Multilevel segmentation

Juan Antonio Pérez Juárez
University of Guadalajara
Guadalajara, Jalisco
juan.perez0996@alumnos.udg.mx

I. INTRODUCTION

Multilevel segmentation is a powerful technique in image processing that involves partitioning an image into multiple segments or regions, each representing different levels of detail and characteristics. This approach enhances the analysis of complex images by allowing for a hierarchical representation, which is crucial in applications such as medical imaging, remote sensing, and object detection. By utilizing various methodologies, including graph-based methods and machine learning algorithms, multilevel segmentation enables more accurate feature extraction and improves the performance of subsequent image analysis tasks. As the demand for precise image interpretation grows, multilevel segmentation stands out as an essential tool in the field of computer vision.

II. MULTI-LEVEL THRESHOLD IMAGE SEGMENTATION

Image segmentation divides an image into a set of non-overlapping regions, each with different features, so that some interesting objects are highlighted. This paper concentrates on pixel-based image segmentation, accomplished by analyzing image features and pixel distances.

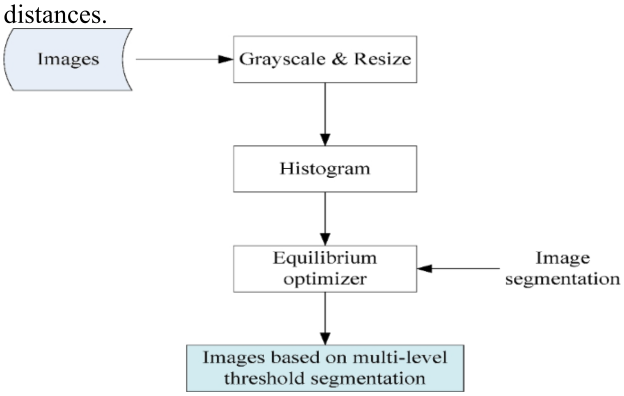

Fig. 1 illustrates the specific image segmentation process.

III. OBJECTIVE FUNCTION

Cross-entropy is a mathematically sound and theoretically justified objective function used in multi-threshold image classification that is directly aimed at reducing classification errors. Compared to other criteria, cross-entropy effectively manages probability distributions and adapts thresholds according to local and global environments. By minimizing the difference between predicted and actual class distributions, cross-entropy ensures that the algorithm learns to classify pixels more accurately, so it is particularly well-suited for complex classification tasks such as multi-threshold image processing.

We utilize the minimization of cross-entropy as the segmentation criterion and introduce the concept of selecting multiple thresholds in grayscale images. Since image histograms can contain valleys and wide peaks with different heights, the cross-entropy method addresses these issues by measuring the uniformity of histogram information between the original and segmented images.

A lower cross-entropy value indicates lower uncertainty and higher uniformity between the original and thresholded images. Let I be the original image and h(i), i = 1, 2, ..., L be the corresponding histogram. The threshold (th) is used to construct the thresholded image (I_{th}):

$$I_{th}(x,y) = \mu(k,m) \qquad if(k < I(x,y) <= m)$$

$$\mu(k,m) = \frac{\sum_{i=k}^{m-1} ih(i)}{\sum_{i=k}^{m-1} h(i)}$$

Where I(x, y) is the gray level of the pixel at coordinates (x, y), and k and m are threshold values.

IV. ADVANCED EQUILIBRIUM OPTIMIZER

Meta-heuristic algorithms are required to balance exploration and exploitation in multi-level threshold segmentation of images. Exploitation results in the algorithms failing to fully cover the search space and reduces population diversity. On the other hand, exploration causes a slow convergence of them and impedes the ability to find the optimal solution in potential threshold areas. We employ a multi-population EO algorithm to segment images with multi-thresholds. The population is split into two sub-populations: one for exploration and the other for exploitation. This method increases the search space coverage and prevents the algorithm from prematurely converging to a local optimum. The sub-populations regularly share information and combine their findings, thereby improving the global search capability and search efficiency. This multi-population strategy increases the probability of finding the global optimal threshold, and fig. 2 describes the pseudo code of AEO.

V. EQUILIBRIUM OPTIMIZER

In the original EO, position updates are performed using the equilibrium pool and Equation

```
Initialize the population
Randomly divide the population into two sub-populations
while (iteration) {
    if (The first sub-population){
        if (mutation) {
            Execute Algorithm 3
        } else{
            Execute Equation (7)
        }
        Execute Algorithm 4
        Execute the objective function
    } else{
        if (mutation) {
            Execute Algorithm 2
        } else{
            Execute Equation (13)
        }
        Execute Algorithm 4
        Execute the objective function
    }
    Obtain the global solution
}
Output the global solution
```

Fig. 2 Example of Advanced Equilibrium optimizer.

$$X_i(it+1) = X_{pool}(it) + (X_i(it) - X_{pool}(it))F + \frac{G}{\lambda}(1-F)$$

$$t = (1 - \frac{it}{MAX\_IT})^{(2\frac{it}{MAX\_IT})}$$

$$F = sign(r-0.5)[e^{-\lambda t} - 1].$$

$$GCP = \begin{cases} 0.5r_1 & if(r_2 \geq 0.5) \\ 0 & else \end{cases}$$

$$G = F * GCP * (X_{pool} - X_i)$$

Where X_i, implies the position of individual i, and it represents the current iteration.

MAX_IT represents the maximum iteration. Sign is the signum function of Matlab. λ, r, r_1, r_2 are random values between [0,1]. The equilibrium pool consists of four optimal solutions and their average position (x_{avg}) and (x_{pool}) is selected randomly from the pool.

VI. MUTATION

Mutation is instrumental in EO. It increases population diversity by introducing random changes and prevents EO from converging to local optimal solutions too early. A larger step size aids in exploring new areas of the search space and enhancing global search ability, while a smaller step size helps in refining the search and improving the accuracy of local exploitation. If the four optimal solutions remain unchanged after 10 iterations, P1 has not yet discovered the potential optimal solution area. It needs to be forced P1 to change the search area through mutation. P2 must fine-tune its search direction to find the optimal solution when it is not updated in 10 iterations. P2 follows the mutation process described in Fig. 3, while P1 mutates in the manner shown in Fig. 4

```
Input: X_i
Output: X_i
for (i = 1 : nTh) {
    if (rand() > 0.5) {
        X_i^j = X_i^j + sign(rand() - 0.5) * randi([1,nTh]);
    }
}
```

Fig. 3 Mutation 1

```
Use nTh to spatially partition the histogram into uniform equal parts a
if (the thresholds are all exactly within this interval) {
    Execute Mutation1;
}
else {
    for (Traverse a without thresholds existence) {
        Randomly generate a value b for this part
        Set the threshold for repeated parts to b
    }
}
```

Fig. 4 Mutation 2.

PSO focuses more on local search exploitation, while FA excels in global exploration. The hybrid algorithm is not always dynamically adjustable, and in some cases, it can result in premature convergence or inefficient exploration. AEO's two mutations provide a consistent balance and robustness in finding high-quality solutions, especially for complex problems such as image segmentation, where local fine-tuning and global exploration are necessary. The mutation operation perturbs certain parts of the existing solutions, so that AEO explores new areas in the search space for improving global search capability and solution quality. The algorithm continuously discovers and optimizes solutions during the iteration process.

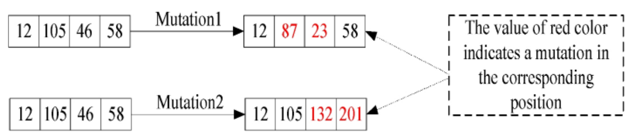

Fig. 5 shows examples of two mutations.

VII. REPAIRING SOLUTIONS

In the process of multi-threshold image segmentation, the presence of the same threshold will lead to loss of detail information and unstable segmentation. Segmentation algorithms cannot accurately distinguish between different regions and often ignore important features. However, duplication may occur since the threshold for image segmentation is constrained to the range of 1 to 255. To address this issue, we propose a method for repairing the solutions, as shown in Fig. 6.

```
Input: X_i
Output: X_i
new_pos = X_i;
[uniqueElements, ~, indexInUnique] = unique(pos);
if (length(uniqueElements) < length(pos)) {
    First_idx= accumarray(indexInUnique(:), (1:numel(pos)).', [], @min);
    duplicateIndices = find(ismember(indexInUnique, find(histc(indexInUnique, 1:numel(uniqueElements)) > 1)));
    result= First_idx(histc(indexInUnique, 1:numel(uniqueElements)) > 1);
    difference = setdiff([1 : dim], pos);
    k = length(duplicateIndices) - length(result);
    indices = randi(length(difference), 1, k);
    a_diff = setdiff(duplicateIndices, result);
    new_pos(a_diff) = difference(indices);
}
X_i = new_pos;
```

Fig. 6 Repairing operation.

VIII. EXPERIMENTAL ANALYSIS ON BENCHMARK IMAGES

PSNR is a metric used to evaluate the quality of a reconstructed image compared to its original version. It is particularly useful in image compression, segmentation, and other image processing applications. PSNR is expressed in decibels (dB) and can be calculated using the following.

$$PSNR = 10log_{10}(\frac{L}{MSE})$$

$$MSE = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [I(i,j) - I_{th}(i,j)]^2$$

where M and N represent the length and width pixels of an image.

SSIM is used to measure the similarity of two images.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

Where μ_x and μ_y are the averages of x and y, respectively.
σ_x^2 and σ_y^2 are the variances of x and y, respectively. σ_{xy} is the covariance of x and y, respectively.
C_1 *and* C_2 are small constants to stabilize the division with a weak denominator. Typically, $C_1 = (k_1L)^2$ and $C_2 = (k_2L)^2$ with $k \approx 0.01$ and $k1, \approx 0.01$ and $k2, \approx 0.03$.
SSIM evaluates the performance of image segmentation algorithms in preserving image structure, brightness, and contrast. By comparing local regions between segmented images and original images, it quantifies the similarity in texture, edges, and detail retention. High SSIM values indicate that segmentation algorithms effectively retain the original features of images.

FSIM combines phase congruency (PC) and gradient magnitude to evaluate the similarity between two images.

$$FSIM(x,y) = \frac{\sum_i S_L(i) \cdot S_P(i) \cdot W(i)}{\sum_i W(i)}$$

where $S_L(i)$ is the similarity measure based on luminance, $s_P(i)$ is the similarity measure based on phase congruency, and $\omega(i)$ is the weight assigned to each pixel based on its significance.
FSIM assesses the performance of image segmentation algorithms in retaining image details and edge information by comparing the phase congruency and gradient magnitude between the original and segmented images. High FSIM values indicate excellent performance in these aspects.

CONCLUSIONS

In conclusion, multilevel segmentation represents a significant advancement in the field of image processing, providing a more nuanced and comprehensive approach to image analysis. By enabling the segmentation of images at various levels of detail, this technique enhances the ability to identify and extract meaningful features, thereby improving the accuracy and effectiveness of subsequent tasks such as classification and recognition. As technology continues to evolve, the integration of multilevel segmentation with emerging methodologies, including deep learning and artificial intelligence, promises to further enhance its applications across diverse fields. Continued research in this area is essential for unlocking new potentials in image analysis and interpretation.

REFERENCES

[1] Hu, P., Han, Y., Zhang, Z., Chu, S., & Pan, J. (2024). A multi-level thresholding image segmentation algorithm based on equilibrium optimizer. Scientific Reports, 14(1), 29728. https://doi.org/10.1038/s41598-024-81075-w

[2] Pramanik, P., Mukhopadhyay, S., Kaplun, D. & Sarkar, R. A deep feature selection method for tumor classification in breast ultrasound images. In International Conference on Mathematics and Its Applications in New Computer Systems. 241–252 (Springer, 2021).

[3] Zhou, G. et al. Orthorectification of fisheye image under equidistant projection model. Remote Sens. 14, 4175 (2022).

[4] Huang, T., Yin, H. & Huang, X. Improved genetic algorithm for multi-threshold optimization in digital pathology image segmentation. Sci. Rep. 14, 22454 (2024).