EG UK Computer Graphics & Visual Computing (2023)
D. Hunter and P. Vangorp (Editors)

An Image-based Model for 3D Shape Quality Measure

Fahd Alhamazani, Paul L. Rosin and Yu-Kun Lai

Cardiff University, UK

Abstract

In light of increased research on 3D shapes and the increased processing capability of GPUs, there has been a significant increase in available 3D applications. In many applications, assessment of perceptual quality of 3D shapes is required. Due to the nature of 3D representation, this quality assessment may take various forms. While it is straightforward to measure geometric distortions directly on the 3D shape geometry, such measures are often inconsistent with human perception of quality. In most cases, human viewers tend to perceive 3D shapes from their 2D renderings. It is therefore plausible to measure shape quality using their 2D renderings. In this paper, we present an image-based quality metric for evaluating 3D shape quality given the original and distorted shapes. To provide a good coverage of 3D geometry from different views, we render each shape from 12 equally spaced views, along with a variety of rendering styles to capture different aspects of visual characteristics. Image-based metrics such as SSIM (Structure Similarity Index Measure) are then used to measure the quality of 3D shapes. Our experiments show that by effectively selecting a suitable combination of rendering styles and building a neural network based model, we achieve significantly better prediction for subjective perceptual quality than existing methods.

CCS Concepts

• ***Computing methodologies*** → *Shape analysis;*

1. Introduction

We exist in a three-dimensional world, making the analysis and processing of 3D shapes essential for many areas. These include fields like design, manufacturing, robotic navigation, as well as virtual and augmented reality (VR/AR). 3D shapes have a wide range of applications, from grasping [dSCR*21,EMF21,YPM*21] to reconstruction [HG22,FYJ*22]. In many applications, measuring the perceptual quality of 3D shapes is required. For example, when 3D objects are manufactured, the produced shapes have unavoidable deviations compared with the original designs, and it is therefore useful to quantify the deviations based on users' subjective perception. Another example is when 3D data is streamed in VR/AR applications, distortions could be introduced due to data compression with limited bandwidth, and measuring the quality of the distorted shapes is not only useful for quality control, but can also help guide how to better allocate the limited bandwidth.

Given a pair of 3D shapes, one original and one distorted, the task we address in this paper is to predict a quality score, that is ideally close to human subjective judgement. Traditional methods tend to directly measure errors on 3D shapes. However, such measures are often inconsistent with human judgement. For example, a scar on a face is geometrically only a small change, but can be perceptually significant. Observing that human eyes essentially perceive 2D views of 3D shapes, whether in the real world or in the virtual settings.

We propose to measure 3D shape quality based on their 2D renderings. To give a sufficient coverage, we start by rendering the shapes to multiple views. We specifically choose centres of dodecahedron faces as camera locations for capturing the shape views and placing a directional light. Perceptual quality of 2D views can also be influenced by rendering styles, as different aspects of 3D shapes would be emphasised with different renderings. For example, even for a relatively small dent on a surface, the local geometric normal may change significantly. Rendering with metal styles can highlight subtle changes on shape areas that cause specular highlights to look different, whereas rim type of rendering is more sensitive to edges (see Figure 3 for some examples of rendering styles.

We therefore propose to use combinations of different rendering styles (different shading and material properties) to better capture the visual quality of 3D shapes. Next, we extract quality measures using a 2D method, such as structural similarity index measure (SSIM) [WBSS04] and mean squared error. To avoid the influence of empty space for image quality measure when rendering shapes, we further propose a modified SSIM that only accounts for the foreground regions (called Mask-SSIM). These features are combined using a neural network based approach to predict the subjective quality measure.

The contributions of this work are as follows:

- We propose an image-based method to measure perceptual quality of 3D shapes. We further combine a variety of rendering styles and 2D image quality measures, along with a neural network based learning approach for improved 3D subjective quality prediction.

- In order to ensure more stable performance when shapes are rendered to different canvas sizes, we extend SSIM to only focus on the foreground region, referred to Mask-SSIM, which is effective for our task.
- Experiments on public datasets demonstrate that our method achieves good prediction for subjective quality scores, outperforming existing techniques.

2. Related Work

For 3D perceptual quality measure, there are two related but different problems. One problem is reference-based, where both the original and distorted shapes are available, and the task is to judge the quality of the distorted shape. The other problem is non-reference-based or blind quality measure where only the distorted shapes are available. These techniques can be useful in practice in different application settings. In this work, we focus on reference-based models.

For reference-based quality measure, Alexiou and Ebrahimi [AE20] suggest using the structural similarity index measure (SSIM) [WBSS04], originally designed for images, to measure geometric similarities, and apply this to compare pairs of point clouds. The original SSIM method uses luminance, contrast and structure, in contrast, pointSSIM uses four features (geometry, colour, curvature and normals). The model is not sensitive to shape resolution [AE18]. The method however relies on calculating a large number of Euclidean distances between two point clouds, making it computationally expensive, and thus this method is not practical for evaluating dense shapes. In [Lav11], Lavoué et al. leverage a statistical approach. It works out the distortion level at each vertex as the difference of Gaussian-weighted local shape statistics. The process is repeated for different scales of the distorted shape. At the end, they calculate the score by using Minkowski pooling. On the other hand, Feng et al. [FWDX*18] weight the distortion map before pooling where severely distorted areas have more weight than other regions. In the work [YMX*20], the authors sample edge points by utilising a high pass filter. After selection, they build a graph to map the structure and calculate the mass, mean and variance of the selected points between shapes to find similarity. Abouelaziz et al. [AOEHC15] also describe a referenced-based approach. They suggest extracting angles between each face and its neighbours and by using angles they build a Weibull-Gamma distribution for each model. Kullback-Leibler (KL) divergence is applied to the pair of distributions for a distorted shape and a reference shape to measure the differences between them. In [ACEHC18], they further apply KL divergence to distributions of deep features extracted from multi-view images rendered from the distorted and reference shapes.

Alternative methods consider blind (no-reference) shape quality measure. These methods are largely based on machine learning. Liu et al. [LYXY20]'s model includes three stages. First, the point cloud is processed using sparse convolution layers. Each layer is treated as a feature block, and the model compresses the features by using pooling operations to calculate a vector. Finally, a model with fully connected layers is used to generate a similarity score. Abouelaziz et al. [AEHC16] handcraft curvature features, and then they utilise a fully connected model to estimate the score. Liu et. al. [LYS*21] also present a deep learning model. In this model, the shape is first projected to multiple 2D views (six images). Then a three convolution layer pipeline is used for each image in parallel. The model then concatenates all the outputs as a vector. This vector is passed on to two different models to measure the image quality and degree of distortion. Some works adapt their previous full-reference approaches to a blind measure. Abouelaziz et al. [AEHC18] utilise Weibull, Gamma and Rayleigh distributions. The model extends previous work by adding support vector regression (SVR) that fits distribution parameters to estimate quality score. In [ACEH*20b], they combine deep features from the multi-view images with handcrafted features such as curvature and angles from the 3D geometry. Then a general regression neural network (GRNN) is trained to predict the score. The model then classifies the shapes by quality. Another approach described in [ACEH*20a] also evaluates mesh representations. They claim that salience determines which patch of a shape should be processed to find distortion. In addition, before processing, the shape is rendered to a fixed view. One patch is then selected based on salience. Finally, a Convolutional Neural Network (CNN) model is trained with these selections to estimate a quality score. The work [ACEH*20c] also uses a salience map. They threshold unwanted patches of projected images through a salience map and then compare the outputs using a pretrained model such as VGG, AlexNet or ResNet. However, this model is not stable due to nature of the selected patches. Although they work out a threshold on saliency to select important patches, there is no guarantee that the selected patches hold useful information. The authors extend their work in [ACEH*18], where they use a vanilla CNN model rather than pretrained model.

Zhang et al. [ZSM*21] develop a model that can evaluate both point clouds and mesh shapes. For a point cloud, K nearest neighbour (KNN) is utilised to find a set of points for each point of concern. For a mesh, curvature could be more sensitive to local distortion. This model also considers colour in the quality assessment. A support vector machine regressor processes the handcrafted features to predict a quality score. The work [ACEH*21] believes that converting a mesh representation to a graph problem may increase the accuracy of the quality score. First, they extract both adjacency matrices and extract handcrafted features such as curvature and saliency. The graph convolution layers are utilised with softmax to classify the shape's level of distortion. Abouelaziz et al. [ACEHC21] also utilise a graph where adjacency matrices are used. However for feature representation, they prefer Gaussian curvatures and mean curvature, and claim curvatures can point to smoothness and roughness of the mesh.

In our work, we focus on reference-based measure, and propose to use various views and rendering styles, along with different 2D image quality measures and a neural network architecture to achieve improved prediction accuracy.

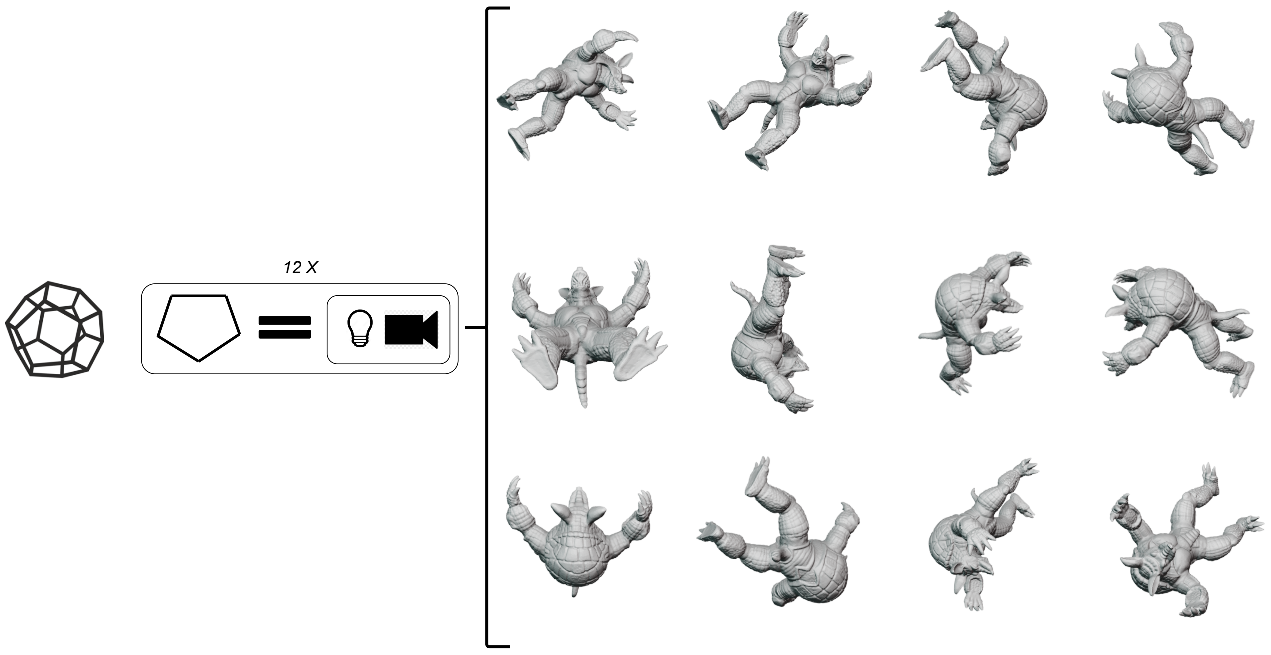

Figure 1: *A list of rendered views of Armadillo using dodecahedron faces, each face consisting of a directional light and a camera both pointing to the centre of the dodecahedron.*

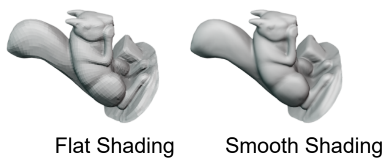

Figure 2: *We use two types of shading (i.e., flat and smooth) to help capture a wider range of features.*

3. Our Method

We first describe how the 3D shapes are rendered using different views, styles and shaders. Next, we will discuss Mask-SSIM, which is a metric that is agnostic to which 3D representation is used. Finally, we list a collection of different 2D measures used as the basis to build the machine learning based model for final quality prediction.

3.1. Rendering Setup

We use centres of the faces of a regular dodecahedron as camera locations for shape rendering. The dodecahedron is one of the five Platonic solids; it ensures that the cameras are equally spaced, providing a good coverage for the entire shape. There are regular polyhedra that have more faces; however for simplicity we chose the dodecahedron. The target shape is placed in the centre of the dodecahedron with all 12 cameras facing it. For simplicity, we normalised (uniform scaling) all shape sizes during the rendering process to fit all shapes inside the dodecahedron. Each camera is paired with a directional light, as shown in Figure 1. Our rendering setup was built in Blender as it offers automation through the Python binding library; however, a similar environment could be built in any appropriate application.

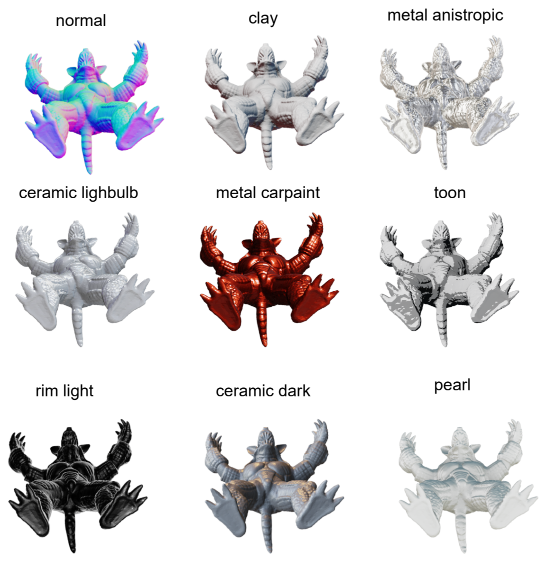

Figure 3: *Some of the styles used in the experiments. As can be seen, different rendering styles tend to highlight different aspects of the shape characteristics.*

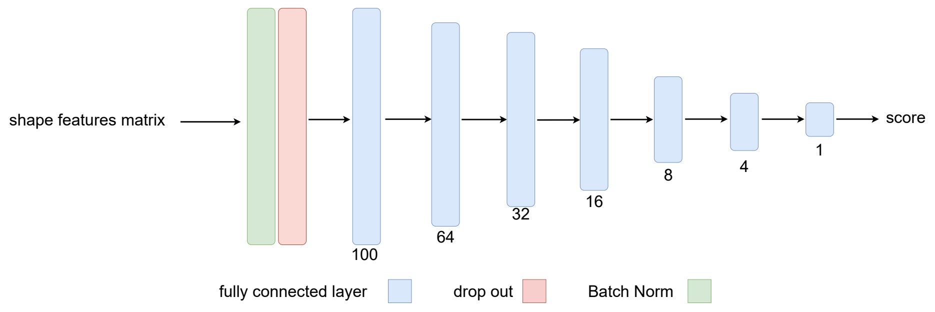

Figure 4: *Our network architecture for learning to predict shape quality score. It consists of batch normalisation and dropout and fully connected layers.*

3.2. Rendering Styles

We utilise two different kinds of shading for each different style. To begin with, a flat shading was utilised; this uses the normal of each triangle to render the triangle, so the triangulation details are clearly visible. In addition to that, we utilise smooth shading, which effectively blurs the boundary between triangular faces so that they appear smooth. See Figure 2 for an example.

We make use of the predefined Blender textures and colours in addition to importing some additional ones, which brings in a total number of 30 styles: flat & clay muddy, smooth & metal lead, basic 1, basic 2, basic dark, basic side, brown, ceramic dark, ceramic lightbulb, normal+y, check rim dark, check rim light, clay brown, metal shiny, orange-blue, pearl, reflection check horizontal, reflection check vertical, resin, skin, toon, clay muddy, clay studio, dark grey, jade, matt blue, matt brown, metal anisotropic, metal car paint, metal lead. We opted for a random selection of styles for the study since a random selection is less likely to introduce any selection bias, which could potentially distort the results of the study. An example of the generated samples can be seen in Figure 3.

The process proceeds as follows: first, we identify the shape and place it within the geometry of the dodecahedron. Second, we decide which kind of shading to use (flat or smooth). We render each style from a total of twelve distinct angles (the geometric faces of a dodecahedron), starting with the shape that serves as a reference and moving on to the distorted versions of that shape. The datasets have been normalised so that all of the shapes are the same size [0-1].

3.3. Mask-SSIM

SSIM can identify the differences between the targets, by measuring luminance, contrast and structure. It is later extended in a multi-scale manner, such as MSSIM and DSSIM. However, the function does not provide consistent results for targets when the same 3D shape is rendered to canvas of different resolutions. This is because background left blank is always consistent with background of another shape, leading to high SSIM scores. Therefore, the similarity scores varies according to image resolution of the rendering canvas.

In our approach, we first separate the target shape from the background. Assume we have two images, A and B. To create the mask, we match each pixel inside image A to the pixel at the same location in image B. If both pixels are inside the boundary of the target shape, a value of 1 is given for the position in the mask; otherwise, the mask is set to 0. Figure 5 shows the illustration of the proposed algorithm.

The operation by definition is pixel-wise. However, since the SSIM calculation is based on windows, we also implement Mask-SSIM Window, which only considers a pixel to be included if the entire neighbourhood window is included. These models (Mask-SSIM and Mask-SSIM window) produce robust results with varying canvas resolutions.

The quantitative comparisons are shown in Table 5. Specifically, Mask-SSIM is implemented on top of SSIM when generating the SSIM map as follows:

$$C = SSIM(A,B) \tag{1}$$

where $SSIM(\cdot)$ returns the SSIM map of A and B, denoted as C. Let A_{mask} and B_{mask} be the foreground masks of images A and B:

$$Mask_{AB} = A_{mask}B_{mask} \tag{2}$$

$Mask_{AB}$ identifies the agreement between the two masks, where 1 means pixels at a position in both images are foreground, and 0 otherwise.

$$MaskSSIM(A,B) = \frac{1}{N}\sum_{i=1}^{n} C_i Mask_{AB,i} \tag{3}$$

$MaskSSIM$ describes how Mask-SSIM works. C_i represents pixels at the ith location. We multiply pixel C_i by $Mask_{ab,i}$ to limit the operation w.r.t. A_i and B_i where $Mask_{AB,i}$ is a binary pixel value $[0,1]$.

n is the number of pixels in the image, and $N = \sum_{i=1}^{n} Mask_{AB,i}$ is the number of selected pixels.

3.4. Selected Features

We select various 2D/3D quality measures and features. For 2D, we use SSIM [WBSS04], FSIM [ZZMZ11], root mean square error (RMSE), Canny edge [Can86], SRE [LBDG*18] and our modified Mask-SSIM or its variants. Different variants of Mask-SSIM are explained in section 4.5.

We also implemented 3D measures, Chamfer distance and Hausdorff distance [HKR93] for comparison.

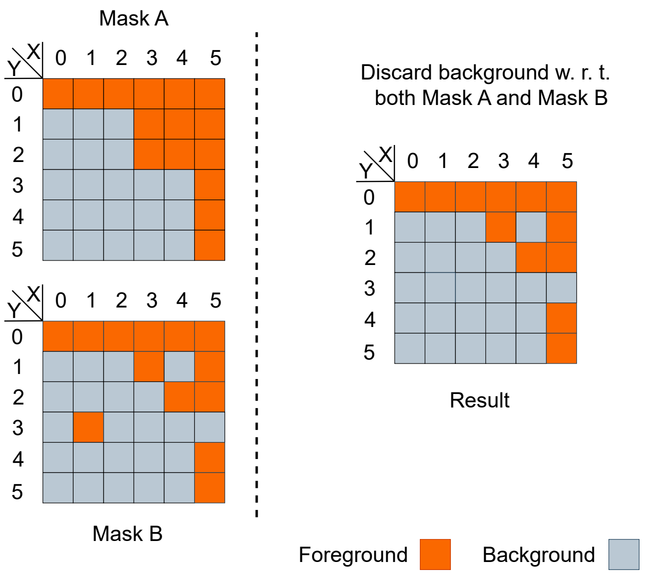

Figure 5: *Illustration of Mask-SSIM operation. We classify background/foreground pixels w.r.t. to both A and B. The left shows both original masks A and B before classification. The right shows the output after classification w.r.t. both A and B. For example, looking to the resulting mask we see pixel at Result[4,1] is classified as background however in Mask A is not, in this situation clearly there is some distortion between the two shapes in the 3D space.*

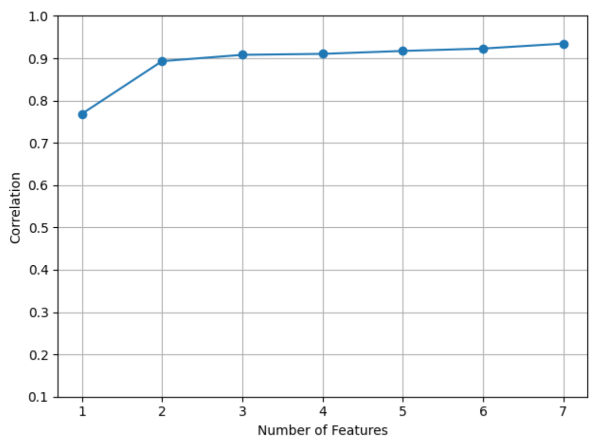

Figure 6: *SFS result on dwarf in Dataset [GVC*16], where it shows using the 7 selected features has the highest performance.*

Table 1: *Selected features using SFS when the model Dwarf (from [GVC*16]) is used as the test shape.*

selected features
Mask SSIM window ceramic lightbulb flat
Mask SSIM metal anisotropic smooth
Mask SSIM matt blue smooth
Mask SSIM brown flat
No mask metal lead smooth
Mask SSIM clay studio smooth
Mask SSIM metal anisotropic flat

3.5. Network Architecture

The learning task is to find a relationship between the results of the individual measures and map them to the final quality prediction, which can be thought of as bridging the gap between evaluation metrics and human judgements. After calculating the values of the Mask-SSIM, we combine these values with the scores from other algorithms to create the input matrix for our neural network model.

The model starts with batch normalization and a dropout layer, followed by fully connected layers. Each layer has a ReLU activation function, except for the final layer which has a sigmoid activation function. We utilise the mean squared error as the loss function (see the model shown in Figure 4).

To compare different quality metrics, since they are often in different ranges, instead of directly comparing values of user voting and prediction, we use Pearson correlation to fit the functions into a {-1,1} domain. Pearson correlation measures the linear correlation between two sets of data, and we use Pearson correlation as the evaluation metric to indicate the correlation between predicted quality scores by a specific method and human rated scores. In addition, the number of features is still high and some may not be helpful in learning the model. So we further utilise Sequential Forward Search (SFS) to select important features. It incrementally adds new features to the selection, and at every stage, a greedy approach is used to choose the feature that leads to be best performance among all choices, based on its performance on the training set. The selected features are then used when applying the model to the unseen test set (see Algorithm 1). Although SFS can often be too expensive to be used in the deep learning setting, our deep network is small and can be trained efficiently, so this strategy is still reasonably efficient, Table 1 shows the selected features for Dataset [GVC*16] on dwarf shape and Figure 6 shows the mapping between selected features and accuracy.

4. Experiments

4.1. Datasets

Three datasets are used in the experiments which include both distorted shapes and subjective scores: [LGD*06], [Lav09] and [GVC*16].

The four shapes in dataset [LGD*06] are {Armadillo, Rockerarm, Venus, Dyno}. The five shapes in dataset [GVC*16] are {Dwarf, Hulk, Squirrel, Statue, Sports Car}. The shapes in dataset

Table 2: *Cross-validation correlation results on dataset [LGD*06]. Our and Our* refer to our results with all features and selected features. Note the setup for MS-SSIM is a flat shader and ceramic lightbulb style, similar to typical rendering styles in previous work.*

	chamfer	Hausdorff	PointSSIM	DCD	MSE	PSNR	MS-SSIM	Our	Our*
armadillo	0.13	0.13	0.20	0.12	0.19	0.17	0.20	0.41	**0.54**
dyno	0.02	0.47	**0.67**	0.16	0.17	0.19	0.34	0.52	0.54
rockerarm	0.08	0.27	0.15	0.23	0.19	0.18	0.17	0.36	**0.62**
venus	0.25	0.13	0.62	0.40	0.21	0.20	0.24	0.36	**0.65**
average	0.12	0.25	0.41	0.23	0.19	0.19	0.23	0.41	**0.58**

Table 3: *Cross-validation correlation results on dataset [GVC*16]. Our and Our* refer to our results with all features and selected features. Note the setup for MS-SSIM is a flat shader and ceramic lightbulb style.*

	chamfer	Hausdorff	PointSSIM	DCD	MSE	PSNR	MS-SSIM	Our	Our*
dwarf	0	0.16	0.62	0.19	0.32	0.31	0.34	0.75	**0.93**
hulk	0.35	0.45	0.65	0.56	0.35	0.35	0.46	0.80	**0.87**
squirrel	0.31	0.22	0.17	0.34	0.26	0.25	0.36	0.43	**0.63**
statue	0.05	0.52	0.92	0.61	0.31	0.29	0.38	0.92	**0.93**
average	0.17	0.33	0.59	0.43	0.31	0.30	0.38	0.72	**0.84**

Algorithm 1 Sequential Forward Search (SFS)

```
Input A list of features: $F = \{f_1, f_2, \cdots, f_d\}$
Output A list of selected features: $Sel_F$
$Sel_F = \{\}$
repeat:
  $f_{\text{next}} = \emptyset$
  $c_{\text{next}} = corr(Sel_F)$;     $corr(\cdot)$ computes the correlation on the
                                        training set when trained with the given set of features
  for  $f = f_1 \ldots f_d$:
    if  $f \notin Sel_F$ then
      $c_f = corr(Sel_F \cup f)$
      if  $c_f > c_{\text{next}}$ then
        $f_{\text{next}} = f$
        $c_{\text{next}} = c_f$
  if  $f_{\text{next}} \neq \emptyset$ then
    $Sel_F = Sel_F \cup f_{\text{next}}$
until $f_{\text{next}} == \emptyset$
```

[Lav09] are {Armadillo, Dyno, Lion, Bimba}. Several types of distortions are used in Dataset [LGD*06]: 1. smoothing with a different number of iterations; 2. simplification (removal of vertices) with different percentages; 3. uniform quantisation using different bit sizes; 4. JPEG texture compression; and 5. sub-sampling to reduce the texture size. As we only examine geometric distortion here, we eliminate textural (2D) distortion. The sports car shape could not be included in our experiments as we identified some issues in loading the geometry (files missing). As a result, each dataset contains 4 usable shapes. Although this may sound quite small, considering the range of distortions, and time consumption for collecting user subjective ratings, collecting such data is onerous, and we are not aware of larger datasets of this kind being available.

A total of 12 distortion types are used in [LGD*06]. The dataset [Lav09] only applies noise to the shape surface with six different levels of noise for each shape. The work [GVC*16] applies two types of distortion with different levels. The first is noise addition, which was done by altering the location of vertices on different levels. The second is Taubin smoothing [Tau95] distortion. A total of 21 distorted shapes were generated for each shape.

4.2. Training

The model was trained for 250 epochs and batch size of three. The learning rate was set to 0.0001. The stochastic gradient descent was used as the optimiser. The model was trained on an NVIDIA GTX 1080ti GPU. The rendering of a shape in a specific style and view takes 22 milliseconds.

4.3. Evaluation

To evaluate our model's results, we use Pearson correlation. We calculate the correlation between the Mean Opinion Score (MOS) and each method because the methods we used for comparison all have either low sensitivity or varying value ranges.

To evaluate our model's performance, we compare it to five methods: (1) the Point SSIM Model [AE20]; this method leverages geometry, vector value and curvature to calculate the similarity; and (2) the Density-Aware Chamfer Distance [TW21] (DCD) Model; this approach is derived from the Chamfer distance and focuses on distribution quality. Two other metrics are traditional and widely used methods (3) MSE (Mean Squared Error), and (4) PSNR (Peak Signal-to-Noise Ratio). Finally, we also compare with an image-based baseline which applies (5) MS-SSIM (multi Scale-SSIM) to 12 rendered views.

4.4. Results

Our model was trained on three different datasets, as stated above. The results of the training for Lavoué. et al.'s dataset [LGD*06] are

Table 4: *Cross-validation correlation results on dataset [Lav09]. Our and Our* refer to our results with all features and selected features. Note the setup for MS-SSIM is a flat shader and ceramic lightbulb style*

	chamfer	Hausdorff	PointSSIM	DCD	MSE	PSNR	MS-SSIM	Our	Our*
Armadillo	0.13	0.13	0.26	0.09	0.12	0.11	0.31	0.40	**0.51**
Dyno	0.02	0.47	0.41	0.19	0.26	0.10	0.42	0.55	**0.63**
Lion	0.19	0.32	0.32	0.21	0.18	0.21	0.47	0.58	**0.64**
Bimba	0.12	0.24	0.22	0.11	0.15	0.17	0.28	0.59	**0.63**
average	0.11	0.29	0.30	0.15	0.17	0.14	0.37	0.53	**0.60**

Table 5: *Comparisons of original SSIM and Mask-SSIM for resolutions of* 500×500 *and* 1000×1000 *canvas sizes. The experiment is based on the dwarf shape with various distortions, rendered using metal anisotropic material and smooth shading.*

Distorted shapes	500 resolution Mask-SSIM	500 resolution original SSIM	1000 resolution Mask-SSIM	1000 resolution original SSIM
dwarf quantization 8 bit	0.29	0.86	0.29	0.97
dwarf quantization 9 bit	0.52	0.90	0.52	0.99
dwarf quantization 10 bit	0.79	0.96	0.79	0.99
dwarf quantization 11 bit	0.92	0.98	0.92	0.99
dwarf Simplification 0.80	0.53	0.90	0.53	0.98
dwarf Simplification 0.92	0.46	0.89	0.46	0.98
dwarf Simplification 0.975	0.27	0.85	0.27	0.97
dwarf Simplification 0.987	0.27	0.85	0.27	0.97
dwarf Smoothing 15 iteration	0.72	0.94	0.72	0.99
dwarf Smoothing 25 iteration	0.64	0.92	0.64	0.99
dwarf Smoothing 40 iteration	0.57	0.91	0.57	0.99
dwarf Smoothing 50 iteration	0.53	0.90	0.53	0.99

Table 6: *An Ablation study on batch-norm layer using the selected features only.*

	without batch-norm	with batch-norm
armadillo	0.42	**0.54**
dyno	0.52	**0.54**
rockerarm	0.56	**0.62**
venus	0.62	**0.65**
average	0.53	**0.58**

shown in Table 2. The results for Dataset [GVC*16] are shown in Table 3, and the results for Dataset [Lav09] are reported in Table 4. For all datasets, a leave-one-shape-out cross-validation method was used to show model generalisation capabilities, as the datasets' sizes are relatively small, and we treat each shape in turn as the test shape with the remaining shapes as the training set. As the tables show, our approach outperforms compared state-of-the-art methods with a large margin. Our method with SFS features are consistently better than our method without feature selection.

4.5. Ablation Studies

We conduct three ablation studies, first on the batch-norm to show its necessity, followed by different versions of MaskSSIM. Finally, cross-dataset features selection.

Batch-norm layer. The batch-norm is introduced first in the model to normalize input data, we choose *Dataset [LGD*06]* for the experiments. The network shows worse results without the batch-norm layer, as shown the comparison in Table 6.

We also tested different versions of Mask-SSIM. Four versions of Mask-SSIM: [*Mask-SSIM - Mask-SSIM Window - Mask-SSIM Negative -Mask-SSIM Merge*] are implemented. As described above, Mask-SSIM calculates the score for the foreground pixels and omits the background pixels. In this ablation, we examine a different approach that utilises pixels at boundaries and spatial relations. The result shown in Table 7.

Mask-SSIM window. As discussed, we only consider pixels as in the foreground where all pixels in the neighbouring window (3×3) are foreground pixels.

Mask-SSIM negative is a pixel-wise operation that is similar to the original (Mask-SSIM). However if this operation encounters a pixel that is considered part of the foreground in the first image but not in the second image, it penalises the score by adding -1. The result was worse than Mask-SSIM.

Mask-SSIM merge. The operation tries to find a smooth middle line between Mask-SSIM and Mask-SSIM negative as the result so we take the average of both values.

Overall, MaskSSIM achieves best performance, and so is used in our model.

4.6. Cross-dataset evaluation with feature selection

Our method relies on neural networks and feature selection to achieve the best performance. Although leave-one-shape-out test-

Table 7: *Comparison of different variants of MaskSSIM on the dataset [LGD*06] with cross-validation correlation results.*

	Mask-SSIM	Mask-SSIM window	Mask-SSIM negative	Mask-SSIM merge
armadillo	**0.28**	**0.28**	0.18	0.23
dyno	**0.44**	0.43	0.35	0.39
rockerarm	**0.24**	**0.24**	0.20	0.22
venus	**0.37**	**0.37**	0.29	0.34
average	**0.33**	**0.33**	0.25	0.29

Table 8: *Cross-dataset correlation results. The model trained on dataset [GVC*16] with SFS feature selection, and then tested on Datasets [Lav09] and [LGD*06]. We compare the performance with same dataset leave-one-shape-out testing results (including feature selection), and the previous best performing model PointSSIM.*

Dataset [Lav09]	within-dataset	cross-dataset	PointSSIM
Armadillo	0.51	0.43	0.26
Dyno	0.63	0.61	0.41
Lion	0.64	0.59	0.32
Bimba	0.63	0.62	0.22
average	0.60	0.58	0.30
Dataset [LGD*06]	within-dataset	cross-dataset	PointSSIM
armadillo	0.54	0.53	0.20
dyno	0.54	0.51	0.67
rockerarm	0.62	0.62	0.15
venus	0.65	0.55	0.62
average	0.58	0.55	0.41

Table 9: *Selected features using SFS for cross dataset evaluation where features are selected based on dataset [GVC* 16].*

selected features
Mask SSIM car paint flat
Mask SSIM matt blue smooth
Mask SSIM brown flat
No mask resin smooth
Mask SSIM clay studio smooth
Mask SSIM metal anisotropic flat
No mask SSIM check rim light smooth

ing ensures training/test separation, feature selection has to be performed for each training/test split. To further evaluate the generalisability of our method, we perform cross-dataset evaluation, where the whole dataset [GVC*16] is used for training (including SFS feature selection), the selected features then used to train and test on other datasets [Lav09] and [LGD*06]. The selected features are shown in Table 9 and the performance is reported in Table 8. We compare the cross-dataset performance with within-dataset performance (cross validation including SFS) and PointSSIM which is the best performing previous method. As can be seen, the model in the cross-dataset setting achieves slightly worse correlation: for the dataset [Lav09], the average correlation drops from 0.60 to 0.58, but still much higher than existing method PointSSIM (0.30). Similar observations can also be made for the dataset [LGD*06]. This demonstrates that our learned model can be generalised to independent datasets with different types of distortions while still achieving good performance. Such models are also more efficient to deploy as only the selected rendering styles need to be generated during testing.

5. Conclusion

In this paper, we presented an image-based method to evaluate 3D shape quality. Shapes are rendered from 12 views, along with a range of rendering styles. A deep learning based approach is then used to learn to predict quality measures more closely related to subjective evaluation. Experiments on three datasets demonstrate that our method outperforms existing methods by a large margin. Our cross-dataset evaluation further demonstrates the generalisability of our learning based model. Future work will consider increasing the number of views to determine whether it significantly enhances the accuracy of the results and may reveals further distinctions in our observations.

References

[ACEH*18] ABOUELAZIZ I., CHETOUANI A., EL HASSOUNI M., LATECKI L. J., CHERIFI H.: Convolutional neural network for blind mesh visual quality assessment using 3D visual saliency. In *IEEE International Conference on Image Processing (ICIP)* (2018), pp. 3533–3537. 2

[ACEH*20a] ABOUELAZIZ I., CHETOUANI A., EL HASSOUNI M., LATECKI L. J., CHERIFI H.: 3D visual saliency and convolutional neural network for blind mesh quality assessment. *Neural Computing and Applications 32*, 21 (2020), 16589–16603. 2

[ACEH*20b] ABOUELAZIZ I., CHETOUANI A., EL HASSOUNI M., LATECKI L. J., CHERIFI H.: Combination of handcrafted and deep learning-based features for 3D mesh quality assessment. In *IEEE International Conference on Image Processing (ICIP)* (2020), pp. 171–175. 2

[ACEH*20c] ABOUELAZIZ I., CHETOUANI A., EL HASSOUNI M., LATECKI L. J., CHERIFI H.: No-reference mesh visual quality assessment via ensemble of convolutional neural networks and compact multi-linear pooling. *Pattern Recognition 100* (2020), 107174. 2

[ACEH*21] ABOUELAZIZ I., CHETOUANI A., EL HASSOUNI M., CHERIFI H., LATECKI L. J.: Learning graph convolutional network for blind mesh visual quality assessment. *IEEE Access 9* (2021), 108200–108211. 2

[ACEHC18] ABOUELAZIZ I., CHETOUANI A., EL HASSOUNI M., CHERIFI H.: Reduced reference mesh visual quality assessment based on convolutional neural network. In *International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)* (2018), IEEE, pp. 617–620. 2

[ACEHC21] ABOUELAZIZ I., CHETOUANI A., EL HASSOUNI M., CHERIFI H.: No-reference mesh visual quality assessment using graph-based deep learning. In *IEEE International Workshop on Multimedia Signal Processing (MMSP)* (2021), pp. 1–6. 2

[AE18] ALEXIOU E., EBRAHIMI T.: Point cloud quality assessment metric based on angular similarity. In *IEEE International Conference on Multimedia and Expo (ICME)* (2018), pp. 1–6. 2

[AE20] ALEXIOU E., EBRAHIMI T.: Towards a point cloud structural similarity metric. In *IEEE International Conference on Multimedia & Expo Workshops (ICMEW)* (2020), pp. 1–6. 2, 6

[AEHC16] ABOUELAZIZ I., EL HASSOUNI M., CHERIFI H.: A curvature based method for blind mesh visual quality assessment using a general regression neural network. In *International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)* (2016), IEEE, pp. 793–797. 2

[AEHC18] ABOUELAZIZ I., EL HASSOUNI M., CHERIFI H.: Blind 3D mesh visual quality assessment using support vector regression. *Multimedia Tools and Applications 77* (2018), 24365–24386. 2

[AOEHC15] ABOUELAZIZ I., OMARI M., EL HASSOUNI M., CHERIFI H.: Reduced reference 3D mesh quality assessment based on statistical models. In *International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)* (2015), IEEE, pp. 170–177. 2

[Can86] CANNY J.: A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, 6 (1986), 679–698. 4

[dSCR*21] DE SOUZA J. P. C., COSTA C. M., ROCHA L. F., ARRAIS R., MOREIRA A. P., PIRES E. S., BOAVENTURA-CUNHA J.: Reconfigurable grasp planning pipeline with grasp synthesis and selection applied to picking operations in aerospace factories. *Robotics and Computer-Integrated Manufacturing 67* (2021), 102032. 1

[EMF21] EPPNER C., MOUSAVIAN A., FOX D.: ACRONYM: A large-scale grasp dataset based on simulation. In *IEEE International Conference on Robotics and Automation (ICRA)* (2021), pp. 6222–6227. 1

[FWDX*18] FENG X., WAN W., DA XU R. Y., PERRY S., LI P., ZHU S.: A novel spatial pooling method for 3D mesh quality assessment based on percentile weighting strategy. *Computers & Graphics 74* (2018), 12–22. 2

[FYJ*22] FREER J., YI K. M., JIANG W., CHOI J., CHANG H. J.: Novel-view synthesis of human tourist photos. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (2022), pp. 3069–3076. 1

[GVC*16] GUO J., VIDAL V., CHENG I., BASU A., BASKURT A., LAVOUE G.: Subjective and objective visual quality assessment of textured 3D meshes. *ACM Transactions on Applied Perception (TAP) 14*, 2 (2016), 1–20. 5, 6, 7, 8

[HG22] HAHNER S., GARCKE J.: Mesh convolutional autoencoder for semi-regular meshes of different sizes. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (2022), pp. 885–894. 1

[HKR93] HUTTENLOCHER D. P., KLANDERMAN G. A., RUCKLIDGE W. J.: Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15*, 9 (1993), 850–863. 4

[Lav09] LAVOUÉ G.: A local roughness measure for 3D meshes and its application to visual masking. *ACM Transactions on Applied perception (TAP) 5*, 4 (2009), 1–23. 5, 6, 7, 8

[Lav11] LAVOUÉ G.: A multiscale metric for 3D mesh visual quality assessment. In *Computer Graphics Forum* (2011), vol. 30, Wiley Online Library, pp. 1427–1437. 2

[LBDG*18] LANARAS C., BIOUCAS-DIAS J., GALLIANI S., BALTSAVIAS E., SCHINDLER K.: Super-resolution of Sentinel-2 images: Learning a globally applicable deep neural network. *ISPRS Journal of Photogrammetry and Remote Sensing 146* (2018), 305–319. 4

[LGD*06] LAVOUÉ G., GELASCA E. D., DUPONT F., BASKURT A., EBRAHIMI T.: Perceptually driven 3D distance metrics with application to watermarking. In *Applications of Digital Image Processing XXIX* (2006), vol. 6312, International Society for Optics and Photonics, p. 63120L. 5, 6, 7, 8

[LYS*21] LIU Q., YUAN H., SU H., LIU H., WANG Y., YANG H., HOU J.: PQA-Net: Deep no reference point cloud quality assessment via multi-view projection. *IEEE Transactions on Circuits and Systems for Video Technology 31*, 12 (2021), 4645–4660. 2

[LYXY20] LIU Y., YANG Q., XU Y., YANG L.: Point cloud quality assessment: Large-scale dataset construction and learning-based no-reference approach. *arXiv preprint arXiv:2012.11895* (2020). 2

[Tau95] TAUBIN G.: A signal processing approach to fair surface design. In *Proceedings of the 22nd Annual conference on Computer Graphics and Interactive Techniques* (1995), pp. 351–358. 6

[TW21] TONG WU LIANG PAN J. Z. T. W. Z. L. D. L.: Density-aware chamfer distance as a comprehensive metric for point cloud completion. In *Advances in Neural Information Processing Systems (NeurIPS)* (2021). 6

[WBSS04] WANG Z., BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing 13*, 4 (2004), 600–612. 1, 2, 4

[YMX*20] YANG Q., MA Z., XU Y., LI Z., SUN J.: Inferring point cloud quality via graph similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020). 2

[YPM*21] YANG W., PAXTON C., MOUSAVIAN A., CHAO Y.-W., CAKMAK M., FOX D.: Reactive human-to-robot handovers of arbitrary objects. In *IEEE International Conference on Robotics and Automation (ICRA)* (2021), pp. 3118–3124. 1

[ZSM*21] ZHANG Z., SUN W., MIN X., WANG T., LU W., ZHAI G.: No-reference quality assessment for 3D colored point cloud and mesh models, 2021. arXiv:2107.02041. 2

[ZZMZ11] ZHANG L., ZHANG L., MOU X., ZHANG D.: FSIM: A feature similarity index for image quality assessment. *IEEE Transactions on Image Processing 20*, 8 (2011), 2378–2386. 4