sensors

MDPI

*Article*

# Visual Perceptual Quality Assessment Based on Blind Machine Learning Techniques

**Ghislain Takam Tchendjou * and Emmanuel Simeu ***

Univ. Grenoble Alpes, CNRS, Grenoble INP (Institute of Engineering Univ. Grenoble Alpes), TIMA, 38000 Grenoble, France

* Correspondence: ghislain.takam@univ-grenoble-alpes.fr (G.T.T.); emmanuel.simeu@univ-grenoble-alpes.fr (E.S.)

**Abstract:** This paper presents the construction of a new objective method for estimation of visual perceiving quality. The proposal provides an assessment of image quality without the need for a reference image or a specific distortion assumption. Two main processes have been used to build our models: The first one uses deep learning with a convolutional neural network process, without any preprocessing. The second objective visual quality is computed by pooling several image features extracted from different concepts: the natural scene statistic in the spatial domain, the gradient magnitude, the Laplacian of Gaussian, as well as the spectral and spatial entropies. The features extracted from the image file are used as the input of machine learning techniques to build the models that are used to estimate the visual quality level of any image. For the machine learning training phase, two main processes are proposed: The first proposed process consists of a direct learning using all the selected features in only one training phase, named direct learning blind visual quality assessment $DLBQA$. The second process is an indirect learning and consists of two training phases, named indirect learning blind visual quality assessment $ILBQA$. This second process includes an additional phase of construction of intermediary metrics used for the construction of the prediction model. The produced models are evaluated on many benchmarks image databases as $TID2013$, $LIVE$, and $LIVE$ in the wild image quality challenge. The experimental results demonstrate that the proposed models produce the best visual perception quality prediction, compared to the state-of-the-art models. The proposed models have been implemented on an $FPGA$ platform to demonstrate the feasibility of integrating the proposed solution on an image sensor.

**Keywords:** blind image quality assessment; regression technique; non-distortion-specific; visual perception; convolutional neural network; deep learning; $ILBQA$; $FPGA$ implementation

check for updates

**Citation:** Takam Tchendjou, G.; Simeu, E. Visual Perceptual Quality Assessment Based on Blind Machine Learning Techniques. *Sensors* **2022**, *22*, 175. https://doi.org/10.3390/s22010175

Academic Editors: Sonya A. Coleman, Dermot Kerr and Yunzhou Zhang

Received: 25 November 2021
Accepted: 18 December 2021
Published: 28 December 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

Digital images are increasingly used in several vision application domains of everyday life, such as medical imaging [1,2], object recognition in images [3], autonomous vehicles [4], Internet of Things ($IoT$) [5], computer-aided diagnosis [6], and 3$D$ mapping [7]. In all these applications, the produced images are subject to a wide variety of distortions during acquisition, compression, transmission, storage, and displaying. These distortions lead to a degradation of visual quality [8]. The increasing demand for images in a wide variety of applications involves perpetual improvement of the quality of the used images. As each domain has different thresholds in terms of visual perception needed and fault tolerance, so, equally, does the importance of visual perception quality assessment.

As human beings are the final users and interpreters of image processing, subjective methods based on the human ranking score are the best processes to evaluate image quality. The ranking consists of asking several people to watch images and rate their quality. In practice, subjective methods are generally too expensive, time-consuming, and not usable in real-time applications [8,9]. Thus, many research works have focused on objective image quality assessment $IQA$ methods aiming to develop quantitative measures that automatically predict image quality [2,8,10]. The objective $IQA$ process is illustrated in Figure 1. This process has been introduced in the paper [11].

In digital image processing, objective $IQA$ can be used for several roles, such as dynamic monitoring and adjusting the image quality, benchmark and optimize image processing algorithms, and parameter setting of image processing [8,12]. Many research investigations explore the use of machine learning ($ML$) algorithms in order to build objective $IQA$ models in agreement with human visual perception. Recent methods include, artificial neural network ($ANN$), support vector machine ($SVM$), nonlinear regression ($NLR$), decision tree ($DT$), clustering, and fuzzy logic ($FL$) [13]. After the propulsion of deep learning techniques in 2012 [14], researchers were also interested in the possibility of using these techniques in the image quality assessment. Thus, in 2014, the first studies emerged on the use of convolutional neural networks in $IQA$ [15]. Many other works have followed, and we find in the literature increasingly efficient $IQA$ models [16–18].

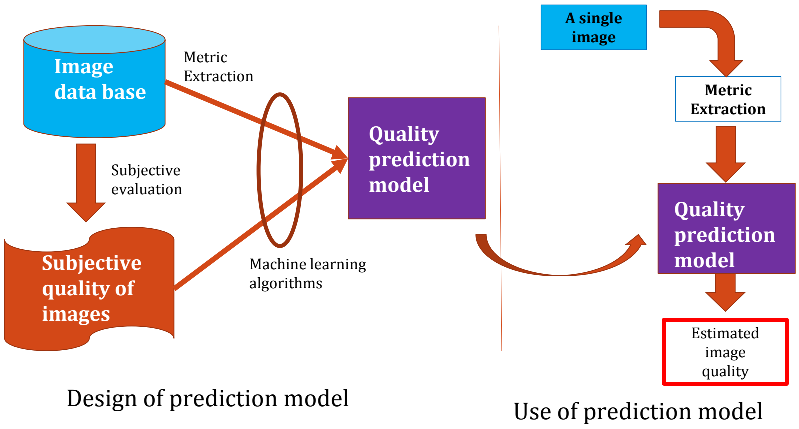


**Figure 1.** Objective image quality assessment process.

The $IQA$ evaluation methods can be classified into three categories according to whether or not they require a reference image: full-reference ($FR$), reduced-reference ($RR$), and no-reference ($NR$) $IQA$ approaches. Full reference image quality assessment ($FR$-$IQA$) needs a complete reference image in order to be computed. Among the most popular $FR$-$IQA$, we can cite the peak signal to noise ratio ($PSNR$), structure similarity index metric ($SSIM$) [8,19], and visual information fidelity ($VIF$) [20]. In reduced reference image quality assessment($RR$-$IQA$), the reference image is only partially available, in the form of a set of extracted features, which help to evaluate the distorted image quality; this is the case of reduced reference entropic differencing ($RRED$) [21]. In many real-life applications, the reference image is unfortunately not available. Therefore, for this application, the need of no-reference image quality assessment ($NR$-$IQA$) methods or blind $IQA$ ($BIQA$), which automatically predict the perceived quality of distorted images, without any knowledge of reference image. Some $NR$-$IQA$ methods assume the type of distortions are previously known [22,23], these objective assessment techniques are called distortion specific ($DS$) $NR$-$IQA$. They can be used to assess the quality of images distorted by some particular distortion types. As example, the algorithm in [23] is for $JPEG$ compressed images, while in [22] it is for $JPEG$2000 compressed images, and in [24] it is for detection of blur distortion. However, in most practical applications, information about the type of distortion is not available. Therefore, it is more relevant to design non-distortion specific ($NDS$) $NR$-$IQA$ methods that examine image without prior knowledge of specific distortions [25]. Many existing metrics are the base units used in $NDS$ methods, such as $DIIVINE$ [26], $IL$-$NIQE$ [27], $BRISQUE$ [28], $GMLOGQA$ [29], $SSEQ$ [30], $FRIQUEE$ [31], DIQA [32], and DIQa-NR [33].

The proposal of this paper is a non-distortion-specific $NR$-$IQA$ approach, where the extracted features are based on a combination of the natural scene statistic in the spatial domain [28], the gradient magnitude [29], the Laplacian of Gaussian [29], as well as the spatial and spectral entropies [30]. These features are trained using machine learning methods to construct the models used to predict the perceived image quality. The process we propose for designing the evaluation of perceived no-reference image quality models is described in Figure 2. The process consists of

- extracting the features from images taken in the $IQA$ databases,
- removing the superfluous features according to the correlation between the extracted features,
- grouping the linearly independent features to construct some intermediate metrics, and
- using the produced metrics to construct the estimator model for perceived image quality assessment.

Finally, we compare the designed models with the state-of-the-art models using the features extraction, but also the process using deep learning with convolutional neural network; as shown in Section 4.1.

To evaluate the performances of the produced models, we measure the correlation between objective and subjective quality scores using three correlation coefficients:

1. Pearson's linear correlation coefficient ($PLCC$), which is used to measure the degree of the relationship between linear related variables.
2. Spearman's rank order correlation coefficient ($SRCC$), which is used to measure the prediction monotony and the degree of association between two variables.
3. Brownian distance ($dCor$), which is a measure of statistical dependence between two random variables or two random vectors of arbitrary, not necessarily equal dimension.

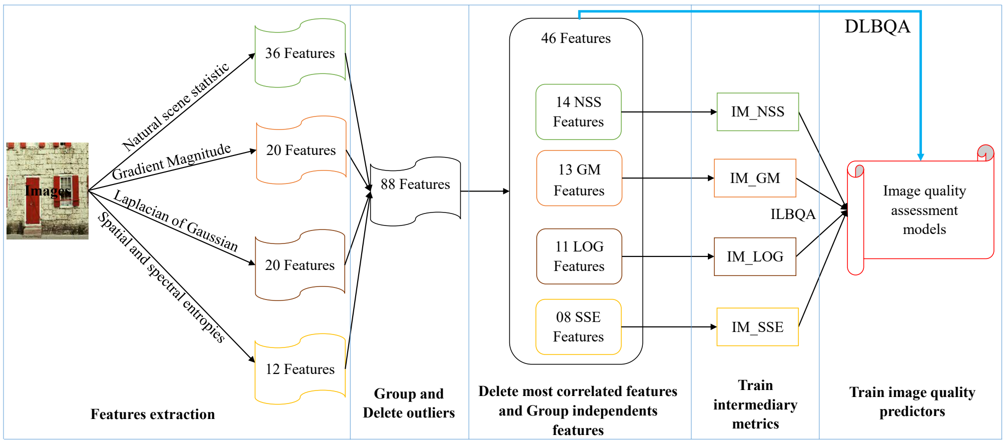


**Figure 2.** Design of prediction models for objective no-reference $IQA$ process.

The paper is organized as follows. Section 2 presents the feature extraction methods. Section 3 explains the feature selection technique based on feature independence analysis. The construction of intermediate metrics is also presented in this section. Section 4 explains the experimental results and their comparison. Section 5 presents the $FPGA$ implementation architectures and results. Finally, Section 6 draws a conclusion and perspectives for future investigations.

## 2. Feature Extraction

Feature extraction in this paper is based on four principal axes: natural scene statistic in the spatial domain, gradient magnitude, Laplacian of Gaussian, and finally spatial and spectral entropies.

### 2.1. Natural Scene Statistic in the Spatial Domain

The extraction of the features based on $NSS$ in spatial domain starts by normalization of the image represented by $I(i,j)$, to remove local mean displacements from zero log-contrast, and to normalize the local variance of the log contrast as observe in [34]. Equation (1) presents normalization of the initial image.

$$\widehat{I}(i,j) = \frac{I(i,j) - \mu(i,j)}{\lambda(i,j) + 1} \tag{1}$$

where $i$ and $j$ are the spatial indices, M and N are the image dimensions, $i \in 1, 2, \ldots, M$ and $j \in 1, 2, \ldots, N$. $\mu(i,j)$ that denotes the local mean is represented by (2) and $\lambda(i,j)$ that estimates the local contract is expressed by (3).

$$\mu(i,j) = \sum_k \sum_l \omega_{k,l} I(i+k, j+l) \tag{2}$$

$$\lambda(i,j) = \sqrt{\sum_k \sum_l \omega_{k,l} [I(i+k, j+l) - \mu(i,j)]^2} \tag{3}$$

where $k = -K, \ldots, K$, and $l = -L, \ldots, L$. $\omega_{k,l}$ is a 2D circularly-symmetric Gaussian weighting function sampled out to 3 standard deviations ($K = L = 3$) and rescaled to unit volume [28].

The model produced in (1) is used as the mean-subtracted contrast normalized (*MSCN*) coefficients. In [28], they take the hypothesis that the *MSCN* coefficients have characteristic statistical properties that are changed by the presence of distortion. Quantifying these changes helps predict the type of distortion affecting an image as well as its perceptual quality. They also found that a Generalized Gaussian Distribution (*GGD*) could be used to effectively capture a broader spectrum of a distorted image, where the *GGD* with zero mean is given by (4).

$$f\left(x; \alpha, \sigma^2\right) = \frac{\alpha}{2\beta\Gamma(1/\alpha)} \exp\left(-(x/\beta)^{\alpha}\right) \tag{4}$$

where $\beta$ is represented by (5) and $\Gamma(.)$ is expressed by (6).

$$\beta = \sigma \sqrt{\frac{\Gamma(1/\alpha)}{\Gamma(3/\alpha)}} \tag{5}$$

$$\Gamma(a) = \int_0^{\infty} t^{a-1} e^{-t} dt \quad a > 0. \tag{6}$$

The parameter $\alpha$ controls the shape of the distribution while $\sigma^2$ controls the variance.

In [28], they also give the statistical relationships between neighboring pixels along four orientations: $H = \widehat{I}(i,j)\widehat{I}(i,j+1); V = \widehat{I}(i,j)\widehat{I}(i+1,j); D1 = \widehat{I}(i,j)\widehat{I}(i+1,j+1)$ and $D2 = \widehat{I}(i,j)\widehat{I}(i+1,j-1)$. This is used with asymmetric density function to produce a practical alternative to adopt a general asymmetric generalized Gaussian distribution (*AGGD*) model [35]. Equation (7) gives the AGGD with zero mode.

$$f\left(x; \gamma, \sigma_l^2, \sigma_r^2\right) = \begin{cases} \frac{\gamma}{(\beta_l + \beta_r)\Gamma(1/\gamma)} \exp\left(-\left(\frac{-x}{\beta_l}\right)^{\gamma}\right) \forall x \leq 0 \\ \frac{\gamma}{(\beta_l + \beta_r)\Gamma(1/\gamma)} \exp\left(-\left(\frac{x}{\beta_r}\right)^{\gamma}\right) \forall x \geq 0 \end{cases} \tag{7}$$

where $\beta_a$ (with $a = l \; or \; r$) is given by (8).

$$\beta_a = \sigma_a \sqrt{\frac{\Gamma(1/\gamma)}{\Gamma(3/\gamma)}} \tag{8}$$

The parameter $\gamma$ controls the shape of the distribution, while $\sigma_l^2$ and $\sigma_r^2$ are scale parameters that control the spread on each side of the mode. The fourth asymmetric parameter is $\eta$ given by (9)

$$\eta = (\beta_r - \beta_l)\frac{\Gamma(2/\gamma)}{\Gamma(1/\gamma)} \tag{9}$$

Finally, the founded parameters are composed of the symmetric parameters ($\alpha$ and $\sigma^2$) and the asymmetric parameters ($\eta, \gamma, \sigma_r^2$, and $\sigma_l^2$), where the asymmetric parameters are computed for the four orientations, as shown in Table 1. All the founded parameters are also computed for two scales, yielding 36 features (2 scales × [2 symmetric parameters + 4 asymmetric parameters × 4 orientations]). More details about the estimation of these parameters are given in [28,36].

**Table 1.** Extracted features based on NSS in spatial domain.

| Feats | Feature Description | Computation Procedure |
|---|---|---|
| **f1–f2** | Shape and variance | Fit GGD to MSCN coefficients |
| **f3–f6** | Shape, mean, left variance, right variance | Fit AGGD to horizontal (H) pairwise |
| **f7–f10** | Shape, mean, left variance, right variance | Fit AGGD to vertical (V) pairwise |
| **f11–f14** | Shape, mean, left variance, right variance | Fit AGGD to first diagonal (D1) pairwise |
| **f15–f18** | Shape, mean, left variance, right variance | Fit AGGD to second diagonal (D2) pairwise |

### *2.2. Gradient Magnitude and Laplacian of Gaussian*

The second feature extraction method is based on the joint statistics of the Gradient Magnitude (*GM*) and the Laplacian of Gaussian (*LoG*) contrast. These two elements *GM* and *LoG* are usually used to get the semantic structure of an image. In [29], they also introduce another usage of these elements as features to predict local image quality.

By taking an image *I*(*i*,*j*), its GM is represented by (10).

$$G_I = \sqrt{[I \otimes h_x]^2 + [I \otimes h_y]^2} \tag{10}$$

where $\otimes$ is the linear convolution operator, and $h_d$ is the Gaussian partial derivative filter applied along the direction $d \in x, y$, represented by (11).

$$h_d(x, y\sigma) = -\frac{1}{2\pi\sigma^2}\frac{d}{\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \tag{11}$$

Moreover, the LoG of this image is represented by (12).

$$L_I = I \otimes h_{LoG} \tag{12}$$

where

$$h_{LoG}(x, y\sigma) = -\frac{1}{2\pi\sigma^2}\frac{x^2+y^2-2\sigma^2}{\sigma^4}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \tag{13}$$

To produce the used features, the first step is to normalize the *GM* and *LoG* features map as in (14).

$$\begin{aligned}\bar{G}_I &= G_I/(N_I + \epsilon)\\ \bar{L}_I &= L_I/(N_I + \epsilon)\end{aligned} \tag{14}$$

where $\epsilon$ is a small positive constant, used to avoid instabilities when $N_I$ is small, and $N_I$ is given by (15).

$$N_I(i,j) = \sqrt{\sum_l \sum_k \omega(l,k) F_I^2(l,k)} \tag{15}$$

where $(l,k) \in \Omega_{i,j}$; and $F_I$ is given by (16).

$$F_I(i,j) = \sqrt{G_I^2(i,j) + L_I^2(i,j)} \tag{16}$$

Then (17) and (18) give the final statistic features.

$$\begin{cases} P_G(G = g_m) = \sum_n K_{m,n} \\ P_L(L = l_n) = \sum_m K_{m,n} \end{cases} \tag{17}$$

$$\begin{cases} Q_G(G = g_m) = \frac{1}{N} \sum_n P(G = g_m L = l_n) \\ Q_L(L = l_n) = \frac{1}{M} \sum_m P(L = l_n G = g_m) \end{cases} \tag{18}$$

where $n = 1, \ldots, N$ and $m = 1, \ldots, M$, and $K_{m,n}$ is the empirical probability function of G and L [37,38]; it can be given by (19).

$$K_{m,n} = P(G = g_m, L = l_n) \tag{19}$$

In [29], the authors also found that the best results are obtained by setting $M = N = 10$; thus, 40 statistical features have been produced as shown in Table 2, 10 dimensions for each statistical features vector $P_G$, $P_L$, $Q_G$ and $Q_L$.

**Table 2.** Extracted features based on GM and LoG.

| Feats | Feature Description |
|---|---|
| **f1–f10** | marginal probability functions of GM |
| **f11–f20** | marginal probability functions of LoG |
| **f21–f30** | measure of the overall dependency of GM |
| **f31–f40** | measure of the overall dependency of LoG |

### *2.3. Spatial and Spectral Entropies*

Spatial entropy is a function of the probability distribution of the local pixel values, while spectral entropy is a function of the probability distribution of the local discrete cosine transform ($DCT$) coefficient values. The process of extracting the spatial and spectral entropies (SSE) features from images in [30] consists of three steps:

- The first step is to decompose the image into 3 scales, using bi-cubic interpolation: low, middle, and high.
- The second step is to partition each scale of the image into 8 × 8 blocks and compute the spatial and spectral entropies within each block. The spatial entropy is given by (20).

$$E_s = -\sum_x p(x) log_2 p(x) \tag{20}$$

and spectral entropy is given by (21).

$$E_f = -\sum_i \sum_j P(i,j) log_2 P(i,j). \tag{21}$$

where $p(x)$ is the probability of x, and $P(i,j)$ is the spectral probability gives by (22).

$$P(i,j) = \frac{C(i,j)^2}{\sum_i \sum_j C(i,j)^2}. \tag{22}$$

where $i = 1, \ldots, 8; j = 1, \ldots, 8$.

- In the third step, evaluate the means and skew of blocks entropy within each scales.

At the end of the three steps, 12 features are extracted from the images as seen in Table 3. These features represent the mean and skew for spectral and spatial entropies, on 3 scales ($2 \times 2 \times 3 = 12$ features).

**Table 3.** Extracted features based on SSE.

| Feats | Feature Description |
|---|---|
| **f1–f2** | Mean and skew of spatial entropy values for the first scale |
| **f3–f4** | Mean and skew of spectral entropy values for the first scale |
| **f5–f6** | Mean and skew of spatial entropy values for the second scale |
| **f7–f8** | Mean and skew of spectral entropy values for the second scale |
| **f9–f10** | Mean and skew of spatial entropy values for the third scale |
| **f11–f12** | Mean and skew of spectral entropy values for the third scale |

### *2.4. Convolutional Neural Network for NR-IQA*

In this paper, we explore the possibility of use the deep learning with convolutional neural network to build the model used to evaluate the quality of the image. In this process, the extraction of features are done by the convolution matrix, constructed using the training process with deep learning.

In CNNs, three main characteristics of convolutional layers can distinguish them from fully connected linear layers in the vision field. In the convolutional layer,

- each neuron receives an image as inputs and produces an image as its output (instead of a scalar);
- each synapse learns a small array of weights, which is the size of the convolutional window; and
- each pixel in the output image is created by the sum of the convolutions between all synapse weights and the corresponding images.

The convolutional layer takes as input and image of dimension $R_l x C_l$ with $S_l$ channels, and the output value of the pixel $p_{l,n,r,c}$ (the pixel of the row $r$ and the column $c$, of the neuron $n$ in the layer $l$) is computed by (23).

$$p_{l,n,r,c} = f_{act}(\sum_{s=0}^{S_l} \sum_{i=0}^{I_l} \sum_{j=0}^{J_l} w_{l,n,s,i,j} p_{l-1,n,r+i,c+j}) \tag{23}$$

where $I_l \times J_l$ are the convolution kernel dimensions of the layer $l$, $w_{l,n,s,j,k}$ is the weight of the row $i$ and the column $j$ in the convolution matrix of the synapse $s$, connected to the input of the neuron $n$, in the layer $l$. In reality, a convolution is simply an element-wise multiplication of two matrices followed by a sum. Therefore, the 2D convolution take two matrices (which both have the same dimensions), multiply them, element-by-element, and sum the elements together. To close the convolution process in a convolutional layer, the results are then passed through an activation function.

In convolution process, a limitation of the output of the feature map is that they record the precise position of features in the input in the convolutional layers. This means that small movements in the position of the feature in the input image will result in a different feature map. This can happen with cropping, rotating, shifting, and other minor changes to the input image [15]. In CNNs, the common approach to solving this problem is the down-sampling using the pooling layers [39]. The down-sampling is a reduction in the resolution of an input signal while preserving important structural elements, without the fine details that are not very useful for the task. The pooling layers are used to reduce the dimensions of the feature maps. Thus, it reduces the number of parameters to learn and the amount of computation performed in the network. The pooling layer summarizes the features present in a region of the feature map generated by a convolution layer. Therefore, further operations are performed on summarized features instead of precisely positioned features generated by the convolution layer. This makes the model more robust to variations in the position of the features in the input image.

In CNNs, the most used pooling function is Max Pooling, which calculates the maximum value for each patch on the feature map. But other pooling function exist, like Average Pooling, which calculates the average value for each patch on the feature map.

Our final CNN model, called $CNN\text{-}BQA$ has 10 layers: four convolutional layers with 128, 256, 128, and 64 channels, respectively; four max pooling layers; and two fully connected layers with 1024 and 512 neurons, respectively. Finally, a output layer with one neuron is computed to give the final score of the image.

## 3. Construction of Prediction Models

In the feature extraction phase, 88 features were extracted in the images. These features are used to construct the prediction models used to evaluate the quality of images. In our case, two processes have been proposed to evaluate the image quality:

- The Direct Learning Blind visual Quality Assessment method ($DLBQA$), in which machine learning methods are directly applied overall set of selected features for training, and producing the final models.
- Indirect Learning Blind visual Quality Assessment method ($ILBQA$), in which ML methods are applied to the four produced intermediary metrics to construct the final model. The $ILBQA$ process requires two training phases: the first is used to merge the independent features of each class in order to generate the adequate intermediary metrics, while the task of the second training is to derive the final $IQA$ model. The inputs of the produced model are the four produced intermediary metrics, and the output is the estimated quality score of the input image.

### *3.1. Remove Superfluous Features*

Having extracted the 88 features on the images, the next step consists of removing the superfluous features, which not add any useful information for the construction of final $IQA$ models. The main type of superfluous features consists of strongly correlated features.

The strongly correlated features give approximately the same information, and this redundancy may cause overfitting problem in the estimation process of the model parameters, and can also produce uselessly complex final models. The proposed way in this paper in order to avoid these problems is to remove the strongly correlated features. The suppression of features is based on the following idea: "if there are two strongly correlated features in the set of extracted features, with a mutual correlation greater than a predefined threshold, we remove the one that has the least correlation with the subjective score without significant loss of information". This reduction process is clearly described in Figure 3, which takes as input the set of extracted features, and produces the set of superfluous features.

The features contained in the output vector of this algorithm are removed from the list of those to be used in the rest of the work. In our use case, we took 0.9 ($thres = 90\%$) as the threshold of mutual correlation between two features. This implies that for any mutual correlation between 2 features higher than 0.9, one of the two features concerned must be removed. With this threshold, the output vector contains 42 features to remove from the initial extracted features list. At the end of this step, 42 features have been found as superfluous features; yielding 46 features are saved for study.

### *3.2. Direct Process (DLBQA)*

In the direct process, the whole set of 46 selected features is used by only one training phase to produce the final quality assessment model. The produced models have been called Direct Learning Blind visual Quality Assessment index based on Machine Learning ($DLBQA\text{-}ML$). Different machine learning methods have been used to construct assessment models. These include methods based on artificial neural networks ($DLBQA\text{-}ANN$), polynomial Nonlinear Regression ($DLBQA\text{-}NLR$), Support Vector Machine ($DLBQA\text{-}SVM$), Decision Tree ($DLBQA\text{-}DT$), and Fuzzy Logic ($DLBQA\text{-}FL$). This different machine learning methods have been presented in [11,40]

Table 4 resumes the results using different correlation scores on the test data by the $DLBQA\text{-}ML$ models produced using different machine learning methods, on the TID2013 [41,42], LIVE [8,43,44], and LIVE challenge [45,46] image databases.

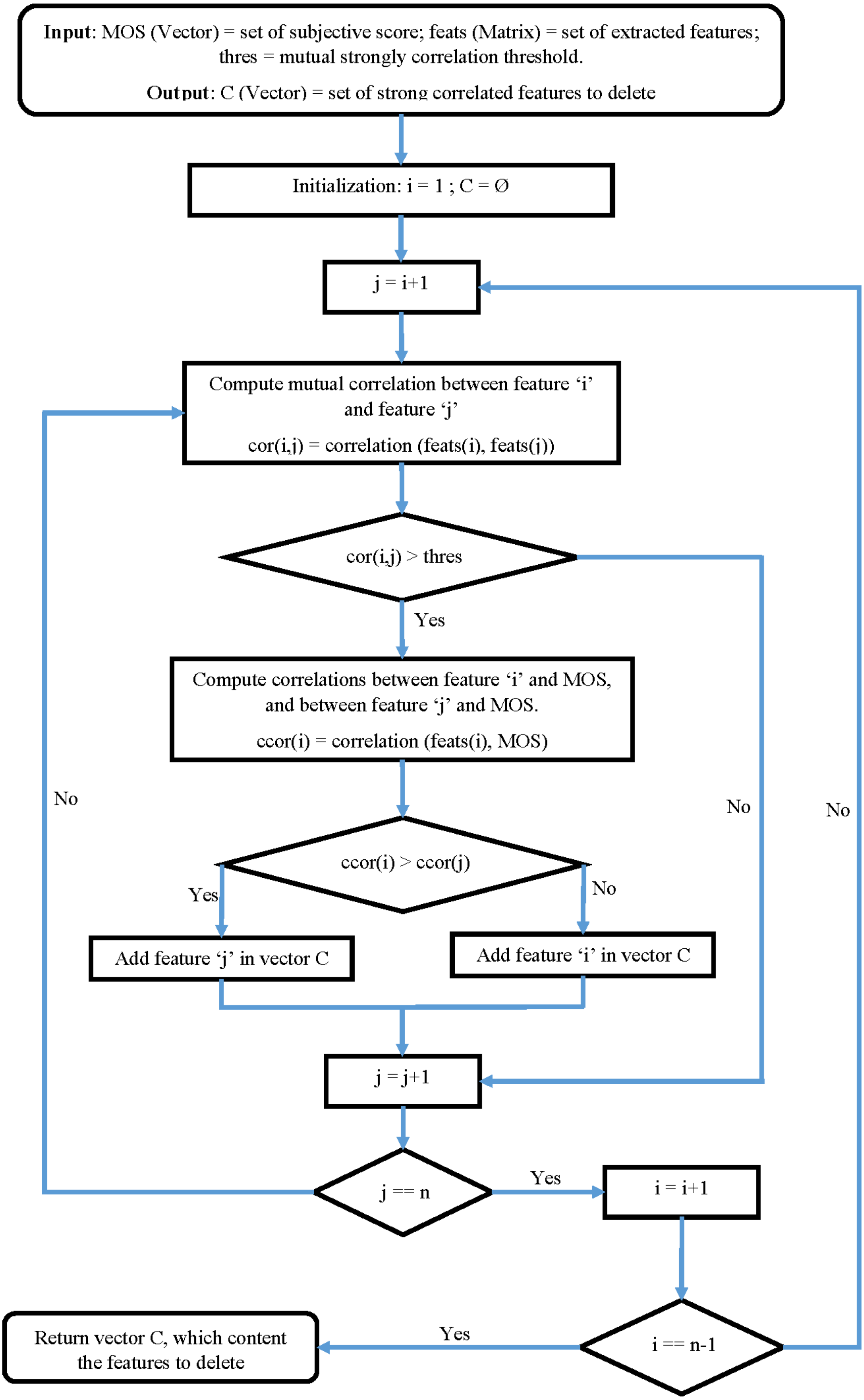

**Input**: MOS (Vector) = set of subjective score; feats (Matrix) = set of extracted features; thres = mutual strongly correlation threshold.
**Output**: C (Vector) = set of strong correlated features to delete
Initialization: i = 1 ; C = Ø
j = i+1
Compute mutual correlation between feature 'i' and feature 'j'
cor(i,j) = correlation (feats(i), feats(j))
cor(i,j) > thres
Yes
Compute correlations between feature 'i' and MOS, and between feature 'j' and MOS.
ccor(i) = correlation (feats(i), MOS)
No
No
No
ccor(i) > ccor(j)
Yes
No
Add feature 'j' in vector C
Add feature 'i' in vector C
j = j+1
j == n
Yes
i = i+1
Return vector C, which content the features to delete
Yes
i == n-1

**Figure 3.** Flow diagram for selection of strongly correlated features to remove.

**Table 4.** Correlation score of MOS vs. estimated image quality of different image databases using different ML methods for DLBQA.

| DB | | ANN | SVM | DT | FL |
|---|---|---|---|---|---|
| **TID** | **PLCC** | 0.871 | **0.899** | 0.878 | 0.786 |
| | **SRCC** | 0.859 | **0.883** | 0.862 | 0.751 |
| | **dCor** | 0.850 | **0.882** | 0.854 | 0.762 |
| **LIVE** | **PLCC** | 0.947 | 0.960 | **0.962** | 0.927 |
| | **SRCC** | 0.946 | 0.958 | **0.960** | 0.936 |
| | **dCor** | 0.939 | 0.950 | **0.954** | 0.927 |
| **NLC** | **PLCC** | 0.627 | **0.801** | 0.741 | 0.673 |
| | **SRCC** | 0.566 | **0.774** | 0.739 | 0.613 |
| | **dCor** | 0.596 | **0.779** | 0.732 | 0.630 |

These results show that the $DLBQA$ based on support vector machine ($SVM$) and decision tree ($DT$) methods provide the best models for predicting image quality from the different image quality databases.

### 3.3. Indirect Process ($ILBQA$)

As the number of selected features is too big (46), the indirect process devises the training phase into three steps:

- The first step, as mentioned in the previous section, is to distribute the features into independent classes, depending on the axes of the extracted features.
- The next step uses machine learning methods to merge the features of each class in order to generate an appropriate intermediary metrics. Then, the pool process has been carried out using different ML methods to create intermediary metrics $IM\_NSS$, $IM\_GM$, $IM\_LOG$, and $IM\_SSE$.
- The third step consists of using the produced intermediary metrics to construct final model and to derive the quality score estimators.

#### 3.3.1. Construction of Intermediate Models

Having removed superfluous features, the next step consists of grouping the selected features into a set of classes. The produced classes are based on four principal axes:

- Natural Scene Statistic in the spatial domain ($NSS$) with 14 extracted features;
- Gradient Magnitude ($GM$) with 13 extracted features;
- Laplacian of Gaussian ($LOG$) with 11 extracted features;
- Spatial and Spectral Entropies ($SSE$) with 8 extracted features.

Each produced class is used to construct one intermediary metric. In each produced class, the features have been merged to generate the appropriate intermediary metrics. The pool process has been carried out, using different machine learning methods.

Table 5 compares the result of applying four ML methods (fuzzy logic, support vector machine, decision tree, and artificial neural network) in order to build the intermediary metrics, using the feature classes. The comparison is represented by Spearman's correlation score between the produced intermediary metrics and the subjective scores give in different image databases.

The selection of the adequate ML method is determined by studying Brownian correlations between the produced intermediary metrics and the subjective scores in multivariate space as shown in Table 6.

The main remark extracts on this Table 6 is that the decision tree ($DT$) ML method produces the best results on all the studied image databases. Thus, the decision tree method has been used as the machine learning method to learn the intermediary metrics.

**Table 5.** Spearman's correlation between intermediary metrics and MOS for different image databases.

| DB | IM_ | ANN | SVM | DT | FL |
|---|---|---|---|---|---|
| TID | **NSS** | 0.647 | 0.703 | **0.842** | 0.615 |
| | **GM** | 0.697 | 0.597 | **0.849** | 0.629 |
| | **LOG** | 0.547 | 0.463 | **0.780** | 0.498 |
| | **SSE** | 0.623 | **0.865** | 0.856 | 0.431 |
| LIVE | **NSS** | 0.931 | **0.939** | 0.930 | 0.915 |
| | **GM** | 0.946 | 0.924 | **0.954** | 0.940 |
| | **LOG** | 0.942 | 0.928 | **0.954** | 0.937 |
| | **SSE** | 0.868 | **0.954** | 0.937 | 0.632 |
| NLC | **NSS** | 0.603 | 0.618 | **0.717** | 0.569 |
| | **GM** | 0.441 | 0.488 | **0.685** | 0.499 |
| | **LOG** | 0.527 | 0.529 | **0.741** | 0.552 |
| | **SSE** | 0.444 | **0.763** | 0.683 | 0.440 |

**Table 6.** Brownian's distance between the group of intermediary metrics and MOS.

| dCor | TID2013 | LIVE | Live Challenge |
|---|---|---|---|
| **ANN** | 0.745 | 0.951 | 0.632 |
| **SVM** | 0.835 | 0.957 | 0.772 |
| **DT** | **0.907** | **0.969** | **0.823** |
| **FL** | 0.691 | 0.937 | 0.634 |

### 3.3.2. Construction of Final Indirect Prediction Model

Once the intermediary metrics are built, the next step in the indirect process is the construction of final prediction model using these intermediary metrics. The final evaluation index constructed consists of several models: four models of construction of the intermediary metrics, and a model of evaluation of the final score using the intermediary metrics. As the intermediary metrics have been constructed in the previous section, all that remains is the construction of the final prediction model, taking as inputs these intermediary metrics previously evaluated. These models are built using different machine learning methods.

Table 7 resumes the prediction performance of $ILBQA$-based image quality models trained on TID213, LIVE, and Live challenge image databases. The design of $IQA$ models consists here of two training steps. The first step uses the decision tree approach in order to merge the quality features into four intermediary metrics. Moreover, the next training step is used different ML approaches ($DT$, $FL$, $SVM$, and $ANN$) for predicting quality score from the corresponding set of intermediary metrics.

These results show that the $ILBQA$ based on decision tree ($DT$) methods provide the best models for predicting image quality from the different image quality databases.

**Table 7.** Correlation score of MOS vs. estimated image quality of different image databases using different ML methods for ILBQA.

| DB | | ANN | SVM | DT | FL |
|---|---|---|---|---|---|
| TID | **PLCC** | 0.925 | 0.936 | **0.954** | 0.924 |
| | **SRCC** | 0.912 | 0.924 | **0.946** | 0.913 |
| | **dCor** | 0.908 | 0.919 | **0.942** | 0.908 |
| LIVE | **PLCC** | 0.971 | 0.978 | **0.982** | 0.970 |
| | **SRCC** | 0.970 | 0.977 | **0.979** | 0.968 |
| | **dCor** | 0.961 | 0.971 | **0.976** | 0.961 |
| NLC | **PLCC** | 0.818 | 0.815 | **0.894** | 0.826 |
| | **SRCC** | 0.777 | 0.848 | **0.877** | 0.793 |
| | **dCor** | 0.792 | 0.851 | **0.879** | 0.802 |

## 4. Experimental Results

Having extracted and removed the superfluous features, the next step is to construct machine learning models, which automatically predict the perceived quality of an input image without a need to reference image. The construction process of the objective NR-IQA block essentially consists of training (with $\alpha$% of data) and validation phases. During the training phase, machine learning algorithms are applied to create an appropriate mapping model that relates the features extracted from an image to subjective score provided in the image database. While the validation phase is used to estimate how well the trained ML models predict the perceived image quality, and to compare the accuracy and robustness of the different ML methods.

The Monte Carlo Cross-Validation ($MCCV$) method [47] is also applied to compare and confirm the accuracy of the obtained results, as follows:

- First, split the dataset with a fixed fraction $(\alpha\%)$ into training and validation set, where $\alpha$% of the samples are randomly assigned to the training process in order to design a model, and the remaining samples are used for assessing the predictive accuracy of the model.
- Then, the first step is repeated k times.
- Finally, the correlation scores are averaged over k.

In this paper, $\alpha\% = 70\%$ and $k = 1000$.

### 4.1. Results Comparison

Having built our evaluation models, the next step is the validation of these models, and the comparison of the results produced with those produced by the state-of-the-art models. The validation process is done using Monte Carlo Cross-Validation ($MCCV$) [47].

In Figures 4–6, different state-of-the-art no-reference $IQA$ methods have been studied and compared with the two models design in this paper ($DLBQA$ and $ILBQA$). These blind $IQA$ methods consist of BRISQUE [28] that uses NSS in the spatial domain, GMLOGQA [29] that uses the gradient magnitude (GM) and the Laplacian of Gaussian (LOG) to make a predictor for image quality, and SSEQ [30] based on the spatial and spectral entropies. These figures depict the box-plot of Brownian's distance in the $MCCV$ phase, of different no-reference $IQA$ methods, trained on $TID$2013, $LIVE$, and $LIVE$ Challenge image databases, respectively. They also show that the $BIQA$ model that obtains the best prediction accuracy (with the highest mean and the smallest standard deviation) is designed by the proposed method $ILBQA$ with two training steps.

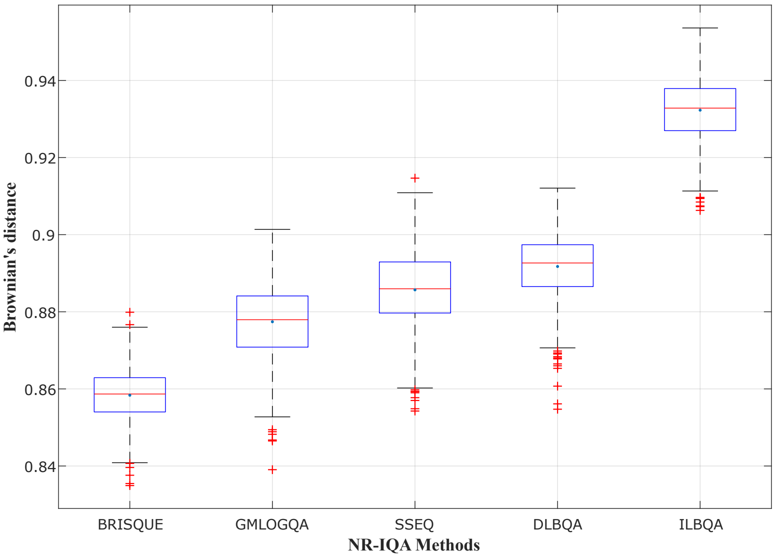


**Figure 4.** Performance comparison for different NR-IQA methods trained on TID2013.

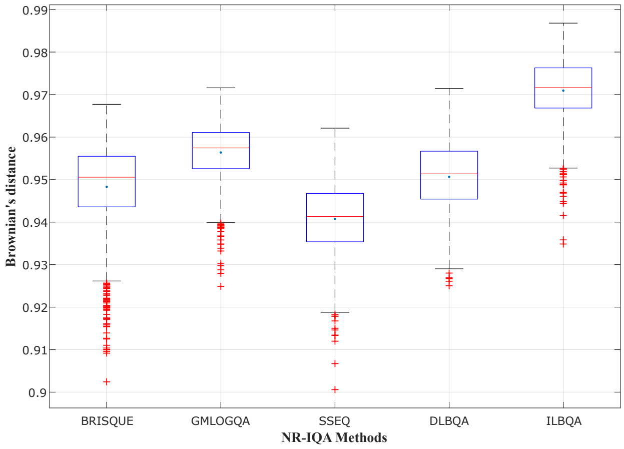


**Figure 5.** Performance comparison for different NR-IQA methods trained on LIVE.

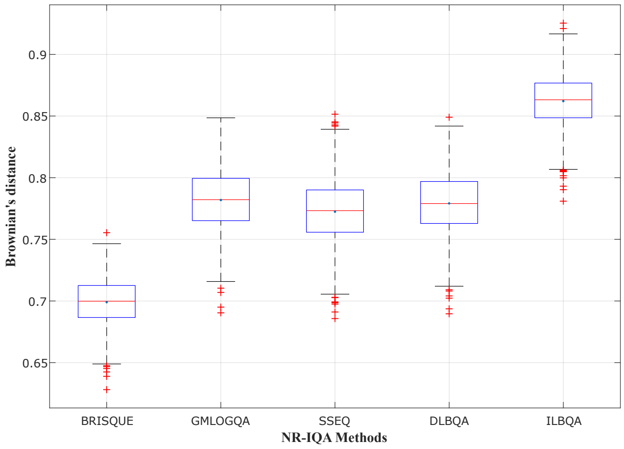


**Figure 6.** Performance comparison for different NR-IQA methods trained on LIVE Challenge.

We compare the proposed models ($DLBQA$, $ILBQA$, and $CNN$-$BQA$) with six feature extraction based BIQA methods, and five CNN-based methods, on three popular benchmark $IQA$ databases. The results are listed in Table 8. In the table, the best two SROCC and PLCC are highlighted in bold. The performances are the correlations between the subjective score (MOS/DMOS) take in image databases, and the evaluated objective scores given by the different studied $NR$-$IQA$ methods in cross-validation phase with $k = 1000$. The main remark is that for every image databases ($TID$2013, $LIVE$, and $LIVE$ Challenge), the proposed $DLBQA$ and $ILBQA$ methods perform the best results.

**Table 8.** Performance comparison on three benchmark $IQA$ databases.

| Methods | LIVE | | TID2013 | | LIVE-CH | |
|---|---|---|---|---|---|---|
| | SRCC | PLCC | SRCC | PLCC | SRCC | PLCC |
| DIQaM [33] | 0.960 | 0.972 | 0.835 | 0.855 | 0.606 | 0.601 |
| BIECON [48] | 0.958 | 0.960 | 0.717 | 0.762 | 0.595 | 0.613 |
| DIQA [32] | **0.975** | **0.977** | 0.825 | 0.850 | 0.703 | 0.704 |
| DB-CNN [49] | 0.968 | 0.971 | 0.816 | 0.865 | **0.851** | **0.869** |
| AIGQA [16] | 0.960 | 0.957 | **0.871** | **0.893** | 0.751 | 0.761 |
| **CNN-BQA(proposed)** | 0.961 | 0.966 | 0.881 | 0.876 | 0.772 | 0.786 |
| DIIVINE [26] | 0.925 | 0.923 | 0.654 | 0.549 | 0.546 | 0.568 |
| BRISQUE [28] | 0.939 | 0.942 | 0.573 | 0.651 | 0.607 | 0.585 |
| ILNIQE [27] | 0.902 | 0.865 | 0.519 | 0.640 | 0.430 | 0.510 |
| FRIQUE [31] | 0.948 | 0.962 | 0.669 | 0.704 | 0.720 | 0.720 |
| SSEQ [30] | 0.941 | 0.943 | 0.883 | 0.886 | 0.772 | 0.807 |
| GMLOGQA [29] | 0.956 | 0.970 | 0.883 | 0.877 | 0.782 | 0.805 |
| **DLBQA(proposed)** | 0.951 | 0.957 | 0.892 | 0.893 | 0.779 | 0.862 |
| **ILBQA (proposed)** | **0.977** | **0.983** | **0.934** | **0.942** | **0.876** | **0.883** |

### 4.2. Computational Complexity

In many image quality assessment applications, the computational complexity is an important factor when evaluating the produced BIQA models. In this paper, the feature extraction is based on four principal axes. Table 9 presents the percentage of time spent by each feature extraction axes.

The run-time of the produced model is compared to the run-times of five other good NR-IQA indexes: BLIINDS-II [50], DIIVINE [26], BRISQUE [28,36], GMLOGQA [29], and SSEQ [30]. Table 10 presents the estimated run-time for an execution on CPU, in second(s), of the different methods, for a $BMP$ image with resolution 512 × 768, on MATLAB, using a computer with 16 $GB$ RAM and Intel Xeon dual-core CPU, 1.70 GHz for each core. We observe that the run-time of the proposed method ($ILBQA$) is significantly inferior to $DIIVINE$ and $BLIIND\text{-}II$ run-time, approximately equal to $SSEQ$ run-time, and superior to $BRISQUE$ and $GMLOGQA$ run-time.

**Table 9.** Percentage of time consumed by each feature extraction axes.

| Axes | Percentage of Time (%) |
|---|---|
| **NSS** | 7.98 |
| **GM** | 3.10 |
| **LOG** | 3.10 |
| **SSE** | 85.77 |

**Table 10.** Comparison of time complexity for different $BIQA$ approaches.

| NR-IQA | Time (second) |
|---|---|
| BLIIND-II | 76.12 |
| DIIVINE | 25.40 |
| BRISQUE | 0.168 |
| GMLOGQA | 0.131 |
| SSEQ | 1.807 |
| **ILBQA** | 2.126 |

## 5. Implementation Architectures and Results

After the construction of the no-reference image quality evaluation models, through the MATLAB tool, the next step in our process is the implementation of the produced models on an FPGA platform. The produced IP-Core can be integrated into the output of an image sensor to evaluate the quality of the produced images. The implemented index is the indirect process ($ILBQA$). This implementation is done with the Xilinx Vivado and Vivado HLS and is implemented on the Xilinx Virtex 7 ($VC$707) and Zynq 7000 ($XC7Z$020) platforms.

The work in [11] introduces this phase of implementation of our proposed process on the objective no-reference image quality assessment. The general implementation process consists of the following.

- HLS implementation of the block IP, allowing the encapsulation of the designed models, through the Vivado HLS tools, in C/C++ codes.
- Integration of the produced IP-cores in the overall design at RTL, through the Vivado tool, in VHDL/Verilog codes.
- Integration on Xilinx's Virtex 7 ($VC$707) and Zynq 7000 ($XC7Z$020) FPGA boards, and evaluation of the obtained results on several test images.

### 5.1. HLS Implementation of the IP-Cores

The implementation of the $ILBQA$ models in HLS enabled the construction of five IP-cores. This IP-Cores are divided into three groups, leading to the evaluation of the objective score of an image taken at the input of the process. The IP-Cores in each group can run in parallel, but the inputs in each group depend on the outputs of the previous group. The three proposed groups consist of feature extraction, construction of intermediate metrics based on decision tree models, and evaluation of the objective score from the model based on fuzzy logic, taking as input the intermediate metrics. The layered architecture of the produced IP-cores is presented in Figure 7.

- Feature extraction: the entries of this group are, the image whose quality is sought to be evaluated, as an AXI-stream, and the dimensions of this input image (row and column sizes as an integer). It contains three IP-cores: *NSS_Core* which extracts 14 features on the image; *GM-LOG_Core* consisting of extraction of 13 and 11 features on *GM* and *LoG* axes, respectively; and *SSE_Core* which extracts 08 features.
- Production of the intermediary metrics: This group contains only one IP-core, the decision tree (*DT*) IP-core. The *DT_Core* uses the features extracted from the input image and generates the intermediary metrics in floating-point format.
- Production of the estimated score of the image using the intermediary metrics: this group also has only one IP-core, fuzzy logic IP-core (*FL_Core*). This IP-core takes as input the four intermediary metrics produced by the *DT_Core* and produces the estimated image quality score of the input image, in floating-point format.

Table 11 presents the estimation of performances and utilization at the synthesis level mapped to Xilinx Virtex 7 (*VC*707) FPGA platform, for the different produced IP-cores, and the global process. These results are produced by an image with a resolution of 512 × 768, and for a clock cycle equal to 25 ns (a frequency equal to 40 Mhz).

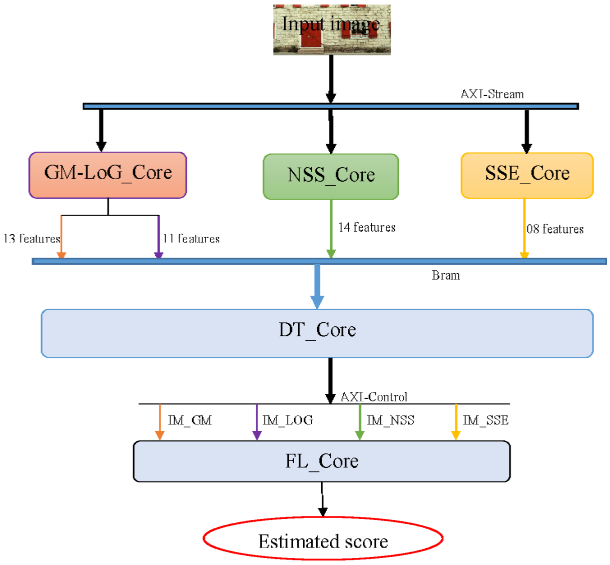


**Figure 7.** Implementation architecture of the *ILBQA* on an *FPGA* platform.

**Table 11.** Device performance utilization estimates in HLS synthesis phase.

| | NSS | GM-LoG | SSE | DT | FL | ILBQA |
|---|---|---|---|---|---|---|
| **clk est. (ns)** | 21.79 | 21.84 | 21.84 | 11.31 | 16.82 | 21.85 |
| **Min clk.** | 79,302 | 1648 | 570 | 2 | 42 | 80,370 |
| **Max clk.** | 9,400,263 | 2,909,751 | 25,231,749 | 17 | 43 | 84,720,454 |
| **BRAM** | 1235 | 1 052 | 554 | 0 | 0 | 2328 |
| **DSP** | 394 | 436 | 141 | 0 | 96 | 1136 |
| **FF** | 59,097 | 69,705 | 20,872 | 10,936 | 5245 | 167,801 |
| **LUT** | 181,656 | 247,775 | 50,903 | 128,122 | 11,779 | 497,622 |

### *5.2. SDK Implementation and Results*

The global design containing our models has been implemented on an *FPGA* board, then a program is implemented in the micro-controller (MicroBlaze) to evaluate the quality of the image thus received through the model implemented on *FPGA*. This program also return the objective score evaluated by the *FPGA* implemented models. The images are retrieved in "AXI-Stream" format and transmitted on the *FPGA* board through an extension using the $HDMI$ protocol. The program is written in C/C++ codes, and passed to micro-controller via the $JTAG$ interface of the $FPGA$ board, thanks to Xilinx SDK tool.

120 images distorted by 24 types of distortion, from the $TID$2013 image database [41,42] were evaluated by the produced models on the $FPGA$ platform. The $FPGA$ implementation results were compared to the experimental results under $MATLAB$ and also to the subjective scores taken the image database.

Figure 8 presents the comparison between $MATLAB$ results in the X-axis and $FPGA$ implementation results in the Y-axis, for the $ILBQA$ index. While Figure 9 presents the comparison between the subjective scores in the X-axis, and $MATLAB$ results (in blue) and $FPGA$ implementation results (in red) in the Y-axis. These figures also present the correlation scores between the scores in the X-axis and the produced objective scores in the Y-axis on each graph.

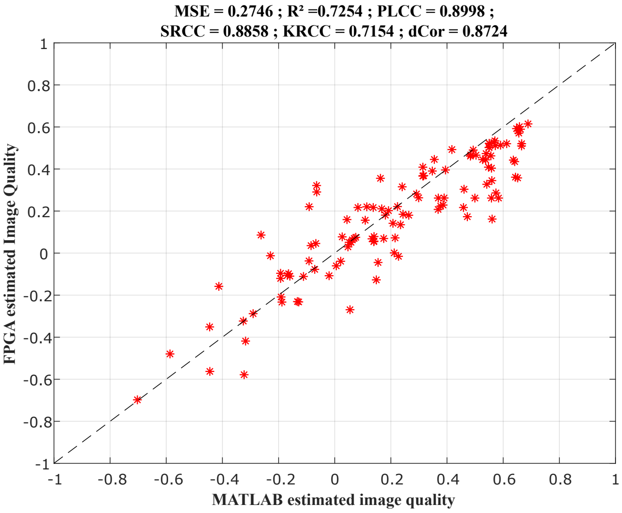


**Figure 8.** Comparison between $MATLAB$ and $FPGA$ implemented scores.

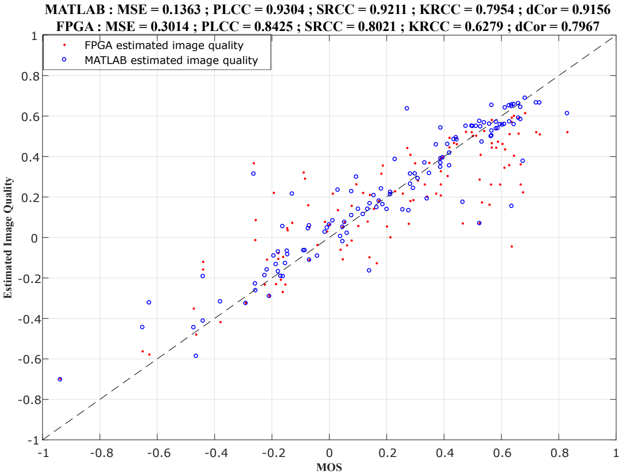


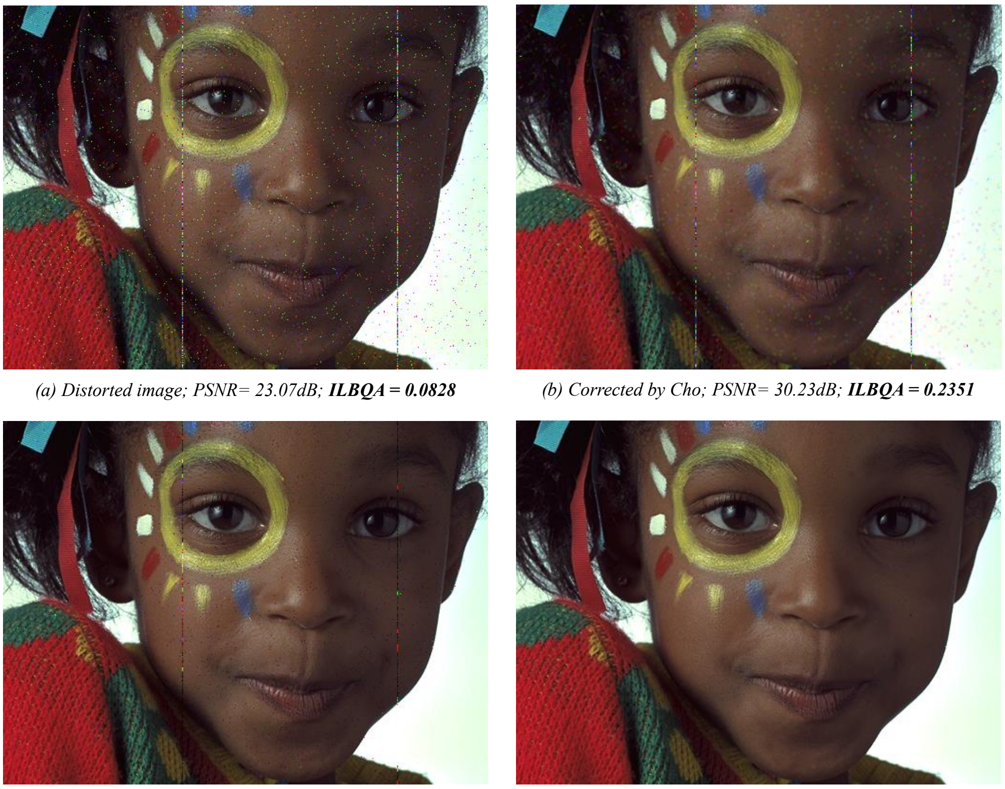
**Figure 9.** Comparison between MOS, $MATLAB$, and $FPGA$ implemented scores.

The difference between the image quality scores estimated under $MATLAB$ and those estimated on $FPGA$ is caused by several factors, including the following.

- Spreading 32-bit floating-point truncation errors in the $FPGA$ system;
- The difference between the results of the basic functions of the MATLAB and Vivado HLS tools. For example, the conversion functions in gray-scale, or the resizing functions using the bi-cubic interpolation, give slightly different results on MATLAB and Vivado HLS. These small errors are propagated throughout the evaluations, to finally cause the differences between the estimated scores shown in the previous figures.

### 5.3. Visual Assessment

Our NR-IQA process has been used in a control loop process [51] to assess the quality of an image and decide if that image needs correction. Gaussian noise is inserted in a reference image, resulting in distorted image illustrated in Figure 10a, with a very low $ILBQA$ score like show in the figure. Figure 10b–d shows the corrected version of the distorted image presented in the graph (a), by the control loop process based on the image quality assessment. Each graph also presents the $PSNR$ between the reference image and the corrected image, as well as the blind image quality ($ILBQA$) score of each image. The graphs (b) and (c) of the Figure 10 show the intermediary corrected images, while the graph (d) presents the best-corrected image by the process.



*(a) Distorted image; PSNR= 23.07dB;* ***ILBQA = 0.0828***

*(b) Corrected by Cho; PSNR= 30.23dB;* ***ILBQA = 0.2351***

*(c) Corrected by CtrlLoop with 2 iterations; PSNR= 32.30dB;* ***ILBQA = 0.5038***

*(d) Corrected by CtrlLoop with 8 iterations; PSNR= 43.61dB;* ***ILBQA = 0.8227***

**Figure 10.** Example of using $ILBQA$ index in a control loop process.

## 6. Conclusions

This paper has presented the construction of models for objective assessment of perceived visual quality. The proposed models use a non-distortion specific no-reference image quality assessment methods. The objective $IQA$ is computed by combining the most significant image features extracted from natural scene statistics ($NSS$) in the spatial domain, gradient magnitude ($GM$), Laplacian of Gaussian ($LoG$), and spectral and spatial entropies ($SSE$). In our proposal, the training phase has been performed using $DLBQA$ and $ILBQA$ process. The $DLBQA$ index evaluates the image quality using all the selected features in one training phase. While the $ILBQA$ index uses two training phases. The first phase consists of training the intermediary metrics using the feature classes based on four extraction axes, and the second training phase evaluates the image quality using the intermediary metrics. Different machine learning methods were used to assess image quality and their performance was compared. The proposed methods consisting of artificial neural network ($ANN$), nonlinear polynomial regression ($NLR$), decision tree ($DT$), support vector machine ($SVM$), and fuzzy logic ($FL$) are discussed and compared. Both the stability and the robustness of designed models are evaluated using a variant of Monte Carlo cross-validation ($MCCV$) with 1000 randomly chosen validation image subsets. The accuracy of the produced models has been compared to the results produced by other state-of-the-art *NR-IQA* methods. Implementation results on *MATLAB* and on the *FPGA* platform demonstrated the best performances for *ILBQA* proposed in this paper, which uses two training phases in the modeling process. One of the future work based on this article is to present how to use our image quality assessment indexes in the control loop process for self-healing of the image sensor. In addition, the implementation on an ASIC and the implementation of the generated IP-core in image sensors.

**Author Contributions:** Conceptualization, G.T.T. and E.S.; methodology, G.T.T. and E.S.; software, G.T.T.; validation, G.T.T. and E.S.; formal analysis, G.T.T. and E.S.; investigation, G.T.T. and E.S.; resources, G.T.T. and E.S.; data curation, G.T.T. and E.S.; writing—original draft preparation, G.T.T.; writing—review and editing, G.T.T. and E.S.; visualization, G.T.T. and E.S.; supervision, E.S.; project administration, E.S.; funding acquisition, E.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Lu, L.; Wang, X.; Carneiro, G.; Yang, L. *Deep Learning and Convolutional Neural Networks for Medical Imaging and Clinical Informatics*; Springer: Berlin/Heidelberg, Germany, 2019.
2. Leonardi, M.; Napoletano, P.; Schettini, R.; Rozza, A. No Reference, Opinion Unaware Image Quality Assessment by Anomaly Detection. *Sensors* **2021**, *21*, 994. [CrossRef] [PubMed]
3. Geirhos, R.; Janssen, D.H.; Schütt, H.H.; Rauber, J.; Bethge, M.; Wichmann, F.A. Comparing deep neural networks against humans: Object recognition when the signal gets weaker. *arXiv* **2017**, arXiv:1706.06969.
4. Plosz, S.; Varga, P. Security and safety risk analysis of vision guided autonomous vehicles. In Proceedings of the 2018 IEEE Industrial Cyber-Physical Systems (ICPS), Saint Petersburg, Russia, 15–18 May 2018; pp. 193–198.
5. Aydin, I.; Othman, N.A. A new IoT combined face detection of people by using computer vision for security application. In Proceedings of the 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, Turkey, 16–17 September 2017; pp. 1–6.
6. Song, H.; Nguyen, A.D.; Gong, M.; Lee, S. A review of computer vision methods for purpose on computer-aided diagnosis. *J. Int. Soc. Simul. Surg* **2016**, *3*, 2383–5389. [CrossRef]
7. Kanellakis, C.; Nikolakopoulos, G. Survey on computer vision for UAVs: Current developments and trends. *J. Intell. Robot. Syst.* **2017**, *87*, 141–168. [CrossRef]
8. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]
9. Ponomarenko, N.; Ieremeiev, O.; Lukin, V.; Jin, L.; Egiazarian, K.; Astola, J.; Vozel, B.; Chehdi, K.; Carli, M.; Battisti, F.; et al. A new color image database TID2013: Innovations and results. In *International Conference on Advanced Concepts for Intelligent Vision Systems*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 402–413.
10. Tsai, S.Y.; Li, C.H.; Jeng, C.C.; Cheng, C.W. Quality Assessment during Incubation Using Image Processing. *Sensors* **2020**, *20*, 5951. [CrossRef]
11. Tchendjou, G.T.; Simeu, E.; Lebowsky, F. FPGA implementation of machine learning based image quality assessment. In Proceedings of the 2017 29th International Conference on Microelectronics (ICM), Beirut, Lebanon, 10–13 December 2017; pp. 1–4.
12. Wang, Z.; Bovik, A.C.; Lu, L. Why is image quality assessment so difficult? In Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Orlando, FL, USA , 13–17 May 2002; Volume 4; p. IV-3313.
13. Tchendjou, G.T.; Alhakim, R.; Simeu, E. Fuzzy logic modeling for objective image quality assessment. In Proceedings of the 2016 Conference on Design and Architectures for Signal and Image Processing (DASIP), Rennes, France, 12–14 October 2016; pp. 98–105.
14. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
15. Kang, L.; Ye, P.; Li, Y.; Doermann, D. Convolutional neural networks for no-reference image quality assessment. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1733–1740.
16. Ma, J.; Wu, J.; Li, L.; Dong, W.; Xie, X.; Shi, G.; Lin, W. Blind Image Quality Assessment With Active Inference. *IEEE Trans. Image Process.* **2021**, *30*, 3650–3663. [CrossRef]
17. Zhang, W.; Ma, K.; Zhai, G.; Yang, X. Uncertainty-aware blind image quality assessment in the laboratory and wild. *IEEE Trans. Image Process.* **2021**, *30*, 3474–3486. [CrossRef]

18. Ullah, H.; Irfan, M.; Han, K.; Lee, J.W. DLNR-SIQA: Deep Learning-Based No-Reference Stitched Image Quality Assessment. *Sensors* **2020**, *20*, 6457. [CrossRef]
19. Li, C.; Bovik, A.C. Content-partitioned structural similarity index for image quality assessment. *Signal Process. Image Commun.* **2010**, *25*, 517–526. [CrossRef]
20. Sheikh, H.R.; Bovik, A.C. Image information and visual quality. *IEEE Trans. Image Process.* **2006**, *15*, 430–444. [CrossRef] [PubMed]
21. Soundararajan, R.; Bovik, A.C. RRED indices: Reduced reference entropic differencing for image quality assessment. *IEEE Trans. Image Process.* **2012**, *21*, 517–526. [CrossRef]
22. Brandão, T.; Queluz, M.P. No-reference image quality assessment based on DCT domain statistics. *Signal Process.* **2008**, *88*, 822–833. [CrossRef]
23. Meesters, L.; Martens, J.B. A single-ended blockiness measure for JPEG-coded images. *Signal Process.* **2002**, *82*, 369–387. [CrossRef]
24. Caviedes, J.; Gurbuz, S. No-reference sharpness metric based on local edge kurtosis. In Proceedings of the International Conference on Image Processing, Rochester, NY, USA , 22–25 September 2002, Volume 3, p. III.
25. Ye, P.; Doermann, D. No-reference image quality assessment using visual codebooks. *IEEE Trans. Image Process.* **2012**, *21*, 3129–3138. [PubMed]
26. Moorthy, A.K.; Bovik, A.C. Blind image quality assessment: From natural scene statistics to perceptual quality. *IEEE Trans. Image Process.* **2011**, *20*, 3350–3364. [CrossRef] [PubMed]
27. Zhang, L.; Zhang, L.; Bovik, A.C. A feature-enriched completely blind image quality evaluator. *IEEE Trans. Image Process.* **2015**, *24*, 2579–2591. [CrossRef]
28. Mittal, A.; Moorthy, A.K.; Bovik, A.C. No-reference image quality assessment in the spatial domain. *IEEE Trans. Image Process.* **2012**, *21*, 4695–4708. [CrossRef]
29. Xue, W.; Mou, X.; Zhang, L.; Bovik, A.C.; Feng, X. Blind image quality assessment using joint statistics of gradient magnitude and Laplacian features. *IEEE Trans. Image Process.* **2014**, *23*, 4850–4862. [CrossRef]
30. Liu, L.; Liu, B.; Huang, H.; Bovik, A.C. No-reference image quality assessment based on spatial and spectral entropies. *Signal Process. Image Commun.* **2014**, *29*, 856–863. [CrossRef]
31. Ghadiyaram, D.; Bovik, A.C. Perceptual quality prediction on authentically distorted images using a bag of features approach. *J. Vis.* **2017**, *17*, 32–32. [CrossRef] [PubMed]
32. Kim, J.; Nguyen, A.D.; Lee, S. Deep CNN-based blind image quality predictor. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *30*, 11–24. [CrossRef]
33. Bosse, S.; Maniry, D.; Müller, K.R.; Wiegand, T.; Samek, W. Deep neural networks for no-reference and full-reference image quality assessment. *IEEE Trans. Image Process.* **2017**, *27*, 206–219. [CrossRef]
34. Ruderman, D.L. The statistics of natural images. *Netw. Comput. Neural Syst.* **1994**, *5*, 517–548. [CrossRef]
35. Lasmar, N.E.; Stitou, Y.; Berthoumieu, Y. Multiscale skewed heavy tailed model for texture analysis. In Proceedings of the 2009 16th IEEE International Conference on Image Processing (ICIP), Cairo, Egypt, 7–10 November 2009; pp. 2281–2284.
36. Mittal, A.; Moorthy, A.K.; Bovik, A.C. Making image quality assessment robust. In Proceedings of the 2012 Conference Record of the Forty Sixth Asilomar Conference on Signals, Systems and Computers (ASILOMAR), Pacific Grove, CA, USA, 4–7 November 2012; pp. 1718–1722.
37. Wainwright, M.J.; Schwartz, O. 10 Natural Image Statistics and Divisive. In *Probabilistic Models of the Brain: Perception and Neural Function*; The MIT Press: Cambridge, MA, USA, 2002; p. 203.
38. Lyu, S.; Simoncelli, E.P. Nonlinear image representation using divisive normalization. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.
39. Jiang, T.; Hu, X.J.; Yao, X.H.; Tu, L.P.; Huang, J.B.; Ma, X.X.; Cui, J.; Wu, Q.F.; Xu, J.T. Tongue image quality assessment based on a deep convolutional neural network. *BMC Med. Inform. Decis. Mak.* **2021**, *21*, 147. [CrossRef]
40. Tchendjou, G.T.; Simeu, E.; Alhakim, R. Fuzzy logic based objective image quality assessment with FPGA implementation. *J. Syst. Archit.* **2018**, *82*, 24–36. [CrossRef]
41. Ponomarenko, N.; Ieremeiev, O.; Lukin, V.; Egiazarian, K.; Jin, L.; Astola, J.; Vozel, B.; Chehdi, K.; Carli, M.; Battisti, F.; et al. Color image database TID2013: Peculiarities and preliminary results. In Proceedings of the European Workshop on Visual Information Processing (EUVIP), Paris, France, 10–12 June 2013; pp. 106–111.
42. Ponomarenko, N.; Ieremeiev, O.; Lukin, V.; Egiazarian, K.; Jin, L.; Astola, J.; Vozel, B.; Chehdi, K.; Carli, M.; Battisti, F.; et al. TID2013 Database. 2013. Available online: http://www.ponomarenko.info/tid2013.htm (accessed on 20 November 2021).
43. Sheikh, H.R.; Wang, Z.; Huang, H.; Bovik, A.C. LIVE Image Quality Assessment Database Release 2. 2005. Available online: http://live.ece.utexas.edu/research/quality (accessed on 20 November 2021).
44. Sheikh, H.R.; Sabir, M.F.; Bovik, A.C. A statistical evaluation of recent full reference image quality assessment algorithms. *IEEE Trans. Image Process.* **2006**, *15*, 3440–3451. [CrossRef]
45. Ghadiyaram, D.; Bovik, A.C. Massive online crowdsourced study of subjective and objective picture quality. *IEEE Trans. Image Process.* **2016**, *25*, 372–387. [CrossRef]
46. Ghadiyaram, D.; Bovik, A. Live in the Wild Image Quality Challenge Database. 2015. Available online: http://live.ece.utexas.edu/research/ChallengeDB/index.html (accessed on 20 November 2021).
47. Xu, Q.S.; Liang, Y.Z. Monte Carlo cross validation. *Chemom. Intell. Lab. Syst.* **2001**, *56*, 1–11. [CrossRef]
48. Kim, J.; Lee, S. Fully deep blind image quality predictor. *IEEE J. Sel. Top. Signal Process.* **2016**, *11*, 206–220. [CrossRef]

49. Xu, J.; Ye, P.; Li, Q.; Du, H.; Liu, Y.; Doermann, D. Blind image quality assessment based on high order statistics aggregation. *IEEE Trans. Image Process.* **2016**, *25*, 4444–4457. [CrossRef]
50. Saad, M.A.; Bovik, A.C.; Charrier, C. Blind image quality assessment: A natural scene statistics approach in the DCT domain. *IEEE Trans. Image Process.* **2012**, *21*, 3339–3352. [CrossRef]
51. Tchendjou, G.T.; Simeu, E. Self-Healing Image Sensor Using Defective Pixel Correction Loop. In Proceedings of the 2019 International Conference on Control, Automation and Diagnosis (ICCAD), Grenoble, France, 2–4 July 2019; pp. 1–6.