User Guided Image Abstraction for Vectorization

Daglar Coban[1] , **Amal Parakatt**[1] , **Elmar Eisemann**[1]
[1]TU Delft

Abstract

Image abstraction plays an important role in converting pixelated or low quality images into vector images. This paper presents a systematic user interactive process to abstract a generic photograph and to produce suitable content for vectorization. The goal of abstraction is to simplify the content of the photograph while retaining important information, creating a stylized and artistic image as a result. The system offers the user to create an importance map, by selecting regions with high importance or salience. Then, two different methods, bilateral filtering with k-means clustering and mean-shift segmentation, can be used as abstraction methods. Various parameters for different parts of the image and for each method, are exposed to the user to extend the spectrum of variety for the end result, and to tune the final image as desired. Canny's edge detection process is used to create a line drawing from the original image. The user can adjust thresholds, delete edges, and change the thickness and smoothness of edges to alter the line drawing. The resulting images are greatly diverse in terms of the number of colors and the amount of edges they can potentially have. In the end, this method proposes a new interactive way to create images suitable for vectorization, by combining and altering previous abstraction techniques.

1 Introduction

Even with the advances in image technology, low resolution images with lacking visual fidelity still exist to this day. This becomes more evident when these images are shown on high definition displays, like the ones present in almost all screens used today. Artists may want to convert photographs to vector graphics for a number of reasons, such as to manipulate and stylize the image, and to eliminate low resolution artifacts.

There have been attempts to develop a system to convert these low resolution images to vector images that can be scaled as desired. Previous solutions have been developed in this context, most notably Depixelizing Pixel Art[18], where pixel art images were converted into vector images. However, these solutions are, for the most part, not interactive but automated [9][33][16][17]. Our solution keeps the user in the loop and enables the user to steer the abstraction process. Another aspect that has not been addressed by previous work is abstracting real photography for vectorization, whether it is a landscape nature image or a portrait of a human.

Through this work, the aim is to find and develop a way to abstract a generic photograph image, to simplify its content with the goal of producing suitable content for vectorization with user interaction. The main challenge is that photos have an immensely varying, and complex set of variables and features that may appear as important, such as the eyes of a human or textures on an object.

An important part of creating vector images is to obtain an abstraction. An image may consist of a foreground and a background, or it may be divided into parts according to light and color distribution, shapes, and objects. One can abstract the entire image into a simpler form and reduce its complexity. This is necessary in order to obtain clear boundaries for the lines and the colors that will make up the vectorized output image. By creating an abstract image, it is expected that further processes can easily identify the boundaries in the vectorized output. The challenging aspect is to segment different parts of the image.

The importance of a region can be based on perception, colors and shapes, or even on its relation to other parts of the image[13]. The areas that seem important in an image are typically salient. When looking at an image, humans focus on different areas based on saliency, which "is the state or quality by which it [an item] stands out from its neighbors."[27]. The goal is to find these parts and combine them accordingly to create an abstract representation of the image.

This paper is structured as follows. We present the related work, followed by the methodology. Then, the process for generating the importance map, creating the line drawing using edge detection, and finally the two different abstraction algorithms, bilateral filtering and mean-shift segmentation, are discussed thoroughly. Afterwards, the experimental design to test the results, the results itself, and lastly the conclusion and discussion are given.

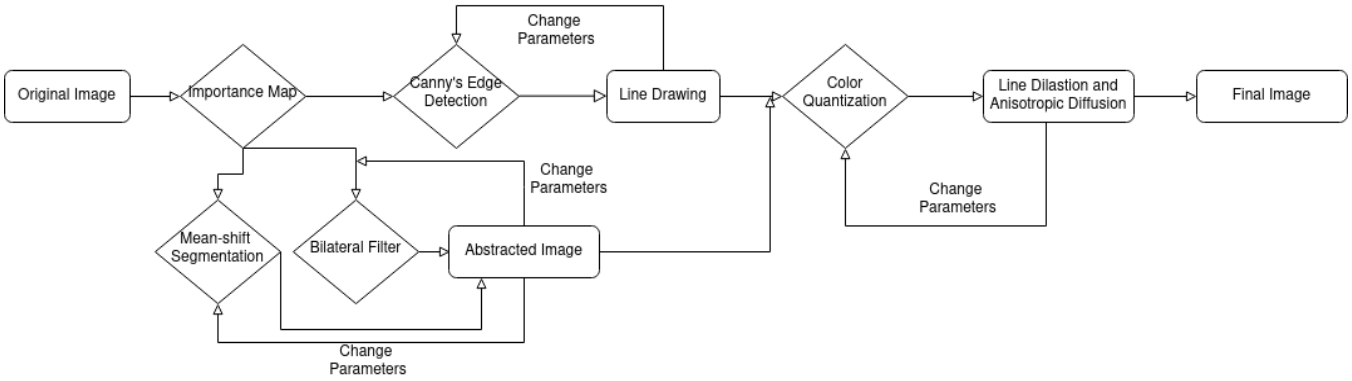

Figure 1: Simplified workflow overview for the systematic procedure to obtain the final image.

2 Background and Related Work

Image abstraction is a subject of non-photorealistic rendering (NPR), which itself is a subdomain of computer graphics. It aims to render images less life-like and more cartoon-like, illustrated and stylized [1]. Different goals can be achieved with NPR. A 3-dimensional model can be rendered to produce a simpler technical diagram, or a 2-dimensional image can be artistically rendered for a particular representation. In this paper, the latter goal is discussed, with abstracting photographs prior to vectorization.

Previous work has investigated the abstraction of images. One of the first works, which also incorporated user interaction and worked on photographs, is by DeCarlo and Santella [9]. They used an eye tracker to detect saliency and eye fixation data from the user to guide the abstraction process, although all other aspects of the actual abstraction are carried out automatically. Their approach built a hierarchical structure by repeatedly down-sampling the image. Then, for each sampled image, a segmentation occurs. DeCarlo and Santella used a slightly modified version of the mean-shift segmentation, from the work of Comaniciu and Meer [8]. Mean-shift segmentation is an iterative approach and has a number of different parameters that can change the results, such as spatial radius, thresholds for color distinction, and changing sizes for the shifted region space. In the end, this segmented result is combined with a line drawing that was produced using Canny's edge detector with a fixed kernel size. The main difference between this paper and their work is that DeCarlo and Santella used an eye tracker to detect the important information in the photograph, which guides the abstraction process, and thus the user has no other control on the final image. Contrasting their work, in this paper, the user is able to alter the entire edge detection process to produce a line drawing, as well as creating an importance map and selecting the main abstraction procedure: mean-shift segmentation or bilateral filtering.

Real-Time Video Abstraction uses a process that abstracts the images "by modifying the contrast of visually important features, namely luminance and color opponency" [33]. Rapid spatial differences usually represent boundaries. These areas have a higher chance of being salient. It analyzes the image for low and high contrast regions, and lowers the contrast further in the low regions while doing the opposite in the high contrast regions. The reduction of contrast is executed using a smoothing filter. The addition of contrast is done by the difference-of-Gaussians (DoG) edges, an alternative to Canny's process which is simpler, both computationally and result wise. This particular solution can be parallelized, thereby increasing its performance. The abstraction carried out by the bilateral filter is applied in an iterative manner. Then the DoG edges are added, and a process called "luminance quantization" is performed to create a more stylistic result. The final process reduces the color space in the image. The main differences in our approach are that Canny's edge detection is used instead of DoG edges, an additional color quantization step is applied, and that the user can guide the abstraction process using the importance map.

Historically, the abstraction process has been kept away from the user, not allowing for the user to guide the process. Although the user cannot significantly change the outcome, the work of Kang and Lee has provided the user to "'paint' a particular area covering important features to protect" [17]. The work uses a completely different approach to image abstraction, called the mean curvature flow with shock filtering, in order to simplify shapes. The iterative and incremental method allows the user to control the level of abstraction, as the iteration can be stopped at any point. Additionally, their approach lets the user draw small spots on the image to let some parts be protected from over-abstraction. However, compared to the importance map depicted in this paper, their method is only effective for finer regions, like the eyes of a person, and not for larger salient components.

Another solution for abstraction, that builds on top of the one previously mentioned by DeCarlo and Santella [9], is the Flow-Based Image Abstraction[16]. This solution is also a fully automated one. It produces a stylized abstraction of a photograph. The difference in this work is that it uses an additional structure, called a vector field, to guide the abstraction. The vector field describes the flow of salient features in the image and is generated automatically [16]. Also, for this process to work, the image must be converted into a specific format called a height field of pixel intensities. After this vector field has been created, the software passes the image through a line detection and extraction phase, followed by a

region smoothing phase. The line extraction process creates a line drawing, while the region smoothing process removes all the insignificant height discontinuities, thus flattening and smoothing the image. In this work, a modified DoG filter and bilateral field is used, which takes the vector field into account. The downside of this framework is the vector field, which may not work well with complex images, and creating it adds overhead. The user is also left out of the process and cannot alter the abstraction.

Research in pixel art, also gives insight into the abstraction of images. Pixelated Image Abstraction [11] accomplishes the opposite of the main goal presented in this paper, where a non-pixelated image is abstracted into a pixelated image. Image segmentation is carried out using superpixels, and the color palette is reduced via Mass Constrained Deterministic Annealing. Although this is somewhat the inverse process, the abstraction phase used in image segmentation and color palette reduction are relevant to the current research question.

One additional matter relevant to the framework depicted here is the process of vectorizing images. Vectorization is the reverse procedure of rasterization, and some characteristics of the original image make it easier to vectorize [18]. Images that were originally made in vector graphics, like pixel art, cartoons and similar, have these underlying characteristics that make it more suitable for vectorization. These images have clearly defined borders around their components, have a reduced color space, and have large continuous color regions [25]. On the other hand, photographs have a huge variety in their characteristics, and usually have the opposite features. There exists software that can automatically vectorize images that have similar features to originally vector images, however they either do not support or do not work well with photographs [28][14][3]. Hence, a successful abstraction would ideally convert photographs into images that have these characteristics, namely: a reduced color space, clearly defined borders, and large continuous color regions.

3 Methodology and Contributions

The aim is to interactively create stylized and artistic images that are suitable for vectorization, by simplifying and abstracting the image, all the while retaining visually important details. The methodology is targeted towards artists, wanting to vectorize their photographs in a stylistic way. Different artists may have different goals in mind for every photograph. Due to this, every step of the method has been conceived to allow for extensive user interaction.

Initially, the user selects the photograph and is prompted to select the important, salient parts, of the image. This is done by drawing over the image. Afterwards, the user is guided through the edge detection and abstraction (bilateral filter and mean-shift segmentation) procedures separately. Finally, there is a color quantization phase that the user can alter, and the detected edges in the beginning are added back to produce the end result. The simplified flow of the entire method can be seen in Figure 1. In order to obtain the result, the concrete implementation has been written in C++14, and the OpenCV (version 4.5.1) library [15] has been used to implement the algorithms.

3.1 Importance map

A photograph might be a portrait, with only one human subject and redundant information in the background. Or it might be taken on a street with one object as the main focus of attention, and many objects and components around the subject, some of which are important and some irrelevant. On the opposite end of the spectrum, the photograph could be a landscape photograph, where almost all information is relevant. Salient components in each type of photograph will be different. Using eye tracking data on each image that will be abstracted could be cumbersome and requires an eye tracker. Furthermore, deploying a solution that uses machine learning, or alternative data driven approaches, is complex and difficult. These methods do not allow the user to stay in the loop, which is achieved in our work.

The approach taken here is to let the user create an importance map, outlining the areas and components that appear visually salient for that particular user. Liao, Shen, and Eisemann [19] used depth annotations to create depth-based effects in images that did not have depth information (non-stereo images). In their work, the user scribbles over the image. Then a depth map is created, and the user can control various aspects of the creation of the depth map to fine tune the result. Similarly, instead of depth, an importance map is created in this framework.

In our work, an interactive and iterative process is devised to segment an object or a region from the image. GrabCut [26] uses an energy minimization function, improving the work of Boykov and Jolly [4], that relates to an acceptable segmentation in the image. This is done iteratively. Additionally, GrabCut allows the user to draw annotations on the image to include or remove parts of the image in the next iteration of the segmentation procedure. The user interacts with the system to improve the segmentation, and does not need to manually outline the image, which can be tedious.

In this implementation, the user is prompted to draw a bounding box over the visually important component. Then, the iterative GrabCut algorithm is used to compute the segmentation. If the user is satisfied with the result, they can move on to the next part of the framework. If not, the user can annotate and draw small lines on the partial segmentation in two different ways to produce a better one: addition and removal annotations. Following this, in the next iteration, the new regions are added or removed from the segmentation. Finally, when the user is satisfied with the segmentation, the importance map is generated. Additionally, in photographs that do not have an important region, but rather the entire photograph itself is important like a landscape photograph, the entire image can be selected.

The importance map is simply a binary matrix with the same dimensions of the image, where a pixel is either important or not depending on the segmentation created previously. This process can be seen in Figure 2.

In the following stages of the workflow, this importance map is used to guide the amount of abstraction done to the final image. The region that the user has selected as salient has different, independent parameters that changes the abstraction only in that section. This is done to yield more flexibility, and less salient parts can be abstracted more with the given

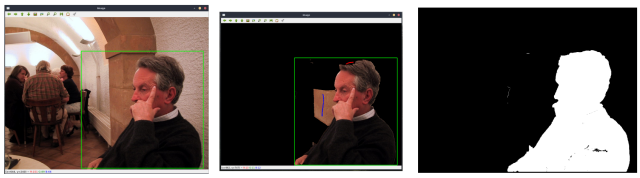

Figure 2: The user selects the visually important component. Annotations add and remove regions to the importance map. The white region is the resulting importance map. Image from [9].

parameters. More salient and important sections, guided by the importance map, can have completely different values to abstract that particular section.

Compared to methods using eye-tracking information and other more complicated data-driven approaches using machine learning, it is a faster and simpler procedure to obtain important features in the image, and keep these features in the abstracted output. In this way, the importance map is used to allow the user to create the final result that is in their control.

3.2 Line drawing

The goal of abstraction is to reduce the complexity of the image. Three components that add complexity to the image are the colors, the shapes and objects, and the lines that enclose these different colors and objects. The latter process, the image abstraction portion, aims to reduce the colors and objects. Inevitably, some visually important salient lines are removed as a consequence. Although it may be beneficial to smooth out some edges, other edges that are important should remain in the image. Without these lines, the visually important object or region does not appear as important as before.

Before any alteration is done to the image, a line drawing is created. The goal is to only have the lines that enclose uniform regions. Edges that appear in the background or the less important regions should not be present in this line drawing. The user interacts with this procedure, since automated solutions may keep irrelevant edges or remove relevant ones.

To create the line drawing, an edge detection algorithm can be used. There are various different edge detection algorithms that work well in different types of settings [30], however their main working principle is to determine higher contrast areas that are likely to be edges in the image. The edge detector used in this paper is the Canny edge detector [6]. Canny's edge detector works as follows: Initially, the image is converted to grayscale, then a Gaussian filter is used to remove noise. The Sobel operator, which approximates the intensity gradient, is used both vertically and horizontally to obtain the value of first derivative and the gradient direction for each pixel in both orientations [5]. Then, each pixel is checked if it is a local maximum in the direction of the gradient, the pixels that are not local maxima are subdued. For pixels that are local maxima, given minimum and maximum thresholds, edges with the gradient more than the maximum are definite edges. Edges that are below the minimum are removed. Those who are in between the two thresholds are determined to be edges if they are connected to definite edges.

Previous work has used different edge detectors, such as the difference-of-Gaussians (DoG) [33][16]. The DoG method produces suitable results with illustrations and other less complicated images, and has the benefit of being computationally easier than Canny[33]. However, as pointed out by Kang et al. [16], it does suffer when trying to portray the direction of the edges, which may be important to correctly portray perspective in the photograph. To obtain more complicated edge information, and to let the user decide whether to keep these edges or not, Canny's edge detector is preferred. Additionally, Canny's edge detector has the advantage of detecting real edges with certainty [33], which is advantageous for vectorizing photographs. With advancements in computation power, Canny's process takes much less time than what was possible previously.

In the proposed system, the kernel size, minimum threshold that an area is detected as an edge, and the ratio to determine the upper threshold can be independently controlled by the user for the important region and the background. Decreasing the minimum threshold results in more connected edges being detected. With the increase in kernel size, more pixels are combined. This makes the edges, the gradient map, more blurry. Furthermore, changing the ratio of the lower to upper threshold allows the user to have finer control in the edges detected, as more or less connected edges are detected. The process is outlined in Figure 3.

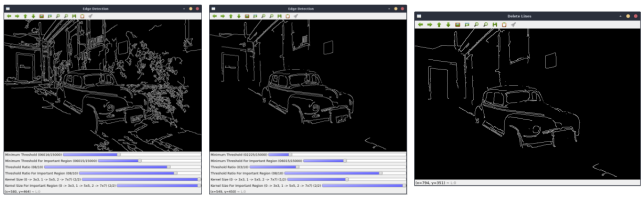

Figure 3: **Left:** Edge detection without importance map applied. **Center:** Importance map applied to image, and the user can change the thresholds to change the number of edges detected separately. **Right:** The user can paint over incorrect edges to remove them. Image from [10].

Being able to independently control the parameters for the important region and the rest of the image is preferential for an output suitable for vectorization. By doing this, less important regions that have been abstracted more, have fewer edges in the final image. This means that a potential vectorization procedure, will not detect incorrect boundaries.

In the next step of the system, the artist can remove erroneously detected edges for better results, as seen in Figure 3. Once again this is done to produce a more suitable result for vectorization, as the edge detection method is not perfect and removing incorrect lines creates a better result. To add more stylistic features, the size and smoothness of the image can also be altered at the very end, after the line drawing is added on top of the abstracted image.

3.3 Abstraction Using Bilateral Filtering and K-Means Clustering

As discussed before, after creating the line drawing, the user has the option to select either the mean-shift segmentation, or the bilateral filtering to accomplish the abstraction phase.

The bilateral filter is a smoothing filter that aims to reduce noise while preserving edges in an image [29]. The way it works is by going over each pixel value and determining a weight according to the intensity of the neighbors of the pixel being analyzed. The weight attached to each pixel is computed through changes in color intensity and distance.

When applied in a non-iterative way, the filter has two parameters: the spatial kernel and the color kernel [29]. The spatial kernel controls the spatial extent of the filter. As this parameter increases, pixels that are farther away influence each other and larger regions are smoothed. The color parameter determines the color space, as it increases, larger regions of the same color are produced. One problem that occurs with the bilateral filter is that a big difference in intensity value can cause cartoon like staircase effects.

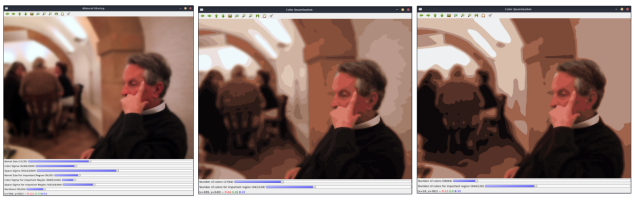

Figure 4: The user can change various parameters to obtain different levels of abstraction from the bilateral filter. A less quantized k-means color quantization is applied on the image in the center. The image on the right has a greater quantization. Image from [9].

The bilateral filter can be changed to become iterative. In this way, each iteration makes the image turn more into a constant signal. The resulting image also resembles a more cartoon like image, and has been used for abstraction previously [33].

The iterative approach in the work of Winnemöller et al. [33] uses a pseudo luminance quantization process after the iterative bilateral filtering abstraction. This is done to produce a more abstracted image, since the bilateral filtering may not produce the best result. They use a gradient-based method to change the sharpness in the image. In contrast to this technique, the work of Shopovska and Ivanovski [21] uses k-means clustering to accomplish color quantization in the image, abstracting the color space further.

In the method used in this paper, however, techniques from both of these work have been combined. The bilateral filter used in this paper is an iterative one, where the result of the previous iteration is fed through to the input of the next. The user has control over the kernel size, spatial kernel, and color kernel. Furthermore, all three of these parameters are independently controllable for the important region. The iterative bilateral filter is applied twice to the entire image, once with the parameters defined for the important region and once with the parameters for the rest of the image. Then, by checking for each pixel whether it belongs to the important region defined by the importance map or not, the correct pixel values are transferred to the result. Although applying the filter twice may seem excessive, due to the nature of how the filter uses a kernel, calculating a pixel considering its neighbors results in more smooth transitioning boundaries, and the overhead is minimal.

The color quantization is done by k-means clustering. K-means is a clustering algorithm that partitions the data into k different parts, where each data point is part of one of these clusters that has the closest possible mean [12]. With images, there are three features, red, green, and blue values for each pixel used for the clustering. In this approach, the termination criteria is either when the desired accuracy is reached or the maximum number of iterations is reached, whichever occurs first. When testing, it was found that a maximum of 10 iterations with an epsilon of 1.0 resulted in decent results without a performance penalty, although this can be easily changed. The user can control the number of clusters to find in the image, for the important and non-important region separately. The number of clusters represents the number of different colors, with fewer colors more is abstracted. Then, each cluster is filled with the average mean color of the region to generate the color quantized effect.

Another consideration with k-means clustering is the initial centers for the clusters. Selecting random initial centroids may result in suboptimal clustering. In this framework, the kmeans++ [2] method is used for more optimal clustering. Moreover, three attempts are made and the clustering with the best compactness is chosen. All of these considerations result in an abstracted image that is color quantized, with distinct boundaries and a reduced color space with large continuous regions.

The abstraction using bilateral filtering with color quantization can be seen in Figure 4.

3.4 Abstraction Using Mean-Shift Segmentation

Another option the user has available, other than the bilateral filtering, is to use mean-shift segmentation [8]. Mean-shift is a different type of clustering algorithm that uses a hill climbing approach, and computes density estimation in order to cluster the data. It is an iterative approach with a kernel, spatial and color bandwidths. [7]

In this approach, density values correspond to the x-y values combined with the red, green, and blue pixel intensities. First, a window is centered around each point according to the spatial bandwidth. The density, and the color distance from the center point to every point in the window, is calculated. For the pixels with the density lying in the color bandwidth, a shift vector is calculated. This shift value is the mean of the points. Afterwards, the shifting occurs, and the process is repeated until convergence or a maximum number of iterations. After this filtering phase, the segmentation takes place. For this, each region with the same density maxima is filled with the same color.

The user can change the maximum iterations for the clustering to change the level of abstraction. Additionally, the user can change the color and spatial bandwidths to alter the search window, leading to different abstractions. As this process is guided by the importance map, separate variables are present for the important region and the background. Similar to what we have done with the bilateral filtering, this method is also computed twice for both regions with different parameters. Then, each pixel is checked if they lie inside the importance map for the final result.

An additional difference between the approach in this paper and the work of DeCarlo and Santella [9] is that there is an optional color quantization process that the user can choose, the same algorithm that is used after the bilateral filtering. Even though the result from mean-shift segmentation is abstracted fairly well, this is done to reduce the color space further, to obtain an abstraction that is even more suitable for vectorization. By opting for parameters that result in more quantized outputs, the user can choose to have a color space that is reduced to a greater extent, while still preserving boundary information.

The output after the mean-shift segmentation is presented in Figure 5.

Figure 5: **Left:** Mean-shift with spatial and color bandwidths of 8 and 14 for the background, and 6 and 12 for the important region. **Center:** Mean-shift with spatial and color bandwidths of 12 and 18 for the background, and 8 and 14 for the important region. **Right:** K-means color quantization applied on the center image. Image from [9].

3.5 Resulting Image

The abstraction process, whether using the bilateral filtering followed by color quantization or using mean-shift segmentation with color quantization, results in lines that are barely visible. To add salient edges back, the line drawing created in the beginning stages is combined with the abstracted image.

While combining, the user has the option to stylize the lines. The user has control over three different aspects: dilation, blurring, and pixel cut-off point. By altering these, the user has greater flexibility over different types of images, for example a thinner line drawing might suit landscape photographs better. Dilation is a morphological transformation computed by calculating the maximum pixel intensity coinciding with a kernel. The pixel that is at the center of the kernel is replaced with this intensity. This causes the line to thicken. The second aspect blurring is carried out using a normalized box filter, the lines can be slightly blurred to blend the line with the image. The user has control over the blur radius. The result is a grayscale image with the background being white (pixel intensity of 255), and the lines being black and gray (intensities of 0 to 254). User interaction is also available when combining. This is accomplished by the user selecting a pixel cut off point, any pixel value in the line drawing below this intensity is regarded as a line and the corresponding pixel in the final image is replaced with this pixel in the line drawing. To overcome artifacts and make the lines blend in even further with the abstracted image, the line drawing is smoothed.

Slightly smoothing the combined abstraction and line drawing at the very end, results in an image that looks more natural with fewer artifacts, and lines that blend in. To achieve this, anisotropic diffusion is coupled with the line drawing, smoothing the lines while retaining as much information as possible. Anisotropic diffusion, a technique developed by Perona and Malik, intends to decrease noise while preserving lines [24]. This preservation of lines is why this method is preferred over a Gaussian filter or a standard box-filter that does not preserve the lines very well. Although this method adds complexity and increases the computation needed, it creates a more natural effect in the image since the artifacts and noise is removed without damaging the lines and goes well with Canny's edge detection. The user is also able to control the amount of smoothing done with the anisotropic diffusion, by changing the sensitivity to edges, the number of iterations, and the difference in each iteration, in order to determine the result with fewer artifacts and more natural looking edges. This is seen in Figure 6.

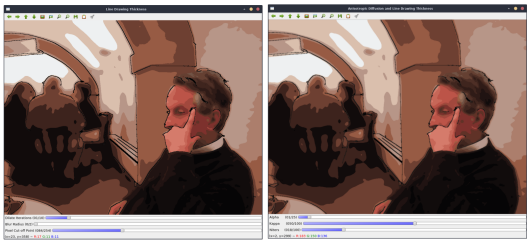

Figure 6: Line thickness is changed on the left image. Anisotropic diffusion is applied with user changeable parameters onto the line drawing on the right image. Image from [9].

4 Results

In this section, the results of the framework are presented using different images with varying potential use cases.

An interesting use case is abstracting and creating vectorized images from landscape photographs, potentially for an animated background in a movie or a game. Landscape images tend to not have a single important region, but rather the entire scene is deemed important. The framework is used by selecting the entire image as important, then following the procedure for both the bilateral filtering approach, and the mean-shift segmentation. The result is presented in Figure 7. The same line drawing is created for both methods, and both use color quantization. Either method resulted in a reduced color space, with visible boundaries separating the colors and regions. Additionally, with the combination of the line drawing, the important edges were more detectable, which is ad-

Figure 7: **Left:** Original image. **Center:** Final output with the bilateral filtering with color quantization method. **Right:** Final output using the mean-shift segmentation with color quantization method. Image from [34].

Figure 8: **Left:** Original image. **Center:** Final output with the bilateral filtering with color quantization method. **Right:** Final output using the mean-shift segmentation with color quantization method. Image from [9].

Figure 9: **Left:** Vectorized original image. **Center:** Vectorized image using the bilateral filtering method. **Right:** Vectorized image using the mean-shift method. Image from [9].

vantageous for vectorization. The bilateral filtering output had a slightly less reduced color palette, with smaller regions of uniform color. This output may be chosen for a higher definition vectorization, retaining more of the salient colors, although the vectorization might be somewhat more expensive. On the other hand, the mean-shift with quantization output is more abstracted, and had a flatter color space with larger areas of uniform color. This output is easier to vectorize and may be desired for its simplicity.

Another use case is a portrait photograph, where a human subject is of importance. The original image and the abstracted outputs are seen in Figure 8. The goal is to abstract the image while retaining some information about the subject. The background, with the table and three people, should be abstracted more than the subject, and the final image should have clear boundaries with large uniform colors. For both the bilateral filter, and the mean-shift method, the importance map is built by selecting the man in front. Then, using the same parameters for the edge detection, a line drawing is created. For both methods, k-means color quantization is applied after the abstraction phase. Finally, the user decides the thickness and diffusion of the combined image. The mean-shift output once again had larger regions of the same color, but in the process had lost detail in the subject's face. The boundaries were wavier in the bilateral output, with more information retained. Both methods have successfully abstracted the image, decreased the color space, and both outputs only have relevant edges that are distinct.

The aim of the framework is to achieve abstracted images suitable for vectorization. To check if the output images are indeed more appropriate, AutoTrace [3], a software to automatically convert bitmap images into vectorized images, is used. The initial photograph is inputted into the software, and is compared to the abstracted outputs of the bilateral filtering and mean-shift methods, both with color quantization and the line drawing added. The resulting outputs are seen in Figure 9, in which the white lines are where the software decided to trace the vectors. The original image is unusable as a vector image. However, in both cases, our framework has produced a usable vectorization, and in the end, the user can decide their desired output based on their needs.

Figure 10: **Left:** Original image. **Center:** Final output with the bilateral filtering with color quantization method. **Right:** Final output using the mean-shift segmentation with color quantization method. Image from [10].

5 Discussion

In previous work in non-realistic rendering, two main methods have been used, mean-shift segmentation and the bilateral filter. Mean-shift segmentation is a clustering algorithm, meanwhile the bilateral filter is a noise reduction and smoothing filter. Both are used iteratively, which makes them well suited for abstraction, to abstract more, further iterations can be done. However, both of these methods have their advantages and disadvantages. The key observation was that previous systems are limited to specific types of images or work better in some images than others [9][33][17][18]. Thus, providing a framework that supports human input at every stage and combining different approaches from both the non-photorealistic rendering field and other computer vision techniques, was expected to improve on this, yielding more choice to the user. This allows the user to choose the resulting image according to their needs.

DeCarlo, Santella [9], and others [31][32], have used image segmentation to abstract. Image segmentation is well suited for abstraction, since it simultaneously retains essential details while simplifying the complexity in the image [17]. Other work that used bilateral filtering [33] achieved abstraction without altering regions, but rather smoothed the image while maintaining boundary details. The two contrasting approaches work well in different images with different goals in mind. Mean-shift segmentation does not work well with textures [9], however adding areas with texture to the importance map in the beginning and using the bilateral filter, achieves a better result which can be seen in Figure 11. Either option is possible and is up to the user. This is the main reason to include both abstraction techniques, alongside additional methods like k-means color quantization and anisotropic diffusion of edges, to offer more options for the user.

The resulting images are suitable for vectorization with decreased color space, regions with the same color, and noticeable boundaries produced by the line drawing. Particularly, significant details are still present in the salient region of the image without the use of an eye tracker. Likewise, as shown in Figure 9, automated vectorization solutions perform significantly better with the output of our framework.

Each part in the system has its own limitations. Canny's edge detection process produces edges that are real in the im-

Figure 11: The pattern of the shirt in the background is added to the importance map. The result is that the pattern is retained and noticeable. Image from [9].

age. However, for larger maximum thresholds, it may produce disconnected lines [6], negatively affecting future vectorization which relies on closed boundaries. Besides this, many unimportant edges may be detected even with changing parameters, which forces the user to manually delete edges or to choose the important region more carefully using the importance map.

The limitations with the importance map are tied to the possible limitations in the GrabCut algorithm, as well as only having one level of importance. Although the user can add as many regions as desired to the importance map, the choice was made to include only one level of importance for performance reasons and to not overwhelm the user with many choices.

Alongside the issues stated before, using the bilateral filter in an iterative way also presents some limitations. Depending on the rest of the parameters, additional iterations may cause false edges to appear. The color quantization, with k-means clustering, is sensitive to outliers in pixel intensities and regions with local minima. Additionally, the same parameters are used for all clusters and the process is somewhat expensive without optimizations.

On the other hand, mean-shift segmentation is robust to outliers, but, images with high dimensionality causes sharp changes that results in artifacts in the output. Furthermore, in mean-shift segmentation, the user cannot directly influence the number of clusters, which is a disadvantage when a specific color space is needed for vectorization. That is why, k-means color quantization is applied afterwards in our approach. Some patterns and textures may not be visible after abstraction. To abstract these regions less, they need to be added to the importance map. Even then, the complicated patterns may appear disproportionate. Additionally, even with optimizations, the process is computationally expensive.

Our system succeeds at giving the user control throughout the entire process. The artist can achieve immense variety in results by changing parameters at each step, as the changes in parameters interact with the methods in later stages, very different outputs can be achieved as seen in Figures 4, 5, and 6. Although the advantages and drawbacks of each individual step are known, one inherent limitation of this system is that not every image can be tested to see how well it performs. An example with less than optimal results is seen in Figure 10. The original image has high-dimensionality and sudden changes in color. Both methods abstract the image reducing the color space and resulting in clear boundaries, however the pink color in the background is lost with bilateral filtering and the image looks much darker with mean-shift segmentation. By reducing the abstraction and color quantization, better results are achieved with the disadvantage of producing results that are less suitable for vectorization. Nevertheless, with the guidance of the importance map and a vast array of mutable parameters, an output close to what the user had in mind should be attainable.

Performance of the framework depends on the image size and the parameters being used, which hinges on the intentions of the user. The actual implementation of the various algorithms used, and their optimizations also affect performance. However, Canny's edge detection and the bilateral filter found in standard implementations in popular libraries prove to be quite fast. The mean-shift segmentation and k-means color quantization are the bottleneck in performance.

Various future additions and alterations may improve our framework. Scale-space theory may be applied to improve the line drawing with fewer artifacts, as well as the overall abstraction by analyzing the images in different scales [23] [20] [22]. Additionally, the importance map may be broken down into multiple layers, allowing for different degrees of abstraction being applied at different levels, compared to only having a single level of importance. For more variety in color quantization, other clustering algorithms can be considered.

6 Conclusion

The framework presented in this paper exhibits a new way to abstract images, specifically photographs, with the goal of obtaining results that are suitable for vectorization. It is geared towards artists, who want to eventually convert their photographs into vector graphics for various reasons. Although this approach does not directly vectorize images, since this needs several more steps, it does output one of the crucial steps in the vectorization pipeline: the abstracted image.

The system abstracts images with two different methods: mean-shift segmentation and bilateral filtering. To reduce the color-space further, a color quantization procedure is used which clusters regions and fills them with the same color. A line drawing is created with Canny's edge detection method, and can be further altered by the user, changing the thickness and smoothness, hence stylizing the edges. The resulting line drawing is added to the final image. The entire framework is guided by an importance map, created by the user at the beginning, in which salient and important features can be abstracted less compared to the rest of the image.

The resulting image has a significantly reduced color space, with clear boundaries and edges between components, and continuous colored areas ready for vectorization. Furthermore, salient features are kept and abstracted less, which is achieved using basic mouse and keyboard guidance from the user without the need of special eye-tracking equipment or machine learning methods. Thus, letting the artist realize their ideas without additional overhead. In the end, a photograph is abstracted while keeping important details, and the resulting image is suitable for vectorization.

7 Responsible Research

The system presented in this paper has very few ethical considerations due to its nature. Since this is only a framework, no user data is stored or used in the process. Furthermore, the framework does not make use of any underlying data in the input image, and thus has no bias. As with most image software, the framework does not have a licensing check, and any copyrighted material that is used incorrectly is the user's responsibility.

This paper is intended to describe a detailed framework that abstracts photographs for vectorization. Due to this, the reproducibility of the results obtained is cardinal. As stated in the methodology, the implementation is written in C++14, and the computer vision library, OpenCV version 4.5.1, is used for the implementations of the algorithms. All the algorithms used have been discovered at least more than a decade ago. Over the years, for many of them, optimized and altered version were implemented. OpenCV was used in this paper, albeit all computer vision libraries have these algorithms as standard implementations. Moreover, the algorithms are widely known and relatively simple enough that they can be implemented even without these additional libraries.

The framework has changed and adapted some of these algorithms. When doing so, which variation is used, why its used, and how it has been implemented to reproduce the results has been specified. For example, instead of the standard k-means clustering algorithm, a variation called kmeans++ has been used and the parameters for this have been specified. Since kmeans++ is also widely available, the entire implementation has not been discussed. Another modification has been used in the mean-shift segmentation algorithm, where each data point is regarded as a five-tuple of x-y coordinates, and red, green, blue pixel values. The motivation behind this, and the implementation, is depicted.

The system is intended for users to create the results based on their needs and their visions. As such, there are many parameters that the user can change with any number of goals in mind. Recreating an output that is identical to another, only by changing the parameters until it looks the same, is not feasible. The final outputs used in the results are based on the sufficiently abstracted notion of the author. Nonetheless, the parameters used to reach the intermediate results have been shown in the images presented and can be reproduced.

Additionally, the source code for the implementation is made public in order to allow anyone to look at the framework in action, and use it for their own needs. It can be found at https://github.com/daglarcoban/image-abstraction.

References

[1] *SIGGRAPH '90: Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*, New York, NY, USA, 1990. Association for Computing Machinery.

[2] David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, page 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics.

[3] AutoTrace. http://autotrace.sourceforge.net, 2002. Version: 0.31.1.

[4] Y.y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, 2001.

[5] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[6] John Canny. A computational approach to edge detection. *Readings in Computer Vision*, page 184–203, 1987.

[7] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[8] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[9] Doug Decarlo and Anthony Santella. Stylization and abstraction of photographs. *ACM Transactions on Graphics*, 21(3):769–776, 2002.

[10] Elvin. Colors of Cuba. https://www.flickr.com/photos/25228175@N08/2786758360/. Flickr.

[11] Timothy Gerstner, Doug Decarlo, Marc Alexa, Adam Finkelstein, Yotam Gingold, and Andrew Nealen. Pixelated image abstraction with integrated user constraints. *Computers & Graphics*, 37(5):333–347, 2013.

[12] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[13] Donald D Hoffman and Manish Singh. Salience of visual parts. *Cognition*, 63(1):29–78, 1997.

[14] Inkspace. https://inkscape.org/, 2021. Version: 1.1.

[15] Itseez. Open source computer vision library. https://github.com/itseez/opencv, 2015.

[16] H. Kang, Seungyong Lee, and C.k. Chui. Flow-based image abstraction. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):62–76, 2009.

[17] Henry Kang and Seungyong Lee. Shape-simplifying image abstraction. *Computer Graphics Forum*, 27(7):1773–1780, 2008.

[18] Johannes Kopf and Dani Lischinski. Depixelizing pixel art. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2011)*, 30(4):99:1 – 99:8, 2011.

[19] Jingtang Liao, Shuheng Shen, and Elmar Eisemann. Depth annotations: Designing depth of a single image for depth-based effects. *Computers & Graphics*, 71:180–188, 2018.

[20] Tony Lindeberg. Scale-space theory: A framework for handling image structures at multiple scales. volume 8, pages 27–38, 09 1996.

[21] I. Lj., Shopovska, and Z. Ivanovski. Image segmentation using bilateral filter. 2014.

[22] Helmut Mayer. Abstraction and scale-space events in image understanding. In *In: International Archives of Photogrammetry and Remote Sensing, Vol. XXXI, Part 3*, pages 523–528, 1996.

[23] Alexandrina Orzan, Adrien Bousseau, Pascal Barla, and Joëlle Thollot. Structure-preserving manipulation of photographs. *Proceedings of the 5th international symposium on Non-photorealistic animation and rendering - NPAR 07*, 2007.

[24] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.

[25] A. Quint. Scalable vector graphics. *IEEE MultiMedia*, 10(3):99–102, 2003.

[26] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23:309–314, 08 2004.

[27] Salience (neuroscience). https://en.wikipedia.org/wiki/Salience_(neuroscience), journal=Wikipedia, Feb 2021.

[28] Peter Salinger. Potrace. http://potrace.sourceforge.net/, 2019. Version: 1.16.

[29] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, 1998.

[30] Emanuele Trucco and Alessandro Verri. *Introductory techniques for 3-D computer vision.* 01 1998.

[31] Jue Wang, Yingqing Xu, Heung-Yeung Shum, and Michael F. Cohen. Video tooning. *ACM Trans. Graph.*, 23(3):574–583, August 2004.

[32] Fang Wen, Qing Luan, Lin Liang, Ying-Qing Xu, and Heung-Yeung Shum. Color sketch generation. NPAR '06, page 47–54, New York, NY, USA, 2006. Association for Computing Machinery.

[33] Holger Winnemöller, Sven C. Olsen, and Bruce Gooch. Real-time video abstraction. *ACM SIGGRAPH 2006 Papers on - SIGGRAPH 06*, 2006.

[34] Yabandeh, Maysam. https://pixnio.com/media/nature-range-high-land-mountains-landscape. Pixnio.