

Sahir, S. (2019*b*), 'Face 2', Online. [Accessed July 2, 2022].
URL: *https://github.com/FienSoP/canny_edge_detector/blob/master/faces_imgs/2.jpg*

Sahir, S. (2019*c*), 'Face 4', Online. [Accessed August 15, 2022].
URL: *https://github.com/FienSoP/canny_edge_detector/blob/master/faces_imgs/4.jpg*

Shapiro, L. G., Stockman, G. C. et al. (2001), *Computer vision*, Vol. 3, Prentice Hall New Jersey.

Shenoi, B. A. (2005), *Introduction to digital signal processing and filter design*, John Wiley & Sons.

Shrivakshan, G. and Chandrasekar, C. (2012), 'A comparison of various edge detection techniques used in image processing', *International Journal of Computer Science Issues (IJCSI)* **9**(5), 269.

Sobel, I. and Feldman, G. (2015), 'An isotropic 3x3 image gradient operator'.

Timbers, T., Campbell, T. and Lee, M. (2022), *Data science: A first introduction*, CRC Press.

Tomasi, C. and Manduchi, R. (1998), Bilateral filtering for gray and color images, *in* 'Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)', IEEE, pp. 839–846.

Truman, E. J. (2010), 'The (in) visible artist: stencil graffiti, activist art, and the value of visual public space', *Queen's Journal of Visual & Material Culture* **3**, 1–15.

Vaughan (N.D.), 'Types of graffiti', Online. [Accessed August 15, 2022].
URL: *https://www.vaughan.ca/services/residential/bylaw_enforcement/graffiti/Pages/TypesofGraffiti.aspx*

Wang, S., Li, W., Wang, Y., Jiang, Y., Jiang, S. and Zhao, R. (2012), 'An improved difference of gaussian filter in face recognition.', *J. Multim.* **7**(6), 429–433.

Winnemöller, H., Kyprianidis, J. E. and Olsen, S. C. (2012), 'Xdog: An extended difference-of-gaussians compendium including advanced image stylization', *Computers & Graphics* **36**(6), 740–753.

Wu, Y.-T., Yeh, J.-S., Wu, F.-C. and Chuang, Y.-Y. (2017), 'Tangent-based binary image abstraction', *Journal of Imaging* **3**(2), 16.

www.pexels.com (2019), 'Porträtfoto einer giraffe lokalisiert auf blauem hintergrund', Online. [Accessed September 5, 2022].
URL: *https://www.pexels.com/de-de/foto/portratfotoeinergiraffelokalisiertaufblauem-hintergrund2541407/*

Appendices

A Tables

σ_c	σ_e	σ_m	p	ϵ	σ_a
2.28	1.4	4.4	21.7	79.5	1.0
2.45	1.0	6.0	18.0	82.2	NA
3.76	1.4	2.20	15.7	78.3	2.4
5.84	0.8	3.2	120	72.6	.75
0.1	2.0	20	40	100	7.2
0.1	6.8	20	70	80.0	0.6
4.16	1.4	12	22	88.0	4.0
4.16	1.4	12	22	79.0	4.0

Table 6: Parameters from Winnemöller et al. (2012)

figure	σ_a	σ_d	σ_r	σ_e	σ_f	p	σ_m	σ_z	ϵ	$area$
30	6	3	4.25	3	4.8	10	5	1	70	0.00035
31	4	3	4.25	2.1	3.36	11.5	3	2	105	0.0025
32	4	3	4.25	2.1	3.35	9.1	8	4	70	0.0025
33	4	3	4.25	1.6	2.56	14	10	3	110	0.001
37	4	3	4.25	1.9	3.5	19	8	2	19	0.00025

Table 8: Parameters

figure	Resolution	Bilateral filter	XDoG (extracting edges)	Connecting areas	combined
30	332x500	140 s	145 s	1 s	286 s = 4 min 46 s
31	411x500	177 s	102 s	60 s	339 s = 5 min 39 s
32	297x345	91 s	120 s	1 s	212 s = 3 min 32 s
33	318x499	134 s	226 s	11 s	371 s = 6 min 11 s
37	434x500	201 s	264 s	529 s	996 s = 16 min 36 s

Table 7: Run time, 1.4 GHz Quad-Core Intel Core i5, 16 GB Ram

B Original Images

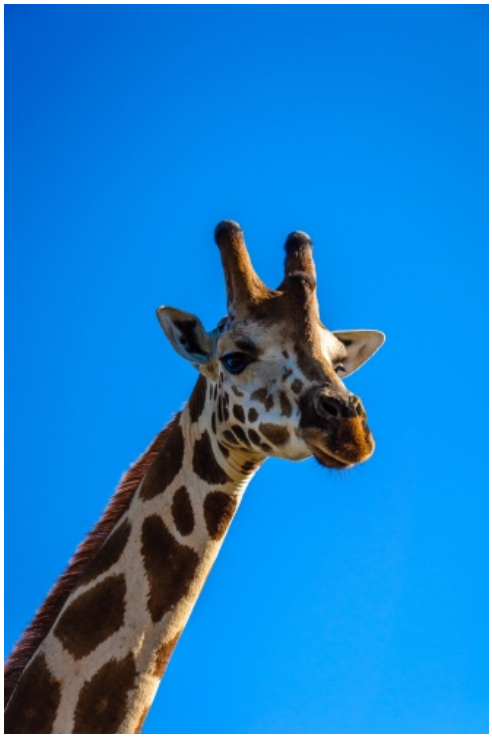


Figure 38: Example stencil one original (www.pexels.com 2019)

Figure 39: Example stencil two original (Linnartz 2010)

Figure 40: Example stencil three original (Sahir 2019*b*)

Figure 41: Example stencil four original (Sahir 2019*c*)

C More Graffiti

Figure 42: Graffiti example one, picture taken by Rüttimann 2022

Figure 43: Graffiti example two, picture taken by Rüttimann 2022

D Bilateral Filter

Figure 44: Bilateral filter example. 3 passes with $\sigma_d = 3$ and $\sigma_r = 7$

Figure 45: Multiple iteration bilateral filter