are hyper-parameters. As the underwater tends to be more blue or green owing the light refraction in the water. The first part is color correction. Calculating the mean and standard deviation of each channel and using parameter mu decides the maximum and minimum value of each channel separately. , . Then, based on the maximum and minimum, we tune the original RGB value by .

After the color correction, we decompose the original value , where S is the observed image, R is the reflectance, and I is illumination. , so, we have We initialize the and as the Gaussian low-pass filtered image of , to be 0. is the luminance layer of color corrected image.

Next, we are going to tune value and in sequence. Use matrix as the filters with to calculate . Update derivatives . Given updated , update by

where is Fast Fourier Transform operator, \* is the complex conjugate. All calculations are component-wise operators. In order to clip , .

To address the problem of under-exposure, a slight improved histogram specification is worked on the illumination . We Cumulative Density Functions (CDF) of the pixel value as the formula in the paper. We define another CDF , a distribution as the paper explains. To lighten dark regions and preserve naturalness to avoid over-enhancement, .

Finally, we get . Then the new Lab color space is transformed into RGB to acquire the final enhanced color image. We use the same parameter as the write used, .

*no* and *nr* denote respectively the cardinal numbers of the set of visible edges in the original image *Io* and in the contrast-restored image *Ir*. According to the paper 3, we propose to compute *e*, the rate of new visible edges in *Ir*: e=(nr-no)/ no The value of *e* evaluates the ability of the method to restore edges which were not visible in *Io* but are in *Ir*.

we propose to compute the number *ns* of pixels which are saturated (black or white) after applying the contrast restoration but were not before. We normalize this value by the size of the image, which gives the σ indicator: σ = *ns* /N, where N denotes the number of pixels in one channel.