GIVING CARTOON EFFECT TO COLORFUL IMAGES

CAN BULUT

For this project, I used 6 different colorful images to give them a cartoon effect. I will explain the experiment for 1 image and add the results of other 5 images at the end of this document.

This is the code snippet where I take the input image inside the code:

```
18
19    # You can change the input image from here
20    input_file = "img1.jpeg"
21
```

You can change the input image from the 20th line of the code.

This will be the image I'll use to explain the experiment with:

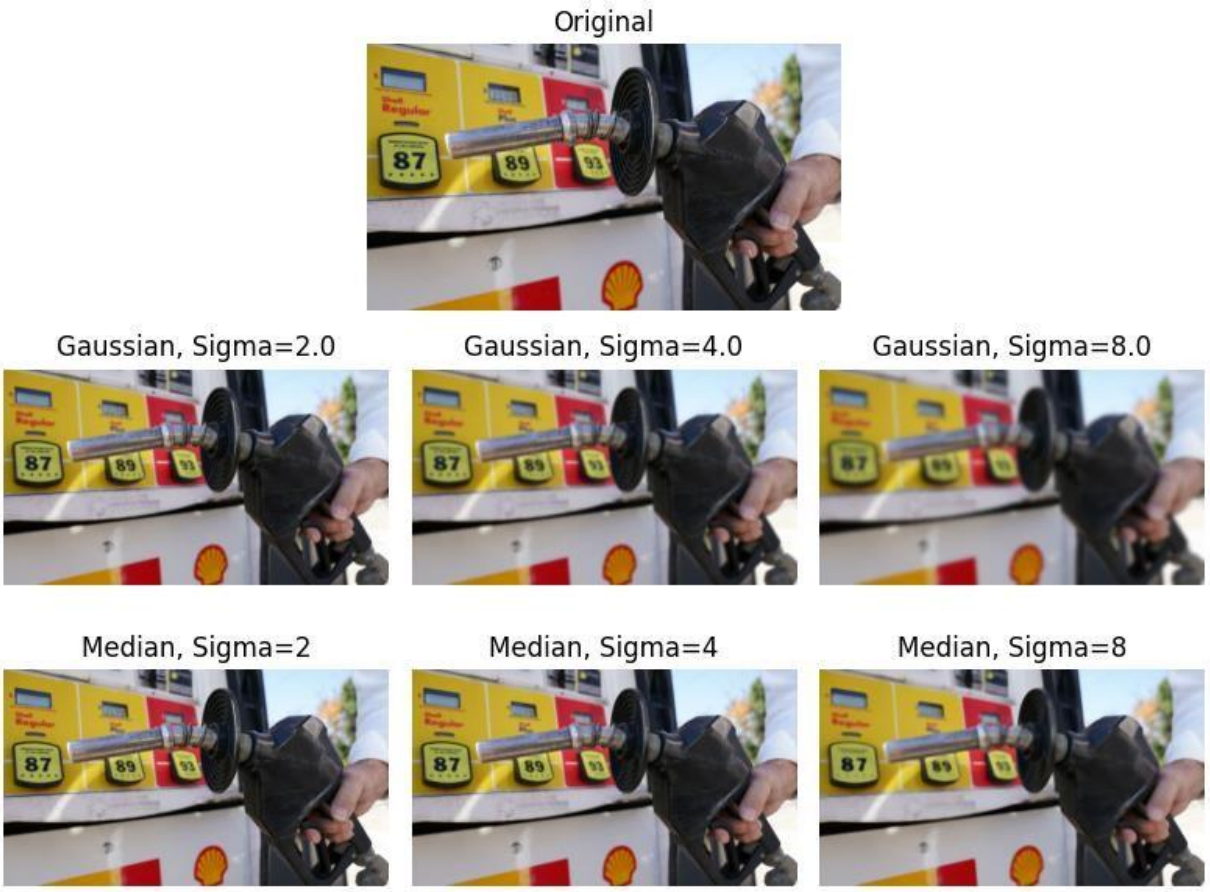
For this purpose, I carried out the following steps:

1. **Image Smoothing**:
   I convolved the input images with both a Gaussian and Median filter to see which of them gave a better result. I created an array containing different sigma values to see the results of the filter techniques. I used the values 2, 4, and 8 for this purpose. You can change You can see the code snippet of the parameters below:

```
66
67  # YOU CAN CHANGE THE PARAMETERS FROM HERE
68
69  gaussian_sigma_values = [2.0, 4.0, 8.0] # You can try out different sigma values for gaussian filter
70  median_sigma_values = [2, 4, 8]      # You can try out different sigma values for median filter
71
```

   You can also change the sigma values from the 69th line of the code.

   After using both Gaussian and Median filters with different sigma values, I got these results:



- As the sigma increased, the output image got more blurred.
- Gaussian filter gives a more blurred output than Median filter as the sigma increased.

2. **Edge Detection**:

I used thresholded Difference of Gaussian (DoG) filter to obtain pencil sketching effect. To do this, I defined DoG kernel as the differences of two Gaussian functions(DoG) as shown in the

Equation 1 and convoluted it with original image(Equation 2). Lastly I thresholded the filtered image with a small number ε (Equation 3). You can see the equations below:

$$D_{\sigma,k} = G_\sigma(x) - G_{k\sigma}(x) : k > 1 \quad (1)$$

$$I_f = D_{\sigma,k} * I \quad (2)$$

$$T_\epsilon(n) = \begin{cases} 1 & \text{if } I_f \geq \epsilon \\ 0 & \text{else} \end{cases} \quad (3)$$

I used 2,4 and 8 values for the threshold epsilon to see how it affects the result.

You can see the code snippet where I defined the parameters needed below:

```
121
122  # YOU CAN CHANGE THE PARAMETERS FROM HERE
123
124  k = 2 # You can change the k value from the first equation
125  sigma1_edge_1 = 2.0   # First sigma value from the first equation
126  sigma2_edge_1 = sigma1_edge_1 * k  # Second sigma value from the first equation
127  thresholds = [2, 4, 8] # You can try outt different threshold values from the third equation
128
```

You can also change the sigma, k and threshold values from 124th line of the code.

I also used k values for both 2 and 3 to see the results more clearly:

- As the threshold increased, the resulting image started to lose its definition and details started to fade away.
- As the k increased, the output gave a better result in terms of definiton as the threshold increased. You can see that the output when k=3 and threshold=8 gave a better result than when k=2 and threshold=8.

3. **Image Quantization**:

   Purpose of quantization is to simplify the color values for a cartoon like effect. I used both Lab and HSV to see how they

affect the result and showed the outputs for different num_colors values.

You can see the code snippet where I define the num_colors values inside an array below. Also, since we have to use the smoothed image for this part, I created an array containing the results of smoothed images from Part 1. I used the first smoothed image which I obtained with the Gaussian filter with the sigma value=2. That's why I'm using the first image of the array. (Please remember the output of Part 1.)

```
215   # YOU CAN CHANGE THE PARAMETERS FROM HERE
216
217   num_colors_values = [8,16]    # You can try out different num_colors values for both Lab and HSV
218   smoothed_image_index = 0      # You can try out different smoothed images from the first part
219
```

You can also change the values inside the num_colors_values array and choose a different smoothed image by changing the array index at the 217th line of the code.

Here's the output of Part 3:

Smoothed Image

Lab Quantized (Num Colors=8)

HSV Quantized (Num Colors=8)

Lab Quantized (Num Colors=16)

HSV Quantized (Num Colors=16)

- As the num_colors increased, the cartoon effect started to disappear.
- HSV technique gave a more cartoonish result than Lab technique. That's why I used HSV result in the next part.

4. Combining Edge and Quantized Image

After I extracted the image edges from the smoothed image in the Part 2 and the quantized image from Part 3, I combined them to get the cartoonish result. For this step, I took the inverse of the estimated edges values and multiplied it with the quantized image for each channel.

I used the first smoothed image which was obtained with Gaussian filter and sigma=2.
I used sigma1=2 and k=2 which led to sigma2=4.
I used threshold=20 for best result.
I used num_colors=8 for best result.

In the code snippet below, you can see which parameters I used to get the most cartoonish result.

```
260  # YOU CAN CHANGE THE PARAMETERS FROM HERE
261
262  smoothed_image_index = 0   # This is from the first part where we created 6 different smoothed images
263  k = 2                      # k value from the second part
264  sigma_1 = 2                # First sigma value from the second part
265  sigma_2 = sigma_1 * k      # Second sigma value from the second part
266  threshold = 20             # Threshold value from the second part
267  num_colors = 8             # num_colors value from the third part
268
```

You can also change these parameters from the 262nd line of the code.
Also, I used HSV instead of Lab because HSV gave a better result than Lab. You can see the related code snippet below:

```
268
269  # Threshold the edges
270  thresholded_edges = apply_threshold(apply_difference_of_gaussian(smoothed_images[smoothed_image_index], sigma_1, sigma_2), threshold)
271
272  # Load the quantized image from part 3 (HSV)
273  quantized_image_hsv = quantize_colors_hsv(smoothed_images[smoothed_image_index], num_colors)
274
275  # Visualize the combined image
276  visualize_combined_image(input_image, thresholded_edges, quantized_image_hsv, "output_combined_image")
```

Original Input Image

Cartoon Image

The ouput was fairly cartoonish version of the original image. Now, I'll show the results from other 5 images I experimented with.

Here’s the image 2:

In the following pages, you can see the outputs of each step for this image.

Original

Gaussian, Sigma=2.0

Gaussian, Sigma=4.0

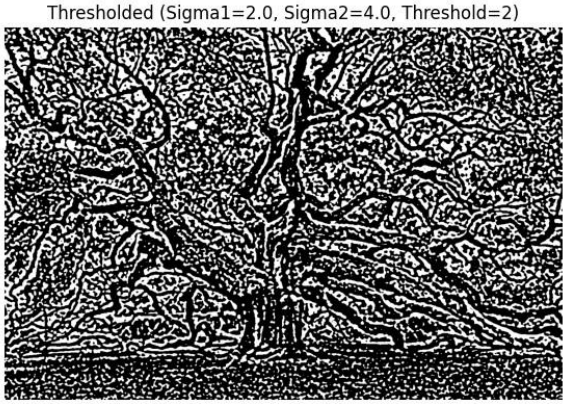
Gaussian, Sigma=8.0

Median, Sigma=2

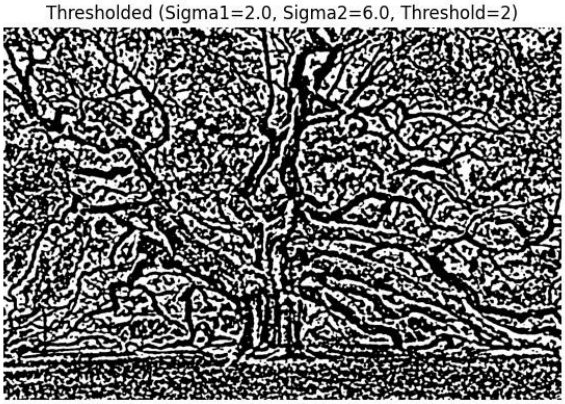
Median, Sigma=4

Median, Sigma=8

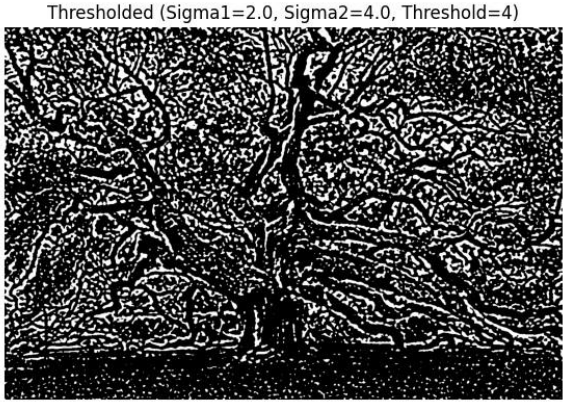
Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=2)



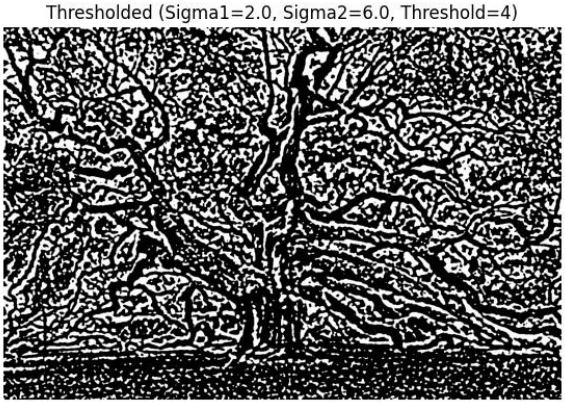
Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=2)



Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=4)

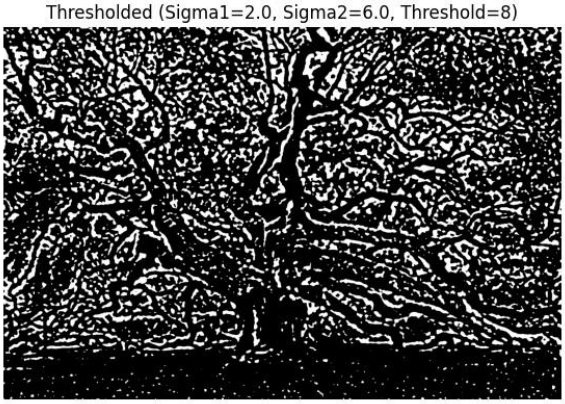


Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=4)



Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=8)

Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=8)



Smoothed Image

Lab Quantized (Num Colors=8)

HSV Quantized (Num Colors=8)

Lab Quantized (Num Colors=16)

HSV Quantized (Num Colors=16)

I used these values for the best cartoonish result:

```
# YOU CAN CHANGE THE PARAMETERS FROM HERE

smoothed_image_index = 0   # This is from the first part where we created 6 different smoothed images
k = 2                      # k value from the second part
sigma_1 = 2                # First sigma value from the second part
sigma_2 = sigma_1 * k      # Second sigma value from the second part
threshold = 40             # Threshold value from the second part
num_colors = 8             # num_colors value from the third part
```

You can see the result in the next page.

Original Input Image

Cartoon Image

Here’s the image 3:

The following pages has the output of every step for this picture.

Original

Gaussian, Sigma=2.0

Gaussian, Sigma=4.0

Gaussian, Sigma=8.0

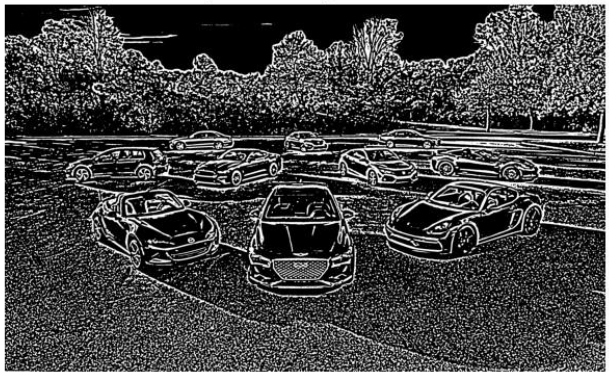
Median, Sigma=2

Median, Sigma=4

Median, Sigma=8

Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=2)

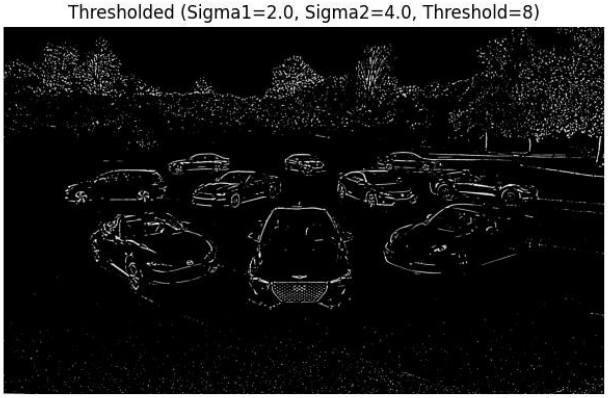
Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=2)



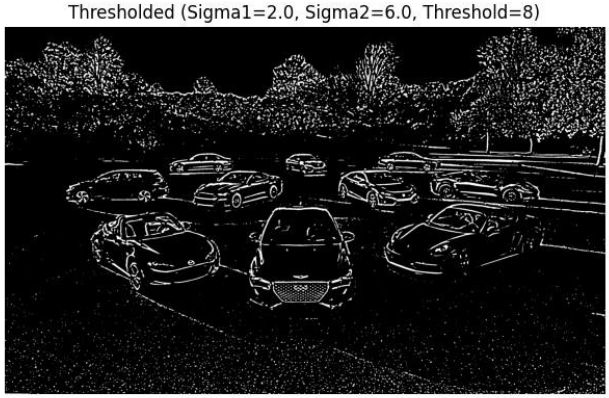
Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=4)

Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=4)

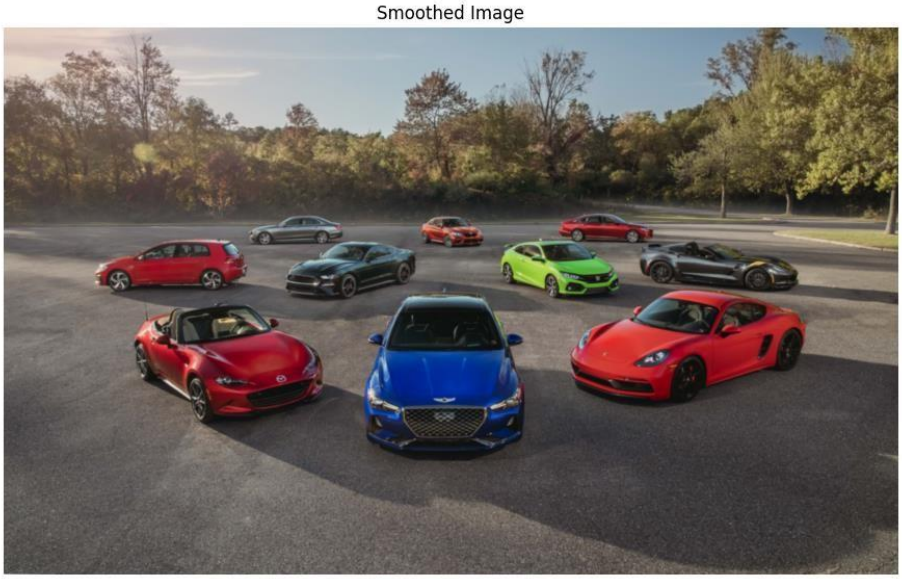
Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=8)



Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=8)



Smoothed Image



Lab Quantized (Num Colors=8)

HSV Quantized (Num Colors=8)

Lab Quantized (Num Colors=16)

HSV Quantized (Num Colors=16)

I used these parameters for best result:

```
# YOU CAN CHANGE THE PARAMETERS FROM HERE

smoothed_image_index = 0   # This is from the first part where we created 6 different smoothed images
k = 2                      # k value from the second part
sigma_1 = 2                # First sigma value from the second part
sigma_2 = sigma_1 * k      # Second sigma value from the second part
threshold = 40             # Threshold value from the second part
num_colors = 8             # num_colors value from the third part
```

The next page has the result.

Original Input Image

Cartoon Image

Here’s the image 4:

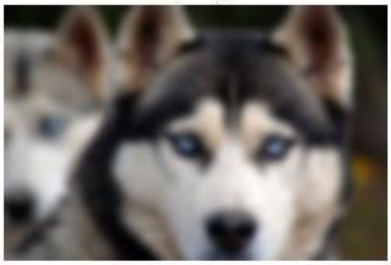
The following pages has the outputs of each part.

Original

Gaussian, Sigma=2.0

Gaussian, Sigma=4.0

Gaussian, Sigma=8.0

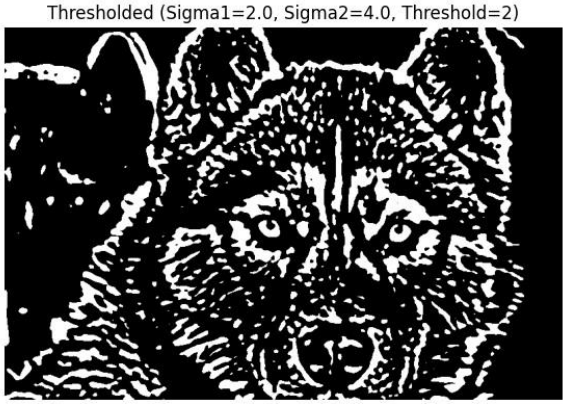


Median, Sigma=2

Median, Sigma=4

Median, Sigma=8

Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=2)



Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=2)

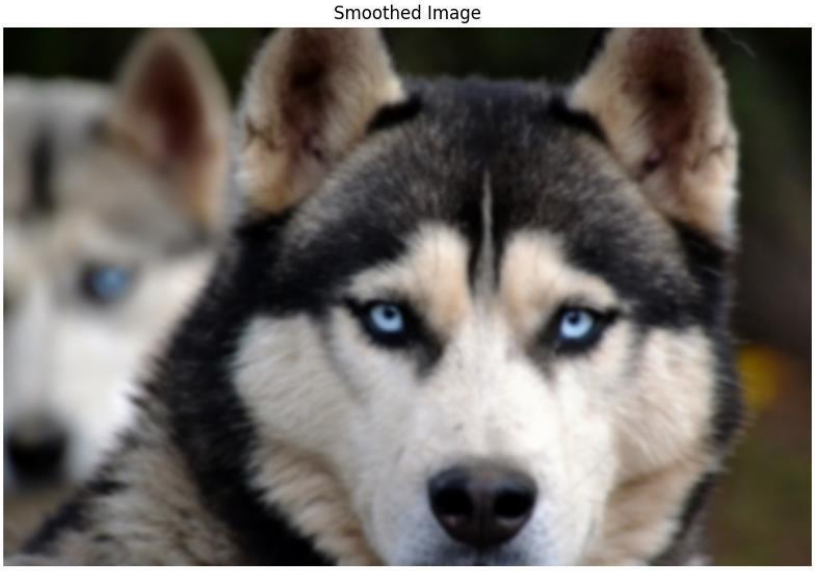
Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=4)

Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=4)

Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=8)

Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=8)

Smoothed Image



Lab Quantized (Num Colors=8)

HSV Quantized (Num Colors=8)

Lab Quantized (Num Colors=16)

HSV Quantized (Num Colors=16)

I used these parameters for the best result:

```
# YOU CAN CHANGE THE PARAMETERS FROM HERE

smoothed_image_index = 0   # This is from the first part where we created 6 different smoothed images
k = 2                      # k value from the second part
sigma_1 = 2                # First sigma value from the second part
sigma_2 = sigma_1 * k      # Second sigma value from the second part
threshold = 30             # Threshold value from the second part
num_colors = 8             # num_colors value from the third part
```

The next page has the result.

Original Input Image

Cartoon Image

Here’s the image 5:

The following pages has the outputs of each part.

Original

Gaussian, Sigma=2.0

Gaussian, Sigma=4.0

Gaussian, Sigma=8.0

Median, Sigma=2

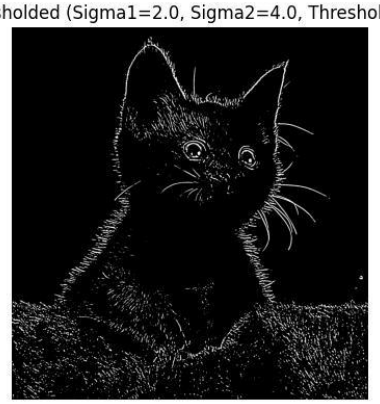
Median, Sigma=4

Median, Sigma=8

Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=2)

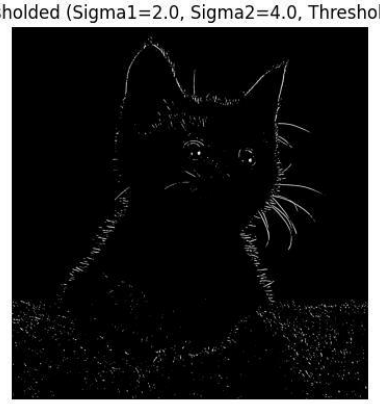
Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=2)

Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=4)



Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=4)

Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=8)

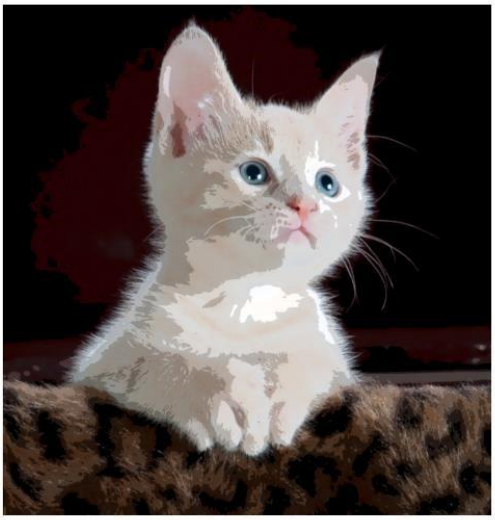


Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=8)

Smoothed Image

Lab Quantized (Num Colors=8)

HSV Quantized (Num Colors=8)



Lab Quantized (Num Colors=16)

HSV Quantized (Num Colors=16)

These are the parameters which gave the best result:

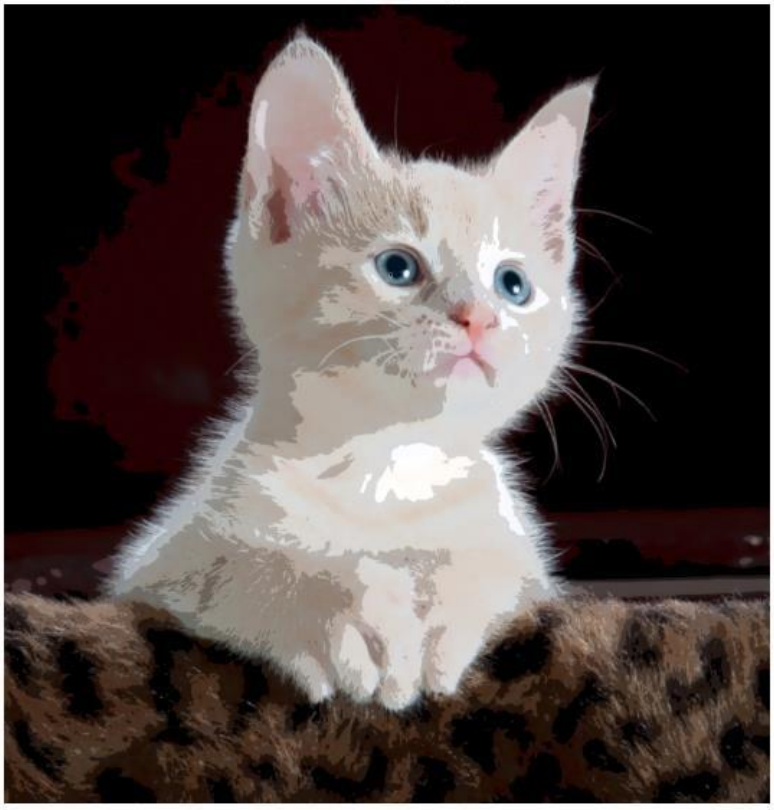
```
# YOU CAN CHANGE THE PARAMETERS FROM HERE

smoothed_image_index = 0   # This is from the first part where we created 6 different smoothed images
k = 2                      # k value from the second part
sigma_1 = 2                # First sigma value from the second part
sigma_2 = sigma_1 * k      # Second sigma value from the second part
threshold = 30             # Threshold value from the second part
num_colors = 8             # num_colors value from the third part
```

The next page has the result.

Original Input Image

Cartoon Image



Here’s the image 6:

The following pages has the outputs from every part.

Original

Gaussian, Sigma=2.0

Gaussian, Sigma=4.0

Gaussian, Sigma=8.0

Median, Sigma=2

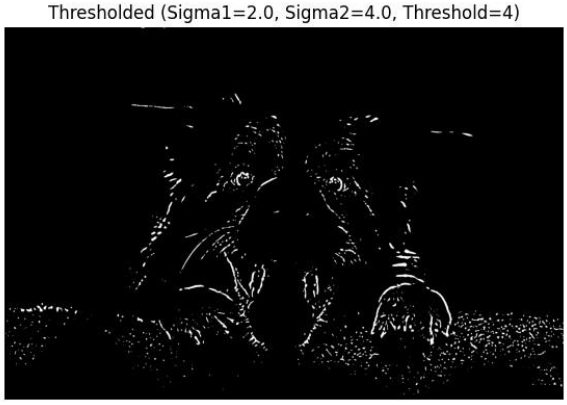
Median, Sigma=4

Median, Sigma=8

Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=2)

Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=2)

Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=4)



Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=4)

Thresholded (Sigma1=2.0, Sigma2=4.0, Threshold=8)

Thresholded (Sigma1=2.0, Sigma2=6.0, Threshold=8)

Smoothed Image

Lab Quantized (Num Colors=8)

HSV Quantized (Num Colors=8)

Lab Quantized (Num Colors=16)

HSV Quantized (Num Colors=16)

Here’s the parameters I used to get the best result:

```
# YOU CAN CHANGE THE PARAMETERS FROM HERE

smoothed_image_index = 0   # This is from the first part where we created 6 different smoothed images
k = 2                      # k value from the second part
sigma_1 = 2                # First sigma value from the second part
sigma_2 = sigma_1 * k      # Second sigma value from the second part
threshold = 20             # Threshold value from the second part
num_colors = 8             # num_colors value from the third part
```

The next page has the result.

Original Input Image

Cartoon Image