



A comparative analysis of machine learning methods for display characterization

Khleef Almutairi^{a,b,*}, Samuel Morillas^{a,1}, Pedro Latorre-Carmona^c, Makan Dansoko^d, María José Gacto^e

^aInstituto de Matemática Pura y Aplicada, Universitat Politècnica de València, Camino de Vera sn, 46022 Valencia, Spain

^bMathematics Department, Faculty of Science, Albaha University, Al Baha, Saudi Arabia

^cDepartamento de Ingeniería Informática, Universidad de Burgos, Avda. Cantabria s/n, Burgos, 09006, Spain

^dTechnopôle du Madrillet, Avenue Galilée - BP 10024 76801 Saint-Etienne du Rouvray Cedex, France

^eDepartment of Software Engineering, DaSCI, University of Granada, 18071 Granada, Spain

ARTICLE INFO

Communicated by K. Almutairi

Keywords:

Display characterization

Machine learning

Regression models

Fuzzy inference systems

Colorimetric Measurements

ABSTRACT

This paper explores the application of various machine-learning methods for characterizing displays of technologies LCD, OLED, and QLED to achieve accurate color reproduction. These models are formed from input (device-dependent RGB data) and output (device-independent XYZ coordinates) data obtained from three different displays. Training and test datasets are built using RGB data measured directly from the displays and corresponding XYZ coordinates measured with a high-precision colorimeter. A key aspect of this research is the application fuzzy inference systems for building interpretable models. These models offer the advantage of not only achieving good performance in color reproduction, but also providing physical insights into the relationships between the RGB inputs and the resulting XYZ outputs. This interpretability allows for a deeper understanding of the display's behavior. Furthermore, we compare the performance of fuzzy models with other popular machine-learning approaches, including those based on neural networks and decision trees. By evaluating the strengths and weaknesses of each method, we aim to identify the most effective approach for display characterization. The effectiveness of each method is assessed by its ability to accurately reproduce and display colors, as measured by the ΔE_{00} visual error metric. Our findings indicate that the Fuzzy Modeling and Identification (FMID) method is particularly effective, allowing for an optimal balance between high accuracy and interpretability. Its competitive performance across all display types, combined with its valuable interpretability, provides insights for potential future calibration and optimization strategies. The results will shed light on whether machine learning methods offer an advantage over traditional physical models, particularly in scenarios with limited data. Additionally, the study will contribute to the understanding of the interpretability benefits offered by fuzzy inference systems in the context of LCD display characterization.

*Corresponding author: kkalmuta@doctor.upv.es

e-mail: smorillas@mat.upv.es (Samuel Morillas), plcarmona@ubu.es (Pedro Latorre-Carmona), makan.dansoko@groupe-esigelec.org (Makan Dansoko), mjgacto@ugr.es (María José Gacto)

1. Introduction

Display characterization is becoming increasingly important due to the ever-expanding variety of displays appearing in the market. Consumers and manufacturers alike look for new visual experiences and more accurate image reproduction [1, 2]. However, achieving this fidelity relies heavily on a thorough display characterization process [1]. Poor characterization can lead to displays with color casts, inaccurate saturation, or even banding, resulting in a distorted and unpleasant viewing experience [3]. Conversely, effective characterization guarantees the vibrant colors and lifelike visuals that consumers expect from modern displays.

Display characterization refers to the measurement and description of a display's behavior. This process is critical because it establishes the objective/mathematical link between the desired colors, and their actual representation on the screen [4]. Designers meticulously craft images using specific colors expressed in device-independent color spaces, such as XYZ tristimulus values or xyY color coordinates [5]. These values define the "ideal" colors intended for display. Display characterization bridges the gap by translating these device-independent values into the display's native language – its RGB values. This translation ensures that the displayed colors closely resemble the designer's original vision [6].

The main goal in display characterization is the creation of a model that maps a digital-to-analog converter (DAC) input featuring an RGB vector to the display's output, expressed in a device-independent color space (typically XYZ tristimulus values or xyY color coordinates) [7]. These relationships are inherently complex, exhibiting non-linearity and interactions between all RGB inputs and all device-independent color coordinates.

The general procedure for developing a display model involves the following steps:

1. **Measure a color dataset:** This process includes the acquisition of DAC RGB values, and their corresponding XYZ or xyY values for model fitting.
2. **Apply model fitting:** Employ the dataset from **step 1** to fit a model that describes the relationship between RGB inputs and XYZ/xyY outputs.
3. **Measure a color test dataset:** This dataset consists of various RGB and XYZ/xyY coordinates.
4. **Predict RGB values:** Use the fitted model to predict the RGB values that should theoretically generate the desired XYZ/xyY values from the test dataset in **step 3**.
5. **Measure actual XYZ/xyY :** Measure the actual XYZ/xyY values produced by the display using the predicted RGB values from **step 4**.
6. **Calculate visual error:** Compare the desired outputs (measured in **step 3**) with the obtained outputs (measured in **step 5**) to determine the visual error between the predicted and actual colors.

Colorimeters (or more sophisticated devices like spectrophotometers) are typically employed to measure the XYZ output values corresponding to the $DACRGB$ vectors. Traditionally,

these measurements were acquired manually, leading to the development of display characterization models based on physical principles. These models were designed to work effectively with a limited number of color measurements, as the manual acquisition process was time-consuming and laborious. This limitation in data often resulted in a higher dependency of the physical model related to the performance [8, 9].

One example of such physically-based models is the family of Rochester Institute of Technology (RIT) models proposed in [10, 2]. These models rely on a relatively simple linear transformation to relate the input RGB values to the output XYZ tristimulus values. However, to account for some non-linearities, they often incorporate pre-processing steps like power functions or look-up tables [11], which can become cumbersome and less effective for displays with highly non-linear behavior [5]. Another example is the model described in [12, 13]. This model uses measurements in the xyY color space and processes them separately. The luminance component (Y) is typically processed with a power function, while the chromaticity (xy) values are processed using linear interpolation between measured data points. This approach can struggle to capture complex interactions between RGB inputs that affect both luminance and chromaticity.

In response to the limitations of traditional models, which assume a simple, independent relationship between input voltage and tristimulus values, researchers have explored more sophisticated approaches. Authors in [14] proposed a two-stage model, the "S-shape model," to address these limitations, particularly for scenarios with limited training data. One such approach involves using linear, non-linear, and hybrid search algorithms, as proposed in [15]. These search algorithms avoid assumptions about the display's internal model, potentially leading to more efficient characterization of various display technologies. Additionally, several other mathematical models have been developed to overcome these limitations. For instance, the model proposed in [16] can independently assess the colorimetric information of red, green, and blue subpixels, potentially offering advantages for displays with complex subpixel structures.

On the other hand, mathematical models also play a role in display characterization. These models are highly data-dependent. In the past, when measurements needed to be taken manually, these models were not very practical. However, today, we have programmable colorimeters and spectrophotometers able to take tens of thousands of measurements in a short period of time. This fact makes mathematical models more appealing today. But not only this, nowadays there is a huge variety of mathematical models that can be used to approximate any mathematical function or data with a high degree of accuracy and computational efficiency.

A classical example of mathematical model for display characterization is the Pseudo-Inverse model [17], which seeks the best linear transformation to capture the relationship between RGB and XYZ color coordinates. This model considers an over-determined system of equations created from measured data. The optimal solution, minimizing the squared error, is achieved using the Penrose pseudo-inverse method. This method was used because even with few measurements limited computa-

tional resources it could provide an acceptable performance. Nowadays, other options of data-driven models exist and machine learning is playing an important role in building them. Taking this into account, how well can machine learning address the display characterization problem?

Therefore, the main aim of our paper is to investigate whether applying different machine learning approaches (in particular, some of them with interpretability features) can improve display characterization compared to using physical models with limited datasets. The remaining parts of this paper are structured as follows. **Section 2** introduces various machine learning methods that used in this study. **Section 3** describe the fitting models. **Section 4** presents the simulation and experimental results within a comparative framework. Finally, **Section 5** concludes the paper and outlines future research directions.

2. Machine learning models

A series of machine learning methods for regression were considered to be applicable to our problem. Some of them are classical methods, while others are black-box type methods. A few of them have an interesting and intrinsic explainability capability. The models were: Neural Network (NN), Decision Tree (DT), K-Nearest Neighbors (KNN), Support Vector Regression (SVR), Random Forest Regression (RFR), Fuzzy Modeling and Identification Toolbox (FMID), Fuzzy MATLAB Toolbox using Mamdani (FMTB-MAM) and Takagi-Sugeno (FMTB-TAK), Fispro Toolbox, Explainable evolutionary multi-objective algorithm combining new linguistic fuzzy grammar and a novel interpretable linear extension (*LING_eExplainable*), Pseudo-Inverse (PI), Multivariate Polynomial Regression (MVPR), and the RIT physical model.

Simple linear regression models the relationship between a single independent variable and a dependent variable using a linear function. Multivariate polynomial regression somehow alleviate this restriction by incorporating higher-order polynomial terms [18]. Multivariate polynomial regression (MVPR) [19], is an extension of the simple linear regression model [20]. It fits a polynomial function with a series of input variables (or features). It is a powerful tool in mathematical modeling and statistics for complex interactions between numerous independent and dependent variables [21]. The coefficients of the polynomial equation are typically found using optimization techniques. Nevertheless, increasing the polynomial degree can lead to over-fitting [22], where the model performs poorly on unseen data due to capturing noise or random variations in the training data. In multivariate polynomial regression, regularization techniques like ridge regression [23] and lasso regression [24] can help mitigate this drawback.

Neural networks have been used intensively for classification as well as for prediction [25, 26]. For display characterization, [1] proposes a trained model using a neural network (NN) that predicts the *RGB* inputs required to achieve a desired color expressed in either *XYZ* or *xyY* coordinates. However, these methods are often categorized as "black box" models, limiting their interpretability.

This lack of interpretability can be a significant drawback in many domains. To circumvent this limitation, we also pro-

pose in this paper the use of fuzzy inference systems (FIS), introduced by [27, 28]. FIS have been shown to be an effective and universal fitting function. If the relevant implication rules, membership functions, inference operators, and defuzzification method are appropriately chosen, any relationship (mapping) between inputs and outputs may be described using a FIS up to any degree of accuracy. A FIS is often created using professional expertise. In such a case, it is necessary to have a solution for the particular problem being addressed. In many instances, however, ground truth data that connects the inputs and outputs of the intended system exists instead of this expert knowledge [29]. Therefore, it is also interesting to build a system like this, just from datasets. Processing the collected data into a model that can be related to a human way of thinking can be achieved by building a fuzzy system [30] that provides a transparent representation of a non-linear system, giving a linguistic interpretation in the form of implication rules.

The rules in a FIS follow two types of (alternative) modelling approaches: (a) The so-called Mamdani system [30], which is more straightforward and intuitive. This form usually has linguistic terms for both antecedents and consequents, such as: **IF** *X* is *low* **AND** *Y* is *medium*, **AND** *Z* is *high*, **THEN** *G* is *medium*, and (b) The Takagi-Sugeno system, which have linguistic terms in the antecedents but functions in the consequents so that it can be seen as a soft combination of different functions, that are activated to a higher or lesser degree depending on antecedent certainty and which provide the final outputs. So, we aim to build a FIS using different toolboxes from a color dataset in order to obtain interpretable models with a high performance.

The Fuzzy Modeling and Identification Toolbox (FMID) [31] is applied in our analysis. This toolbox allows for the creation of a fuzzy logic rule system that determines the required *RGB* coordinates to generate a desired color expressed in device-independent color coordinates. We considered the FMID in this study [32] and the results indicated suboptimal performance when employing the *xyY* coordinates. Consequently, we opted to use the *XYZ* coordinates to enhance the accuracy and reliability of our model's predictions.

The Fispro toolbox [33] was also considered. It is a widely used and adaptable software tool for FIS. Fispro is a powerful and comprehensive software suite offering a variety of features to simplify FIS creation, analysis, and optimization. Its user-friendly interface makes FIS development accessible even for users without extensive programming experience. A key advantage of Fispro is its ability to automatically construct fuzzy rule bases. Users can simply provide their data, and the toolbox will derive optimal fuzzy rules. Additionally, Fispro offers rule optimization tools that allow users to refine existing fuzzy inference systems to achieve a specific desired performance.

In this context, the MATLAB Fuzzy Logic Toolbox [34, 35] also provides a comprehensive environment for design, implementation, and training. It offers a rich set of features and graphical tools for building and evaluating fuzzy systems. The toolbox facilitates the construction and modification of fuzzy systems through a user-friendly interface. This includes tools for defining fuzzy rule sets, simulating system behavior, and vi-

sualizing membership functions. Additionally, it supports various inference techniques, such as Mamdani and Sugeno, and provides functionalities for evaluating and validating the designed FIS.

A recent study [36] proposes a novel methodology that prioritizes both model accuracy and interpretability. This approach combines two key elements. On the one hand, it uses a New Linguistic Fuzzy Grammar. This grammar allows the model to represent relationships between variables using human-understandable terms like "low," "medium," and "high." This significantly enhances the model interpretability. On the other hand, the approach incorporates a Novel Interpretable Linear Extension. This extension allows the consequent of the rule to explain the specific variability of the rule through a simple linear relationship with only a single input variable.

Apart from models based on the theory of fuzzy sets, there are other, well established and standard (classic) methods, like the k-nearest neighbors algorithm (kNN), described by [37]. This is a clustering method, which groups data points based on their distance, i. e., it identifies groups by considering the k nearest neighbors of each point [38].

This method offers several advantages. First, it is known for its simplicity and ease of use. kNN can handle a wide range of inputs without requiring a training phase or complex model construction. Additionally, kNN is interpretable, particularly through visualizations like graphs, which can be helpful for understanding the groupings.

Decision trees [39] is another group of methods that have shown its capabilities both for classification [40] and regression tasks. As a non-parametric supervised learning algorithm, it effectively handles complex and non-linear relationships.

Decision trees represent a collection of decisions and their potential outcomes as a tree structure. Each internal node denotes a choice or test on a feature, while each connected branch represents one of that decision's potential outcomes. Finally, each leaf node refers to a class label or a predicted value [41]. They are interpretable as well [42, 43]. Their structure closely resembles human decision-making processes, making them easy to visualize and understand. Additionally, decision trees are adaptable to various data types, handling both categorical and numerical data effectively. They can efficiently manage large datasets with high-dimensional features and they are computationally efficient. Furthermore, decision trees are robust to outliers and missing values because they rely on decision rules rather than statistical assumptions. However, overfitting can be an issue, particularly with overly complex trees [44]. Pruning techniques, which simplify the tree by removing unnecessary branches or nodes, can help mitigate this problem.

Despite the limitations of traditional methods, ensemble approaches like Random Forest Regression (RFR) [45] offer promising solutions. They are built upon the strengths of decision trees while addressing inherent weaknesses. Decision trees, used for both regression and classification tasks, represent decisions and their possible outcomes in a tree-like structure [46]. In regression, the process starts at the root node, splitting data based on variable values until reaching a leaf node that provides the prediction. While decision trees are straightforward

and interpretable, they can be prone to bias and overfitting [47]. This means they may capture noise in the training data, leading to poor performance on unseen data.

Ensemble learning (the use of ensemble approaches), combines predictions from several models, and then produces a more accurate and robust output. The Random Forest algorithm employs bootstrapping, where random subsets of the dataset are sampled with replacement over numerous iterations. Each tree in the forest is trained on a different subset of the data, addressing the issue of overfitting and enhancing the model's robustness. Additionally, Random Forests select a random subset of features for each tree, unlike decision trees that consider all possible feature splits. This random feature selection ensures a low correlation among the trees, further improving the ensemble's performance.

Support Vector Regression (SVR) [48] offers a powerful non-linear regression method, which has been intensively used due to its capability to deal with datasets with a low number of points in high dimensions. SVR works in a two-step process. First, it maps the input variables into a higher-dimensional feature space. This allows the model to capture complex, non-linear relationships within the data. Second, it finds a hyperplane within this high-dimensional space that maximizes the margin – the distance between the hyperplane and the closest data points. Crucially, SVR also minimizes prediction error during this process. This focus on maximizing the margin enhances the model's ability to generalize, meaning it performs well on new, unseen data [49].

A significant advantage of SVR is its ability to handle non-linear relationships between the input and output variables. This is achieved using a kernel function [50], which essentially maps the data into a higher-dimensional space where a simple linear hyperplane can effectively model the data. Common kernel functions include the polynomial kernel and the radial basis function (RBF, also called Gaussian) kernel. The selection of the correct kernel function is critical, because it directly impacts the model's ability to capture the underlying patterns in your data. Selecting an appropriate kernel can significantly improve SVR's performance, ensuring it can effectively handle complex, non-linear relationships. SVR offers several other advantages [51]. It can be effective in high-dimensional spaces, is memory efficient due to its reliance on support vectors, and is flexible due to various kernel functions that enhance its adaptability to different data types.

However, there are also challenges associated with SVR. SVR models can be computationally expensive, requiring efficient algorithms and optimization techniques for handling large datasets and ensuring timely model training and prediction [52].

3. Fitting the models

We employed the three displays and colorimeter, as described in 4, to create different datasets for model fitting and validation for each display in the study. These datasets were constructed by considering a mesh of equally spaced points within the *RGB* color space (represented as a cube). In this space, *RGB* values are integers ranging from 0 to 255.

Specifically, for the training dataset, we divided each *DACRGB* input into 22 levels, ranging from 0 to 252 with a step size of 12 units. This resulted in all possible combinations within the *RGB* cube, leading to a total of 10,648 unique samples. The corresponding *XYZ* color values for these samples were obtained using the colorimeter. Therefore, the training dataset consists of 10,648 samples, each containing both *RGB* and *XYZ* data.

For the validation dataset, we defined another mesh with points positioned between those of the training dataset. Here, we employed 21 equally spaced *RGB* levels ranging from 6 to 246 with the same step size of 12 units. As with the training dataset, all possible combinations were considered, resulting in a total of 9,261 samples with corresponding *RGB* – *XYZ* data.

3.1. Fuzzy modelling and identification toolbox (FMID)

We use the Fuzzy Modeling and Identification Toolbox (FMID) [31] to create a fuzzy inference system (FIS) based on the Takagi-Sugeno (TS) type. The FMID toolbox allows for different parameter settings depending on the problem [31]. After feeding the data into the system, FMID uses fuzzy clustering, with a user-defined degree of overlap, to divide the dataset into a specific number of clusters. Since the FIS rules are created using the clustering results, the designer decides in advance how many clusters to obtain and, therefore, how many local sub-models of the fuzzy model there should be.

The Gustafson-Kessel (GK) algorithm [53] (extensively used in clustering) is applied to determine the division and therefore to define the membership functions of the antecedents. Then, the activation functions of the consequents need to be fitted by least squares minimization. There are different options of function types that can be tested, and the optimal option depends heavily on the particular problem and data.

Once all of the FMID parameters have been determined and set to initial parameters, the next step is to test the outcome using a new, independent color dataset. The trained system receives the validation dataset of desired colors expressed in device-independent color coordinates and predicts what *RGB* values need to be used to generate each of them. Here, we can use the mean squared error between computed *RGB* and corresponding *RGB* of the validation dataset ($MS E_{RGB}$) as a performance metric to describe the model's level of accuracy, which is useful for FMID parameter fitting. However, the final visual error should be measured between colors displayed with the computed *RGB* and desired colors in the validation dataset.

We study the three most important parameters, explained in details in the appendix A, that have a significant impact on $MS E_{RGB}$ performance: (1) the type of response functions in the rules' consequents, (2) the number of clusters (rules) in the system, and (3) the overlapping parameter m between the clusters membership functions (fuzziness in the system).

First, we analyze performance depending on the type of activation function in the consequents. We considered linear, parabolic, and square root functions. In the case of *XYZ* inputs, the linear option is the best in terms of $MS E_{RGB}$ regardless of the number of clusters for the three displays under investigation.

Second, we analyze the performance as a function of the number of rules (clusters) that will be used in the model and

lastly, the overlapping degree between the clusters (m value in FMID). For this, we varied the number of rules (Clusters) between 2 – 15 and the m parameter between 1.05 – 2.25.

A decision about the number of rules/clusters is more complex. As we increase the number of clusters, the performance of the system approaches optimal performance, but having so many rules increases the model's complexity. However, performance does not drop significantly when reducing the number of rules slightly, and this would result in a simpler model that is easier to interpret. Table 1 summarizes the hyper-parameters for the three displays under investigation.

3.2. The FisPro open source software

The study used the FisPro toolkit to train FIS. The FIS was configured with three inputs and one output (resulting in three separate systems) to generate *RGB* values, and trapezoidal membership functions were used. Two rule generation methods were employed: Fast Prototyping Algorithm (FBA) and Wang & Mendel (WM) [54]. The system was optimized for 50 iterations. The Mean Absolute Error (MAE) [55] served as the performance evaluation metric.

However, some challenges were encountered during the training process. Since FisPro has limitations on the number of outputs per system, we had to create separate systems for each output channel (*R*, *G*, and *B*) to achieve the desired *RGB* output. While the study successfully trained FIS using FisPro for display characterization, the performance did not translate to accurate *RGB* predictions. Consequently, the model was built to a specific display (*ASUS VA27EHE*) to assess its effectiveness in that particular context.

3.3. Matlab fuzzy logic toolbox (FMTB)

A Fuzzy Matlab toolbox (FMTB) is a popular approach for modelling complex systems where deriving precise mathematical models is difficult. In this experiment, a Mamdani Fuzzy Inference System (FIS) using FMTB is employed to train a model for display characterization. The model takes *XYZ* color space values as inputs and predicts the corresponding *RGB* color space values as outputs.

The experimental procedure involves several steps to ensure an optimized and accurate FIS. First, the minimum and maximum ranges are extracted from the training data to define the input and output variable ranges for the FIS. Then, various configurations of membership functions (MFs) for the inputs are tested, with the number of MFs ranging from 2 to 6. This allows for a comprehensive exploration of the FIS performance under different granularities of the fuzzy sets.

For each configuration, a Mamdani FIS is initialized and tuned using particle swarm optimization (PSO) [56] to optimize the system's rules and parameters. The optimization process is carried out in two stages: learning and tuning. During the learning stage, the FIS learns the rules from the training data. In the tuning stage, a local optimization method called pattern search [57] is used to fine-tune the parameters for better convergence and accuracy. This two-tier optimization approach ensures that the FIS not only captures the underlying patterns in the training data but also generalizes well to unseen validation data.

Table 1: Key Hyperparameters for Fuzzy Inference Models (FMIDs) of Three Display.

| Parameters | ASUS VA27EHE Eye Care LED | MSI G274QPF-QD Monitor | ASUS ROG Swift OLED PG27AQDM |
|---------------------|---------------------------|------------------------|------------------------------|
| Function type | Linear | Linear | Linear |
| Clusters number | 5 | 8 | 8 |
| Overlapping params. | 1.5 | 1.25 | 1.25 |

Table 2: Key Hyperparameters for the Fuzzy Matlab Toolbox with Two Types of FIS (Mamdani and Sugeno) for Three Displays.

| LCD ASUS display | | | | |
|-------------------|---------|-----------------|-------------|-------|
| Method | MSE | Hyperparameters | | |
| | | MFS inputs | MFs outputs | Rules |
| Mamdani | 1101.41 | 3 | 27 | 23 |
| Sugeno | 989.40 | 3 | 3 | 26 |
| QLED MSI display | | | | |
| Method | MSE | Hyperparameters | | |
| | | MFS inputs | MFs outputs | Rules |
| Mamdani | 749.51 | 3 | 27 | 24 |
| Sugeno | 722.01 | 3 | 3 | 27 |
| OLED ASUS display | | | | |
| Method | MSE | Hyperparameters | | |
| | | MFS inputs | MFs outputs | Rules |
| Mamdani | 706.61 | 2 | 8 | 7 |
| Sugeno | 786.79 | 2 | 2 | 4 |

The performance of each FIS configuration is evaluated using Mean Squared Error (MSE) on both the training and validation datasets. This provides a quantitative measure of how well the model has learned to map *XYZ* inputs to *RGB* outputs.

Similar to the Mamdani approach, a Sugeno FIS is also investigated for the same task of display characterization, with *XYZ* values as inputs and *RGB* values as outputs. The Sugeno FIS experimental setup follows a similar procedure as the Mamdani FIS. The input data ranges are extracted, and different configurations of input MFs are tested.

Optimization of the Sugeno FIS is also performed using PSO, followed by the local optimization method. The tuning process involves both learning the rules and fine-tuning the parameters to achieve minimal MSE on the training and validation datasets. The use of PSO helps in efficiently searching the parameter space, while local optimization ensures fine adjustments for better performance.

The hyperparameter settings for both Mamdani and Sugeno FIS types are illustrated in Table 2.

3.4. Random Forest Regression (RFR)

This study investigates the use of Random Forest Regression (RFR) to predict optimal *RGB* values for the displays under investigation. We developed separate models for red, green, and blue color channels. This approach allows us to analyze the performance of each model on a specific color and subsequently combine them for the best overall result.

Hyperparameter tuning is crucial for optimal RFR model performance. A key hyperparameter is the number of trees in the forest (*n*-estimators). While increasing the number of trees generally improves accuracy by averaging individual tree errors, it also increases computational cost. We explored a range of values from 10 to 1,000 for this parameter to find the optimal balance between accuracy and efficiency.

Another important parameter is the maximum depth of each tree (max-depth), which controls its complexity. Deeper trees

can capture intricate data patterns but are more prone to overfitting. To mitigate this risk, we explored a range of values between 10 and 100 for max-depth.

Balancing model complexity and efficiency involves tuning additional hyperparameters: the minimum number of samples required to split an internal node (min-samples-split) and the minimum number required for a leaf node (min-samples-leaf). Lower values for these parameters can improve the model's fit to the training data, but they also increase the risk of overfitting. Conversely, higher values lead to more generalized trees that might underfit the data. We explored the parameter space of min-samples-split (2 to 10) and min-samples-leaf (1 to 10) to find the optimal settings.

Finding the optimal balance for all these hyperparameters is achieved through a gradient descent-based optimization function like custom gradient boosting [58]. This function iteratively adjusts the hyperparameters to minimize the model's mean squared error (MSE), guiding it towards the best configuration.

Custom gradient boosting works by randomly selecting hyperparameter values in each iteration, training the model, and evaluating its performance based on MSE. The difference in MSE between iterations, known as the gradient, indicates the direction and magnitude for improvement. If the gradient falls below a predefined tolerance, the process stops, suggesting we have reached near-optimal hyperparameters. This iterative approach explores a vast space of possible hyperparameter combinations, significantly increasing the likelihood of finding an optimal configuration that minimizes MSE. Efficiently tuning these key hyperparameters allows the RFR model to achieve a robust balance between complexity and computational efficiency, resulting in more accurate and reliable predictions. The summary of the hyperparameter settings of the displays under study is in Table 3.

Table 3: Key Hyperparameters for Random Forest Regressor RFR of Three Displays.

| LCD ASUS display | | | | | |
|-------------------|--------|-----------------|-----------|-------------------|------------------|
| Colour | MSE | Hyperparameters | | | |
| | | n_estimators | max_depth | min_samples_split | min_samples_leaf |
| R | 140,24 | 29 | 66 | 2 | 2 |
| G | 15,97 | 942 | 70 | 5 | 2 |
| B | 28,27 | 49 | 28 | 6 | 1 |
| QLED MSI display | | | | | |
| Colour | MSE | Hyperparameters | | | |
| | | n_estimators | max_depth | min_samples_split | min_samples_leaf |
| R | 27,85 | 99 | 18 | 3 | 1 |
| G | 10,31 | 80 | 60 | 4 | 1 |
| B | 29,56 | 88 | 16 | 7 | 5 |
| OLED ASUS display | | | | | |
| Colour | MSE | Hyperparameters | | | |
| | | n_estimators | max_depth | min_samples_split | min_samples_leaf |
| R | 65,54 | 91 | 91 | 5 | 3 |
| G | 20,77 | 44 | 14 | 4 | 2 |
| B | 31,34 | 89 | 78 | 5 | 2 |

3.5. Support Vector Regression (SVR)

Support Vector Regression (SVR) is employed to identify a function that closely approximates the relationship between input XYZ variables and a continuous target variable RGB , minimizing prediction errors. The key hyperparameters in SVR are the regularization coefficient (C), epsilon (ϵ), and the kernel type. The regularization coefficient (C) manages the trade-off between a large margin and classification error on the training data. Higher values of C aim to classify all training points correctly, which can lead to overfitting. Conversely, lower values allow for a larger margin with more classification errors, potentially improving generalization. Typically, the range for C spans from 1 to 15. The epsilon (ϵ) parameter defines the margin of tolerance in the SVR loss function. A higher ϵ value makes the model more tolerant to errors within the margin, while a lower value forces the model to fit more closely to the training data. The range for ϵ is usually between 0.1 and 1.

The choice of kernel is crucial as it directly impacts the model's ability to capture underlying data patterns. Common kernel functions include the linear kernel, polynomial kernel, and radial basis function (RBF) kernel. The linear kernel is suitable for linearly separable data, while the polynomial and RBF kernels handle non-linear relationships effectively. The RBF kernel, in particular, maps the data into a higher-dimensional space, enabling the model to capture intricate patterns. The flexibility of the kernel function allows SVR to adapt to various data types, making it a versatile tool for regression tasks.

Tuning these hyperparameters is essential to balance model complexity and computational efficiency. For instance, a high value of C may lead to overfitting by capturing noise in the training data, while a low value might underfit, failing to capture essential patterns. Similarly, selecting an appropriate value for ϵ is critical; too high a value might ignore relevant data points, whereas too low a value may make the model overly sensitive to noise. Choosing the right kernel function and its parameters can significantly enhance the model's performance, ensuring it effectively handles both linear and non-linear relationships. The hyperparameter settings for the displays being studied are summarized in Table 4.

Table 4: Key Hyperparameters for Support Vector Regression SVR of Three Displays.

| LCD ASUS display | | | | |
|-------------------|--------|-----------------|---------|--------|
| Colour | MSE | Hyperparameters | | |
| | | C | Epsilon | Kernel |
| R | 92.84 | 14.895 | 0.795 | RBF |
| G | 26.02 | 14.167 | 0.723 | RBF |
| B | 9.69 | 13.962 | 0.922 | RBF |
| QLED MSI display | | | | |
| Colour | MSE | Hyperparameters | | |
| | | C | Epsilon | Kernel |
| R | 27.7 | 14.911 | 0.177 | RBF |
| G | 19 | 14.671 | 0.776 | RBF |
| B | 4.19 | 14.636 | 0.987 | RBF |
| OLED ASUS display | | | | |
| Colour | MSE | Hyperparameters | | |
| | | C | Epsilon | Kernel |
| R | 114.95 | 14.794 | 0.992 | RBF |
| G | 73.83 | 14.933 | 0.657 | RBF |
| B | 31.651 | 12.627 | 0.695 | RBF |

3.6. Mix model

In this model, we leverage an ensemble approach to enhance the prediction accuracy of RGB values for the displays types in this study. This method combines the strengths of two established regression techniques: Random Forest Regressor (RFR) and Support Vector Regressor (SVR).

Tables 3 and 4 present the Mean Squared Error (MSE) values for each RGB component (red, green, and blue) across three display types: ASUS LCD, MSI QLED, and ASUS OLED.

In the ASUS LCD display, SVR achieves the lowest MSE for both the red (R) and blue (B) components, demonstrating its efficacy in accurately predicting these color channels. However, for the green (G) component, RFR exhibits the lowest MSE , indicating its suitability in this specific scenario. The MSI QLED display, we obtain the mix approach. Similar to the LCD display, SVR demonstrates superior performance in predicting the red (R) and blue (B) components. Conversely, RFR delivers the lowest MSE for the green (G) component.

While the proposed ensemble approach effectively combines RFR and SVR for LCD and QLED displays, it's not directly

applicable to the ASUS OLED display due to technical limitations. Since RFR surpasses SVR in terms of overall *MSE* for the OLED display, employing RFR as the sole prediction model might be the most suitable strategy.

3.7. Multivariate Polynomial Regression (MVPR)

This section investigates the application of polynomial regression for modeling the relationship between spatial coordinates (X, Y, Z) and their corresponding *RGB* color values in digital environments. Polynomial regression offers a versatile approach to capture non-linear relationships between variables, making it suitable for the complex and continuous nature of the *RGB* color space.

The implementation process involved polynomial features, transformation, and model evaluation. Polynomial features were generated using `PolynomialFeatures`, creating new matrices containing all polynomial combinations of the features up to a specified degree.

A crucial step was determining the optimal polynomial degree. This involved fitting polynomial regression models of varying degrees and evaluating their performance on both training and validation datasets using Mean Squared Error (*MSE*) as the metric. The elbow method [59] was employed to identify the optimal degree by plotting the *MSE* against different degrees and selecting the point where the *MSE* decrease slows down. This approach balances model complexity with performance, avoiding underfitting and overfitting.

The evaluation process involved data scaling, polynomial transformation for degrees ranging from 1 to 17, model fitting and prediction on training and validation sets, and error calculation using *MSE*. The elbow method revealed that a polynomial degree of around 6 offered the best balance for each *RGB* component (R, G, B). The final model with this degree was assessed using *MSE* on the training and validation datasets.

The performance of the Multivariate Polynomial Regression (MVPR) model varied across different displays. The ASUS-VA27E achieved an *MSE* of 8.48 with polynomial degrees of 5, 6, and 7 for R, G , and B components, respectively. The QLED MSI display achieved the best results with a significantly lower *MSE* of 1.82 using a degree of 6 for all components. In contrast, the ASUS OLED Gaming display exhibited higher prediction errors, reflected in an *MSE* of 16.84 with a degree of 7 for all components.

3.8. Decision Trees (DT)

A grid search with cross-validation was employed to identify the optimal hyperparameters for the decision tree model. This involved systematically evaluating various combinations of parameters, including the splitting criterion (gini or entropy) and maximum tree depth (3 to 21), while monitoring performance through cross-validation. The data preparation stage involved scaling the features to ensure a consistent range. Following hyperparameter selection, separate decision tree models were trained for each *RGB* component (R, G, B) using the identified optimal parameters. Finally, model performance was evaluated on both the training and test datasets, with Mean Squared Error (*MSE*) used as the evaluation metric.

Decision trees exhibited varying levels of success in predicting *RGB* values across the three displays. The ASUS-VA27E display yielded the poorest performance, with a high *MSE* of 217.30, indicating significant prediction inaccuracies. Conversely, the QLED MSI display demonstrated better results with a considerably lower *MSE* of 73.48, suggesting the model's ability to effectively capture the color characteristics of this specific display. The ASUS OLED Gaming display achieved a moderate performance with an *MSE* of 112.77, surpassing the VA27E but not reaching the level of the QLED MSI. These findings highlight the influence of display-specific features on model performance and emphasize the necessity for tailored optimization to achieve accurate color predictions.

A closer examination revealed mixed performance across individual *RGB* components. The model struggled with the red channel (R), evidenced by a high *MSE* (202.28) for the OLED display. This could be attributed to factors such as over-fitting or the inherent characteristics of the data for the R component being less suited for the decision tree's splitting criteria. Conversely, the model performed well for the green (G) and blue (B) components, achieving low *MSE*s of 41.78 and 36.00 respectively for the OLED display. This indicates the decision tree's effectiveness in capturing underlying patterns within the data for these channels, making it a reliable predictor for both green and blue.

3.9. K-Nearest Neighbors (KNN)

KNN is a non-parametric machine learning algorithm well-suited for both classification and regression tasks. Its core principle involves assigning the average value of the k -nearest neighbors in the feature space to the target point. In this context, the feature space represents the spatial coordinates, and the target values are the *RGB* values.

The k -NN model was implemented using the `scikit-learn` library [60]. A crucial step involved determining the optimal number of neighbors (k) that minimizes prediction errors. This was achieved by evaluating the model's performance with different k values and analyzing the Mean Squared Error (*MSE*) for each *RGB* component (Red, Green, Blue).

The results revealed that the optimal number of neighbors varied depending on the display. For the QLED MSI display, the model achieved the best performance with 19 neighbors, resulting in *MSE* values of 123.45 (R), 90.45 (G), and 40.82 (B). This indicates a reasonable prediction accuracy, with the lowest *MSE* observed for the blue component. The ASUS-VA27E display presented a higher overall *MSE* (93.19), suggesting a greater challenge for the KNN model in predicting *RGB* values accurately. Interestingly, the ASUS OLED Gaming Display exhibited the highest *MSE* (144.01), implying a significant difficulty in capturing the color characteristics of this display using KNN.

3.10. Neural Network (NN)

A standard feed-forward neural network architecture was employed for predicting *RGB* values from XYZ color coordinates. The network comprised: (a) An input layer with three neurons corresponding to the X, Y , and Z coordinates; (b) Two

hidden layers followed, each containing 256 and 64 neurons, respectively, with ReLU activation functions, and (c) The final output layer, formed by three neurons, representing the predicted red, green, and blue values, also utilizing ReLU activation.

During the training process, the Mean Squared Error (MSE) served as the loss function, measuring the discrepancy between the predicted and actual *RGB* values. The Adam optimizer [61], with a learning rate of 0.00005, facilitated gradient descent for efficient model training. Determining optimal hyperparameters involved experimentation and analysis of the MSE curve. For the *XYZ* domain, the learning rate remained constant at 5.0×10^{-6} , while the number of epochs varied across displays (ranging from 300 to 800) and the batch size was set to 64 for all displays.

The neural network exhibited varying performance in predicting *RGB* values from *XYZ* coordinates for the different displays. The ASUS-OLED achieved the most favorable outcome with the lowest MSE recorded during validation (3.79). The QLED MSI display followed closely with a lower MSE (1.08) compared to the ASUS-VA27E display (5.93).

3.11. Explainable evolutionary multi-objective algorithm combining new linguistic fuzzy grammar and a novel interpretable linear extension (*LING_{eXplainable}*)

This method is a two-stage tree-based hybrid evolutionary multi-objective algorithm [36] designed to obtain interpretable fuzzy rules for regression problems. It features a new extension of the classic fuzzy linguistic grammar based on the Composed Fuzzy Linguistic Term Sets (CFLTSS) and a novel interpretable linear extension, both proposed in [36]. This effectively reduces the number of rules and maximizes the semantic interpretability using GM3M [62] (Geometric Mean of 3 Metrics, which measures semantic interpretability at the database level) and RMI [62] (Rule Meaning Index, which measures semantic interpretability at the rulebase level). The method also minimizes the Mean Square Error (MSE) while maintaining rule length within reasonable bounds.

It is based on two stages: In the first stage, the algorithm employs an embedded multi-objective evolutionary learning process to derive initial linguistic partitions and rules. It aims to minimize the number of rules and the error through a fast linguistic tree-based rule learning technique that extends the M5-prime method; In the second stage, an advanced multi-objective evolutionary algorithm refines the solutions by fine-tuning membership functions and selecting rules to further minimize the number of rules, enhance the interpretability (GM3M and RMI improvements) and reduce errors.

This method obtains interpretable and quite accurate results and is capable of automatically learning its parameters. It automatically determines the appropriate number of membership functions, variables, and rules, enabling the creation of simple models with very few rules and semantically well formed membership functions.

4. Results and discussion

In order to correctly characterize how display models work and the colour reproduction capability each of these displays, the reproduced colour has to be objectively measured. In our case, an X-rite i1Display Pro Plus colorimeter was used. Three displays were used in this study:

- **ASUS VA27EHE Eye Care LED:** This 27-inch display features a resolution of 1920×1080 pixels.
- **MSI G274QPF-QD Monitor:** This 27-inch monitor is based on WQHD technology, and its resolution is 2560×1440 pixels.
- **ASUS ROG Swift OLED PG27AQDM:** This 27-inch display features a resolution of 2560×1440 pixels and is based on OLED technology.

Table 5 summarizes the key image quality and performance specifications of the three displays. These include: (a) Display panel technology, (b) Resolution, (c) Refresh rate, (d) Response time, (e) Color gamut, (f) Contrast ratio, (g) Viewing angle, and (h) Brightness.

Model performance was assessed using the ΔE_{00} visual error [63, 64] between the desired colors of the validation dataset and the displayed colors using *RGB* inputs computed by each method. Results for the models considered in section 3, are summarized in Tables 6 - 8.

Tables 6, 7, and 8 present the performance metrics for each model and for each display, including: (a) The mean square error between computed RGB values and original RGB values used in the measurement of the validation dataset (MSE_{RGB}), (b) The mean of the visual error (ΔE_{00}), (c) The standard deviation (STD) of ΔE_{00} , and (d)-(f) The number of colors for which the model produces a correct display value, with a $\Delta E_{00} \leq 1$, or it makes a small mistake, i. e., $1 < \Delta E_{00} \leq 2$, and the number of colors for which the error would be noticeable to a typical observer, i. e., $\Delta E_{00} > 2$.

Table 6 presents the results for the ASUS VA27EHE Eye Care LED display. The RIT (physical) model shows the best performance, followed by NN_{XYZ} with a performance that is close to that shown by FMID using 5 clusters, and then MVPR. For these models, the average ΔE_{00} value is less than one. However, the difference between these models lies in the number of colors for which the model makes a clear error. Furthermore, it is notable that the RIT model has no colors with ΔE_{00} greater than 2, highlighting its precision in color reproduction.

In contrast, FMID and NN_{XYZ} , while performing well overall, produce 132 and 154 colors with $\Delta E_{00} > 2$, respectively. This suggests that the RIT model physical assumptions (i.e., constant chromaticity of the primaries, as well as an additive color generation model) hold well for this display, explaining its superior performance. Machine learning methods such as FMID, NN_{XYZ} , and MVPR also perform well for many colors, but other models with an average $\Delta E_{00} > 1$ fall within the human discrimination threshold, displaying a larger number of wrongly rendered colors. FisPro and FMTB exhibit the highest

Table 5: Key specifications for three display monitors under investigation: ASUS VA27EHE, MSI G274QPF-QD, and ASUS PG27AQDM. This table highlights the key features relevant to image quality and performance.

| Feature | ASUS VA27EHE Eye Care LED | MSI G274QPF-QD Monitor | ASUS ROG Swift OLED PG27AQDM |
|---------------------------------|---------------------------|-------------------------------|--------------------------------|
| Display Type | LED | QLED | OLED |
| Panel Size (inches) | 27 | 27 | 26.5 |
| Resolution | 1920 × 1080 (Full HD) | 2560 × 1440 (WQHD) | 2560 × 1440 (WQHD) |
| Refresh Rate (Hz) | 75 | Up to 165 | 240 |
| Response Time (ms) | 5 (GTG) | 1 (MPRT) | 0.03 (GTG) |
| Color Gamut | 100% sRGB | 95% DCI-P3 (estimated) | 99% DCI-P3 |
| Contrast Ratio | 1000:1 | 1000:1 (typical) | 1,500,000:1 |
| Viewing Angles (degrees) | 178 (H/V) | 178 (H/V) | 178 (H/V) |
| Brightness (cd/m ²) | 250 (typical) | 300 (typical), 400 (HDR peak) | 450 (typical), 1000 (HDR peak) |

error, with $\Delta E_{00} > 2$, and therefore a bigger number of color errors easily noticeable to humans, and therefore making FisPro unsuitable for accurate color display applications.

Table 7 shows the corresponding analysis of the QLED MSI display. RIT emerges as the best-performing method with the lowest mean ΔE_{00} of 0.30. It also has the fewest colors with a ΔE_{00} greater than 2 demonstrating its accuracy and consistency. This is closely followed by FMIDXYZ with 8 clusters, which has a mean ΔE_{00} of 0.40, showing competitive performance. FMIDXYZ manages to keep the number of colors with ΔE_{00} above 2 to a minimal 17. In contrast, the machine learning methods such as NN_{XYZ} and MVPR also perform well but with slightly higher error metrics. NN_{XYZ} has a mean ΔE_{00} of 0.60, while MVPR shows a mean ΔE_{00} of 0.57. Although these models exhibit more colors with ΔE_{00} greater than 2 compared to RIT and FMID, they still maintain a relatively low count, making them viable alternatives for accurate color rendering.

On the other hand, models like FMTB-MAM and FMTB-TAK show significantly higher error values, with mean ΔE_{00} values of 10.25 and 10.12, respectively. These models also exhibit high MSE_{RGB} values and a large number of colors with $\Delta E_{00} > 2$, indicating substantial perceptual errors. Similarly, models like DT and $LING_{explainable}$ also demonstrate poor performance, with mean ΔE_{00} values of 2.93 and 3.92, respectively.

The results for the third display (ASUS ROG Swift OLED PG27AQDM) appear in Table 8. Among the models tested, the neural network model NN_{XYZ} demonstrated the best overall performance with the lowest mean ΔE_{00} of 1.23. It also had the highest number of colors within the non-perceptually distinguishable error range ($\Delta E_{00} \leq 1$), for 3824 colors. However, it still produced 1251 colors with a $\Delta E_{00} > 2$, indicating noticeable errors to human observers. Following closely, the FMIDXYZ with 8 clusters had a mean ΔE_{00} of 1.37, slightly higher than NN_{XYZ} , but it performed well overall with 3105 colors within the non-perceptually distinguishable range and 1644 colors with significant errors.

RIT showed competitive performance, with a mean ΔE_{00} of 1.48. However, it had fewer colors in the non-perceptually distinguishable range compared to NN_{XYZ} and FMID, with 2027 colors, and a higher number of significant errors (1969 colors). This indicates that while RIT maintains good performance, its accuracy is outperformed by the top two machine learning models. On the other hand, models such as FMTB-MAM and FMTB-TAK exhibited the poorest performance, with mean

ΔE_{00} values of 12.56 and 11.13, respectively. These models also produced the most significant number of colors with $\Delta E_{00} > 2$, making their color errors highly visible. Models like DT, KNN, and $LING_{explainable}$ also performed poorly, with high mean ΔE_{00} values and substantial numbers of significant errors.

Overall, RIT, NN_{XYZ} , and FMID showed superior performance in terms of ΔE_{00} for the three displays under analysis, achieving high accuracy and consistency in color rendering. These models stood out for their ability to reproduce colors with minimal perceptual errors, ensuring that the displayed colors closely matched the desired colors. The RIT model showed remarkable performance due to its assumptions about constant chromaticity and additive color generation, which held true for many colors on the displays tested. The NN_{XYZ} model, despite being a neural network often referred to as a "black box" due to its complex and opaque internal processes, exhibited exceptional accuracy. It managed to produce the lowest mean ΔE_{00} .

FMID, not only performed competitively in terms of the ΔE_{00} value, but it also has additional advantages in terms of interpretability. Unlike the NN_{XYZ} model, FMID allows for a clearer understanding of the steps involved in the modeling process. This interpretability can be crucial for diagnostic and optimization purposes, since it allows to understand how specific decisions are made within the model. This transparency is a key advantage, making FMID a valuable tool not just for its performance but also for its ability to provide insights into the modeling process itself.

Figures 1, 2, and 3 show the ΔE_{00} visual error metrics in terms of the luminance Y and chrominance xy values. These figures offer a visual representation of the color accuracy achieved by FMID, RIT, NN_{XYZ} , and MVPR models.

Figure 1 shows the analysis corresponding to the ASUS VA27EHE Eye Care LED display. In this case, RIT shows varying performance depending on the luminance level (Y). Specifically, the RIT model performs worse as Y increases, indicating lower accuracy for brighter colors. Conversely, it performs better for lower Y values, suggesting higher accuracy for darker colors. The NN_{XYZ} model also struggles with increasing Y values, particularly for bright greenish and yellowish colors, where its performance deteriorates significantly. This indicates a limitation in handling high luminance levels, which is critical for accurate color rendering in brighter scenes. The FMID model demonstrates a more balanced performance across different luminance levels, with errors remaining low at both ends of the luminance spectrum. This suggest, FMID works worse for cer-

tain chromaticities like those close to cyan, grey or dark red. Lastly, the MVPR model shows higher error in both dark and bright colors, with 32 data points exceeding 5 visual error units and better performance for medium luminances.

In terms of the chromaticity coordinates (xy), both FMID and NN_{XYZ} exhibit non-negligible errors for a small number of colors, particularly dark greenish and bluish colors. Despite these errors, they still show acceptable performance. The RIT model, however, exhibits excellent performance across all colors in the xy space, indicating its superior ability to accurately render colors regardless of their chromaticity.

In the analysis of the QLED MSI display shown in Figure 2, RIT generally exhibits strong performance across various luminance levels, maintaining accuracy except for higher errors observed in darker colors. This suggests that while the RIT model can handle a wide range of luminance levels effectively, it encounters challenges with lower luminance as well as mid luminance yellows and bright greens. The error of the FMID model decreases as the luminance (Y) increases, indicating that FMID struggles to render colors accurately at lower luminance levels. The NN_{XYZ} model presents a lower error at high luminance levels and variable error in lower luminances depending on chromaticity: it seems worse for light yellows and cyans and some very bright reds. This suggests that NN_{XYZ} struggles with both very dark color and certain chromaticities, making it less reliable across the full spectrum of luminance. Similarly, MVPR exhibits higher errors at lower luminance and medium luminance red, oranges, yellows and cyans.

Examining Figure 3 for the ASUS ROG Swift OLED PG27AQDM display, it shows that RIT exhibits increasing errors as the luminance (Y) increases, particularly in greenish and yellowish colors but also high error for dark blue colors. This trend indicates that the RIT model struggles with accurately rendering brighter hues within these specific color ranges. Conversely, the NN_{XYZ} and FMID models display a different error distribution pattern. Both models exhibit peak errors in dark reddish and bluish colors, indicating a significant challenge in accurately rendering these hues. Despite this, both models show suitable performance for brighter colors, maintaining a higher degree of accuracy at elevated luminance levels compared to the RIT model. While the MVPR model generally performs worse overall, it shows particularly high errors in rendering black, dark green, and dark red colors.

5. Conclusions

This study investigated the efficacy of various characterization models for accurate color reproduction (RGB), for three commercially available displays, with different properties: (1) ASUS LCD VA27EHE, (2) MSI QLED G274QPF-QD, and (3) ASUS OLED PG27AQDM. We aimed to assess the performance of these models by computing the visual error metric ΔE_{00} between the desired colors of a validation dataset, and the displayed colors using RGB inputs computed by each method. To measure the colors accurately and creating the training and validation datasets, we used an X-rite i1Display Pro Plus colorimeter. Our goal was to determine which models best repro-

duce accurate colors in RGB , ensuring minimal perceptual errors.

Overall, the RIT, NN_{XYZ} , and FMID models demonstrated superior performance in terms of ΔE_{00} across all displays, achieving high accuracy and consistency in color rendering. The traditional physical model, RIT, performed particularly well for the LCD displays due to its assumptions aligning with their operating principles. Specifically, the RIT model's effectiveness can be attributed to its reliance on the constant chromaticity of primaries and additive color generation models, which are well-suited for LCD technology. However, its effectiveness diminished for the OLED display, highlighting the limitations of relying solely on physical models for diverse display technologies.

Conversely, the NN_{XYZ} neural network exhibited an excellent accuracy, particularly for the OLED display. Its performance was marked by the lowest mean ΔE_{00} values, indicating superior color reproduction. However, its lack of interpretability makes it a "black box" model, hindering insights into its decision-making process. While such a model can be effective for specific applications where accuracy is paramount, it offers limited value in understanding and optimizing display performance. This opacity can be a significant drawback when diagnosing issues or attempting to fine-tune the model for improved results.

The Fuzzy Modeling and Identification (FMID) method emerged as a compelling alternative. It demonstrated competitive performance across all displays, rivaling the accuracy of the other two models. More importantly, FMID offers valuable interpretability. This transparency allows for a deeper understanding of the model's decision-making process, aiding in diagnostics and optimization efforts. By analyzing the model's reasoning, we can identify potential issues and refine the characterization process for enhanced color accuracy. The interpretability of FMID is particularly advantageous in practical applications where understanding the cause of errors and adjusting parameters can lead to significant improvements in performance. Given the advantages of FMID and its interpretability, we have included detailed information about the learned model in the Appendix A for the LCD display model only as an illustrative example. While we focus on the interpreting the LCD model here, a similar analysis can be conducted for QLED and OLED displays using the information provided in the other tables and figures.

This study highlights the strengths of these models and offers insights into their applicability for different displays, contributing to advancements in color science and technology. The findings suggest that a hybrid approach, leveraging the strengths of both physical and machine learning models, could be the most effective strategy for achieving optimal color accuracy across various display technologies. The RIT model's robust performance on LCDs, the NN_{XYZ} model's exceptional accuracy on OLEDs, and the FMID model's balance of accuracy and interpretability collectively provide a comprehensive toolkit for addressing diverse color reproduction challenges.

Future work could explore integrating the interpretability of FMID with the high accuracy of neural networks, poten-

Table 6: The analysis table for LCD ASUS display of $MS E_{RGB}$, mean of the ΔE_{00} visual error, visual error standard deviation (STD), and number of points bounded by error equal to 1 (non perceptually distinguishable) and 2. The best result is highlighted in red, and the second-best result is highlighted in blue.

| Method | $MS E_{RGB}$ | Mean ΔE_{00} | STD | $\Delta E_{00} \leq 1$ | $1 < \Delta E_{00} \leq 2$ | $\Delta E_{00} > 2$ |
|--------------------------------|--------------|----------------------|-------|------------------------|----------------------------|---------------------|
| FMID _{XYZ} 5 Clusters | 6.27 | 0.69 | 0.40 | 7822.00 | 1307.00 | 132.00 |
| NN _{XYZ} | 6.01 | 0.66 | 0.45 | 7606.00 | 1501.00 | 154.00 |
| PI | 36.86 | 1.58 | 0.77 | 2163.00 | 4782.00 | 2316.00 |
| RIT | 10.35 | 0.58 | 0.27 | 8588.00 | 673.00 | 0.00 |
| DT | 219.57 | 3.55 | 2.38 | 463.00 | 2264.00 | 6534.00 |
| FMTB-MAM | 1101.29 | 11.77 | 7.54 | 84.00 | 169.00 | 9008.00 |
| FMTB-TAK | 989.50 | 11.32 | 6.55 | 21.00 | 149.00 | 9091.00 |
| KNN | 104.81 | 2.98 | 1.95 | 415.00 | 2788.00 | 6058.00 |
| MVPR | 8.87 | 0.74 | 0.77 | 7416.00 | 1410.00 | 435.00 |
| SVR | 42.84 | 1.42 | 1.10 | 3775.00 | 3864.00 | 1622.00 |
| RFR | 60.81 | 1.93 | 1.25 | 2180.00 | 3592.00 | 3489.00 |
| Mix1 | 39.58 | 1.51 | 1.24 | 3398.00 | 3793.00 | 2070.00 |
| FisPro | 2815.35 | 19.71 | 10.10 | 0.00 | 5.00 | 9256.00 |
| LING _{eXplainable} | 159.75 | 3.68 | 2.78 | 993.00 | 1694.00 | 6574.0 |

Table 7: The analysis table for Qled MSI display of $MS E_{RGB}$, mean of the ΔE_{00} visual error, visual error standard deviation (STD), and number of points bounded by error equal to 1 (non perceptually distinguishable) and 2. The best result is highlighted in red, and the second-best result is highlighted in blue.

| Method | $MS E_{RGB}$ | Mean ΔE_{00} | STD | $\Delta E_{00} \leq 1$ | $1 < \Delta E_{00} \leq 2$ | $\Delta E_{00} > 2$ |
|--------------------------------|--------------|----------------------|------|------------------------|----------------------------|---------------------|
| FMID _{XYZ} 8 Clusters | 0.91 | 0.40 | 0.27 | 8973.00 | 271.00 | 17.00 |
| NN _{XYZ} | 1.16 | 0.60 | 0.35 | 8296.00 | 920.00 | 45.00 |
| PI | 20.63 | 1.48 | 0.80 | 2760.00 | 4519.00 | 1982.00 |
| RIT | 1.52 | 0.30 | 0.16 | 9246.00 | 14.00 | 1.00 |
| DT | 73.48 | 2.93 | 1.83 | 211.00 | 3406.00 | 5644.00 |
| FMTB-MAM | 749.57 | 10.25 | 7.04 | 38.00 | 180.00 | 9043.00 |
| FMTB-TAK | 722.07 | 10.12 | 5.60 | 7.00 | 97.00 | 9157.00 |
| KNN | 53.50 | 2.64 | 1.64 | 124.00 | 4162.00 | 4975.00 |
| MVPR | 1.82 | 0.57 | 0.35 | 8115.00 | 1114.00 | 32.00 |
| SVR | 23.59 | 1.14 | 1.05 | 5616.00 | 2466.00 | 1179.00 |
| RFR | 26.15 | 1.79 | 1.02 | 1572.00 | 4984.00 | 2705.00 |
| Mix2 | 19.23 | 1.32 | 0.97 | 4198.00 | 3560.00 | 1503.00 |
| LING _{eXplainable} | 124.45 | 3.92 | 2.89 | 744.00 | 1763.00 | 6754.00 |

Table 8: The analysis table for OLED ASUS display of $MS E_{RGB}$, mean of the ΔE_{00} visual error, visual error standard deviation (STD), and number of points bounded by error equal to 1 (non perceptually distinguishable) and 2.

| Method | $MS E_{RGB}$ | Mean ΔE_{00} | STD | $\Delta E_{00} \leq 1$ | $1 < \Delta E_{00} \leq 2$ | $\Delta E_{00} > 2$ |
|-------------------------------|--------------|----------------------|------|------------------------|----------------------------|---------------------|
| FMID _{XYZ} 8 Cluster | 6.77 | 1.37 | 0.71 | 3105.00 | 4512.00 | 1644.00 |
| NN _{XYZ} | 4.33 | 1.23 | 0.65 | 3824.00 | 4186.00 | 1251.00 |
| PI | 67.63 | 2.42 | 1.15 | 986.00 | 2589.00 | 5686.00 |
| RIT | 24.10 | 1.48 | 0.58 | 2027.00 | 5265.00 | 1969.00 |
| DT | 116.52 | 3.45 | 2.03 | 463.00 | 1601.00 | 7197.00 |
| FMTB-MAM | 706.74 | 12.56 | 8.41 | 10.00 | 93.00 | 9158.00 |
| FMTB-TAK | 786.93 | 11.13 | 7.14 | 42.00 | 152.00 | 9067.00 |
| KNN | 144.01 | 3.35 | 1.83 | 196.00 | 1752.00 | 7313.00 |
| MVPR | 16.84 | 1.70 | 1.00 | 2328.00 | 4049.00 | 2884.00 |
| SVR | 73.58 | 2.11 | 1.47 | 2128.00 | 3260.00 | 3873.00 |
| RFR | 39.36 | 2.33 | 1.13 | 630.00 | 3475.00 | 5156.00 |
| LING _{eXplainable} | 201.56 | 5.29 | 3.23 | 205.00 | 653.00 | 8403.00 |

tially developing new models that combine the best of both approaches. Overall, this study provides a foundation for ongoing research and development in the field of color science, with the potential to improve the quality and accuracy of digital displays in various applications.

6. Acknowledgements

This work was supported by Generalitat Valenciana under grant IMaLeVICS CIAICO-2022-051 and Spanish Agencia Estatal de Investigación under grant PID2022-140189OB-C21. We would like to express our sincere gratitude to Eduardo

Gómez Fernández for his help in the experimental part of this study.

References

- [1] J. Prats-Climent, et al, A study of neural network-based lcd display characterization, in: London Imaging Meeting, volume 2021, Society for Imaging Science and Technology, 2021, pp. 97–100.
- [2] R.S. Berns, Methods for characterizing crt displays, Displays 16 (1996) 173–182.
- [3] J.E. Gibson, M.D. Fairchild, Colorimetric characterization of three computer displays (lcd and crt), Munsell color science laboratory technical report 40 (2000).

- [4] P. Yeh, C. Gu, Optics of liquid crystal displays, volume 67, John Wiley & Sons, 2009.
- [5] M.D. Fairchild, Color appearance models, John Wiley & Sons, 2013.
- [6] E.A. Day, L. Taplin, R.S. Berns, Colorimetric characterization of a computer-controlled liquid crystal display, Color Research & Application 29 (2004) 365–373.
- [7] J. Bryant, W. Kester, Data converter architectures, in: Analog-Digital Conversion, Analog Devices, 2005, pp. 1–14.
- [8] C. Fernandez-Maloigne, Advanced color image processing and analysis, Springer Science & Business Media, 2012.
- [9] D.H. Brainard, D.G. Pelli, T. Robson, Display characterization, Signal Process 80 (2002) 2–067.
- [10] M. Fairchild, D. Wyble, Colorimetric characterization of the apple studio display (flat panel lcd) (1998).
- [11] P.C. Hung, Colorimetric calibration in electronic imaging devices using a look-up-table model and interpolations, Journal of Electronic Imaging 2 (1993) 53–61.
- [12] J. Malo, M. Luque, Colorlab: A matlab toolbox for color science and calibrated color image processing, Servei de Publicacions de la Universitat de Valencia (2002).
- [13] P. Capilla, M.A. Diez-Ajenjo, M.J. Luque, J. Malo, Corresponding-pair procedure: a new approach to simulation of dichromatic color perception, JOSA A 21 (2004) 176–186.
- [14] Y. Wang, H. Xu, Colorimetric characterization of liquid crystal display using an improved two-stage model, Chinese Optics Letters 4 (2006) 432–434.
- [15] H. Ban, H. Yamamoto, A non-device-specific approach to display characterization based on linear, nonlinear, and hybrid search algorithms, Journal of Vision 13 (2013) 20–20.
- [16] J.M. Kim, S.W. Lee, Universal color characterization model for all types of displays, Optical Engineering 54 (2015) 103103–103103.
- [17] R. Penrose, On best approximate solutions of linear matrix equations, in: Mathematical Proceedings of the Cambridge Philosophical Society, volume 52, Cambridge University Press, 1956, pp. 17–19.
- [18] D.G. Kleinbaum, L.L. Kupper, A. Nizam, E.S. Rosenberg, Applied regression analysis and other multivariable methods, Cengage Learning, 2013.
- [19] P. Sinha, Multivariate polynomial regression in data mining: methodology, problems and solutions, Int. J. Sci. Eng. Res 4 (2013) 962–965.
- [20] K.H. Zou, K. Tuncali, S.G. Silverman, Correlation and simple linear regression, Radiology 227 (2003) 617–628.
- [21] V. Shah, S.C.K. Jagupilla, D.A. Vaccari, D. Gebler, Non-linear visualization and importance ratio analysis of multivariate polynomial regression ecological models based on river hydromorphology and water quality, Water 13 (2021) 2708.
- [22] M. Su, Q. Zhong, H. Peng, Regularized multivariate polynomial regression analysis of the compressive strength of slag-metakaolin geopolymer pastes based on experimental data, Construction and Building Materials 303 (2021) 124529.
- [23] A.E. Hoerl, R.W. Kennard, Ridge regression: applications to nonorthogonal problems, Technometrics 12 (1970) 69–82.
- [24] J. Ranstam, J. Cook, Lasso regression, Journal of British Surgery 105 (2018) 1348–1348.
- [25] H. Abdi, D. Valentin, B. Edelman, Neural networks, 124, Sage, 1999.
- [26] S. Haykin, N. Network, A comprehensive foundation, Neural networks 2 (2004) 41.
- [27] L.A. Zadeh, Fuzzy sets, Information and control 8 (1965) 338–353.
- [28] T. Terano, K. Asai, M. Sugeno, Fuzzy systems theory and its applications, Academic Press Professional, Inc., 1992.
- [29] M. Amiri, Morteza, Spectral reflectance reconstruction using fuzzy logic system training: Color science application, Sensors 20 (2020) 4726.
- [30] T.J. Ross, Fuzzy logic with engineering applications, John Wiley & Sons, 2009.
- [31] R. Babuška, Fuzzy modeling for control, volume 12, Springer, 2012.
- [32] K. Almutairi, S. Morillas, P. Latorre-Carmona, M. Dansoko, A fuzzy logic inference system for display characterization, in: Iberian Conference on Pattern Recognition and Image Analysis, Springer, 2023, pp. 54–66.
- [33] S. Guillaume, B. Charnomordic, Parameter optimization of a fuzzy inference system using the fispro open source software, in: 2012 IEEE International Conference on Fuzzy Systems, IEEE, 2012, pp. 1–8.
- [34] MathWorks, Fuzzy Logic Toolbox User's Guide, MathWorks, 2023. URL: https://ch.mathworks.com/help/pdf_doc/fuzzy/fuzzy_ug.pdf.
- [35] MathWorks, Fuzzy Logic Toolbox Release Notes, MathWorks, 2023. URL: https://ch.mathworks.com/help/pdf_doc/fuzzy/rn.pdf.
- [36] C. Biedma-Rdquez, M.J. Gacto, A. Anguita-Ruiz, J. Alcalá-Fdez, R. Alcalá, Transparent but accurate evolutionary regression combining new linguistic fuzzy grammar and a novel interpretable linear extension, International Journal of Fuzzy Systems 24 (2022) 3082–3103.
- [37] G. Guo, H. Wang, D. Bell, Y. Bi, K. Greer, Knn model-based approach in classification, in: On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3–7, 2003. Proceedings, Springer, 2003, pp. 986–996.
- [38] S. Zhang, X. Li, M. Zong, X. Zhu, D. Cheng, Learning k for knn classification, ACM Transactions on Intelligent Systems and Technology (TIST) 8 (2017) 1–19.
- [39] B. Charbuty, A. Abdulazeez, Classification based on decision tree algorithm for machine learning, Journal of Applied Science and Technology Trends 2 (2021) 20–28.
- [40] R. Kumar, M. Verma, Classification algorithms for data mining: A survey, International Journal of Innovations in Engineering and Technology (IJiet) 1 (2012) 7–14.
- [41] B. Mahesh, Machine learning algorithms-a review, International Journal of Science and Research (IJSR).[Internet] 9 (2020) 381–386.
- [42] G. Stein, B. Chen, A.S. Wu, K.A. Hua, Decision tree classifier for network intrusion detection with ga-based feature selection, in: Proceedings of the 43rd annual Southeast regional conference-Volume 2, 2005, pp. 136–141.
- [43] K. Mittal, D. Khanduja, P.C. Tewari, An insight into 'decision tree analysis'', World Wide Journal of Multidisciplinary Research and Development 3 (2017) 111–115.
- [44] M. Bramer, Avoiding overfitting of decision trees, Principles of data mining (2007) 119–134.
- [45] L. Breiman, Random forests, Machine learning 45 (2001) 5–32.
- [46] J. Furnkranz, Decision Tree, Springer US, Boston, MA, 2010, pp. 263–267. URL: https://doi.org/10.1007/978-0-387-30164-8_204.
- [47] M. Bramer, Avoiding Overfitting of Decision Trees, Springer London, London, 2013, pp. 121–136.
- [48] F. Zhang, L.J. O'Donnell, Support vector regression, in: Machine learning, Elsevier, 2020, pp. 123–140.
- [49] A.J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and computing 14 (2004) 199–222.
- [50] A. Patle, D.S. Chouhan, Svm kernel functions for classification, in: 2013 International Conference on Advances in Technology and Engineering (ICATE), 2013, pp. 1–9.
- [51] M. Awad, R. Khanna, Support Vector Regression, Apress, Berkeley, CA, 2015, pp. 67–80. URL: https://doi.org/10.1007/978-1-4302-5990-9_4. doi:10.1007/978-1-4302-5990-9_4.
- [52] N. Cristianini, E. Ricci, Support Vector Machines, Springer US, Boston, MA, 2008, pp. 928–932. URL: https://doi.org/10.1007/978-0-387-30162-4_415. doi:10.1007/978-0-387-30162-4_415.
- [53] D.E. Gustafson, 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, IEEE, 1979, pp. 761–766.
- [54] L.X. Wang, J.M. Mendel, et al., Fuzzy basis functions, universal approximation, and orthogonal least-squares learning, IEEE transactions on Neural Networks 3 (1992) 807–814.
- [55] T. Chai, R.R. Draxler, et al., Root mean square error (rmse) or mean absolute error (mae), Geoscientific model development discussions 7 (2014) 1525–1534.
- [56] D. Wang, D. Tan, L. Liu, Particle swarm optimization algorithm: an overview, Soft computing 22 (2018) 387–408.
- [57] M. Madić, M. Radovanović, Optimization of machining processes using pattern search algorithm, International Journal of Industrial Engineering Computations 5 (2014) 223–234.
- [58] A. Natekin, A. Knoll, Gradient boosting machines, a tutorial, Frontiers in neuroinformatics 7 (2013) 21.
- [59] M. Syakur, B.K. Khotimah, E. Rochman, B.D. Satoto, Integration k-means clustering method and elbow method for identification of the best customer profile cluster, in: IOP conference series: materials science and

- engineering, volume 336, IOP Publishing, 2018, p. 012017.
- [60] O. Kramer, O. Kramer, Scikit-learn, Machine learning for evolution strategies (2016) 45–53.
 - [61] Z. Zhang, Improved adam optimizer for deep neural networks, in: 2018 IEEE/ACM 26th international symposium on quality of service (IWQoS), Ieee, 2018, pp. 1–2.
 - [62] M. Galende, M.J. Gacto, G. Sainz, R. Alcalá, Comparison and design of interpretable linguistic vs. scatter frbss: Gm3m generalization and new rule meaning index for global assessment and local pseudo-linguistic representation, Information Sciences 282 (2014) 190–213.
 - [63] M.R. Luo, G. Cui, B. Rigg, The development of the cie 2000 colour-difference formula: Ciede2000, Color Research & Application 26 (2001) 340–350.
 - [64] G. Sharma, W. Wu, E.N. Dalal, The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations, Color Research & Application 30 (2005) 21–30.

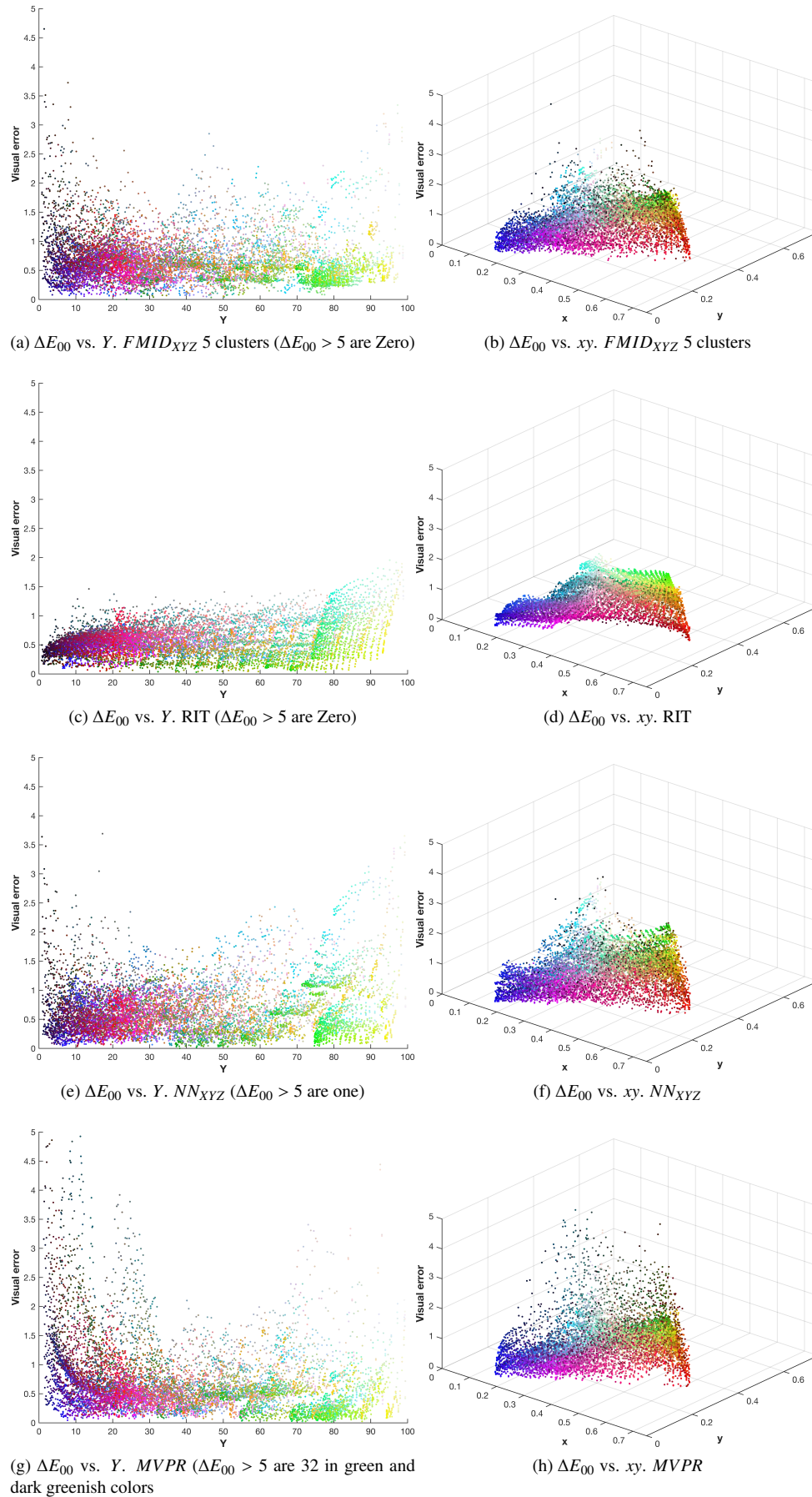


Fig. 1: ΔE_{00} visual error analysis of ASUS LCD display for four trained models ($FMID_{XYZ}$ with 5 clusters, RIT model, NN_{XYZ} , and $MVPR$). First column: ΔE_{00} versus Y luminance values of desired colors; Second column: ΔE_{00} versus xy chrominance values of desired colors.

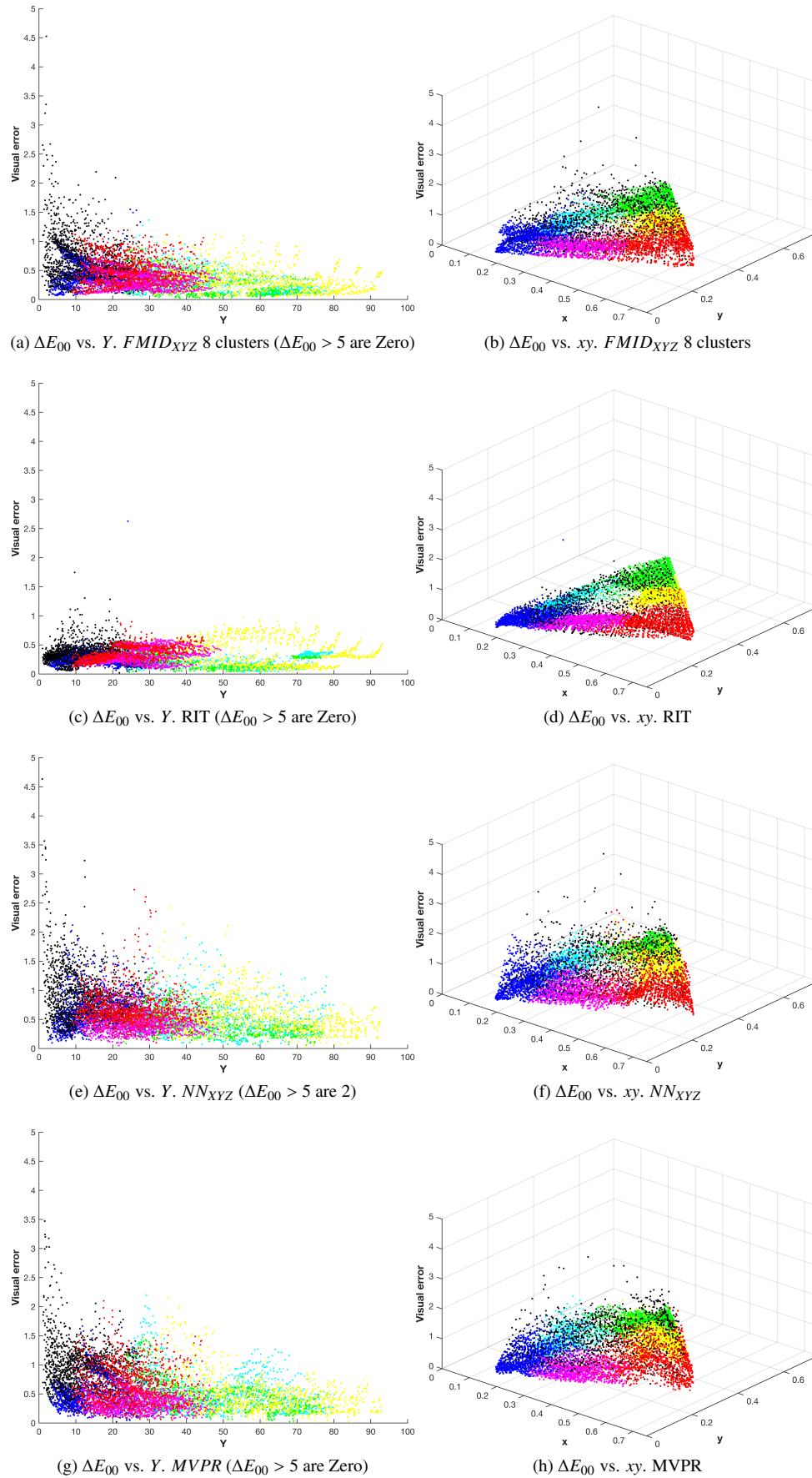


Fig. 2: ΔE_{00} visual error analysis of MSI QLED display for four trained models ($FMID_{XYZ}$ with 8 clusters, RIT model, NN_{XYZ} , and MVPR). First column: ΔE_{00} versus Y luminance values of desired colors; Second column: ΔE_{00} versus xy chrominance values of desired colors.

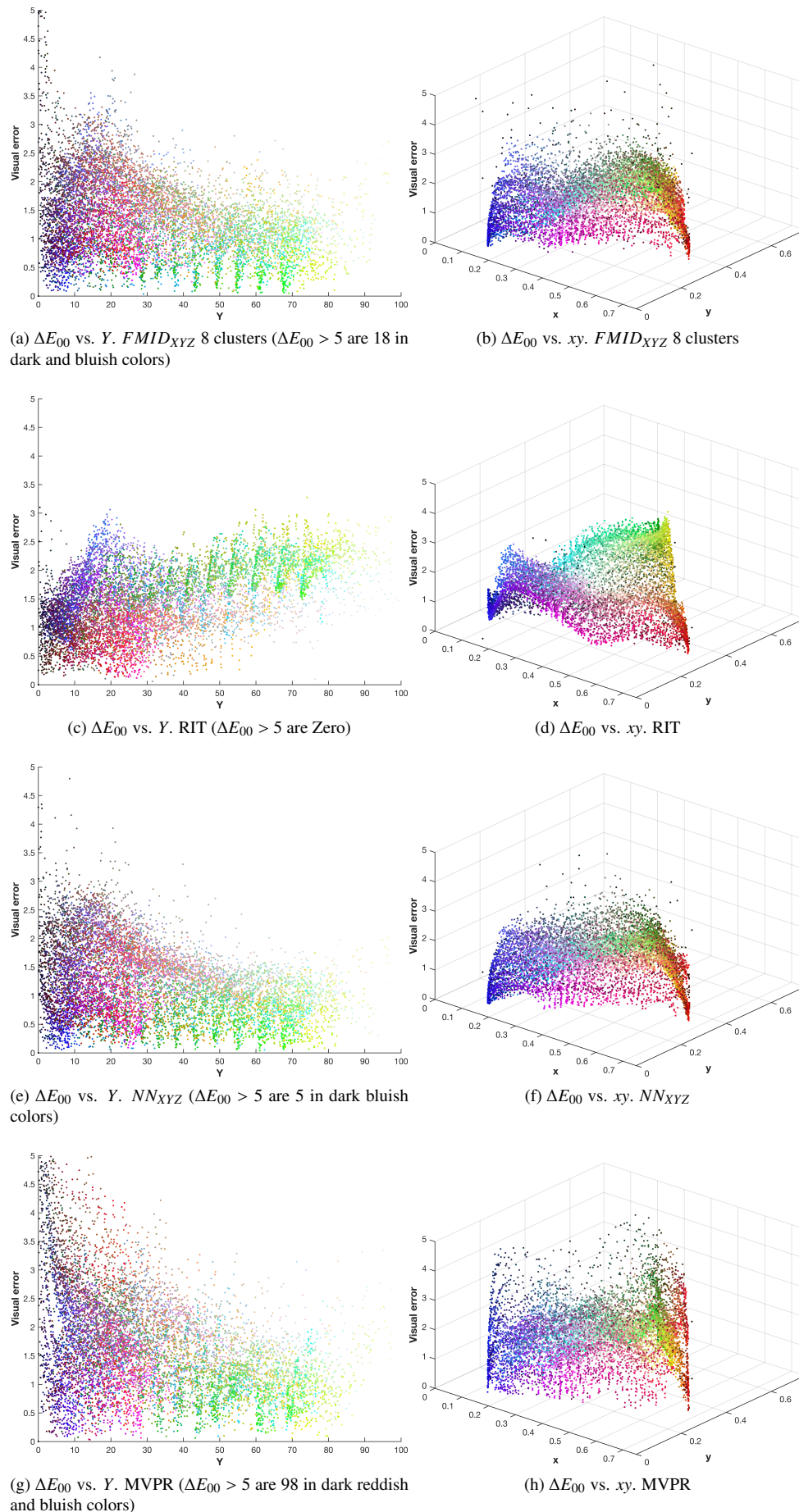


Fig. 3: ΔE_{00} visual error analysis of ASUS OLED display for four trained models ($FMID_{XYZ}$ with 8 clusters, RIT model, NN_{XYZ} , and MVPR). First column: ΔE_{00} versus Y luminance values of desired colors; Second column: ΔE_{00} versus xy chrominance values of desired colors.

Table A.1: $MS E_{RGB}$ performance varying the number of rules between 2 – 15 and the overlapping parameter m between 1.25 – 2.25 for the FIS system trained for ASUS LCD display.

| | Clusters | | | | | | | | | | | | | | |
|------|----------|------|------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|--|
| m | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | |
| 1.25 | 34.0 | 12.6 | 8.3 | 6.6 | 6.4 | 6.2 | 5.8 | 6.1 | 5.7 | 5.8 | 4.9 | 4.9 | 4.8 | 5.5 | |
| 1.5 | 34.5 | 14.0 | 8.9 | 6.3 | 6.5 | 6.2 | 5.8 | 5.3 | 5.7 | 4.9 | 5.0 | 4.7 | 4.6 | 5.2 | |
| 1.75 | 36.3 | 16.3 | 10.1 | 7.4 | 6.8 | 6.1 | 4.7 | 5.1 | 4.8 | 5.2 | 5.6 | 5.2 | 5.2 | 5.4 | |
| 2 | 39.9 | 22.5 | 11.6 | 8.6 | 6.9 | 6.1 | 5.7 | 5.0 | 2.7 | 2.9 | 2.9 | 3.3 | 3.1 | 3.4 | |
| 2.25 | 45.3 | 27.1 | 13.5 | 12.8 | 9.5 | 7.8 | 6.8 | 4.7 | 5.0 | 5.6 | 3.7 | 3.1 | 3.0 | 2.8 | |

A. Appendix

This section contains implementation details and complementary results for the FMID models used to characterize the displays. We aim to highlight the key factors contributing to their excellent RGB prediction performance and interpretability. The model developed for the ASUS LCD display will be used as an example, since it exemplifies the general structure with its 5 clusters/rules. For the other displays, the same principles that we will use to describe the ASUS display, apply except for the addition of more rules, for marginal cases.

Our focus will be on the most important parameters which impact the $MS E_{RGB}$ performance: (1) The type of response functions used within each rule to predict RGB values. These functions are the mathematical core that translates the display model understanding display into the corresponding RGB predictions. (2) The number of clusters (rules) in the system. This determines the level of detail the model might be able to obtain. In this sense, more clusters allow for a finer-grained analysis, potentially leading to improved accuracy. (3) The overlapping parameter m between the clusters' membership functions, often referred to as the fuzziness factor. This parameter controls the degree of overlap between clusters, impacting the model's ability to handle imprecise data.

First, we analyse performance depending on the type of activation function in the so-called consequents. We considered linear, parabolic and square root functions. In all cases, using XYZ values as inputs, we found the linear option is the best in terms of $MS E_{RGB}$ independently of the number of clusters.

Second, we analyze the performance as a function of the number of rules (clusters) that will be used in the model and the overlapping degree between the clusters (m value in FMID). We varied the number of rules in the 2 – 15 range, and the m parameter, in the 1.25 – 2.25 range. Performance in terms of $MS E_{RGB}$ is shown in Tables A.1, A.2 and A.3. For the ASUS LED display, we can see that $m = 1.5$ is a good choice as it provides a good performance for many numbers of rules. A decision about the number of rules is more complex. Optimal performance is achieved considering higher number of rules but having so many rules makes the system more difficult to interpret. However, performance does not drop much when slightly reducing the number of rules, making it easier to interpret. We consider this to be the best trade-off between performance and interpretability. Therefore, we conclude that it could also be worth to build an FIS with only 5 rules.

For the system interpretation we need to provide a linguistic meaning to the fuzzy inference rules. For that we can have a look first at Tables A.4, A.5, and A.6. We will explain the ta-

ble A.4 for the ASUS LCD display and the other tables for the other displays follow the same reasoning. This table shows the centers of each one of the 5 clusters for each one of the RGB outputs. We can see that clusters for the red component R , the input X consistently shows a strong positive influence, while Y and Z generally have negative impacts. The green component G is predominantly influenced by the Y input, with Y showing a positive effect across all rules, while X typically reduces G , and Z has a minor positive role. For the blue component B , the Z input plays a crucial role, consistently exhibiting a strong positive influence, Y generally having a negative effect and X a minimal one. However, the cluster centers are only the peak of the membership functions in the antecedent of the rules. Therefore, for a better understanding we must have a look at these functions and analyze what they are modeling. Figures A.1, A.2, A.2 show the membership function of each output for each displays.

By analyzing all this information we can identify an overall behaviour consisting on the 5 rules for each R , G and B channels, 5 levels for the correspondingly related variables X , Y , and Z , in each case. This behaviour is expected. In addition, each output ruleset accounts for the correlation with the less related variables in different ways. This latter point is more display dependent. In more detail, the implication rules can be interpreted as follows:

For the red component, we have the following rules:

1. IF X is low AND Y is low, AND Z is medium THEN use f_1^R .
2. IF X is low AND Y is low, AND Z is medium THEN use f_2^R .
3. IF X is medium-low AND Y is low, AND Z is low THEN use f_3^R .
4. IF X is medium AND Y is medium, AND Z is medium-high THEN use f_4^R .
5. IF X is high AND Y is high, AND Z medium THEN uses f_5^R .

For the red component, the rules primarily consider combinations of the three variables X , Y , and Z at various levels. Interestingly, the first two rules have identical conditions, whilst they call for different functions. This might suggest a need for redundancy or a context-based choice between the two functions which is not immediately apparent from the conditions alone. Rules 4 and 5 deal with higher values for X and Y , and slightly different Z values.

For the green component, the following rules apply:

1. IF X is low AND Y is low AND Z is low THEN use f_1^G .

Table A.2: $MS E_{RGB}$ performance varying the number of rules between 2 – 15 and the overlapping parameter m between 1.25 – 2.25 for the FIS system trained for MSI QLED display.

| | Clusters | | | | | | | | | | | | | |
|------|----------|------|------|------|------|------|------|------|------|------|------|-----|-----|-----|
| m | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 1.25 | 84.5 | 30.9 | 16.6 | 11.5 | 8.1 | 6.6 | 6.4 | 6.0 | 5.8 | 4.0 | 3.6 | 3.9 | 3.6 | 3.4 |
| 1.5 | 85.9 | 32.8 | 18.2 | 11.6 | 7.9 | 6.7 | 6.4 | 5.8 | 5.3 | 4.3 | 3.9 | 3.8 | 3.5 | 3.5 |
| 1.75 | 89.3 | 35.8 | 19.3 | 12.7 | 9.3 | 7.1 | 6.9 | 6.3 | 6.0 | 5.6 | 4.5 | 4.4 | 4.1 | 4.0 |
| 2 | 95.9 | 40.6 | 22.3 | 15.2 | 11.9 | 8.6 | 8.2 | 7.6 | 7.4 | 6.4 | 6.0 | 5.5 | 5.1 | 4.6 |
| 2.25 | 105.3 | 47.1 | 27.0 | 19.0 | 14.9 | 11.4 | 10.4 | 9.8 | 9.5 | 8.7 | 7.8 | 7.5 | 7.3 | 6.5 |
| 2.5 | 115.5 | 54.8 | 32.8 | 23.5 | 18.7 | 15.0 | 13.2 | 12.3 | 11.8 | 11.1 | 10.3 | 9.7 | 9.0 | 9.0 |

Table A.3: $MS E_{RGB}$ performance varying the number of rules between 2 – 15 and the overlapping parameter m between 1.25 – 2.25 for the FIS system trained for ASUS OLED display.

| | Clusters | | | | | | | | | | | | | |
|------|----------|------|------|------|------|------|------|------|------|------|------|------|-----|------|
| m | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 1.25 | 72.1 | 26.8 | 14.6 | 10.6 | 8.4 | 7.4 | 7.3 | 6.5 | 6.0 | 4.4 | 4.2 | 4.7 | 4.1 | 4.0 |
| 1.50 | 73.5 | 28.4 | 15.8 | 10.9 | 8.3 | 7.4 | 7.2 | 6.7 | 6.3 | 5.4 | 4.9 | 4.7 | 4.1 | 4.2 |
| 1.75 | 76.6 | 31.2 | 17.1 | 12.2 | 9.4 | 7.8 | 7.7 | 7.3 | 6.7 | 6.3 | 5.4 | 5.3 | 4.9 | 5.0 |
| 2.00 | 82.4 | 35.4 | 20.0 | 14.5 | 11.7 | 9.3 | 9.1 | 8.6 | 8.6 | 7.3 | 6.8 | 6.5 | 6.2 | 5.9 |
| 2.25 | 90.6 | 41.1 | 24.1 | 17.6 | 14.2 | 11.7 | 11.1 | 10.8 | 10.7 | 9.6 | 8.3 | 8.2 | 8.4 | 8.2 |
| 2.50 | 99.7 | 47.9 | 29.1 | 21.4 | 17.2 | 14.7 | 13.6 | 12.9 | 12.6 | 11.9 | 10.5 | 10.1 | 9.8 | 10.4 |

2. IF X is low AND Y is low AND Z low THEN use f_2^G .
3. IF X medium-low Y is medium-low AND Z medium-low THEN use f_3^G .
4. IF X is medium-high AND Y is high AND Z is medium THEN use f_4^G .
5. IF X is high AND Y is high AND Z is high THEN use f_5^G .

The green component rules also demonstrate a progression in the X , Y , and Z levels. Similarly to the red component rules, the first two rules share the same conditions, but suggest different functions. Rule 3 moves to medium-low levels for all variables, indicating a balanced intermediate state. The higher complexity comes in Rule 4 and Rule 5, where medium-high and high values for X , Y , and Z call for more specific functions, reflecting how higher values necessitate different handling in the green component.

For the blue component, the following rules apply:

1. IF X is medium-low AND Y is medium-low AND Z is low, THEN use f_1^B .
2. IF X is medium-low AND Y is low AND Z is medium-low, THEN use f_2^B .
3. IF X is medium-low AND Y is medium-low AND Z is low, THEN use f_3^B .
4. IF X is medium AND Y is medium-high AND Z is low, THEN use f_4^B .
5. IF X is high AND Y is high AND Z is high, THEN use f_5^B .

The blue component rules show a unique pattern where certain conditions are repeated across multiple rules but suggest different functions. For instance, Rules 1 and 3 both have the same conditions but use different consequent functions. This repetition suggests that despite identical conditions, different contextual applications. Rules 4 and 5 indicate a higher values of X , Y , and Z , showing more complex conditions which likely

correspond to more specific and complex behaviors in the blue component's response.

We can then extract more information about the model by analyzing the f_i^K functions, where, $K = R, G, B$, and $i = 1, \dots, 5$, used as a consequence of the rules. For each output, the most influential input variable is X for R , Y for G , and Z for B . These functions reveal a proportional relationship between the primary input and the output. The variability in the coefficients among the different rules captures the nonlinear relationships within the system, effectively modeling these relationships as piece-wise linear functions. For instance, in the red component R , functions such as $f_1^R = 21.73X - 10.99Y - 3.53Z - 2.80$ show that X has a substantial positive impact on R , while Y and Z reduce R . This pattern suggests that the red output is primarily dependent on the X input, with Y and Z serving as secondary modifiers. Similarly, in the green component G , functions like $f_1^G = -7.46X + 14.88Y + 0.61Z + 0.96$ highlight that Y has a strong positive effect on G , while X generally reduces G . The influence of Z is minimal, indicating that the green output is largely governed by the Y input, with X and Z playing smaller roles.

For the blue component B , the functions f_i^B indicate that Z is the primary factor, with functions such as $f_1^B = 0.10X - 0.78Y + 6.74Z + 2.60$. Z consistently shows a strong positive effect on B , while Y tends to have a negative impact, and X has a negligible effect. This suggests that the blue output is predominantly influenced by the Z input, with minor contributions from X and Y . Additionally, the model reveals some inverse relationships to compensate for the correlations and overlaps among the RGB primaries in the XYZ color space. These inverse relationships adjust for the inherent overlap in the color space, ensuring balanced and accurate color output. For instance, since increasing G also increases Z , and higher Y values lead to higher G values, the model compensates this by establishing an inverse relationship between B and Y . This mechanism ensures accurate color representation despite the complex inter-dependencies between the inputs and outputs.

Table A.4: Cluster centres of the trained model $FMID_{XYZ}$ of ASUS LCD display and fitted output regression TS functions for each rule and RGB output.

| Rules for R | Cluster (X, Y, Z) centroid | Output function |
|-------------|----------------------------|---|
| 1 | (25.93, 33.34, 46.96) | $R = f_1^R(X, Y, Z) = 21.73X - 10.99Y - 3.53Z - 2.80$ |
| 2 | (36.07, 19.15, 58.46) | $R = f_2^R(X, Y, Z) = 6.32X - 3.31Y - 1.03Z + 64.88$ |
| 3 | (37.61, 38.14, 11.92) | $R = f_3^R(X, Y, Z) = 6.21X - 3.17Y - 1.04Z + 65.97$ |
| 4 | (52.70, 46.31, 74.34) | $R = f_4^R(X, Y, Z) = 6.05X - 3.09Y - 0.99Z + 70.21$ |
| 5 | (67.98, 82.81, 58.91) | $R = f_5^R(X, Y, Z) = 6.22X - 3.11Y - 1.03Z + 62.70$ |
| Rules for G | Cluster (X, Y, Z) centroid | Output function |
| 1 | (24.92, 12.17, 43.81) | $G = f_1^G(X, Y, Z) = -7.46X + 14.88Y + 0.61Z + 0.96$ |
| 2 | (29.91, 21.69, 44.87) | $G = f_2^G(X, Y, Z) = -3.25X + 6.41Y + 0.26Z + 28.96$ |
| 3 | (37.98, 37.51, 46.04) | $G = f_3^G(X, Y, Z) = -2.15X + 4.23Y + 0.17Z + 52.31$ |
| 4 | (52.09, 64.58, 48.89) | $G = f_4^G(X, Y, Z) = -1.91X + 3.83Y + 0.16Z + 50.73$ |
| 5 | (57.54, 73.89, 51.27) | $G = f_5^G(X, Y, Z) = -1.51X + 3.04Y + 0.13Z + 63.76$ |
| Rules for B | Cluster (X, Y, Z) centroid | Output function |
| 1 | (31.92, 37.72, 7.61) | $B = f_1^B(X, Y, Z) = 0.10X - 0.78Y + 6.74Z + 2.60$ |
| 2 | (34.74, 19.47, 81.80) | $B = f_2^B(X, Y, Z) = 0.05X - 0.23Y + 1.49Z + 80.81$ |
| 3 | (35.74, 39.31, 24.96) | $B = f_3^B(X, Y, Z) = -0.02X - 0.32Y + 2.92Z + 33.93$ |
| 4 | (46.09, 46.80, 67.65) | $B = f_4^B(X, Y, Z) = 0.01X - 0.17Y + 1.84Z + 61.48$ |
| 5 | (60.20, 64.52, 97.12) | $B = f_5^B(X, Y, Z) = -0.02X - 0.08Y + 1.61Z + 61.47$ |

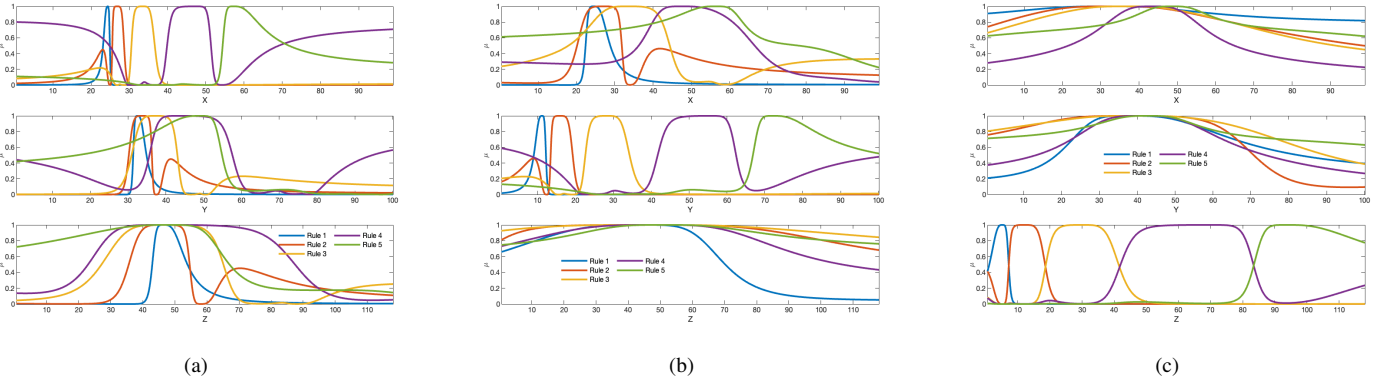
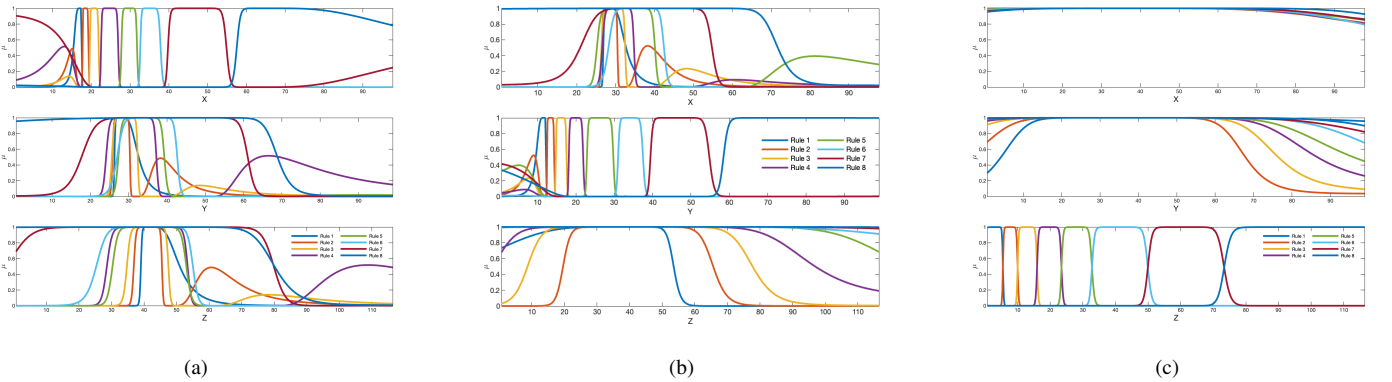
Fig. A.1: Membership functions (μ) for $FMID_{XYZ}$, for the ASUS LCD display using different clustering results for each output: (a) The first output (R), (b) The second output (G), (c) The third output (B).Fig. A.2: Membership functions (μ) for $FMID_{XYZ}$, for the MSI QLED display using different clustering results for each output: (a) The first output (R), (b) The second output (G), (c) The third output (B).

Table A.5: Cluster centres of the trained model $FMID_{XYZ}$ for MSI QLED display and fitted output regression TS functions for each rule and RGB output.

| Rules for R | Cluster (X, Y, Z) centroid | Output function |
|-------------|----------------------------|---|
| 1 | (17.08, 27.58, 41.47) | $R = f_1^R(X, Y, Z) = 32.32X - 10.85Y - 5.74Z - 9.15$ |
| 2 | (18.45, 28.21, 41.52) | $R = f_2^R(X, Y, Z) = 15.45X - 5.19Y - 2.75Z + 5.40$ |
| 3 | (21.40, 29.56, 41.59) | $R = f_3^R(X, Y, Z) = 10.12X - 3.41Y - 1.80Z + 19.01$ |
| 4 | (26.53, 31.90, 41.70) | $R = f_4^R(X, Y, Z) = 7.06X - 2.38Y - 1.26Z + 37.10$ |
| 5 | (33.62, 35.13, 41.86) | $R = f_5^R(X, Y, Z) = 5.22X - 1.76Y - 0.93Z + 57.42$ |
| 6 | (42.95, 39.40, 42.12) | $R = f_6^R(X, Y, Z) = 4.06X - 1.37Y - 0.72Z + 78.15$ |
| 7 | (58.53, 46.45, 42.42) | $R = f_7^R(X, Y, Z) = 3.44X - 1.15Y - 0.61Z + 94.21$ |
| 8 | (68.78, 51.20, 42.89) | $R = f_8^R(X, Y, Z) = -0.00045X + 0.00015Y + 0.00008Z + 252.00$ |
| Rules for G | Cluster (X, Y, Z) centroid | Output function |
| 1 | (28.13, 12.05, 40.39) | $G = f_1^G(X, Y, Z) = -9.34X + 20.51Y + 0.84Z - 6.10$ |
| 2 | (29.02, 14.58, 40.58) | $G = f_2^G(X, Y, Z) = -4.76X + 10.43Y + 0.42Z + 10.97$ |
| 3 | (30.16, 17.87, 40.79) | $G = f_3^G(X, Y, Z) = -3.39X + 7.41Y + 0.30Z + 23.63$ |
| 4 | (32.11, 23.47, 41.14) | $G = f_4^G(X, Y, Z) = -2.58X + 5.63Y + 0.23Z + 37.43$ |
| 5 | (35.19, 32.35, 41.70) | $G = f_5^G(X, Y, Z) = -1.92X + 4.19Y + 0.17Z + 57.21$ |
| 6 | (38.49, 41.85, 42.29) | $G = f_6^G(X, Y, Z) = -1.54X + 3.35Y + 0.13Z + 75.33$ |
| 7 | (42.53, 53.48, 43.02) | $G = f_7^G(X, Y, Z) = -1.30X + 2.85Y + 0.11Z + 90.33$ |
| 8 | (49.18, 72.61, 44.22) | $G = f_8^G(X, Y, Z) = -1.13X + 2.48Y + 0.10Z + 104.81$ |
| Rules for B | Cluster (X, Y, Z) centroid | Output function |
| 1 | (20.97, 30.01, 27.73) | $B = f_1^B(X, Y, Z) = 0.02X - 0.2Y + 2.61Z + 46.38$ |
| 2 | (29.06, 34.60, 30.39) | $B = f_2^B(X, Y, Z) = 0.10X - 0.87Y + 13.80Z - 8.89$ |
| 3 | (29.79, 34.79, 67.02) | $B = f_3^B(X, Y, Z) = 0.04X - 0.39Y + 6.04Z + 7.79$ |
| 4 | (31.30, 35.18, 14.48) | $B = f_4^B(X, Y, Z) = 0.02X - 0.25Y + 3.74Z + 25.62$ |
| 5 | (37.60, 36.74, 46.93) | $B = f_5^B(X, Y, Z) = 0.007X - 0.13Y + 1.94Z + 69.43$ |
| 6 | (41.75, 37.78, 68.30) | $B = f_6^B(X, Y, Z) = 0.004X - 0.10Y + 1.51Z + 92.16$ |
| 7 | (47.67, 39.25, 98.84) | $B = f_7^B(X, Y, Z) = 0.007X - 0.10Y - 1.29Z + 109.76$ |
| 8 | (50.71, 43.28, 28.54) | $B = f_8^B(X, Y, Z) = -0.002X + 0.17Y + 2.62Z + 47.17$ |

Table A.6: Cluster centres of the trained model $FMID_{XYZ}$ of the ASUS OLED display and fitted output regression TS functions for each rule and RGB output.

| Rules for R | Cluster (X, Y, Z) centroid | Output function |
|-------------|----------------------------|--|
| 1 | (16.50, 26.59, 38.53) | $R = f_1^R(X, Y, Z) = 53.98X - 20.99Y - 8.26Z - 2.189$ |
| 2 | (18.14, 28.06, 37.75) | $R = f_2^R(X, Y, Z) = 16.88X - 6.54Y - 2.56Z + 23.24$ |
| 3 | (21.30, 29.83, 33.95) | $R = f_3^R(X, Y, Z) = 9.33X - 3.62Y - 1.39Z + 42.65$ |
| 4 | (25.20, 23.14, 41.28) | $R = f_4^R(X, Y, Z) = 5.90X - 2.35Y - 0.85Z + 65.58$ |
| 5 | (40.00, 30.92, 26.99) | $R = f_5^R(X, Y, Z) = 3.99X - 1.65Y - 0.62Z + 95.62$ |
| 6 | (49.64, 65.11, 13.43) | $R = f_6^R(X, Y, Z) = 4.02X - 1.35Y - 0.86Z + 83.49$ |
| 7 | (53.91, 46.93, 87.50) | $R = f_7^R(X, Y, Z) = 3.80X - 1.54Y - 0.54Z + 96.90$ |
| 8 | (56.48, 35.06, 36.67) | $R = f_8^R(X, Y, Z) = 3.25X - 1.48Y - 0.46Z + 119.14$ |
| Rules for G | Cluster (X, Y, Z) centroid | Output function |
| 1 | (23.98, 11.48, 36.78) | $G = f_1^G(X, Y, Z) = -11.47X + 24.45Y + 0.26Z + 14.70$ |
| 2 | (24.04, 11.01, 36.74) | $G = f_2^G(X, Y, Z) = -44.97X + 95.77Y + 0.99Z - 3.41$ |
| 3 | (26.92, 18.63, 39.28) | $G = f_3^G(X, Y, Z) = -2.972X + 6.475Y + 0.10Z + 46.69$ |
| 4 | (27.20, 14.79, 40.05) | $G = f_4^G(X, Y, Z) = -5.323X + 11.411Y + 0.14Z + 28.66$ |
| 5 | (29.34, 27.08, 26.67) | $G = f_5^G(X, Y, Z) = -1.898X + 4.289Y + 0.05Z + 65.92$ |
| 6 | (34.56, 47.92, 27.80) | $G = f_6^G(X, Y, Z) = -1.313X + 2.959Y + 0.03Z + 93.03$ |
| 7 | (42.83, 68.79, 42.47) | $G = f_7^G(X, Y, Z) = -1.060X + 2.263Y + 0.03Z + 121.68$ |
| 8 | (51.27, 48.25, 58.12) | $G = f_8^G(X, Y, Z) = -1.488X + 3.260Y + 0.06Z + 87.74$ |
| Rules for B | Cluster (X, Y, Z) centroid | Output function |
| 1 | (26.26, 29.59, 10.13) | $B = f_1^B(X, Y, Z) = 0.26X - 0.37Y + 4.14Z + 37.59$ |
| 2 | (26.48, 31.26, 2.39) | $B = f_2^B(X, Y, Z) = 1.93X - 4.34Y + 39.29Z + 2.49$ |
| 3 | (29.20, 34.39, 5.07) | $B = f_3^B(X, Y, Z) = 0.40X - 0.84Y + 8.02Z + 22.43$ |
| 4 | (32.46, 35.07, 19.47) | $B = f_4^B(X, Y, Z) = 0.18X - 0.22Y + 2.83Z + 54.05$ |
| 5 | (33.13, 31.96, 33.51) | $B = f_5^B(X, Y, Z) = 0.16X - 0.11Y + 1.99Z + 71.82$ |
| 6 | (36.50, 34.53, 53.56) | $B = f_6^B(X, Y, Z) = 0.10X - 0.06Y + 1.60Z + 88.61$ |
| 7 | (38.25, 35.05, 75.34) | $B = f_7^B(X, Y, Z) = 0.06X - 0.07Y + 1.42Z + 101.18$ |
| 8 | (40.84, 36.33, 101.10) | $B = f_8^B(X, Y, Z) = 0.00X - 0.82Y + 1.16Z + 126.20$ |

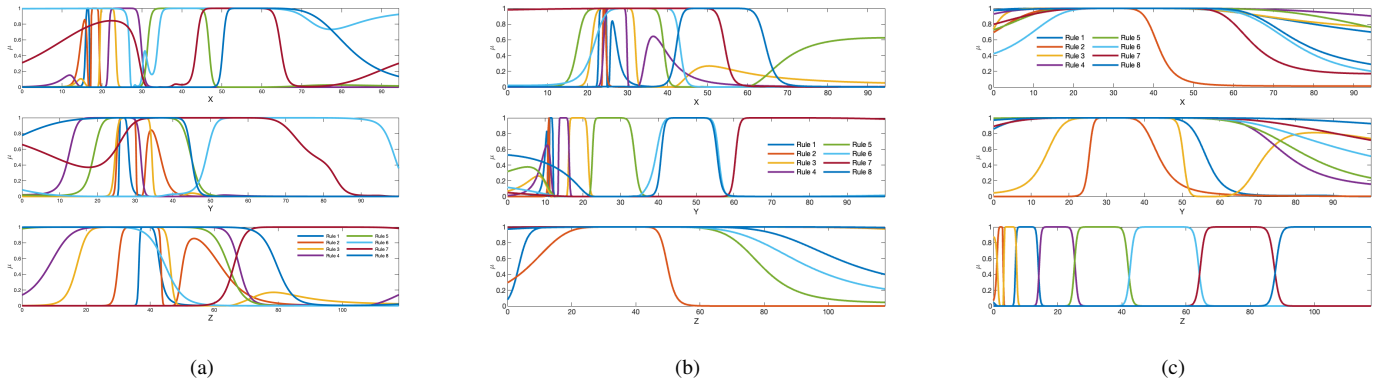


Fig. A.3: Membership functions (μ) for $FMID_{XYZ}$ model trained for ASUS OLED display, using different clustering results for each output: (a) The first output (R), (b) The second output (G), (c) The third output (B).