ECEN 642, Fall 2019

Texas A&M University

Electrical and Computer Engineering Department

Dr. Raffaella Righetti

Project Report

Hazer Removal Using Dark Channel Prior

Name: Abhay Vashist

UIN: 524008495

Date: December 10, 2019

Aggie Code of Honor

An Aggie does not lie, cheat or steal or tolerate those who do.

Abstract

The removal of haze has applications in consumer/computation photography and computer vision. The main focus of the project is to implement the Dark Prior Haze removal method. The main idea of the Dark channel prior method is the assumption has haze will have similar intensity in the entire three different color channels. This assumption bound the application of the algorithm too only outdoors images and bright images. The method also assumes that haze is only absorbed and scatter the light in an image. This assumption allows us to reconstruct a haze-free image by only knowing the ambiance of the image. One can use the information present in the dark before finding a good estimate of the ambiance. The paper had implemented, the Laplace matrix or soft matting was used to preserve edge information. In our implementation, we are using the guided filter to refine the transmission map and perform the edge preservation. The algorithm was successful to implement and apply the algorithm on 10 different images. The resulting image has a significant decrease in brightness, which not significant for a bright image

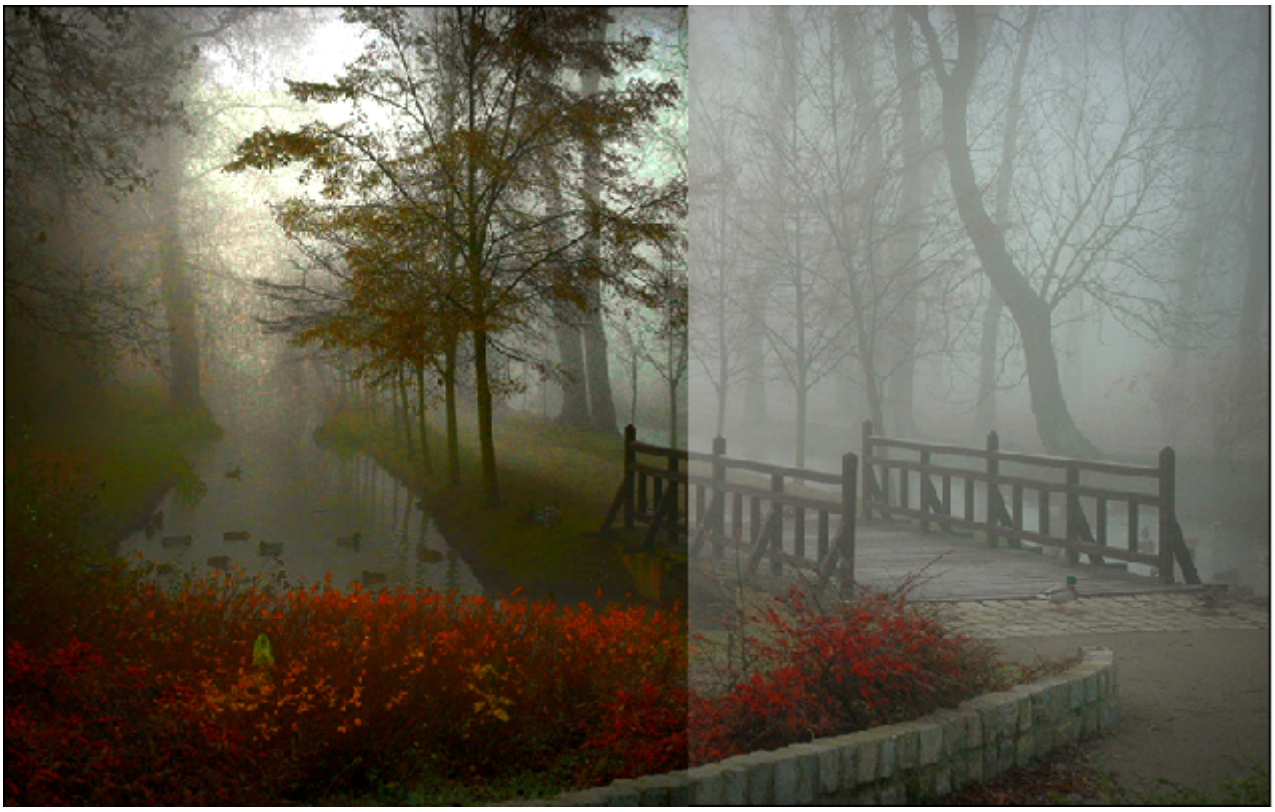


Figure 1 (Haze and Haze free image combined)

Methods

First we calculated the dark channel image using the equation 1. In the first stage we perform a min filter with a 15 by 15 window all color channel independently. In the next we perform a second min operator on the whole image base on the color dimension of the image. The stage of the process can be seen in Figure 1.

$$J^{dark}(\mathbf{x}) = \min_{c \in \{r,g,b\}} (\min_{\mathbf{y} \in \Omega(\mathbf{x})} (J^c(\mathbf{y}))),$$

Equation 1 (Dark channel equation)

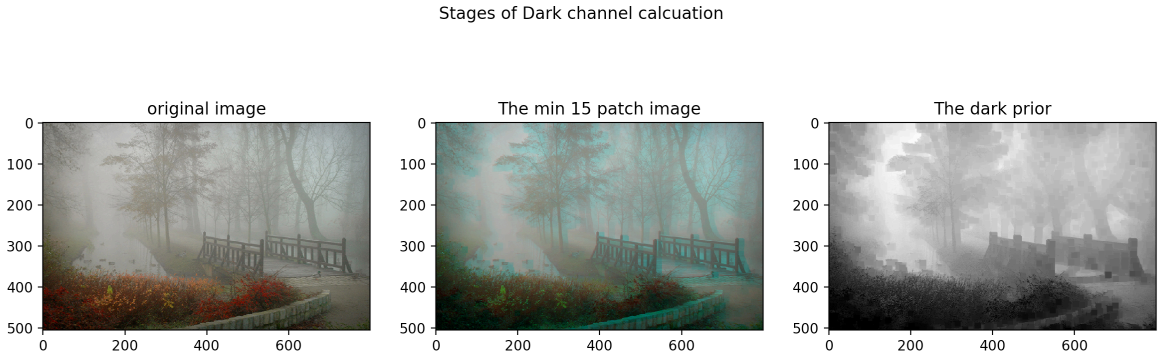


Figure 2 (Dark Channel Stages)

The next stage of the project is to extract an estimate of the atmospheric light of the original image. In the past the brightest pixel is used as the estimate values. This approach does not work for real world images, because there may be white objects present in the image. To avoid this problem one can used the information present in dark prior channel. We find the top 0.1% brightest pixels in the dark prior image. The pixel with highest intensity in this pool is used as the estimate for the atmospheric light. One can use equation 2 and the value of the atmospheric light to extract a Transmission map.

$$\tilde{t}(\mathbf{x}) = 1 - \omega \min_{c} (\min_{\mathbf{y} \in \Omega(\mathbf{x})} (\frac{I^c(\mathbf{y})}{A^c})).$$

Equation 2(Used to estimate the Transmission map)

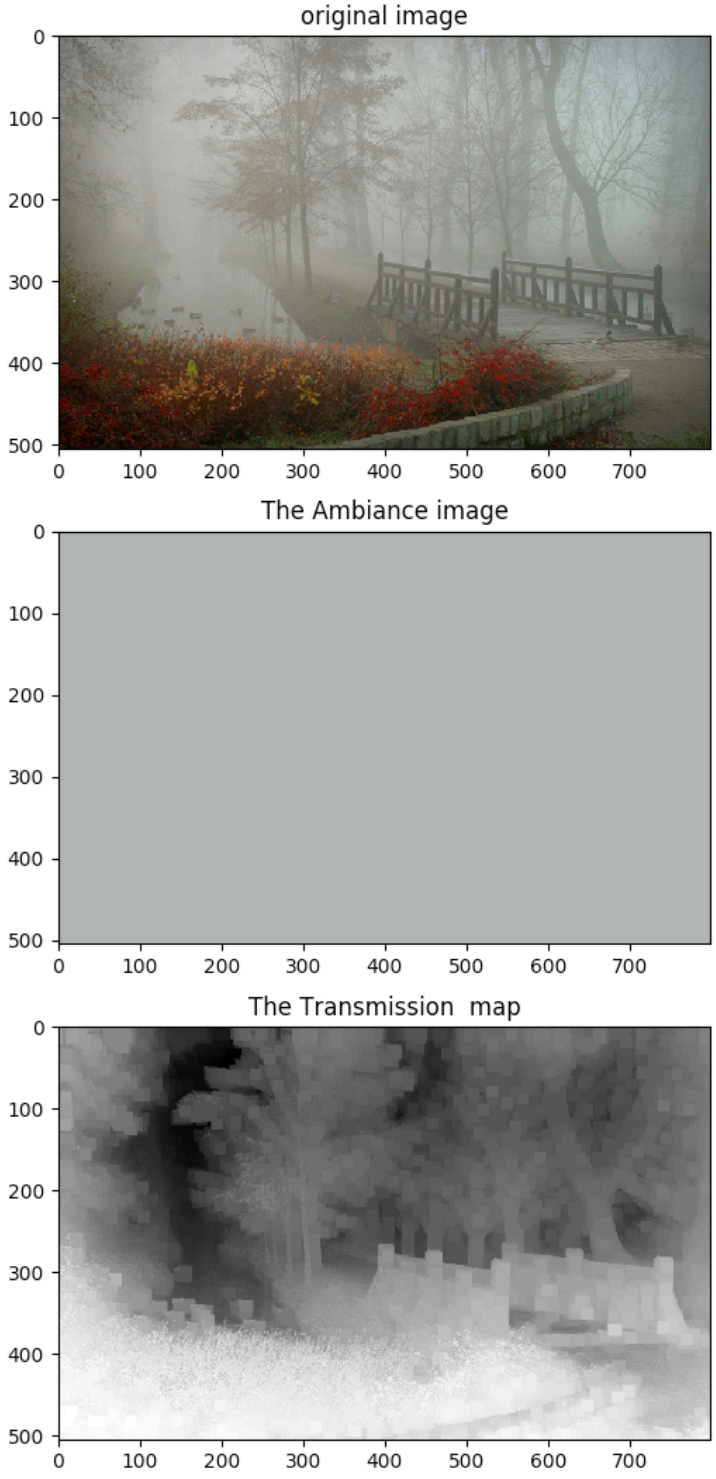


Figure 3 (The estimate of Ambiance and Transmission map)

After having good estimate for the transmission map one can use equation 3 to recover the original image. The denominator of the equation has a lower bound, because the original image is prone have noise and small amount of haze are preserved in areas with thick haze.

$$\mathbf{J}(\mathbf{x}) = \frac{\mathbf{I}(\mathbf{x}) - \mathbf{A}}{\max(t(\mathbf{x}), t_0)} + \mathbf{A}.$$

Equation 3 (Used to recover the haze image)

Figure 4 (The Transmission estimate of the image)

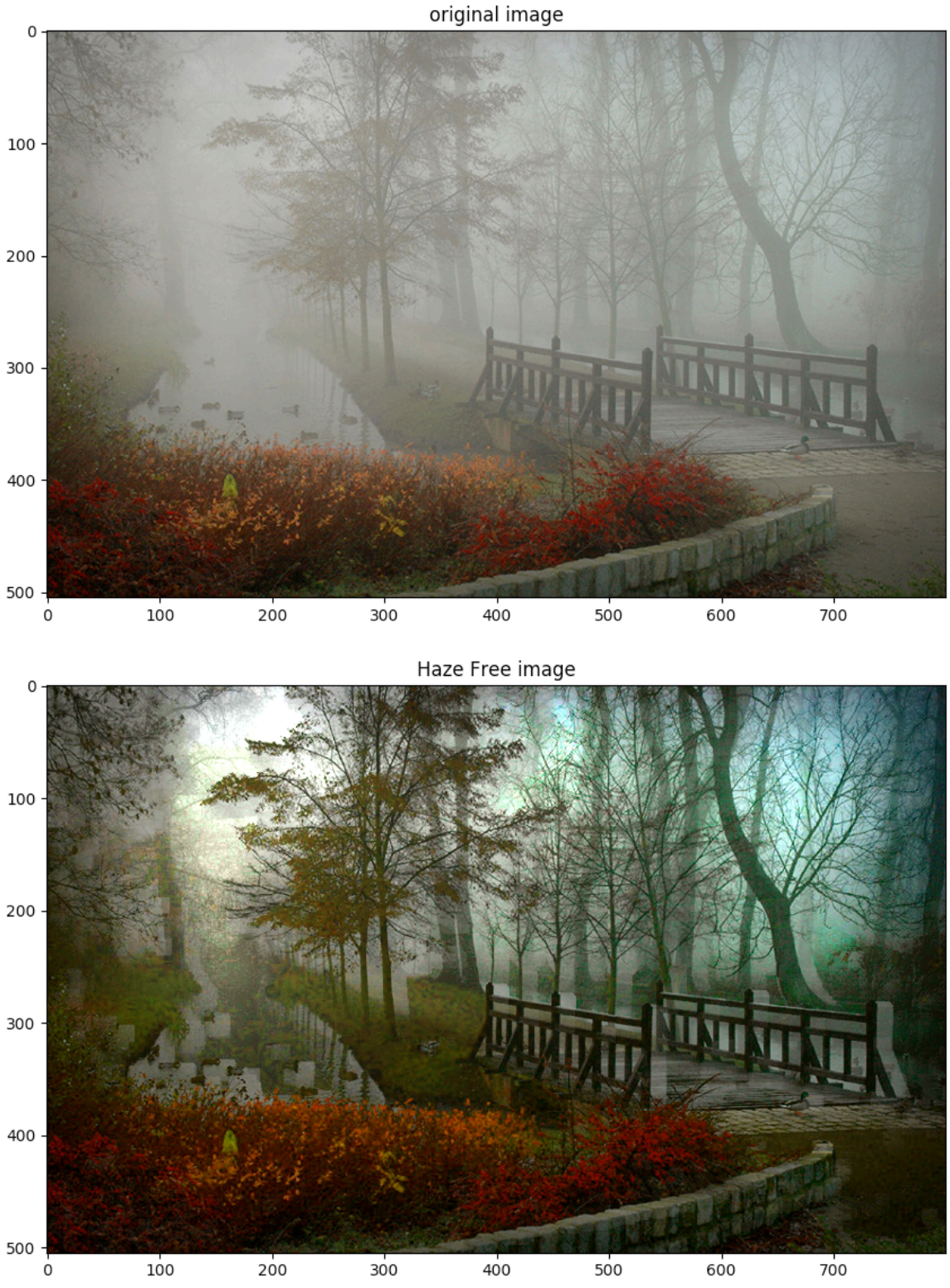


Figure 5 (Original and Recovered image)

The transmission map can be refining using soft matting of the map. However, the process of calculating the Laplacian matrix is very computation expensive. In this project, we used the idea of guided filter to refine the transmission, so the image will preserve the edge information. The guided filter was chosen over bilateral filtering, because it allows for O(N) complexity independent of the window size. Since guided filter only works with gray scale image, the original was convert to gray scale by averaging the 3 color levels.

$$\mathbf{a}_k = (\Sigma_k + \epsilon \mathrm{U})^{-1}(\frac{1}{|\omega|}\sum_{i \in \omega_k} \mathbf{I}_i p_i - \mu_k \bar{p}_k)$$
$$b_k = \bar{p}_k - \mathbf{a}_k^{\mathrm{T}} \mu_k$$
$$q_i = \bar{\mathbf{a}}_i^{\mathrm{T}} \mathbf{I}_i + \bar{b}_i.$$

Equation 3 (Equation used to extract the refined transmission map)

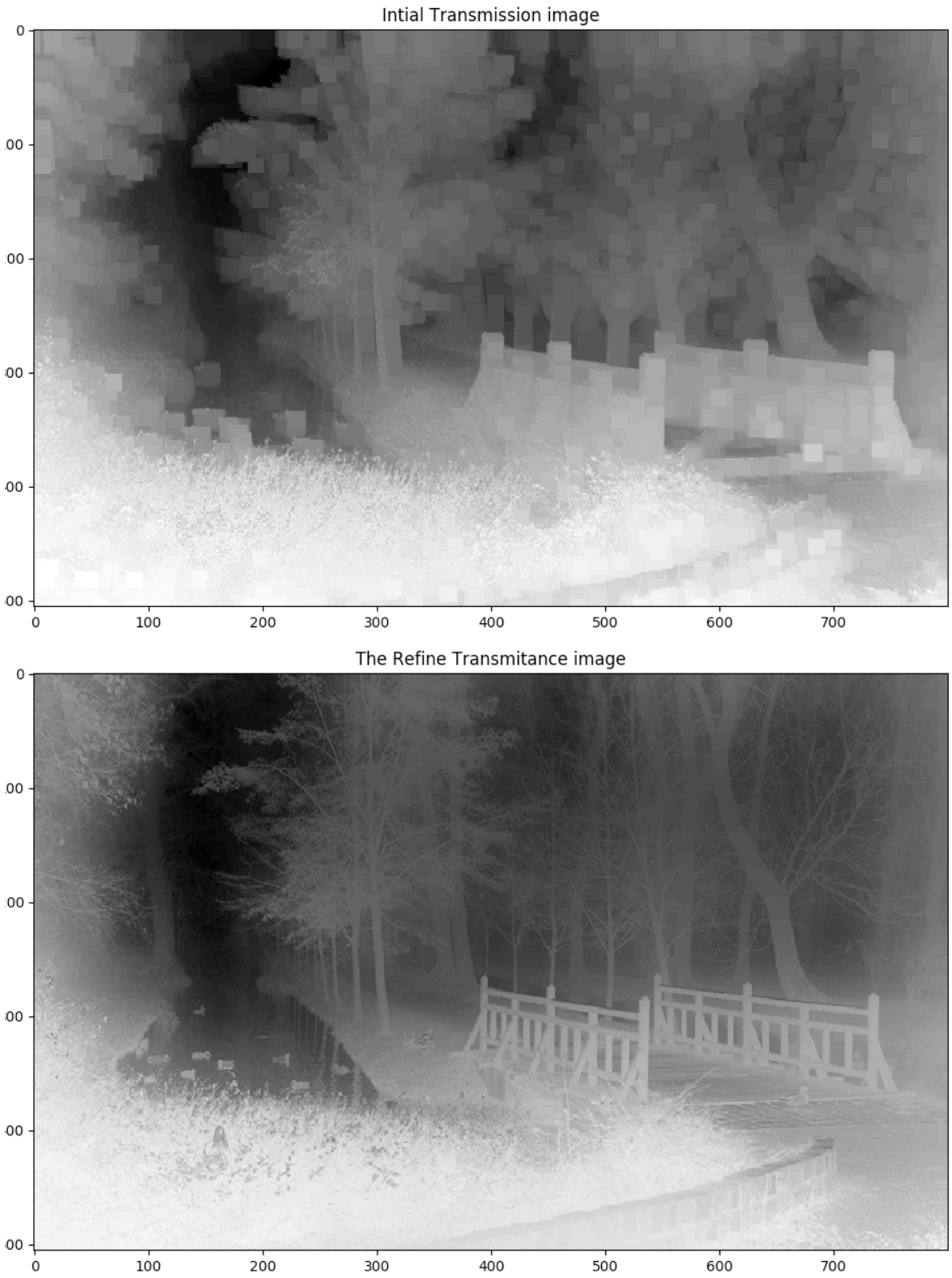


Figure 6 (Original and refined Transmission map)

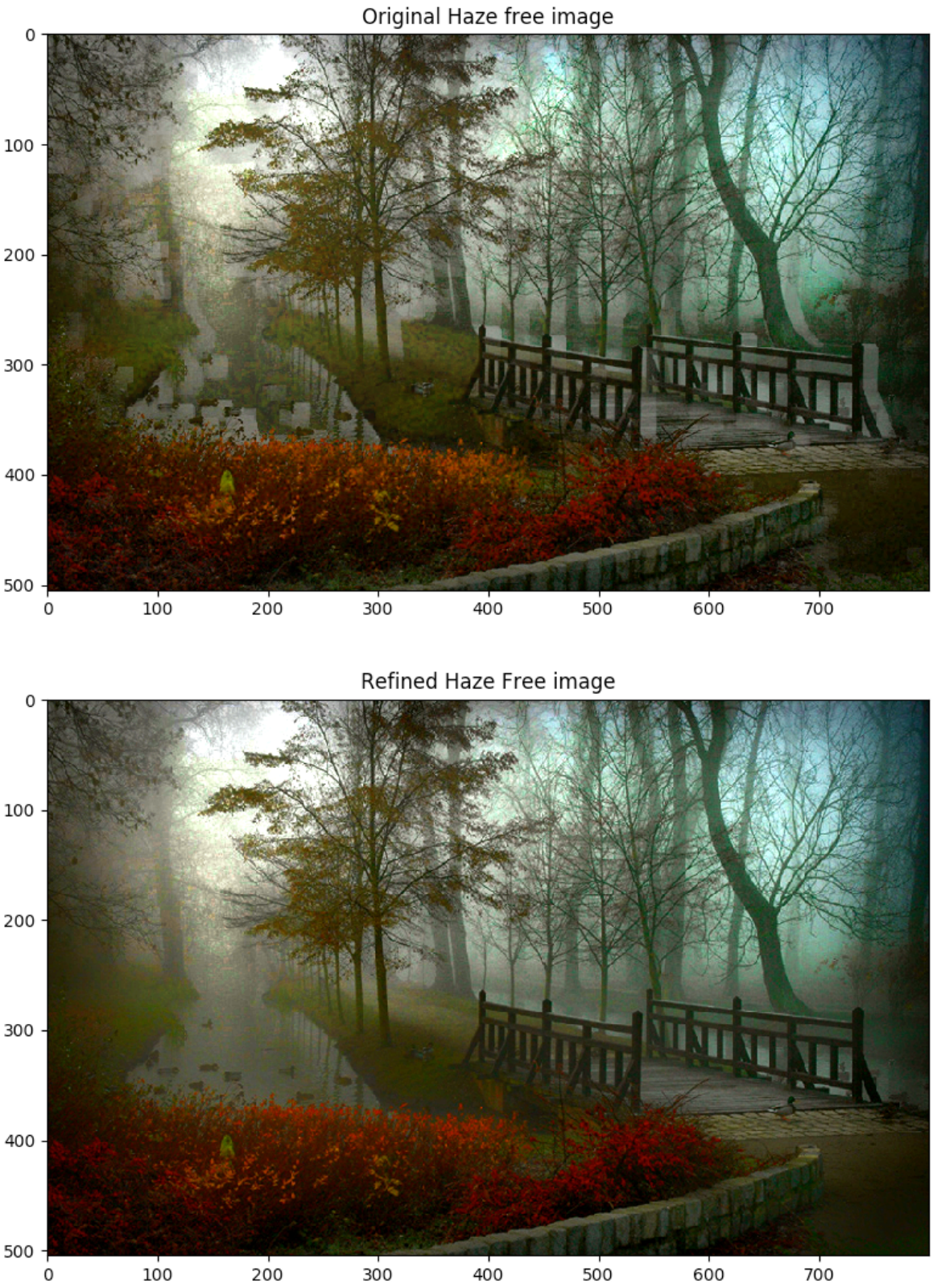


Figure 7 (Recovered image of original transmission map and refined)

Implementation

```
def min_filter(image):
    # perfroms the min filter on 15 by 15 area
    for k in range (3):
        # creating a copy of the filter
        i_image = image.copy()
        # extracting one channel of the image
        temp_image = image[:,:,k].copy()
        [row,col] = temp_image.shape
        # padding the iamge
        temp_image = cv2.copyMakeBorder(temp_image, 14, 14, 14, 14, cv2.BORDER_REFLECT)
        # perfroming the min filter with 15 x 15 window
        for i in range(row):
            for j in range(col):
                i_image[i,j,k] = (temp_image[i:15+i,j:15+j]).min()
    return i_image
```

Figure 8 (15 x 15 Min Filter)

```
def rgb_min_image(image):
    # extractes the min of the rgb values and outputs
    # a gray scale image
    rgb_image = np.amin(image, axis= 2)
    return rgb_image
```

Figure 9 (color min filter)

```
def dark_channel(image):
    # output the dark channel as the image
    new_image = image.copy()
    # perfroming the 15 x 15 min filter
    min_image = min_filter(new_image)
    # perfroming the color min operation
    dark_prior = rgb_min_image(min_image)
    return dark_prior
```

Figure 10 (dark channel function)

```python
def A_estimator(image,dark_prior):
    #Used the information extracted from the dark prior
    #find a value for A
    image_copy = image.copy()
    [row,col,dem] = image_copy.shape
    dark_copy = dark_prior.copy()
    # finding the number of 0.01% values
    num = np.round(row*col*0.001).astype(int)
    j = sorted(np.asarray(dark_copy).reshape(-1), reverse=True)[:num]
    # getting the location of the top 0.01%
    ind = np.unravel_index(j[0], dark_copy.shape)
    # Pefroming a search for the max value in the group
    max_val = image_copy[ind[0],ind[1],:]
    for element in j:
        ind = np.unravel_index(element, dark_copy.shape)
        if (sum(max_val[:]) < sum(image_copy[ind[0],ind[1],:])):
            max_val[:] = image_copy[ind[0],ind[1],:]
    # creating a color image of the max value
    A = image_copy
    A[:,:,:] = max_val[:]
    return A
```

Figure 11 (Function used to estimated the atmospheric light)

```python
def transmission_map(image,A,w):
    #finds the transmission map for the image
    image_new =  np.divide(image,A).astype(float)
    # finding the dark channel of the divide image
    new_dark = dark_channel(image_new)
    # Saling and subtracting the image
    transmission  = 1 - w*new_dark
    return transmission
```

Figure 12 (Code for calculating the Transmission map)

```python
def Radience_cal(image,A,Transmission_map,t_not):
    #Used information from the transmit map to remove haze from the image.
    image_copy = image.copy()
    Transmission_map_copy = (Transmission_map.copy()).astype(float)
    # Pefroming the min operation between Ttransmission map and 0.1
    divisor = np.maximum(Transmission_map_copy,t_not)
    radience = (image.copy()).astype(float)
    # Perfroming the eqution 3 for every color channel
    for i in range(3):
        radience[:,:,i] = np.divide(((image_copy[:,:,i]).astype(float) - A[0,0,i]),divisor) + A[0,0,i]
    # Capping all of the out of bound values
    radience[radience>255]=255
    radience[radience<0]=0
    return radience.astype('uint8')
```

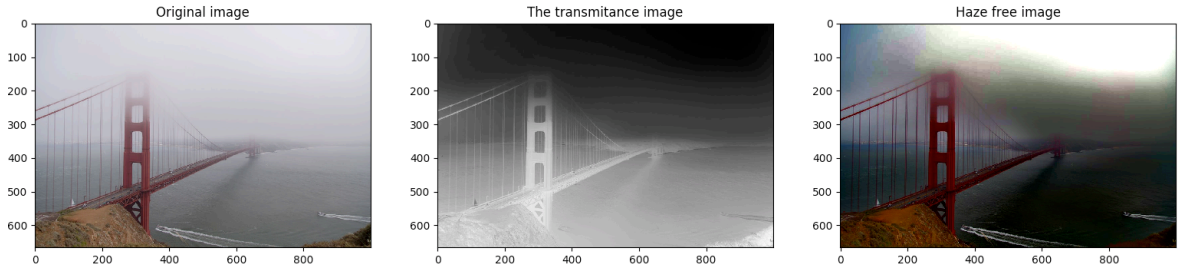
Figure 13 (Code Recovering the original image)

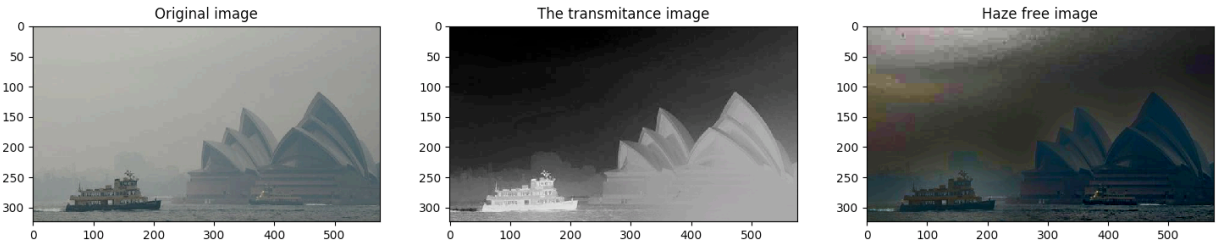
```python
def guided_filter(image,guide,diameter,epsilon):
    w_size = diameter+1
    # Exatrcation the mean of the image by blurring
    meanI=cv2.blur(image,(w_size,w_size))
    mean_Guide=cv2.blur(guide,(w_size,w_size))
    # Extracting the auto correlation
    II=image**2
    corrI=cv2.blur(II,(w_size,w_size))
    # Finding the correlation between image and guide
    I_guide=image*guide
    corrIG=cv2.blur(I_guide,(w_size,w_size))
    # using the mean of the image to find the variance of each point
    varI=corrI-meanI**2
    covIG=corrIG-meanI*mean_Guide
    #covIG normalized with a epsilon factor
    a=covIG/(varI+epsilon)
    #a is used to find the b
    b=mean_Guide-a*meanI
    meanA=cv2.blur(a,(w_size,w_size))
    meanB=cv2.blur(b,(w_size,w_size))
    # using the mean of a b to fix refine the transmission map
    transmission_rate=meanA*image+meanB
    # normalizaing of the transimational map
    transmission_rate = transmission_rate/np.max(transmission_rate)
    return transmission_rate
```

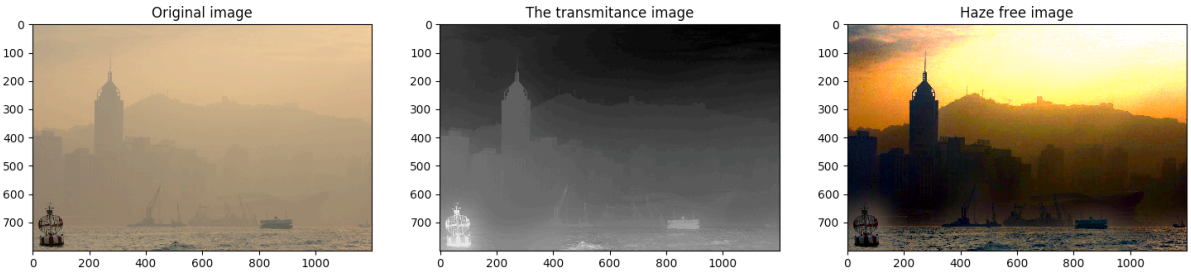
Figure 14 (Code for refining the Transmission map)

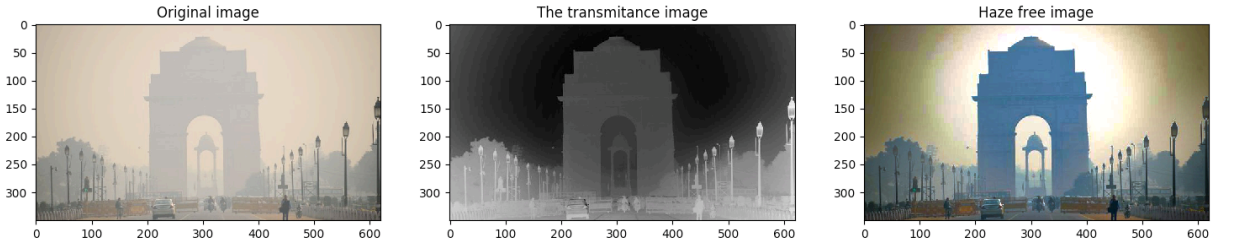
Results

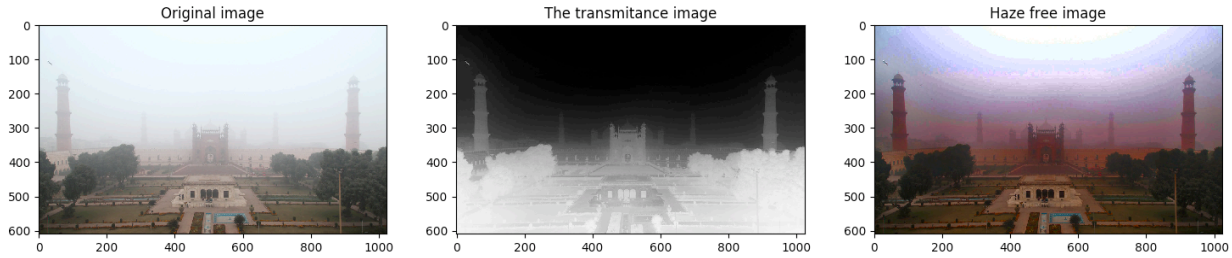

Original image
The transmitance image
Haze free image

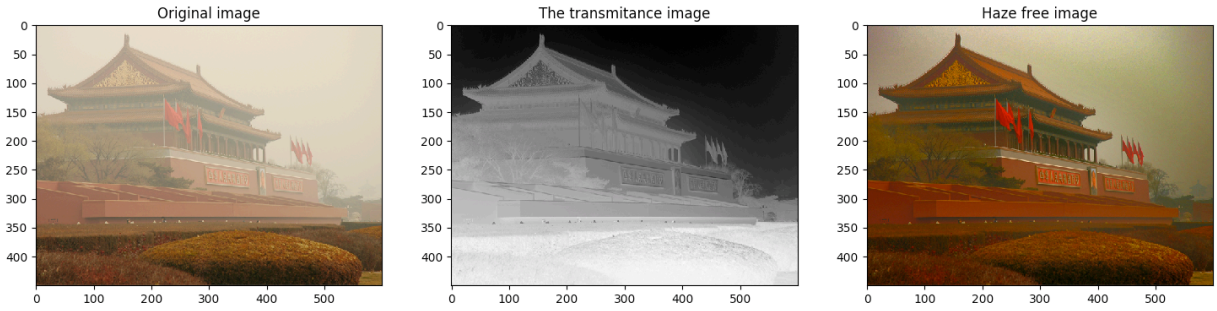

Original image
The transmitance image
Haze free image

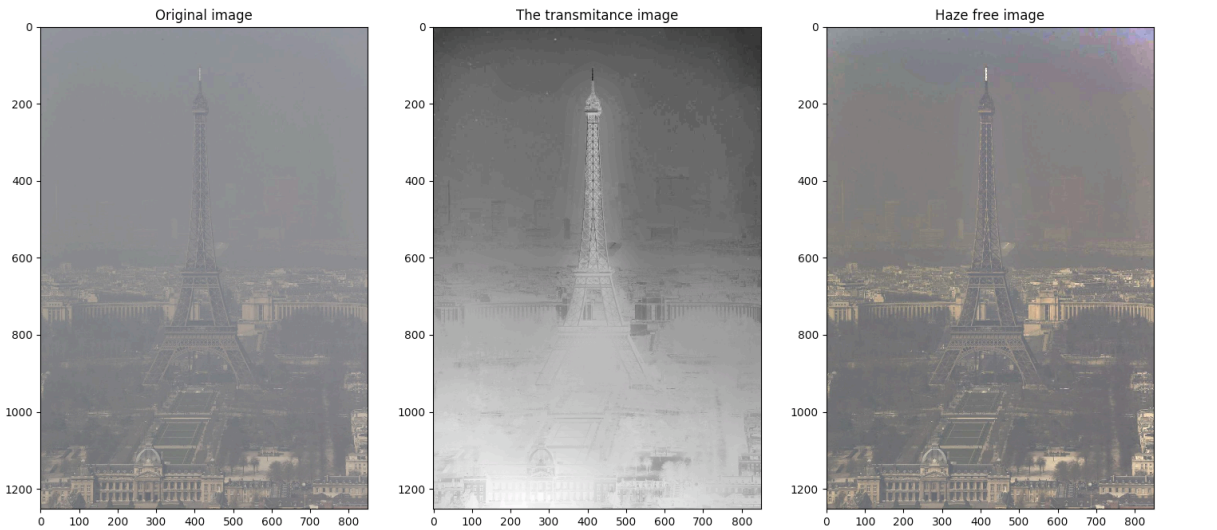

Original image
The transmitance image
Haze free image

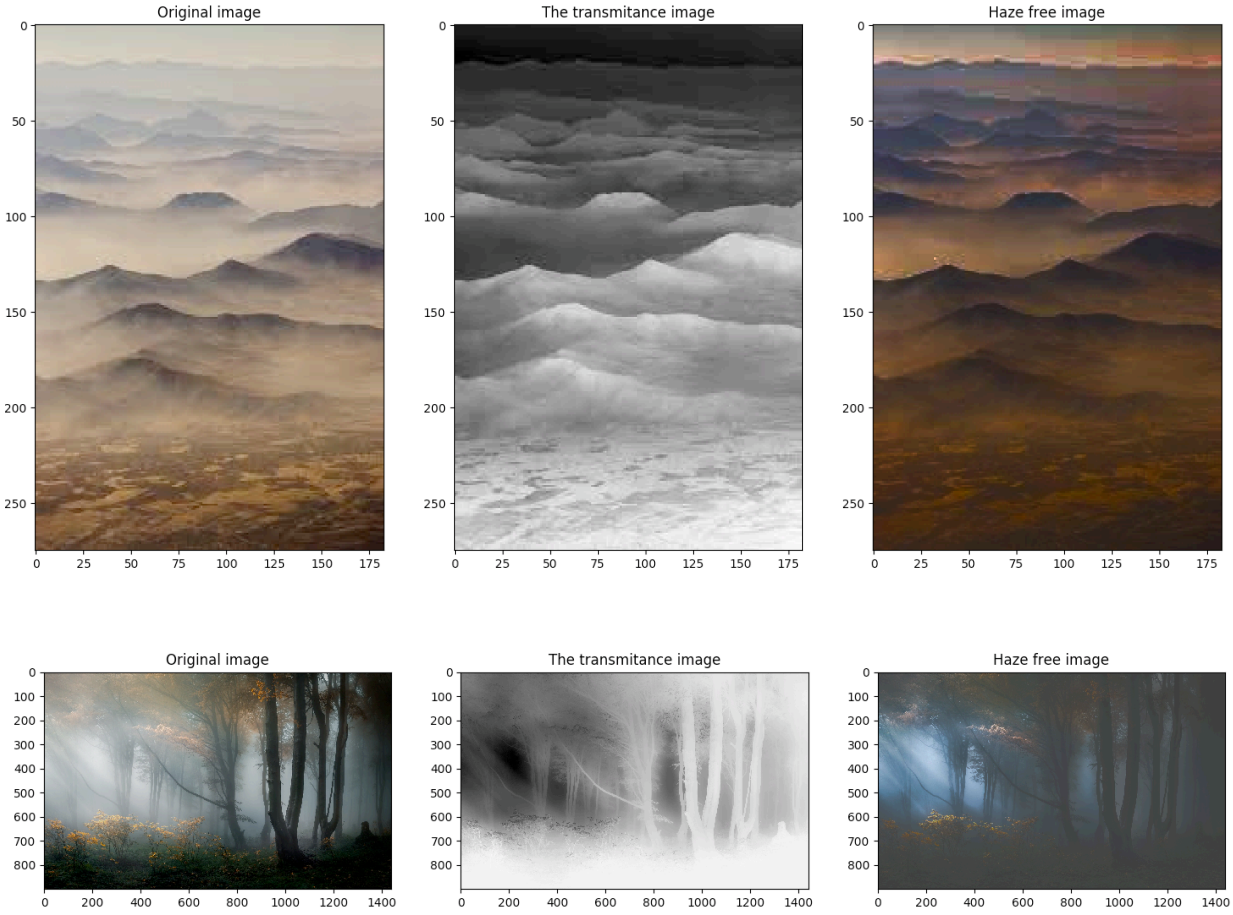

Original image
The transmitance image
Haze free image


Original image
The transmitance image
Haze free image


Original image
The transmitance image
Haze free image


Original image
The transmitance image
Haze free image


Original image
The transmitance image
Haze free image



Conclusion

We implement the algorithm on 10 different images and the algorithm is not perfect in the removal of all of the images. In our implementation, we perform scaling by clipping image into the desired range, because it helped maintain image contrast. This was different than, what was recommended by the paper. In our results, we found that image, which haze needs to be bright for the algorithm to produce optimal results. In the image of the Sydney Opera house, the haze-free image was worse than the original image, because there was not enough lighting present in the image. On the other hand, the image of the Hong Kong port has a huge bright area, which dominates the dark prior. For the algorithm to produce optimal results, the input image haze is bright, and should not have large bright areas. The result indicates the limitation of the dark prior method.

Reference

K. He, J. Sun and X. Tang, "Single Image Haze Removal Using Dark Channel Prior," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 12, pp. 2341-2353, Dec. 2011.
doi: 10.1109/TPAMI.2010.168

He, Kaiming, Jian Sun, and Xiaoou Tang. "Guided image filtering." *European conference on computer vision*. Springer, Berlin, Heidelberg, 2010.

Implementation

https://github.com/Avashist1998/Morning_view_mode