TJHSST Computer Systems Research Lab 2019
Senior Research

ClearCam: Real-time Imaging Through Fog Via GPU-Accelerated Stereo Depth Estimation and Parameter Fitting of an Atmospheric Scattering Model

Neal Bayya, George Tang

Abstract
We propose a system for the realtime removal of haze from video input. Existing research focuses on single image dehazing, which cannot exploit the correlation between depth and haze scattering. To this end, our approach leverages stereo depth estimation and parameter fitting of an atmospheric scattering model to extract the clear scene from the hazy scene in each frame of the video sequence. We propose an algorithm to refine the stereo disparity maps in low visibility conditions. We then describe a Convolutional Neural Network model which learns the coefficient of scattering in each frame. Due to the lack of natural stereo haze data, we synthesized a dataset from the indoor Middlebury 2005 stereo dataset with a variety of haze conditions. Experimental results show an increase in the structured similarity with the reference frames, indicating an improvement in visibility. As extensions of our system, we propose a method of non-homogeneous haze synthesis which more accurately models low visibility conditions and an approach towards scattering estimation in the outdoor HazeRD dataset.

Contents

1. Introduction

In fog, haze particles scatter light, making it impossible to see or image properly. This causes many problems in photography, navigation, automation, and many other fields. For example, self-driving cars have great difficulty recognizing traffic signs, pedestrians, and vehicles in fog, which is a major obstacle to their commercialization. Low-visibility conditions greatly hinder security systems or search and rescue missions. Multiple solutions have been proposed, but each has its shortcomings.

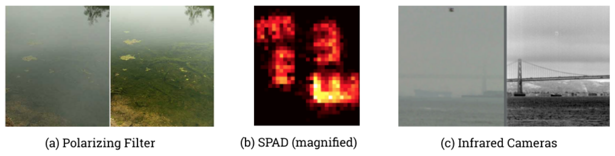


Figure 1. Current Solutions to Imaging Through Fog

As scattered light is partially polarized, we can apply a polarizing filter to the lens of the camera, but its performance is extremely poor (see Figure 1). In 2018, researchers at MIT used a laser in conjunction with a single-photon-avalanche-diode (SPAD) to reconstruct object silhouettes through fog. However, the SPAD has extremely low spatial resolution (32x32 pixels), rendering it useless for practical applications [12]. By far, the most popular choice in industry are infrared camera systems. Since they rely on the longer infrared waves that are not affected by scattering as much as visible range waves, the resolution of the resulting image is poor and is not colored. Furthermore, infrared camera systems are costly, which a system's price ranging from \$10,000 to \$25,000 apiece.

Thus, we propose ClearCam, a realtime realtime dehazing software based on atmosphere scattering physics that directly removes the effects of haze from video input.

2. Background

2.1 Atmospheric Scattering Model

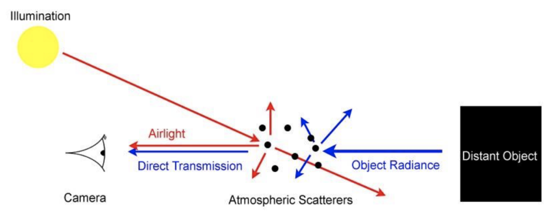


Figure 2. Degradation of Scene Radiance Through Haze

Mie scattering theory can be used to analyze light scattering in hazy conditions in which the haze particles are much larger in diameter than the wavelength of the light. The atmospheric scattering model, represented in Figure 2, applies Mie scattering theory to image sensing in hazy conditions. The spectral irradiance incident on the camera sensor can be decomposed into the direct transmission from the distant object and ambient light that scatters into camera sensor as airlight. The direct transmission of light is modeled to attenuate exponentially as a function of distance from the object. The irradiance incident on the camera sensor, $E(x)$, is given by

$$E(x) = J(x)t(x) + \alpha(1 - t(x)) \tag{1}$$

where x is a pixel location in the image, $J(x)$ is the object radiance, α is a coefficient of airlight, and $t(x)$ is the transmission. The transmission can be parameterized in terms of a coefficient of scattering, β, and depth [4, 3, 9]:

$$t(x) = e^{-\beta d(x)} \tag{2}$$

Optical thickness, defined as the product of β and depth, is inversely proportional to the transmission of light to the camera sensor [14]. Additionally, our dehazing system assumes constant scattering (β) throughout the scene. Section 4.4 details our experiments in generalizing to non-homogeneous haze.

2.2 Depth from Stereo Disparity

Stereo imaging allows us to extract the depth map of a hazy image. Two cameras are arranged such that they lie on the same surface and the lens are offset by T_x (see Figure 3). A point in the real world is projected onto both the left and right cameras. The coordinates of the projection on each of the cameras is shown below Figure 3, where f is the focal length of the cameras [5]. Notice the point on the left camera is to the right of the point on the right camera.

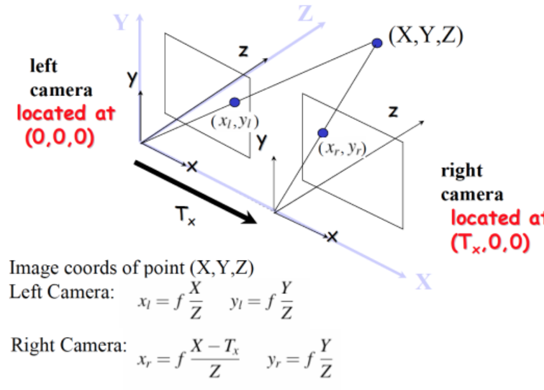

Y
Z
(X,Y,Z)
left camera
located at (0,0,0)
z
Y
(x_l, y_l)
Y
z
(x_r, y_r)
x
T_x
x
right camera
located at (T_x,0,0)
X
Image coords of point (X,Y,Z)
Left Camera: $x_l = f\frac{X}{Z}$ $y_l = f\frac{Y}{Z}$
Right Camera: $x_r = f\frac{X - T_x}{Z}$ $y_r = f\frac{Y}{Z}$

Figure 3. Geometric Representation of the Stereo Camera System

If we take the difference in x coordinates of the projection on the two cameras, and define that as the disparity, d, we can relate d to the depth of the projection.

$$d = x_l - x_r = f\frac{X}{Z} - (f\frac{X}{Z} - f\frac{T_x}{Z}) = f\frac{T_x}{Z} \tag{3}$$

Thus, if we take the difference in x coordinates for every pair of corresponding pixels in the left and right images, we can obtain a disparity map and use that to calculate the depth map. Figure 4 shows the disparity map for two stereo images that was obtained using a technique known as structured light [5].

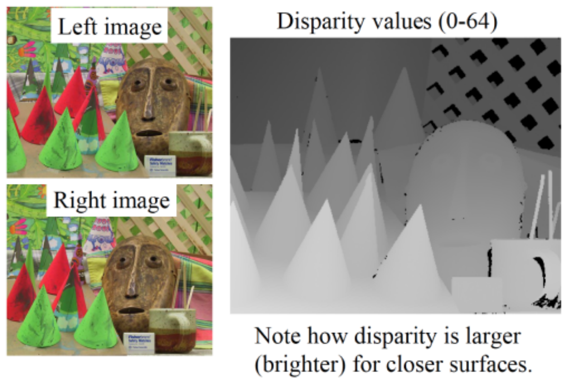

Left image
Right image
Disparity values (0-64)
Note how disparity is larger (brighter) for closer surfaces.

Figure 4. Disparity Map for a Stereo Image Set

2.3 Stereo Matching

To determine which pixels are corresponding, we first note that because the cameras are placed on a level surface, corresponding pixels must lie in the same row. Therefore, given a pixel in a row, we can search the same row of the other image for that pixel. To do so, we define a cost function, which measures how dissimilar two patches of two images are. The matching pixel will be the center of the patch with the least matching cost. An effective matching cost is

$$M(l,r) = 0.5(1 - NCC)$$

$$= 0.5\left(1 - \frac{\sum_{\sigma\varepsilon\Omega}(I^l_{p_{l+\sigma}} - \overline{I^l_{p_l}})\sum_{\sigma\varepsilon\Omega}(I^r_{p_{r+\sigma}} - \overline{I^r_{p_r}})}{\sqrt{\sum_{\sigma\varepsilon\Omega}(I^l_{p_{l+\sigma}} - \overline{I^l_{p_l}})^2)\sum_{\sigma\varepsilon\Omega}(I^r_{p_{r+\sigma}} - \overline{I^r_{p_r}})^2}}\right) \quad (4)$$

where NCC is the Normalized Cross Correlation, shown above in equation 4 [6].

There are two major issues that lead naive stereo matching to produce poor quality disparity maps. First, the naive approach does take into account that the matching between pixels is one-to-one, leading to several areas that should be matched to not be. Second, there is no way to account for occlusions, which are areas visible in one image but obscured in the other (e.g. along the edges of images).

3. Methods

The objective of our system is to determine α, β, and the depth map in equations (1) and (2) to extract the true scene radiance from the radiance on at camera sensor. We compute a disparity map with a novel post-processing algorithm under an equipolar constraint to determine the depth map. We implement a heuristic to solve for α and a Convolutional Neural Network to solve for β.

3.1 Efficient Depth Estimation in Fog

In this section, we outline our approach to obtaining a high quality disparity map in realtime given a set of foggy stereo images. We first assume no fog is present before adding that element in Section 3.1.3.

3.1.1 Disparity Space Image

We introduce the concept of the disparity space image (DSI). Given a pixel in a row, the cost function is evaluated with every pixel of the same row in the other image. If we consider all pixels in a row, we can construct a matrix, namely the DSI, such that the (i,j) entry represents the cost of matching pixel (r,i) of the right image to pixel (r,j) of the left image [5].

Define a path through the DSI to adhere to the following constraints: 1) if must start at $(0,0)$ and end at (N,M) where N and M are the height and width of the image, respectively and 2) the path cannot double back on itself. Define the cost of a path to the sum of the cost entries the path goes through. The minimum cost path yields the best matching solution for a row, as the global matching cost is minimized. The constraints of the path addresses both issues of naive matching. Define a successful match such that path moves from (i,j) to $(i+1,j+1)$ and an occlusion such that the path to $(i+1,j)$ or $(i,j+1)$. Note this guarantees a one-to-one correspondence between matching pixels.

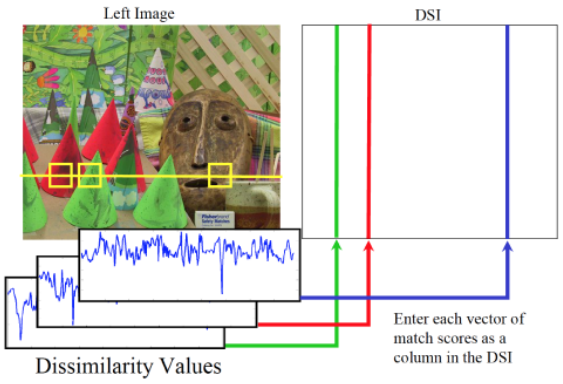


Figure 5. Construction of the Disparity Space Image

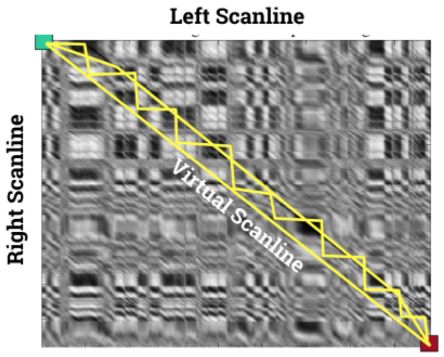


Figure 6. Possible Paths Through a Disparity Space Image

3.1.2 Dynamic Programming

The definition of the minimum cost path naturally leads to a three-state dynamic programming (DP) solution to computing it, proposed by Cox et al in 1998 [10].

$$\mathtt{C}(l,r) = \min \begin{cases} \mathtt{C}(l-1,r) & +\ \beta \\ \mathtt{C}(l-1,r-1) & +\ M(l,r) \\ \mathtt{C}(l,r-1) & +\ \beta \end{cases}$$

where β is the occlusion constant, which is added so matches are preferred. The matching pairs can be determined via backtracking the minimum cost path. However, three-state DP does not handle occlusions well, so our implementation involves a four-state DP model proposed by researchers at Microsoft in 2006, which distinguishes between a left match and a right match and defines a true match to be a consecutive left and right match [6].

$$\mathtt{C}_{Lo}[l,r] = \min \begin{cases} \mathtt{C}_{Lo}[l,r-1] & +\ \alpha \\ \mathtt{C}_{Lm}[l,r-1] & +\ \beta \\ \mathtt{C}_{Rm}[l,r-1] & +\ \beta \end{cases}$$

$$\mathtt{C}_{Lm}[l,r] = M(l,r) + \min \begin{cases} \mathtt{C}_{Lo}[l,r-1] & +\ \beta' \\ \mathtt{C}_{Lm}[l,r-1] & +\ \gamma \\ \mathtt{C}_{Rm}[l,r-1] & \\ \mathtt{C}_{Ro}[l,r-1] & +\ \beta' \end{cases}$$

The transitions for C_{Ro} and C_{Rm} can be deduced from symmetry.

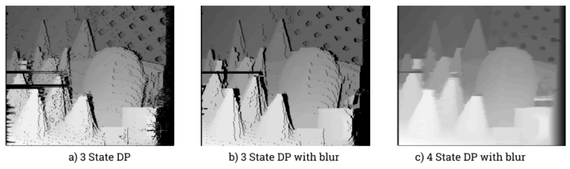


Figure 7. Comparison of three-state DP with four-state DP.

In addition to four-state DP, we also implement a blurring technique that reduces 'streaks' (see Figure 7 a)) that result from rows being computed independently. We first precompute the DSIs and then we arrange them in order of their row number. Then, we pass a Gaussian blur across the virtual scanline, propagating information across rows [6].

3.1.3 GPU Acceleration

We now address the realtime constraint of computing the disparity map. Because there are N rows, each with a DSI of size $N*M$, the complexity of computing the disparity map is $O(N^2M) \sim O(N^3)$. Notice that both computing the disparity values and the blurring are embarrassingly parallel. Thus we can leverage a Graphics Processing Unit (GPU) to reduce computation to $O(NM)$. Specifically, we used the Nvidia Ti-1060 6GB GPU.

However, computation of the disparity maps in parallel requires $O(N^3)$ memory, which may become an issue for larger images as we have a high multiplicative constant in the complexity expression (due to 4-states being used in the DP and memory required for blurring). To address this, notice that the path never falls below the virtual scanline because $j > i$ since the left image is shifted to the right of the right image. Moreover, observe that $j - i < K$ for all matching j and i by intuition as the displacement is small compared to the size of the image. To be safe, we set K to $\frac{1}{4}$ the width of the image. These observations allows us to cut the runtime and memory used by a factor of 8 on the GPU.

3.1.4 Disparity Map Refinement

Now we consider the addition of fog. Even with four-state DP, our base algorithm performs very poorly, as much information is lost and the cost function becomes less effective. We propose a fast $O(NM)$ post-processing algorithm that refines the disparity map, where N and M are the dimensions of the image. The post-processing algorithm is not parallelizable and run on the CPU, and it requires the same computation time as four-state DP run on the GPU.

The intuition behind the refinement algorithm is the majority of disparity values in the region (determined by the edge map) are correct; however there is also a large amount of error that cannot be accounted for with simple blurring or filling techniques; we must first determine what values are correct. Observe a region can either contain similar disparity values in the case the region is directly facing the camera, or contain a continuous sequence of disparity values in the case it is angled towards the camera (e.g. buildings on a street converging at the horizon).

To approach the problem, we first construct a histogram of the disparity values for each region. Notice that in both cases outlined above, the histogram will have a continuous range of values which the majority of the disparity values take. We can make a conservative estimate of this range (in case of the first case) by defining a sliding window length. Then, we can find the sliding window containing the maximum number of disparity values. This can be done quickly by precomputing the prefix sums of the histogram.

Then, we can take the maximum histogram disparity value to be the center of a 'hill' of disparity values. We can then find the valleys of the 'hill' and every pixel within the region that is not contained by the hill is set to either the lower or upper histogram disparity value, depending on which one is closer.

Algorithm 1 Proposed Refinement Algorithm

```
function REFINE(dismap, img)
    edge ← cannyEdgeDetection(img)
    edge ← dialate(edge)                                  ▷ Thicken edges to close gaps
    labels, components ← connectedComponents(edge)        ▷ Iterative Depth First Search
    labels ← labelEdges(labels, edge)    ▷ Ensure all pixels belong to a component
    assignPixelsValues(labels, components)
    for comp in component do
        SW = 20
        histogram ← getHistogram(comp)                    ▷ Sliding Window
        ps ← prefixSum(histogram)
        if ps[255]-ps[0] $<$ A then    ▷ Excludes occlusions, which have value of 0
            continue    ▷ Prevents components with small area from participating, reducing noise
        end if
        window = maxSWSum(histogram, ps, SW)
        mx = findMax(window)    ▷ Max SW Sum first prevents lone peaks from masking true max
        lb = findLeftValley(mx, R, N)
        rb = findRightValley(mx, R, N) ▷ bin count $<$ mx/R but greater than noise filter, N
        for pixel in comp do                              ▷ Remove incorrect matching
            if dismap.pixel.value not in [lb, rb] then
                dismap.pixel.value = nearest(lb, rb)
            end if
        end for
    end for
end function
```

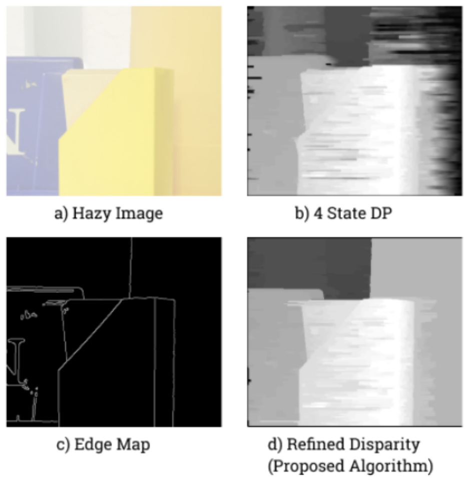


Figure 8. Our refinement algorithm performs fairly well.

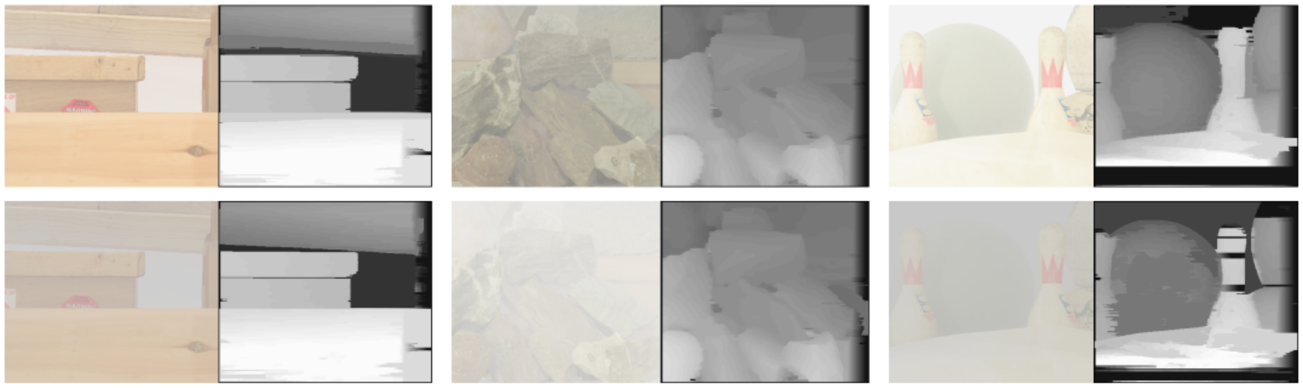


Figure 9. Examples of our algorithm's performance. The left column of each set of image contains the reference image under differing amounts of fog while the right column contains the computed disparity maps.

Parameter	Value
Patch Size	5x5
Gaussian Size	7x3
σ_a, $\sigma_{a'}$	3.0, 2.0
α	0.7
β, β'	1.0, 1.0
γ	0.25
SW	20
R	3
N	10

Table 1. Disparity Map Extraction Algorithm Parameters

3.2 Estimating Scattering Coefficient β

Existing single-image dehazing models [1, 4] estimate a transmission map or the clear scene directly, making little use of the atmospheric scattering model. With the depth map computed from stereo matching, we reduce the computation of the transmission map to solving for the scattering coefficient β.

3.2.1 Color Space Transformation

A method of conversion between RGB image data and irradiance maps must be established to use the atmospheric scattering model equation (1). In particular, the irradiance incident on the camera sensor must be computed from the input hazy image and the output clear image must be computed from the object radiance. The standard RGB (sRGB) color space used to encode images from $[0-255]$ establishes a non-linear relationship between the stored value and true intensity. On the other hand, a linear RGB (lRGB) system is needed to represent the irradiance maps which allows for the addition and multiplication operations in Equation 1. After normalizing sRGB values to the range $[0-1]$, we convert to lRGB with the operation:

$$C_L = \begin{cases} C_S/12.92, & \text{if } C_S \leq 0.04045 \\ \left(\frac{C_S+0.055}{1.055}\right)^{2.4}, & \text{otherwise} \end{cases} \tag{5}$$

The inverse function may be applied to convert from lRGB to sRGB [14].

3.2.2 Training Data

Natural haze datasets are cumbersome to create because data collection entails imaging a scene in low visibility and clear conditions, forcing the data collectors to wait until the weather conditions change. As a result, existing haze datasets largely consist of synthesized haze datasets from the atmospheric scattering model. In this paper, we assume that the model accurately simulates natural scattering. Due to the predictable method of haze synthesis, solving for the individual parameters of the equation is more effective than directly solving for a more global feature such as the transmission map or the clear image itself.

We use the Middlebury 2006 *small-resolution* dataset, which contains 7 horizontal image viewpoints of 21 indoor scenes. The focal length of the camera and the distance between the camera positions in the 7 orientations are given [13]. Selecting viewpoints 1 and 5 for depth triangulation, we use Equation 3 to solve for the depth map of the scene. We generate 80 hazy images per scene with random values of α and β in the range $\alpha \in (0.75, 1.0)$, $\beta \in (0, 1.5)$. Synthesized hazy images with a β value larger than 1.5 tend to produce unnatural optical thickness values on the Middlebury dataset. 10 128×128 pixel crops are taken per synthesized hazy image to augmented the CNN training data (see Figure 10 for diagram).

3.2.3 Model Architecture

The β regression model consists of Convolution, Max-Pooling, and non-linear activation layers (diagrammed in Figure 11). The model receives a $128 \times 128 \times 3$ sRGB hazy image crop. The pixel intensities in the image are normalized in the range $[0-1]$. The network consists of five cycles of Convolutions and ReLU activation followed by Max-Pooling. Each 2D Convolution has a filter size of 3×3, which was chosen to limit the number of trainable parameters of the model while maximizing the depth of the network. There are 32 filters for each Convolution for retaining important features before the dimensions are scaled down in the Max-Pooling layer. The

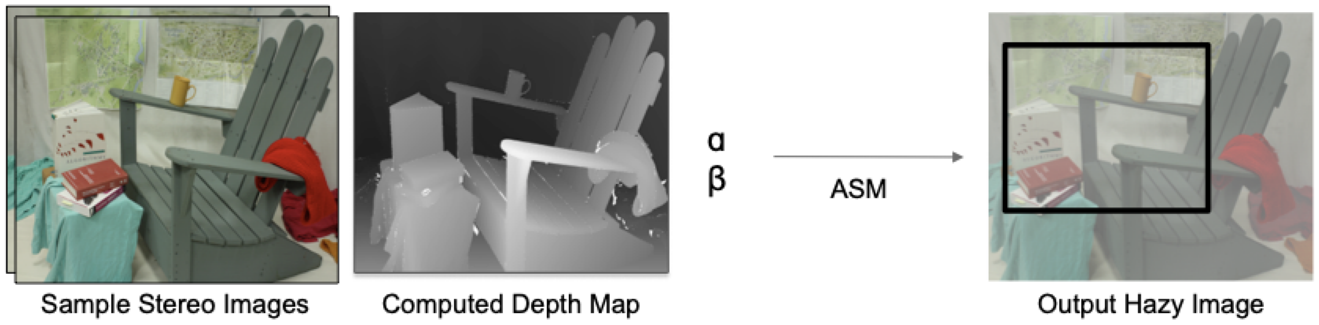
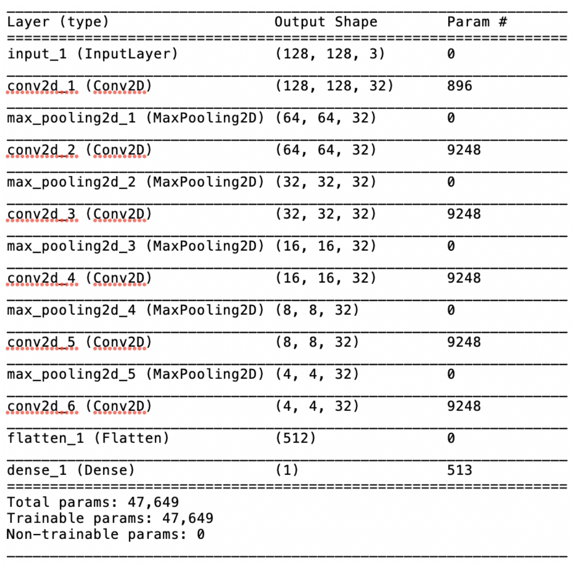


Figure 10. Haze Synthesis in Middlebury 2006 Stereo Dataset



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (128, 128, 3)             0
_________________________________________________________________
conv2d_1 (Conv2D)            (128, 128, 32)            896
_________________________________________________________________
max_pooling2d_1 (MaxPooling2D) (64, 64, 32)            0
_________________________________________________________________
conv2d_2 (Conv2D)            (64, 64, 32)              9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2D) (32, 32, 32)            0
_________________________________________________________________
conv2d_3 (Conv2D)            (32, 32, 32)              9248
_________________________________________________________________
max_pooling2d_3 (MaxPooling2D) (16, 16, 32)            0
_________________________________________________________________
conv2d_4 (Conv2D)            (16, 16, 32)              9248
_________________________________________________________________
max_pooling2d_4 (MaxPooling2D) (8, 8, 32)              0
_________________________________________________________________
conv2d_5 (Conv2D)            (8, 8, 32)                9248
_________________________________________________________________
max_pooling2d_5 (MaxPooling2D) (4, 4, 32)              0
_________________________________________________________________
conv2d_6 (Conv2D)            (4, 4, 32)                9248
_________________________________________________________________
flatten_1 (Flatten)          (512)                     0
_________________________________________________________________
dense_1 (Dense)              (1)                       513
=================================================================
Total params: 47,649
Trainable params: 47,649
Non-trainable params: 0
_________________________________________________________________
```

Figure 11. β Model Architecture

network tensor is padded by one pixel on each side before each Convolution operation to prevent the operation from reducing the width and height of the network tensor. Since a single Convolution layer precedes each Max-Pooling layer and the amount of information lost increases in a quadratic manner with increases in the receptive field of the Max-Pooling, each Max-Pooling operation has a receptive field of 2 and a stride of 2 [7].

Following the cycles of Convolution and Max-Pooling, the network tensor is flattened into a vector of length 512 and inputted into a dense layer which condenses the vector into a single scalar prediction of β. A ReLU non-linearity is then applied to prevent negative predictions. Alternative activation functions such as Sigmoid and the hyperbolic tangent function would prevent the model from predicting the β coefficient for images with a ground truth value greater than 1.0 because of their clipped ranges.

3.2.4 Implementation and Training

Each hazy image crop is compressed and split between the training, validation, and test sets with a 2/3, 1/6, and 1/6 split respectively. The network, which is implemented in

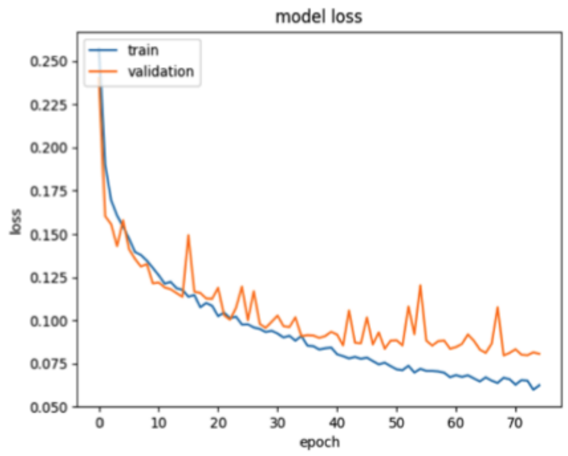


Figure 12. Training β Regression Model on Synthesized Haze with MAE loss

Keras with Tensorflow backend, contains a generator which opens each batch of compressed image and maps it to its corresponding β value. The generator frees memory on the Nvidia TI-1060 6GB GPU used to train the model, allowing for the proposed data augmentation of 10 crops per scene. We use a batch size of 32, meaning that 32 training examples are read and forward-propagated before updating the parameters. The model is trained with a Mean Absolute Error (MAE) as the loss function and an Adam optimizer to update the parameters of the CNN. Figure 12 depicts the MAE of the train and validation set over 75 epochs, at which point the validation loss plateaus at a lower rate than the train loss, indicating that the model is over fitting. The model's predictions had a MAE of 0.081 on the test set.

3.3 Estimating Airlight Parameter α

We adopt a heuristic proposed by Cai et al. to solve for the α parameter. In the atmospheric scattering model equation 1, we make use of the transmission map which can be computed with Equation 2 after solving for the depth map and β. At points in the image which have a particularly low transmission (or great depth), we make the approximation that $E(x) = \alpha$.

After solving for the transmission map, a naive approach to solving for α is to locate the point in the image which has

Model Type	Test MAE (75 Epochs)
3 Channel	0.081
4 Channel	0.171
Depth Buffer	0.183

Table 2. Performance of Optimized Models

the minimum transmission and equate α to the linear RGB value of the hazy scene at the specified point. However, to ensure the robustness of the approximation, Cai et al. propose

$$\alpha = \max_{y \in t(x) \leq t_0} E(y) \tag{6}$$

, effectively sampling all points which have a transmission less than a tunable parameter t_0 and taking the maximum pixel irradiance value incident on the camera sensor. The heuristic achieves optimal performance when the hazy image contains only ambient light where the transmission falls below t_0 and contains both for transmission values above t_0 [4].

4. Experiments

4.1 Optimizations to β-estimation

Hazy environments with high optical thickness values may have a relatively small value of β if the scene is sufficiently deep. Likewise, a shallow scene will need a relatively large value of β to produce a fixed optical thickness. As a result of the implicit relationship between β and depth, the β regression CNN was modified to input depth information.
In the first optimization, we concatenate the depth map of the scene to the raw hazy image data to the model. In particular, the depth maps provided from the four-state DP stereo matching and refinement algorithm were given as a fourth channel in the model input. The remaining layers were transferred from the 3 channel model and retrained.

In the second optimization, which was not applied in conjunction with the first, we append the average depth of the scene to a fully connected layer of the neural network. The cycles of 2D Convolutions and Max-Pooling layers are transferred from the 3 channel model and a 9 node dense layer is added to the CNN after the network tensor is flattened. The average depth is appended to form a 10 node dense layer, which is then condensed to a scalar prediction.
Table 2 shows the MAE results on the test set for each of the three model types over 75 epochs. The 3 channel model achieved the lowest error, indicating that the depth information was not useful for the synthesized Middlebury 2006 images.

4.2 Evaluation on Synthetic Data

We first discuss the runtime performance of our dehazing process. Table 3 shows the time in seconds each component of the process took. Notice that although parameter fitting seems like a bottle neck compared to the other components, we can only run it once every second; even if the camera is moving, over a short period of time, the parameters are likely to not vary by any noticeable amount. Thus, we are able to achieve a realtime frames-per-second (FPS).

Process	Time
Stereo Algorithm	0.0420s
Parameter Fitting	0.104s
Refining Time	0.0419s
Total FPS	10.679

Table 3. Evaluation Times

We use two metrics to evaluate the quality of the output of of our dehazing process, the Mean Absolute Error (MAE), and the Structural Similarity Measurement Index (SSIM).

$$MAE = \frac{1}{N}\sum_{1}^{N} |y_i - \bar{y}| \tag{7}$$

$$SSIM = \frac{(2\mu_u \mu_y + C_1) + (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{8}$$

Statistic)	Value
Hazy MAE	52.992
Hazy SSIM	0.628
Dehazed MAE	35.545
Dehazed SSIM	0.950

Table 4. Evaluation Metrics Values

To test out dehazing process, we randomly synthesized a test set of 380 images from the Middlebury dataset under varying conditions of scattering. From Table 3, we see that the MAE decreases, indicating the images have become more similar after the dehazing process, but the new MAE is not close to zero, indicating there is still some color distortion. The large jump in SSIM, however, demonstrates the robustness of our algorithm, indicating that the structure of the dehazed image compared to the reference is very similar. This implies that most of the image features, which might be used in other processes such as object detection, become noticeable.

4.3 β Estimation in Outdoor HazeRD Data

Results from 3-Channel Model The Synthesized Middlebury Dataset was limited to indoor data, making the dataset an imperfect model for natural haze. To test the robustness of our 3 channel model on outdoor data, we HazeRD dataset [14], which contains 15 outdoor scenes with complete depth maps. The depth at atmospheric regions is set to twice the maximum depth in the scene. The hazy images are augmented as in Section 3.2.2 with 10 hazy crops per scene. Figure 15 shows the training and validation MAE statistics over a 150 epochs. Even with data augmentation, the model lacks sufficient cases to achieve a similar error as with the Middlebury dataset. The model reaches an MAE statistic of 5.15 on the test data, which is relatively high for the HazeRD data, which has a β range from $[4.0 - 72.0]$.

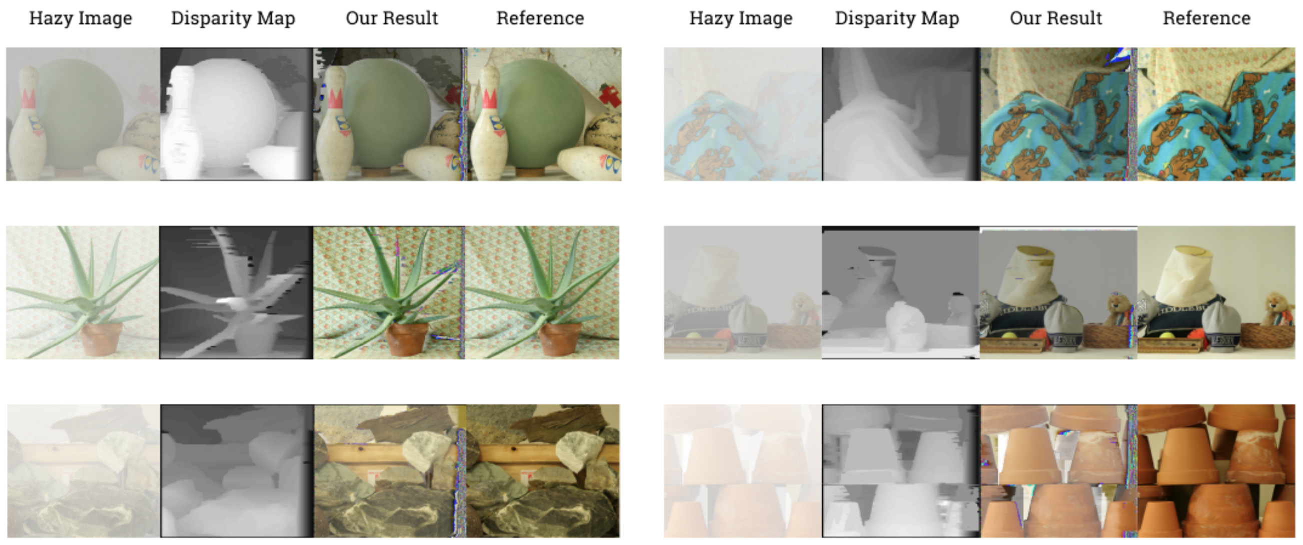


Figure 13. Results of Dehazing Algorithm

Range of Illumination Prior We detail an alternative approach towards β regression in the HazeRD dataset which relies on the observation that scenes with higher scattering are more homogeneous in intensity [8]. For example, in Figure 14, the color contrast in the hazy images decreases along each row in the direction of increasing β. We define the Range of Illumination of a hazy image as

$$RI = max(R_x + G_x + B_x) - min(R_x + G_x + B_x) \tag{9}$$

and demonstrate that the statistic is correlated with the β in Figure 16. In the scatter plot, we observe that the synthesized hazy images for each individual scene form a line. Since deeper scenes may reach a lower *RI* statistic with a given value of β, we expect deeper scenes to have a more level slope. Figure 17 demonstrates that the expected correlation holds true but with little confidence in shallow scenes. Further, with a limited sample size of 15 outdoor points, there is no statistically significant correlation that can be drawn.

4.4 Realistic Fog Synthesis

Insofar, we have assumed a constant scattering coefficient β throughout the scene. However, when imaging through natural haze, the scattering coefficient is not necessarily constant [14]. We introduce a method of creating non-homogeneous β maps in which the β value at each pixel in the image is dependant on the superposition of randomized Gaussian filters (see Figure 18). The β map is centered in a frame where Gaussian filters may be placed as either sources or sinks of scattering. The parameters for the noise synthesis include the size of the larger frame, the maximum number of Gaussian patches, the size of each Gaussian filter, the standard deviation of the Gaussian function, and a scalar to amplify the Gaussian distribution. Figure 19 applies the Gaussian noise to a sample image from the Middlebury 2006 dataset.

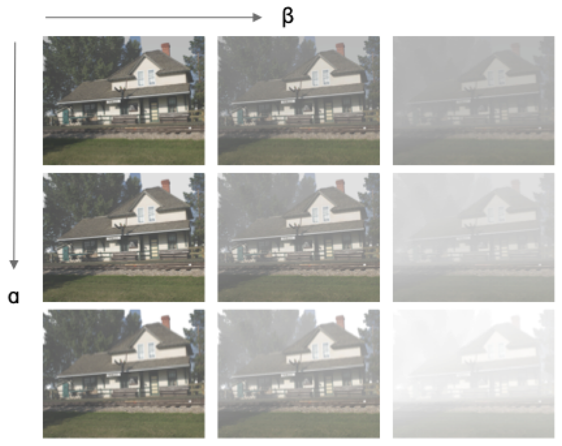


Figure 14. Synthesized Hazy Images from HazeRD Dataset

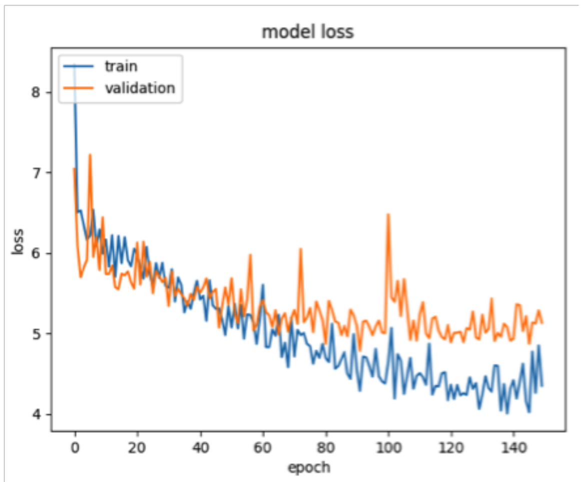


Figure 15. Training 3 Channel β Regression Model on HazeRD data with MAE loss

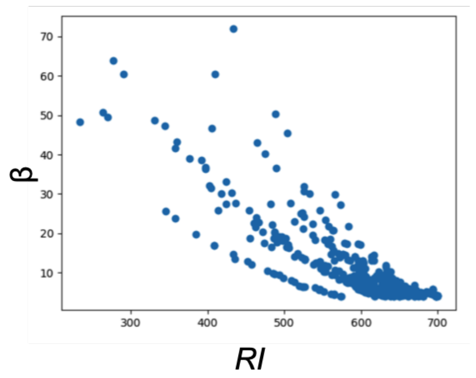


Figure 16. Correlation between Range of Illumination Statistic and β value for Synthesized Hazy Images in HazeRD

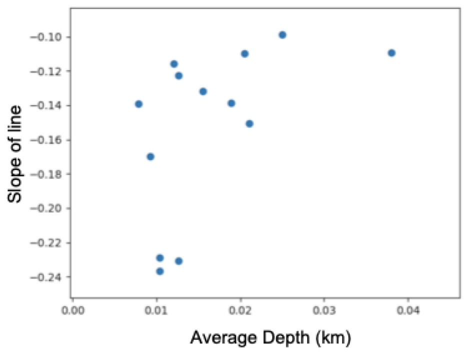


Figure 17. Correlation between Average Depth (km) and Slope of RI vs β scatter plot

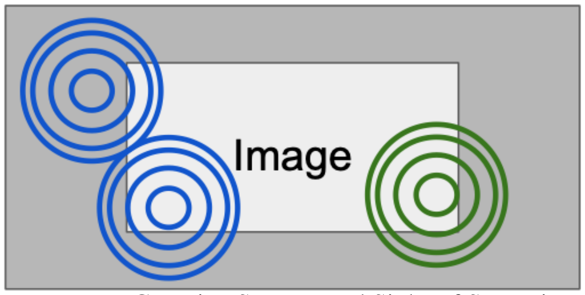


Figure 18. Gaussian Sources and Sinks of Scattering

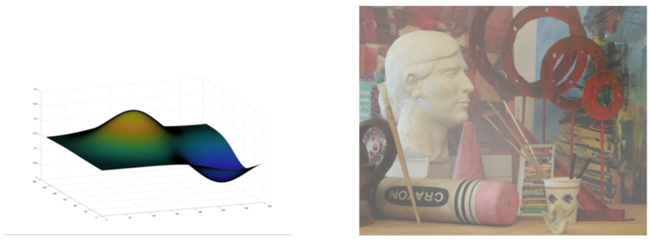


Figure 19. Gaussian Noise Synthesis: (a) 3D β map; (b) Corresponding Heterogeneous Hazy Image

5. Conclusions and Open Questions

We have successfully constructed an end-to-end approach to image and video dehazing. Based on our runtime performance, our software functions in real-time, making it applicable to platforms such as security systems or autonomous vehicles. Another application is driver assistant cameras. Based on qualitative evaluation, our dehazing process exceeds human vision in fog and can be used as a substitute for the solutions discussed in Section 1.

For future work, we would like to collect real stereo foggy/clear images. We would also like to explore single-shot depth estimation algorithms as they are fast and do not require a stereo camera system. The algorithms also can be improved in many ways; for instance, the refinement algorithms may contain many holes for certain cases.

6. Acknowledgments

We would like to thank our Computer System Research Lab Director, Mr. White, for his mentoring of the project. We would also like to thank Dr. Zacharias, Dr. Gabor, and Dr. Torbert, the other directors for helping review our project. Finally, we would like to thank our parents for helping us buy and transport materials.

7. Appendix

We now present the derivation of the Atmospheric Scattering Model. We begin by defining the scattering parameter

$$\beta = \frac{k}{\lambda^{\gamma}}$$

where k is a constant, λ is the wavelength of light, and γ accounts for the size of the scattering medium ($\gamma \approx 0$ for large particles and 4 for air). This is known as Raleigh's law [11].

The incoming signal can be decomposed into attenuated light (irradiance) from the object we are trying to image and airlight (noise due to scattering). Intuitively, irradiance E decreases as distance from the object d increases as there is more scattering. In 1729, Bouguer proposed [9]

$$dE = -\beta E dx \rightarrow \int_{E_0}^{E} \frac{dE}{E} = \int_0^d -\beta dx$$

$$E = E_0 e^{-\beta d}$$

where E_0 is the irradiance at $d = 0$. However, Bouguer assumed that light can only take the shape of a column. Allard (1876) remedied this by incorporating the inverse square law for diverging beams [2].

$$E = \frac{\Phi_0 e^{-\beta d}}{4\pi d^2} = \frac{I_0 e^{-\beta d}}{d^2}$$

where Φ_0 is the radiant flux emitted by the object and I_0 radiant intensity.

Now we consider the effect of airlight E'. Airlight due to the scattering of environmental illumination by the atmosphere, increases the brightness of the image as as d increases. Koschmieder (1924) proposed the change in radiant intensity of atmospheric scattering is proportional the change in view volume [11].

$$dI = K\beta dV$$
$$dV = x^2 d\omega dx$$

where the constant K accounts for the nature of illumination and $d\omega$ is the solid angle of the view.

Each dV on the path from the object to the camera is a source of airlight and produces an irradiance which can be quantified using Alllard's Law.

$$dE' = \frac{e^{-\beta x} dI}{x^2} \rightarrow \int_0^{E'} dE' = \int_0^d K d\omega \beta e^{-\beta x} dx$$

$$E' = \alpha(1 - e^{\beta d})$$

where $\alpha = Kd\omega$. The total transmission can be written as the sum of irradiance and airlight [11].

$$E_{total} = E + E'$$
$$E_{total} = Jt + \alpha(1 - t)$$
$$t = e^{-\beta d}, \quad J = \frac{I_0}{d^2}$$

References

[1] Cosmin Ancuti, Codruta O. Ancuti, and Radu Timofte. "NTIRE 2018 Challenge on Image Dehazing: Methods and Results". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. June 2018.

[2] Alec Bennett. "Introduction to atmospheric visibility estimation". In: (2012).

[3] Guoling Bi et al. "Image Dehazing Based on Accurate Estimation of Transmission in the Atmospheric Scattering Model". In: *IEEE Photonics Journal* 9 (2017).

[4] Bolun Cai et al. "DehazeNet: An End-to-End System for Single Image Haze Removal". In: *CoRR* abs/1601.07661 (2016). arXiv: 1601.07661. URL: http://arxiv.org/abs/1601.07661.

[5] Robert Collins. *Lecture 09: Stereo Algorithms*. University Lecture.

[6] A. Criminisi et al. "Efficient Dense Stereo with Occlusions for New View-Synthesis by Four-State Dynamic Programming". In: *Microsoft Vision Research* ().

[7] Chao Dong et al. "Image Super-Resolution Using Deep Convolutional Networks". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 38.2 (Feb. 2016), pp. 295–307. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2015.2439281. URL: http://dx.doi.org/10.1109/TPAMI.2015.2439281.

[8] Kaiming He, Jian Sun, and Xiaoou Tang. "Single Image Haze Removal Using Dark Channel Prior". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 33.12 (Dec. 2011), pp. 2341–2353. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2010.168. URL: http://dx.doi.org/10.1109/TPAMI.2010.168.

[9] *Introduction to Light Scattering: An Imaging Sciences Perspective*. University Lecture.

[10] Ingemar J. Cox et al. "A Maximum Likelihood Stereo Algorithm". In: *Computer Vision and Image Understanding* 63 (May 1996), pp. 542–567. DOI: 10.1006/cviu.1996.0040.

[11] Srinivasa G. Narasimhan and Shree K. Nayar. "Vision and the Atmosphere". In: *International Journal of Computer Vision* 48.3 (July 2002), pp. 233–254. ISSN: 1573-1405. DOI: 10.1023/A:1016328200723. URL: https://doi.org/10.1023/A:1016328200723.

[12] G. Satat, M. Tancik, and R. Raskarr. "Towards Photography Through Realistic Fog". In: *IEEE International Conference on Computational Photography (ICCP)* ().

[13] Daniel Scharstein et al. "High-Resolution Stereo Datasets with Subpixel-Accurate Ground Truth". In: *German Conference on Pattern Recognition* (2014).

[14] Y Zhang, L Ding, and G Sharma. "HazeRD: an outdoor scene dataset and benchmark for Single image dehazing". In: *Proc. IEEE Intl. Conf. Image Proc.* Sept. 1, 2017, pp. 3205–3209. DOI: 10.1109/ICIP.2017.8296874. URL: http://www.ece.rochester.edu/~gsharma/papers/Zhang_ICIP2017_HazeRD.pdf,%20paper%20https://labsites.rochester.edu/gsharma/research/computer-vision/hazerd/,%20project%20page%20and%20dataset. published.