Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH Open Access

Automated visual quality assessment for virtual and augmented reality based digital twins

Ben Roullier[1], Frank McQuade[1], Ashiq Anjum[2], Craig Bower[2*] and Lu Liu[2]

Abstract

Virtual and augmented reality digital twins are becoming increasingly prevalent in a number of industries, though the production of digital-twin systems applications is still prohibitively expensive for many smaller organisations. A key step towards reducing the cost of digital twins lies in automating the production of 3D assets, however efforts are complicated by the lack of suitable automated methods for determining the visual quality of these assets. While visual quality assessment has been an active area of research for a number of years, few publications consider this process in the context of asset creation in digital twins. In this work, we introduce an automated decimation procedure using machine learning to assess the visual impact of decimation, a process commonly used in the production of 3D assets which has thus far been underrepresented in the visual assessment literature. Our model combines 108 geometric and perceptual metrics to determine if a 3D object has been unacceptably distorted during decimation. Our model is trained on almost 4, 000 distorted meshes, giving a significantly wider range of applicability than many models in the literature. Our results show a precision of over 97% against a set of test models, and performance tests show our model is capable of performing assessments within 2 minutes on models of up to 25, 000 polygons. Based on these results we believe our model presents both a significant advance in the field of visual quality assessment and an important step towards reducing the cost of virtual and augmented reality-based digital-twins.

Keywords Digital twins, Visual quality assessment, Virtual reality, Augmented reality, Machine learning, 3D modelling

Introduction

The adoption of virtual and augmented reality (VR/AR) digital-twins is currently growing at a rapid pace, driven by advances in hardware and the use of these technologies in commercial and industrial applications.

VR and AR are immersive digital experiences that place the user in a simulated environment. VR allows users to experience and interact with a completely virtual world, while AR overlays virtual elements onto the user's real-world surroundings. A digital-twin is a digital representation of a real-life object or system, created by gathering data from sensors and cameras, as well as using product design data or physics simulations. Digital-twins mirror the life-cycle of their physical counterparts in real-time and allow for in-depth analysis and testing. When combined with VR/AR, digital-twins can be visualised and investigated as interactive 3D virtual objects. VR and AR provide intuitive ways to observe and collaborate on digital-twins. Users can view digital-twins from entirely new perspectives, visualise complex data in 3D and test design changes safely. The combination of VR, AR, and digital-twins help users gain insights for innovation and make better-informed decisions by bridging the gap between the physical and virtual world.

*Correspondence:
Craig Bower
csb28@leicester.ac.uk
[1] Bloc Digital, Derby, UK
[2] Computing & Mathematical Sciences, University of Leicester, Leicester, UK

The global market for VR/AR is expected to rise from $28.4b in 2022 to over $87b by 2030 [1], with applications already prevalent in fields as diverse as medicine [2, 3], engineering [4, 5], architecture [6, 7], retail [8], entertainment [9] and training [10, 11]. At present, the development of these applications is costly and time consuming, limiting their use to large organisations with the budget to afford them. Research is ongoing in many fields with the aim of reducing the time and cost of VR/AR-based digital-twin development, opening these technologies up for use to wider audiences.

One of the principal costs of VR/AR-based digital-twin development is the need to produce 3D assets which are both visually representative and performant enough to be rendered in real time at the high frame rates required for VR. In most digital-twins, these assets are polygonal meshes. Polygonal meshes offer a lightweight representation of arbitrary 3D objects, approximating any given shape as a finite collection of interconnected 2D polygons and vertices. As the number of polygons increases, the accuracy of the approximation improves, however the rendering time of the mesh increases accordingly. While render times are important in any 3D application, they are especially important in VR. Low frame rates caused by long render times are annoying on a screen, however in VR they can lead to motion sickness, ruining the experience, [12]. The creation of assets for VR/AR is thus often a balance between visual accuracy and rendering performance. In many cases, VR assets are produced manually by skilled digital artists. These artists must manage the balance between accuracy and polygon count, and often use computer aided design (CAD) drawings as a reference on which to base the asset. Rendering the CAD drawings themselves requires considerable effort, and as such they are often unsuitable for direct VR/AR use [4, 13, 14].

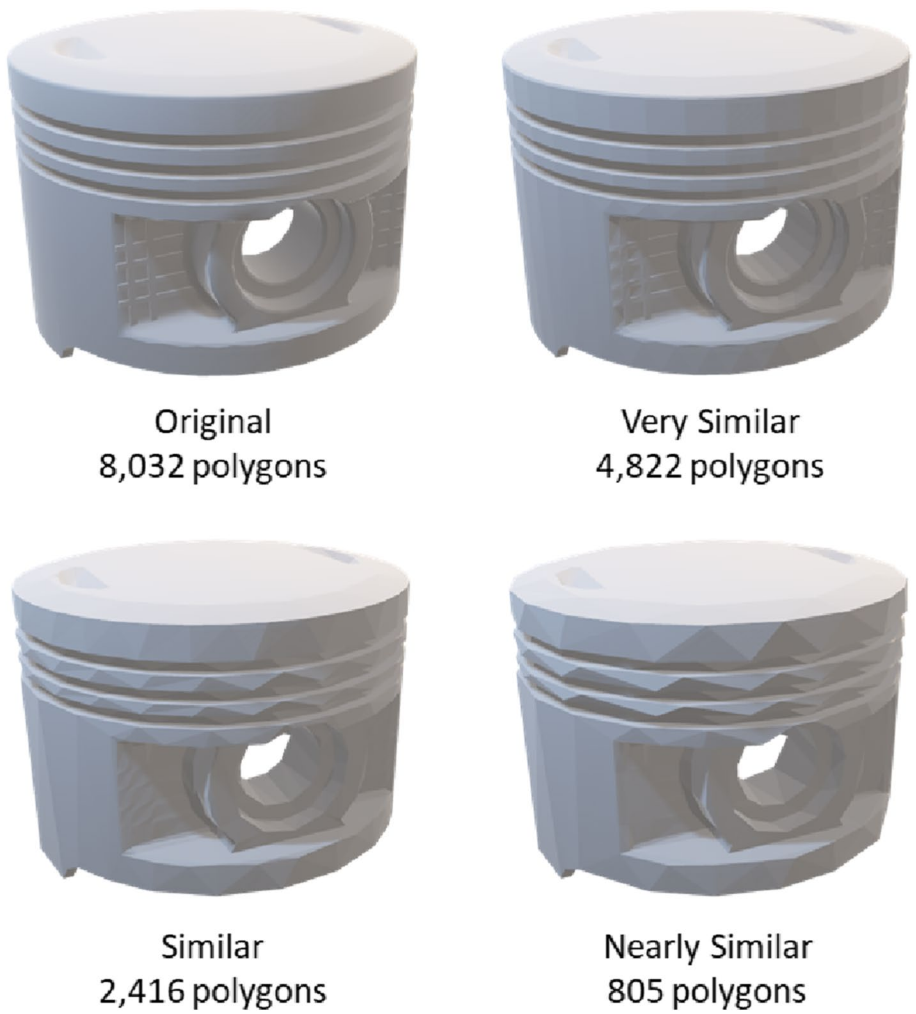
In recent years, research and development efforts have attempted to automate the production of high quality, performant 3D assets [13, 15–18]. A method of particular interest is that of mesh decimation: Decimation is an automated process which takes a mesh and attempts to reduce the total number of polygons without affecting visual appearance. Decimation can be applied to meshes converted directly from CAD to produce assets suitable for VR/AR use. The most common decimation method is Hoppe's progressive meshes algorithm [19]. In this method, the vertices in a mesh are ordered according to a cost function. Vertices are removed one by one starting with the cheapest, and the polygons sharing that vertex are merged into a single polygon. The algorithm allows the user to stop decimation at any point to choose the desired quality and polygon count.

While Hoppe's algorithm is ubiquitous in industry, there are two issues with its use in an automated system: Firstly, the cost function used must accurately determine the impact of removing each vertex on the overall visual quality of the mesh. A number of cost functions have been proposed, including several based on curvature, surface energy, and quadric error metrics [20, 21], however these are often weakly related to the subjective visual quality of the mesh. Second, the stopping point of the algorithm is usually manually chosen using a target number of vertices. Recent applications attempt to set the stopping point automatically, however these applications depend on many user selected variables and often still result in unsuitable geometry. For example, the Polygon Cruncher (see [22]) commercial software allows the user to set the desired similarity of the result to the original mesh as either "nearly similar", "similar" or "very similar", as well as setting other options to control the behaviour around corners, borders, etc. While this reduces much of the need for manual work, human judgment is still required, precluding fully automatic decimation. Figure 1 shows an example mesh decimated using Polygon Cruncher.

While tools like Polygon Cruncher can greatly speed up the manual production of VR/AR assets, it is clear that full automation requires the computerisation of the human visual quality assessment (VQA) process. An automated VQA model is required if the system is to guarantee that any meshes produced are visually accurate. Automatic visual quality assessment of 3D meshes is a highly active area of research. An in-depth review of this research is given in Related work.

In this work we have developed a machine learning algorithm for the visual quality assessment of decimated 3D meshes. Our algorithm differs from most of the VQA algorithms presented in the literature, which tend to focus on other methods of mesh alteration such as smoothing, compression, noise addition, and watermarking (see [23, 24]). Unlike decimation, these methods do not alter the number or connectivity of vertices within the mesh. As such, these methods are much simpler to model, as the properties of a given vertex can be easily compared between the reference and test meshes. In contrast, assessing decimated meshes must also involve identifying which vertices on the test mesh correspond to a given vertex on the reference mesh (the mesh correspondence problem, [25]). Though this complicates the VQA algorithm, the ubiquitous use of decimation in the production of 3D meshes for VR/AR drives the need for this research.

Our algorithm compares test meshes of unknown quality to a reference mesh of the same object and classifies how well the test mesh represents the reference mesh.



Original
8,032 polygons

Very Similar
4,822 polygons

Similar
2,416 polygons

Nearly Similar
805 polygons

Fig. 1 Results of Polygon Cruncher decimation process on a sample mesh at 3 different qualitative levels of similarity

The reference mesh is known to be of sufficient visual quality, but contains too many polygons to be rendered performantly in VR/AR. The test meshes are produced by decimating the reference mesh to varying levels of quality using Polygon Cruncher. The algorithm classifies each test mesh as either “ruined”, “bad”, “good” or “perfect” based on training data collected through human assessment. Classification is performed using a random forest algorithm to amalgamate the influence of over 100 visual quality metrics based on geometric features of both the reference and test meshes. Our model is trained on a set of 3,996 test meshes of various mechanical objects.

Given its intended use in controlling an automated decimation process, we define the success of our algorithm based on precision. In this case, precision is defined as the likelihood that a mesh classified as “good” or “perfect” has been correctly classified. Our results show a precision of 91.5%, rising to 97.3% when the model is weighted to strongly reject false positives. We choose to focus on false positives due to the intended use of the algorithm: In a decimation control process, false positives indicate that a visually poor mesh has been classified as good, resulting in the production of unusable assets. Conversely, false negatives imply the rejection of a good mesh as poor, leading to the adoption of a different mesh, likely with a slightly increased polygon count. The latter error is of much less importance than the former in the production of usable 3D assets for VR/AR. Given these considerations, we believe our VQA model represents a key step towards the production of automated decimation algorithms and processes for the automatic creation of high quality assets for VR and AR.

This paper presents our work as follows: Related work provides a detailed discussion of the VQA algorithms produced to date, and their influence upon our model. Model design discusses in detail the design of our algorithm and the visual quality metrics used. Experimental methodology discusses the implementation of the model, including details on the training process and collection of associated data. Results discusses the results of our model in terms of both visual assessment performance and processing time. Finally, Conclusions presents the conclusions of our work and suggestions for further research in this field.

Related work

Visual quality assessment of 3D meshes has been an active area of research for over two decades (see [26]) with several methods being developed throughout this time. In all cases, the principal aim of VQA is to determine the impact on the visual quality of a 3D mesh caused by one or more modification processes, [24]. In many cases, VQA methods are applied to predict if a given process is likely to lead to unacceptable distortion in the appearance of particular mesh, [27]. The likelihood of this distortion is seen to depend heavily on both the geometry of the mesh and the modification process used. As such, many methods are limited to predicting the effects of a particular process, or are limited in the range of meshes considered. In [28], the influence of a decimation process on 3D model morphology is studied.

VQA methods

Mesh visual quality models can be broadly grouped into two classes: image-based methods and model (geometry) based methods, [23]. Image based methods assess the quality of 3D models by considering the quality of one or more 2D projections of the objects. As such, the accuracy of such metrics is dependent on the viewpoint of the projection (see [29, 30]). Many image-based metrics are based around similar methods used in 2D image processing, such as the structural similarity index (SSIM) (see [31]) and visible difference predictor (VDP), [32]. While such methods were initially popular, later authors suggest that image-based methods are not suitable for assessing the quality of 3D objects due to the influence of other factors such as lighting and viewing angle, [33]. As such, most developments in the field of VQA in recent years have focused on model-based metrics. Image based metrics are thus not considered further within this paper.

Model based metrics are based directly on observable geometric features of the mesh itself. Commonly used features include simple measurements such as mesh volume and surface area, as well more complex features such as curvature and dihedral angles [23, 27, 30, 34–36]. Model-based methods are further classified as full-reference, part-reference, or no-reference (blind) according to the availability of the reference mesh. The authors of [37] deal with simplifying meshes reconstructed from 3D point clouds of buildings, using an edge collapse algorithm constrained by preserved structural points and focus on shape preservation during simplification. This work uses a model-driven approach with hand-crafted constraints and parameters then evaluate on individual buildings and a large urban scene point cloud. The work in [38] also utilises edge collapse, whilst taking advantage of parallel computing and hardware often used in machine learning.

Full-reference methods have access to both the test mesh and the reference mesh on which it was based. Part-reference methods do not have access to the reference mesh but do have access to some data regarding it. No-reference models only have access to the test mesh, [27]. Full and part-reference methods tend to work by comparing the values of one or more geometric parameters between the reference and test meshes. In contrast, blind methods often use machine learning techniques such as convolutional neural networks (CNN) (see [33]) or support vector machines (see [29]) to assess model quality. In general, full-reference methods tend to be the most accurate and are therefore preferred. Part-reference and no-reference models are generally only used when reference data are unavailable or otherwise unusable, [29]. Given the intended use of our model in controlling the decimation of existing (reference) meshes, we only focus on the discussion of full-reference methods within this paper.

Full-reference VQA methods

Full-reference VQA models predominantly use metrics which allow for a direct comparison between the reference and test meshes. One of the earliest such metrics was the root mean squared distance (RMS), [26], which finds the average distance between points on the reference mesh and their equivalent regions on the test mesh. Many VQA models also made use of the Hausdorff distance, defined as the maximum distance between any point on one mesh and the equivalent point on another mesh [23, 35, 39–41]. Research later showed that such purely geometric measures rarely correlate well with human visual perception however (see [27, 42]).

Figure 2 shows an example of the drawbacks of purely geometric measures, as previously shown by Wang et al., [43]. The figure shows an original mesh (left) and two meshes subjected to distortion through noise addition (centre, right). While it appears that the mesh on the right is much more heavily distorted than the one in the centre, both have the exact same value of RMS distance. As such, RMS value alone is not sufficient to assess the effect of a distortion on the visual quality of a mesh.

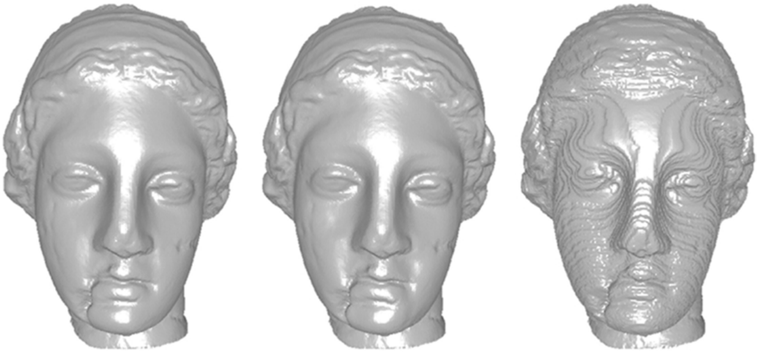


Fig. 2 Three meshes - original (left), lightly distorted (centre) and heavily distorted (right). The two distorted meshes have the same value of RMS distance. From [43]

Perceptual quality metrics

In order to address the shortcomings of purely geometric measures, later work aimed to better account for human perception using perceptual metrics [23, 24, 36, 42, 44, 45]. The (subjective) distortion perceived by a human is found to be a combination of the actual geometric (objective) distortion, setting features (e.g. lighting, viewing angle), and intricacies of the human visual system [27, 46]. Later models therefore attempt to use metrics which take some of these factors into account. Such metrics commonly include measures of curvature [36, 42, 44, 47, 48], dihedral angle [30, 35], normal vectors (see [49]) and surface roughness (see [36, 43]). These metrics are seen to have a stronger correlation to visual perception than simpler measures such as the Hausdorff distance [27, 42].

Particularly influential methods based on perceptual metrics include the *Dihedral Angle Mesh Error* (DAME) [30], the *Tensor Based Perceptual Distance Measure* (TPDM) [44], and the (Multiscale) *Mesh Structural Distortion Measures* (MSDM/MSDM2) [36, 47]. TPDM, MSDM and MSDM2 are based on various definitions of curvature. While curvature is a useful proxy for visual appearance, the piecewise nature of mesh surfaces greatly complicates its calculation (see [50, 51]).

Methods based on perceptual metrics tend to operate in a three-stage process (see [36, 52]). First, the method calculates the value of one or more metrics for every point (polygon or vertex, depending on the metric in question) on the mesh. Next, these values are summarised over the entire mesh to produce an overall score for the object (see [42, 46]). Finally, this score is compared between the reference and test meshes and used to elicit a judgment on the overall model quality. While early methods used the arithmetic mean of a metric, later works attempt to use alternative strategies to approximate key features of the human visual system.

Accounting for the human visual system

It is well known that there is only a weak correlation between perceived visual quality and actual geometric distortion (see [24, 44]). Much of this discrepancy is related to two factors: Visual impact and visual saliency. Visual impact refers to the noticability of a given distortion. For example, it is observed that humans are likely to report a model with few large distortions as more distorted than one with many small distortions, even if the total geometric impact is the same, [47]. Visual saliency refers to the relative importance of particular mesh features on the subjective quality of the distorted mesh (see [36, 48]). For example, it has been shown that certain forms of distortion are more noticeable when applied to previously smooth areas of a mesh than to previously rough areas (see [24, 53]).

Visual impact estimation

To better estimate visual impact, a perceptual metric must highlight particularly large distortions to a degree higher than their geometric value might suggest. Many authors achieve this using pooling techniques to summarise distortion metrics, rather than the arithmetic mean. Minkowski pooling is a particularly common method to account for the impact of larger distortions (see [24, 36, 43, 46]). As with the arithmetic mean, Minkowski pooling takes the contributions of all the distortions on a mesh and returns a single value for the overall mesh. Unlike the mean however, Minkowski pooling weights this value towards over-representation of effects with higher visual impacts. The Minkowski pooling X_M of variable X is found using 1; where M is the Minkowski parameter and N is the number of observations. Higher values of M imply a higher weighting towards large distortions. In many studies, M is chosen arbitrarily, but tends to have a value between 2 and 3, [52]. Note that setting $M = 1$ reduces Minkowski pooling to the arithmetic mean,

$$X_M = \left[\frac{1}{N} \sum_{i=1}^{N} X_i^M \right]^{\frac{1}{M}}. \tag{1}$$

Minkowski pooling is used to approximate visual impact in many full-reference VQA models. While Minkowski pooling somewhat accounts for human perceptual behaviour, this method has its drawbacks: Firstly, Minkowski pooling treats all distortions as independent, when in reality, the effects of multiple distortions overlap to influence visual quality perceptions, [52]. Second, Minkowski pooling assumes the visual impact of a distortion is directly related to the extent of geometric displacement with no other compounding factors. Later models aim to account for this in several ways: Feng et al. [52], use a novel weighting method based on polygon surface area. This method is seen to overemphasise the impact of roughness however, as rough areas necessarily contain smaller polygons than smooth areas. Nouri et al. [46], combine Minkowski pooling with a weighting factor based on visual saliency. This method results in a limited improvement in results, however this is outweighed by a significant additional computational complexity. In our model we use standard Minkowski pooling to approximate the visual impact of distortions, as later developments are not seen to offer significant benefits over this method.

Saliency estimation

Visual saliency is a somewhat more complex concept than visual impact, relating to how important the human visual system considers certain aspects of visual stimuli. Visual saliency is a major area of research in both computer vision and neuroscience, [54]. Saliency is occasionally incorporated into mesh VQA models as a weighting factor, such that perceptual metrics are weighted more heavily in regions that contribute more to visual quality [29, 39, 44, 46, 52]). Saliency has been approximated in several studies as a function of curvature [36, 44, 48]) or surface roughness, [46]. Many authors apply this weighting over multiple scales, as what is considered salient at one scale may not be at another, [48]. Despite significant research, most saliency-based methods are seen to give conflicting results, [52]. While saliency is a key feature in many model driven VQA methods, its complexity and poor understanding has led to it being considered an unnecessary complication in recent data driven VQA systems, [34]. As such, our model does not attempt to account for visual saliency.

The mesh correspondence problem

A major complication in many full-reference VQA models is the need to compare certain metrics at equivalent points between the reference and test meshes. For example, Roy et al. [49], assess visual quality by considering the deviation in the average normal vector for each polygon around a vertex before and after distortion is applied. Similarly, the MSDM metric [36], uses curvature at equivalent points on the mesh to determine quality. The complication with these approaches stems from the need to identify which point (vertex or face) on the test mesh corresponds to a given point on the reference mesh. This issue is generally referred to as the mesh correspondence problem.

In many studies, the mesh correspondence problem is negated entirely by avoiding distortion processes which alter the connectivity of the mesh. Smoothing, noise addition, compression and watermarking all distort meshes by moving vertices without changing their number or connectivity. As such, finding the correspondence between the reference and test vertices is trivial, as they will be at the same index in both meshes. Many VQA models are therefore limited to use only these methods of distortion [29, 30, 34–36, 42–44, 46, 52, 53]. In earlier work, smoothing and noise addition were considered sufficient methods to represent a wider range of distortions, [36]. Later assessments however suggest that only considering models in which the vertex count and connectivity remain unchanged is a serious drawback of previous VQA models (see [24, 47]). Given that our work focuses on mesh decimation, a method which by its nature destroys the connectivity of a mesh, we must consider the correspondence problem in our model. The relative lack of previous models which consider correspondence is problematic however, as it precludes the comparison of our work with many of the metrics presented by previous authors. Solving the correspondence problem involves determining, for every vertex on the reference mesh, which vertex on the test mesh is the closest to it in 3D space. For any given pair of meshes this is a well-defined problem with a unique solution, however the calculation of this solution is computationally expensive. Solving the correspondence problem through brute force requires assessing the Euclidean distance between every pair (r, t) of reference and test vertices. The complexity of this problem is therefore seen to be $\mathcal{O}(RT)$ where R and T are the vertex counts of the reference and test meshes, respectively [25].

Many authors have proposed schemes for solving the correspondence problem in less time than is required by brute force methods. Several authors attempt to speed up the calculation by approximating correspondence rather than solving it exactly [25, 51, 55, 56]. Given the non-uniform distribution of vertices within arbitrary 3D shapes however, this does not guarantee a sufficiently accurate solution for the visual quality assessment of decimated meshes. Other methods attempt to use more efficient spatial search paradigms to speed up the exact solution. Roy et al. [49], use a method of bounding grids to reduce the number of test vertices searched for a given reference vertex. Many later works use *octrees* (3-dimensional space partitioning trees) to speed up the search [20, 44, 47, 57]. Octrees partition arbitrary 3D shapes by recursively subdividing them into eight octants, and are commonly used in a number of algorithms related to 3D graphics processing. Such methods are generally preferred in most modern VQA models due to their increased speed and simple implementation.

VQA with machine learning

In recent years, an increased focus has been given to the use of machine learning in visual quality assessment. Unlike earlier approaches which attempt to link visual quality to a single perceptual metric, machine learning methods allow assessments to be based on multiple metrics. Lavoue et al. [42], present a method in which eight separate metrics are combined using linear regression to produce a single quality score which is then trained against human judgments. The authors suggest that multiple metrics in combination give a much more accurate prediction of quality than any single metric. The number of variables contained in the model is further expanded by using a number of statistical measures of these variables, including the mean, standard deviation, skewness and kurtosis.

Later machine learning based models use a similar approach to Lavoue, with encouraging results (see [34, 35]). The model presented by Yildiz et al. [34], is of particular interest. This model aims to predict distortion caused by smoothing and noise addition using a combination of 28 features, found as the mean, standard deviation, skewness and kurtosis of six particular curvature measures plus a local roughness measure. In addition, the authors considered metrics relating to visual saliency and mesh dihedral angles, but these were removed from the final model due to a lack of contribution to the results. The authors use a linear optimisation process to combine these features into a prediction of visual quality trained against crowdsourced human assessments. Training was performed against a set of eleven low polygon meshes of various objects subjected to distortions through noise addition and smoothing. Their results show better performance than any single metric method, and this accuracy is shown to be independent of the mesh under test.

VQA and digital twins

The *Metaverse* is a virtual world that maps and interacts with the real world, [58]. The quality of content, device and interaction in the metaverse all have an important impact on the Quality of Experience (QoE). In this context, VQA is a direct measure of a users' QoE in Digital Twin (DT) environments. The work in [59] identifies the latest developments in DT technologies and AI, associated with current challenges in creating a system that can truly bring reality and virtual closer together. A QoE model of DT systems was introduced for transmission in a large scale immersed haptic virtual reality over the Internet and objectively infers important DT QoE physiological aspects, such as fatigue (see [60]). The authors of [61] present a novel workflow for combining voxel representations and coloured point clouds, to create digital twins of physical objects with 0.1 mm precision. A concept DT framework for prefabricated construction was developed in [62] which also considers automated VQA, The authors of [63] introduce using Digital-Twins integrated with VR to enhance digital learning for driving an industrial mobile robot. Experiments validated that the Virtual Reality environment helped improve digital learning. In [64], the authors address the problem of subjective visual quality assessment protocols, but do not consider using machine learning to automate the process.

Influence on model design

The model presented within this paper is influenced by the work of Yildiz et al. [34]. In particular we attempt to extend Yildiz' method towards the visual assessment of meshes produced through decimation. As with this work, our model uses a machine learning approach to combine the influence of several geometric and perceptual metrics into a single result. Where Yildiz et al. consider 28 metrics, our model considers 108 variables based on geometric properties of both the original and distorted meshes. As with Yildiz et al., we utilize the mean, standard deviation, skewness and kurtosis of certain properties to better classify the mesh. We use Minkowski pooling to incorporate visual impact into our method, however we have chosen not to consider visual saliency as observations from both Yildiz et al. and others suggest its inclusion does not add sufficient performance (see [34, 52]). Unlike Yildiz et al., we incorporate solution of the correspondence problem into our model. This is necessary for the application of the model towards decimation rather than smoothing and noise addition as presented in the previous work. Finally, we train our model using a similar crowdsourcing approach, applying our model to 3, 996 unique meshes (108 decimations of 37 objects) compared to Yildiz' 168 meshes produced from 11 objects).

As mentioned earlier, our model differs from much of the literature in three important ways: First, while previous authors have applied machine learning to visual quality analysis, our model considers substantially more variables than any previously known work. Second, our model is trained on a total of 3, 996 meshes. This is considerably more than that seen in many approaches in the literature, which tend to use between 4 and 10 meshes [30, 35, 36, 44, 46, 47, 52, 53, 65].

Finally, our model focuses on decimation, a method which is greatly underrepresented in the literature in favour of constant connectivity methods. We choose to analyse decimation based on its importance to the commercial production of VR assets, however the lack of previous studies hinders the direct comparison between our model and previous efforts. We hope that our publication of a methodology specifically aimed at mesh decimation will encourage further research in this area, driving developments towards further automation of this commercially important process.

Model design

Our model presents a full-reference, machine learning approach to mesh visual quality assessment based on a combination of geometric and perceptual measures. The metrics used in our assessment are split into two categories: 1) Shape metrics, which depend only on either the reference or test mesh, and 2) similarity metrics, which represent a difference between the two meshes.

Shape metrics are found by extracting the relevant data from the mesh. Some similarity metrics are found simply by taking the ratio of a shape metric between the two meshes, while others require the solution of the correspondence problem before they are calculated. For every pair of meshes, a total of 108 variables (18 shape metrics, 9 shape ratios and 81 similarity metrics) are calculated. These variables are then fed into a random forest classifier to assess if the test mesh is visually representative of the reference mesh. This random forest model is trained based on recorded human visual quality judgments on over 21, 000 test meshes. Figure 3 gives an overview of the assessment process. The remainder of this section breaks up the operation of the visual quality assessment model in terms of the steps shown in Fig. 3.

Shape metrics

Shape metrics are those metrics which can be derived from a single mesh. Table 1 defines these variables, while the remainder of this subsection gives details on the justification and calculation of each variable. Each of the 9 variables in Table 1, is collected for both the reference and test meshes, giving a total of 18 shape metrics.

Many of the variables in Table 1 give rough approximations of the overall shape of the mesh. These are useful as they allow an assessment of visual quality which is independent of the shape of the object under test. Metrics which indicate only the scale of the model (e.g. volumes) are not used, so that the model is scale independent.

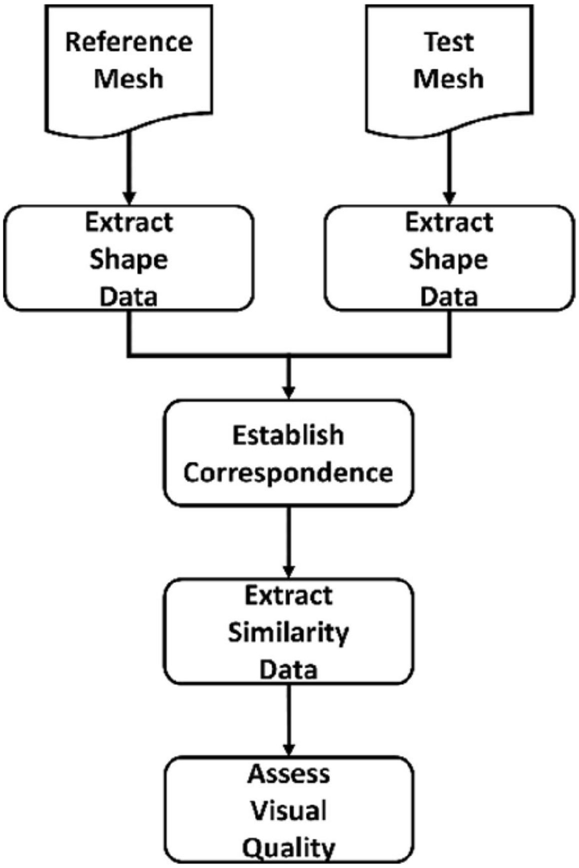


Fig. 3 Schematic of full Visual Quality Assessment process

Table 1 Shape variables in the Visual Quality Model

Symbol	Variable	Eq.
S_1	major squareness	2
S_2	minor squareness	3
ϕ	sphericity	4
ρ	bounding box density	5
E	shape efficiency	6
A_{skew}	skewness in polygon area	
A_{kurt}	kurtosis in polygon area	
A_{cov}	COV in polygon area	
α	connectivity	7

The major and minor squareness metrics are calculated from the dimensions of the non-axis-aligned bounding box surrounding the mesh. The box is found using Moore's brute force approximation, [66]. The squareness variables give a simple metric to approximate the overall shape of an object without including the object's scale. These variables are found according to (2) and (3), where L_1, L_2 and L_3 are the primary, secondary and tertiary lengths of the bounding box.

$$S_1 = \frac{L_1}{L_2}, \tag{2}$$

and

$$S_2 = \frac{L_1}{L_3}. \tag{3}$$

Sphericity is a further metric for approximating the object shape. Sphericity is found as the inverse ratio of an object's surface area to that of a sphere of equal volume. Sphericity is calculated using (4), where VM and AM are the mesh volume and surface area respectively. Mesh volume is found using the method of Zhang and Chen [67], while mesh surface area is the sum of polygon areas,

$$\phi = \frac{\sqrt[3]{\pi}(6V_M)^{\frac{2}{3}}}{A_M}. \tag{4}$$

Bounding box density refers to the ratio between the volume of the mesh and that of its bounding box, as found by (5). Shape efficiency is found as the surface area to volume ratio of the mesh divided by that of the bounding box, as given by (6), where V_B and A_B are the bounding box volume and surface area respectively,

$$\rho = \frac{V_M}{L_1 L_2 L_3}, \tag{5}$$

and

$$E = \frac{A_M V_B}{A_B V_M}. \tag{6}$$

The skewness, kurtosis, and coefficient of variation (COV) of polygon area are simply derived from the full set of polygon areas within the mesh. These variables are used as they present simple metrics for the complexity of the mesh. While the coefficient of variation depends on the mean and standard deviation of polygon area, these values are not reported themselves as they encode the scale of the object. Connectivity is found using (7), where p and v represent the number of polygons and vertices within the mesh. This has a value of approximately 2 for any manifold 3D mesh, with the exact value giving an indication of the complexity of the mesh,

$$E = \frac{p}{v}. \tag{7}$$

Note that despite their use in calculating α, the polygon and vertex counts are not used as shape metrics. This reduces the influence of polygon count on assessment results. As mentioned earlier, mesh visual quality tends to degrade as polygon count is reduced. Despite this, it is possible that a low polygon mesh may have a better visual quality than a high polygon mesh. As such, it is important that the model is not unfairly biased towards rejecting parts with fewer polygons.

Table 2 Shape ratio metrics

Symbol	Variable
R_V	volume ratio
R_A	surface area ratio
R_ϕ	sphericity ratio
R_ρ	bounding box density ratio
R_E	efficiency ratio
R_α	connectivity ratio
R_{skew}	polygon area skewness ratio
R_{kurt}	polygon area kurtosis ratio
R_{COV}	polygon area COV ratio

Table 3 Similarity metrics

Symbol	Variable	Eq.
D	matching distance	8
n	normal deviation	
θ	dihedral angle change	

Shape ratios

Shape ratio metrics are found simply as ratios of shape metrics between the reference and test meshes. Table 2 lists these metrics

All the metrics listed in Table 2 are found simply by dividing the value of the relevant measurement for the test mesh by that given for the reference mesh. Many of these metrics give a simple indication of the global extent of distortion between the meshes. For example, the surface area ratio of a completely undistorted test mesh would be exactly 1. The further from 1, the greater the distortion from the reference mesh to the test mesh.

Note that volume and surface area ratios are used as shape ratios despite their absolute values not being used as shape metrics as they encode object scale. Note also that squareness is not used in a shape ratio due to its approximate nature.

Similarity metrics

Similarity metrics are those variables which directly compare the appearance of the two meshes. Table 3 lists the similarity metrics used in the model. Note that all of these metrics depend on the solution for mesh correspondence, which is discussed further in Establishing correspondence.

Matching distance is the Euclidean distance between a reference vertex and its corresponding test vertex, normalised with respect to the characteristic length of the reference mesh. The characteristic length is given as the radius of a sphere of equal volume to the mesh. This normalisation is performed in order to remove the influence of model scale from the results of the analysis. Equation 8 gives the definition of matching distance Di for a given reference vertex i and corresponding test vertex i',

$$D_i = \frac{\sqrt{(x_i - x_{i'})^2 + (y_i - y_{i'})^2 + (z_i - z_{i'})^2}}{\sqrt[3]{\frac{3V_{M_i}}{4\pi}}}. \tag{8}$$

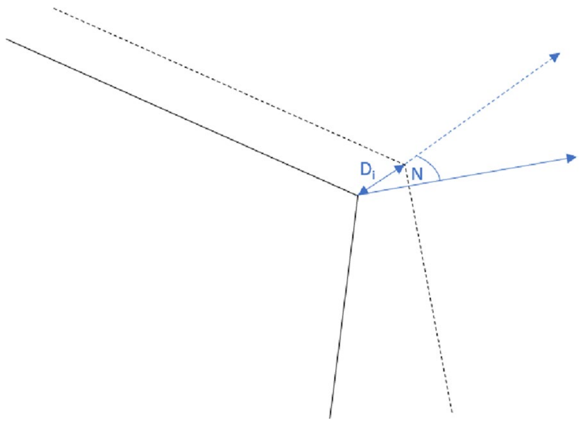
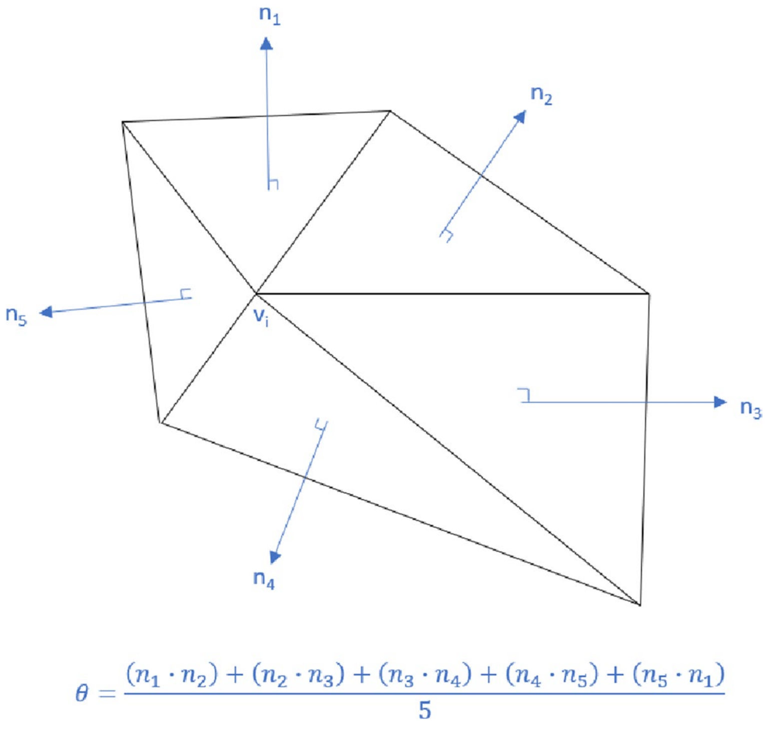


Fig. 4 Illustration of matching distance and normal angle deviation

The normal deviation metric is based on Roy's method (see [49]) and compares the vertex normal direction of each reference vertex to that of the corresponding test vertex. The vertex normal vector is found as the average normal vector for all the polygons sharing that vertex. The deviation between the vectors is found as their scalar product, in degrees. Figure 4 illustrates the matching distance and normal deviation metrics on a single vertex. In the diagram, the reference mesh is given as a solid line, with the test mesh given as a dashed line.

The dihedral angle change is the unsigned difference between the average dihedral angle of a reference vertex and that of the corresponding test vertex. Figure 5 shows the definition of mesh dihedral angle for vertex v_i. The average dihedral angle of a vertex is found using Algorithm 1, where V is the full list of vertices in a mesh.



$$\theta = \frac{(n_1 \cdot n_2) + (n_2 \cdot n_3) + (n_3 \cdot n_4) + (n_4 \cdot n_5) + (n_5 \cdot n_1)}{5}$$

Fig. 5 Definition of mesh vertex dihedral angle

Algorithm 1 Calculating Dihedral Angle

```
Require: $V$
Ensure: $\theta$
  for $v_i$ in $V$ do
      Find polygons $P_i$ containing $v_i$
      Find vertices $V_j$ contained in $P_i$
      $V_j \leftarrow V_j \backslash v_i$
      for $v_j$ in $V_j$ do
          Find two polygons $p_1, p_2$ containing $V_j$ and $V_i$
          Find polygon normals $N_1, N_2$
          $\theta_j \leftarrow N_1 \cdot N_2$
      end for
      $\theta \leftarrow$ mean$(\theta_j)$
  end for
```

Each of the three metrics in Table 3 is measured for every vertex on the reference mesh, with the results stored in three $1 \times N$ arrays. Minkowski pooling is then performed by raising each measurement to the power M, where M takes values of 1.0, 1.5, 2.0, 2.5 and 3.0, giving 15 $1 \times N$ arrays. Next, the mean, standard deviation, skewness, kurtosis and coefficient of variation are measured for every exponent of every metric, giving 75 metric values. Finally, the minimum and maximum value of the original ($M = 1.0$) metrics are also recorded, giving a final total of 81 similarity metrics.

Minkowski pooling was applied to these metrics at multiple levels ($i.e.withmultiplevaluesofM$) in an attempt to produce a set of measures with varying dependence on the scale of distortions. It is hoped that doing so will allow the VQA model to better consider meshes subject to a set of distortions of varying intensity. Similarly, we use the mean, standard deviation, skewness, kurtosis and coefficient of variation in order to further extend the variables considered, as suggested by Lavoue et al. [42].

Establishing correspondence

As mentioned in The mesh correspondence problem, many metrics for mesh visual quality assessment rely on a correspondence between the vertices of the reference and test meshes. In our work, we use a $k-d$ tree implementation to determine correspondence. The tree is built using the method of Maneewongvatana and Mount [68] as implemented in the SciPy analysis package, using a leaf size of 10 nodes. The tree is built from the vertices of the test mesh, and the algorithm then loops through the reference mesh to determine the nearest point in the tree to each reference vertex. This method returns a $2 \times N$ array, where N is the number of vertices in the reference mesh. For each reference vertex, the array stores both the index of and distance to the corresponding test vertex. A $k-d$ tree implementation was chosen as the results of initial tests showed this method to perform up to 50 times faster than a bounding grid method on representative meshes.

Visual quality assessment

The 108 objective metrics derived in Shape metrics through Similarity metrics sections are related to the subjective visual quality of the test mesh using a random forest classifier. The metrics recorded for a given pair of meshes are passed to the classifier as a 108-element array. No additional pre-processing is required, as all the metrics are already normalised real numbers. The classifier operates on this array and returns the predicted visual quality class for the mesh. We have chosen to use a random forest classifier in this work for several reasons: Firstly, unlike artificial neural networks, random forests are highly explainable models. This allows both authors and end users to interrogate the results and internal workings of the model, leading to a better understanding of the visual quality assessment process. Discussion around this interrogation is given in Influence of metrics, and this information is being used to drive further developments in 3D model processing.

Second, as they are based on decision trees, random forests are somewhat able to account for the interdependence between variables. For example, it may be seen that the relationship between surface area ratio and visual quality is much stronger for shapes with lower values of sphericity. Models based on decision trees allow for such effects by gating the surface area ratio relationship behind a sphericity threshold. Finally, the probabilistic nature of random forests helps to reduce the chance of over-fitting the model to the training data. This is important in the training of a visual quality model that is intended to work across a range of different mesh geometries

Experimental methodology

In this section we set out how we collected and verified data used to train the VQA model, how the model is trained and implemented.

Model Implementation

The model defined in Model design was implemented in Python 3. The SciKit-Learn package was used to build the random forest classifier. The Blender open-source 3D modelling tool was used to simplify the extraction of shape metrics from 3D meshes. The model was developed and tested on a laptop computer with a 1.8GHz Intel $i7$ CPU and 16GB of RAM running Windows 10.

Training data set

The model was trained against a set of 37 different objects. The objects were chosen to give a representative sample of mechanical parts including gears, screws, pistons etc. These parts were chosen to aid the development of an automated system for converting mechanical CAD models into visually representative 3D meshes, as detailed in Introduction. Figure 6 shows the parts that were used,

The authors attempted to find a set of publicly available data for use in training the model, however no appropriate set was found. Most public data sets either contained too few meshes to guarantee sufficient training data, or primarily contained models that were too visually simple to show significant distortion. Furthermore, most of the available data sets deal with other forms of mesh degradation, such as watermarking, rather than decimation. As explained in Introduction, decimation is a necessary step in the processing of CAD models to meshes, such that models showing alternative forms of deformation are unsuitable for this work.

Training data collection

The training data for the objects shown in Fig. 6 were collected using the following process: First, for every object, a high polygon “reference mesh” was produced. Next, each reference mesh was decimated using the Polygon Cruncher commercial software package [22]. Every possible combination of Polygon Cruncher’s input parameters (corresponding to each visual quality objective metric) was tested, resulting in 108 “test meshes” for each object (3, 996 test meshes in total). Finally, each test mesh was manually compared with its corresponding reference mesh for visual quality. The manual comparison of test and reference meshes was performed by a number of volunteers using a custom made tool. The tool showed the user the reference and test meshes side by side and asked the user to judge how well the test mesh represented the reference mesh. The tool allowed the user to rotate and scale both meshes simultaneously in order to fully compare the objects. Both the reference and test meshes were lit and rendered in exactly the same fashion in order to avoid the influence of external factors on the user’s judgment of mesh quality. Test meshes could be judged as belonging to one of four categories:

- Ruined - Either the test mesh is not recognisable as the same object as in the reference mesh, or the test mesh contains a flaw which would hinder the object’s intended function.
- Bad - The test mesh is recognizable but contains significant visual deformations compared to the reference mesh.
- Good - The test mesh is an adequate representation of the reference mesh, with only minor differences.
- Perfect - The reference and test meshes are indistinguishable.

Figure 7 shows example test and reference meshes for each of the four quality categories.

Every combination of reference and test mesh was duplicated 5 times to reduce human error (and combat against conflicting opinions on quality. Human volunteers may have conflicting opinions of the same mesh quality. The same volunteer may also change their opinion over time. Duplicating test meshes multiple times during construction of the training dataset alleviates this potential limitation.), giving a total of 19, 980 assessments. An additional 1, 120 meshes known to be perfect representations were also added to the test set to verify user performance. A total of 21, 100 meshes were thus assessed. The test meshes were finally split into 1, 055 sets of 20, with each set containing at least one known “perfect” mesh. The user did not know which mesh this was within a set. Whilst the authors concede there are limitations to using a customised training dataset, we mitigate this by implementing a machine learning approach that is robust to overfitting. These assessments were performed by 23 volunteers at Bloc Digital and the University of Derby.

Training data verification

The visual quality assessments collected were validated to reduce the likelihood of erroneous assessments polluting the training data. Given their subjective nature visual quality assessments must be made by human beings, however the tedious nature of the work can lead to boredom, distractions, and erroneous results. The use of 5 repeats of each test mesh somewhat alleviates this risk by allowing the data to be averaged, reducing the impact of one-off errors. Such an addition does not reduce errors caused by systematic or intentional mislabelling of meshes however.

In order to verify the accuracy of the human visual quality assessments, three further tests were performed: First, as mentioned in Training data collection, at least one mesh in every set of 20 was known to be a perfect representation of the test mesh. For each set, the human score for this model was interrogated. Any set which scored anything other than perfect for this mesh was discarded, and the whole set was later reassessed. Reassessment was required in 79 sets out of 1, 055.

Second, the results for each volunteer were collated, the scores converted to integer values (ruined = 1, bad = 2, good = 3, perfect = 4) and their mean and standard deviation calculated. All users gave a mean score between 2.4 and 3.3, with a standard deviation between 0.74 and 1.05. These scores were well within the expected range for the data and agreed with the results of sets assessed by the authors.

Finally, each set of 20 meshes was inspected to ensure that all sets had a sufficient range of scores. Out of 1, 055

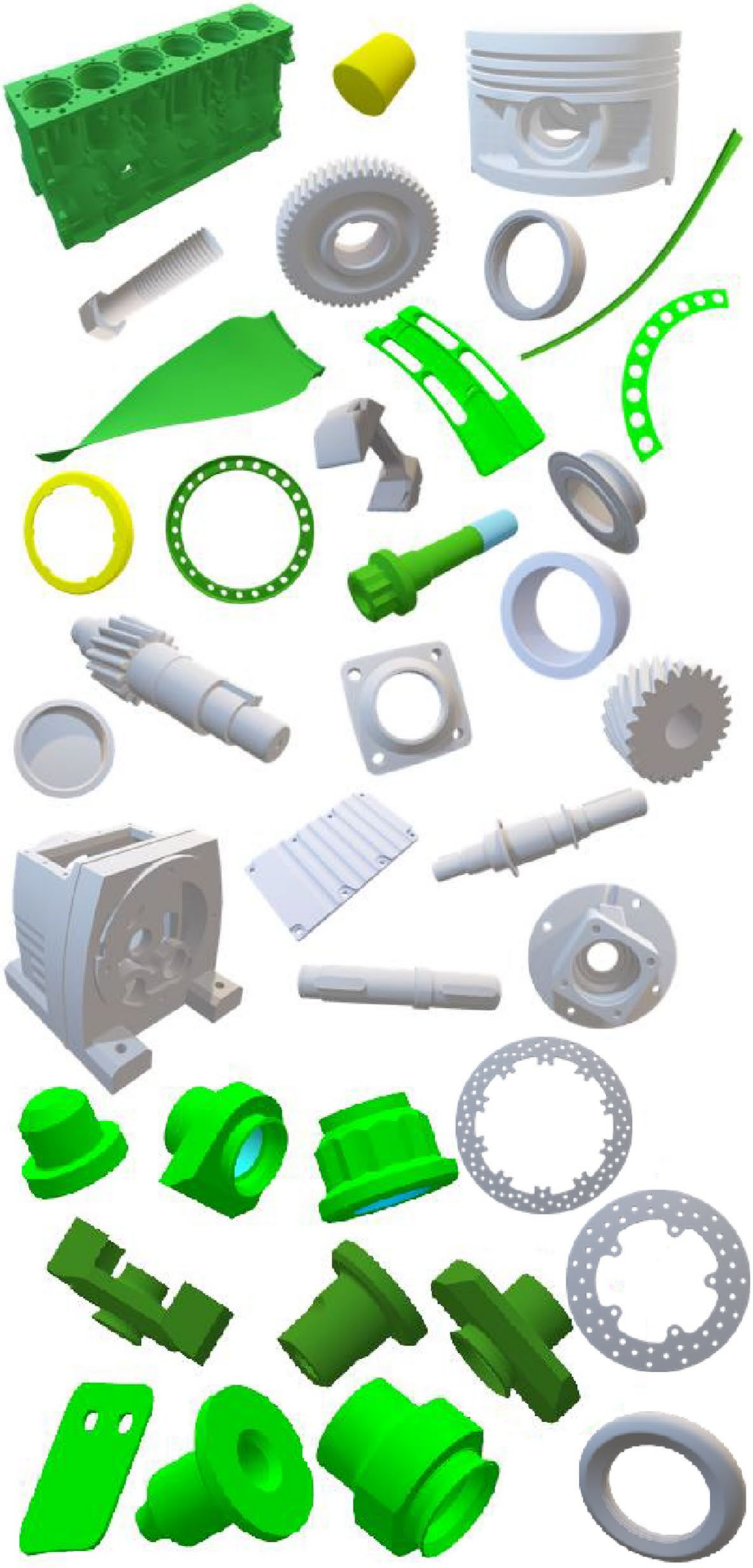


Fig. 6 Mechanical parts used in training the visual quality model. Each model was represented by 108 test meshes, giving 3, 996 meshes in total

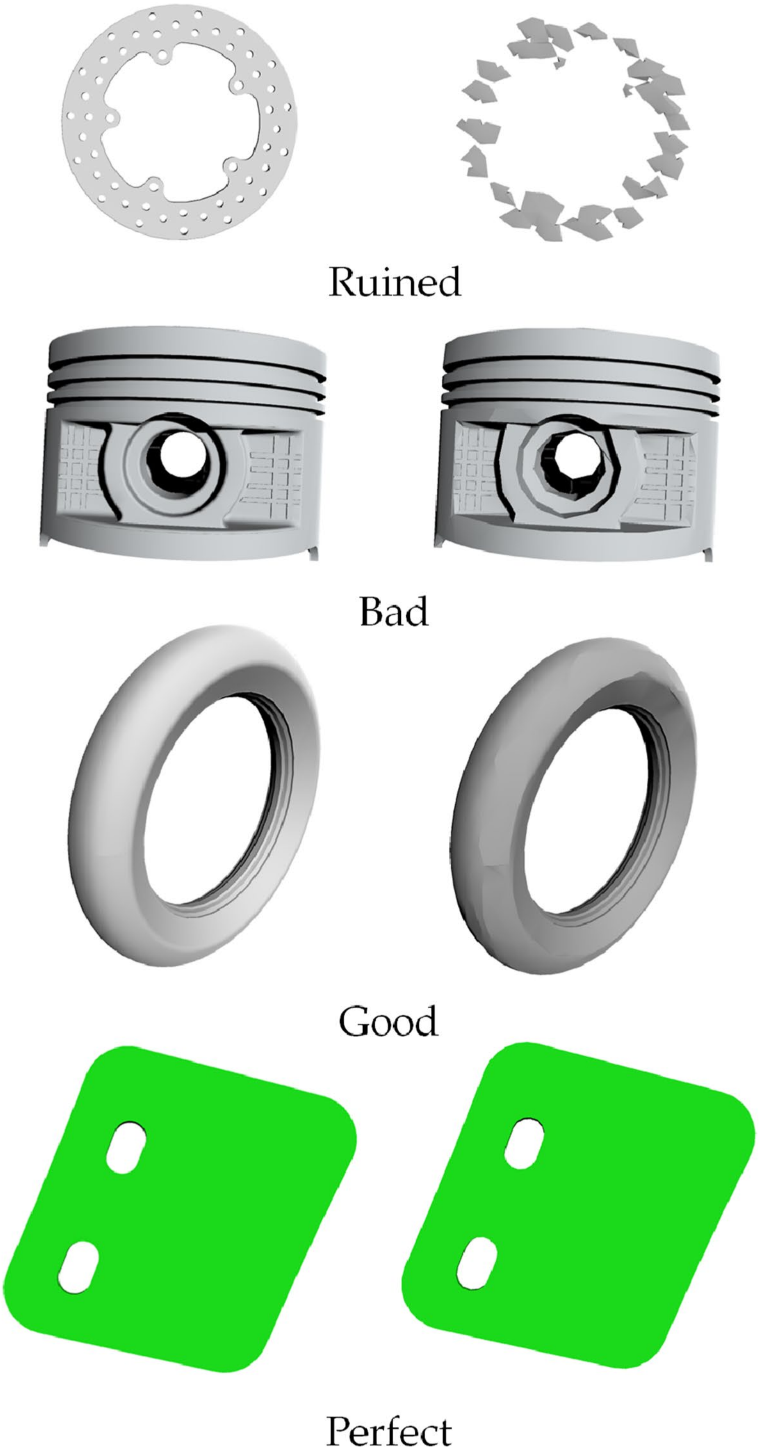


Fig. 7 Examples of meshes belonging to each of the four assessment categories

sets of 20, 1, 053 sets had a range of at least 2, which was considered acceptable. Two sets had a range of 1, although on reassessment by the authors, these results were deemed appropriate and allowed to stand. No sets had a range of 0.

These tests were considered appropriate to determine that all visual quality assessments had been undertaken correctly. Any differences in results are therefore expected to be caused by human error and differences in opinion over the subjective quality of the meshes.

Training data processing

Once verified, the visual quality data collected by the volunteers underwent a small amount of pre-processing before being used to train the machine learning model: First, the data from all 1, 055 test sets were collated into a single table of 21, 100 observations. Second, the 1, 120 meshes known to be perfect were removed from the data set to leave 19, 980 observations. Third, the arithmetic mean score over five repeats was taken for each pair of reference/test meshes to give an average score for each mesh. Finally, the score data (the target variable) was merged with the metrics defined in Model design (the predictor variables) to produce the final data set. This data set contained 3, 996 observations, with 108 predictor variables and a target variable for each observation.

Model training procedure

The training of the random forest model was performed according to Algorithm 2. Repeating the training process 30 times (lines 2 through 7) helps to reduce chances of over-fitting to a specific dataset by ensuring hyperparameters are chosen based on average results. Testing the approved model with a coarser train/test split (lines 14 through 21) further reduces the chance of overfitting by training the model against a smaller data set. Each iteration of the training process takes approximately five minutes, excluding the time taken to manually assess and adjust the hyperparameters.

Algorithm 2 Random Forest Training Methodology

```
for $i$ in range 1 to 30 do
    Split data randomly into train/test sets with an $80 : 20$ ratio
    Train the model to fit the training data
    Test the model against the test data
    Return the confusion matrix for the model
end for
Find the mean for all confusion matrix cells
Find the standard deviation for all confusion matrix cells
Assess model quality based on the confusion matrix
if Quality is inadequate then
    Go to step 1
end if
Split data randomly into train/test sets with a $50 : 50$ ratio
Train the model to fit the training data
Test the model against the test data
Return the confusion matrix of the model
Assess model quality based on the confusion matrix
if Quality is inadequate then
    Go to step 1
end ifreturn Final model hyperparameters
```

Two separate training processes were undertaken during this research, with two distinct goals: The first process aimed simply to produce a model with the highest accuracy. In this case, accuracy is defined as the fraction of test meshes which were correctly classified. Accuracy A is defined in (9), where $N_{i,j}$ refers to the number of observations of meshes with predicted quality i and observed quality j. The subscripts R, B, G and P refer to the “ruined”, “bad”, “good” and “perfect” classes.

In the second process, the model was biased to minimise the number of ruined/bad meshes reported as good/perfect. This method was inspired by the intended use of the model in controlling an automated mesh decimation process. In this case, the reporting of ruined/bad meshes as good/perfect is undesirable, resulting in the production of meshes of poor quality. Conversely, the reporting of good/perfect meshes as bad/ruined simply results in a mesh being rejected in favour of another mesh (likely with a higher polygon count). We define the metric of generalised precision GP in (10) to measure the performance of the model towards this aim,

$$A = \frac{N_{R,R} + N_{B,B} + N_{G,G} + N_{P,P}}{\sum_{i,j} N_{i,j}}, \tag{9}$$

and

$$GP = \frac{N_{G,G} + N_{G,P} + N_{P,G} + N_{P,P}}{\sum_{i=G,P} N_{i,j}}. \tag{10}$$

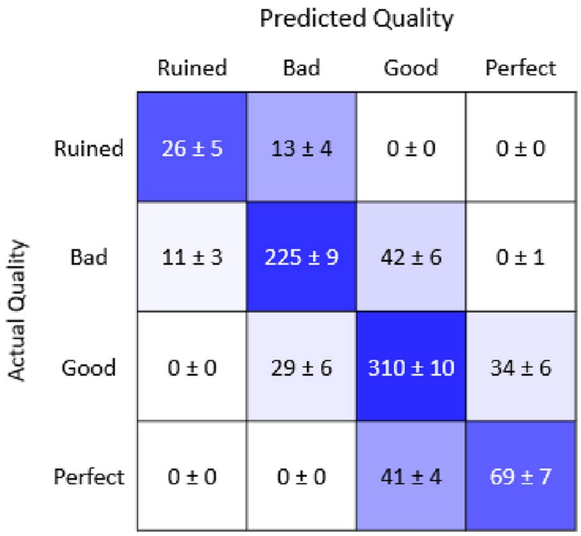
Results

In this section, we present the results obtained from our experimentation. This includes an analysis on the accuracy of our VQA model, the influence of several metrics and the efficiency of our approach.

Visual assessment results

The training process given in Algorithm 2 was performed to select an appropriate set of model hyperparameters. Figure 8 shows the results obtained using overall accuracy as the target. Results are given as the mean and standard deviation of model count after 30 trials for data with an 80 : 20 training/test split.

As shown in Fig. 8, the final model gives a reasonable prediction of the overall class into which each model was scored. The overall accuracy is found as 78.8 ± 3.9%. Note that in all 800 tests however, the predicted class is always within 1 step of the actual class. The generalised precision of this model is found as 91.5 ± 1.5%. Given the intended use of the model to control an automated decimation process, these results are promising. As mentioned in Model training procedure, this use case led to a training process aimed to improve general precision rather than overall accuracy. The results of this model are shown in Fig. 9.



Predicted Quality

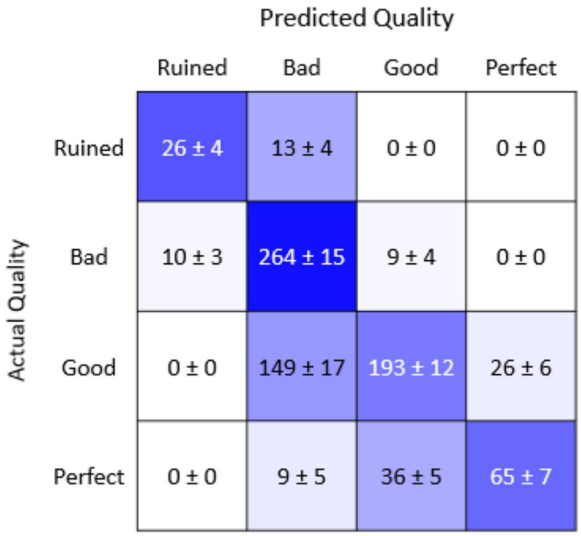
Actual Quality	Ruined	Bad	Good	Perfect
Ruined	26 ± 5	13 ± 4	0 ± 0	0 ± 0
Bad	11 ± 3	225 ± 9	42 ± 6	0 ± 1
Good	0 ± 0	29 ± 6	310 ± 10	34 ± 6
Perfect	0 ± 0	0 ± 0	41 ± 4	69 ± 7

Fig. 8 Model results for 800 test meshes, with the model trained to maximise accuracy

As shown in Fig. 9, the retrained model substantially reduces the number of ruined/bad models reported as good/perfect, at the expense of reducing the overall accuracy of the model. This behaviour was achieved by weighting ruined/bad observations in the training process at a rate of 50 : 1 compared to good/perfect observations, causing the model to reduce the likelihood of predicting the good/perfect class for a given data point. With the 50 : 1 weighting, we achieve an overall accuracy of 68.5 ± 5.3%, and a generalised precision of 97.3 ± 1.3%. Despite the drop in accuracy, the increased precision is considered to be highly advantageous for the intended use of the model. Additional gains in precision may be obtained by further increasing the weighting ratio, however the reduction in accuracy this causes makes this approach unsuitable. As such, the authors suggest that a 50 : 1 weighting is appropriate for the intended use of this model.

As shown in Figs. 8 and 9, our model performs well on a random sample of meshes from the training set. In order to further evaluate model performance, we have also tested the classifier against a number of unseen meshes. Figure 10 shows the meshes used for this test. These meshes were chosen to give representative samples of mechanical parts like those in the training set (spark plug, crankshaft) as well as an example non-mechanical part (head). 108 test meshes were produced for each of these models and their visual quality manually inspected using the process detailed in Training data collection. The model was then used to predict the quality of each test mesh without any of these meshes being included in the training data set. The results of these verification tests are given in Table 4.

As seen in Table 4, the accuracy of the visual quality assessment process varies considerably for unseen models. While the spark plug results are of similar accuracy to the test data, accuracy is notably reduced for the crankshaft and head meshes. The higher accuracy of the spark plug is to be expected given the



Predicted Quality

Actual Quality	Ruined	Bad	Good	Perfect
Ruined	26 ± 4	13 ± 4	0 ± 0	0 ± 0
Bad	10 ± 3	264 ± 15	9 ± 4	0 ± 0
Good	0 ± 0	149 ± 17	193 ± 12	26 ± 6
Perfect	0 ± 0	9 ± 5	36 ± 5	65 ± 7

Fig. 9 Model results for 800 test meshes, with the model trained to maximise generalised precision

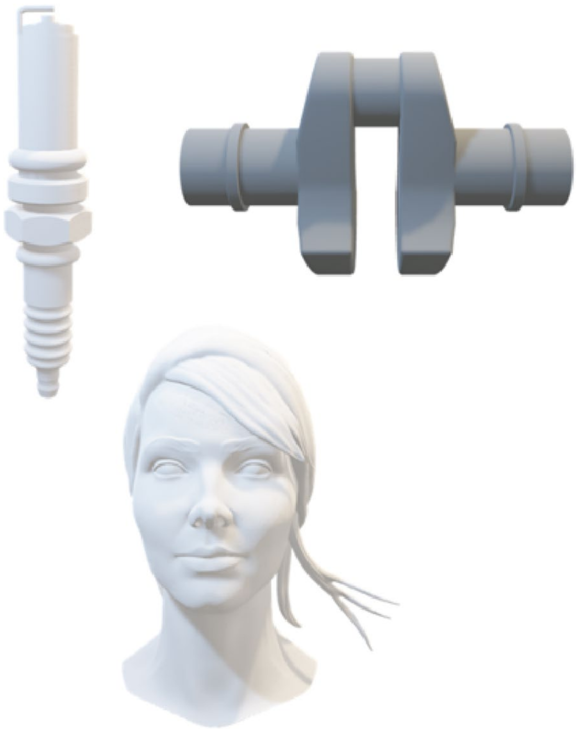


Fig. 10 Meshes used to verify VQA model performance

Table 4 Results of verification against unseen meshes. ∗ General precision for the weighted head test is undefined, as the model predicted that all 108 test meshes were of bad quality

Mesh	Balanced Model		Weighted Model	
	A(%)	*GP*(%)	*A*(%)	*GP*(%)
Spark plug	81.5	82.5	80.6	100.0
Crankshaft	45.4	100.0	38.9	100.0
Head	36.1	95.8	18.5	*N*/*A*∗

similarity of this object to those used in training. Similarly, the low accuracy of the head is to be expected given its lack of similarity to these models. The results for the crankshaft are concerning however, given the relative similarity of this mesh to the training data. This lack of accuracy suggests that our model may be over-fitting the training data. We intend to tackle this issue by considerably increasing the size of the training data set, however this approach is hindered by the time-consuming nature of the training data collection process and the lack of available data sets of this type. Unlike accuracy, the precision of the process against the unseen meshes is seen to be high, especially for the weighted model. Given the intended end use of the process, this is promising. It is of note that this increased precision is mainly driven by an increase in the number of good/perfect meshes predicted as bad/ruined. In particular, the head model predicted all 108 test meshes as bad/ruined, whereas 67 of these meshes were reported as good/perfect by human judges. This imbalance could be redressed through reducing the weighting factor of the model, however this was not performed as high precision is seen as important for the model's intended use in controlling an automated decimation process.

The results given above show promise for the use of machine learning in visual quality assessment. While the overall accuracy of the model is questionable for some meshes, the precision is more than adequate for the model's use in controlling decimation processes. Improvements to model accuracy primarily depend on the need for a greater quantity of training data. Further suggestions for model improvements are given in Conclusions.

Influence of metrics

As mentioned in Model design, the VQA model uses 108 measured variables to predict mesh quality. Every decision tree in the random forest uses a random subset of these variables and is trained on a random subset of the training data. The variable used at each decision node within a tree is chosen to give the greatest possible separation of results between the four assessment classes. Each tree therefore contains a different set of variables, with each variable having a different level of influence on the model. The distribution of variables is further affected by whether the model is trained for accuracy or precision.

Tables 5 and 6 list the overall influence of the top 20 variables in the balanced and weighted models respectively. Influence is given as the percentage of all decisions in the model which are based on this variable, and is averaged across 30 trials.

Both Tables 5 and 6 show that model results are based on many variables, with no single metric dominating the prediction. In the balanced and weighted models, the top 20 variables respectively account for 48.5% and 56.6% of the total prediction. This distribution of influences clearly shows the benefit of using multiple metrics for visual quality assessment, as well as the power of machine learning to meaningfully combine these metrics.

The top metrics used by both models are a mixture of shape ratios, similarity metrics and shape metrics of the test mesh, showing that all three types of metric contribute greatly to the overall assessment of visual quality. Furthermore, the balanced and weighted models respectively depend on 12 and 8 metrics based on correspondence. As such, this shows the importance of solving the correspondence problem in visual quality assessment.

Table 5 Influence of metrics in the balanced VQA model

Metric	Influence (%)
Surface area ratio	7.48 ± 0.45
Normal deviation, 1.0, mean	4.45 ± 0.40
Matching distance, 1.0, mean	4.33 ± 0.41
Normal deviation, 1.5, mean	3.05 ± 0.40
Matching distance, 1.5, mean	2.88 ± 0.35
Matching distance, 2.0, mean	2.41 ± 0.25
Polygon area COV ratio	2.04 ± 0.21
Convexity ratio	2.03 ± 0.10
Matching distance, 2.5, mean	2.00 ± 0.20
Test mesh bounding box density	1.97 ± 0.10
Sphericity ratio	1.92 ± 0.19
Shape efficiency ratio	1.86 ± 0.09
Normal deviation, 1.0, minimum	1.85 ± 0.17
Normal deviation, 2.0, mean	1.73 ± 0.25
Matching distance, 3.0, mean	1.49 ± 0.15
Dihedral angle change, 1.0, COV	1.42 ± 0.21
Normal deviation, 2.5, mean	1.41 ± 0.06
Polygon area skewness ratio	1.36 ± 0.16

Table 6 Influence of metrics in the weighted VQA model

Metric	Influence (%)
Normal deviation, 1.0, mean	7.01 ± 1.11
Surface area ratio	5.47 ± 1.02
Normal deviation, 1.5, mean	5.14 ± 0.70
Convexity ratio	4.89 ± 0.72
Sphericity ratio	4.57 ± 0.93
Shape efficiency ratio	4.29 ± 0.54
Volume ratio	4.02 ± 0.63
Normal deviation, 1.0, minimum	2.76 ± 0.61
Normal deviation, 1.5, kurtosis	1.96 ± 0.67
Connectivity ratio	1.88 ± 0.30
Dihedral angle change, 1.0, mean	1.79 ± 0.47
Normal deviation, 2.0, mean	1.70 ± 0.48
Polygon area COV ratio	1.67 ± 0.30
Test mesh connectivity	1.57 ± 0.23
Normal deviation, 2.0, kurtosis	1.41 ± 0.46
Test mesh convexity	1.34 ± 0.16
Test mesh bounding box density	1.34 ± 0.16
Normal deviation, 1.0, kurtosis	1.32 ± 0.45
Test mesh polygon area COV	1.25 ± 0.18
Test mesh minor squareness	1.20 ± 0.20

The differences in metric usage between the two models is minor, however it is of note that the balanced model depends more heavily on similarity metrics, while the weighted model contains more shape metrics for the test mesh. It is therefore suggested that similarity metrics, which depend on observable differences between the reference and test mesh, are better predictors of how well the test mesh approximates the reference mesh. Shape metrics, which depend only on the test mesh, appear to be better predictors of mesh quality regardless of the method in which the mesh was produced.

Processing time

In addition to the accuracy of visual quality assessment, we also consider the performance of our model in terms of the run-time required to analyse a given mesh. As shown in Fig. 11, run-time is strongly dependent upon the number of polygons in the reference mesh. The variance in run-time seen at a given reference polygon count is believed to be driven mainly by the polygon count of the test mesh.

As seen in Fig. 11, the model can return results for reference meshes of up to approximately 25, 000 polygons in around 2 minutes or less. This is a reasonable run-time given the complexity of the calculation, but this could be improved. As shown, run-time appears to have a quadratic relationship with reference polygon count. In some cases, the conversion of CAD models to VR may involve the assessment of reference meshes with considerably higher polygon counts than those shown. As such, a quadratically increasing run-time could prove problematic given the intended end use of this process.

There are several methods through which model performance could be improved. Firstly, note that many of the metrics derived are based around a single reference vertex. This independence on other vertices can be exploited through parallelisation of the process. Splitting

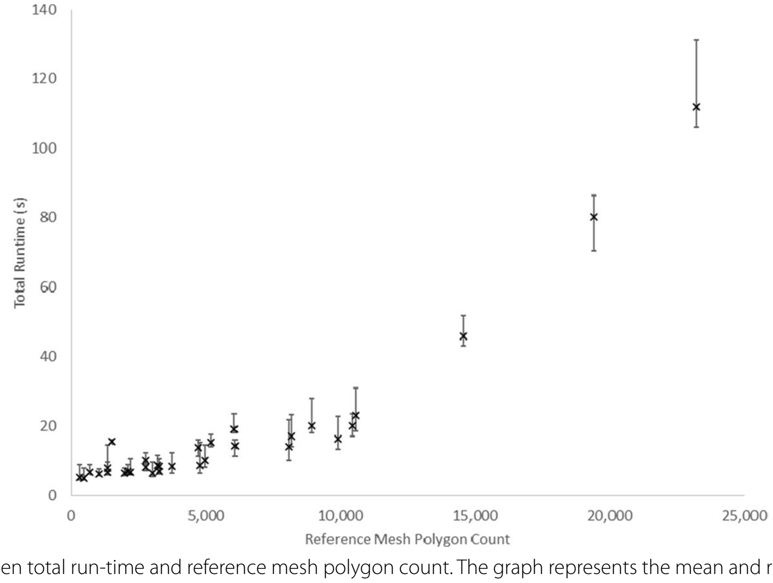


Fig. 11 Relationship between total run-time and reference mesh polygon count. The graph represents the mean and range of run-times for each reference mesh

reference mesh data across multiple CPU or GPU cores could greatly decrease run-time. Second, as shown in Fig. 12, shape data extraction is the primary component of run-time at high polygon counts. More efficient coding of the algorithms used in this step would therefore be expected to further decrease run-time. In particular, this step depends heavily on the use of the Blender 3D modelling software to extract data. Developing a system to extract this data without the need to load a heavy 3D modelling package would likely decrease run-times significantly. Work is currently ongoing to both parallelise key aspects of the algorithm and reduce the dependence on Blender.

Conclusions

This paper introduces a new full-reference machine learning model for the visual quality assessment of decimated 3D meshes in digital-twins. Over one hundred shape factors and perceptual similarity metrics are extracted from a pair of meshes (reference and test) and combined using a random forest classifier to determine how well the test mesh visually represents the reference mesh. The results of this assessment are intended to be used to control automated decimation processes, producing visually accurate low polygon meshes from CAD data for use in AR and VR applications.

Our focus on the assessment of decimated meshes differs from much of the existing VQA literature. Decimation is an important process in many commercial 3D modelling applications, however it is greatly under-represented in the literature. By its nature, decimation alters the connectivity of a mesh, complicating VQA processes. The majority of VQA literature instead focuses on methods such as watermarking or compression, in which the connectivity of the mesh is unchanged by the distortion process. The development of a method capable of handling this issue thus provides a means by which any two meshes can be compared regardless of the distortion processes involved. As such, our method can be applied to a much greater number of cases than many existing methods.

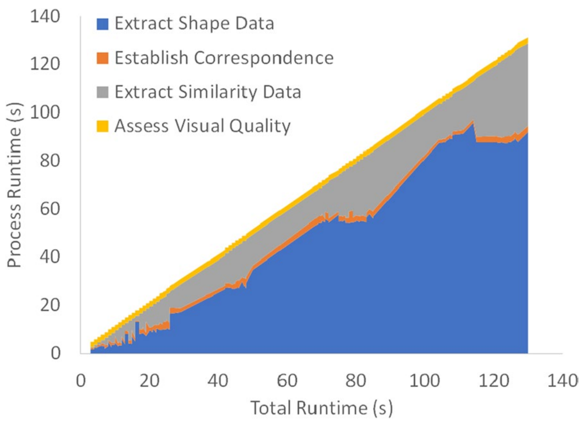


Fig. 12 Contribution of individual process stages to total run-time

A particular challenge in developing our method also stems from the lack of literature in this area: As well as few published papers, there are very little publicly available training data for use in developing assessment models. In our paper, we use data collected from several volunteers using a specially designed application. Our model is trained on nearly a set of nearly 4, 000 test meshes based on 37 different mechanical objects. While the total number of meshes is considerably greater than most models in the literature, the relatively small number of objects reduces the applicability of the model to many shapes, and may lead to over-fitting. The results of our tests show that our model can classify visual quality to a good degree of accuracy. On models similar to those in the training set, the model is seen to have a precision of over 97% when weighted to prioritise the rejection of poor quality models. Considerably reduced accuracy is seen for models different from those in the training set however, suggesting that the model may be over-fitting. As mentioned above, we believe this over-fitting to be primarily due to a lack of available training data. As such, and given the application of decimation in a number of commercial processes, the authors suggest that the publication of representative training data sets should be a key priority for the research community.

Despite the above issue, we believe our model is precise enough to classify the results of decimation processes. Performance results show that complex meshes can be analysed in approximately two minutes or less on a standalone PC, with the potential for much faster operation to be achieved through parallelisation. This combination of accuracy and speed suggests that our method is more than capable of controlling automated mesh decimation processes. As such, we are intending to integrate our model within our developing CAD to VR processor. This integration will be discussed in a later publication.

As well as integrating the VQA model with mesh conversion methods, further research will also consider increasing the speed of our VQA model through parallelisation on both CPU and GPU architectures. In addition, training of the model on a wider range of objects will be performed to increase accuracy and reduce over-fitting. We believe that with these additions, our model can provide a fast and powerful method for visual quality assessment on decimated meshes and will perform a key role in future automated mesh processing systems.

Acknowledgements

Code availability
Not applicable.

Authors' contributions
B.R and F.M. performed the analysis and wrote the initial manuscript. C.B. compiled the work into the final submission. A.A and L.L. provided support and guidance throughout the process. All authors reviewed the manuscript.

Funding
Not applicable.

Availability of data and materials
Not applicable.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no competing interests.

Received: 24 October 2023 Accepted: 18 February 2024
Published online: 26 February 2024

References

1. Zaman M (2022) Effects of Virtual Hands on Physical Demands and Task Performance for Typing in Virtual Reality. In: Proceedings of the XXXIVth Annual International Occupational Ergonomics and Safety Conference. Presented at the XXXIVth Annual International Occupational Ergonomics and Safety Conference, International Society for Occupational Ergonomics and Safety, p 47–53. https://doi.org/10.47461/isoes.2022_zaman
2. Delshad SD, Almario CV, Fuller G, Luong D, Spiegel BM (2018) Economic analysis of implementing virtual reality therapy for pain among hospitalized patients. NPJ Digit Med 1(1):22
3. Clus D, Larsen ME, Lemey C, Berrouiguet S (2018) The use of virtual reality in patients with eating disorders: systematic review. J Med Internet Res 20(4):e157
4. Martin P, Masfrand S, Okuya Y, Bourdot P (2017) A VR-CAD Data Model for Immersive Design: The cRea-VR Proof of Concept. In: Augmented Reality, Virtual Reality, and Computer Graphics: 4th International Conference, AVR 2017, Proceedings, Part I 4. Springer International Publishing, Ugento, p 222–241
5. Wolfartsberger J, Zenisek J, Sievi C, Silmbroth M (2017) A virtual reality supported 3D environment for engineering design review. In: 2017 23rd International Conference on Virtual System & Multimedia (VSMM). IEEE, p 1–8
6. Chi HL, Kang SC, Wang X (2013) Research trends and opportunities of augmented reality applications in architecture, engineering, and construction. Autom Constr 33:116–122
7. Portman ME, Natapov A, Fisher-Gewirtzman D (2015) To go where no man has gone before: Virtual reality in architecture, landscape architecture and environmental planning. Comput Environ Urban Syst 54:376–384
8. Riar M, Korbel JJ, Xi N, Zarnekow R, Hamari J (2021) The Use of Augmented Reality in Retail: A Review of Literature. In: Hawaii International Conference on System Sciences. p 638–647
9. Pucihar KČ, Coulton P (2015) Exploring the evolution of mobile augmented reality for future entertainment systems. Comput Entertainment (CIE) 11(2):1–16
10. Westerfield G, Mitrovic A, Billinghurst M (2015) Intelligent augmented reality training for motherboard assembly. Int J Artif Intell Educ 25:157–172
11. Mokmin NAM, Rassy RP (2024) Review of the trends in the use of augmented reality technology for students with disabilities when learning physical education. Educ Inf Technol 29(2):1251–1277
12. Fujii Y, Kio OG, Au D, Wilcox LM, Allison RS (2019) Effects of frame rate on vection and postural sway. Displays 58:33–43
13. Lorenz M, Spranger M, Riedel T, Pürzel F, Wittstock V, Klimant P (2016) Cad to vr-a methodology for the automated conversion of kinematic cad models to virtual reality. Procedia Cirp 41:358–363
14. Roach DM, Demirkiran I (2017) Computer aided drafting virtual reality interface. In: 2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC). IEEE, p 1–13
15. Abidi MH, Al-Ahmari A, Ahmad A (2018) A systematic approach to parameter selection for cad-virtual reality data translation using response surface methodology and moga-ii. PLoS ONE 13(5):e0197673
16. Cimolin V, Paraskevopoulos IT, Sala M, Tarabini M, Galli M (2023) The smart body concept as a demonstration of the overarching utility and benefits of 3D avatars in retail, health and wellbeing: an accuracy study of body measures from 3D reconstruction. Multimedia Tools Appl 82(7):11079–11098
17. Raposo A, Corseuil ET, Wagner GN, dos Santos IH, Gattass M (2006) Towards the use of cad models in VR applications. In: Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications. p 67–74
18. Tang Y, Gu H (2010) CAD model's Simplification and Conversion for Virtual Reality. In: 2010 third international conference on information and computing, vol 4. IEEE, p 265–268
19. Hoppe H (1997) View-Dependent Refinements of Progressive Meshes. In: Computer Graphics (SIGGRAPH'97 Proc.). p 189–198
20. Guthe M, Borodin P, Klein R (2005) Fast and accurate Hausdorff distance calculation between meshes. J WSCG 13(2):41–48
21. Melax S (1998) A simple, fast, and effective polygon reduction algorithm. Game Developer 11(Nov):44–49
22. Polygon Cruncher (2023) Polygon Cruncher (Version 10) [Software]. Mootools. Available at: https://www.mootools.com/products/polygon-cruncher. Accessed 9 Mar 202
23. Corsini M, Larabi MC, Lavoué G, Petřík O, Váša L, Wang K (2013) Perceptual metrics for static and dynamic triangle meshes. In: Computer graphics forum, vol 32, No. 1. Blackwell Publishing Ltd., Oxford, p 101–125
24. Lavoué G, Corsini M (2010) A comparison of perceptually-based metrics for objective evaluation of geometry processing. IEEE Trans Multimed 12(7):636–649
25. Van Kaick O, Zhang H, Hamarneh G, Cohen-Or D (2011) A survey on shape correspondence. In: Computer graphics forum, vol 30, No. 6. Blackwell Publishing Ltd., Oxford, p 1681–1707
26. Cignoni P, Rocchini C, Scopigno R (1998) Metro: Measuring error on simplified surfaces. Comput Graph Forum 17(2):167–174. https://doi.org/10.1111/1467-8659.00236
27. Muzahid AM, Wan W, Feng X (2018) Perceptual quality evaluation of 3d triangle mesh: a technical review. In: 2018 International Conference on Audio, Language and Image Processing (ICALIP). IEEE, p 266–272
28. Wysocki M, Doyle S (2022) Optimization of decimation protocols for advancing the validity of 3D model data. In: Medical Imaging 2022: Physics of Medical Imaging, vol 12031. SPIE, p 971–975
29. Abouelaziz I, Chetouani A, El Hassouni M, Latecki LJ, Cherifi H (2020) No-reference mesh visual quality assessment via ensemble of convolutional neural networks and compact multi-linear pooling. Pattern Recog 100:107174
30. Váša L, Rus J (2012) Dihedral angle mesh error: a fast perception correlated distortion measure for fixed connectivity triangle meshes. In: Computer graphics forum, vol 31, No. 5. Blackwell Publishing Ltd., Oxford, p 1715–1724
31. Wang Zhou, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612. https://doi.org/10.1109/TIP.2003.819861
32. Daly SJ (1992) Visible differences predictor: an algorithm for the assessment of image fidelity. In: Rogowitz BE (ed) Human Vision, Visual Processing, and Digital Display III, International Society for Optics and Photonics, vol 1666. SPIE, pp 2 – 15. https://doi.org/10.1117/12.135952
33. Rogowitz BE, Rushmeier HE (2001) Are image quality metrics adequate to evaluate the quality of geometric objects?. In: Human Vision and Electronic Imaging VI, vol 4299. SPIE, p 340–348

34. Yildiz Z, Oztireli A (2018) A machine learning framework for full-reference 3d shape quality assessment. Vis Comput 36(1):127–139. https://doi.org/10.1007/s00371-018-1592-9
35. Chetouani A (2017) A 3d mesh quality metric based on features fusion. Electron Imaging 29:4–8
36. Lavoué G, Gelasca ED, Dupont F, Baskurt A, Ebrahimi T (2006) Perceptually driven 3D distance metrics with application to watermarking. In: Tescher AG (ed) Applications of Digital Image Processing XXIX, International Society for Optics and Photonics, vol 6312. SPIE, pp 150–161. https://doi.org/10.1117/12.686964
37. Li J, Chen D, Hu F, Wang Y, Li P, Peethambaran J (2024) Shape-preserving mesh decimation for 3d building modeling. Int J Appl Earth Observation Geoinformation 126:103623
38. Gautron P, Kubisch C (2023) Interactive GPU-based Decimation of Large Meshes. In: ACM SIGGRAPH 2023 Talks. p 1–2
39. Shapiro MD, Blaschko MB (2004) On hausdorff distance measures. Computer Vision Laboratory University of Massachusetts, Amherst
40. Taha AA, Hanbury A (2015) An efficient algorithm for calculating the exact hausdorff distance. IEEE Trans Patt Anal Mach Intell 37(11):2153–2163
41. Zhang D, He F, Han S, Zou L, Wu Y, Chen Y (2017) An efficient approach to directly compute the exact hausdorff distance for 3d point sets. Integr Comput Aided Eng 24:261–277. https://doi.org/10.3233/ICA-170544
42. Lavoué G, Cheng I, Basu A (2013) Perceptual quality metrics for 3d meshes: Towards an optimal multi-attribute computational model. In: 2013 IEEE International Conference on Systems, Man, and Cybernetics. pp 3271–3276. https://doi.org/10.1109/SMC.2013.557
43. Wang K, Torkhani F, Montanvert A (2012) A fast roughness-based approach to the assessment of 3d mesh visual quality. Comput Graph 36(7):808–818. https://doi.org/10.1016/j.cag.2012.06.004. Augmented Reality Computer Graphics in China
44. Torkhani F, Chassery JM, Wang K (2014) A curvature-tensor-based perceptual quality metric for 3d triangular meshes. Mach Graph Vis 23(1/2):59–82
45. Yildiz ZC, Çapin TK (2019) A Fully Object-space Approach for Full-reference Visual Quality Assessment of Static and Animated 3D Meshes. In: VISIGRAPP (1: GRAPP). p 169–176
46. Nouri A, Charrier C, Lézoray O (2016) Full-reference saliency-based 3d mesh quality assessment index. In: 2016 IEEE International Conference on Image Processing (ICIP). IEEE, pp 1007–1011
47. Lavoué G (2011) A multiscale metric for 3d mesh visual quality assessment. Comput Graph Forum 30(5):1427–1437. https://doi.org/10.1111/j.1467-8659.2011.02017.x
48. Lee CH, Varshney A, Jacobs DW (2005) Mesh saliency. ACM Trans Graph 24(3):659–666. https://doi.org/10.1145/1073204.1073244
49. Roy M, Foufou S, Truchetet F (2004) Mesh comparison using attribute deviation metric. Int J Image Graphics 4:127. https://doi.org/10.1142/S0219467804001324
50. Gatzke T, Grimm C (2006) Estimating curvature on triangular meshes. Int J Shape Model 12:1–28. https://doi.org/10.1142/S0218654306000810
51. Rusinkiewicz S (2004) Estimating curvatures and their derivatives on triangle meshes. In: Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. pp 486–493. https://doi.org/10.1109/TDPVT.2004.1335277
52. Feng X, Wan W, Da Xu RY, Perry S, Li P, Zhu S (2018) A novel spatial pooling method for 3d mesh quality assessment based on percentile weighting strategy. Comput Graph 74:12–22
53. Abouelaziz I, El Hassouni M, Cherifi H (2018) Blind 3d mesh visual quality assessment using support vector regression. Multimed Tools Appl 77(18):365–386
54. Frintrop S, Rome E, Christensen HI (2010) Computational visual attention systems and their cognitive foundations: A survey. ACM Trans Appl Percept 7(1). https://doi.org/10.1145/1658349.1658355
55. Guezlec A (2001) "Meshsweeper": dynamic point-to-polygonal mesh distance and applications. IEEE Trans Vis Comput Graph 7(1):47–61
56. Jones MW, Baerentzen JA, Sramek M (2006) 3d distance fields: a survey of techniques and applications. IEEE Trans Vis Comput Graph 12(4):581–599. https://doi.org/10.1109/TVCG.2006.56
57. He Y, Liang B, Yang J, Li S, He J (2017) An iterative closest points algorithm for registration of 3d laser scanner point clouds with geometric features. Sensors 17(8):1862
58. Mystakidis S (2022) Metaverse. Encyclopedia 2(1):486–497
59. Iqbal, M.Z. and Campbell, A.G., 2023, October. Metaverse as tech for good: Current progress and emerging opportunities. In Virtual Worlds (Vol. 2, No. 4). MDPI.
60. Alja'Afreh M (2021) A qoe model for digital twin systems in the era of the tactile internet. PhD thesis, Université d'Ottawa/University of Ottawa
61. Fritsch D, Wagner J, Ceranski B, Simon S, Niklaus M, Zhan K, Mammadov G (2021) Making historical gyroscopes alive-2d and 3d preservations by sensor fusion and open data access. Sensors 21:957
62. Tran H, Nguyen TN, Christopher P, Bui DK, Khoshelham K, Ngo TD (2021) A digital twin approach for geometric quality assessment of as-built prefabricated façades. J Build Eng 41:102377
63. Martínez-Gutiérrez A, Díez-González J, Verde P, Perez H (2023) Convergence of virtual reality and digital twin technologies to enhance digital operators' training in industry 4.0. Int J Hum Comput Stud 180:103136
64. Testolina M, Lazzarotto D, Rodrigues R, Mohammadi S, Ascenso J, Pinheiro AM, Ebrahimi T (2023) On the performance of subjective visual quality assessment protocols for nearly visually lossless image compression. In: Proceedings of the 31st ACM International Conference on Multimedia. p 6715–6723
65. Nouri A, Charrier C, Lézoray O (2017) 3D Blind Mesh Quality Assessment Index. Electron Imaging 29:9–26
66. Bauszat P, Eisemann M, Magnor MA (2010) The Minimal Bounding Volume Hierarchy. In: VMV. p 227–234
67. Zhang C, Chen T (2001) Efficient feature extraction for 2D/3D objects in mesh representation. In: Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205), vol 3. IEEE, p 935–938
68. Maneewongvatana S, Mount DM (1999) Analysis of approximate nearest neighbor searching with clustered point sets. arXiv preprint cs/9901013

Publisher's Note