Implementation of Non-Local Image Dehazing Based on OpenCV

Jiangnan Li
Ph.D. student in Computer Engineering
Instructor: Dr. Mongi Abidi

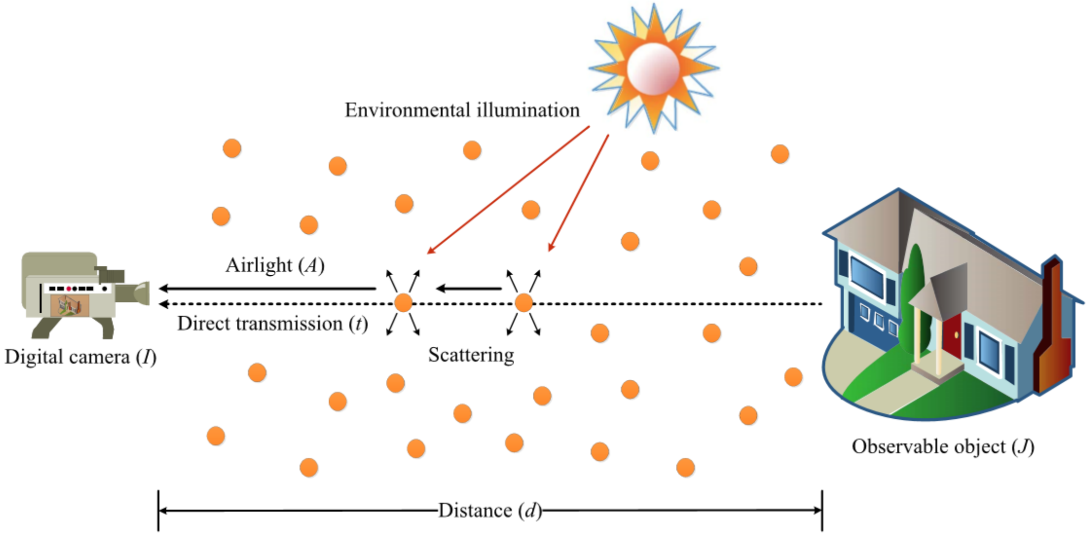
THE UNIVERSITY OF
TENNESSEE
KNOXVILLE

BIG ORANGE. BIG IDEAS.®

1. Background

Outdoor images often suffer from low contrast and limited visibility due to small particles in the air. Image Hazing can cause problems for some systems.(i.e. Obstacle detection systems)

2. Haze Model



$$I(x) = t(x) \cdot J(x) + [1 - t(x)] \cdot A$$

$$t(x) = e^{-\beta d(x)}$$

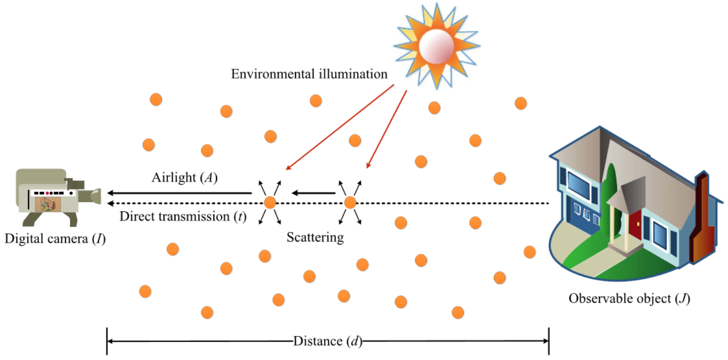
$\mathbf{x}$	Pixel coordinate
I	Observed hazy image
J	True radiance of scene point
A	Airlight
t	Scene Transmission
β	Attenuation coefficient

3. Analysis of Haze Model

An ill-posed problem

$$\boldsymbol{I}(\boldsymbol{x}) = t(\boldsymbol{x}) \cdot \boldsymbol{J}(\boldsymbol{x}) + [1 - t(\boldsymbol{x})] \cdot \boldsymbol{A}$$

$$\begin{pmatrix} b_I \\ g_I \\ r_I \end{pmatrix} = t(\boldsymbol{x}) \begin{pmatrix} b_J \\ g_J \\ r_J \end{pmatrix} + [1 - t(\boldsymbol{x})] \begin{pmatrix} B \\ G \\ R \end{pmatrix}$$



$$\begin{cases} b_I = t \cdot b_J + (1 - t) \cdot B \\ g_I = t \cdot g_J + (1 - t) \cdot G \\ r_I = t \cdot r_J + (1 - t) \cdot R \end{cases}$$

Unknown for every pixel

Known for every pixel

Airlight

3. Analysis of Haze Model

Assumption :

1. The Airlight is known (The Airlight can be estimated)

2. In a clear image, the number of distinct colors in an image is orders of magnitude smaller than the number of pixels

Assumption2 indicates that a small number of distinct colors can holds for haze-free images.

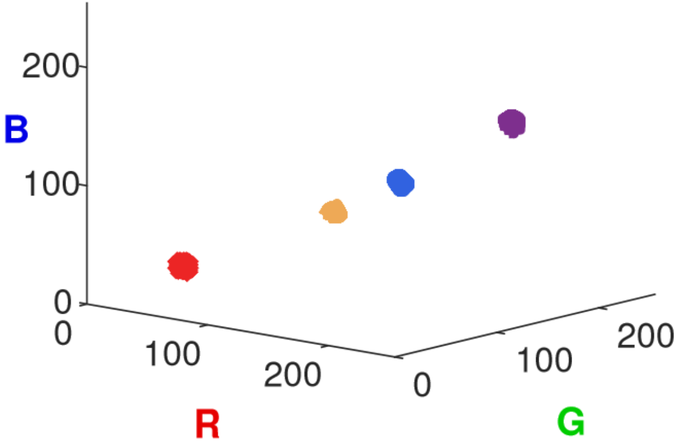
Before color quantization

After color quantization

3. Analysis of Haze Model

Assumption2 indicates that a small number of distinct colors can holds for haze-free images.

Clear image pixels form clusters in RGB space.



So, what will happen if the image is hazy ?

3. Analysis of Haze Model

So, what will happen if the image is hazy ?

$$\begin{cases} b_I = t \cdot b + (1-t) \cdot B \\ g_I = t \cdot g + (1-t) \cdot G \\ r_I = t \cdot r + (1-t) \cdot R \end{cases} \Longrightarrow \begin{cases} b_I = t \cdot (b-B) + B \\ g_I = t \cdot (g-G) + G \\ r_I = t \cdot (r-R) + R \end{cases}$$

$$\xrightarrow[(r-R)=Z]{\substack{(b-B)=X \\ (g-G)=Y}} \begin{cases} (b_I - B)/X = t \\ (g_I - G)/Y = t \\ (r_I - R)/Z = t \end{cases} \Longrightarrow \boxed{\frac{(b_I - B)}{X} = \frac{(g_I - G)}{Y} = \frac{(r_I - R)}{Z} = t}$$

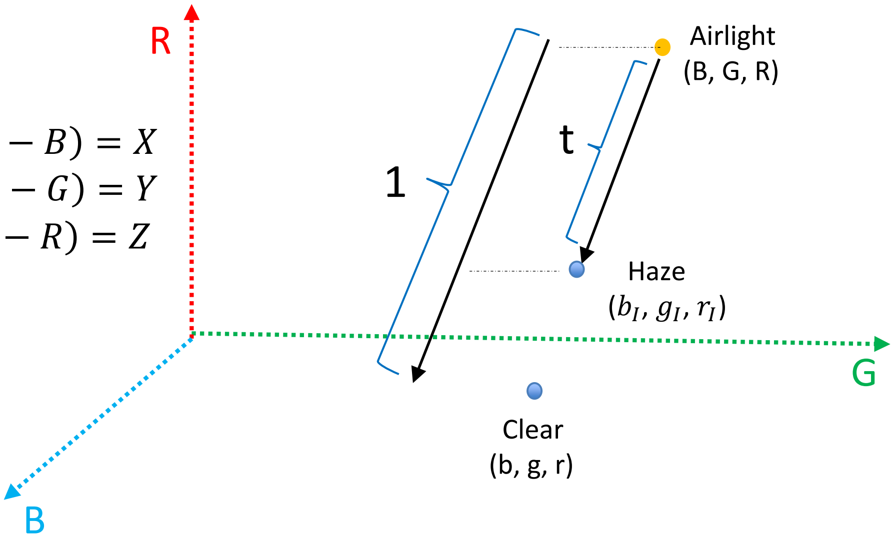
(Standard linear equation)

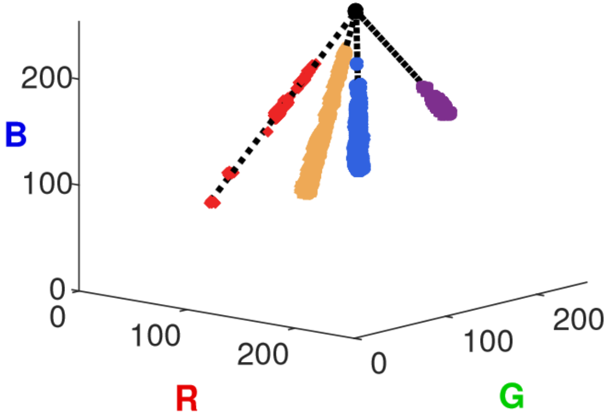
Original point: (B, G, R) Vector: (b-B, g-G, r-R)

3. Analysis of Haze Model

$$\boxed{\frac{(b_I - B)}{X} = \frac{(g_I - G)}{Y} = \frac{(r_I - R)}{Z} = t}$$

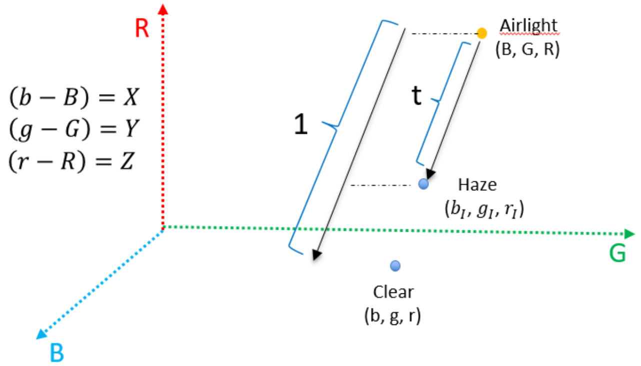
$$(b - B) = X$$
$$(g - G) = Y$$
$$(r - R) = Z$$





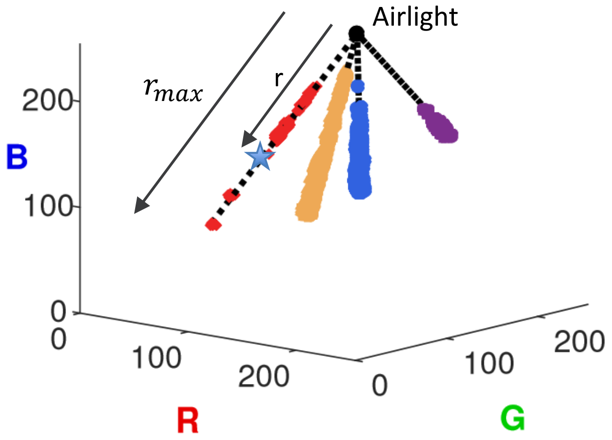
3. Analysis of Haze Model

$$\boxed{\frac{(b_I - B)}{X} = \frac{(g_I - G)}{Y} = \frac{(r_I - R)}{Z} = t}$$



$$\frac{r}{r_{max}} = t$$

(Assume the farthest pixel is clear)



To simply calculation, use spherical coordinates

4. Dehazing Algorithm and Implementation

$$\boldsymbol{I}(\boldsymbol{x}) = t(\boldsymbol{x}) \cdot \boldsymbol{J}(\boldsymbol{x}) + [1 - t(\boldsymbol{x})] \cdot \boldsymbol{A}$$

Input : $\boldsymbol{I}(\boldsymbol{x})$ $\boldsymbol{A}$ **Output:** $t(\boldsymbol{x})$ $\boldsymbol{J}(\boldsymbol{x})$

1: $\boldsymbol{I}_{\mathrm{A}}(\boldsymbol{x}) = \boldsymbol{I}(\boldsymbol{x}) - \boldsymbol{A}$
2: Convert $\boldsymbol{I}_{\mathrm{A}}$ to spherical coordinates to obtain $[r(\boldsymbol{x}), \phi(\boldsymbol{x}), \theta(\boldsymbol{x})]$
3: Cluster the pixels according to $[\phi(\boldsymbol{x}), \theta(\boldsymbol{x})]$. Each cluster H is a *haze-line*.
4: **for** each cluster H **do**
5: Estimate maximum radius: $\hat{r}_{\max}(\boldsymbol{x}) = \max_{\boldsymbol{x} \in H}\{r(\boldsymbol{x})\}$
6: **for** each pixel $\boldsymbol{x}$ **do**
7: Estimate transmission: $\tilde{t}(\boldsymbol{x}) = \frac{r(\boldsymbol{x})}{\hat{r}_{\max}}$
8: Perform regularization by calculating $\hat{t}(\boldsymbol{x})$ that minimizes Eq. 15
9: Calculate the dehazed image using Eq. (16)

Date Structure

```
Class Pixel
{
Private:
        int red, blue, green;
        int gray;
        int row, col ;
Public:
        ……
}
```

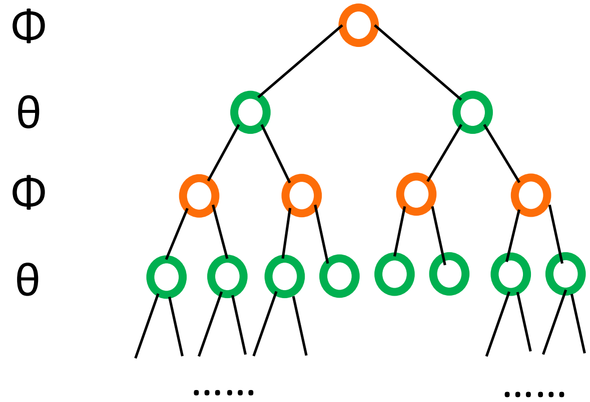
4. Dehazing Algorithm and Implementation

Important step :

Cluster the pixels according to [Φ(x), θ(x)]

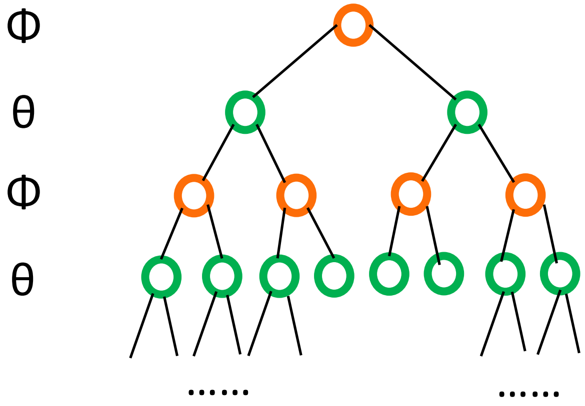
Implementation Method : K-dimensions Tree Data Structure

```
Struct KdtreeNode
{
        int x, y, depth;
        float SpherePoint [3]; //store the r,  Φ, θ value
        KdtreeNode * left, * right;
}
```



4. Dehazing Algorithm and Implementation

Implementation Method : K-dimensions Tree Data Structure



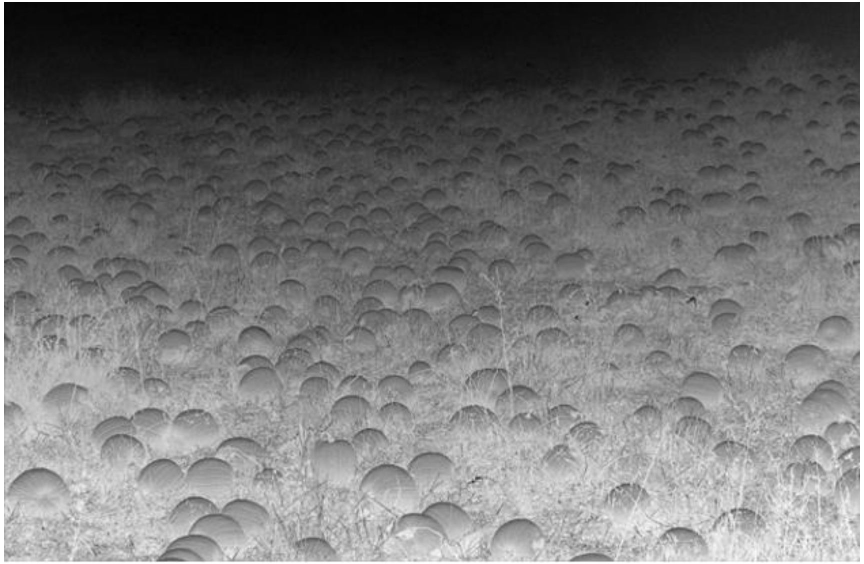

```
Struct KdtreeNode * BuildKdtree(Mat * image, … ){
      struct KdtreeNode * root = NULL;
      while(Pixels) {
             root =  Insert_KdtreeNode()
      }
}

Struct KdtreeNode * Insert_KdtreeNode(){
      if(root == NULL) return Insert_KdtreeNode()
      if(dep%2 == 0) (root.phi ) ? Root->left = Insert_KdtreeNode() : root->right = Insert_KdtreeNode();
      if(dep%2 != 0) (root.theta ) ? Root->left = Insert_KdtreeNode() : root->right = Insert_KdtreeNode();
      return root;
}

Struct KdtreeNode * cluster(KdtreeNode*root){
       vector<KdtreeNode * > clusters;
       while() { clusters.pushback( findKdtreeNode(depth)) }
}
```

5. Experiment Result

Input Image

My Result

5. Experiment Result

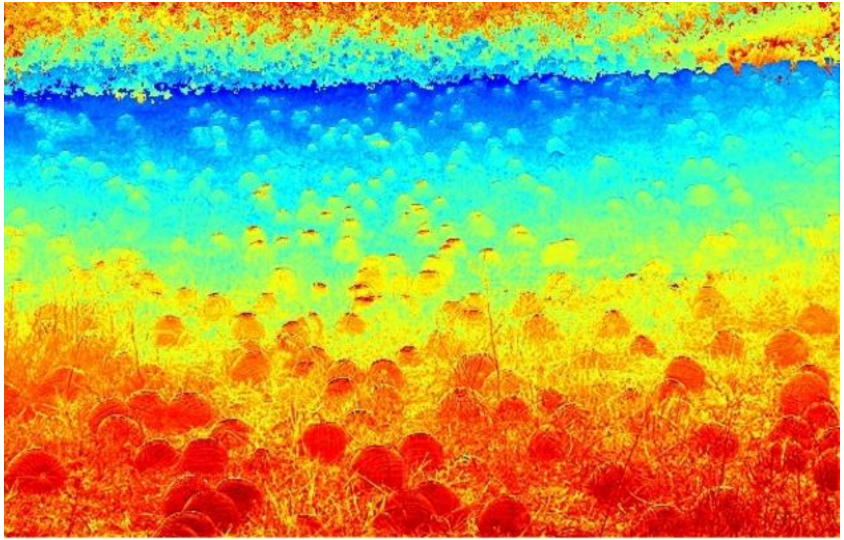


Author's r(x)

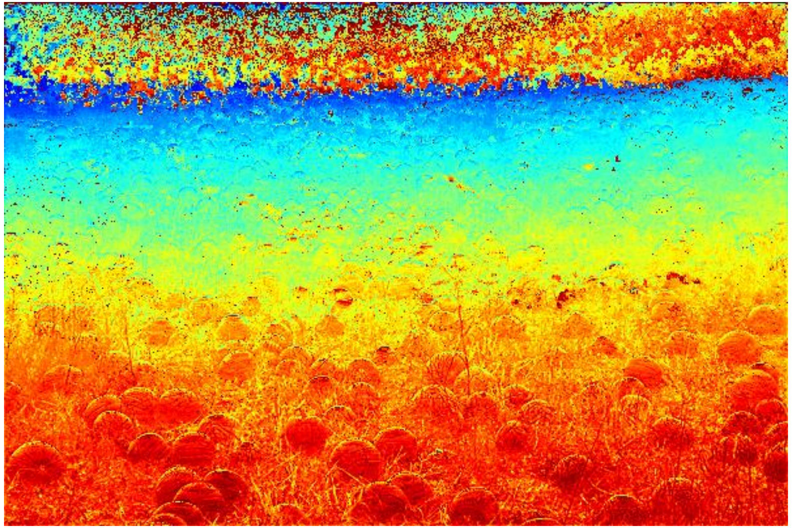
My r(x)

BIG ORANGE BIG IDEAS

5. Experiment Result



Author's Transmission Map



My Transmission Map

5. Experiment Result

Author’s Output

My Output

6. Analysis for Difference

(1) Different Method for Airlight Estimation

(2) Different Algorithm for Clustering Pixels

(3) Lacking the last smoothing step (easy for Matlab but difficult for C++)

7. Modification

Thought: Using the farthest (clear) Pixel whose r is Rmax is a cluster to replace all other pixels in the cluster.

Hazy Image Input

Modified Output

7. Demo

Thank you !