The Best Symmetry by Finding the Optimal Clustering Filters for Specific Lighting Conditions

Volodymyr Hrytsyk *, Anton Borkivskyi, Taras Oliinyk

Posted Date: 16 July 2024

doi: 10.20944/preprints202407.1309.v1

Keywords: computer vision; adaptive systems; pattern recognition estimation

 doi:10.20944/preprints202407.1309.v1

Article

The Best Symmetry by Finding the Optimal Clustering Filters for Specific Lighting Conditions

Volodymyr Hrytsyk *, Taras Oliinyk and Anton Borkivskyi

Department of Automated Control Systems, Lviv Polytechnic National University, 79013 Lviv, Ukraine
* Correspondence: volodymyr.v.hrytsyk@lpnu.ua

Abstract: This article explores the efficiency of various clustering methods for image segmentation under different luminosity conditions. Image segmentation plays a crucial role in computer vision applications, and clustering algorithms are commonly used for this purpose. The search for an adaptive clustering mechanism aims to ensure maximum symmetry of real objects with objects/segments in their digital representations. However, clustering methods performance can fluctuate with varying lighting conditions during image capture. Therefore, we assess the performance of several clustering algorithms — including k-means, K-Medoids, fuzzy c-means, Possibilistic C-Means, Gustafson-Kessel, Entropy-based Fuzzy, Riddler-Calvard, Kohonen Self-Organizing Maps, and Meanshift — across images captured under different illumination conditions. Additionally, we develop an adaptive image segmentation system utilizing empirical data. Conducted experiments highlight varied performance among clustering methods under different luminosity conditions. This research enhance better understanding of luminosity's impact on image segmentation and aiding in method selection for diverse lighting scenarios.

Keywords: computer vision; adaptive systems; pattern recognition estimation

1. Introduction

The constant increase in requirements for the image processing and the growing complexity of the tasks to be solved have made pattern recognition one of the major trends that has persisted for decades. If at the dawn of computer processing, the task was to highlight an image on a binary field, now it is necessary not only to perform a simple semantic segmentation, but also to solve pattern recognition of 3D tasks [1], smart city tasks [2] or medical areas [3], which are partially solved problems. This study examines the characteristics of the functioning of adaptive systems to assess the efficacy of diverse clustering methods in image segmentation for automated object extraction across varying lighting conditions. This is crucial as the visual data processed by computers is pivotal for tasks such as classification, identification, and verification. Therefore, segmentation is an important stage of image processing, which consists of dividing the image into separate segments or regions corresponding to different objects in the image. Clustering methods can be used for automatic identification of object groups in images based on their properties, such as colors, textures, shapes, etc [4]. In prior studies, upon which the foundation is based [5,6], one of the authors examined the impact of evaluating line detection methods across varied lighting conditions. In [5], an external sensor was used to estimate the number of lux. In this study, the authors introduce an integrated approach where the camera's collected data serves as a simultaneous source for lux meter readings. This decision aims to enhance the dependence of the objective assessment by integrating lighting adaptation procedures seamlessly. While [5] employed MSE/PSNR for evaluation, this study adopts the Structural Similarity Index (SSIM) as a more precise metric. SSIM evaluates similarity between images by considering brightness, color, and structural differences [7]. SSIM values range from -1 to 1, with 1 indicating identical images and -1 indicating complete dissimilarity. Typically, an SSIM above 0.9 signifies high image similarity. SSIM computes three components: luminance, contrast, and

 doi:10.20944/preprints202407.1309.v1

structure. Luminance assesses the overall brightness of the image, contrast measures variation across the image, and structure takes into account the interaction between different parts of the image.. By comparing histograms of these components, SSIM identifies image regions with the greatest disparities. In [5], adaptive approach to line recognition was implemented, while this study focuses on evaluating common clustering methods. The authors used the SSIM method instead of artificial neural networks [8] since the main goal of the work is the comparative evaluation of clustering methods. Using neural networks [9] as a classifier would necessitate training and stability considerations.

The following clustering methods were selected for the study:

1. K-Means
2. K-Medoids
3. Fuzzy C-Means
4. Possibilistic C-Means
5. Fuzzy Possibilistic C-Means
6. Possibilistic Fuzzy C-Means
7. Gustafson-Kessel
8. Entropy-based Fuzzy
9. Riddler-Calvard
10. Kohonen Self-Organizing Maps
11. MeanShift

Clustering is the process of grouping objects into subsets or clusters based on similarities between them. Each cluster contains objects that are more similar to each other than to objects from other clusters. Similarity can be determined using various metrics such as Euclidean distance, cosine similarity, correlation, etc.

2. Materials and Methods

The purpose of the work is to develop an adaptive system for choosing segmentation methods depending on external conditions (in particular, the level of illumination of the field of attention).

Experimental design:

1. Creating a database of benchmarks of 9 classes using the global threshold method, which should reflect the approximation of human vision, since the threshold is chosen by a person with a subjective assessment of the result.

2. Objective assessment of the effectiveness of clustering methods under the given conditions. Clustering methods are used to extract objects from images, the effectiveness of which is assessed by calculating the difference between extracted objects and benchmarks using SSIM. Note that the results are averaged taking into account the accuracy and number of objects seen.

3. The results in the tables are averaged over three experiments (3 frames in a row are evaluated independently and the result is the average value). Note that there are no significant jumps in values between individual experiments. Therefore, there is no influence of a single experiment on the selection of the final winning method.

2.1. K-Means

The k-means clustering algorithm is one of the simplest unsupervised learning algorithms for solving the well-known clustering problem. The term "k-means" was first used by James McQueen in 1967 [1], using an idea proposed by Hugo Steinhaus in 1957 [2].

Let $X = \{x_1, x_2, \dots x_n\}$ – be the set of data points, and $c = \{c_1, c_2, \dots c_k\}$ – be the set of cluster centers. The *k*-means clustering algorithm attempts to partition (or cluster) n data points into k disjoint subsets C_j containing the data points in such a way as to minimize the sum-of-squares criterion:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i - c_j \right\|^2, \quad (1)$$

 doi:10.20944/preprints202407.1309.v1

here x_i – is a sample in the data set, and c_j – is the geometric centroid of the data points in the cluster C_j. Clustering is performed by minimizing the sum of squared distances between data points and the corresponding cluster center.

An implementation from the opencv-python library was used for the experiment [3].

2.2. K-Medoids

The k-medoids algorithm, introduced by Leonard Kaufman and Peter Rousseau along with their PAM algorithm [4], is a clustering technique akin to the k-means method. Both algorithms involve partitioning the dataset into groups, aiming to minimize the distance between the data points assigned to a cluster and the designated center point of that cluster. However, there are notable distinctions between them.

Unlike the k-means algorithm, which selects the average value of points within a cluster as its center point, k-medoids opt for actual data points (referred to as medoids or samples) as the cluster centers. This characteristic enhances the interpretability of cluster centers, as they directly correspond to existing data points. Furthermore, k-medoids offer the flexibility of utilizing various distance measures, while k-means typically rely on the Euclidean distance for efficient solutions.

One advantageous aspect of the k-medoids algorithm is its robustness to noise and outliers. By minimizing the sum of pairwise differences rather than the sum of squared Euclidean distances, k-medoids exhibit greater resilience in aberrant data points. This sets it apart from k-means, making it a valuable tool in scenarios where noise and outliers are prevalent.

An implementation from the scikit-learn-extra library was used for the experiment [5].

2.3. Fuzzy C-Means (FCM)

The FCM algorithm belongs to fuzzy (soft) clustering methods, which is a form of clustering in which each data point can belong to more than one cluster.

Fuzzy c-means clustering was developed by James Dunn in 1973 [6] and improved by James Bezdek in 1981 [7].

Suppose that it is necessary to cluster n×m-dimensional data points represented by x_i (i = 1, 2,...,n).

The algorithm returns a list of c cluster, enters $C = \{c_1, c_2, \dots, c_c\}$ and a partition matrix $U = [u_{i,j}] \in [0,1],\ i = 1,2, \dots, k, j = 1,2, \dots,$ n, where $u_{i,j}$ indicates the degree of belonging of the element x_i to cluster c_j. Here $u_{i,j} \in [0,1]$ and $\sum_{j=1}^{k} u_{i,j} = 1, \forall\, i$. The FCM algorithm is aimed at minimizing the current objective function:

$$J_m = \sum_{j=1}^{k} \sum_{i=1}^{n} u_{i,j}{}^m \left\| x_i - c_j \right\|^2, \quad (2)$$

where m – parameter of the fuzzy partitioning of the matrix.

An implementation from the fuzzy-c-means library was used for the experiment [8].

2.4. Possibilistic C-Means (PCM)

To prevent outliers, another clustering technique was proposed by Krishnapuram and Keller (1993), called PCM [9]. In contrast to the FCM algorithm, the membership value generated by the PCM algorithm can be interpreted as "the degree of membership or compatibility or typicality" (Krishnapuram and Keller, 1993). Degrees of typicality are determined to construct prototypes that characterize subcategories of data, taking into account both the common features of category members and their distinctive features compared to other categories. compared to other categories. Typical values about one cluster do not depend on other clusters' prototypes. The degree of typicality helps distinguish between a very atypical and a partially atypical members of a cluster [10].

The PCM algorithm relaxes the row sum constraint of the FCM algorithm. The main limitation of the PCM algorithm is that each membership value in U can be anything between 0 and 1 or equal to any of them, i.e., that is $0 \leq u_{i,j} \leq 1$. So, these values are called the typical characteristics of the data points in each cluster. The objective function of the PCM algorithm can be formulated as follows:

 doi:10.20944/preprints202407.1309.v1

$$J_m = \sum_{j=1}^{k}\sum_{i=1}^{n} u_{i,j}^m d_{i,j}^2 + \sum_{j=1}^{k}\eta_j \sum_{i=1}^{n}\left(1-u_{i,j}\right)^m, \quad (3)$$

where n – the total number of samples in a given data set; c – number of clusters; m – a parameter that determines the degree of blurring of the partition; $d_{i,j}^2$ – distance; $U = [u_{i,j}]$ – fuzzy matrix partitioning.

η_i, is called the scale or typicality parameter and is calculated from the data with the following formula:

$$\eta_j = \frac{\sum_{i=1}^{n} u_{i,j}^m \|x_i - c_j\|^2}{\sum_{i=1}^{n} u_{i,j}^m}, \quad (4)$$

where n – the total number of samples in a given data set; $m \in [1, \infty)$ – it is a parameter that determines the degree of blurring of the partition; $X = \{x_1, \ldots, x_n\}$ and $C = \{c_1, \ldots, c_c\}$ – data attributes and cluster centroids; $U = [u_{i,j}]$ – fuzzy partitioning of the $k * n$ matrix, consisting of degrees of membership of the sample x_i to each cluster j.

The membership of $u_{i,j}$ value, in the case of the PCM algorithm, is calculated from the following formula:

$$u_{i,j} = \left[1 + \frac{d_{i,j}^{2\frac{1}{m-1}}}{\eta_j}\right]^{-1}, \quad (5)$$

where $d_{i,j}^2$ – distances; η_j – scale parameter.

An implementation from the scikit-*c*-means library was used for the experiment [11].

2.5. Possibilistic Fuzzy C-Means (PFCM)

To obtain a stronger candidate for fuzzy clustering, Pal, Pal, Keller, and Bezdek proposed the PFCM algorithm in 2005 [12]. The PFCM algorithm can avoid overlapping clusters and at the same time is less sensitive to outliers (Pal et al. 2005). The PFCM algorithm uses a combination of the objective functions of the PCM and FCM algorithms. The objective function of the PFCM algorithm is:

$$J_{m,\eta} = \sum_{j=1}^{k}\sum_{i=1}^{n}(au_{i,j}^m + bt_{i,j}^{\eta})\left\|x_i - c_j\right\|^2 + \sum_{j=1}^{k}\delta_j\sum_{i=1}^{n}(1-t_{i,j})^{\eta}, \quad (6)$$

Provided that

$$\sum_{j=1}^{k} u_{i,j} = 1, \forall\, i,$$

$$a > 0, b > 0, m > 1, \eta > 1, 0 \le u_{i,j}, t_{i,j} \le 1$$

The relative significance between membership values and typicality values is determined by parameters a and b (Timm et al., 2004) [13].

Objective function $J_{m,\eta}$ is minimized by $d_{i,j} = \left\|x_i - c_j\right\|^2 > 0$, $\forall$ *i,j*, m and $\eta > 1$, and X containing at least k different data.

The degree of belonging is updated according to the following formula:

$$u_{i,j} = \left[\sum_{l}^{k} \frac{d_{i,j}^{2\frac{1}{m-1}}}{d_{i,l}^2}\right]^{-1}, 1 \le j \le k; 1 \le i \le n, \quad (7)$$

The value of typicality is according to the following formula:

$$t_{i,j} = \frac{1}{1+\left(\frac{b\left\|x_i - c_j\right\|^2}{\delta_j}\right)^{1/(\eta-1)}}, 1 \le j \le k; 1 \le i \le n, \quad (8)$$

Prototypes are based on the following formula:

 doi:10.20944/preprints202407.1309.v1

$$c_j = \frac{\sum_{i=1}^{n}\left(au_{i,j}^{m}+bt_{i,j}^{\eta}\right)x_i}{\sum_{i=1}^{n}\left(au_{i,j}^{m}+bt_{i,j}^{\eta}\right)}, 1 \le j \le k \ , \quad (9)$$

An implementation presented in [14] was used for the experiment.

2.6. Fuzzy Possibilistic C-Means (FPCM)

Fuzzy Possibilistic c-Means (FPCM) is an extension of the classic Fuzzy c-Means (FCM) clustering algorithm. Similar to FCM, FPCM is a soft clustering algorithm that assigns to each data point several clusters with different degrees of membership. However, unlike FCM, FPCM allows you to take into account additional uncertainty in the clustering process by introducing a possible term to the objective function.

In FPCM, each data point is represented by a vector of membership values, where each value reflects the degree to which the point belongs to a certain cluster. The possibility term of the objective function suggests that a data point may not belong to any cluster, not with absolute certainty, but with some degree of possibility. This allows FPCM to better handle noise and outliers in the data compared to FCM.

The objective function of the FPCM algorithm includes degrees of membership and typicality as shown in the following equation:

$$J_{m,\eta} = \sum_{j=1}^{k}\sum_{i=1}^{n}(u_{i,j}^{m} + t_{i,j}^{\eta})\left\|x_i - c_j\right\|^2, \quad (10)$$

Provided that

$$\sum_{j=1}^{k} u_{i,j} = 1, \forall\, i,$$

$$\sum_{i=1}^{n} t_{i,j} = 1, \forall\, j,$$

$$m > 1, \eta > 1, 0 \le u_{i,j}, t_{i,j} \le 1$$

where m and η exponents of vagueness and typicality. Taking into account the given restrictions and optimization conditions of c-means $\sum_{j=1}^{k} u_{i,j} = 1$, we determine the following initial conditions or extrema of the objective function in terms of the theorem of Lagrange multipliers:

$$u_{i,j} = \left[\sum_{l}^{k} \frac{d_{i,j}^{2\frac{1}{m-1}}}{d_{i,l}^{2}}\right]^{-1}, 1 \le j \le k; 1 \le i \le n, \quad (11)$$

$$t_{i,j} = \left(\sum_{l}^{n}\left(\frac{d_{i,j}^{2}}{d_{l,j}^{2}}\right)^{2/(\eta-1)}\right)^{-1}, \forall\, i,j \quad (12)$$

$$c_j = \frac{\sum_{i=1}^{n}\left(u_{i,j}^{m}+t_{i,j}^{\eta}\right)x_i}{\sum_{i=1}^{n}\left(u_{i,j}^{m}+t_{i,j}^{\eta}\right)}, \forall\, i \ . \quad (13)$$

2.7. Gustafson-Kessel (GK)

The Gustafson-Kessel (GK) algorithm is a clustering algorithm that extends the well-known fuzzy *c*-means (FCM) algorithm to handle data with different cluster shapes and sizes. It was proposed by Dr. David Gustafson and William Kessel in 1979 [15].

The algorithm returns a list of k clusters with centers $C = \{c_1, c_2, \ldots, c_k\}$. The main feature of the GK algorithm is the local adaptation of the distance metric to the cluster shape by estimating the cluster covariance matrix and the corresponding adaptation of the distance matrix. The objective function for the GK algorithm is defined as

$$J_m = \sum_{j=1}^{k}\sum_{i=1}^{n}\left(u_{i,j}\right)^{m} d_{i,j}^{2}, \quad (14)$$

 doi:10.20944/preprints202407.1309.v1

In this algorithm, each cluster is associated with a separate matrix A_j. Matrices A_j are used as optimization variables in the c-means functional, thus allowing each cluster to adapt the distance norm to the local topological structure of the data. The distance between the data point x_i and the center of the cluster c_j is

$$d_{i,j}^2 = (x_i - c_j)^T A_j (x_i - c_j), \forall\, i = 1,2,\dots,n, \forall\, j = 1,2,\dots,c\ , \tag{15}$$

This objective function cannot be directly minimized concerning A_j, because it is linear concerning A_j. To obtain a feasible solution, A_j must be bounded in some way. A common way to achieve this is to restrict the determinant of A_j:

$$|A_j| = \rho_j, \rho_j > 0, \forall\, j, \tag{16}$$

The coefficient ρ_j determines the volumes of individual clusters (if we do not know about the problem, we can assume $\rho_j = 1$). Using the method of Lagrange multipliers, the following expression for A_j was obtained

$$|A_j| = [\rho_j\, det(F_j)]^{1/n} F_j^{-1}, \tag{17}$$

where F_j, the so-called fuzzy covariance matrix of jth cluster is obtained from the formula:

$$F_j = \frac{\sum_{i=1}^{n} (u_{i,j})^m (x_i - c_j)(x_i - c_j)^T}{\sum_{i=1}^{n} (u_{i,j})^m}\ , \tag{18}$$

The initialization of the algorithm requires the definition of the same parameters as in the FCM algorithm. The GK algorithm finds clusters of any shape but requires more calculations than the FCM algorithm due to the need to calculate the determinant and the inverse matrix F_j at each iteration.

An implementation presented in [16] was used for the experiment.

2.8. Entropy-Based Fuzzy (EBF)

Yao et al. in 2000 presented an entropy-based fuzzy clustering algorithm [17]. In this algorithm, the entropy values of the data points are first calculated. Then the data point with the minimum entropy value is selected as the center of the cluster. Data points that are not chosen in any of the clusters are called outliers. Consider a set X of N data points in an M-dimensional hyperspace, where each data point $x_i\ (i = l, 2, \dots, N)$ is represented by a set of M values (i.e., $x_{i1}, x_{i2}, x_{i3}, \dots x_{iM}$). Thus, the data set can be represented by an $N \times M$ matrix. The values of each dimension are normalized in the range [0.0 – l.0]. The Euclidean distance between any two data points (for example, i and j) is defined as follows:

$$d_{i,j} = \sqrt{\sum_{k=1}^{M} (x_{i,k} - x_{j,k})^2}, \tag{19}$$

The entropy value between two data points is in the range [0.0 – 1.0]. It is very small (close to 0.0) for very close or very distant pairs of data points and very high (close to 1.0) for those data points separated by a distance close to the average distance of all pairs of data points.

The total entropy value at data points x_i relative to all other data points is calculated as

$$E_i = -\sum_{j \in X}^{j \neq i} \left(S_{i,j}\, log_2 S_{i,j} + (1 - S_{i,j}) log_2 (1 - S_{i,j})\right), \tag{20}$$

where $S_{i,j}$ – similarity between x_i and x_j, is normalized on the [0.0 – l,0] interval. During clustering, the data point with the minimum entropy value is selected as the center of the cluster. The similarity between any two points (i.e., i and j) can be calculated as follows:

$$S_{i,j} = e^{-\alpha d_{i,j}}\ , \tag{21}$$

 doi:10.20944/preprints202407.1309.v1

where α – a numerical constant. Experiments with different values for α show that it should be robust for all types of data sets, not just for certain data sets. The α value is calculated based on the assumption that the similarity value $S_{i,j}$ is set to 0.5, when the distance between two data points $d_{i,j}$ is equal to the average distance $\bar{d}$, which is calculated as

$$\bar{d} = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j>i}^{n} d_{i,j}\,. \tag{22}$$

From (21), we can calculate α as

$$\alpha = -\frac{\ln 0.5}{\bar{d}}. \tag{23}$$

So, α is determined by the data and can be calculated automatically.

2.9. Ridler-Calvard (RC)

The Ridler-Calvard method [18] – is a method for determining the threshold value of an image, which is a process of converting a grayscale image into a binary image by dividing pixels into two groups: pixels that exceed a certain threshold value and those that are below it.

The method is based on the idea of maximizing the interclass variance of two groups of pixels. Interclass variance is a measure of how well two groups are separated from each other. The threshold value that maximizes this variance is chosen as the optimal threshold value.

The Riedler-Calvard method begins by assuming an initial threshold value and computing the average values of pixels above and below the threshold. After that, it iteratively adjusts the threshold value based on the average values until the difference between them is minimized.

The foreground and background cluster values are given as m_f and m_b, respectively, and are defined mathematically as:

$$m_f(T_n) = \sum_{g=0}^{T_n} gp(g)\,, \tag{24}$$

$$m_b(T_n) = \sum_{g=T_n+1}^{L-1} gp(g)\,, \tag{25}$$

where g – gray level values $g = \{0,1,2,\ldots\ldots,L-1\}$, $p(g)$ – is the gray-level probability mass function (PMF) of g. The PMF is calculated from the image histogram by normalizing it to the total number of samples.

$T_n + 1$ – the new threshold value is calculated by averaging m_f and m_b as

$$T_{n+1} = \frac{m_f(T_n) + m_b(T_n)}{2}, \tag{26}$$

These operations are repeated until the difference value is less than the given value of ε.

An implementation from the Mahotas library was used for the experiment [19].

2.10. Kohonen Self-Organizing Maps (SOM)

The Self-Organizing Map (SOM) is a specific type of artificial neural network that differs from other neural networks in its training approach proposed by Teuvo Kohonen [20]. Instead of employing error-correcting learning methods like backpropagation with gradient descent, SOM utilizes concurrent learning.

Similar to most artificial neural networks, self-organizing maps operate in two distinct modes: learning and mapping. During the learning phase, a set of input data, known as the "input space," is utilized to construct a reduced-dimensional representation called the "map space." This mapping process enables the classification of additional input data using the generated map.

 doi:10.20944/preprints202407.1309.v1

The map space is composed of components referred to as "nodes" or "neurons," arranged in a two-dimensional hexagonal or rectangular grid. The number and specific locations of these nodes are predetermined based on the desired objectives of the data analysis and research.

Each node in the map space is associated with a "weight" vector, representing its position in the input space. While the nodes in the map space remain fixed, the learning process entails adjusting the weight vectors towards the input data, typically by reducing a distance metric like Euclidean distance. Importantly, this adjustment must not disrupt the topology established by the map space.

Following the training phase, the map can be employed to classify additional observations from the input space. This is achieved by identifying the node with the closest weight vector (i.e., the smallest distance metric) to the input space vector.

The primary objective of self-organizing map learning is to induce similar responses to specific input patterns across different parts of the network. This phenomenon partly mirrors the processing of visual, auditory, or sensory information in specific regions of the human cerebral cortex.

The weights of the neurons are initialized either with small random values or by uniformly selecting values within the subspace spanned by the two largest eigenvectors of the principal components. The latter alternative leads to faster learning since the initial weights provide a reasonable approximation of the SOM weights.

To train the network effectively, a considerable number of example vectors, ideally representing the expected vector types during mapping, are fed into the network. These examples are often introduced multiple times through iterations.

During training, when an example is presented to the network, its Euclidean distance to all weight vectors is computed. The neuron with the weight vector most similar to the input is designated as the "best-matching unit" (BMU). The weights of the BMU and the neurons in proximity to it in the SOM grid are adjusted based on the input vector. The magnitude of this adjustment decreases over time and with increasing distance from the BMU. The update formula for neuron v ith weight vector $W_v(s)$ is calculated accordingly.

$$W_v(s+1) = W_v(s) + \theta(u, v, s) * \alpha(s) * (D(t) - W_v(s)), \quad (27)$$

where s – step index, t – index into the training sample, u – the BMU index for the input vector $D(t)$, $\alpha(s)$ – monotonically decreasing learning rate; $\theta(u, v, s)$ – a neighborhood function that defines the distance between neuron u and neuron v at step s.

The neighborhood function, denoted as θ(u, v, s) or the lateral interaction function, plays a vital role in the self-organizing map. It depends on the distance between the best matching unit (BMU) neuron u and neuron v within the grid. The simplest form of the neighborhood function assigns a value of 1 to neurons that are close enough to the BMU and 0 to others. However, Gaussian functions and Ricker wavelets are also commonly used alternatives. Regardless of the specific form chosen, the neighborhood function gradually decreases over time.

During the initial stages when the neighborhood is broad, self-organization occurs on a global scale. As the neighborhoods shrink to pairs of neurons, the weights start to converge toward local estimates. In some implementations, both the learning coefficient α and the neighborhood function θ decrease gradually as the parameter s increases. In other cases, particularly when the training data set is traversed by the parameter t, the decrease occurs stepwise, once every T steps. This iterative process is repeated for each input vector over a typically large number of λ cycles. Ultimately, the network associates the output nodes with groups or patterns present in the input data set. If these patterns are identifiable, their names can be linked to the corresponding nodes in the trained network.

During the mapping phase, a single winning neuron is determined—the neuron whose weight vector is closest to the input vector. This determination can be made by simply calculating the Euclidean distance between the input vector and the weight vector.

An implementation from the sklearn-som library was used for the experiment [21].

2.11. MeanShift

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2024 doi:10.20944/preprints202407.1309.v1

MeanShift is a clustering algorithm that assigns data points to clusters iteratively by shifting the points towards the mode (the mode is the highest density of data points in a region in the context of MeanShift). Therefore, it is also known as the fashion search algorithm [22].

We start with an initial estimate of x. Let the kernel function $K(x_i - x)$ is given. This function determines the weight of the closest points to re-estimate the mean. A Gaussian kernel at the distance to the current estimate is usually used:

$$K(x_i - x) = e^{-c\|x_i - x\|^2}, \quad (28)$$

The weighted average value of the density in the window defined by K is calculated as:

$$m(x) = \frac{\sum_{x_i \in N(x)} K(x_i - x) x_i}{\sum_{x_i \in N(x)} K(x_i - x)}, \quad (29)$$

where $N(x)$ is a neighborhood of x, a set of points for which $is\ K(x_i - x) \neq 0$.

Difference $m(x) - x$ is called the mode shift (mean shift) according to Fukunaga and Hostetler [23]. The mode shift algorithm set $x \leftarrow m(x)$, and repeats the evaluation until $m(x)$ converges.

An implementation from the scikit-learn library was used for the experiment [24].

3. Results

Following tables display experimental results showing SSIM value received after comparison between benchmark of a class and object extracted under specified conditions.

Table 1. Evaluation of the *K*-means method using SSIM.

class number / lux	1	2	3	4	5	6	7	8	9
100	0.377	0.635	0.824	0.774	0.740	0.688	0.785	0.633	0.528
150	0.441	0.707	0.711	0.660	0.618	0.735	0.875	0.805	0.640
200	0.437	0.590	0.753	0.827	0.889	0.871	0.967	0.966	0.919
250	0.484	0.602	0.703	0.552	0.996	0.857	0.641	0.845	0.670

Table 2. Evaluation of the k-medoids method using SSIM.

class number / lux	1	2	3	4	5	6	7	8	9
100	0.361	0.622	0.824	0.774	0.737	0.536	0.785	0.615	0.600
150	0.802	0.702	0.692	0.657	0.661	0.705	0.743	0.796	0.677
200	0.468	0.605	0.753	0.851	0.879	0.844	0.839	0.874	0.679
250	0.432	0.584	0.723	0.598	0.842	0.507	0.641	0.874	0.692

Table 3. Evaluation of the Fuzzy C-Means method using SSIM.

class number / lux	1	2	3	4	5	6	7	8	9
100	0.361	0.622	0.824	0.774	0.737	0.536	0.785	0.615	0.600
150	0.760	0.709	0.698	0.688	0.742	0.776	0.902	0.805	0.615
200	0.511	0.591	0.807	0.871	0.904	0.844	0.920	0.909	0.703
250	0.613	0.578	0.703	0.574	0.999	0.857	0.658	0.885	0.707

 doi:10.20944/preprints202407.1309.v1

Table 4. Evaluation of the Possibilistic C-Means method using SSIM.

lux \ class number	1	2	3	4	5	6	7	8	9
100	-	-	-	-	-	-	-	-	-
150	-	-	-	-	-	-	-	-	-
200	-	-	-	-	-	-	-	-	-
250	-	-	-	-	-	-	-	-	-

Table 5. Evaluation of the Possibilistic Fuzzy C-Means method using SSIM.

lux \ class number	1	2	3	4	5	6	7	8	9
100	0.361	0.622	0.824	0.774	0.737	0.536	0.785	0.615	0.600
150	0.841	0.699	0.711	0.662	0.661	0.717	0.790	0.785	0.679
200	0.511	0.591	0.807	0.871	0.904	0.844	0.920	0.909	0.703
250	0.613	0.578	0.703	0.574	0.999	0.857	0.658	0.885	0.707

Table 6. Evaluation of the Fuzzy Possibilistic C-Means method using SSIM.

lux \ class number	1	2	3	4	5	6	7	8	9
100	0.956	0.634	0.757	0.683	0.744	0.664	0.844	0.605	0.499
150	0.390	0.687	0.719	0.797	0.669	0.822	0.935	0.815	0.615
200	-	-	-	-	-	-	-	-	-
250	0.493	0.790	0.816	0.999	0.980	0.880	0.973	0.987	0.993

Table 7. Evaluation of the Gustafson-Kessel method using SSIM.

lux \ class number	1	2	3	4	5	6	7	8	9
100	-	-	-	-	-	-	-	-	-
150	0.364	0.706	0.757	0.727	0.759	0.815	0.953	0.798	0.594
200	-	-	-	-	-	-	-	-	-
250	-	-	-	-	-	-	-	-	-

Table 8. Evaluation of the Entropy-based Fuzzy method using SSIM.

lux \ class number	1	2	3	4	5	6	7	8	9
100	0.361	0.622	0.824	0.774	0.737	0.536	0.785	0.615	0.600
150	0.760	0.709	0.698	0.688	0.742	0.776	0.902	0.805	0.615
200	0.511	0.591	0.807	0.871	0.904	0.844	0.920	0.909	0.703
250	0.613	0.578	0.703	0.574	0.999	0.857	0.658	0.885	0.707

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 16 July 2024 doi:10.20944/preprints202407.1309.v1

Table 9. Evaluation of the Riddler-Calvard method using SSIM.

lux \ class number	1	2	3	4	5	6	7	8	9
100	0.358	0.623	0.824	0.774	0.738	0.684	0.785	0.634	0.621
150	0.441	0.707	0.711	0.660	0.618	0.735	0.875	0.805	0.640
200	0.544	0.591	0.821	0.881	0.918	0.844	0.922	0.909	0.898
250	0.484	0.602	0.703	0.552	0.996	0.857	0.641	0.845	0.670

Table 10. Evaluation of the Kohonen Self-Organizing Maps method using SSIM.

lux \ class number	1	2	3	4	5	6	7	8	9
100	-	-	-	-	-	-	-	-	-
150	0.572	0.711	0.647	0.702	0.736	0.787	0.902	0.630	0.536
200	-	-	-	-	-	-	-	-	-
250	-	-	-	-	-	-	-	-	-

Table 11. Evaluation of the MeanShift method using SSIM.

lux \ class number	1	2	3	4	5	6	7	8	9
100	0.956	0.628	0.750	0.679	0.744	0.633	0.828	0.605	0.483
150	0.322	0.686	0.803	0.728	0.759	0.780	0.953	0.780	0.587
200	0.459	0.797	0.967	0.780	0.941	0.948	0.939	0.899	0.749
250	0.916	0.999	0.918	0.734	0.941	0.999	0.953	0.925	0.749

4. Discussion

In this article, the search for maximum symmetry is carried out on the example of the visible spectrum, which provides the largest amount of information that needs to be processed in real-time today. However, the very concept and specific methods can also be applied to ultrasound, X-ray, infrared, or sound spectra without significant changes requiring fewer calculations with processing specific to each spectrum. The search for maximum symmetry in the reproduction/interpretation of real-world objects in digital format is a priority factor in research since the accuracy of symmetry in real-time is a decisive factor today. Examples of global requirements include medical applications, where automatic recognition distinguishes between healthy and various problematic cells or organs, impacting life and death, such as advancements in disability quality of life [34], enhanced surveillance [35], and biometric security systems [36]. Also, the recognition of weather conditions underscore the criticality of system symmetry. For example, rapid hurricane image detection aids in timely preparations or evacuations, directly influencing life-or-death outcomes. Therefore, this study investigates the property of systems that is responsible for the symmetry of perception during dynamic changes in conditions.

Additional methods of image retouching [25] are used to improve the classification features for different tasks or under different lighting conditions. In this work, the authors assumed that the preprocessing should be the same for all methods since the goal is real-time quality. Applying additional refinement methods will degrade the real-time integrated quality assessment. Therefore, the preprocessing applied by the video camera manufacturer is sufficient.

 doi:10.20944/preprints202407.1309.v1

5. Conclusion

Table 12. Averaged results of methods evaluated using SSIM.

lux / method	100	150	200	250
K-Means	0.665	0.688	0.802	0.706
K-Medoids	0.65	0.715	0.755	0.655
FCM	0.65	0.744	0.784	0.73
PCM	-	-	-	-
PFCM	0.65	0.727	0.784	0.73
FPCM	0.71	0.717	-	0.879
GK	-	0.719	-	-
EBF	0.65	0.744	0.784	0.73
RC	0.671	0.688	0.814	0.706
SOM	-	0.691	-	-
MeanShift	0.701	0.711	0.831	0.904

Based on the data obtained as a result of the application of SSIM, it can be concluded that in conditions of illumination equal to 100 lux, the best algorithm is FPCM; 150 lux – FCM/EBF; 200 and 250 lux – MeanShift.

Despite good SSIM results, MeanShift's computational complexity of $O(n^2)$ makes it too slow for real-time application.

The obtained results of PCM, GK, and SOM algorithms are unsuitable for use in the problem of image segmentation by pixel clustering in most cases.

References

1. D. Bazazian and M. Parés, "EDC-Net: Edge Detection Capsule Network for 3D Point Clouds.," *Appl.Sci.*, vol. 11, no. 1833, 2021.
2. R. S. Moura, S. R. R. Sanches, P. H. Bugatti and P. T. M. Saito, "Pedestrian traffic lights and crosswalk identification," *Multimedia Tools and Applications*, vol. 81, no. 16497–16513, 2022.
3. P. G. Sathvik, M. Rohith Kumar, G. Harsha Neeli, I. Y. Narasimha, T. Singh and P. Duraisamy, "RESNET-50, CNN and HNN Medical Image Registration Techniques for Covid-19, Pneumonia and Other Chest Ailments Detection," in *13th International Conference on Computing Communication and Networking Technologies, ICCCNT 2022,*, 2022.
4. L. G. Shapiro and G. C. Stockman, "Computer vision," in *Computer Vision*, New Jersey, Prentice-Hall, 2001, pp. 279-325.
5. V.Hrytsyk, M. Medykovskyy, M. Nazarkevych, "Estimation of Symmetry in the Recognition System with Adaptive Application of Filters," Symmetry, vol. 14, no. 5, p. Article number 903, May 2022.
6. V. Hrytsyk and M. Nazarkevych, "Real-Time Sensing, Reasoning and Adaptation for Computer Vision Systems," in *Lecture Notes in Computational Intelligence and Decision Making*, Springer, 2022, pp. 573-585.
7. Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity"," *IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, Apr. 2004.*, vol. 13, no. 4, pp. 600-612, April 2004.
8. Y. Fu, Y. Kang and G. Chen, "Stochastic resonance based visual perception using spiking neural networks," *Frontier in Computational Neuroscience*, vol. 14, no. Art 24, 2020.
9. Z. Xu, Y. Zhai and Y. Kang, "Mutual information measure of visual perception based on noisy spiking neural networks," *Frontiers in Neuroscience*, no. 17:1155362., 2023.
10. J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," in *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, University of California Press, 1967, pp. 281-297.
11. H. Steinhaus, "Sur la division des corps matériels en parties," *Bulletin L'Académie Polonaise des Science*, no. 4, pp. 801-804, 1957.
12. "K-Means Clustering in OpenCV," [Online]. Available: https://docs.opencv.org/3.4/d1/d5c/tutorial_py_kmeans_opencv.html.
13. L. Kaufman and P. J. Rousseeuw, Partitioning Around Medoids (Program PAM), Hoboken, New Jersey: John Wiley & Sons, Inc., 1990, pp. 68-125.

 doi:10.20944/preprints202407.1309.v1

14. "sklearn_extra.cluster.KMedoids," [Online]. Available: https://scikit-learn-extra.readthedocs.io/en/stable/generated/sklearn_extra.cluster.KMedoids.html.
15. J. C. Dunn, "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters," *Journal of Cybernetics,* vol. 3, no. 3, pp. 32-57, 1 January 1973.
16. J. C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Springer, 1981.
17. "Color quantization," [Online]. Available: https://fuzzy-c-means.readthedocs.io/en/latest/examples/01%20-%20Colour%20quantization/.
18. R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering," *IEEE Transactions on Fuzzy Systems,* vol. 1, no. 2, pp. 98-110, May 1993.
19. Ö. Özdemir and A. Kaya, "Comparison of FCM, PCM, FPCM and PFCM Algorithms in Clustering Methods," *Afyon Kocatepe Üniversitesi Fen ve Mühendislik Bilimleri Dergisi,* vol. 19, no. 1, pp. 92-102, 2019.
20. "skcmeans.algorithms Module," [Online]. Available: https://bm424.github.io/scikit-cmeans/skcmeans.algorithms.html.
21. N. R. Pal, K. Pal, J. M. Keller and J. C. Bezdek, "A Possibilistic Fuzzy c-Means Clustering Algorithm," *IEEE Transactions on Fuzzy Systems,* pp. 517-530, September 2005.
22. H. Timm, C. Borgelt, C. Döring and R. Kruse, "An Extension of Possibilistic Fuzzy Cluster," *Fuzzy Sets and Systems,* vol. 147, no. 1, pp. 3-16, October 2004.
23. "IbraDje/PFCM," [Online]. Available: https://github.com/IbraDje/PFCM/blob/master/PFCM.py.
24. D. E. Gustafson and W. C. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," in *1978 IEEE Conference on Decision and Control including the 17th Symposium on Adaptive Processes*, San Diego, California, 1979, pp. 761-766.
25. "ITE-5th/fuzzy-clustering," [Online]. Available: https://github.com/ITE-5th/fuzzy-clustering/blob/master/algorithms/gk.py.
26. J. Yao, M. Dash, S. T. Tan and H. Liu, "Entropy-based fuzzy clustering and fuzzy modeling," *Fuzzy Sets and Systems,* vol. 113, no. 3, pp. 381-388, 1 August 2000.
27. T. W. Ridler and S. Calvard, "Picture Thresholding Using an Iterative Selection Method," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 8, no. 8, pp. 630-632, August 1978.
28. "Thresholding," [Online]. Available: https://mahotas.readthedocs.io/en/latest/thresholding.html.
29. T. Kohonen, "The self-organizing map," *Proceedings of the IEEE,* vol. 78, no. 9, pp. 1464-1480, September 1990.
30. "sklearn-som v. 1.1.0," [Online]. Available: https://sklearn-som.readthedocs.io/en/latest/.
31. Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 17, no. 8, pp. 790-799, August 1995.
32. K. Fukunaga and L. D. Hostetler, "The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition," *IEEE Transactions on Information Theory,* vol. 21, no. 1, pp. 32-40, January 1975.
33. "sklearn.cluster.MeanShift," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MeanShift.html.
34. V. Hrytsyk, A. Grondzal and A. Bilenkyj, "Augmented reality for people with disabilities," in *CSIT 2015*, 2015.
35. "Software: Running Commentary for Smarter Surveillance?," Research∗eu Results Supplement, 2010.
36. M. Nazarkevych, V. Hrytsyk, M. Kostia, L. Parkhuts and H. Nazarkevych, "Biometric Protection Information System with Extension of Segmentation Methods," in *Cybersecurity Providing in Information and Telecommunication Systems II, CPITS-II-2 2021*, 2021.
37. M. Nazarkevych, O. Riznyk, V. Samotyy and U. Dzekendzyak, "Detection of regularities in the parameters of the Ateb-Gabor method for biometric image filtration," *Eastern-European Journal of Enterprise Technologies,* vol. 1, no. 2-97, pp. 57-65, 2019.