Team Number:	apmcm24212229
Problem Chosen:	A

2024 APMCM summary sheet

In ocean exploration, high-quality underwater images are crucial for deep-sea surveys and resource investigations. The complex underwater environment leads to light absorption and scattering, degrading image quality. This research classifies images using methods based on mean, variance, histogram differences, and Fourier transformation. Evaluation metrics are calculated via histogram equalization. A complex-scenario enhancement model is built, and practical suggestions are provided for enhancing image quality.

Question 1: Apply similar statistical analysis techniques to process the underwater images in Attachment 1. Classify them into color cast, low-light, and blur types based on certain characteristics. Dynamically adjust thresholds to reduce misjudgment. Store the results and explain the classification basis for further processing.

Question 2: Based on the degradation types from Problem 1, combine relevant models and images to construct an underwater image degradation model. Analyze the effects of light propagation factors on image quality and their roles in different degradations to understand the mechanism and support enhancement methods.

Question 3: According to the model in Problem 2, propose an enhancement for single-degradation scenarios. Process test images in Attachment 2, calculate metrics to validate. The enhanced images show improvements, and the results are recorded to provide a solution for single-degradation image enhancement.

Question 4: Addressing existing models' adaptability problems, propose a complex-scenario underwater image enhancement model. Integrate techniques via uwredcomp considering light propagation. Test with Attachment 2 images and calculate metrics. Results confirm its effectiveness, providing a new way for such enhancement.

Question 5: Compare specific and complex scenario enhancement techniques. Specific ones target single degradations precisely with low cost, while complex ones handle multiple degradations with strong ability. In practice, choose techniques based on the scene, optimize parameters, consider fusion, and build a mechanism to adapt and achieve the best effect for guiding underwater visual enhancement.

Keywords: Statistical differences; Fourier transformation; Histogram equalization; Reconstruction based on channel weighting; Global and local contrast enhancement

Contents

1. Introduction **1**
1.1 Background 1
1.2 Restatement of Problem 2
2. Problem Analysis **3**
2.1 Analysis of Problem One 3
2.2 Analysis of Problem Two 3
2.3 Analysis of Problem Three 4
2.4 Analysis of Problem Four 4
2.5 Analysis of Problem Five 5
3. Models Hypothesis **5**
3.1 Assumption 1 5
3.2 Assumption 2 5
3.3 Assumption 3 6
3.4 Assumption 4 6
4. Symbol Explanation **8**
5. Model Establishment and Solution of Problem 1 . . . . **9**
5.1 Image Data Preprocessing 9
5.2 Color Cast Processing 10
5.2.1 *Color Channel Histogram Computation* 10
5.2.2 *Color Channel Mean Value Computation* 10
5.2.3 *Color channel variance calculation* 11
5.2.4 *Judgment Mechanism* 11
5.3 Low light Processing 13
5.3.1 *Grayscale Image Conversion* 13
5.3.2 *Grayscale Image Mean Value calculation* 13
5.3.3 *Judgment Mechanism* 14
5.4 Blur Processing 14
5.4.1 *Image grayscaling* 14
5.4.2 *Fourier Transformation and Spectrum Analysis* 14
5.4.3 *Judgment Mechanism* 16
5.5 Result 16

6. Model Establishment and Solution of Problem 2. 16
6 . 1 Color Cast Degeneration Model 16
6 . 2 Low light Degeneration Model 17
6 . 3 Blur Degeneration Model 17
6 . 4 Causes, Similarities and Differences 17
7. Model Establishment and Solution of Problem 3. 18
7 . 1 Image Acquisition and Preliminary Configuration 18
7 . 2 Computation of the Mean Value for Each Channel 19
7 . 3 Histogram-Related Processing for Each Channel 19
7 . 3 . 1 *Initiation of Pixel Gray-scale Statistical Operations* 19
7 . 3 . 2 *Construction of Histograms* 20
7 . 3 . 3 *Compute the Density of the Gray Scale Distribution* 20
7 . 3 . 4 *Calculation of the Distribution of Cumulative Histograms* 20
7 . 3 . 5 *Rounding and Normalization of Cumulative Distribution* 21
7 . 4 Equalization Processing of Each Channel 21
7 . 5 Adjustment and Display of the Processed Image 21
8. Model Establishment and Solution of Problem 4. 21
8 . 1 Image Data Manipulation 21
8 . 2 Principles of Image Enhancement Algorithms 22
8 . 3 Calculate quality assessment indicators 23
8 . 3 . 1 *Calculate the UIQM metric* 23
8 . 3 . 2 *Calculate the PSNR metric* 24
8 . 3 . 3 *Calculate the UCIQE metric* 24
8 . 4 Result 24
9. Model Establishment and Solution of Problem 5. 25
9 . 1 Practical Applications 25
10. References 27
11. Appendix 29

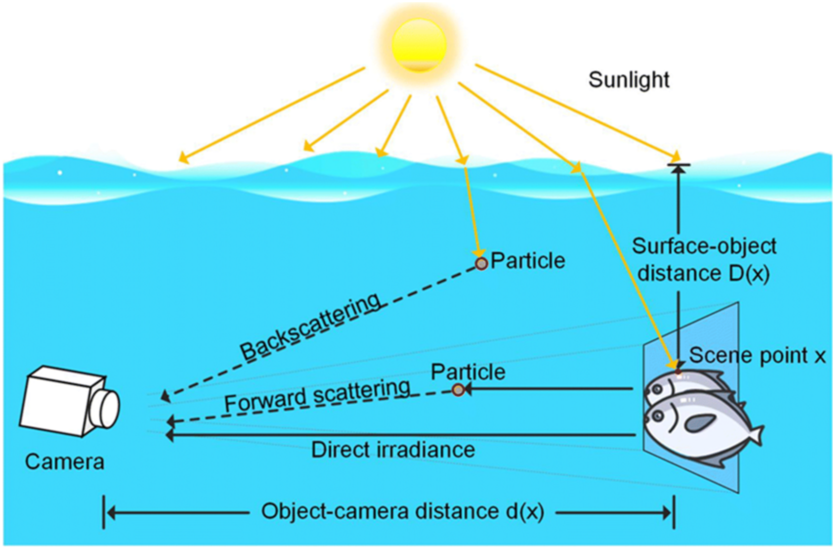

Figure 1 Schematic diagram of the principle of underwater image degradation

I. Introduction

1.1 Background

For ocean exploration, clear and high-quality underwater images are crucial for deep-sea topography surveying and seabed resource investigation. However, the complex underwater environment causes a severe degradation in image quality due to phenomena such as the absorption and scattering of light in water.

The complex optical properties of water, such as the scattering caused by suspended particles and the absorption, result in a remarkable degradation of underwater images. This degradation is exhibited as color casts, artifacts, and details that become blurred, which obstructs the precise interpretation and analysis.

Owing to the complexity of underwater scenes, existing methods often fail to handle all situations. Consequently, underwater scene enhancement algorithms customized for complex scenarios are of crucial importance for subsequent underwater vision tasks. To estimate the degree of degradation of underwater images in different scenarios and provide targeted enhancement methods, it is necessary to conduct in-depth research on the degradation mechanism of underwater images, establish degradation models, and propose effective enhancement algorithms.

Meanwhile, comparing the performance of different enhancement techniques and providing feasible suggestions for underwater vision enhancement in practical applications is also one of the key focuses of the research.

1.2 Restatement of Problem

Considering the background information and restricted conditions in the problem statement,the following problems should be solved: **Question 1**: Please deploy image statistical analysis methodologies that are akin to those cited to execute a multi-pronged analysis on the underwater image supplied in Attachment 1. Dissect the image from Attachment 1 into three classifications: color cast, low light, and blur. Thereafter, input the filenames into the three corresponding loci in the ”Answer.xls” attachment. Moreover, explicate the rationales underpinning this classification.

Question 2: In accordance with the degradation types posited in Question 1, by leveraging the underwater imaging model furnished within the problem, formulate an underwater scene image degradation model in conjunction with the appended images. Examine the degradation causes of underwater images procured from diverse scenes (encompassing but not restricted to color cast, low light, and so forth). Scrutinize the similarities and disparities among these degradation models, for instance, by classifying them from the vantage points of color, illumination, sharpness, and other relevant aspects.

Question 3: Propose an underwater image enhancement method specifically designed for a single scene such as color cast, blur, or low light, which is based on the underwater scene image degradation model established in Question 2. Then, validate this proposed enhancement method using the image data provided in the attachment. Include both the enhanced results of the test images from Attachment 2 and their corresponding evaluation metrics. Calculate and present evaluation metrics like PSNR, UCIQE, UIQM, and others for the output images, and fill these values into the table of Attachment 1 results provided in ”Answer.xls”.

Question 4: Existing underwater image enhancement models have varying modeling adaptability. Based on the prior question and attached images, propose a complex-scenario-tailored underwater image enhancement model (e.g., a non-physical model). It should address underwater image degradation in diverse complex scenes. Present the enhanced test image results from Attachment 2 and their evaluation metrics .Calculate and record PSNR, UCIQE, UIQM, etc. of the output images in the Attachment 2 results table in ”Answer.xls”.

Question 5: Compare specific-scenario enhancement techniques with a complex-scenario one, and give feasibility suggestions for underwater visual enhancement in practice.

II. Problem Analysis

2.1 Analysis of Problem One

In Problem 1, specific image statistical analysis techniques are employed to conduct processing and analysis on underwater images. The images within Annex 1 are precisely categorized into three classes, color cast, low light, and blur, in accordance with the characteristics of the images. Regarding the determination of color cast, the mean difference threshold and variance ratio threshold are adaptively adjusted in a dynamic manner, taking into account the overall brightness, contrast, and other attributes of the image. In the assessment of low light conditions, the low light threshold is calibrated by integrating factors such as the color saturation of the image. For the evaluation of blur, the fuzzy threshold is modified according to the resolution and other relevant factors of the image. Through such an approach, the threshold settings can be more congruent with the actual circumstances of diverse seabed images, thereby minimizing the occurrence of misjudgment instances resulting from irrational threshold configurations.

2.2 Analysis of Problem Two

In problem 2, with the objective of resolving the image degradation issue that is elicited by light absorption and scattering in complex underwater environments and gives rise to image degradation phenomenons such as colour aberration, low illumination, and blurring, the construction of an underwater scene image degradation model is of paramount importance. Moreover, upon analyzing the degradation-inducing factors, it is evident that light propagation loss, forward scattering, backward scattering, and water turbidity exert profound and far-reaching influences. More precisely, light propagation loss is the causation of low illumination, while forward and backward scattering are the instigators of blurring, and turbidity influences light attenuation, thereby resulting in colour aberration and exacerbating the low illumination and blurring effects. By formulating a model to undertake an elaborate and profound analysis of the degradation origins and exploring the similarities and differences among various degradation models, it is conducive to the implementation of bespoke enhancement methodologies to enhance the quality of underwater images.

2.3 Analysis of Problem Three

In Problem 3, the image is initially read and its RGB channels are segregated, followed by the computation of channel mean values to underpin subsequent adjustments. Subsequently, histogram processing is executed for each channel, entailing the tallying of pixel gray levels, calculation of distribution density and cumulative distribution, and normalization. Subsequently, histogram equalization is performed to render the gray level distribution of each channel more rational, thereby augmenting contrast. Thereafter, the equalization outcomes are further refined by integrating channel mean values. In the event of a color image post-channel processing, the data type is converted, and the channel data is recalibrated by computing the average gray value predicated on each channel mean. Eventually, the image is transformed into an appropriate format for display, thereby attaining the optimization and enhancement of pivotal quality elements such as image color balance and contrast via this sequential operation chain.

2.4 Analysis of Problem Four

In view of the current situation where the modeling adaptability of existing underwater image enhancement models varies across different scenes, it is necessary to propose the construction of a novel underwater image enhancement model capable of handling complex scenarios, along with a series of image processing and quality assessment procedures. Accordingly, this paper presents a methodology for optimizing the processing of RGB images captured underwater. Through an analysis of the image quality issues resulting from the wavelength-dependent attenuation and diffusion effects of light in the underwater environment, the uwredcomp function is designed and implemented. This function combines the techniques of channel reconstruction, noise reduction, as well as global and local contrast enhancement, thereby effectively enhancing the visual quality of underwater images and furnishing more favorable image data for subsequent applications such as underwater image analysis.

2.5 Analysis of Problem Five

III. Models Hypothesis

3.1 Assumption 1

In this study, hypotheses for classifying underwater image damage are presented. In color cast judgment, if the mean differences of RGB channels exceed meanDiffThreshold, variance ratios surpass varRatioThreshold, or histogram peak color intensity differences are significant (over 30), color cast is suspected. For low-light judgment, a grayscale image with a mean below lowLightThreshold or a high proportion of pixels in the 0 - 50 low-brightness range (beyond histogramThreshold) is considered low-light. Blur is determined when the Fourier-transformed high-frequency component to total energy ratio is less than blurThreshold. The image damage category is then identified by comparing the normalized comprehensive scores of color cast, low light, and blur.

Explanation: The underwater environment induces image degradation. Color cast, due to water's differential light absorption and scattering, alters channel means, variances, and histogram peaks. A multi-factorial consideration ensures accurate detection. Low light darkens images and increases low-brightness pixels; mean and histogram evaluation aids illumination assessment. Blurring, from detail loss, shows as a reduced high-frequency component proportion, a metric for severity. Comparing overall scores reveals the dominant damage, enabling targeted fixes. This framework underpins underwater image processing. Yet, thresholds need optimizing per environment and equipment, and future research could seek smarter, adaptive methods.

3.2 Assumption 2

It is hypothesized that water turbidity exerts a differential impact on the attenuation of diverse light colors. The ambient light is postulated to possess an exponential correlation with turbidity. It is further assumed that the fuzzy kernel exhibits an exponential relationship with turbidity.

Explanation: Water turbidity manifests disparate effects on the attenuation of various light colors, as the suspended particles within water exhibit distinct scattering and absorption characteristics for different wavelengths of light. Practical observations also corroborate that alterations in turbidity can precipitate image color deviation, and this supposition elucidates the genesis of color aberration. The exponential correlation between ambient light and turbidity stems from the fact that an increment in turbidity accelerates light propagation while intensifying attenuation, which aligns with the attributes of light propagation and is substantiated by prior research, thereby endowing the model with scientific validity and reliability. The exponential relationship between the fuzzy kernel and turbidity arises from the exponential augmentation of the forward scattering effect due to the rise in turbidity, which in turn engenders an increase in image blurring. Concurrently, the exponential relationship is mathematically concise and versatile, facilitating the management of image degradation across diverse scenarios and simulations. These postulations can render the model more congruent with the actual underwater imaging circumstances and efficaciously explicate and address the issue of image degradation.

3.3 Assumption 3

In the color correction segment, compute the mean values of the red, green, and blue channels within the enhanced image. Subsequently, it derives the adjustment coefficient for each channel based on the average of these three channel mean values. Then, corresponding adjustments are made to each channel. This process hinges on the grayscale world assumption, which posits that in a natural scene image, the average grayscale values of the red, green, and blue channels ought to be approximately equal. Consequently, when the image exhibits color deviation or other such anomalies, color correction can be achieved by adjusting each channel such that its mean value approaches a common average grayscale value.

Explanation: The adoption of the grayscale world assumption is attributed to the fact that in numerous real-scene images, lighting conditions and object reflective properties can induce color deviation, resulting in disparities among the average grayscale values of distinct color channels. This assumption offers a relatively straightforward and often efficacious approach for estimating the genuine color of the image. Through adjusting each channel to equilibrate their average values, it can partially rectify color deviation and recuperate the image's natural coloration.

3.4 Assumption 4

In view of the current situation where the modeling adaptability of existing underwater image enhancement models varies across different scenes, it is necessary to propose the construction of a novel underwater image enhancement model capable of handling complex scenarios, along with a series of image processing and quality assessment procedures. Accordingly, this paper presents a methodology for optimizing the processing of RGB images captured underwater. Through an analysis of the image quality issues resulting from the wavelength-dependent attenuation and diffusion effects of light in the underwater environment, the uwredcomp function is designed and implemented. This function combines the techniques of channel reconstruction, noise reduction, as well as global and local contrast enhancement, thereby effectively enhancing the visual quality of underwater images and furnishing more favorable image data for subsequent applications such as underwater image analysis.

Explanation: Existing underwater image enhancement models show limited adaptability in various underwater scenes due to factors like water turbidity, light, and shooting parameters. To solve this, a new model is proposed. In the underwater environment, light attenuation and diffusion lead to image problems such as color distortion, low contrast, and noise. The uwredcomp function is designed with multiple techniques. Channel reconstruction corrects color cast, noise reduction improves clarity, and local contrast enhancement highlights details. This improves image visual quality and provides better data for underwater image analysis and related applications, enabling more accurate research in underwater imaging.

IV. Symbol Explanation

Symbol	Explanation
W	Width of the image.
H	Height of the image.
Deviation	Result of formula for deviation measure.
$\sigma(\text{mean}_R, \text{mean}_G, \text{mean}_B)$	Std dev of RGB channel means.
$\text{mean}_R, \text{mean}_G, \text{mean}_B$	Avg values of RGB channels.
$\text{mean}_{\text{total}}$	Overall average value.
Brightness	Result of brightness formula.
N	Total elements in calc.
Gray(i)	Gray value of i-th element.
$\frac{1}{N}$	For average gray calc.
$h(x)$	fuzzy kernel function.
$I_r(x), I_g(x), I_b(x)$	RGB channel intensities at x.
k_{r0}, k_{g0}, k_{b0}	Initial RGB channel coeffs.
μ_r, μ_g, μ_b	RGB channel attenuation coeffs.
$J_r(x), J_g(x), J_b(x)$	Initial RGB channel intensities at x.
λ	Coefficients associated with turbidity.
$t(x)$	Variable function (time/distance).
$B(t(x))$	Background contribution function.

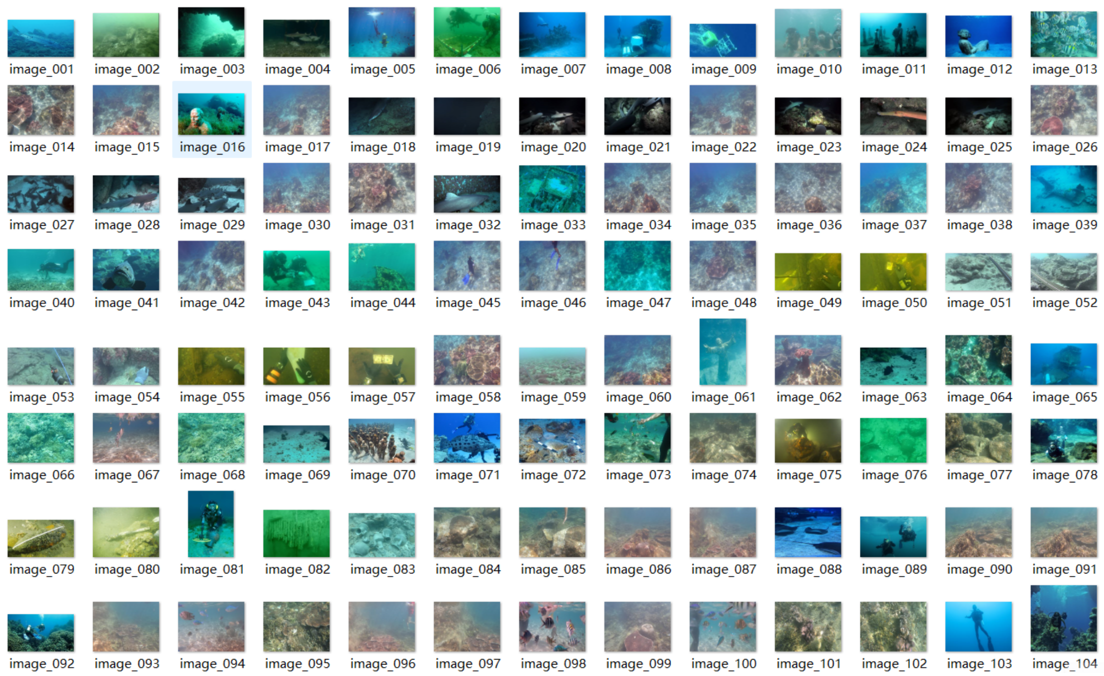

Figure 2 All Image

V. Model Establishment and Solution of Problem 1

5.1 Image Data Preprocessing

At first, data format standardization and channel disassociation constitute the initial and fundamental steps in the preprocessing of underwater images. The acquired image data, which be in the uint16 format, is converted to uint8 using the im2uint8 function. This conversion is not only a technical necessity but also a strategic maneuver to ensure seamless integration and compatibility within the subsequent processing pipeline. It lays the groundwork for more sophisticated operations by establishing a unified data format.

Subsequently, the channel disassociation process is initiated. By meticulously exploiting the channel index, the image is deconstructed into its red, green, and blue color channel components. This dissection results in the generation of three distinct two-dimensional matrices, namely redChannel, greenChannel, and blueChannel. Each of these matrices serves as a repository of pixel information specific to its corresponding color channel. This granular level of separation is not a mere technicality but a cornerstone for in-depth chromatic analysis. It enables the extraction of nuanced color characteristics, which in turn, paves the way for accurate color bias determination and other advanced image processing tasks.

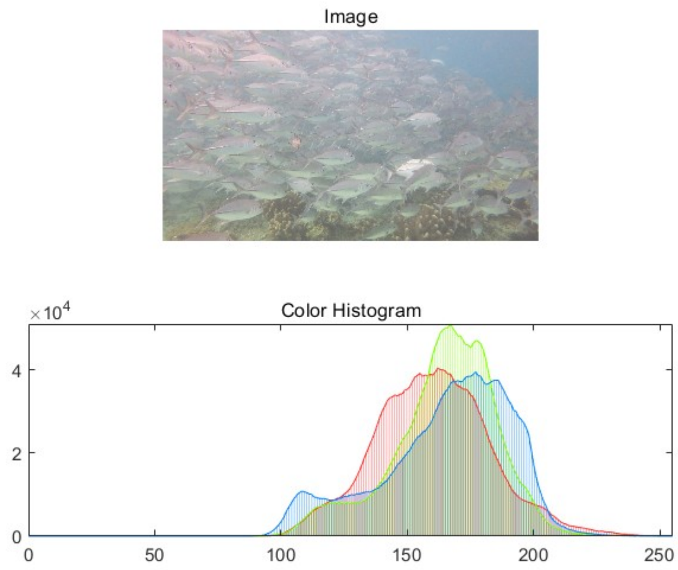

Figure 3 Color Channel Histogram

5.2 Color Cast Processing

5.2.1 *Color Channel Histogram Computation*

The histograms of the red, green, and blue channels are computed respectively using the imhist function, yielding histRed, histGreen, and histBlue. These histograms can intuitively reflect the distribution of each color within the image and serve as one of the crucial data sources for color aberration judgment.

The following presents an instance illustrating the methodology employed for computing the color channel histogram of an image:

5.2.2 *Color Channel Mean Value Computation*

The mean values of all pixel values within the red channel, green channel, and blue channel are computed and denoted as redMean, greenMean, and blueMean respectively. These mean values constitute significant indicators of the average luminance level for each individual color channel and occupy a central role in the assessment of color deviation. When the disparity between the mean values of the channels exceeds a reasonable range, it suggests a potential imbalance in the image's color composition, thereby indicating the occurrence of color cast.

$$\mu_R = \frac{1}{W \times H} \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} R(x, y)$$

$$\mu_G = \frac{1}{W \times H} \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} G(x, y)$$

$$\mu_R = \frac{1}{W \times H} \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} G(x, y)$$

5.2.3 *Color channel variance calculation*

The variances of the red channel, the green channel, and the blue channel are computed via the var function. Such variances are capable of precisely reflecting the dispersion of the pixel values within each color channel, that is, the uniformity of color distribution. In the context of color aberration assessment, the abnormal variation of the variance ratio can serve as one of the crucial bases for determining the color aberration of an image. It can be mutually complementary with indicators such as the mean difference to construct a multi-dimensional basis framework for evaluating color cast.

$$\sigma_R^2 = \frac{1}{W \times H} \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} (R(x, y) - \mu_R)^2$$

$$\sigma_G^2 = \frac{1}{W \times H} \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} (G(x, y) - \mu_G)^2$$

$$\sigma_B^2 = \frac{1}{W \times H} \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} (B(x, y) - \mu_B)^2$$

5.2.4 *Judgment Mechanism*

The color cast assessment procedure consists of three main phases. Firstly, in the evaluation of mean differences, the differences in means between the red and green channels, the red and blue channels, as well as the green and blue channels are strictly compared with the previously established mean difference threshold. If the absolute value of any one of these differences exceeds the threshold, it indicates a significant disparity in the average luminance of the color channels, thus leading to the conclusion that the image may potentially show color cast. At the same time, in order to quantitatively measure the extent of the likelihood of color cast, the mean difference score is calculated. It is obtained by adding up the ratios of the differences in means of each channel to the threshold. This score vividly reflects the degree of deviation of the color cast based on the mean difference.

Subsequently, in the assessment of variance ratios, the variance ratios between the red and green channels, the red and blue channels, and the green and blue channels are meticulously compared with the defined variance ratio threshold. If any one of these ratios is greater than the threshold, it implies an abnormal spread of the pixel values within the color channels, that is, the uniformity of the color distribution is disrupted, and consequently, the image is judged to potentially have color cast. Correspondingly, the variance ratio score is calculated. It is calculated by summing up the ratios of the variance ratios of each channel to the threshold, which numerically quantifies the probability of color cast from the perspective of variance.

Finally, in the comprehensive evaluation, regarding the determination of the color intensity values corresponding to the histogram peaks, the peak positions of the red channel histogram, the green channel histogram, and the blue channel histogram are accurately identified using the function that finds the maximum value. The color intensity values corresponding to these peaks reflect the color characteristics of the image from a different and distinctive perspective. In the comprehensive judgment process, if the conditions of the previous mean difference or variance ratio are met, or if the differences in the color intensity values of the histogram peaks of each channel are relatively large (for example, the absolute value of the difference between redPeakBin and greenPeakBin is greater than 30, etc.), then the image is considered to have a relatively high probability of color cast. Based on this, the comprehensive score is calculated. It is composed of the sum of the mean difference score, the variance ratio score, and the score related to the difference in histogram peak values (histogram peak difference score). This comprehensively combines multiple criteria for determining color cast. Eventually, the comprehensive score is normalized to enable a unified comparison and analysis of the magnitudes of color cast among different images. variance ratio:

$$\text{Ratio}_{R-G} = \frac{\sigma_R^2}{\sigma_G^2}$$

$$\text{Ratio}_{R-B} = \frac{\sigma_R^2}{\sigma_B^2}$$

Judgment formula as follow:

$$\text{Deviation} = \frac{\sigma(\text{mean}_R, \text{mean}_G, \text{mean}_B)}{\text{mean}_{\text{total}}}$$

5.3 Low light Processing

5.3.1 *Grayscale Image Conversion*

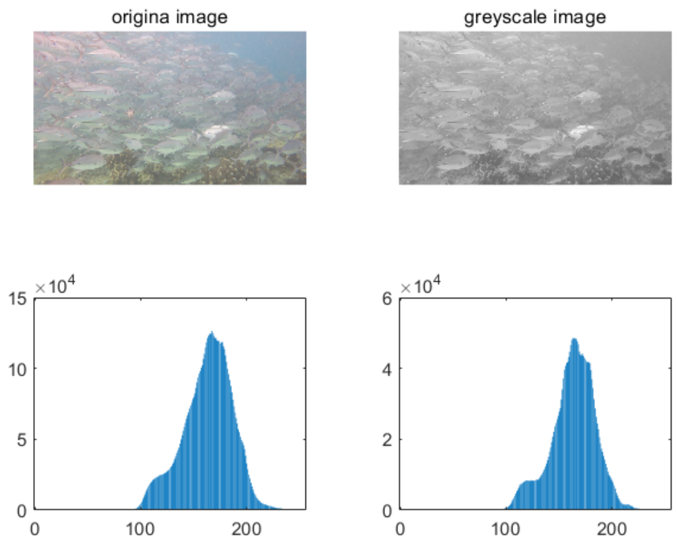

Figure 4 Gray-scale Image and Histogram

For the purpose of analyzing whether the image is in a low-light condition from the luminance perspective, the rgb2gray function is employed to transform the original color image into a grayscale image, namely grayImage. This conversion effectively simplifies the color information of the image into a single luminance datum, thereby rendering the subsequent low-light judgment based on luminance more straightforward and efficient.

5.3.2 *Grayscale Image Mean Value calculation*

The average gray value of all the pixels within the grayscale image is calculated. This average gray value serves as a vital indicator for assessing the overall luminance level of the image. Subsequently, this calculated value is compared with the preset low-light threshold. If the average gray value is lower than the preset threshold, the image is initially identified as a low-light image. At the same time, in order to quantify the degree of the low-light intensity of the image, the mean light score is computed. The mean light score is obtained by taking the ratio of the difference between the low-light threshold and the average gray value to the low-light threshold. This score can intuitively reflect the extent to which the brightness of the image is lower than the low-light threshold.

5.3.3 *Judgment Mechanism*

At first,The histogram of the grayscale image is computed by calculating the proportion of the number of pixels within the range designated as 0 - 50 (in the context of the grayscale histogram) to the total number of pixels. Then, this ratio is compared with the preset histogram threshold. If this ratio is greater than the histogram threshold, the image is further determined to be a low-light image.

Correspondingly, the histogram light score is calculated by taking the difference between the ratio of low-light pixels and the histogram threshold, and then dividing this difference by the histogram threshold.

Finally, the maximum value between the mean light score and the histogram light score is taken as the light score. This light score serves as a comprehensive quantitative index for assessing the low-light level of an image and is capable of reflecting the low-light condition of the image in a more comprehensive and accurate way.

Judgment formula as follow:

$$\text{Brightness} = \frac{1}{N}\sum_{i=1}^{N}\text{Gray}(i)$$

5.4 Blur Processing

5.4.1 *Image grayscaling*

For color images, the function of rgb2gray is first employed to convert them into grayscale images. The purpose of this conversion is to carry out an analysis on the degree of blurring and make relevant judgments. If the image is originally a grayscale image, it will directly move on to the next step.

$$\text{Variance} = \frac{1}{N}\sum_{i=1}^{N}(\text{Laplacian}(i) - \mu)^2$$

5.4.2 *Fourier Transformation and Spectrum Analysis*

The two-dimensional fast Fourier transform is carried out on the grayscale image. The fft2 function is employed to obtain F, following which the fftshift function is utilized to shift the zero-frequency component to the center of the spectrum, thereby obtaining F-shifted. Subsequently, the magnitude spectrum of magnitude-spectrum is computed. The Fourier transform has the capability to convert the image from the spatial domain to the frequency domain, rendering the frequency characteristics of

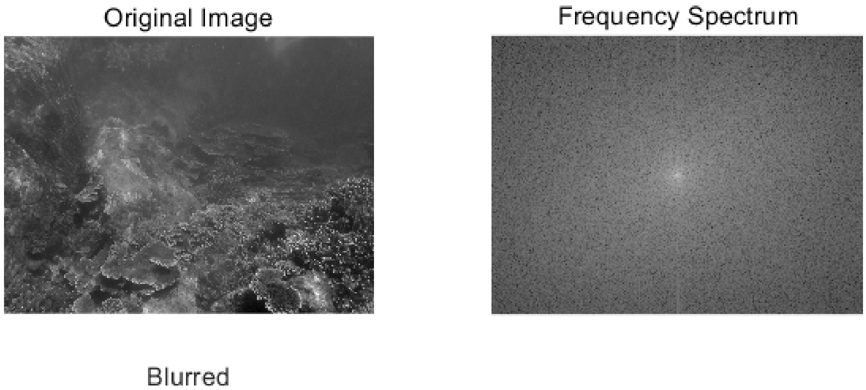

Figure 5 Spatial Spectral Transformations of Gray Scale Maps and Fourier

the image observable and furnishing the foundation for the subsequent analysis of the high-frequency components of the image, thus enabling the revelation of the image's characteristics and providing a basis for the subsequent examination of its high-frequency constituents. The formula is as follows:

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(\frac{ux}{M}+\frac{vy}{N})}$$

The following presents the outcomes of our image conversion to a grayscale map and the Fourier space spectrum transformation:

5.4.3 *Judgment Mechanism*

The high frequency energy radius is compared with the preset blurring threshold. In the event that the high frequency energy radius is less than the said threshold, the image is adjudged to be blurred. For the purpose of quantifying the degree of fuzziness of the image, the fuzzy score is computed as the quotient of the absolute value of the difference between the high-frequency ratio value and the fuzzy threshold value divided by the fuzzy threshold value. This score can intuitively reflect the extent to which the degree of fuzziness of the image deviates from the fuzzy threshold, thereby providing a significant quantitative basis for the subsequent processing and analysis of the fuzzy image.

Judgment formula as follow:

$$\text{Variance} = \frac{1}{N}\sum_{i=1}^{N}(\text{Laplacian}(i) - \mu)^2$$

5.5 Result

The final results can be found in “Answer.xls” attachment.

VI. Model Establishment and Solution of Problem 2

6.1 Color Cast Degeneration Model

It is hypothesized that color deviation is principally induced by the varying degrees of attenuation of different light colors during the propagation of light in water. Moreover, water turbidity exerts diverse impacts on the attenuation of different light colors. When light propagates in water, the attenuation coefficients of different light colors are designated as k_r(attenuation coefficient of red light), k_g(attenuation coefficient of green light), and k_b(attenuation coefficient of blue light), with $k_r \neq k_g \neq k_b$. To account for the influence of water turbidity on the attenuation coefficient, it is postulated that the attenuation coefficient is exponentially related to turbidity, specifically, $k_r = k_{r0}e^{-t\mu_r}$, $k_g = k_{g0}e^{-t\mu_g}$, $k_b = k_{b0}e^{-t\mu_b}$ (where k_{r0}, k_{g0}, k_{b0} are the initial attenuation coefficients, and μ_a, μ_b, μ_c are the coefficients associated with turbidity).

$$\begin{pmatrix} I_r(x) \\ I_g(x) \\ I_b(x) \end{pmatrix} = \begin{pmatrix} k_{r0}e^{-t\mu_r}J_r(x) \\ k_{g0}e^{-t\mu_g}J_g(x) \\ k_{b0}e^{-t\mu_b}J_b(x) \end{pmatrix} t(x) + B(t(x))$$

6.2 Low light Degeneration Model

Low light is postulated to originate from the confluence of light propagation losses, faint ambient light, and the phenomena of shading and scattering of light that are instigated by impurities and particulate matter within the water body. Light propagation losses lead to the decay of light intensity in a specific proportion. Denoting the proportion of decay as α, the ambient light B is of relatively lower intensity. The presence of impurities and particulate matter in the aqueous medium further curtails the light flux reaching the imaging system. The magnitude of such an effect correlates with the impurity concentration c, the mean particle diameter d, and the particle distribution density ρ. An attenuation factor,$beta(c, d, \rho)$, is introduced to delineate this impact.

$$I(x) = \alpha J(x)t(x) + B_0 e^{-t\lambda} t(x)$$

6.3 Blur Degeneration Model

It is assumed that low light is primarily caused by a combination of energy loss during light propagation and diminished ambient light due to water turbidity. Light propagation losses cause the light intensity to decay by a certain ratio, let the decay ratio be α. Water turbidity affects the ambient light intensity B. It is assumed that ambient light is exponentially related to turbidity, i.e., $B(x) = B_0(x)e^{-t\lambda}$ (B_0 is the initial ambient light intensity, and λ is a coefficient related to turbidity).

$$I(x) = J(x) \otimes (h_0(x)e^{-t\sigma})t(x) + B(t(x))$$

6.4 Causes, Similarities and Differences

Causes of Color Cast Degeneration:

Color Cast Image: During the propagation of light in water, the attenuation coefficients of different spectral components of light exhibit variations as a consequence of the disparity in water turbidity levels. This phenomenon gives rise to an imbalance in the color distribution within the image. For instance, under conditions of relatively elevated turbidity, red light may experience a more rapid attenuation rate, thereby inducing a color cast in the image, manifesting as a bluish or greenish tint.

Low Light Image: The energy dissipation that occurs during the propagation of light, in conjunction with the attenuation of ambient light attributable to water turbidity, collectively contributes to a reduction in the overall luminance of the image. A higher degree of turbidity correlates with a weaker ambient light intensity, consequently resulting in a darker appearance of the image.

Blur Image: The forward scattering effect, compounded by the augmented light scattering induced by water turbidity, leads to the perturbation of pixel values within the image by neighboring pixels. As the turbidity increases, the severity of scattering intensifies, thereby leading to a blurring of the edges and a loss of fine details within the image.

A comprehensive analysis of similarities and differences:

similarities: All three degradation types are intrinsically associated with the propagation characteristics of light within the aqueous medium and are subject to the combined influence of the light transmission function and the ambient light, the latter being modulated by water turbidity. Invariably, they culminate in a deterioration of image quality, thereby exerting an adverse impact on visual recognition and analytical processes.

differences: Color cast predominantly induces alterations in the color distribution of the image, manifested as asymmetrical variations in the pixel values across different color channels. Low light conditions principally lead to a diminution in the luminance of the image, yielding a reduction in the overall magnitude of pixel values. Blur primarily compromises the sharpness of the image, as evidenced by a weakened contrast among pixel values. The genesis of color cast lies in the correlation between the differential attenuation of diverse spectral components of light and water turbidity. Low light scenarios are the consequence of the combined effect of light propagation losses and the attenuation of ambient light attributable to turbidity. Blur is instigated by the augmentation of light scattering, which is a direct result of forward scattering and the exacerbating influence of turbidity.

VII. Model Establishment and Solution of Problem 3

7.1 Image Acquisition and Preliminary Configuration

The imread function is employed to retrieve the specified image file, and the resultant image data, which is presented in the form of a three-dimensional array with the third dimension denoting the color channels (RGB), is stored in the variable t. Subsequently, the red channel data is isolated from the read image t and stored in the variable I through indexing. Concurrently, the height and width of the image data corresponding to this channel are also acquired.

Figure 6 Processed Low Light image

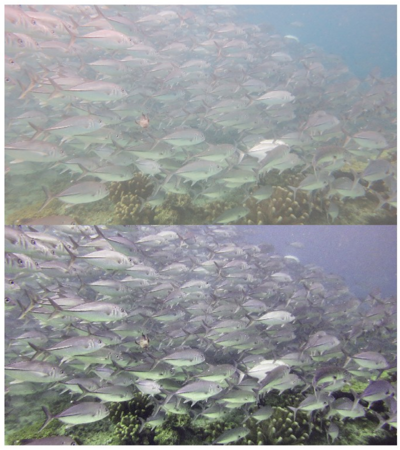

Figure 7 Processed Blur image

7.2 Computation of the Mean Value for Each Channel

Regarding the red channel data, its mean value is computed through two successive applications of the mean function. Initially, the two-dimensional channel data is averaged along the columns, yielding a row vector. Subsequently, this row vector is further averaged along the rows, thereby obtaining a scalar value. Ultimately, the obtained mean value is divided by 255 to attain the normalized mean value specific to the red channel. Similarly, the mean values of the green channel and the blue channel are respectively calculated following an analogous procedure. That is, for each of these channels, the two-dimensional channel data undergoes the same sequence of column-wise and then row-wise averaging operations. Thereafter, the resultant values from these calculations are divided by 255 to derive the normalized mean value for the green channel and the mean value for the blue channel.

7.3 Histogram-Related Processing for Each Channel

7.3.1 *Initiation of Pixel Gray-scale Statistical Operations*

For each channel (taking the red channel as an illustrative example), two zero vectors, namely s and sg, each with a length of 256, are created for the purpose of tallying the number of pixels at each grayscale level. It is postulated herein that the grayscale value range of the image spans from 0 to 255, thereby comprising a total of 256 grayscale levels.

7.3.2 *Construction of Histograms*

Through iterating over the range of gray values from 0 to 255, for each specific gray value K , the Find function is employed to identify the pixel positions within the color channel data that are equivalent to the given gray value. Subsequently, the Length function is utilized to tally the number of these identified positions. Then, this count is divided by the total pixel count of the image, thereby obtaining the proportion of pixels within the red channel corresponding to each individual gray value. The resultant value is then stored in the vector gp. Analogous operations are carried out for both the green and blue channels in a similar manner.

7.3.3 *Compute the Density of the Gray Scale Distribution*

The range of gray values spanning from 0 to 255 is traversed once again through looping. For each gray value , the number of pixels s(i) that has been previously counted for this specific gray value is divided by the total number of pixels within the image. This operation yields the density of the gray distribution for the red channel, and the resultant value is stored in the vector p. Similar operations are respectively carried out for the green channel and the blue channel. Specifically, for each gray value i within the range of 0 to 255, the corresponding previously counted pixel numbers for the green channel and the blue channel are divided by the total pixel count of the image. The outcomes of these operations for the green channel and the blue channel are respectively stored in the vectors pg.

7.3.4 *Calculation of the Distribution of Cumulative Histograms*

Firstly, the distribution density p(1) of the first gray value within the red channel is assigned to the cumulative histogram distribution vector c(1). Subsequently, the range of gray values from to is traversed via looping. For each gray value within this range, the cumulative histogram distribution c(i-1) value of the preceding gray value is added to the distribution density p(i) of the current gray value, thereby obtaining the cumulative histogram distribution value c(i) of the current gray value. Analogous operations are performed for both the green channel and the blue channel. The results of these operations for the green channel and the blue channel are respectively stored in the vector cg.

7.3.5 *Rounding and Normalization of Cumulative Distribution*

A rounding and normalization operation is carried out on the cumulative histogram distribution vectors, namely c for the color channel and cg for the green channel. The range of cumulative distribution values is normalized to fall within the interval of 0-255. This is achieved by multiplying each cumulative distribution value by 255, subsequently adding 0.5, and then converting the resultant values into an unsigned 8-bit integer type.

7.4 Equalization Processing of Each Channel

By traversing each row and each column of the image through looping, for each pixel position (i, j), the red channel data is used as an index to retrieve the corresponding equalized value from the already processed cumulative histogram distribution vector c, and this value is then stored in the variable Ir. Subsequently, based on the previously calculated mean value of the red channel, further adjustments are made to the equalized red channel data Ir. Specifically, Ir is multiplied by (1 - mean-red) * 0.5, and the resultant value is stored in the relevant variable as the red channel data of the processed image.

7.5 Adjustment and Display of the Processed Image

If the channel number ch equals 3, it implies that the image is a color image. In this situation, the variable dis is converted to the double-precision type and stored in a double-precision image. Subsequently, the mean values of the red, green, and blue channels of the converted image are calculated respectively. Then, the average gray value is computed based on these three mean values. Next, adjustments are made to the data of each channel in the double-precision image according to the proportional relationship between the average gray value and the mean values of each channel. Finally, the adjusted double-precision image data is multiplied by 600 and then converted to the unsigned 8-bit integer type, which is stored back in dis. The processed image is then displayed using the imshow function to exhibit the effect of the image after the processing.

VIII. Model Establishment and Solution of Problem 4

8.1 Image Data Manipulation

First and foremost, the image files are traversed via a loop. Specifically, with the aid of a ”for” loop, the loop variable ”i” is set to range from 1 to 12, thereby enabling the sequential processing of twelve image files. During each iteration of the loop, in accordance with the current value of ”i”, the ”sprintf” function is employed to generate the name of the image file. Subsequently, the ”imread” function is utilized to read the file, and the retrieved image data is stored in the variable ”A”, which serves as the data of the original image. Thereafter, the custom-defined ”uwredcomp” function is called upon to process the original image ”A”, with the resulting processed image data being stored in the variable ”A-1”.

Subsequently, the original image data ”A” is assigned to the variable representing the original image, and the processed data ”A-1” is assigned to the variable ”img-processed” for the purposes of subsequent unified operations and quality assessment. Moreover, specific statements are executed to convert these two image variables into double-precision types and to normalize the pixel values to the interval [0, 1]. This is done in order to establish a suitable data foundation for the subsequent calculations of quality assessment.

8.2 Principles of Image Enhancement Algorithms

Based on the propagation characteristics of light in underwater images, light of different wavelengths attenuates to different degrees in water, with the attenuation of red light being relatively more pronounced. To compensate for this wavelength-dependent attenuation, we have adopted a method of channel weighted reconstruction. By analyzing the relationships between the attenuation coefficients of light of different wavelengths, we calculate the weights of each channel when reconstructing the red channel, thereby enhancing the information of the red channel and improving the overall color representation of the image. Meanwhile, considering that underwater images also suffer from contrast loss problems caused by the diffusion effect, we have adopted a strategy of combining global (linear + gamma) and local (CLAHE) contrast enhancement. Global contrast enhancement can adjust the overall brightness and contrast range of the image, while local contrast enhancement can better highlight the details of local regions of the image. The combination of the two can more comprehensively enhance the visual quality of the image.

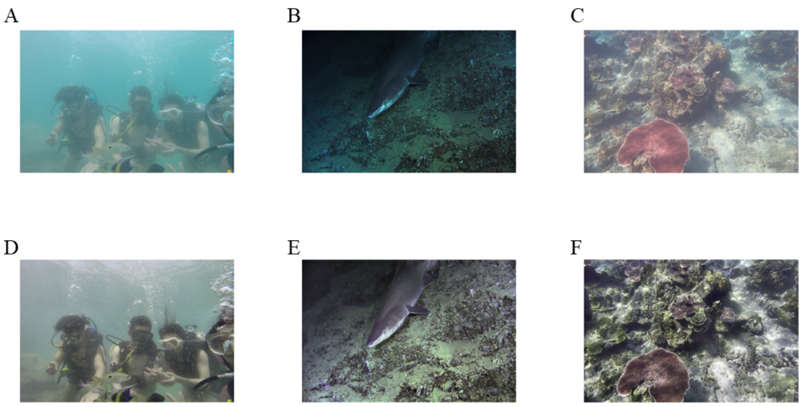

Figure 8 The effect of complex scene models

8.3 Calculate quality assessment indicators

8.3.1 *Calculate the UIQM metric*

For both the original image and the processed image, the computeUIQM function is invoked to calculate the UIQM (Underwater Image Quality Measure) metric. Within this function, it first employs an if statement to examine whether the input image is in RGB format. If the input image is in RGB format, three sub-metrics for the comprehensive scoring of UIQM are calculated: For colorfulness, the data of the R, G, and B channels are extracted, two difference arrays are computed, their standard deviations and means are calculated, and the UICM is calculated using a specific formula. For sharpness, the RGB image is converted into a grayscale image. The Sobel operators are then utilized to calculate the horizontal and vertical gradients. Subsequently, the gradient magnitude is calculated, and the mean of the gradient magnitude is taken as the value of the UISM.

For contrast, the RGB image is converted into a grayscale image, and the standard deviation of the grayscale pixel values is regarded as the value of the UIConM Finally, the values of the three sub-metrics obtained through the above calculations are weighted and summed up according to specific weights. The formula for calculating the UIQM value is

$$uiqm = 0.0282 \times uicm + 0.2953 \times uism + 3.5753 \times uiconm$$

The UIQM values calculated for the original image and the processed image are respectively stored in the variables.

8.3.2 *Calculate the PSNR metric*

The computePSNR function is invoked to compute the PSNR (Peak Signal-to-Noise Ratio) metric between the original image and the processed image. Within the computePSNR function, initially, an if statement is employed to verify whether the two input images possess the same dimensions. In the event that they do not, an error will be triggered, given that the PSNR calculation mandates that the two input images have identical sizes. Should the two images have the same size, the Mean Squared Error (MSE) of the disparity in pixel values is calculated. Firstly, the squared difference is computed for each corresponding pair of pixels, and subsequently, the MSE value is derived by averaging these squared differences. The PSNR value is then determined based on the MSE value. Specifically, if the MSE equals 0, signifying that the two images are completely identical, the relevant variable is set to infinity. Otherwise, the PSNR value is calculated using the formula:

$$psnrvalue = 10 * log10((max_I^2)/mSE)$$

where max-I is set to 1 due to the fact that the images have been normalized to the range of [0, 1]. The computed PSNR value is then stored in the designated storage variable.

8.3.3 *Calculate the UCIQE metric*

For original image and processed image, call computeUCIQE to calculate UCIQE metric. In computeUCIQE:Check if input is RGB via if statement. If not, raise error as UCIQE needs RGB.If RGB, convert to CIELAB, extract L, a, b channels.Calculate UCIQE sub-metrics:Color saturation (S): std of combined a and b channels mean as S;Contrast (C): std of L channel data.Sharpness (H): define Sobel templates, calculate L channel gradients, magnitude,mean as H .Weight and sum sub-metrics (0.468, 0.274, 0.257) for UCIQE using the formula:

$$uciqe = 0.468 * S + 0.274 * C + 0.257 * H$$

Store original and processed UCIQE values in uciqe-original and uciqe-processed.

8.4 Result

The final results can be found in “Answer.xls” attachment.

Table 1 Image Quality Metrics for Different Tests

Image File Name	PSNR	UCIQE	UIQM
test_001	17.7025	11.7739	0.5736
test_002	15.2175	10.2449	0.6996
test_003	12.2648	8.5838	0.5245
test_004	12.1592	14.8994	0.7835
test_005	18.0474	8.1459	0.5159
test_006	14.7696	8.9579	0.6797
test_007	19.9934	12.4318	0.6916
test_008	13.4897	7.2398	0.4034
test_009	13.7616	11.4415	0.595
test_010	15.0275	10.7758	0.4001
test_011	15.8215	11.9929	0.4549
test_012	17.9202	4.9795	0.3701

IX. Model Establishment and Solution of Problem 5

9.1 Practical Applications

In the field of underwater image enhancement, both specific-scenario and complex-scenario enhancement techniques possess their own characteristics and advantages.

Specific-scenario enhancement techniques are designed to handle single degradation types such as color cast, low light, and blur. For instance, when dealing with color cast, by precisely computing multiple factors such as the mean differences, variance ratios, and histogram peak color intensity differences of each color channel in the image, it is possible to accurately identify the color cast situation and perform targeted corrections to restore the natural color of the image. For low-light images, through grayscale image conversion and mean value calculation, combined with comparisons with preset thresholds, the low-light condition can be effectively recognized, and the degree of low light can be evaluated quantitatively by the low-light score, thereby enabling a reasonable adjustment of the image brightness. In the processing of blurred images, with the aid of image grayscaling and Fourier transformation and spectrum analysis, by comparing the high-frequency energy radius with the blurring threshold, the degree of image blurring can be accurately determined, and the blurring situation can be measured by the blurring score, followed by the implementation of corresponding enhancement measures.

Complex-scenario enhancement techniques, on the other hand, aim to address complex situations where multiple degradation types coexist. For example, the non-physical model-based enhancement technique proposed in the paper integrates multiple technical means such as channel reconstruction, noise reduction, and global and local contrast enhancement through the design of the uwredcomp function. This technique can comprehensively consider complex factors such as the wavelength-dependent attenuation and diffusion effects of light during underwater propagation, conduct comprehensive processing of the image, effectively improve the visual quality of the image, and provide more favorable data support for subsequent applications such as underwater image analysis.

In practical applications, the following feasibility suggestions can be considered to better achieve underwater visual enhancement:

Selecting techniques based on scene characteristics: In cases where the main degradation type of the image is known to be a single type (such as only color cast or only low light) and the scene is relatively simple, specific-scenario enhancement techniques should be preferentially selected. This is because they can more precisely optimize for a single problem, with relatively lower computational costs and significant effects. In complex underwater environments, such as deep-sea exploration scenarios where multiple degradation factors may coexist simultaneously, complex-scenario enhancement techniques should be employed to ensure the overall quality of the image is effectively improved.

Optimizing parameter settings: The performance of both specific-scenario and complex-scenario enhancement techniques is closely related to parameter settings. Before practical application, key parameters such as thresholds and weights should be optimized and adjusted according to the specific characteristics of the underwater environment and the parameters of the imaging equipment. For example, in waters with different turbidity levels, the most suitable parameter values can be determined through experiments or statistical analysis based on a large number of samples to enhance the accuracy and effectiveness of image enhancement.

Multi-technique fusion strategy: Consider the organic integration of specific-scenario and complex-scenario enhancement techniques. For example, for images with overall severe degradation but dominated by a certain degradation type, specific-scenario enhancement techniques can be first used to preliminarily handle the main problem, and then complex-scenario enhancement techniques can be applied for overall optimization. This way, the image quality can be maximally enhanced while ensuring processing efficiency.

Real-time monitoring and feedback adjustment: During the practical application process, a real-time monitoring mechanism should be established to conduct real-time evaluation of the enhanced image quality. Based on the evaluation results, timely feedback and adjustment of the parameters or strategies of the enhancement techniques should be carried out to adapt to the constantly changing underwater environment and task requirements. For example, when an underwater robot is conducting seabed exploration, as it moves to different water quality areas, the image quality may change. Through real-time monitoring and feedback adjustment at this time, the best visual effect can always be ensured.

X. References

[1] ian Zhixin, Jiang Qiuping. Quality Evaluation of Underwater Image Enhancement Based on Quality-aware Domain Adaptation [J/OL]. Journal of Signal Processing, 1-12 [2024-11-25]. http://kns.cnki.net/kcms/detail/11.2406.TN.20241118.1421.002.html.

[1] Bai Jiaqing. Research on Underwater Image Restoration Methods Based on Color Correction and Physical Models [D]. Ningxia University, 2022. DOI: 10.27257/d.cnki.gnxhc.2022.000244.

[2] Yang Gaolin. Research on Underwater Image Restoration Models in Different Scenarios [D]. Zhongkai University of Agriculture and Engineering, 2023. DOI: 10.27700/d.cnki.gzcny.2023.000264.

[3] Liu Min. Underwater Image Enhancement Based on Domain Adaptation [D]. Harbin Institute of Technology, 2022. DOI: 10.27061/d.cnki.ghgdu.2022.002086.

[4] Zhang Weidong. Research on the Technology of Underwater Optical Image Sharpening [D]. Dalian Maritime University, 2022. DOI: 10.26989/d.cnki.gdlhu.2022.000025.

[5] Chen Yahui. Research on Histogram Enhancement and Evaluation Algorithms for Underwater Color Images [D]. Henan University of Technology, 2024. DOI: 10.27791/d.cnki.ghegy.2024.000338.

[6] Li Sheng, Hu Jialong, Jiang Guoqing. An Automatic Segmentation Algorithm for Low-Light Images Based on an Improved Neural Network [J]. Journal of Xuchang University, 2024, 43(05): 105-110.

[7] Ye Zhenghao, Yang Min. A Real Image Deblurring Algorithm Based on the Improved SDE Diffusion Model [J/OL]. Microelectronics and Computer, 1-9 [2024-11-25]. http://kns.cnki.net/kcms/detail/61.1123.tn.20241113.1642.010.html.

XI. Appendix

Listing 1: The Matlab Source code of Question 1

```
clc;
clear all;
close all;
fileID = fopen('result.csv', 'w');
% fprintf(fileID, '');
folder_path = 'C:\Users\22477\Desktop\Attachment 1';
file_list = dir(fullfile(folder_path, '*.jpg'));
file_list = [file_list; dir(fullfile(folder_path, '*.png'))];
meanDiffThreshold = 45;
varRatioThreshold = 2;
lowLightThreshold = 95;
histogramThreshold = 0.7;
blurThreshold = 0.003;
max_d = 0;

for i = 1:length(file_list)
    file_name = fullfile(folder_path, file_list(i).name);
    image = imread(file_name);
    if isa(image, 'uint16')
        image = im2uint8(image);
    end
    redChannel = image(:, :, 1);
    greenChannel = image(:, :, 2);
    blueChannel = image(:, :, 3);
    histRed = imhist(redChannel);
    histGreen = imhist(greenChannel);
    histBlue = imhist(blueChannel);
    redMean = mean(redChannel(:));
    greenMean = mean(greenChannel(:));
    blueMean = mean(blueChannel(:));
    redVar = var(double(redChannel(:)), 1);
    greenVar = var(double(greenChannel(:)), 1);
    blueVar = var(double(blueChannel(:)), 1);
```

```
meanDiffResult = 'No';
meanDiffScore = 0;

if abs(redMean - greenMean) > meanDiffThreshold || abs(redMean -
    blueMean) > meanDiffThreshold || abs(greenMean - blueMean) >
    meanDiffThreshold
    meanDiffResult = 'Yes';
    meanDiffScore = sum([abs(redMean - greenMean) /
        meanDiffThreshold, abs(redMean - blueMean) /
        meanDiffThreshold, abs(greenMean - blueMean) /
        meanDiffThreshold]);
end
varRatioResult = 'No';
varRatioScore = 0;

if redVar / greenVar > varRatioThreshold || redVar / blueVar >
    varRatioThreshold || greenVar / blueVar > varRatioThreshold
    varRatioResult = 'Yes';
    varRatioScore = sum([redVar / greenVar / varRatioThreshold,
        redVar / blueVar / varRatioThreshold, greenVar / blueVar /
        varRatioThreshold]);
end
[~, redPeakBin] = max(histRed);
[~, greenPeakBin] = max(histGreen);
[~, bluePeakBin] = max(histBlue);
comprehensiveResult = 'No';
comprehensiveScore = 0;

if (abs(redMean - greenMean) > meanDiffThreshold || abs(redMean -
    blueMean) > meanDiffThreshold || abs(greenMean - blueMean) >
    meanDiffThreshold) ||...
        (redVar / greenVar > varRatioThreshold || redVar / blueVar >
            varRatioThreshold || greenVar / blueVar >
            varRatioThreshold) ||...
        (abs(redPeakBin - greenPeakBin) > 30 || abs(redPeakBin -
            bluePeakBin) > 30 || abs(greenPeakBin - bluePeakBin) > 30)
    comprehensiveResult = 'Yes';
```

```
    comprehensiveScore = meanDiffScore + varRatioScore +
        sum([abs(redPeakBin - greenPeakBin) / 30, abs(redPeakBin -
        bluePeakBin) / 30, abs(greenPeakBin - bluePeakBin) / 30]);
end
grayImage = rgb2gray(image);
meanGrayValue = mean(grayImage(:));
grayHist = imhist(grayImage);
meanLightResult = 'No';
meanLightScore = 0;

if meanGrayValue < lowLightThreshold
    meanLightResult = 'Yes';
    meanLightScore = (lowLightThreshold - meanGrayValue) /
        lowLightThreshold;
end
lowLightPixelsRatio = sum(grayHist(1:50))/numel(grayImage);
histogramLightResult = 'NO';
histogramLightScore = 0;

if lowLightPixelsRatio > histogramThreshold
    histogramLightResult = 'Yes';
    histogramLightScore = (lowLightPixelsRatio - histogramThreshold)
        / histogramThreshold;
end

if size(image, 3) == 3
    img_gray = rgb2gray(image);
else
    img_gray = image;
end
F = fft2(double(img_gray));
F_shifted = fftshift(F);
magnitude_spectrum = log(1 + abs(F_shifted));
[rows, cols] = size(img_gray);
center_row = floor(rows / 2);
center_col = floor(cols / 2);
radius = min(center_row, center_col) / 4;
```

```
    [X, Y] = meshgrid(1:cols, 1:rows);
    distance = sqrt((X - center_col).^2 + (Y - center_row).^2);
    high_freq = distance > radius;
    high_freq_energy = sum(sum(abs(F_shifted(high_freq)).^2));
    total_energy = sum(sum(abs(F_shifted).^2));
    high_freq_ratio = high_freq_energy / total_energy;
    blurResult = 'No';
    blurScore = 0;

    if high_freq_ratio < blurThreshold
        blurResult = 'Yes';
        blurScore = abs(high_freq_ratio - blurThreshold) / blurThreshold;
    end
    % if max_d < comprehensiveScore
    %     max_d = comprehensiveScore;
    % end
    comprehensiveScore = comprehensiveScore / 27.1241;
    % comprehensiveScore=comprehensiveScore/max(comprehensiveScore);
    LightScore=max(meanLightScore, histogramLightScore);

    if comprehensiveScore > LightScore && comprehensiveScore > blurScore
        Degraded_Image_Classification = 'color_cast';
    elseif LightScore > comprehensiveScore && LightScore > blurScore
        Degraded_Image_Classification = 'low_light';
    elseif blurScore > comprehensiveScore && blurScore > LightScore
        Degraded_Image_Classification = 'blur';
    end
    fprintf(fileID,
        '%s,%s,%s,%s,%s,%s,%.4f,%.4f,%.4f,%.4f,%.4f,%.4f,%s,%.4f,%s\n',
        file_name,meanDiffResult, varRatioResult, comprehensiveResult,
        meanLightResult, histogramLightResult, meanDiffScore,
        varRatioScore, comprehensiveScore, meanLightScore,
        histogramLightScore, LightScore,blurResult,
        blurScore,Degraded_Image_Classification);
end
fclose(fileID);
```

```
%% Plotting colour histograms
clc;
clear all;
close all;
f=imread('test_001.png');
subplot(211);
imshow(f);title('Image');
fcal=double(f);
[m,n,h]=size(f);
Y=zeros(h,256);

for k=1:h
    for i=1:m
        for j=1:n
            Y(k,fcal(i,j,k)+1)=Y(k,fcal(i,j,k)+1)+1;
        end
    end
end
X=0:1:255;
subplot(212);
histogram=bar(X,Y);
axis([0 255,-inf inf])
% xlabel('gray level');ylabel('number of pixels');

if h==3
    title('Color Histogram');
    set(histogram(1),'FaceColor',[1 0.1882 0.1882]);
    set(histogram(2),'FaceColor',[0.5 1 0]);
    set(histogram(3),'FaceColor',[0 0.5 1]);
    FreqNum=zeros(size(f,3),256);
    for i=1:size(f,3)
        for j=0:255
            FreqNum(i,j+1)=sum(sum(f(:,:,i)==j));
        end
    end
    hold on
    plot(X,Y(1,:),'Color',[1 0.1882 0.1882]);
```

```
    plot(X,Y(2,:),'Color',[0.5 1 0]);
    plot(X,Y(3,:),'Color',[0 0.5 1]);
    hold off
else
    title('grey histogram');
end
```

Listing 2: The Matlab Source code of Question 3

```
clear;
clear all;
close all;
fileID = fopen('iddddd.csv', 'w');
fprintf(fileID, 'PSNR,UCIQE,UIQM\n');
for ii=1:272
    % for ij=273:400
    fileName = sprintf('image_%03d.png',ii);
    % fileName = sprintf('image_%03d.jpg',ij);
    t = imread(fileName);
    % t = imread('image_080.png');
    I=t(:,:,1);
    mean_red = mean(mean(I)) / 255;
    Green = t(:,:,2);
    mean_green = mean(mean(Green)) / 255;
    Blue = t(:,:,3);
    mean_blue = mean(mean(Blue)) / 255;
    [height,width] = size(I);
    %subplot(121);
    %imshow(t),title('original image')
    s = zeros(1,256);
    sg = zeros(1,256);
    gp=zeros(1,256);
    for k=0:255
        gp(k+1)=length(find(I==k))/(height*width);
    end
    for i = 1:height
        for j = 1: width
            s(I(i,j) + 1) = s(I(i,j) + 1) + 1;
```

```
        sg(Green(i,j) + 1) = sg(Green(i,j) + 1) + 1;
    end
end
p = zeros(1,256);
pg = zeros(1,256);
for i = 1:256
    p(i) = s(i) / (height * width * 1.0);
    pg(i) = s(i) / (height * width * 1.0);
end
c = zeros(1,256);
cg = zeros(1,256);
c(1) = p(1);
for i = 2:256
    c(i) = c(i - 1) + p(i);
    cg(i) = c(i - 1) + p(i);
end
c = uint8(255 .* c + 0.5);
cg = uint8(255 .* c + 0.5);
clear Ir Ig Ib dis;
for i = 1:height
    for j = 1: width
        Ir(i,j) = c(I(i,j)+1);
        Ig(i,j) = cg(I(i,j)+1);
    end
end
Ir = Ir * (1 - mean_red) * 0.5 ;
% Ir = Ir * 0.5;
% Ig = Ig * 1;
dis(:,:,1)=Ir;
% dis(:,:,2)=Ig;
%subplot(122)
%imshow(Ir)
I=t(:,:,2);
[height,width] = size(I);
s = zeros(1,256);
gp=zeros(1,256);
for k=0:255
```

```
    gp(k+1)=length(find(I==k))/(height*width);
end
for i = 1:height
    for j = 1: width
        s(I(i,j) + 1) = s(I(i,j) + 1) + 1;
    end
end
p = zeros(1,256);
for i = 1:256
    p(i) = s(i) / (height * width * 1.0);
end
c = zeros(1,256);
c(1) = p(1);
for i = 2:256
    c(i) = c(i - 1) + p(i);
end
c = uint8(255 .* c + 0.5);
for i = 1:height
    for j = 1: width
        Ig(i,j) = c(I(i,j)+1);
    end
end
%subplot(122)
%imshow(Ig)
dis(:,:,2)=Ig * (1 - mean_green) * 0.5;
I=t(:,:,3);
[height,width] = size(I);
s = zeros(1,256);
gp=zeros(1,256);
for k=0:255
    gp(k+1)=length(find(I==k))/(height*width);

end
for i = 1:height
    for j = 1: width
        s(I(i,j) + 1) = s(I(i,j) + 1) + 1;
    end
```

```
end
p = zeros(1,256);
for i = 1:256
    p(i) = s(i) / (height * width * 1.0);
end
c = zeros(1,256);
c(1) = p(1);
for i = 2:256
    c(i) = c(i - 1) + p(i);
end
c = uint8(255 .* c + 0.5);
for i = 1:height
    for j = 1: width
        Ib(i,j) = c(I(i,j)+1);
    end
end
dis(:,:,3)=Ib * (1 - mean_blue) * 0.5;
% subplot(122)
% imshow(Ib)
% subplot(122);
% figure
[height, width, ch] = size(dis);
if ch == 3
    image_double = im2double(dis);
    r_mean = mean(image_double(:,:,1));
    g_mean = mean(image_double(:,:,2));
    b_mean = mean(image_double(:,:,3));
    mean_grey = (r_mean + g_mean + b_mean)/3;
    image_double(:,:,1) = image_double(:,:,1) * (mean_grey/r_mean);
    image_double(:,:,2) = image_double(:,:,2) * (mean_grey/g_mean);
    image_double(:,:,3) = image_double(:,:,3) * (mean_grey/b_mean);
    dis = uint8(image_double*600);
end
img_original = t;
img_processed = dis;
% figure;
% imshow(img_original);
```

```
    % title('original image');
    % figure;
    % imshow(img_processed);
    % title('processed image');
    img_original = im2double(img_original);
    img_processed = im2double(img_processed);
    psnr_value = computePSNR(img_original, img_processed);
    uciqe_original = computeUCIQE(img_original);
    uciqe_processed = computeUCIQE(img_processed);
    uiqm_original = computeUIQM(img_original);
    uiqm_processed = computeUIQM(img_processed);
    %fprintf('Quality assessment results\n');
    % fprintf('PSNR: %.2f dB\n\n', psnr_value);
    %fprintf('UCIQE of original image: %.4f\n', uciqe_original);
    % fprintf('UCIQE for processed images: %.4f\n', uciqe_processed);
    %fprintf('UIQM of original image: %.4f\n', uiqm_original);
    %fprintf('UIQM for processed images: %.4f\n\n', uiqm_processed);
    fprintf(fileID, '%.4f,%.4f,%.4f\n',psnr_value,uciqe_processed,
        uiqm_processed);
end
fclose(fileID);

function uiqm = computeUIQM(img)
if size(img,3) ~= 3
    error('The input image must be an RGB image');
end
uicm = computeUICM(img);
uism = computeUISM(img);
uiconm = computeUIConM(img);
uiqm = 0.0282 * uicm + 0.2953 * uism + 3.5753 * uiconm;
end

function uicm = computeUICM(img)
R = img(:,:,1);
G = img(:,:,2);
B = img(:,:,3);
rg = R - G;
```

```
yb = 0.5 * (R + G) - B;
std_rg = std(rg(:));
std_yb = std(yb(:));
mean_rg = mean(rg(:));
mean_yb = mean(yb(:));
uicm = sqrt(std_rg^2 + std_yb^2) + 0.3 * sqrt(mean_rg^2 + mean_yb^2);
end

function uism = computeUISM(img)
sobel_x = [-1 0 1; -2 0 2; -1 0 1];
sobel_y = sobel_x';
gray = rgb2gray(img);
Ix = conv2(double(gray), sobel_x, 'same');
Iy = conv2(double(gray), sobel_y, 'same');
gradient_magnitude = sqrt(Ix.^2 + Iy.^2);
uism = mean(gradient_magnitude(:));
end

function uiconm = computeUIConM(img)
gray = rgb2gray(img);
uiconm = std(double(gray(:)));
end

function psnr_val = computePSNR(img1, img2)
if ~isequal(size(img1), size(img2))
    error('Both images must be the same size to calculate the PSNR.');
end
mse = mean( (img1(:) - img2(:)).^2 );
if mse == 0
    psnr_val = Inf;
else
    max_I = 1;
    psnr_val = 10 * log10( (max_I^2) / mse );
end
end

function uciqe = computeUCIQE(img)
```

```
if size(img,3) ~= 3
    error('The input image must be an RGB image');
end
lab = rgb2lab(img);
L = lab(:,:,1);
a = lab(:,:,2);
b = lab(:,:,3);
saturation = std([a(:), b(:)], 0, 1);
S = mean(saturation);
C = std(L(:));
sobel_x = [-1 0 1; -2 0 2; -1 0 1];
sobel_y = sobel_x';
Ix = conv2(double(L), sobel_x, 'same');
Iy = conv2(double(L), sobel_y, 'same');
gradient_magnitude = sqrt(Ix.^2 + Iy.^2);
H = mean(gradient_magnitude(:));
uciqe = 0.468 * S + 0.274 * C + 0.257 * H;
end
%imshow(dis),title('Processed image')
```

Listing 3: The Matlab Source code of Question 4

```
clc;
clear all;
close all;
fileID = fopen('q4.csv', 'w');
fprintf(fileID, 'PSNR,UCIQE,UIQM\n');
% figure;
% size = [800,1200];
hold on;
% num = 3;
for i = 1:12
    fileName = sprintf('test_%03d.png', i);
    A = imread(fileName);
    A_1 = uwredcomp(A);
    % A_1 = imresize(A_1,size);
    % subplot(2,num,i)
    % imshow(A)
```

```
    % subplot(2,num,i+num)
    % imshow(A_1)
    img_original = A;
    img_processed = A_1;
    % figure;
    % imshow(img_original);
    % title('original image');
    % figure;
    % imshow(img_processed);
    % title('processed image');
    img_original = im2double(img_original);
    img_processed = im2double(img_processed);
    uiqm_original = computeUIQM(img_original);
    uiqm_processed = computeUIQM(img_processed);
    psnr_value = computePSNR(img_original, img_processed);
    uciqe_original = computeUCIQE(img_original);
    uciqe_processed = computeUCIQE(img_processed);
    %fprintf('Quality assessment results\n');
    % fprintf('PSNR: %.2f dB\n\n', psnr_value);
    %fprintf('UCIQE of original image: %.4f\n', uciqe_original);
    % fprintf('UCIQE for processed images: %.4f\n', uciqe_processed);
    %fprintf('UIQM of original image: %.4f\n', uiqm_original);
    %fprintf('UIQM for processed images: %.4f\n\n', uiqm_processed);
    fprintf(fileID, '%.4f,%.4f,%.4f\n',psnr_value,uciqe_processed,
        uiqm_processed);
end
fclose(fileID);
% saveas(gcf,'q4.jpg')

function RGBnew = uwredcomp(RGB,varargin)
smoothing = 0;
kclip = 0.01;
gammaadj = 1;
alpha = 0.4;
if nargin>1
    k = 1;
    while k <= numel(varargin)
```

```
        thisarg = lower(varargin{k});
        switch thisarg
            case 'smoothing'
                smoothing = varargin{k+1};
                k = k+2;
            case 'kthresh'
                kclip = varargin{k+1};
                k = k+2;
            case 'gamma'
                gammaadj = varargin{k+1};
                k = k+2;
            case 'alpha'
                alpha = varargin{k+1};
                k = k+2;
            otherwise
                error('UWREDCOMP: unknown option %s',thisarg)
        end
    end
end
if smoothing ~= 0 && (ifversion('<','R2014a') || ~hasipt())
    error('UWREDCOMP: smoothing option requires imguidedfilter() from
        IPT (R2014a or newer)');
end
if size(RGB,3)~=3
    error('UWREDCOMP: expected INPICT to be RGB')
end
[RGB inclass] = imcast(RGB,'double');
lam = [620 540 450];
blam = (-0.00113*lam + 1.62517);
Blam = ctflop(quantile(RGB,1-0.001,[1 2]));
cgcr = (blam(2)*Blam(1))/(blam(1)*Blam(2));
cbcr = (blam(3)*Blam(1))/(blam(1)*Blam(3));
wrgb = [1 cgcr cbcr]/(1 + cgcr + cbcr);
Rnew = imappmat(RGB,wrgb);
if smoothing > 0
    Rnew = imguidedfilter(Rnew,RGB(:,:,2),'degree',smoothing);
end
```

```
RGBnew = RGB;
RGBnew(:,:,1) = Rnew;
inlim = stretchlimFB(RGBnew,kclip);
RGBcont = imadjustFB(RGBnew,inlim,[0 1],gammaadj);
RGBahq = zeros(size(RGBnew));
for c = 1:3
    RGBahq(:,:,c) =
        adapthisteqFB(RGBnew(:,:,c),'distribution','rayleigh');
end
RGBnew = RGBcont*alpha + RGBahq*(1-alpha);
RGBnew = imcast(RGBnew,inclass);
end
% To avoid repeating the explanations of functions like computeUIQM,
    computeUICM, computeUISM, computeUIConM, computePSNR, and
    computeUCIQE that have already been presented above in the paper,
    one can refer back to their initial definitions, highlight their
    specific purposes along with the required inputs and resulting
    outputs.
% imshow(dis),title('processed image')

figure;
size = [800,1200];
hold on;
num = 3;
for i = 1:num
    idx = [10,18,26];
    fileName = sprintf('image_%03d.png', idx(i));
    A = imread(fileName);
    A = imresize(A,size);
    A_1 = uwredcomp(A);
    A_1 = imresize(A_1,size);
    subplot(2,num,i)
    imshow(A)
    subplot(2,num,i+num)
    imshow(A_1)
end
saveas(gcf,'q4.jpg')
```