Journal of Communication and Computer 13 (2016) 351-365
doi:10.17265/1548-7709/2016.07.004

Saarbrucken Synthetic Image Database - An Image Database for Design and Evaluation of Visual Quality Metrics in Synthetic Scenarios

Christopher Haccius and Thorsten Herfet
Telecommunications Lab, Saarland University, Saarbrucken 66123, Germany

Abstract: This paper presents a new image database, SSID, which provides images for evaluation and design of visual quality assessment metrics. It currently contains 1,688 images, 8 reference images, 7 types of distortions per reference image and 30 applications of the distortion types with varying parameters. The distortion types address image errors arising in visual compositions of real and synthetic content, thus provide a basis for visual quality assessment metrics targeting augmented realities and other scenarios where synthetic objects are added to existing scenes. In over 17,000 subjective experiments Mean Opinion Scores for the database have been obtained. These MOS can assist the evaluation of existing and design of novel image quality metrics for scenarios including synthetic content. The evaluation of several existing and widely used quality metrics on the SSID database is included in this paper. The database is made freely available, reproducible and extendable for further scientific research.

Key words: Synthetic image database, subjective quality, mean opinion scores.

1. Introduction

Assessment of image quality is essential in several areas of image processing and coding. Whenever visual media is processed impairments can reduce the perceived quality of this information. Processing steps include all steps from the acquisition to the reproduction of the visual information. The perceived quality can be impaired by camera settings at acquisition time already (e.g., blur due to wrong focal settings, dark noise due to wrong exposure settings), due to further image processing steps (e.g., intensity cropping after brightness changes, artistic filters and modifications), and often images are coded lossy for storage which introduces further artifacts.

As quality degradation depends largely on human perception it is important to measure and quantify quality degradations. The best available metric for perceived image quality is derived from human beings directly. Subjective tests can gather quality opinions human observers have about image degradations, and from multiple such experiments MOS (mean opinion scores) can be calculated. While subjective tests lead to the best results, they are both time and resource expensive. It is therefore desirable to design algorithms which assess the quality of visual data conforming to the human perception.

Existing quality metrics (like SSIM [1] or HDR-VDP [2]) already offer decent solutions to approximate a subjectively generated MOS algorithmically for classical image errors, like random noise, illumination change or blocking artifacts. Novel scenarios leading to visual output, however, create novel sources of errors. Today, an increasing number of visual content are combined from real and synthetic sources or purely synthesized.

Images generated from synthetic content pose a new challenge to image quality metrics, as perceived image qualities do often correspond only very little to the image statistics. If, exemplary, an object is synthetically shifted slightly in a scene, a human observer might not notice the change at all. A metric

Corresponding author: Thorsten Herfet, Prof. Dr.-Ing., research fields: telecommunications and visual computing.

based on image statistics however will detect wrong image values everywhere where either original or new object is placed, and assign a large error, thus a low quality to the evaluated image.

In order to design and evaluate image quality metrics prepared for novel image contents and suitable for purely synthetic and augmented reality scenarios we propose this synthetic image database, which includes a new source of image distortions: errors that occur during the scene composition, before rendering.

This paper extends and fully details the synthetic image database which was introduced to the scientific community in Ref. [3].

2. Related Work

The first widely used image database with image distortions was the LIVE Image Quality Assessment Database developed by Sheikh et al. in 2004, with a second release published in 2005 [4]. The LIVE Database features a variety of photos distorted by compression artifacts (JPEG2000 and JPEG compression with different quality levels), white noise of varying standard deviations, Gaussian blur with kernels of varying size and artifacts created by a fast fading Rayleigh channel for data transmission.

In 2008, Ponomarenko et al. [5] created the TID (Tampere image database) which was updated in 2013, now including 3,000 distorted images created from 25 reference images with 24 different distortion types. The 24 different distortion types include different kinds of additive noise, quantization-, compression- and transmission errors, blurs, intensity shifts, contrast and saturation changes. TID and LIVE database use the same set of images from the Kodak Photo CD [6] as their reference images, but TID exceeds LIVE with respect to the number of distortions and subjective assessments.

With the growing demand for image quality assessments of synthetic image contents the ESPL Synthetic Image Database was created by Kundu et al. in 2014 and updated in 2015 [7]. The ESPL Database covers image distortions comparable to the distortions introduced in LIVE and TID (High Frequency Noise, Interpolation-, Banding- and Ringing-Artifacts, Gaussian Blur and JPEG compression artifacts). Other than LIVE and TID the ESPL database uses synthetic images and not photos as reference images.

Our proposed database contains images distorted with classical error sources, to include elements comparable to LIVE, TID and ESPL. In addition, we include image distortions that are due to faulty scene compositions before rendering. This requires that our reference images are rendered synthetically, thus we only include synthetic images, as ESPL.

Table 1 gives a direct comparison of the main characteristics between LIVE, TID, ESPL and our proposed SSID. Here the number of error assessments is the number of assessments of the same image distortion on one reference image. While our database has less per-image evaluations than the other databases, due to the finer granularity of distortion levels we have more user assessments than LIVE and ESPL for the analysis of the effect of a certain error type on a given reference image.

3. Proposal for a Novel Image Database

Novel scene compositions cause new kinds of image

Table 1 Comparison of existing and our proposed image databases.

	LIVE	TID	ESPL	SSID
# of reference images	29	25	25	8
# of distortions	5	24	5	7
# of test images	1000	3000	525	1680
# of assessments	30.000	250.000	25.000	17.000
# of assessments per image	20-30	350	50	10
# of error assessments	100	1.000	200	300

errors to appear. Core to these new image distortions are errors created by misplaced or misaligned synthetic objects in 3D scenes. This motivates the use of fully synthetic scenes as reference scenes. Fully synthetic scenes come with the benefit of a complete ground truth scene description in three spatial dimensions, allowing the modification of individual objects, which is a necessary requirement for the creation of synthetic image errors. In addition to that synthetic scenes can be used to generate further data like depth maps. Here we focus on image errors caused by object transformations, which are translation, rotation and scaling in 3D space. We therefore propose the use of eight synthetic scenes (shown in Fig. 1). All of these scenes are publicly available and may be modified and redistributed. Central elements of the scenes, e.g. the car, bowling ball or alarm clock, can be modified by affine 3D transformations to simulate possible scene composition errors.

The existing databases, LIVE, TID, and ESPL, have some image distortions in common. These are JPEG compression artifacts, blur and Gaussian noise. For comparison reasons among the existing and novel databases it is advisable to include these distortions into novel databases as well.

We therefore propose to deteriorate the reference scenes by seven different error sources, which are JPEG and JPEG2000 compression artifacts, blurring, Gaussian noise, object translation, object rotation and object scaling. All images are rendered at the same size of 1,920 × 1,080 pixels, thus representing a realistic rendering resolution for many currently used applications.

Each of the image distortions is defined by a set of parameters (as explained in the following sections). For each scene and each distortion we apply 30 different parameter choices, which are normally distributed with mean μ chosen such that a parameter equal to μ results in no error. Exemplary, for translation $\mu = 0$ results in the reference image, but for scaling $\mu = 1$ is the parameter leading to a duplicate of the reference image.

The whole dataset then consists of 8 reference images + (8 scenes × 7 errors × 30 parameters) = 1,688 images. For each error image the parameter choices are recorded in an “info”-file, which is available both in text and in Matlab format.

For image manipulation and the implementation of image errors Matlab [8] provides a suitable environment. For the novel image errors which require rendering of 3D content the open-source 3D computer graphics software Blender [9] was used, and we employed Python [10] to automatize the content manipulation and rendering steps. The following sections describe the different error sources, their mathematical background as well as their implementation.

3.1 JPEG Compression Artifacts

JPEG compression is described extensively in the literature. A thorough description is, for example,

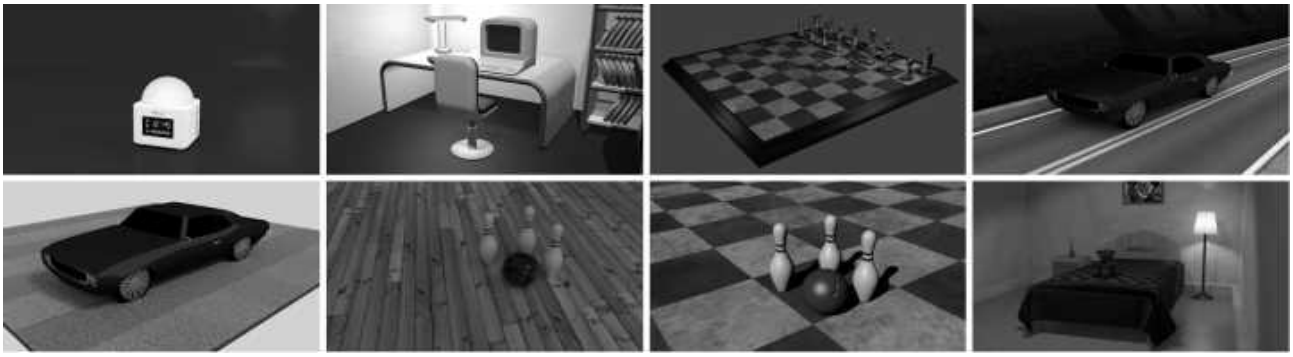

Fig. 1 Images with synthetic content for generation of synthetic errors.

provided by Wallace [11]. The JPEG algorithm for lossy image compression operates on image blocks of 8 × 8 pixels. In the context of compression artifacts the quantization of the 64 frequency components calculated by a Discrete Cosine Transform is the major contributor. This quantization is influenced by multiplication of the frequency components by a percentage: Multiplication by q = 100% does not influence the quantized values, but multiplication with a low value causes many of the frequency components to be rounded to zero after quantization [12]. A quantization matrix Q_k,l with $0 < k, l < 8$ is modified by quality parameter q as follows:

$$Q'_{k,l} = \begin{cases} \dfrac{\left(Q_{k,l} \cdot \left(\frac{5000}{Q_S}\right) + 50\right)}{100} & for\ q < 50 \\ \dfrac{\left(Q_{k,l} \cdot (200 - 2 \cdot Q_S) + 50\right)}{100} & for\ q \geq 50 \end{cases}$$

JPEG coding artifacts can be produced by running the JPEG encoding process and observation of the results. MATLAB offers a function to write images in many desired file formats, among them JPEG. When writing to a JPEG file MATLAB can take the image quality q as an additional parameter. Listing 1.1 gives the source code necessary to create the desired image distortion. An example of such JPEG artefacts for the extreme case of quality q = 0 is given in Fig. 2.

Listing 1.1. Generating JPEG Compression Artifacts

```
q = 5; % quality
imwrite(img,'tmp.jpg','Quality',q);
out = imread('tmp.jpg');
delete('tmp.jpg');
```

3.2 JPEG2000 Compression Artifacts

JPEG2000 is described by ITU-T Recommendation T.809 [13] and —similar to JPEG —discussed in several publications [14-16]. Most significant differences between JPEG and JPEG2000 are the following. JPEG applies a 8 × 8 discrete cosine transform on image macro blocks of size 16 × 16, while JPEG2000 uses a wavelet transform and partitions the image into macro blocks in the wavelet domain, thus reducing blocking artifacts significantly and achieving higher coding gain and scalable coding, main design criteria for JPEG2000 [17]. While compression artifacts in JPEG mostly result from the quantization step (see Section 3.1), in JPEG2000 the bit stream assembler subsequent to domain transformation and quantization is the main source of artifacts.

Similar to artifacts created for JPEG, JPEG2000 coding artifacts can also best be modeled by encoding image data with a JPEG2000 encoder for different bit-rates and reconstructing the encoded image. MATLAB considers target data reduction rates r in its JPEG2000 encoder, as shown in Listing 1.2. Significant reduction rates are necessary to result in visible artifacts. For example, Fig. 3 was reduced with a target compression rate of r = 1000 resulting in a size of only 0.05 *bits/pixel*.

Listing 1.2. Generating JPEG2000 Compression Artifacts

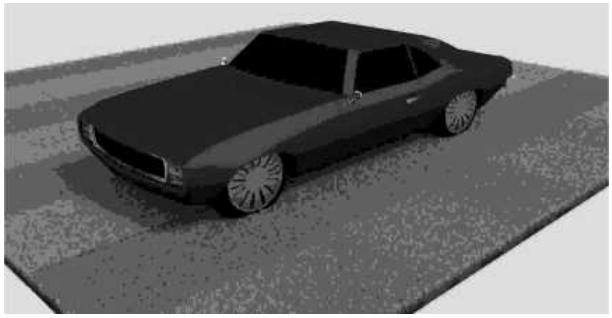

Fig. 2 Blocking artefacts created by JPEG compression of Quality q = 0.

```
r = 500; % compression rate
imwrite(img,'tmp.jp2','CompressionRatio',r);
out = imread('tmp.jp2');
delete('tmp.jp2');
```

3.3 White Gaussian Noise

White noise is noise that occurs uniformly over all frequencies, which means it has a constant power spectral density. Gaussian noise is noise that can be statistically described by a probability density function *p(x)* of a normal distribution

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot exp\left(\frac{(x-\mu)^2}{2\sigma^2}\right) \qquad (2)$$

with mean μ and standard deviation σ. White Gaussian Noise is therefore noise with constant power spectral density and distribution according to Eq. (2).

The *imnoise*-function which MATLAB provides to generate such noise (see Listing 1.3) internally generates a matrix of normally distributed noise **N** and calculates

$$out = in + \sqrt{\sigma^2} \cdot N + \mu \qquad (3)$$

where σ^2 is the variance, which is given as an input parameter to the *imnoise*- function. Fig. 4 illustrates white Gaussian noise, here with a mean of $\mu = 0$ and variance $\sigma^2 = 1$.

Listing 1.3. Modeling White Gaussian Noise

```
m = 0; % mean
v = 0.05; % variance
out = imnoise(img, 'gaussian', m, v);
```

In 1928 the physicists H. Nyquist and J. Johnson published the theoretical background confirming that "thermal agitation of electric charge in conductors" [18] and "thermal agitation of electricity in conductors" [19] can be modeled as white noise, which became known as Johnson-Nyquist noise. As thermal noise is omnipresent this is an important image distortion model.

Fig. 3 Compression artefacts created by JPEG2000 compression of compression rate *r* = 1,000.

3.4 Gaussian Blur

Image blur is an image distortion often caused by objects being out of focus, a too shallow depth of field or either moving camera or moving object during the exposure time. Blurring is achieved by filtering an image with a 2D Gaussian kernel. Extending the 1D Gaussian distribution from Eq. (2) to 2D it is

$$p(x,y) = \frac{1}{\sigma\sqrt{2\pi}} \cdot exp\left(\frac{(x-\mu_x)^2 + \left(y-\mu_y\right)^2}{2\sigma^2}\right) \qquad (4)$$

where [μ_x, μ_y] is the mean (or center) of the 2D Gaussian bell. For filter design a mean offset is

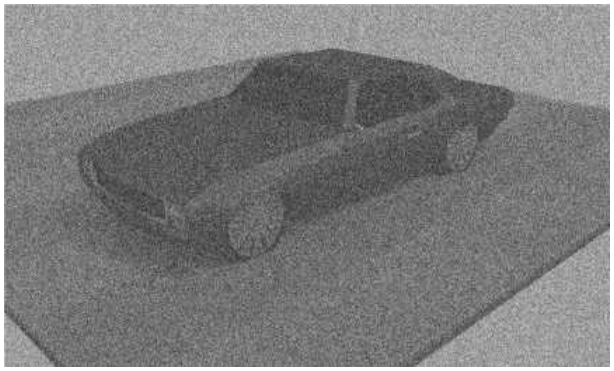

Fig. 4 Average White Gaussian Noise with Mean μ = 0 and Variance σ^2 = 1.

(usually) not desired, therefore $\mu_x = \mu_y = 0$. In MATLAB filters can be generated with the *fspecial*-function, which for the Gaussian kernel only requires the filter size and the standard deviation a of the normal distribution.

A 2D Gaussian blurring kernel of 3 × 3 pixels with $\sigma = 0.5$ is given in Fig. 5. Blurring of an image with a filtering kernel is generated by convolution of the image with the filter.

The MATLAB implementation for generating image distortions using Gaussian blur is given in Listing 1.4. A visualization of this distortion for filter size $s = 30 \times 30$ and standard deviation $\sigma = 10$ is shown in Fig. 6.

Listing 1.4. Modeling Gaussian Blur

```
h = [3 3]; % kernel size
s = 0.5; % standard deviation
filter = fspecial('gaussian', h, s);
out = imfilter(img, filter,'replicate','same');
```

3.5 Object Scaling

Adequate object scaling is necessary to integrate computer generated objects into a real scene. Scaling of objects is achieved by moving the object vertices in 3D according to a scaling factor. This scaling factor can be freely chosen along the object axes, which leads to three independent scaling factors, s_x, s_y and s_z along the *x*-, *y*- and *z*-axis respectively. A vertex position *p* is than scaled to vertex position p' by multiplication with the scaling matrix *S*

$$p' = S \cdot p \tag{5}$$

where *S* is defined by

$$S = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Object scaling implemented in Python to be used in a rendering software is given by the lines of code given in Listing 1.5. The values of *x*, *y* and *z* are the original dimensions of the input object, which are scaled according to a normally distributed random variable with mean $\mu = 1$ and variance $\sigma^2 = 0.33$. As negative scaling factors are not defined, the absolute value of the scaling factors are taken for scaling. With $s_x = 0.9$, $s_y = 0.9$ and $s_z = 0.9$ the car shown in Fig. 1 is scaled to the version shown in Fig. 7.

Listing 1.5. Object Scaling Error

```
x = object.scale[0]
y = object.scale[1]
z = object.scale[2]
# 3 random variables for scaling in x-,y- and z-direction
sx = x * math.fabs(random.gauss(1,0.33))
sy = y * math.fabs(random.gauss(1,0.33))
sz = z * math.fabs(random.gauss(1,0.33))
# scaling with random variables
object.scale=((sx,sy,sz))
# file rendering
bpy.ops.render.render( write_still=True )
```

3.6 Object Translation

The position of an object in a scene is a crucial factor for the realistic appearance of the rendered scene. Deviations from the correct position can have several effects: objects can merge into other scene objects, they can loose contact from surfaces or shift on a surface. Spatial translation in 3D can be expressed by the translation summands t_x, t_y and t_z, which cause translations along the x-, **y-** and z-axis respectively. A

0.011	0.084	0.011
0.084	0.619	0.084
0.011	0.084	0.011

Fig. 5 2D Gaussian Kernel of size 3 × 3 with σ = 0.5.

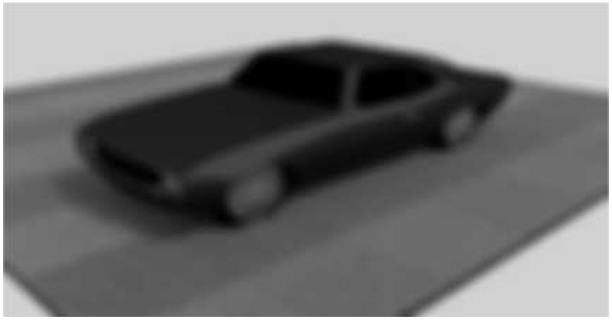

Fig. 6 Gaussian Blur with *s* = 30 × 30 filter size and standard deviation σ = 10.

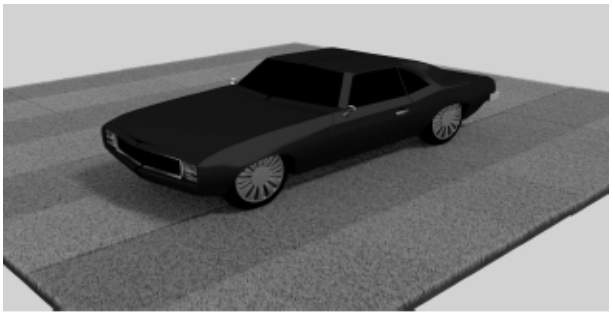

Fig. 7 Car scaled with S_x = 0.9, S_y = 0.9 and S_z = 0.9.

vertex position p is therefore shifted to position p' by multiplication with the translation matrix T:

$$p' = T \cdot p \tag{7}$$

where T is defined as

$$T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8}$$

The lines of code implementing object translation in Python to generate translated objects in Blender are given in Listing 1.6. The translations summands are determined in relation to the original object position. A translation summand of 0 implies no change in position; positive and egative summands cause an object translation in the corresponding direction according to the coordinate axes. Fig. 8 shows that has been used for the previous example as well, this time translated by the summands t_x = 0.0, t_y = -0.1 and t_z = 0.1.

Listing 1.6. Object Translation Error

```
# b as vector of object dimensions
b=bpy.data.objects["OBJECT"].dimensions
# x,y,z as dimensions in x-,y- and z- direction
x=b[0]
y=b[1]
z=b[2]
# 3 values for absolute translation
tx = x * random.gauss(0,0.33)
ty = y * random.gauss(0,0.33)
tz = z * random.gauss(0,0.33)
# translation in x-,y- and z-direction
bpy.ops.transform.translate(value=(tx, ty, tz))
# file rendering
bpy.ops.render.render( write_still=True )
```

3.7 Object Rotation

Most objects are not rotationally invariant. For these objects it is crucial to not only determine their position and scale, but also their alignment with the environment. Alignment is possible with respect to the three coordinate axes. Different from scaling and translation the rotation needs to be defined per axis in a single matrix. For rotational angles a_x, a_y and a_z around *x*-, *y*- and *z*-axis respectively the rotation matrices R_x, R_y and R_z are defined as

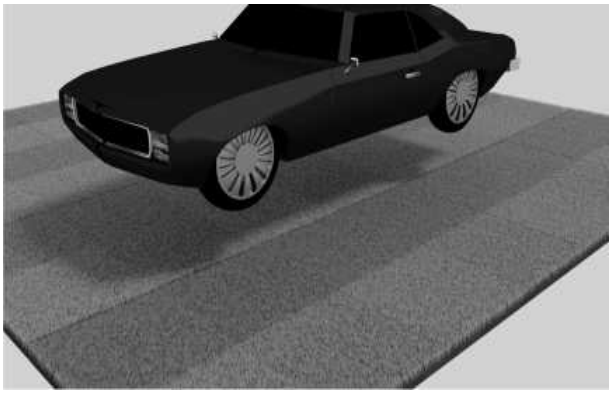

Fig. 8 Car translated by t_x = 0.0, t_y = -0.1 and t_z =0.1.

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha_x) & -sin(\alpha_x) & 0 \\ 0 & sin(\alpha_x) & cos(\alpha_x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

$$R_y = \begin{bmatrix} cos(\alpha_y) & 0 & -sin(\alpha_y) & 0 \\ 0 & 1 & 0 & 0 \\ sin(\alpha_y) & 0 & cos(\alpha_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (10)$$

and

$$R_z = \begin{bmatrix} cos(\alpha_z) & -sin(\alpha_z) & 0 & 0 \\ sin(\alpha_z) & cos(\alpha_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (11)$$

These rotation matrices can me multiplied to a vector p defining an object vertex to rotate this vertex to a new position defined by vector p':

$$p' = R_x \cdot R_y \cdot R_z \cdot p \quad (12)$$

It is important to note that the transformations are not commutative. The order of applying shift, scaling and rotations are important, even the order of rotations around different axis result in differently aligned results. The Python code shown in Listing 1.7 which was employed for the creation of the data base with synthetic errors executes first the rotation around the x-axis, afterwards rotation around the y-axis and finally rotation around the z-axis. The sample image shown in Fig. 9 shows the car rotated by a_x = 0.1, a_y = 0.2 and a_z = -0.1, where all angle measures are given in radians.

Listing 1.7. Object Rotation Error

```
# rotate over x-axis
bpy.ops.transform.rotate(value = random.gauss(0,60), axis
    = (1, 0, 0))
# rotate over y-axis
bpy.ops.transform.rotate(value = random.gauss(0,60), axis
    = (0, 1, 0))
# rotate over z-axis
bpy.ops.transform.rotate(value = random.gauss(0,60), axis
    = (0, 0, 1))
# file rendering
bpy.ops.render.render( write_still=True )
```

4. Image Evaluation

Distorted images can be evaluated subjectively and by machines. Subjective evaluations are time consuming, but important for the design and verification of automatic evaluation algorithms. In the following Section we introduce our experimental setup to obtain subjective evaluation scores. In the succeeding Section we introduce image quality Metrics, which we compare our subjective scores to.

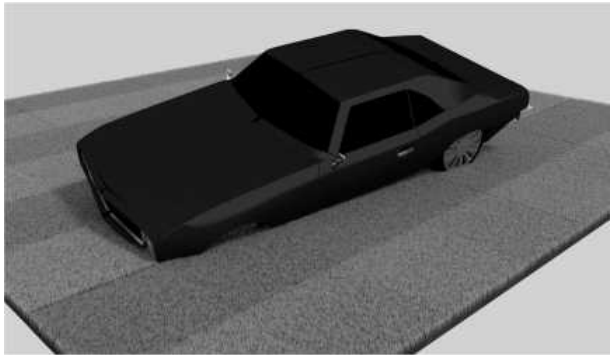

Fig. 9 Car rotated with a_x =0.1, a_y = 0.2 and a_z = -0.1.

4.1 Subjective Image Evaluation

In order to develop and test image quality metrics on our proposed database, subjective quality scores need to be assigned to the error images. A large group of subjects has evaluated the images contained in our proposed database, and of the evaluations Mean Opinion Scores have been calculated. For a given number of N information assessors with their individual opinion$o_i$for $1 < i < N$ the MOS is calculated as

$$MOS = \frac{1}{N}\sum_{i=1}^{N} o_i \qquad (13)$$

Different ways to obtain assessor scores for information have been used and researched. Mantiuk et al. mention four major methods and compare their effectiveness [20]. These four methods are the so called “Single Stimulus”, “Double Stimulus”, “Forced Choice”, and “Similarity Judgments”. Additionally, in 2002 Keelan introduced the “Quality Ruler” method with the goal to overcome some of the negative effects observed in single stimulus methods. All five methods differ with respect to the required observations, the effort of the experiment and the quality of their results. A common part of all subjective studies however is the quality score. Different implementations have been tested from 5 to 100 quality levels of which the assessors where able to choose. A second common attribute of all studies is the timing of the individual stimuli. Especially with media that has no inherent time (like images), this information can be exposed to the observer for any amount of time, which might again lead to different quality opinions.

According to Section 2.7 of Recommendation ITU-R BT.500-11 a test session should not last more than half an hour to prevent fatigue effects. Additionally, each session should start with detailed experimental instructions and training sequence, followed by a break in which sufficient time for questions concerning the experiment is given. Afterward a series of experiments are run with the purpose of stabilizing the experimental outcomes. This stabilizing sequence is not used evaluated. Subsequent to the stabilizing sequence the main experiment starts, of which opinion scores are recorded and further processed [21]. The general structure of a test session is given in Fig. 10.

Due to the number of test images and based on the ITU recommendations as well as the scenario comparison conducted by Mantiuk et al. we designed a single stimulus, hidden reference test which was made available over the internet. This internet-based test on the hand allowed accessing many users all over the world, and second enabled usage on a range of display devices from smart-phones to TV screens or projectors, thus covering a wide range of usage scenarios.

The quality scores in our experiment are chosen on an 11-level scale. Rouse et al. have analyzed the scores given by assessors on a quasi-continuous scale (100 quality levels) [22]. The histogram of the recorded scores is given in Fig. 11. With over 23% of the assessors directly using one of the five MOS categories and over 42% evaluating stimuli quality by

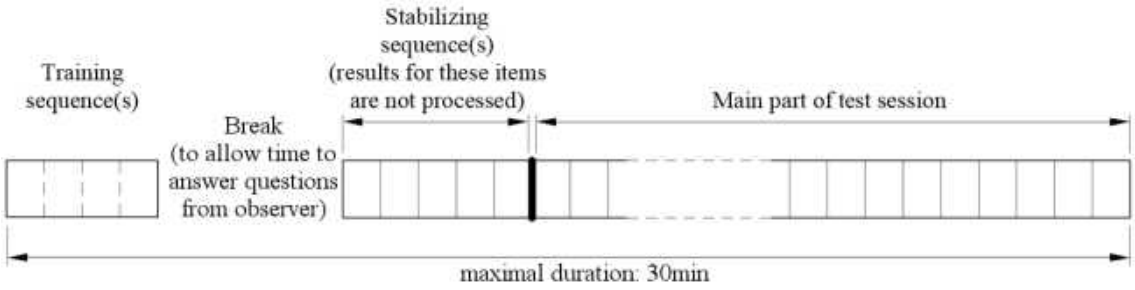

Fig. 10 Structure of a test session for subjective quality assessments [21].

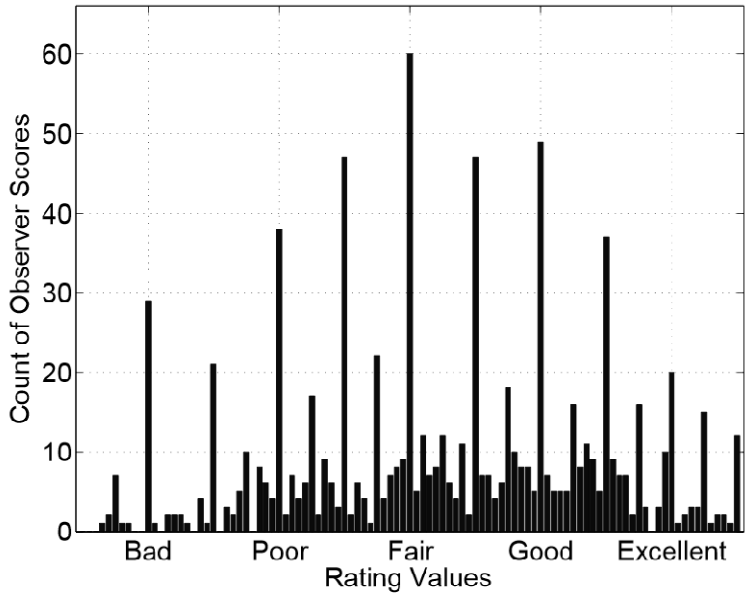

Fig. 11 Histogram of raw assessor scores on quasi-continuous scale [22].

either the MOS categories or their midpoints, it becomes obvious that a continuous scale for quality assessment is superfluous. According to the analysis of Rouse et al. a quality scale with 9 or 11 quality levels — depending on the experimental conditions — suffices fully.

Since synthetic renderings often lack the realism of photos, before the start of the experiment the reference images were presented to the assessors. In order to prohibit a direct relation between reference images and test images, user information (age, gender, kind of device used) was queried after the reference images were shown. In a second stage viewers went through a 2 min testing phase. This testing phase serves two purposes: first, assessors familiarize themselves with the task of image evaluations. Second, the range of image distortions (best and worst cases) are shown to the assessor. This prevents cases where a subject gives a low score to an image, but later observes an even worse image for which he would like to give an even lower score.

The structure of the experiment is outlined in Fig. 12. First, an introductory text explains the purpose, setup and the duration of the experiment. Second, the reference images are presented to the observer for 5 seconds each. Afterward some user data, including display size and viewing distance, are requested as user input. Additional screen information is queried in the background from the browser. We than present an instruction for the test phase to the assessor. This instruction is followed by 2 minutes of iteratively shown test images (3 seconds each) and evaluation scales. The 2 minute test phase is followed by a 10 minute evaluation phase. This phase is again introduced by an instruction, followed by 3 second test images iteratively with evaluation scales. The total experimental time therefore remains at roughly 15 minutes, which stays well inside the attention span of 30 minutes recommended by ITU-R BT.500-11.

For the MOS achieved in this experiment we calculated the least-square fit to an exponential curve as the ideal MOS based on the error parameter. Fig. 13 shows the different fitted curves for the Mean Opinion Scores collected experimentally. It shows a clear correlation between the error parameter and the ideal opinion scores. Note that for JPEG the quality becomes better, the higher the error parameter (q = 100 is best quality, q = 0 worst quality) which leads to an inverted curve compared to the other error kinds, where a larger error parameter directly corresponds to a larger error.

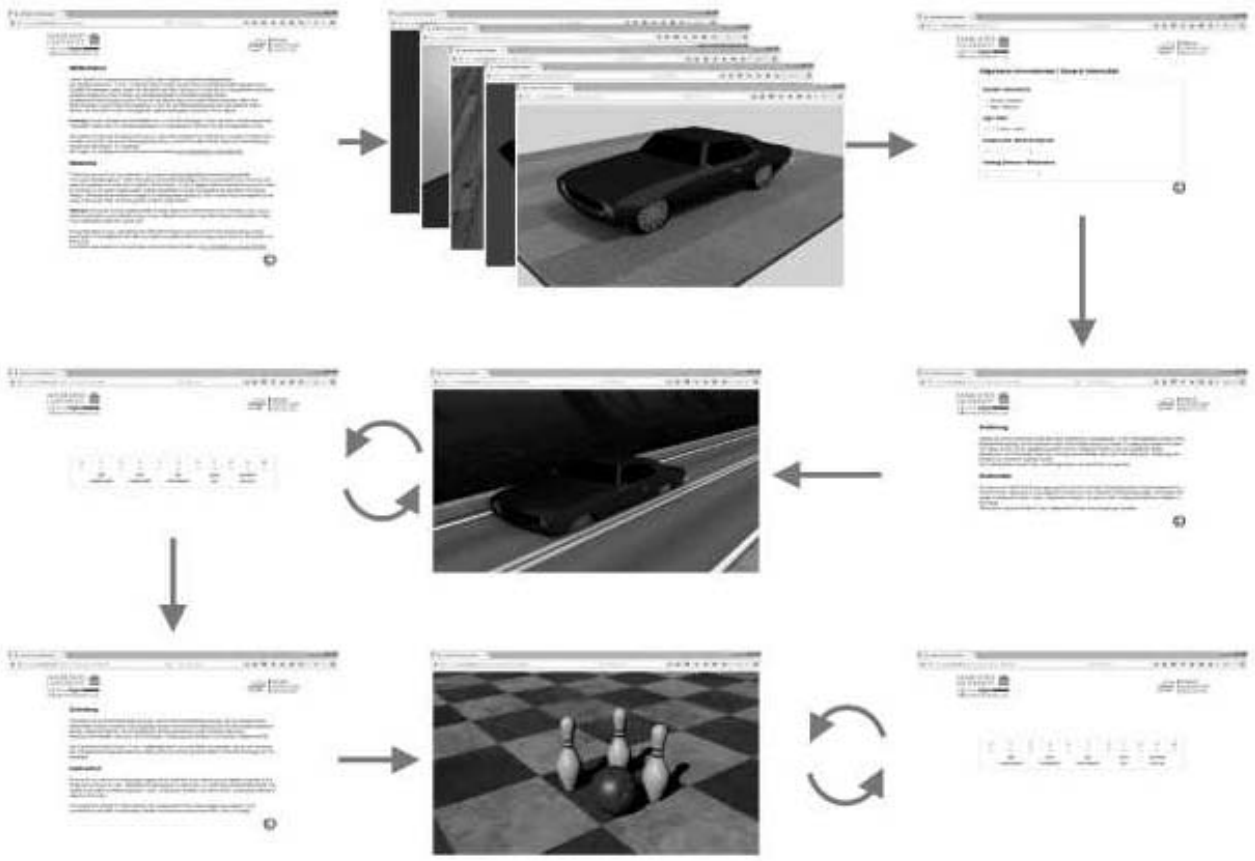

Fig. 12 Outline of experiment to gather assessor opinions.

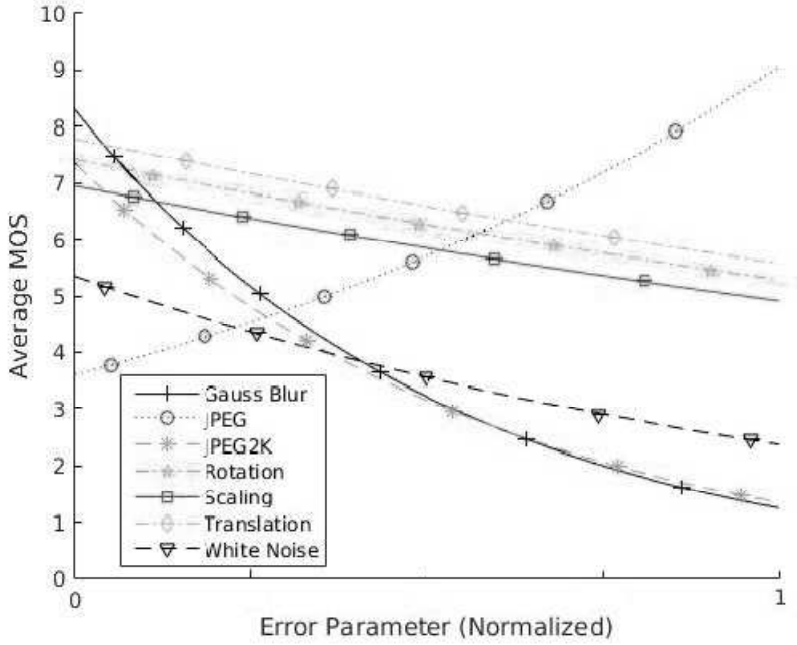

Fig. 13 Ideal MOS for different Error Types.

4.2 Full-Reference Metrics

Full reference metrics currently still achieve the best results for automatic image quality evaluations. A common scenario is the encoding of image and video information. As the encoder receives an undistorted input image, after the encoding process input and output can be compared. Therefore, full reference metrics offer a valuable contribution to the quality analysis of encoded information.

We have computed image quality scores for our image database using three very different but widely used image quality metrics. PSNR (peak signal to noise ratio), the first full reference image quality metric we employ, is purely based on image statistics. Statistical methods do not consider the human viewer at the end at all: purely deviations of the information content are considered.

More advanced image quality metrics have been designed with the Human Visual System in mind. Two metrics we employ for comparison are the Structural Similarity Index [1] and HDR-VDP-2 [2], which was developed based on the SSIM index.

We compute the PSNR value between reference image r with dimensions $x \times y$ and test image t with the same dimensions as

$$PSNR = \frac{max_{i\in[0,x]}\left(max_{j\in[0,y]}(r(i,j)^2)\right)}{MSE} \quad (14)$$

with the Mean Square Error computed as

$$MSE = \frac{1}{x \cdot y}\sum_{i=0}^{x}\sum_{j=0}^{y}\left(r(i,j) - t(i,j)\right)^2 \quad (15)$$

Considering not only image statistics but also the Human Visual System Wang et al. have introduced an image quality metric based on structural similarity. The core idea is that the human visual system is most sensitive to brightness changes in an image. Brightness changes occurring in all color channels are perceived as image structures. With this observation Wang et al. postulate the —in 2004 novel— philosophy of image degradations corresponding to perceived changes in structural information [1].

The system proposed by Wang et al. compares three different image components: luminance, contrast and structure. Structural similarity between a test and a reference image *SSIM(R, T)* is calculated as the weighted product of luminance *l*, contrast c and structure *s*:

$$SSIM(T,R) = l(T,R)^{\alpha} \cdot c(T,R)^{\beta} \cdot s(T,R)^{\gamma} \quad (16)$$

with $0 < \alpha, ß, \gamma$ [1].

Based on the ideas developed by Wang et al. in their works on structural similarity, Mantiuk et al. have extended this visual model to evaluate image qualities in more complex scenarios. A high dynamic range visible difference predictor (HDR-VDP) was introduced in 2005 [23] and completely overhauled in 2011, forming HDR-VDP 2 [2]. According to Mantiuk et al. the vision model presented in Ref. [2] is applicable to a wide range of viewing conditions, especially luminance changes [2].

5. Evaluation and Conclusion

We have calculated rank correlations between the MOS scores obtained experimentally and our ideal MOS (the least square fit exponential function), PSNR metric, SSIM Index and HDR-VDP2 Metric. Correlations are calculated for different error classes individually and for all distorted images together. Error classes are JPEG, Noise, Transformation, Classical, and All. The JPEG class includes JPEG and JPEG2000 image compression artifacts. The Noise class contains Gaussian white noise and Gaussian blur. Transformation includes rotations, scaling and translations. The Classical error class is a super class of JPEG and Noise errors, and the All-error class is a super class of JPEG, Noise and Transformation errors. Table 2 gives Spearman's *p* for the described correlations, Table 3 gives Kendall's *T* for the same. The outperforming metric with respect to the calculated correlation measure for each error class is marked bold in both tables.

Table 2 Spearman - Correlation between MOS and existing metrics.

	JPEG	Noise	Transformation	Classical	All
Ideal MOS	$p = 0.84$	$p = 0.88$	$p = 0.46$	$p = 0.81$	$p = 0.83$
PSNR	$p = 0.72$	$p = 0.59$	$p = 0.31$	$\boldsymbol{p} = 0.69$	$p = 0.42$
SSIM	$p = 0.69$	$\boldsymbol{p} = 0.64$	$\boldsymbol{p} = 0.36$	$p = 0.67$	$\boldsymbol{p} = 0.60$
HDR-VDP 2	$p = 0.51$	$p = 0.56$	$p = 0.24$	$p = 0.52$	$p = 0.37$

Table 3 Kendall - Correlation between MOS and existing metrics.

	JPEG	Noise	Transformation	Classical	All
Ideal MOS	$T = 0.65$	$T = 0.69$	$T = 0.32$	$T = 0.64$	$T = 0.65$
PSNR	$T = 0.53$	$T = 0.41$	$T = 0.21$	$T = 0.50$	$T = 0.29$
SSIM	$T = 0.50$	$T = 0.47$	$T = 0.25$	$T = 0.48$	$T = 0.41$
HDR-VDP 2	$T = 0.35$	$T = 0.41$	$T = 0.17$	$T = 0.36$	$T = 0.25$

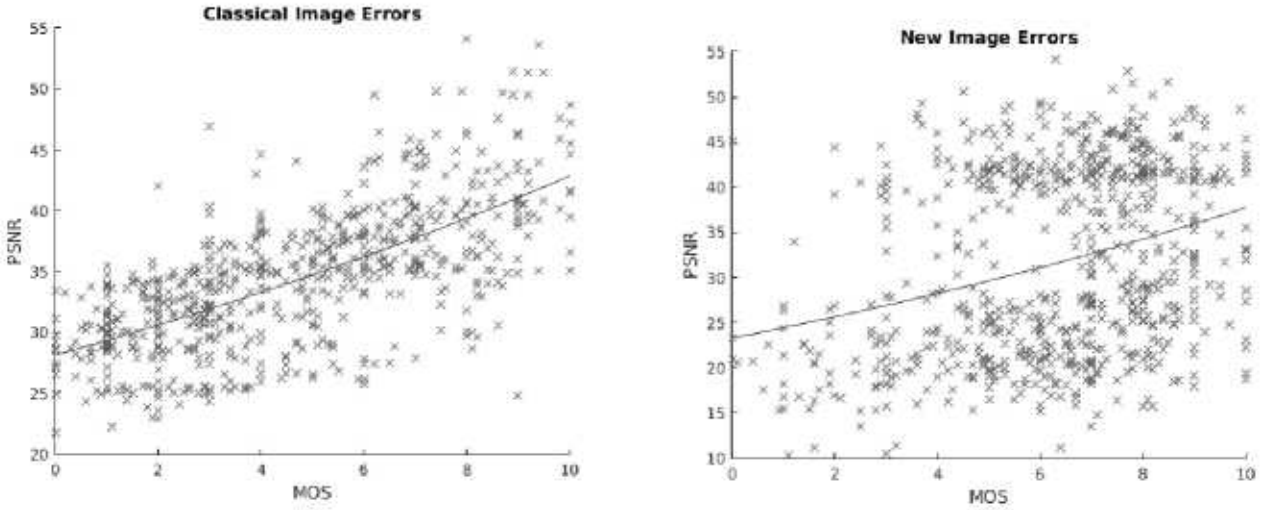

Fig. 14 PSNR vs. MOS for Classical and Novel Image Errors.

Analysis of the correlations allows several conclusions. First of all, the correlation between MOS values and ideal MOS is significant enough to make from the subjective experiments. At the same time, the correlation could probably be improved: TID for example has a Spearman-Correlation of $p = 0.99$ with the Ideal Metric, however at a cost of 250，000 evaluations (we only have 10，000). Second, the correlation between ideal MOS and MOS is smallest for the class of new errors, the transformation errors. A considerable amount of large error parameters seems to lead to a small perceived error, and the other way around small error parameters lead to disturbing results. A logical explanation is, that whenever objects are moved on a surface, scaled uniformly or rotated along a symmetry axes these changes remain unnoticed, even for larger error parameters. However, if objects are shifted through a surface, deformed or rotated around a not rotationally invariant axes, small parameters already lead to significant results.

A third observation is that for chosen images under “simple”（constant lighting conditions) external conditions PSNR and SSIM produce the best prediction results. However, object transformations are predicted worst of all error sources, leading to the lowest correlation between metric and MOS.

6. Access to SSID and Future Work

Our Synthetic Image Database is available for download from Ref. [24]. The database is structured as presented in Fig. 15: for each type of error there exists one folder, containing the distorted images. In

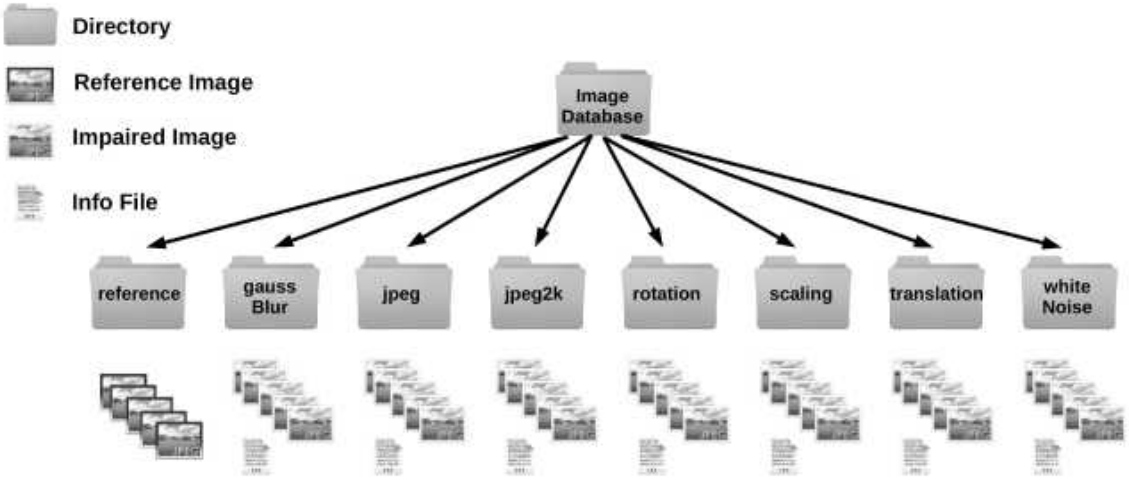

Fig. 15 Structure of our image database.

addition to the distorted images there is one text file and one MatLab file per folder, both containing the same details which are image name, reference image name and error parameter.

The main folder contains an additional folder for reference images. Each reference image is provided with the Blender source and settings to render the exact reference image. In addition, we provide the Python and MatLab source- code to generate the distorted images. For all distorted images MOS and DMOS, obtained from the subjective evaluations, are provided in a MatLab file.

Further subjective evaluations will be obtained. We intend to update the database regularly with MOS and DMOS values based on even more subjective tests. Furthermore, detailed descriptions of how to generate sample scenarios and how to generate distorted images are made public as well. We will extend the database with further images and further scenes that present interesting research scenarios not only for us but for the community.

References

[1] Wang, Z., Bovik, A. C., Sheikh, H. R., and Eero, P. S. 2004. “Image Quality Assessment: From Error Visibility to Structural Similarity.” *IEEE Transactions on Image Processing* 13 (4): 600-12.

[2] Rafal, K. M., Kim, K. J., Rempel, A. G., and Wolfgang, H. 2011. “Hdr- vdp-2: A Calibrated Visual Metric for Visibility and Quality Predictions in All Luminance Conditions.” *ACM Transactions on Graphics (TOG)* 30: 40.

[3] Christopher, H., and Thorsten, H. 2016. “SSID — A Synthetic Image Database.” In *Proceedings of the 2016 International Conference on Image Analysis and Recognition (ICIAR).*

[4] Hamid, R. S., Muhammad, F. S., and Alan, C. B. 2006. “A Statistical Evaluation of Recent Full Reference Image Quality Assessment Algorithms.” *IEEE Transactions on Image Processing* 15 (11): 3440-51.

[5] Nikolay, P. et al. 2013. “A New Color Image Database TID2013: Innovations and Results.” In *Advanced Concepts for Intelligent Vision Systems*, Springer.

[6] Rich, F. 1999. “Kodak Lossless True Color Image Suite.” http://r0k.us/graphics/ kodak/.

[7] Debarati, K., and Brian, L. E. 2015. “Full-Reference Visual Quality Assessment for Synthetic Images: A Subjective Study.” In *Proc. IEEE Int. Conf. on Image Processing*.

[8] MATLAB, 2014. Version 8.4.0 (R2014b), the Math Works Inc., Natick, Massachusetts.

[9] Blender Online Community, 2015. Blender —A 3D Modelling and Rendering Package. Blender Foundation, Blender Institute, Amsterdam.

[10] Guido, van R., and Fred L Drake J. 2014. The Python Language Reference.

[11] Gregory, K. W. 1991. “The Jpeg Still Picture Compression Standard.” *Communications of the ACM* 34 (4): 30-44.

[12] Viraktamath, S. V., and Girish, V. A. 2011. “Impact of Quantization Matrix on the Performance of JPEG.” *International Journal of Future Generation Communication and Networking (IJFGCN)* 4 (3): 107-18.

[13] International Telecommunication Union. 2000. “T.809:

JPEG 2000 Image Coding System." ITU-T RECOMMENDATION, T.

[14] Christopoulos, C., Skodras, A., and Ebrahimi, T. 2000. "The JPEG2000 Still Image Coding System: An Overview." *IEEE Transactions on Consumer Electronics* 46 (4): 1103-27.

[15] Skodras, A., Christopoulos, C., and Ebrahimi, T. 2001. "The JPEG 2000 Still Image Compression Standard." *IEEE Signal Processing Magazine* 18 (5): 36-58.

[16] Rabbani, M., and Joshi, R. 2002. "An Overview of the JPEG 2000 Still Image Compression Standard." *Signal Processing: Image Communication* 17 (1): 3-48.

[17] Li, J. 2003. "Image Compression: The Mathematics of JPEG 2000." *Modern Signal Processing* 46: 185-221.

[18] Nyquist, H. 1928. "Thermal Agitation of Electric Charge in Conductors." *Physical review* 32 (1): 110-3.

[19] Johnson, J. B. 1928. "Thermal Agitation of Electricity in Conductors." *Physical review* 32 (1): 97.

[20] Mantiuk, R. K., Tomaszewska, A., and Radoslaw, M. 2012. "Comparison of Four Subjective Methods for Image Quality Assessment." In *Computer Graphics Forum*, Wiley Online Library.

[21] International Telecommunication Union. 2002. "Bt.500-11, Methodology for the Subjective Assessment of the Quality of Television Pictures." ITU-R RECOMMENDATION, BT.

[22] Rouse, D. M. et al. 2010. "Tradeoffs in Subjective Testing Methods for Image and Video Quality Assessment." In IS&T/SPIE Electronic Imaging. *International Society for Optics and Photonics*: 75270F-75270F.

[23] Mantiuk, R. et al. 2005. "Predicting Visible Differences in High Dynamic Range Images: Model and Its Calibration." In Electronic Imaging. *International Society for Optics and Photonics*: 204-14.

[24] Haccius, C., and Herfet, T. 2016. "SSID —A Synthetic Image Database." http://www.nt.uni-saarland.de/SSID/, Accessed: 2016-09-15.