https://doi.org/10.2352/EI.2026.38.3.MOBMU-319

Measuring the Complexity of Image Enhancement and Restoration Algorithms Using a Logarithmic Model

***Julia Schnitzer*[1]*, Zohair Al-Ameen*[2]*, Basim Mahmood*[3]**
[1]*Department of Computer Science and Media, Brandenburg University of Applied Sciences, Brandenburg, Germany*
[2]*Computer Center, University of Mosul, Mosul, Iraq*
[3]*Department of Computer Science, College of Computer Science and Mathematics, University of Mosul, Mosul, Iraq*

Abstract

Measuring complexity is important because it delivers insight into the practicality, usability, and efficiency of imaging algorithms. Although quality assessment metrics are perceptual indices of an algorithm's effectiveness, quantifying complexity addresses the computational cost based on selected traits. Many existing enhancement and restoration algorithms can deliver high-quality results, but at the expense of heavy computational demands. This limits the utilization of such algorithms in many scenarios, especially those with resource limitations. In this context, many have used execution time (ET) as an indicator of complexity. Still, ET is not 100% precise, and a more hands-on approach should be considered. In this study, a logarithmic model is introduced to measure complexity based on ET and memory usage, providing more meaningful complexity scores. The model's outcome is a numerical value, where higher values indicate more complexity involvement. By quantifying complexity, experts can determine a balance between efficiency and practicality. This ensures that an algorithm is not only hypothetically sound but also practical for various real-world scenarios. Likewise, this facilitates fair comparison between algorithms, guiding toward both accurate and feasible answers.

Introduction

In the era of digital image proliferation, the analysis and development of image processing algorithms have become essential in various real-world applications. These algorithms range from simple special-based filters to complicated artificial intelligence and deep learning approaches [1]. Despite their ubiquitous growth, their performances cannot be determined only by the output accuracy or quality. However, memory usage, computational demands, and execution times are also critical in determining their applicability in real-world scenarios [2]. Complexity measurement (CM) of image processing algorithms provides a controlled and structured framework for assessing performance in diverse dimensions [3]. This enables practitioners and researchers to make informed decisions regarding the deployment, selection, and design of these algorithms. CM, in essence, quantifies traits such as data size, iteration count, arithmetic operations, memory usage, and execution time related to image processing, delivering scalability in practical scenarios and comparative evaluation.

Without CM, a method that performs well in lab settings or on small datasets may be unsuccessful in real-life deployment, where variability and hardware limitations introduce additional challenges. The significance of CM in image processing extends beyond mere computational assessment. From an operational standpoint, understanding algorithmic complexity enables the identification of computational bottlenecks, which can be extremely informative in terms of processing concepts, pipelines, efficiency, and practicality [4]. Moreover, CM is also central for fair comparisons and benchmarking among competing algorithms. For instance, an algorithm that slightly increases accuracy but doubles execution time is less favorable in practical solutions [5]. This highlights the need for a hands-on approach that considers key performance features to quantify complexity. This allows predictive scaling to forestall performance degradation of algorithms when computational resources change and image resolution increases.

It is also vital in scientific research, where rigorous assessment of newly developed algorithms requires an understanding of theoretical and practical performance implications. In modern research, reproducibility and practicality are becoming more serious concerns. The absence of feasible CM measures hampers the authentication of experimental claims. For instance, two algorithms may produce analogous results in terms of quality, but have substantial differences in terms of computational demands. Hence, without complexity analysis, it cannot be truthfully determined which algorithm is practically superior. As for quantitative benchmarking, CM can also facilitate comparative analysis. Thus, utilizing CM in scientific research ensures that a given algorithm is theoretically sound and practically viable [6]. Previous CM approaches are assorted, each with definite advantages and limitations. One of the most utilized approaches is the Big-O notation [7]. It utilizes execution time and image sizes for CM. Its advantage is that it can show how an algorithm scales with different image sizes.

Its limitation is that it does not account for the same image-size scenario, and execution time does not represent structural factors. Despite having several CM methods, noticeable gaps remain unanswered in this field. Existing approaches are either too abstract to capture performance behaviors or too dependent on certain dataset features. This leads to a misleading evaluation in practicality for various algorithms. Such a mislead can cause unfair comparisons, limit algorithmic structure analysis, and hamper reproducibility. Moreover, the lack of standard datasets, evaluation protocols, and experimental guidelines hinders the comparability of results across various studies. Collectively, the existing gaps highlight the need for a more practical framework for CM. Accordingly, the central motivation here is rooted in practical and scientific considerations. In practice, a CM model that utilizes always-available logarithmic features yet has a simple structure is highly needed to attain practicality. In scientific research, diverse research domains in image processing also require a CM model to achieve fair comparisons for competing algorithms.

Addressing these issues requires a model that is structurally simple, practically relevant, and adaptable to various images and environments. Hence, the key objective of this study is to introduce a new logarithmic model that measures algorithmic complexity based on execution time and memory usage, bridging theoretical notions with empirical realism. The remaining article sections are organized as follows: Section 2 presents a detailed design of the developed CM model. Section 3 provides critical usage guidelines to deliver interpretability. Section 4 reports experimental results, discussions, and highlights insights derived from real-world data analysis. Lastly, Section 5 concludes the study, stating key findings and future directions.

Proposed Model

As mentioned earlier, measuring (quantifying) complexity is essential in evaluating the efficiency of any given algorithm. It provides deeper insights into the usability and practicality of an algorithm, in addition to its visual assessment scores. It also helps to determine whether an algorithm is suitable for specific situations [8]. In this context, an effective complexity measure should be sensitive to certain meaningful traits that are usable in every situation [9]. The most noteworthy traits to consider are: execution time, memory usage, image size, iteration count, arithmetic operations, and hardware reliance [10-12]. Examining these six traits, not all of them are applicable and informative in every situation. Accordingly, when the same device captures all images and all have the same size, the image size in this situation no longer affects the computational cost. Moreover, the iteration count trait becomes unusable if the algorithm is non-iterative. Furthermore, arithmetic operations and hardware reliance are often unusable if the source code is locked or only contains the execution files. The two most important traits that are available in every saturation are execution time (ET) and memory usage (MU) [13].

ET refers to the number of calculations needed by an algorithm achieve a specific task. It is usually measured in seconds, and a lower ET designates less computational demand and faster implementation [14]. Moreover, MU refers to the amount of memory needed by an algorithm to execute and complete its task. It includes the memory used by input data and auxiliary memory for variables, data structures, recursion, and so forth. It is usually measured in bytes or its multipliers (i.e., kilobytes, megabytes, and gigabytes). It is determined by recording the difference in memory usage before and after implementing the algorithm, with the main focus on peak memory usage. A lower MU reflects reduced memory requirements and is therefore preferred [15]. In the context of complexity, ET gives insights into time complexity [16], while MU gives insights into space complexity [17]. To measure complexity based on these two traits, a customized logarithmic model is introduced, which employs and blends the average readings for ET and MU. The blending process was inspired by the one given in [18], which is a simple yet effective procedure for combining and weighing two arrays, formulated as follows:

$$Q = \beta I_1 + \left(1 - \beta\right) I_2 \tag{1}$$

where Q is the output, I_1 and I_2 are two different arrays, and β is a blending coefficient, which satisfies $0 \le \beta \le 1$. If $\beta = 0$, the output corresponds entirely to I_2. If $\beta = 1$, the output corresponds entirely to I_1. When $\beta = 0.5$, both I_1 and I_2 contribute equally (each 50%) to the resulting value. Using this notion and logarithmic scaling, the complexity score of the proposed model can be computed as:

$$\xi = \beta \log\left(T\right) + \left(1 - \beta\right) \log\left(M\right) \tag{2}$$

where ξ is a numerical value representing the complexity score; Accordingly, a lower ξ value indicates lower detected complexity, which is sought after. T and M are the average ET and MU readings obtained from processing multiple images, and β is a blending coefficient that controls the relative contribution of log(T) and log(M) in forming the complexity score ξ. By default, it is set to $\beta = 0.5$, because both time and memory are equally important. However, if one trait is more important than the other, the β value can be changed. For example, if a case demands (75%) time influence and (25%) memory influence, β can be set to $\beta = 0.75$. As for logarithmic scaling, it is used to accommodate the wide dynamic range of T and M, capture nonlinear data relationships, and more meaningfully demonstrate the changing trends across different algorithms.

Usage Guidelines

The proposed complexity-measure model should be used at an algorithm comparison level, so that each algorithm produces its own score, and the scores are compared to determine which is more favorable. Hence, measuring ET and MU is needed for several different images, preferably three or more. All operations should be performed on the same computer under the same programming environment with the same power mode (ex., high-performance mode, charger plugged in) to establish a fair and controlled comparison. Accordingly, this approach ensures the elimination of differences caused by system-level optimization, hardware performance, and interpreter/compiler efficiency. As a result, any observed differences in MU and ET can be attributed to the algorithms themselves rather than external factors.

Moreover, when performing the companions, it is recommended to disconnect the computer from the internet, close all the unnecessary applications, and pause the Windows update feature. These precautions reduce system interruptions and background processes that can consume CPU, memory, and other resources, yielding imprecise ET and MU measurements. To obtain the complexity score for each algorithm, do the following: **(i)** Apply the designated algorithm to each image and record the ET and MU. **(ii)** Find the average ET in seconds and MU in megabytes (MB) for all images. **(iii)** Input the determined averages to Eq. (2) and obtain the ξ value. **(iv)** Report the obtained value alongside the corresponding algorithm name to facilitate comparison (preferably present that in a tabular format). **(v)** Provide an all-inclusive plot that includes all the obtained scores to visualize the complexity and enable easier interpretation to reveal patterns at a glance.

Results & Analysis

This section presents and analyzes the obtained results of experiments, which are used to assess the introduced complexity measuring model. In this context, eleven image enhancement algorithms were utilized, published from 2015 to 2025. These algorithms employ different processing concepts. The utilized algorithms are: probabilistic enhancement (PE) [19], fusion-based enhancement (FBE) [20], LIME [21], LightenNet [22], gradient-based enhancement (GBE) [23], biological vision inspired framework (BVIF) [24], principal component analysis enhancement (PCAE) [25], plug-and-play Retinex (PNPR) [26], sharpening-smoothing (SHSM) [27], contrast-residual enhancement (CRE) [28], and IMS [29]. In addition, to ensure a representative and fair evaluation, images with different sizes and different illumination conditions were used as test cases.

The used evaluation framework considers both perceptual and computational perspectives with particular emphasis on algorithmic complexity. Accordingly, perceptuality is evaluated using the no-reference metric, the Natural Image Quality Evaluator (NIQE) [30]. NIQE measures perceptual naturality and outputs a numerical value, where a lower value means better naturalness. Moreover, the proposed complexity measuring model provides insights into the algorithmic complexity in different scenarios. The outcomes support a thorough comparative analysis in terms of quality and complexity. This enables an inclusive discussion of performance, efficiency, practicality, and suitability to different real-world settings. Hence, the holistic performances of the competing algorithms are given in Table 1. The graphical depiction of the ξ complexity scores in Table 1 is shown in Figure 1. The resulting images of the competing algorithms are demonstrated in Figures 2 to 4. By inspecting the data in Table 1, a multidimensional and comprehensive evaluation can be attained. It jointly includes perceptual measures, ET, MU, and ξ scores in diverse scenarios, thereby enabling advanced analysis of results. At a perceptual level, NIQE suggests that SHSM, LIME, and PCAE are the top-performing algorithms. In contrast, LightenNet and IMS are the worst performing and appear less favorable.

Table 1: Performances of the competing algorithms.

#	Algorithms	Fig		NIQE↓		ET↓ (S)		MU↓ (MB)		ξ↓ (β=0.5)	*R*	ξ↓ (β=0.75)	*R*	ξ↓ (β=0.25)	*R*
1	PE 2015	Fig 1		2.3053		45.460		1088.484		4.8916	8	4.1430	7	5.6402	7
		Fig 2		2.5419		34.811		594.589							
		Fig 3		1.9171		9.125		102.500							
		Av	*R*	2.2547	4	29.798	7	595.191	6						
2	FBE 2016	Fig 1		2.2491		10.330		201.386		3.1186	2	2.4298	2	3.8075	2
		Fig 2		2.6507		5.079		59.250							
		Fig 3		1.9004		1.700		8.437							
		Av	*R*	2.2667	6	5.703	6	89.691	2						
3	LIME 2017	Fig 1		2.0370		5.140		855.687		3.7689	3	2.4939	3	5.0438	3
		Fig 2		2.5198		3.845		630.726							
		Fig 3		1.9476		1.167		178.316							
		Av	*R*	2.1681	2	3.384	2	554.909	5						
4	LightenNet 2018	Fig 1		4.4377		196.521		7734.844		6.6344	11	5.6966	11	7.5722	11
		Fig 2		2.4736		119.616		5549.426							
		Fig 3		2.0831		33.695		1608.484							
		Av	*R*	2.9981	11	116.610	9	4964.251	11						
5	GBE 2019	Fig 1		2.3864		69.953		567.351		4.8395	7	4.2733	8	5.4057	6
		Fig 2		2.6614		38.895		475.867							
		Fig 3		1.9338		13.367		133.574							
		Av	*R*	2.3272	7	40.738	8	392.264	3						
6	BVIF 2020	Fig 1		2.3747		179.324		620.371		5.3956	9	5.0881	9	5.7030	8
		Fig 2		3.0573		140.945		469.507							
		Fig 3		2.2561		37.302		133.210							
		Av	*R*	2.5627	9	119.190	10	407.696	4						
7	PCAE 2021	Fig 1		2.0075		6.926		2531.391		4.4568	6	2.9816	6	5.9319	9
		Fig 2		2.7734		5.012		1880.887							
		Fig 3		1.8353		1.595		530.765							
		Av	*R*	2.2054	3	4.511	5	1647.681	10						
8	PNPR 2022	Fig 1		2.1828		193.692		1099.711		5.7104	10	5.2628	10	6.1581	10
		Fig 2		2.6262		136.462		871.914							
		Fig 3		2.1759		39.982		246.339							
		Av	*R*	2.3283	8	123.378	11	739.321	8						
9	SHSM 2023	Fig 1		1.9068		5.544		1302.012		4.0020	5	2.6316	4	5.3723	5
		Fig 2		2.8184		3.998		966.257							
		Fig 3		1.5815		1.049		275.289							
		Av	*R*	2.1022	1	3.530	3	847.852	9						
10	CRE 2024	Fig 1		2.2092		4.422		79.386		2.5438	1	1.8427	1	3.2449	1
		Fig 2		2.5772		3.879		59.148							
		Fig 3		2.0012		1.096		16.656							
		Av	*R*	2.2625	5	3.132	1	51.730	1						
11	IMS 2025	Fig 1		2.6185		6.233		1037.141		3.9174	4	2.6436	5	5.1911	4
		Fig 2		2.7499		4.344		650.035							
		Fig 3		2.5766		1.230		239.496							
		Av	*R*	2.6483	10	3.935	4	642.224	7						

Av → Average | R → Rank | (↓) → lower is **better** | (*β*=0.5) → Standard Setting | (*β*=0.75) → Favoring **ET** | (*β*=0.25) → Favoring **MU**

However, the perceptual standing indicated by NIQE does not essentially translate into overall algorithmic favorability. Thus, once ET, MU, and ξ scores are taken into account, the rules of analysis change. Under the standard setting, when (β=0.5), the proposed complexity measure ξ evidently distinguished practical lightweight algorithms from those that are memory-intensive. In this setting, CRE performed the best, followed closely by FBE and LIME, even though none of these algorithms are among the top-performing in NIQE. CRE in particular is interesting, as although it ranked 5th in NIQE, it exhibited extremely low ET and MU, resulting in lower ξ scores across all β settings. This highlights a key insight that above moderate performance can still be preferable due to exceptional computational efficiency. This point is often overlooked in purely quality-driven comparison or analysis. Quite the opposite, the top-scoring algorithm in NIQE, such as SHSM, attained the 5th rank under the standard setting due to its somewhat high MU reading, highlighting that perceptual distinction may come at a notable computational cost. When the weighting shifts toward ET with (β=0.75), some changes were noticed in the algorithm's ranking. Still, CRE remained dominant, followed by FBE and LIME, which maintained low ET average readings. Moreover, algorithms like LightenNet and PNPR performed poorly in this setting due to their extensive average ET readings.

Notably, the SHSM rank slightly improved (ranking 4th), signifying that its ET has compensated for its high MU. Still, it fails to keep up with CRE or FBE, whose designs attained a noticeable balance in ET and MU. In contrast, when the emphasis shifted toward MU with (β=0.25), the ranking showed a different but equally informative perspective. Accordingly, CRE secured the top position again, followed by FBE and LIME. This confirms the robustness of these algorithms across diverse operational priorities. However, algorithms with high MU, such as PCEA, PNPR, and LightenNet, fell back decidedly in this setting regardless of having competitive NIQE scores. This highlights an essential fact: algorithms such as PCAE, despite ranking 3rd in NIQE, may be attractive from a perceptual standpoint, but may be impractical for memory-constrained setups. Another illustrative case is SHSM. Despite ranking 1st in NIQE, it dropped to rank 5th under this setting. This exemplifies the key argument that high perceptual-metric readings alone do not imply overall superiority.

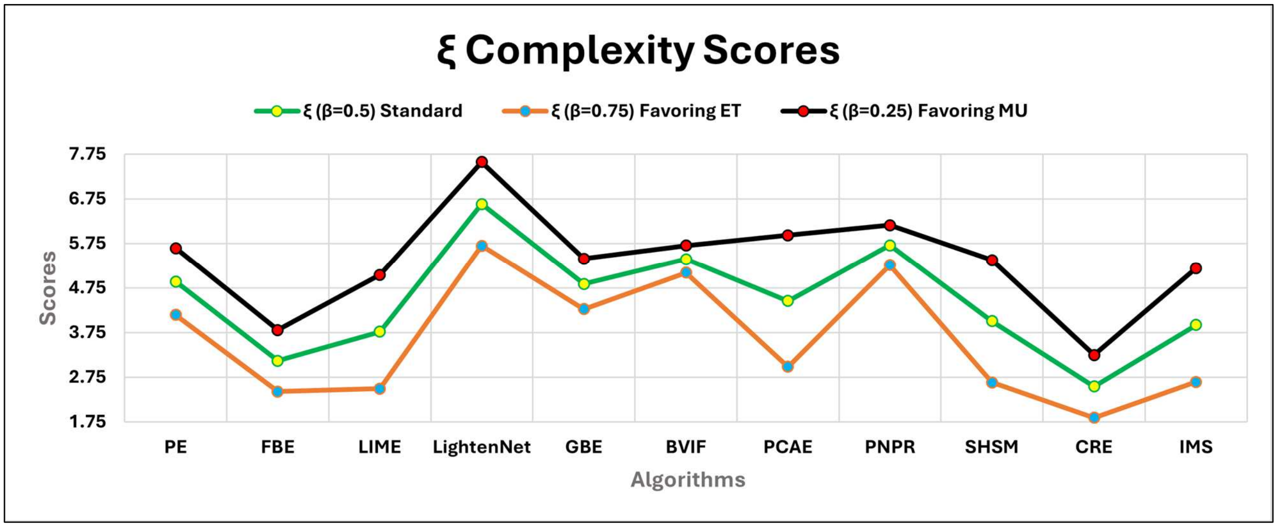

Figure 1. *Graphical representation of the ξ complexity scores given in Table 1.*

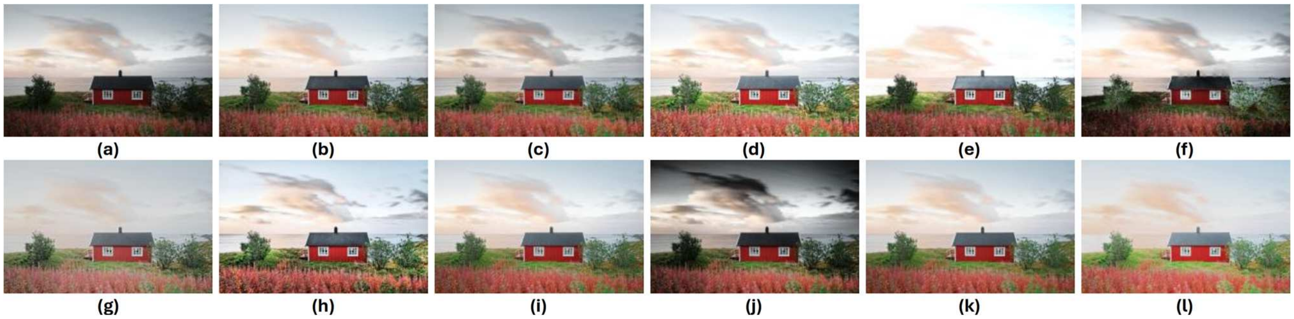

Figure 2. *The comparison outcomes* ***(Batch 1)****: (a) Pristine image (4662×2958); The upcoming images are processed by: (b) PE; (c) FBE; (d) LIME; (e) LightenNet; (f) GBE; (g) BVIF; (h) PCAE; (i) PNPR; (j) SHSM; (k) CRE; (l) IMS.*

Figure 3. *The comparison outcomes* ***(Batch 2)****: (a) Pristine image (3710×2783); The upcoming images are processed by: (b) PE; (c) FBE; (d) LIME; (e) LightenNet; (f) GBE; (g) BVIF; (h) PCAE; (i) PNPR; (j) SHSM; (k) CRE; (l) IMS.*

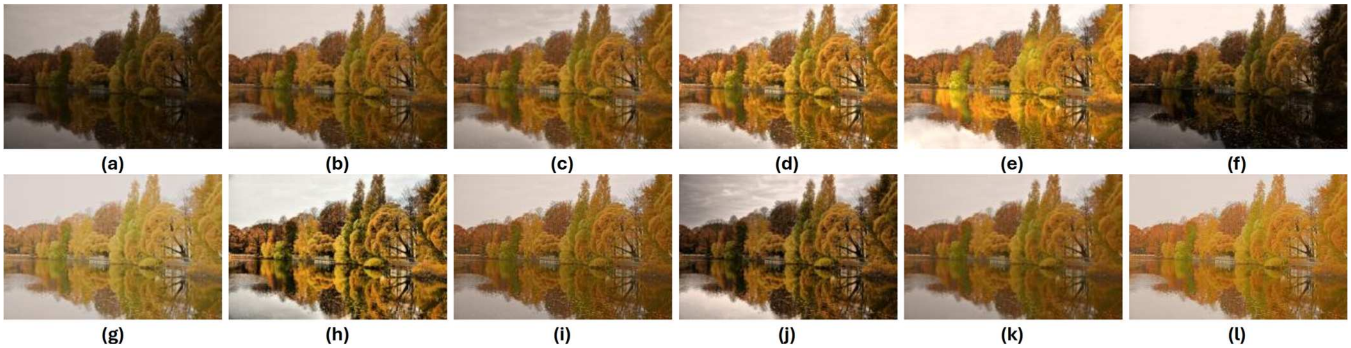

***Figure 4.** The comparison outcomes **(Batch 3)**: (a) Pristine image (2092×1392); The upcoming images are processed by: (b) PE; (c) FBE; (d) LIME; (e) LightenNet; (f) GBE; (g) BVIF; (h) PCAE; (i) PNPR; (j) SHSM; (k) CRE; (l) IMS.*

LightenNet, despite being a learning-based algorithm, was unsuccessful in attaining adequate performances in terms of perceptuality and complexity, performing the worst in both aspects. The mediocre-performing algorithms, such as GBE, PE, IMS, and BVIF, showed varying sensitivity to β settings, indicating divergent design trade-offs. As for GBE, it presented moderate MU but high ET, while BVIF is limited by its high ET measures, despite its acceptable NIQE performances. Moreover, PE showed middling performances, not highly affected by β settings. More importantly, this study demonstrated the practicality and discriminative ability of the proposed model in identifying efficient algorithms that are robust enough to be used with different application priorities. From a wider operational viewpoint, this study validates the need for multi-criteria assessment in image processing research. Moreover, while perceptual metrics are indispensable for quality evaluation, they must be understood in conjunction with complexity measures. This notion helps prevent misleading deductions and provides fair algorithmic assessment. Hence, this study offers an application-aware and more holistic basis for algorithm performance comparison and analysis. This enables practitioners and researchers to select or analyze an algorithm more fairly based on computational demands and quality expectations.

Conclusion

The proposed model provided an insightful and meaningful qualitative descriptor regarding the performances of diverse algorithms. It reflected a customized balance between ET and MU, quantifying the complexity core in a way that existing conventional measures cannot achieve. Unlike existing measures, the proposed model responded positively to changes in time and memory, which are critical and always-available factors for realistic and analytical evaluations. In this context, some algorithms exhibited consistent behaviors across different β settings, while others changed their behaviors. This demonstrated the proposed model's ability to discriminate between the performances of various algorithms in dissimilar scenarios. Likewise, the ξ model is validated as an informative indicator rather than an abstracted numerical construct. Moreover, it is believed to be a central evaluation model bridging the gap between quantification, visual quality, computational demand, and practicality. Moreover, its potential applicability across various real-world scenarios makes it generalizable beyond specific experimental arrangements. Overall, utilizing a complexity measure strengthens the analytical rigor by delivering additional dimensions of evaluation. As such, it is a vital tool for performance validation, comparative analysis, practicality assessment, and development directions for a specific algorithm. Future directions can be the inclusion of other always-available traits to be used for tailored needs. Moreover, a model that analyzes scene intricacy in an image can be utilized to deliver a more meaningful complexity score.

References

[1] R. Szeliski, Computer Vision: Algorithms and Applications. Springer Nature, 2022.

[2] S. Essahraui, A. Elhanashi, Q. Zheng, N. Almutairi, and S. Saponara, "Tiny deep learning models for real-time and efficient embedded driver state detection," Journal of Real-Time Image Processing, vol. 23, no. 1, p. 39, 2026.

[3] F. De Simone, J. Gutiérrez, and P. Le Callet, "Complexity measurement and characterization of 360-degree content," Electronic Imaging, vol. 31, pp. 1–7, 2019.

[4] H. Zenil, "A review of methods for estimating algorithmic complexity: options, challenges, and new directions," Entropy, vol. 22, no. 6, p. 612, 2020.

[5] Z. Sun, D. Yu, H. Fang, J. Yang, X. Qu, J. Zhang, and C. Geng, "Are we evaluating rigorously? Benchmarking recommendation for reproducible evaluation and fair comparison," in Proc. 14th ACM Conf. Recommender Syst., Sep. 2020, pp. 23–32.

[6] G. Fioretti, "A subjective measure of complexity," Advances in Complex Systems, vol. 2, no. 4, pp. 349–370, 1999.

[7] M. Ivanovici, R. M. Coliban, C. Hatfaludi, and I. E. Nicolae, "Color image complexity versus over-segmentation: A preliminary study on the correlation between complexity measures and number of segments," Journal of Imaging, vol. 6, no. 4, p. 16, 2020.

[8] H. Zenil, "A review of methods for estimating algorithmic complexity: Options, challenges, and new directions," Entropy, vol. 22, no. 6, p. 612, 2020.

[9] J. R. Cano, "Analysis of data complexity measures for classification," Expert Systems with Applications, vol. 40, no. 12, pp. 4820–4831, 2013.

[10] P. M. Kuhn, Algorithms, complexity analysis and VLSI architectures for MPEG-4 motion estimation. Springer Science & Business Media, 2013.

[11] M. Imran, K. Khursheed, N. Ahmad, M. O'Nils, N. Lawal, and M. A. Waheed, "Complexity analysis of vision functions for comparison of wireless smart cameras," International Journal of Distributed Sensor Networks, vol. 10, no. 1, p. 710685, 2014.

[12] D. Bertsekas and J. Tsitsiklis, Parallel and distributed computation: numerical methods. Athena Scientific, 2015.

[13] M. Maule, J. L. Comba, R. P. Torchelsen, and R. Bastos, "A survey of raster-based transparency techniques," Computers & Graphics, vol. 35, no. 6, pp. 1023–1034, 2011.

[14] Y. Fan, S. Feng, and J. Du, "Spatial data mining-focused research on image processing techniques for remotely sensed images," The Imaging Science Journal, vol. 73, no. 4, pp. 428–442, 2025.

[15] K. Yan and H. Gao, "Network information big data analysis platform based on deep learning technology," Procedia Computer Science, vol. 247, pp. 727–734, 2024.

[16] O. Appiah and J. B. Hayfron-Acquah, "Fast generation of image's histogram using approximation technique for image processing algorithms," International Journal of Image, Graphics and Signal Processing, vol. 10, no. 3, pp. 25–35, 2018.

[17] S. Hakak, A. Kamsin, P. Shivakumara, M. Y. Idna Idris, and G. A. Gilkar, "A new split based searching for exact pattern matching for natural texts," PLOS ONE, vol. 13, no. 7, p. e0200912, 2018.

[18] M. Nilsson, M. Dahl, and I. Claesson, "Gray-scale image enhancement using the SMQT," in Proceedings of the IEEE International Conference on Image Processing, vol. 1, September 2005, pp. I-933.

[19] X. Fu, Y. Liao, D. Zeng, Y. Huang, X. P. Zhang, and X. Ding, "A probabilistic method for image enhancement with simultaneous illumination and reflectance estimation," IEEE Transactions on Image Processing, vol. 24, no. 12, pp. 4965–4977, 2015.

[20] X. Fu, D. Zeng, Y. Huang, Y. Liao, X. Ding, and J. Paisley, "A fusion-based enhancing method for weakly illuminated images," Signal Processing, vol. 129, pp. 82–96, 2016.

[21] X. Guo, Y. Li, and H. Ling, "LIME: Low-light image enhancement via illumination map estimation," IEEE Transactions on Image Processing, vol. 26, no. 2, pp. 982–993, 2017.

[22] C. Li, J. Guo, F. Porikli, and Y. Pang, "LightenNet: A convolutional neural network for weakly illuminated image enhancement," Pattern Recognition Letters, vol. 104, pp. 15–22, 2018.

[23] M. Tanaka, T. Shibata, and M. Okutomi, "Gradient-based low-light image enhancement," in Proceedings of the 2019 IEEE International Conference on Consumer Electronics (ICCE), Jan. 2019, pp. 1–2.

[24] K. F. Yang, X. S. Zhang, and Y. J. Li, "A biological vision inspired framework for image enhancement in poor visibility conditions," IEEE Transactions on Image Processing, vol. 29, pp. 1493–1506, 2020.

[25] N. Singh and A. K. Bhandari, "Principal component analysis-based low-light image enhancement using reflection model," IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1–10, 2021.

[26] Y. H. Lin and Y. C. Lu, "Low-light enhancement using a plug-and-play Retinex model with shrinkage mapping for illumination estimation," IEEE Transactions on Image Processing, vol. 31, pp. 4897–4908, 2022.

[27] Y. Demir and N. H. Kaplan, "Low-light image enhancement based on sharpening-smoothing image filter," Digital Signal Processing, vol. 138, p. 104054, 2023.

[28] T. Pu and Q. Zhu, "Non-uniform illumination image enhancement via a retinal mechanism inspired decomposition," IEEE Transactions on Consumer Electronics, vol. 70, no. 1, pp. 747–756, 2024.

[29] M. Ahmadnia, M. Maghrebi, J. Wong, and A. A. F. Fini, "Illumination map smoothing (IMS): a convex and differentiable mathematical model to rapidly enhance low-light images," Signal, Image and Video Processing, vol. 19, no. 3, p. 213, 2025.

[30] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a 'Completely Blind' Image Quality Analyzer," IEEE Signal Processing Letters, vol. 20, no. 3, pp. 209–212, 2013.

Author Biography

Julia Schnitzer is Professor of Interaction Design in the Department of Computer Science and Media at Brandenburg University. Her research focuses on user-centered design of interactive environments. As part of the UNESCO Chair on the Digitalization of Cultural Heritage in MENA Regions at Risk, she has explored efficient low-cost photogrammetry workflows. Since March 2026, she has served as Vice President for Education and Internationalization.

Zohair Al-Ameen received his Ph.D. degree in Computer Science from the Technological University of Malaysia in 2015. Since 2016, he has been a tenured faculty member at the University of Mosul, where he is engaged in teaching and research. His research interests focus on developing low-complexity algorithms that produce high-quality results for digital image processing, with specific emphasis on image enhancement and restoration.

Basim Mahmood received his Ph.D. degree in Computer Science from Florida Institute of Technology, USA, in 2015. He is currently the head of the Computer Science Department at the University of Mosul. His research interests focus on Complex Networks and Systems. He is also interested in the area of digitalizing and preserving cultural heritage.


Image Engineering


STUDENT/YP SHOWCASE
PURDUE