IQM-Vis: A User-Centric Toolbox for Visualising and Evaluating Image Quality Metrics

Matt Clifford[a*], Alexander Hepburn[a], Ricardo Kleinlein[b], Jorge Vila Tomás[c], Pablo Hernández Cámara[c], Paula Dauden[c], Jesús Malo[c], Nathan Lepora[a], Valero Laparra[c], Raúl Santos-Rodríguez[a]

[a]School of Engineering Mathematics and Technology, University of Bristol, Ada Lovelace Building, Tankard's Close, Bristol, BS8 1TW United Kingdom

[b]Havard Medical School, 25 Shattuck St, Boston, MA 02115, United States

[c] Image Processing Lab, Universitat de Valencia, E4 building - 4th floor, Parc Científic Universitat de València, C/ Cat. Agustín Escardino Benlloch, 9 46980 Paterna (València). Spain

[*]Corresponding author email: `mc15445@bristol.ac.uk`

Abstract

In computer vision Image Quality Metrics (IQMs) are designed to assess image data in situations where human judgement is either too expensive to obtain or not objective enough. Although a multitude of IQMs have been proposed in the literature, it is not clear how to evaluate their suitability for different application domains. This type of evaluation needs to be both qualitative and quantitative, lending itself best to interactive graphical tools. To this end, we have created IQM-Vis, the first open source toolbox dedicated to analysing, experimenting and conducting human image perception experiments with IQMs.

Keywords

Image Quality Metrics, Human Image Perception, Perceptual Distances

Code metadata

Nr.	Code metadata description	Please fill in this column
C1	Current code version	v0.2.5.80
C2	Permanent link to code/repository used for this code version	https://github.com/mattclifford1/IQM-Vis
C3	Permanent link to Reproducible Capsule	
C4	Legal Code License	BSD-3-Clause
C5	Code versioning system used	Git
C6	Software code languages, tools, and services used	Python
C7	Compilation requirements, operating environments & dependencies	Linux, macOS, or Windows; Python 3.9+, PyQt6, PyTorch, OpenCV, NumPy, Matplotlib, Pandas, scikit-image, SciPy
C8	If available Link to developer documentation/manual	https://mattclifford1.github.io/IQM-Vis/
C9	Support email for questions	matt.clifford@bristol.ac.uk

1. Introduction

Image quality metrics (IQMs) serve as an objective evaluation of the perceived quality of an image and aim to capture how humans perceive differences between images. For example, reference IQMs compare a reference image, and the same image after a distortion is applied, as shown in Figure 1. The goal is to recreate how humans perceive this difference according to human psychophysical experiments. They are utilised in scenarios such as regularisation of deep learning models [7] and benchmarking the performance of image processing algorithms where human evaluation is too expensive to obtain [8].

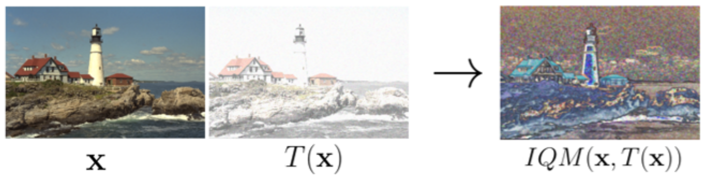

Figure 1: Example of a full reference image quality metric (IQM). A distortion is applied to image $\mathbf{x}$ to transform it into $T(\mathbf{x})$. The IQM then calculates the difference between these two images. The mean of the IQM image is often taken to produce a scalar score.

There are a plethora of such metrics to choose from, with the simplest being the distance in Euclidean space such as the mean squared error. In traditional perceptual literature, IQMs can be categorised into two groups. The first group consists of metrics which operate on the premise that the image's structure remains unchanged despite the presence of distortion. The principle these metrics adhere to is *structural similarity*. The best example of this kind is the SSIM index [17]. The second group aims to measure *visibility of error*, i.e., how much distortion is visible to humans. Traditional metrics of this type model the characteristics or processing that occurs in the human visual system. They transform both images to a more perceptually meaningful space and compute a Euclidean distance in this space. Recent literature utilises deep learning models in an attempt to mimic human perception by correlating the model's response with image quality ratings from human experiments [4, 19, 6]. Figure 2 shows a comparison of IQMs for different distortions.

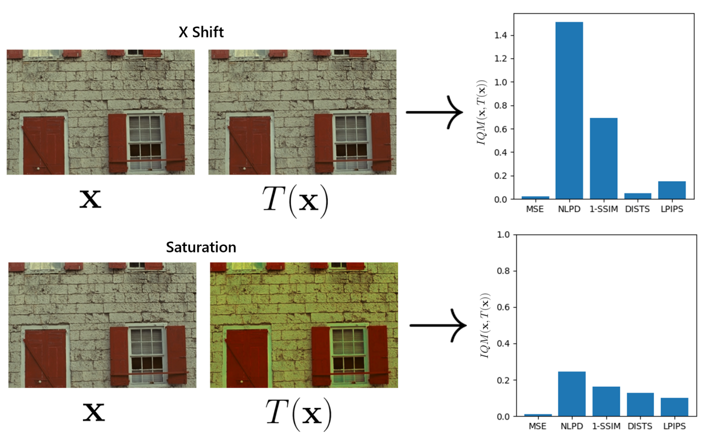

Figure 2: Comparing IQM scores for different distortions. DISTS is designed to be invariant to small spacial perturbations as shown in the top row with X shifting the image.

1.1 Evaluating IQMs

The process of evaluating IQMs is both qualitative and quantitative. It is necessary to gather empirical data on the response profiles of different IQMs with respect to specific image distortions and parameter ranges as shown in Figure 3. It is also important for a human to oversee the images produced from the distortion process to give context to the image data, as shown in Figure 4. This ensures that the desired qualities of an image distortion are present for a specific image sample.

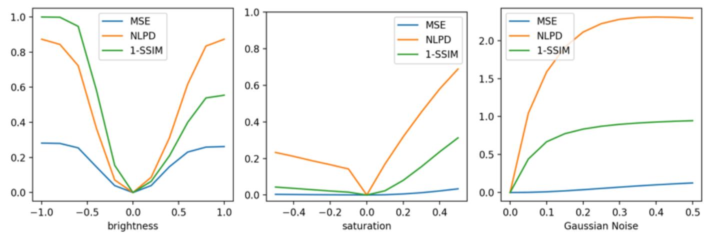

Figure 3: Quantitative evaluation of IQM response profiles over each distortion's parameter range

Figure 4: Examples of images transformed by a distortion. It is important to visualise image distortions over a their range of parameter values to understand their effects on image quality.

The practitioner may also require specific behaviour from an IQM, which is not observable in just measuring correlation between human ratings and distances given by the IQM. For example, invariances are often extremely important or specific spacial properties of an IQM, which can be viewed in Figure 5. This type of evaluation and experimentation lends itself best to *interactive* and *graphical* software.

Figure 5: Visualisation of two IQM images: SSIM and MSE.

2. Software Description

IQM-Vis is a toolbox that enables quick access for visualising image distortions and evaluating IQMs through a simple and convenient python graphical interface. It provides many standard distortions and IQMs out of the box [4, 10, 17, 19] whilst adding custom distortions, IQMs and image datasets is also straightforward. Comparison graphs of IQMs are automatically generated as well as the option to compare with human scores to understand failures of the metric with particular distortions or images. This improves the quality and timescale of the IQM evaluation process. IQM-Vis is also able to perform two alternate force choice (2AFC) experiments, popular in the perceptual literature [14, 13, 19]. The package manages the data storage from the experiments and shows the results with correlation plots against desired IQMs. Any image distortions which do not conform to the correlation can be selected for further analysis of the image properties and visually inspected by the user.

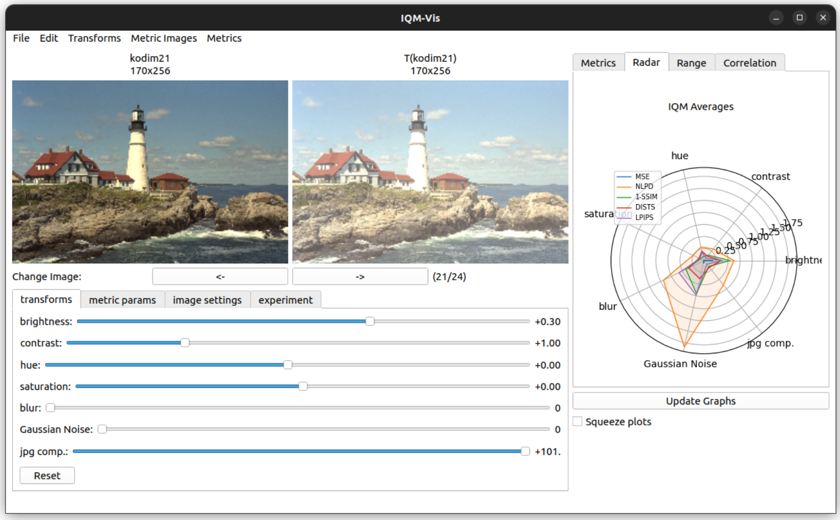

Figure 6: Screenshot of the IQM-Vis UI. Reference and distorted images are shown on the top left, parameters to view distortions are on the bottom left. Access to graphs are on the right. Additional IQM and distortions can be selected via the drop down menu bars.

2.1 Quantitative Analysis

IQM-Vis provides a comprehensive analysis of IQMs through various quantitative graphs. These graphs offer practitioners a well-rounded understanding of how different IQMs perform under different scenarios. Figure 3 shows an example of range plots produced from IQM-Vis. This compares response profiles of IQMs over a the range of a distortions parameters, showing what each IQM is specifically sensitive to. On the right hand side of Figure 6 there is a radar plot which shows the mean score of each IQM over all parameter values on a distortion. This is useful to give an overarching view of which IQMs are sensitive to each distortion. Where data is available, users can view correlations to human perception scores for each IQM.

2.2 Qualitative Analysis

IQM-Vis provides the user with multiple ways to monitor image data. As shown in Figure 4, distortion effects can be viewed by adjusting their respective parameter slides and the IQM scores can be observed on the metrics graph tab as shown in Figure 2. The user is able to view an image showing the perceived differences for IQMs which produce an image, these can be viewed by selecting them in the drop down menu and appear as shown in Figure 5. When investigating correlation to human perception scores, any specific image sample which breaks the correlation can be quickly examined by clicking on the data point. This is useful for understanding where breakdowns in correlations might occur.

2.3 Human Perception Experiments

Since most IQMs aim to capture an understanding or aspect of how humans perceive images, it is necessary to compare IQM scores to image quality rating from humans. Popular datasets of human scoring exist such as TID[14, 13], KADID-10K[11], and BAPPS[19]. However, these are static datasets using a subset of natural images and distortions. When designing and evaluating IQMs for use cases outside of these datasets it is important to run human perception experiments to gather supplementary data. This ensures that the robustness and limitations of IQMs are better understood and not limited within the scope of pre-existing datasets.

IQM-Vis has functionality for conducting human perception experiments where participants provide a ranking of the perceived quality of an image exposed to specific distortions. The ranking is determined by a two alternative forced choice (2AFC) test which is standard procedure for IQM datasets [19, 14, 13]. The 2AFC asks a test participant to choose out of two distorted images, which is most similar to a reference. The 2AFC is repeated until the underlying sorting algorithm (quick sort) provides an ordered list of the distorted images rated by the participants from most similar to the reference getting progressively less similar.

Experiments enable practitioners to analyse custom distortions and image datasets, which is crucial for real-world design and evaluation of IQMs. Integrating the capacity to conduct experiments into IQM-Vis lowers the barriers for IQM research and analysis to non computer science experts.

IQM-Vis facilitates the management and analysis of the collected data through in application correlation graphs and statistics as well as simple csv data structures for further out of application data analysis and sharing. An overview of the steps to conduct an experiments are shown in Figure 7 and can also be viewed in our demonstration video.

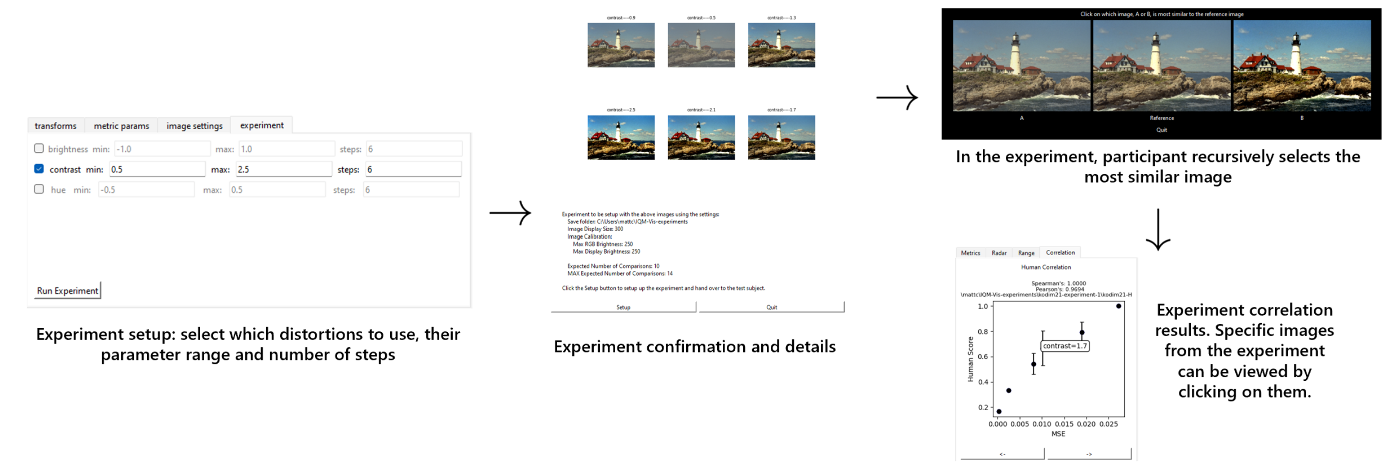

Figure 7: Running a human perception experiment in IQM-Vis.

2.4 Installation and usage

Our software requires python 3.9 or above and has several scientific libraries as dependencies such as numpy, scipy, pytorch. The UI is built upon PyQt6 for cross-platform compatibility. A full list of dependencies can be found in the requirements.txt file of the package.

IQM-Vis can be downloaded and installed directly from the PyPi repository for ease of use. We recommend installing IQM-Vis in a virtual environment which can be easily create with conda or the venv python package. Detailed instructions of how to install IQM-Vis are found in the getting started page of the documentation.

2.5 Customisation

IQM-Vis is easily customisable through a python interface. Custom IQMs, distortions and parameters follow a simple API. All of the UI, graph generation, data storage and optimisation caching is handled by IQM-Vis. An example of simple IQM functions or image distortions is shown in Figure 8 and is explained in detail in our tutorial.

```python
import numpy as np
def custom_MAE_function(im_ref, im_comp, **kwargs):
    L1 = np.abs(im_ref - im_comp)
    return L1.mean()
```

Custom IQM

```python
def custom_brightness(image, value=0):
    return np.clip(image + value, 0, 1)
```

Custom Distortion

Figure 8: Code format needed to use with IQM-Vis. Left: defining a simple IQM. Right: defining a simple distortion.

2.6 Support material

IQM-Vis is implemented in Python, the graphical user interface (GUI) is developed using PyQt6, the image handling and processing is covered by OpenCV [2]. Numpy [5] is used for image distortions and other backend tasks and many of the IQMs provided take advantage of the PyTorch [12] framework to enable GPU hardware acceleration. Other software used is: pandas [18] for database management, matplotlib [9] for plotting graphs, SciPy [16], scikit-image [15].

To accompany the software, we have prepared online documentation, videos and tutorials that provide step-by-step instructions on installing, running and customising IQM-Vis.

3. Software Impacts

Both quantitative and qualitative analysis of IQMs are essential to understanding their behaviour and suitability for a given use case. To the best of our knowledge, IQM-Vis is the only dedicated application and toolbox for this exact purpose.

IQM-Vis helps to lower the boundaries of research on human perception by removing the overhead of explicitly programming 2AFC experiments which requires UI, algorithmic and data management expertise. This opens the doors to researchers with a non computer science background to conduct experiments testing metrics with different properties from vision science. For example, IQM-Vis allows experimental psychologists to quickly and easily design simple tests for IQMs in order to test a hypothesis, for example invariance to rotations.

One current field of research that can benefit greatly from IQM-Vis is work looking into model invariances or sensitivities [3, 1]. For example, a user can conduct an experiment to see how sensitive humans are to affine transforms, and see if the distance in the features of a model designed to be invariant are actually invariant to the transformations. This enables researchers to target the design IQMs with greater effect as well as enabling researchers to verify properties of IQMs as described in the literature.

4. Conclusion

The indented purpose of IQM-Vis is to facilitate the analysis of IQMs to researchers which do not have the time or expertise to develop the graphical software, data storage and algorithmic knowledge required for thorough investigations. The software package includes many standard distortions and IQMs, whilst still being easily customisable to a researcher's requirements.

Both quantitative and qualitative analysis can be simultaneously undertaken when using the graphical software. This is necessary when analysing images for human consumption and algorithms which aim to mimic aspects of human perception. Inbuilt functionality to conduct 2AFC human perception experiments provides researchers with the tools to create custom datasets and explore IQM performance in areas of image distributions beyond what is currently available within the literature.

IQM-Vis is an open source toolbox and aims to continually add to its list of included distortions and IQMs. User experience has been improved throughout the development of the toolbox with feedback from users. This is something that will continue into future versions as the user base's demands and needs evolve.

Acknowledgements
This work is supported by the UKRI Centre for Doctoral Training in Interactive AI EP/S022937/1, UKRI Turing AI Fellowship EP/V024817/1.

References

[1] Gregory Benton, Marc Finzi, Pavel Izmailov, and Andrew G Wilson. Learning invariances in neural networks from training data. *Advances in neural information processing systems*, 33:17605–17616, 2020.

[2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[3] Valentin Delchevalerie, Adrien Bibal, Benoît Frénay, and Alexandre Mayer. Achieving rotational invariance with bessel-convolutional neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 28772–28783. Curran Associates, Inc., 2021.

[4] Keyan Ding, Kede Ma, Shiqi Wang, and Eero P Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE transactions on pattern analysis and machine intelligence*, 44(5):2567–2581, 2020.

[5] Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

[6] Alexander Hepburn, Valero Laparra, Jesús Malo, Ryan McConville, and Raul Santos-Rodriguez. Perceptnet: A human visual system inspired neural network for estimating perceptual distance. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 121–125, 2020.

[7] Alexander Hepburn, Valero Laparra, Ryan McConville, and Raul Santos-Rodriguez. Enforcing perceptual consistency on generative adversarial networks by using the normalised laplacian pyramid distance. In *Proceedings of the Northern Lights Deep Learning Workshop*, volume 1, pages 6–6, 2020.

[8] Alexander Hepburn, Valero Laparra, Raul Santos-Rodriguez, Johannes Ballé, and Jesus Malo. On the relation between statistical learning and perceptual distances. In *International Conference on Learning Representations*, 2022.

[9] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[10] Valero Laparra, Alexander Berardino, Johannes Ballé, and Eero P Simoncelli. Perceptually optimized image rendering. *JOSA A*, 34(9):1511–1525, 2017.

[11] Hanhe Lin, Vlad Hosu, and Dietmar Saupe. Kadid-10k: A large-scale artificially distorted iqa database. In *2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–3. IEEE, 2019.

[12] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[13] Nikolay Ponomarenko, Lina Jin, Oleg Ieremeiev, Vladimir Lukin, Karen Egiazarian, Jaakko Astola, Benoit Vozel, Kacem Chehdi, Marco Carli, Federica Battisti, et al. Image database tid2013: Peculiarities, results and perspectives. *Signal processing: Image communication*, 30:57–77, 2015.

[14] Nikolay Ponomarenko, Vladimir Lukin, Alexander Zelensky, Karen Egiazarian, Marco Carli, and Federica Battisti. Tid2008-a database for evaluation of full-reference visual quality assessment metrics. *Advances of modern radioelectronics*, 10(4):30–45, 2009.

[15] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014.

[16] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[17] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[18] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[19] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.