

PERCEPTUALLY-DRIVEN COMPUTER GRAPHICS AND VISUALIZATION

A DISSERTATION SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

By
Zeynep Çipiloğlu Yıldız
October 2016

Perceptually-driven Computer Graphics and Visualization

By Zeynep ipilođlu Yıldız

October 2016

We certify that we have read this dissertation and that in our opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Halil Bülent Özgüç (Advisor)

Tolga Kurtuluş apın (Co-advisor)

Uđur Güdükbay

Hüseyin Boyacı

Haşmet Gürçay

Ahmet Ođuz Akyüz

Approved for the Graduate School of Engineering and Science:

Ezhan Karaşan
Director of the Graduate School

ABSTRACT

PERCEPTUALLY-DRIVEN COMPUTER GRAPHICS AND VISUALIZATION

Zeynep Çipiloğlu Yıldız

Ph.D. in Computer Engineering

Advisors: Halil Bülent Özgüç and Tolga Kurtuluş Çapın

October 2016

Despite the rapid advances in computer graphics technology, enhancing the visual quality and lowering the rendering cost is still the essential goal for computer graphics researchers; since improvements in computational power raise the users' expectations too. Especially in interactive 3D games and cinema industry, very realistic graphical contents are desired in real-time. In the meantime, due to the increasing popularity of social networking systems and data sharing, there is a huge amount of data to be visualized effectively. When used carefully, 3D introduces a new data channel for information visualization applications. For that reason, improving the visual quality of 3D computer-generated scenes is still of great interest in the computer graphics and visualization community.

In the last decade, utilization of visual perception findings in computer graphics has started to get popular since visual quality is actually judged by the human perception and there is no need to spend additional cost for the physical realism of the details that cannot be perceived by the observer. There is still room for employing the perceptual principles in computer graphics.

We contribute to the perceptual computer graphics research in two main aspects: First we propose several perceptual error metrics for evaluating the visual quality of static or animated 3D meshes. Second, we develop a system for ameliorating the perceived depth quality and comprehensibility in 3D visualization applications.

A measure for assessing the quality of a 3D mesh is necessary in order to determine whether an operation on the mesh, such as watermarking or compression, affects the perceived quality. The studies on this field are limited when compared to the studies for 2D. A bottom-up approach incorporating both the spatial and temporal components of the low-level human visual system processes is suggested to develop a general-purpose quality metric designed to measure the local distortion visibility on dynamic triangle meshes. In addition, application of crowdsourcing and machine learning methods to implement a novel data-driven

error metric for 3D models is also demonstrated.

During the visualization of 3D content, using the depth cues selectively to support the design goals and enabling a user to perceive the spatial relationships between the objects are important concerns. In this regard, a framework for selecting proper depth cues and rendering methods providing these cues for the given scene and visualization task is put forward. This framework benefits from fuzzy logic for determining the importance of depth cues and knapsack method for modeling the cost-profit tradeoff between the rendering costs of the methods and their contribution to depth perception.

All the proposed methods in this study are validated through formal user experiments and we obtain encouraging results for further research. These results are made publicly available for the benefit of graphics community. In conclusion, we try to make the gap between visual perception and computer graphics fields narrower with the suggested methods in this work.

Keywords: computer graphics, visual perception, visual quality assessment, 3D mesh quality, depth perception, perceptually-based graphics.

ÖZET

GÖRSEL ALGI ODAKLI BİLGİSAYAR GRAFİKLERİ VE GÖRSELLEŞTİRME

Zeynep Çipilođlu Yıldız

Bilgisayar Mühendisliđi, Doktora

Tez Danışmanları: Bülent Özgüç ve Tolga Çapın

Ekim 2016

Bilgisayar grafikleri teknolojisindeki hızlı ilerlemelere rağmen, görsel kaliteyi geliřtirmek ve görüntüleme maliyetini düşürmek halen arařtırmacıların temel hedefidir; çünkü bilgisayar gücü yükseldikçe kullanıcıların beklentileri de yükselmektedir. Özellikle interaktif 3B oyun ve sinema endüstrisinde gerçekçi içerikler gerçek zamanlı olarak talep edilmektedir. Bunun yanında, sosyal ağların ve veri paylaşımının popülerleşmesine bađlı olarak, büyük miktarda verinin verimli bir şekilde görselleřtirilmesine gereksinim vardır. Dikkatli bir şekilde kullanıldığında, bilgi görselleřtirmesi uygulamaları için üçüncü boyut yeni bir veri kanalı sağlamaktadır. Tüm bu nedenlerden dolayı, 3B sahnelerin görsel kalitesinin artırılması bilgisayar grafikleri ve görselleřtirme topluluđunun ilgisi dahilindedir.

Son yıllarda, görsel algı alanındaki bulguların bilgisayar grafiklerinde kullanılması yaygınlaşmaya başlamıştır; çünkü aslında görsel kalite insan algısı tarafından ölçülmektedir ve algılanamayacak detayların fiziksel gerçekliđe uygun olması için fazladan maliyete gerek yoktur. Halen algısal prensiplerin bilgisayar grafiklerinde uygulanması alanında arařtırmaya ihtiyaç vardır.

Bu çalıřma ile algıya dayalı bilgisayar grafikleri alanına iki temel katkı sağlamaktayız: İlk olarak, statik ve dinamik 3B modellerin görsel kalitesinin deđerlendirilmesi amacıyla birkaç tane algısal hata ölçütü önermekteyiz. İkinci olarak, 3B görselleřtirme uygulamalarında algılanan derinlik kalitesini artırmayı amaçlayan bir sistem geliřtiriyoruz.

3B model üzerinde uygulanan sayısal damgalama veya sıkıřtırma gibi bir operasyonun algılanan kaliteyi nasıl etkilediđini deđerlendirmek için bir ölçüte ihtiyaç vardır. Bu alandaki çalıřmalar 2B üzerindeki çalıřmalara kıyasla çok limitlidir. Bu çalıřmada, dinamik 3B modeller üzerinde bölgesel distorsiyonların görünürlüđünü ölçen genel amaçlı bir kalite ölçütü geliřtirmek için insan görme sisteminin uzamsal ve zamansal bileşenlerini içeren ařađıdan yukarıya bir yaklařım önerilmektedir. Buna ek olarak, yine 3B modeller için yeni bir veri

tabanlı kalite ölçütü geliřtirmek amacıyla makine öğrenmesi ve kitle kaynaklı çalışma kullanımı örneklendirilmektedir.

3B içeriğinin görselleřtirilmesi sırasında, derinlik ipuçlarının tasarım amaçlarını destekleyecek şekilde seçilmesi ve kullanıcının sahnedeki nesnelere arasındaki uzamsal ilişkiyi algılamasının kolaylaştırılması önemli bir meseledir. Bu doğrultuda, verilen bir sahne ve görev için uygun derinlik ipuçlarını ve bu ipuçlarını sağlayan görüntüleme yöntemlerini belirleyen bir çerçeve sistem önerilmiştir. Bu sistem, derinlik ipuçlarının önemini belirlemede bulanık mantıktan, görüntüleme yöntemlerinin maliyetleri ve derinlik algısına katkıları arasındaki maliyet-fayda analizinin modellenmesinde sırt çantası (knapsack) metodundan yararlanmaktadır.

Bu tezde önerilen bütün yöntemler kullanıcı deneyleri ile doğrulanmış ve yeni arařtırmalar için teşvik edici sonuçlar elde edilmiştir. Ayrıca çalışmada kullanılan veriler ve elde edilen sonuçlar bilgisayar grafikleri topluluğunun kullanımına sunulmuştur. Sonuç olarak, bu çalışmada önerilen yöntemler ile görsel algı ve bilgisayar grafikleri disiplinleri arasındaki açıklık biraz daha daraltılmaya çalışılmıştır.

Anahtar sözcükler: bilgisayar grafikleri, görsel algı, görsel kalite ölçümü, 3B model kalitesi, derinlik algısı, algıya dayalı grafik.

Acknowledgement

First of all, I would like to express my sincere gratitude to my supervisors Tolga apın and Bülent Özgüç for their endless support, guidance, and encouragement.

I would also like to thank to my jury members Uğur Gdkbay, Hseyin Boyacı, Haşmet Grçay, and Ahmet Oğuz Akyz for spending their valuable time to read and evaluate this thesis.

I am also grateful to my friends for their friendship, patience, and contributions during the user experiments.

I thank each member of my family and special thanks to my husband Baha, for his endless support and patience during my education.

Finally, I would like to thank to ACM for permitting me to reprint my paper in my thesis and to TBTAK-BDEB for the financial support during my Ph.D study.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Scope of the Work	2
1.3	Contributions	3
1.4	Outline of the Thesis	4
2	Background	6
2.1	Visual Quality Assessment for 3D Meshes	6
2.1.1	Methods for Visual Quality Assessment	7
2.1.2	Perceptual Concepts in Visual Quality Assessment	11
2.2	Depth Perception	17
2.2.1	Depth Cues and Cue Combination Models	17
2.2.2	Depth Perception in Computer Graphics and Visualization	19
3	Visual Quality Assessment of Dynamic Meshes	24
3.1	Voxel-based Approach	25
3.1.1	Overview	25
3.1.2	Preprocessing	25
3.1.3	Perceptual Quality Evaluation	30
3.2	Mesh-based Approach	33
3.2.1	Overview	34
3.2.2	Preprocessing	34
3.2.3	Perceptual Quality Evaluation	36
3.3	Summary and Discussion	38
4	Learning the Visual Quality of Static Meshes	41

4.1	Overview	41
4.2	Data Collection through Crowdsourcing	42
4.2.1	Data	43
4.2.2	Experiment Design	44
4.2.3	Reliability Check for the Crowdsourced Data	45
4.3	Feature Extraction	46
4.4	Metric Learning	48
4.5	Summary and Discussion	50
5	Enhancing the Perceived Depth Quality	52
5.1	Overview	53
5.2	Cue Prioritization	54
5.2.1	Fuzzification	55
5.2.2	Inference	58
5.2.3	Defuzzification	58
5.3	Mapping Selected Depth Cues to Rendering Methods	61
5.3.1	Method Selection	62
5.3.2	Method Level Selection	65
5.3.3	Cue Conflict Avoidance and Resolution	66
5.3.4	Rendering Methods	67
5.4	Summary and Discussion	69
6	Evaluation	70
6.1	Visual Quality Assessment of Dynamic Meshes	70
6.1.1	User Experiment	70
6.1.2	Comparison to State-of-the-art Techniques	77
6.1.3	Performance Evaluation	81
6.1.4	Evaluation of the Mesh-based Approach	85
6.2	Learning the Visual Quality of Static Meshes	87
6.3	Perceived Depth Quality Enhancement Method	88
6.3.1	Experiment 1 - 3D Tree Visualization	91
6.3.2	Experiment 2 - 3D Graph Visualization	94
6.3.3	Experiment 3 - 3D Scatter Plot Visualization	95
6.3.4	Experiment 4 - 3D Surface Visualization	96

7 Conclusion	99
A Data	116
A.1 Supplemental Material for Animated Mesh Quality Assessment . .	116
A.2 Supplemental Material for Learning the VQA of 3D Meshes . . .	117
A.3 Supplemental Material for Perceived Depth Quality Enhancement Framework	118

List of Figures

2.1	Cortex Transform - Organization of the filter bank.	13
2.2	Cortex Transform - Decomposition of the image into radial and orientation selective channels (frequency values estimate the center frequency of each frequency band.)	14
2.3	Contrast sensitivity at various spatial frequencies.	15
2.4	Spatial and temporal Contrast Sensitivity Functions.	16
2.5	Spatiotemporal and spatiovelocity CSF.	16
2.6	Depth cues.	17
3.1	Overview of the perceptual quality evaluation for dynamic triangle meshes.	24
3.2	Method overview for the voxel-based approach.	26
3.3	Difference of Mesa (DOM) filters. (x-axis: spatial frequency in <i>cycles/pixel</i> , y-axis: response)	31
3.4	Frequency domain filtering in Cortex Transform.	32
3.5	Method overview for the mesh-based approach.	35
3.6	Pipeline for Manifold Harmonics Transform.	36
3.7	Filtering different attributes of a mesh with MHT.	38
3.8	Application of Cortex Transform on an image.	39
3.9	Output of the Channel Decomposition step for the hand mesh.	39
4.1	Method overview.	42
4.2	Meshes used in the AMT experiment.	43
4.3	Screenshot from our AMT experiment.	44
4.4	Learned weights of the feature vector. (x-axis: Index of the feature, y-axis: weight of the feature.)	50

5.1	Illustration of the proposed framework.	52
5.2	General architecture of the system.	54
5.3	Cue prioritization stage.	55
5.4	Membership functions for fuzzification.	57
5.5	Left: Membership functions for defuzzification. Right: A sample fuzzy output of the system for shadow depth cue.	58
5.6	Cue to rendering method mapping stage.	63
5.7	Method selection step.	64
6.1	Sample frames from the reference animations.	71
6.2	Experimental setup.	73
6.3	<i>Top-left</i> : reference mesh, <i>top-right</i> : modified mesh, <i>bottom-left</i> : mean subjective response, <i>bottom-right</i> : estimated visual response. Blue regions in the mean subjective response and estimated response maps demonstrate the high perceptual differences.	75
6.4	Top row: original models. Bottom row: Metric outputs. (a) Original model (b) High noise on smooth regions (MOS = 8.80, MSDM = 0.64, Our metric = 0.69) (c) High noise on the whole object (MOS = 9.40, MSDM = 0.70, Our metric = 0.85), (d) High smoothing on the whole object (MOS = 8.10, MSDM = 0.58, Our metric = 0.54).	78
6.5	Subjective MOS vs. our metric estimation for each model. Spearman correlation coefficients and trendlines are also displayed.	79
6.6	Subjective testing results vs. metric estimation.	81
6.7	Effect of the <i>minResolution</i> parameter on the mean correlation.	82
6.8	Processing time (in seconds) of one frame with respect to the <i>minResolution</i> parameter.	84
6.9	Processing time (in seconds) of one frame for several meshes.	86
6.10	Top rows: The scenes with basic cues. Bottom rows: The scenes with the automatically selected methods. (Multi-view, face tracking, keyboard-control methods cannot be shown here.) Left to right: Scene Complexity Level 1, Level 2, Level 3.	92
6.11	Experiment results.	93
6.12	Test scenes for 3D surface experiment.	97

List of Tables

2.1	Methods for enhancing depth perception, according to depth cues.	20
5.1	Scene suitability guidelines and their formulation in our system.	59
5.2	Fuzzy logic operators.	60
5.3	Sample fuzzy rules.	60
5.4	Rendering methods for providing the depth cues.	62
5.5	Rendering methods with levels.	66
5.6	Some of the rendering methods in our system.	68
6.1	Information about the meshes.	71
6.2	Pearson (r) and Spearman (ρ) correlation coefficients for each mesh.	76
6.3	Spearman correlation coefficients for each model and metric (highest values are marked with bold font).	79
6.4	Pearson (r) and Spearman (ρ) correlation coefficients for each mesh.	81
6.5	Effect of the <i>minResolution</i> parameter on the correlation strengths of each mesh.	82
6.6	Processing times (in seconds) for several meshes.	84
6.7	Pearson (r) and Spearman (ρ) correlation coefficients for each mesh. (Mesh-based approach is shown with bold font.)	85
6.8	Prediction accuracy of each metric for each mesh (highest values are marked with bold font).	89
6.9	Tasks in the experiments (<i>Exp.#1</i> : Tree Visualization, <i>Exp.#2</i> : Graph Visualization, <i>Exp.#3</i> : Scatter Plot Visualization, <i>Exp.#4</i> : Surface Visualization).	89

Chapter 1

Introduction

1.1 Motivation

Improving the graphical quality and decreasing the rendering cost is still an important research area in computer graphics and visualization, since interactive applications such as games are very popular and very realistic graphical contents are desired by the users. The ultimate measure of realism is the human perception rather than the physics and this topic is extensively studied in cognitive science literature. However, findings in cognitive science are not much applied on computer graphics applications. There are several attempts for employing visual attention in designing perceptually adaptive rendering frameworks [1, 2, 3, 4, 5]. In this work, we aim to employ the principles of human visual perception in order to enhance the perceived quality of 3D computer graphics.

3D mesh modeling, representation, and rendering methods have advanced to the level that they are now common in 3D games, virtual environments, and visualization applications. Conventional way of improving the visual quality of a 3D mesh is to increase the number of vertices and triangles. This provides a more detailed view; nevertheless, it also leads to a performance degradation. As a result, we need a measure for estimating the visual quality of 3D models, to be

able to balance the visual quality of 3D models and their computational time.

Most of the operations on 3D meshes cause certain distortions on the mesh surface and requires an estimation of the distortion. For instance, 3D mesh compression and streaming applications require a tradeoff between the visual quality and transmission speed. Watermarking techniques introduce artifacts and one should guarantee the invisibility of these artifacts. Most of the existing 3D quality metrics have focused on static meshes, and they do not target animated 3D meshes. Detection of distortions on animated meshes is particularly challenging since temporal aspects of seeing are complex and only partially modeled.

Information visualization, as an important application area of computer graphics, deals with presenting information effectively. Rapid development in 3D rendering methods and display technologies has also increased the use of 3D content for visualizing information, which may improve presentation. However, such developments lead to the problem of providing suitable depth information and using the third dimension in an effective manner, because it is important that spatial relationships between objects should be apparent for an information visualization to be comprehensible. Hence, it is clear that depth is an important component of visualization, and it should be handled carefully.

1.2 Scope of the Work

Our primary purpose in this study is to utilize the principles of visual perception in computer graphics and visualization applications by developing perceptually-driven computer graphics techniques in order to enhance the perceived quality in computer-generated scenes. Visual Perception is a broad discipline embracing visual attention, visual illusions, depth perception, motion perception, color vision, etc. We have to narrow down this field to obtain tangible results. Therefore, we mainly consider the “quality” aspect and restrict the major focus of this thesis to the following topics: visual quality assessment of 3D models and perceived depth quality in 3D scenes.

First of all, quality assessment of 3D meshes is generally understood as the problem of evaluation of a modified mesh with respect to its original form based on detectability of changes. Full-reference quality metrics are given a reference mesh and its processed version, and compute geometric differences to reach a quality value. Besides, it is required to handle topographical changes in the input meshes because several mesh processing methods, such as simplification, change the number of vertices and topology in the input mesh. As a result, we are interested in perceptual quality assessment methods for 3D meshes in this thesis.

Meanwhile, depth cues construct the core part of depth perception; the human visual system (HVS) uses depth cues to perceive spatial relationships between objects. Nevertheless, using the depth property in 3D visualization is not straightforward. Application designers choose proper depth cues and rendering methods based on the nature of the task, scene attributes, and computational costs of rendering. Improper use of depth cues may lead to problems such as reduced comprehensibility, unnecessary scene complexity, and cue conflicts, as well as physiological problems such as eye strain. Thus, developing a system that selects appropriate depth cues and rendering methods in support of target visualization tasks is another interest point of this thesis.

1.3 Contributions

The key contributions of the thesis can be summarized as follows:

- We propose an objective visual quality metric for measuring the visibility of local distortions on dynamic triangulated meshes. The proposed metric incorporates both spatial and temporal aspects of the human visual system by extending image-space sensitivity models for 2D imagery in 3D space. This visual quality metric is a general-purpose metric, which is independent of connectivity, shading, and material properties.

- Another contribution of this thesis is a machine learning approach which employs crowdsourcing methodology and metric learning techniques for assessing the visual quality of static 3D meshes.
- We also suggest a framework for enhancing the perceived depth quality on 3D computer-generated scenes, to support the current visualization task. This framework makes use of fuzzy logic for determining which depth cues are appropriate for the given scene and task. A cost-profit analysis is then performed to select a subset of the commonly-used rendering methods to provide these cues.
- Lastly, formal experimental evaluations of the proposed methods are presented. An open dataset including the subjective evaluations for the visibility of local distortions on 3D dynamic mesh sequences is attached to this study, to be used as a benchmark by other researchers.

1.4 Outline of the Thesis

This thesis is organized as follows: First, background information and fundamental concepts are given and then the proposed methodologies and technical contributions are explained. Lastly, a formal evaluation of the proposed methods are presented. Detailed outline of the thesis is listed below:

- Chapter 2 includes background information and a comprehensive literature survey under two main topics: Visual quality assessment and depth perception.
- In Chapter 3, we propose a perceptual method for assessing the visual quality of dynamic 3D meshes.
- Chapter 4 presents another perceptual visual quality metric designed for 3D static meshes, with a machine learning based approach.

- We explain the details of a framework we developed for automatically enhancing the perceived depth quality in a 3D scene, in Chapter 5.
- Chapter 6 contains experimental evaluations and discussion of the results, in separate sections for each of the proposed systems.
- Finally, Chapter 7 concludes the thesis with a summary of the current work and future research directions for the improvements.

Chapter 2

Background

In this chapter, we review the fundamental concepts required to comprehend the proposed methodologies, in two main sections. The first section is about visual quality assessment and it contains the main concepts and related work used in Chapter 3 and Chapter 4. Theoretical background and previous studies in the field of depth perception are elaborated in Section 2.2 to be used as the infrastructure of the study described in Chapter 5.

This study is formed by knowledge from two different fields: Computer Graphics and Visual Perception. However, we explain the concepts from a computer graphics perspective. Details of the human visual system, visual cortex, etc. are out of scope of this thesis.

2.1 Visual Quality Assessment for 3D Meshes

Section 2.1.1 includes a literature review of visual quality assessment methods in computer graphics, while Section 2.1.2 presents the significant mechanisms of the human visual system and it constructs the perceptual basis of the algorithms we propose in Chapter 3.

2.1.1 Methods for Visual Quality Assessment

Methods for quality assessment of triangle meshes can be categorized according to their approach to the problem and the solution space. Non-perceptual methods approach the problem geometrically, without taking human perception effects into account. On the other hand, perceptual methods integrate human visual system properties into computation. Moreover, solutions can further be divided into image-based and model-based solutions. Model-based approaches work in 3D object space, and use structural or attribute information of the mesh. Image-based solutions, on the other hand, work in 2D image space, and use rendered images to estimate the quality of the given mesh. Several quality metrics have been proposed; [6, 7, 8] present surveys on the recently proposed 3D quality metrics.

2.1.1.1 Geometry-Distance-Based Metrics

Several methods use purely geometrical information to compute a quality value of a single mesh or a comparison between meshes. Therefore, methods that fall into this category do not reflect the perceived quality of the mesh.

Model-based Metrics. The most straightforward object space solution is the Euclidean distance or root mean squared distance between two meshes. These methods restrict the compared meshes to have the same number of vertices and connectivity. To overcome this constraint, more flexible geometric metrics have been proposed. One of the most commonly used geometric measure is Hausdorff distance [9]. The Hausdorff distance defines the distance between two surfaces as the maximum of all pointwise distances. This definition is one-sided ($D(A, B) \neq D(B, A)$). Extensions to this approach have been proposed, such as taking the average, root mean squared error, or combinations [10].

Image-based Metrics. The most straightforward view dependent approach is the Root Mean Squared Error (RMSE) of two rendered images, by comparing

them pixel by pixel. This metric is highly affected by luminance, shifts and scales, therefore is not a good approach [6]. Peak signal to noise ratio (PSNR) is also a popular quality metric for natural images where RMS of the image is scaled with the peak signal value. Wang et al. [11] show that alternative pure mathematical quality metrics do not perform better than PSNR, although results indicate that PSNR gives poor results on pictures of artificial and human-made objects.

2.1.1.2 Perceptually Based Metrics

Perceptually-aware quality metrics or modification methods integrate computational models or characteristics of the human visual system into the algorithm. Lin and Kuo [12] present a recent survey on perceptual visual quality metrics; however, as this survey indicates, most of the studies in this field focus on 2D image or video quality. A large number of factors affect the visual appearance of a scene, and several studies only focus on a subset of features of the given mesh.

Model-based Perceptual Metrics. Curvature is a good indicator of structure and surface roughness which highly affect visual experience. A number of studies focus on the relation between curvature-linked characteristics and perceptual guide, and integrate curvature in quality assessment or modification algorithms. Karni and Gotsman [13] introduce a metric (*GL1*) by calculating roughness for mesh compression using Geometric Laplacian of every vertex. The Laplacian operator takes into account the geometry and topology. This scheme uses variances in dihedral angles between triangles to reflect local roughness and weigh mean dihedral angles according to the variance. Sorkine et al. [14] modifies this metric by using slightly different parameters to obtain the metric called *GL2*.

Following the widely-used structural similarity concept in 2D image quality assessment, Lavouè et al. [15] propose a local mesh structural distortion measure (*MSDM*), which uses curvature for structural information. *MSDM2* [16] method improves this approach in several aspects: The new metric is multi-scale and symmetric, the curvature calculations are slightly different to improve

robustness, and there is no requirement of fixed connectivity for the compared meshes.

Spatial frequency is linked to variance in 3D discrete curvature, and studies have used this curvature as a 3D perceptual measure [17, 18]. Roughness of a 3D mesh has also been utilized in measuring the quality of watermarked meshes [19, 20]. In [20], two objective metrics (*3DWPM1* and *3DWPM2*) derived from two definitions of surface roughness are proposed as the change in roughness between the reference and test meshes. Pan et al. [21] use the vertex attributes in their proposed quality metric.

Another metric developed for 3D mesh quality assessment is called *FMPD* which is based on local roughness estimated from Gaussian curvature [22]. Torkhani and colleagues [23] propose another metric (*TPDM*) based on curvature tensor difference of the meshes to be compared. Both of these metrics are independent of connectivity and designed for static meshes. Dong et al. [24] propose a novel roughness-based perceptual quality assessment method. The novelty of the metric lies in the incorporation of structural similarity, visual masking, and saturation effect which are highly employed in quality assessment methods separately. This metric is also similar to ours in the sense that it uses a HVS pipeline but it is designed for static meshes with connectivity constraints. Besides, they capture structural similarity which is not handled in our method.

Alternatively, Nader et al. [25] propose a Just Noticeable Distortion (JND) profile for flat-shaded 3D surfaces in order to quantify the threshold for the change in vertex position to be detected by a human observer, by defining perceptual measures for local contrast and spatial frequency in 3D domain. Guo et al. [26] evaluate the local visibility of geometric artifacts on static meshes by means of a series of user experiments. In these experiments, users paint the local distortions on the meshes and the prediction accuracies of several geometric attributes (curvatures, saliency, dihedral angle, etc.) and quality metrics such as Hausdorff distance, *MSDM2*, and *FMPD* are calculated. According to the results, curvature-based features outperform the others. They also provide a local distortion dataset as a benchmark.

A perceptually-based metric for evaluating dynamic triangle meshes is the *STED* error [27]. They first perform a subjective experiment to show the insufficiency of non-perceptual metrics. Using the observations from this experiment, they develop a perceptual metric including both spatial and temporal components. The metric is based on the idea that perception of distortion is related to local and relative changes rather than global and absolute changes [7]. The spatial part of the error metric is obtained by computing the standard deviation of relative edge lengths within a topological neighborhood of each vertex. Similarly, the temporal error is computed by creating virtual temporal edges connecting a vertex to its position in the subsequent frame. The hypotenuse of the spatial and temporal components then gives the *STED* error.

Another attempt for perceptual quality evaluation of dynamic meshes is by Torkhani et al. [28]. Their metric is a weighted mean square combination of three distances: speed-weighted spatial distortion measure, vertex speed related contrast, and vertex moving direction related contrast. Experimental studies show that the metric performs quite well; however, it requires fixed connectivity meshes. They also provide a publicly available dataset and a comparative study to benchmark existing image and model based metrics.

Image-based Perceptual Metrics. Human visual system characteristics are also used in image-space solutions. These metrics employ the Contrast Sensitivity Function (CSF), an empirically driven function that maps human sensitivity to spatial frequency. Daly’s widely used Visible Difference Predictor [29] gives the perceptual difference between two images. Longhurst and Chalmers [30] study VDP to show favorable image-based results with rendered 3D scenes. Lubin proposes a similar approach with Sarnoff Visual Discrimination Model (VDM) [31], which operates in spatial domain, as opposed to VDP’s approach in frequency domain. Li et al. [32] compare VDP and Sarnoff VDM with their own implementation of the algorithms. Analysis of the two algorithms shows that the VDP takes place in feature space and takes advantage of FFT algorithms, but a lack of evidence of these feature space transformations in the HVS gives VDM an advantage.

Bolin et al. [33] incorporate color properties in 3D global illumination computations. Studies show that this approach gives accurate results [34]. Minimum detectable difference is studied as a perceptual metric [35] that handles luminance and spatial processing independently. Another approach for computer-generated images is Visual Equivalence Detector [36]. Visual impressions of scene appearance are analyzed and the method outputs a visual equivalence map.

Visual masking, which refers to the decrease in the visibility of one stimulus because of the presence of another stimulus, is taken into account in 3D graphical scenes with varying texture, orientation and luminance values [37]. Several approaches with color emphasis is introduced by Albin et al. [38], which predict differences in LLAB color space. Dong et al. [39] exploit entropy masking, which accounts for the lower sensitivity of the HVS to distortions in unstructured signals, for guiding adaptive rendering of 3D scenes to accelerate rendering.

An important question that arises is whether model-based metrics are superior over image-based solutions. Although there are several studies on this issue, it is not possible to clearly state that one type of metrics is better than the other. It is stated that image quality metrics are not suitable for measuring the quality of 3D meshes since the results are highly affected by lighting and animation [40]. On the other hand, it is claimed that image-based metrics predict perceptual quality better than metrics working on 3D geometry [41]. A recent study [42] investigates the best set of parameters for the image-based metrics when evaluating the quality of 3D models and compares them to several model-based methods. The implications from this study show that image-based metrics perform well for simple use cases such as determining the best parameters of a compression algorithm or in the cases when model-based metrics are not applicable.

2.1.2 Perceptual Concepts in Visual Quality Assessment

In this section, we summarize and discuss several mechanisms of the human visual system that construct the core part of the perceptually-based visual quality assessment methods.

2.1.2.1 Luminance Adaptation

The luminance that falls on the retina may vary in significant amount from a sunny day to moonless night. The photoreceptor response to luminance forms a nonlinear S-shaped curve, which is centered at the current adaptation luminance and exhibits a compressive behavior while moving away from the center [43].

Daly [29] has developed a simplified local amplitude nonlinearity model in which the adaptation level of a pixel is merely determined from that pixel. Equation 2.1 provides this model.

$$\frac{R(i, j)}{R_{max}} = \frac{L(i, j)}{L(i, j) + c_1 L(i, j)^b} \quad (2.1)$$

where $R(i, j)/R_{max}$ is the normalized retinal response, $L(i, j)$ is the luminance of the current pixel, and c_1 and b are constants.

2.1.2.2 Channel Decomposition

The receptive fields in the primary visual cortex are selective to certain spatial frequencies and orientations [43]. There are several alternatives to account for modeling the visual selectivity of the HVS such as Laplacian Pyramid [44], Discrete Cosine Transform (DCT) [45], and Cortex Transform [46]. Most of the studies in the literature tend to choose Cortex Transform among these alternatives, since it offers a balanced solution for the tradeoff between physiological plausibility and practicality [43].

2D Cortex Transform combines both frequency selectivity and orientation selectivity of the HVS. Frequency selectivity component is modeled by the band-pass filters called Difference of Mesa (DoM), given in Eq. 2.2.

$$dom_k = \begin{cases} mesa_{k-1} - mesa_k & \text{for } k = 1 \dots K - 2 \\ mesa_{k-1} - baseband & \text{for } k = K - 1 \end{cases} \quad (2.2)$$

where K is the total number of spatial bands [43]. Low-pass filters $mesa_k$ and

baseband are calculated using Eq. 2.3.

$$mesa_k = \begin{cases} 1 & , \rho \leq r - \frac{tw}{2} \\ \frac{1}{2}(1 + \cos(\frac{\pi(\rho - r + \frac{tw}{2})}{tw})) & , r - \frac{tw}{2} < \rho \leq r + \frac{tw}{2} \\ e^{-\frac{\rho^2}{2\sigma^2}} & , \rho < r_{K-1} + \frac{tw}{2} \\ 0 & , \text{otherwise} \end{cases} \quad (2.3)$$

where $r = 2^{-k}$, $\sigma = \frac{1}{3}(r_{K-1} + \frac{tw}{2})$ and $tw = \frac{2}{3}r$. For the orientation selectivity, *fan filters* are used (Eq. 2.4 and 2.5).

$$fan_l = \begin{cases} \frac{1}{2}(1 + \cos(\frac{\pi|\theta - \theta_c(l)|}{\theta_{tw}})) & \text{for } |\theta - \theta_c(l)| \leq \theta_{tw} \\ 0 & \text{otherwise} \end{cases} \quad (2.4)$$

$$\theta_c(l) = (l - 1) \cdot \theta_{tw} - 90 \quad (2.5)$$

where $\theta_c(l)$ is the orientation of the center and $\theta_{tw} = 180/L$ is the transitional width. Then the cortex filter (Eq. 2.6) is obtained by multiplying the *dom* and *fan* filters.

$$B^{k,l} = \begin{cases} dom_k \cdot fan_l & \text{for } k = 1 \dots K - 1 \text{ and } l = 1 \dots L \\ baseband & \text{for } k = K \end{cases} \quad (2.6)$$

Figure 2.1 illustrates the resulting band-pass and orientation selective filters. Each filter selects a different range of spatial frequency and orientation as depicted in Figure 2.2.

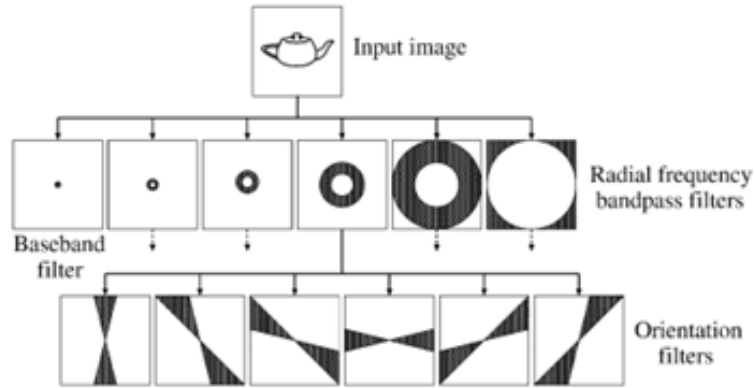


Figure 2.1: Cortex Transform - Organization of the filter bank. (Image from [47], © 2000 IEEE, reprinted with permission.)

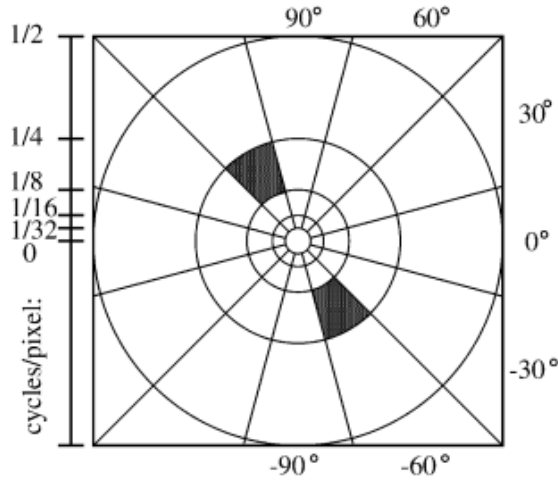


Figure 2.2: Cortex Transform - Decomposition of the image into radial and orientation selective channels (frequency values estimate the center frequency of each frequency band.) (Image from [47], © 2000 IEEE, reprinted with permission.)

2.1.2.3 Contrast Sensitivity

Contrast can be defined as the difference between the lightest and darkest part of an image [48] and contrast sensitivity refers to the ability to recognize subtle contrast changes in an image. Contrast sensitivity phenomena is highly exploited for rendering optimization.

Spatial Contrast Sensitivity

Spatial features of an image are constructed by the intensity values across the image. The sensitivity of the HVS to a change in spatial features depends on several parameters such as contrast, frequency, orientation, and phase.

In the psychophysical experiments to measure the sensitivity, luminance gratings of sine waves with different spatial frequencies, measured in *cycles per degree*, are shown to the observers and HVS response to spatial frequency is measured. Luminance grating in Figure 2.3 depicts the behavior of contrast sensitivity with respect to varying spatial frequencies. One can observe our decreased sensitivity to contrast difference in the left and right ends of the figure. In other words, we are most sensitive to the contrast changes in medium

spatial frequencies.

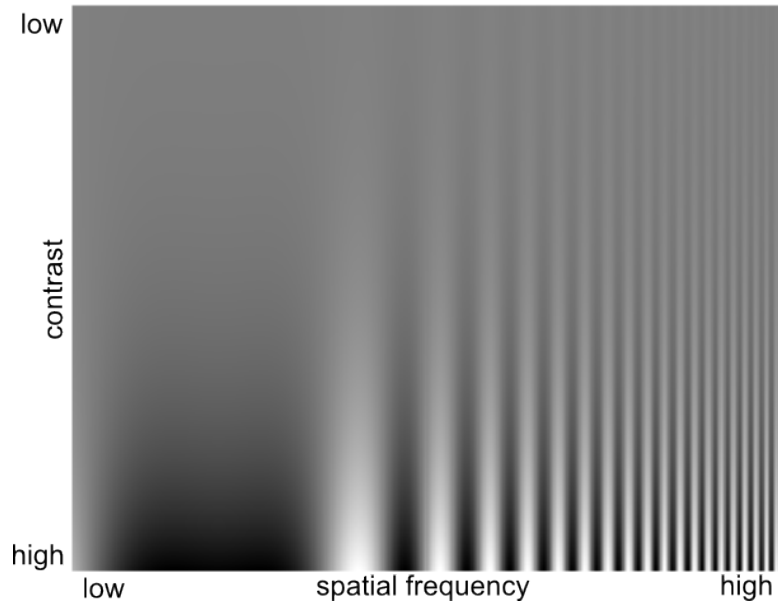


Figure 2.3: Contrast sensitivity at various spatial frequencies. (Figure adapted by the author from [49])

Figure 2.4a plots Blakemore et al.'s [50] experimental results without adaptation effects. The Contrast Sensitivity Function (CSF) measures the sensitivity to luminance gratings as a function of spatial frequency, where sensitivity is defined as the inverse of the threshold contrast. Mostly used spatial CSF models are Daly's [29], Barten's [51], and Mannos and Sakrison's [52] models.

Temporal Contrast Sensitivity

Intensity change across time constructs the temporal features of an image. In a user study conducted by Kelly [53], the sensitivity with respect to *temporal frequency* is estimated by displaying a simple shape with alternating luminance as a stimuli. The results of the experiment are used to plot the temporal CSF shown in Figure 2.4b.

Another issue to consider is the eye's tracking ability, known as *smooth pursuit*, which compensates for the loss of sensitivity due to motion by reducing the retinal speed of the object of interest to a certain degree. Daly [54] draws a heuristic for

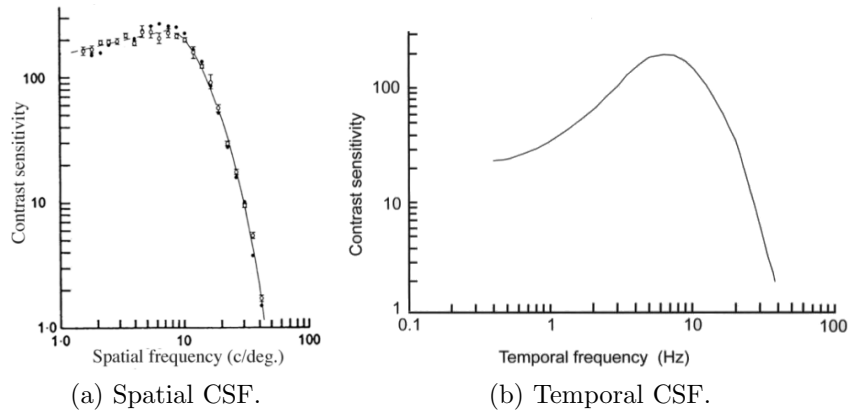


Figure 2.4: Spatial and temporal Contrast Sensitivity Functions. (a - Image from [50], © 1969 John Wiley and Sons, reprinted with permission. b - Constructed using Kelly’s [53] temporal adaptation data.)

smooth pursuit according to the experimental measurements.

It is also important to note the distinction between the spatiotemporal and spatiovelocity CSF [54]. Spatiotemporal CSF (Figure 2.5a) takes spatial and temporal frequencies as input, while spatiovelocity CSF (Figure 2.5b) takes directly the retinal velocity instead of the temporal frequency. Spatiovelocity CSF is more suitable for our application since it is more straightforward to estimate the retinal velocity than temporal frequency and it allows the integration of the smooth pursuit effect.

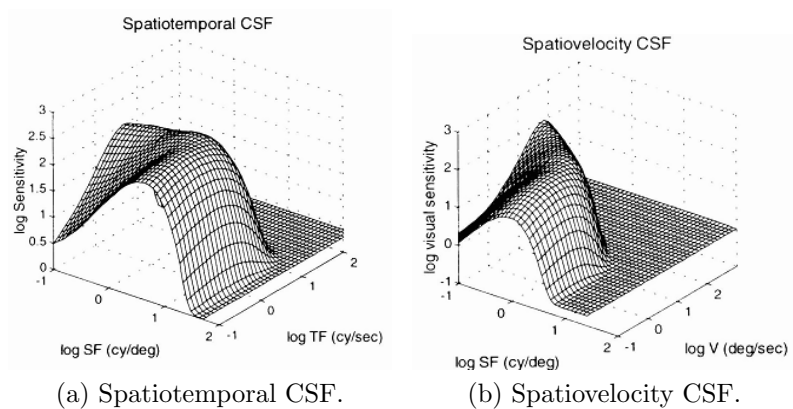


Figure 2.5: Spatiotemporal and spatiovelocity CSF. (Image from [54], © 1998 SPIE, reprinted with permission.)

2.2 Depth Perception

In this section, we first present the principles of depth perception (Section 2.2.1) and then we investigate how to apply these perceptual principles on Computer Graphics and Information Visualization problems (Section 2.2.2).

2.2.1 Depth Cues and Cue Combination Models

Depth cues, which help the human visual system to perceive the spatial relationships between the objects, construct the core part of depth perception. Depth cues can be categorized as pictorial, oculomotor, binocular, and motion-related [55, 56, 57]. These depth cues are illustrated in Figure 2.6 and detailed explanations are available in [58].

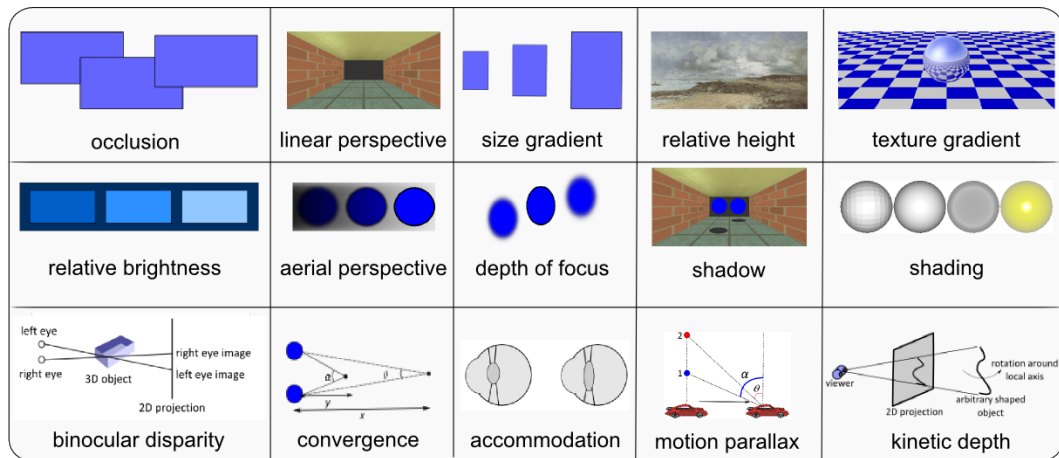


Figure 2.6: Depth cues. (Image from [59], © 2013 ACM, reprinted with permission.)

The interaction between depth cues and how they are unified in the human visual system for a single knowledge of depth is widely studied. Several cue combination models have been proposed such as *cue averaging*, *cue specialization*, and *range extension* [55, Section 27.1].

Most of the research on cue combination focuses on the *cue weighted averaging*

models [60, 61], in which each cue is associated with a weight determining its reliability. The overall percept is obtained by summing up the individual depth cues multiplied by their weights.

In *cue specialization* model, different cues may be used for interpreting different components of a stimulus. For instance, when the aim is to detect the curvature of a surface, binocular disparity is more effective; on the other hand, if the target is to interpret the shape of the object, shading and texture cues are more important [55, Section 27.1]. Based on this model, several researchers consider the target task as an important factor in determining cues that enhance depth perception [62, 63]. Ware [57] presents a list of possible tasks and a survey of depth cues according to their effectiveness under these tasks. For instance, he finds that perspective is a strong cue when the task is “judging objects’ relative positions”; but it becomes ineffective for “tracing data paths in 3D graphs”.

According to the *range extension* model, different cues may be effective in different ranges. For example, binocular disparity is a strong cue for near distances, while perspective becomes more effective for far distances [55, Section 27.1]. Cutting and Vishton [64] provide a distance-based classification of depth cues by dividing the space into three ranges and investigating the visual sensitivity of the HVS to different depth cues in each range.

Cue dominance is a model proposed to consider cue conflict situations, by vetoing some sources of information totally [55, Section 27.1]. In other words, if two depth cues provide conflicting information, one of them may be suppressed and the final percept may be based on the other cue.

Recent studies in the perception literature aim the incorporation of these models and have focused on a probabilistic model [65, 66]. In this approach, Bayesian probability theory is used for modeling how the HVS combines multiple cues based on prior knowledge about the objects. The problem is formulated as a posterior probability distribution ($P(s|d)$ where s is the scene property to estimate, i.e. the depth values of the objects and d is the sensory data, i.e. information provided by the available depth cues). Using Bayes’ rule with the assumption

that each cue is conditionally independent, posterior distribution is computed from the prior knowledge ($P(s)$) about the statistics of s and likelihood function ($P(d|s)$). After computing the posterior, Bayesian decision maker chooses a course of action by optimizing a loss function.

There are also a variety of experimental studies that investigate the interaction between different depth cues. Hubona et al. [67] investigate the relative contributions of binocular disparity, shadows, lighting, and background to 3D perception. Their most obvious finding is that stereoscopic viewing strongly improves depth perception with respect to accuracy and response time. Wanger et al. [68] explore the effects of pictorial cues and conclude that the strength of a cue is highly affected by the task. Zannoli et al. [69] state that a reliable depth ordering can be performed by providing correct blur and accommodation information, based on their experimental study.

2.2.2 Depth Perception in Computer Graphics and Visualization

2.2.2.1 Depth Perception in Computer Graphics

Based on the depth cues and principles discussed in the previous section, different rendering methods have been developed for enhancing depth perception in 3D rendered scenes. It is appropriate to examine these methods according to the cues they provide as listed in Table 2.1.

Perspective-based cues: It is possible to obtain the cues *occlusion*, *size gradient*, and *relative height* by transforming the objects in the scene or changing the camera position. For the *relative height* cue, drawing lines from the objects to the ground plane is widely used to make the height between the object and the ground more visible [57]. A ground plane or a room facilitates the interpretation of the cues *relative height* and *size gradient*. In addition, placing objects of known sizes is a method for enabling the user to judge the sizes of unknown objects [57].

Table 2.1: Methods for enhancing depth perception, according to depth cues.

Depth Cues	Depth Enhancement Methods
Occlusion, Size gradient, Relative height	Matrix transformations, ground plane, room, placing objects of known sizes, dropping lines to ground
Relative brightness, Aerial perspective	Fog, proximity luminance
Texture gradient	Texture mapping, bump mapping
Shading, Shadow	Cast shadows, ambient occlusion, vicinity shading, cool-to-warm shading, boundary enhancement
Linear perspective	Perspective projection
Depth of focus	Depth-of-field
Accommodation, Convergence, Binocular disparity	Stereo rendering, multi-view rendering
Motional Cues	Eye tracking, face tracking Mouse, keyboard controlled motion

Focus related cues: Depth-of-field method is used to simulate the *depth-of-focus* cue. According to this method, objects in the range of focus are rendered sharp, while the objects outside of this range are rendered blurry and the blurriness level increases as the objects get further away from the range of focus [70]. Fog is commonly used to provide *aerial perspective* and *relative brightness* cues on the graphical contents and obtained by interpolating the color of a pixel between the surface color and the fog color with respect to the distance of the object. To make the *relative brightness* more obvious, Doshier et al. have proposed another method called proximity luminance covariance, which alters the contrast of the objects in the direction of the background color as the distance increases [57].

Shading and shadows: Several techniques have been proposed to approximate the global illumination calculation for real-time rendering. The ambient occlusion technique aims to increase the realism of 3D graphics in real time without a complete global illumination calculation. In Bunnell’s [71] work, an accessibility value, which represents the amount of hemisphere above the surface element not occluded by the geometry, is calculated by approximation for each surface

element. The surfaces are darkened according to these accessibility values.

Gooch shading is a non-photorealistic (NPR) shading model which is performed by interpolating between cool colors (blue tones) to warm colors (yellow tones) according to the distance from the light source [72, 73]. This kind of shading also provides atmospheric effect on the scene.

Boundary enhancement using silhouette and feature edges is a commonly-used tool in NPR [74, 75]. An image-space approach is proposed by Luft et al. [76] to enhance images that contain depth information. In this method, the difference between the original and the low-pass filtered depth buffer is computed to find spatially important areas. Then, color contrast on these areas is increased.

Binocular and oculomotor cues: To obtain binocular and oculomotor cues, there is a need for apparatus that provides multiple views of a 3D scene. There are several 3D display technologies such as shutter glasses, parallax barrier, lenticular, holographic, and head-tracked displays [77]. Rendering on 3D displays is an active topic in itself [78].

Motion related cues: Tracking the user’s position and controlling the motion of the scene elements according to the position of the user can be a tool for motion parallax. For instance, Bulbul et al. [79] propose a face tracking algorithm in which the user’s head movements control the position of the camera and enables the user to see the scene from different viewpoints.

2.2.2.2 Depth Perception in Information Visualization

Although there are a number of studies that investigate the perception of depth in scientific and medical visualizations [80, 81, 82], relatively few studies exist in the field of information visualization. The main reason behind this is that information visualization considers abstract data sets without inherent 2D or 3D semantics and therefore lacks a mapping of the data onto the physical screen space

[83]. However, when applied properly, it is hoped that using 3D will introduce a new data channel. Several applications would benefit from the visualization of 3D graphs, trees, scatter plots, etc. Earlier research claims that with careful design, information visualization in 3D can be more expressive than 2D [84]. Some common 3D information visualization methods are cone trees [85], data mountains [86], and task galleries [87]. We refer the reader to [88] and [89] for a further investigation of these techniques.

Ware and Mitchell [90] investigate the effects of binocular disparity and kinetic depth cues for visualizing 3D graphs. They determine binocular disparity and motion to be the most important cues for graph visualization for several reasons. Firstly, since there are no parallel lines in graphs, perspective has no effect. Shading, texture gradient, and occlusion also have no effect unless the links are rendered as solid tubes. Lastly, shadow is distracting for large graphs. According to their experimental results, binocular disparity and motion together produce the lowest error rates. Staib et al. [91] aim improving the scatter plot visualizations by employing depth-of-field effect, depending on the area of interest.

A number of rendering methods such as shadows, texture mapping, fog, etc. are commonly used for enhancing depth perception in computer graphics and visualization. However, there is no comprehensive framework for uniting different methods of depth enhancement in a visualization. A recent study by Hall et al. [92] present a survey about the existing emphasis frameworks for information visualization and propose a new mathematical framework for information visualization emphasis, based on visual prominence sets. Studies by Tarini et al. [81], Weiskopf and Ertl [93], and Swain [94] aim to incorporate several depth cues but they are limited in terms of the number of cues they consider.

Our aim is to investigate how to apply widely-used rendering methods in combination, to support the current visualization task. Our primary goal is to reach a comprehensible visualization in which the user is able to perform the given task easily. According to Ware [95], “depth cues should be used selectively to support design goals and it does not matter if they are combined in ways that are inconsistent with realism”. Depth cueing used in information visualization

is generally stronger than any realistic atmospheric effects, and there are several artificial techniques such as proximity luminance, dropping lines to ground plane, etc. [57]. Moreover, Kersten et al. [96] found no effect of shadow realism on depth perception. These findings show that it is possible to exaggerate the effects of these methods to obtain a comprehensible scene at the expense of deviating from realism.

Chapter 3

Visual Quality Assessment of Dynamic Meshes

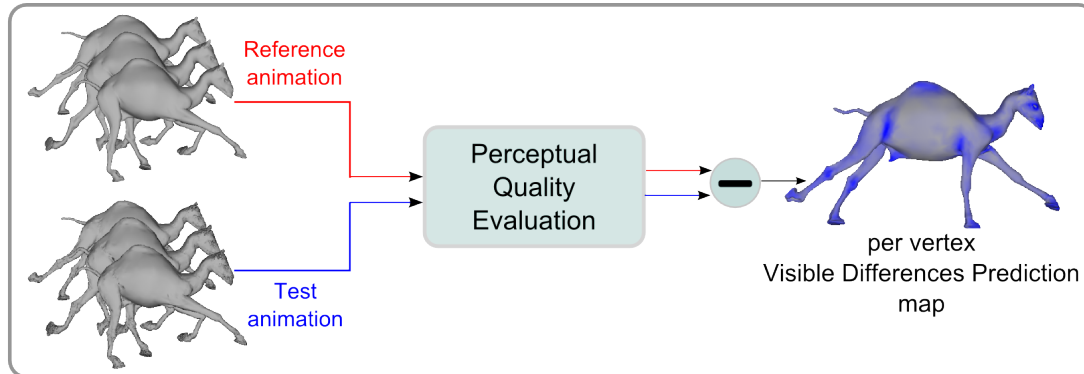


Figure 3.1: Overview of the perceptual quality evaluation for dynamic triangle meshes.

In this chapter, we propose a method to estimate the 3D spatiotemporal response, by incorporating temporal as well as spatial HVS processes. For this purpose, our method extends the image-space sensitivity models for 2D imagery in 3D space. These models, based on vast amount of empirical research on retinal images, allow us to follow a more principled approach to model the perceptual response to 3D meshes. The result of our perceptual quality metric is the probability of distortion detection as a 3D map, acquired by taking the difference between estimated visual response 3D map of both meshes (Figure 3.1).

We propose two alternative methods for evaluating the perceived quality of dynamic triangle meshes. In the first approach, which we call *voxel-based*, we construct a 4D space-time (3D+time) volume and extend several HVS correlated processes used for 2D images, to operate on this volume. On the other hand, the second approach called *mesh-based*, directly operates on the mesh vertices. Following sections include detailed explanations of both methods.

3.1 Voxel-based Approach

3.1.1 Overview

Our work shares some features of the VDP method [29] and recent related work. These methods have shown the ability to estimate the perceptual quality of static images [29] and 2D video sequences for animated walkthroughs [47].

Figure 3.2 portrays the overview of the method. Our method has a full reference approach in which a reference and a test mesh sequence are provided to the system. Both the reference and test sequences undergo the same perceptual quality evaluation process and the difference of these outputs is used to generate a per-vertex probability map for the animated mesh. The probability value at a vertex estimates the visible difference of the distortions in the test animation, when compared to the reference animation. Below, the steps of the algorithm are explained in detail.

3.1.2 Preprocessing

Calculation of the illumination, construction of the spatiotemporal volume, and estimation of vertex velocities are performed in the preprocessing step, since we do not need to recalculate them later.

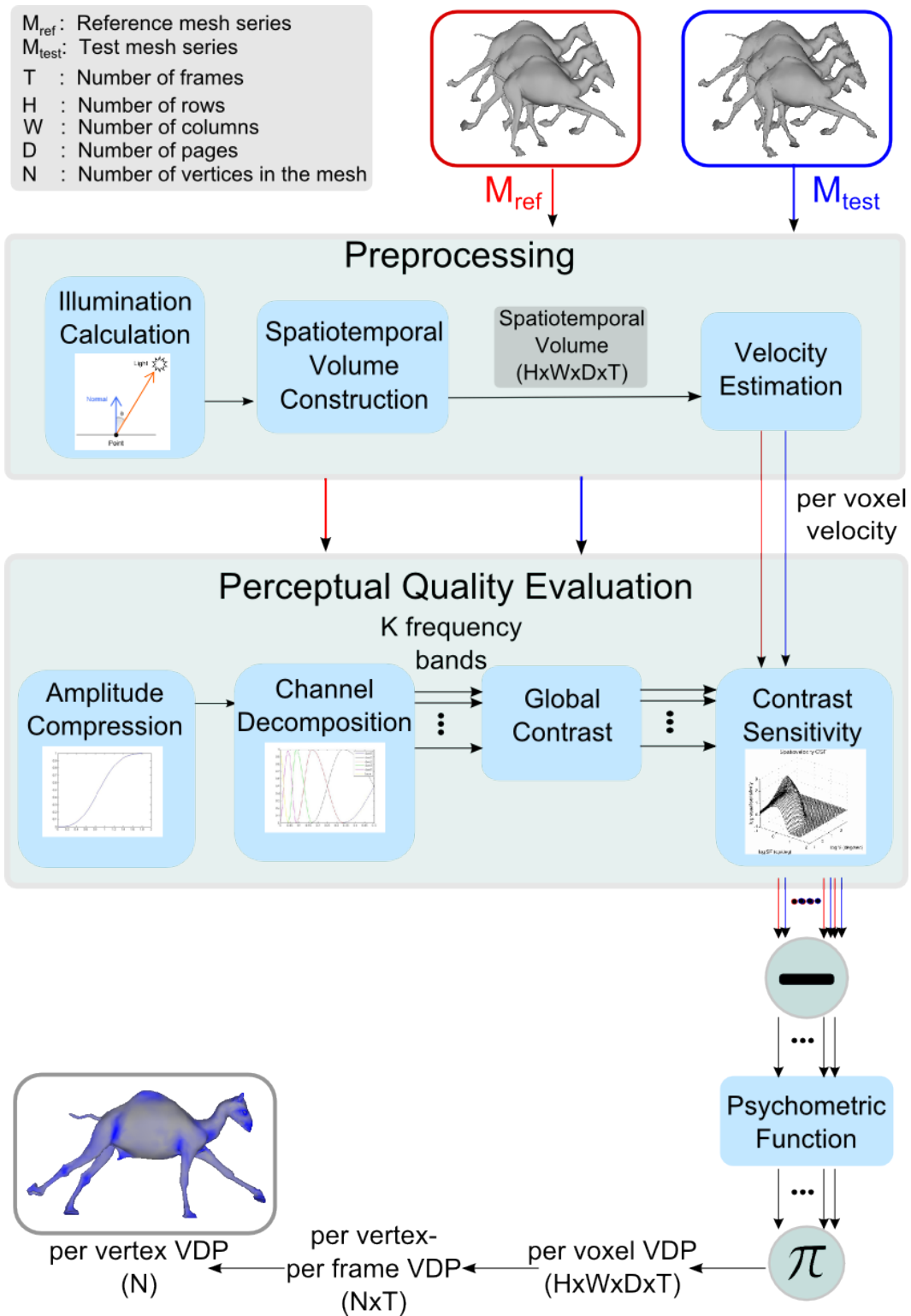


Figure 3.2: Method overview for the voxel-based approach.

Illumination Calculation. First we calculate the vertex colors assuming a Lambertian surface with diffuse and ambient components (Eq. 3.1).

$$\mathbf{I} = k_a \mathbf{I}_a + k_d \mathbf{I}_d (\mathbf{N} \cdot \mathbf{L}) \quad (3.1)$$

where \mathbf{I}_a is the intensity of the ambient light, \mathbf{I}_d is the intensity of the diffuse light, \mathbf{N} is the vertex normal, \mathbf{L} is the direction to the light source, and k_a and k_d are ambient and diffuse reflection coefficients, respectively.

In this study, we aim a general-purpose quality evaluation that is independent of connectivity, shading, and material properties. Therefore, information about the material properties, light sources, etc. are not available. A directional light source from left-above of the scene is assumed in accordance with the human visual system’s assumptions [55, Section 24.4.2].

The lighting model with the aforementioned assumptions can be generalized to incorporate multiple light sources, specular reflections, etc. using Eq. 3.2; if light sources and material properties are available.

$$\mathbf{I} = k_a \mathbf{I}_a + \sum_{i=1}^n [k_d \mathbf{I}_d^i (\mathbf{N} \cdot \mathbf{L}^i) + k_s \mathbf{I}_d^i (\mathbf{N} \cdot \mathbf{H}^i)^p] \quad (3.2)$$

where n is the number of light sources, k_s is the specular reflection coefficient, and \mathbf{H} is the halfway vector.

Construction of the Spatiotemporal Volume. We convert the object-space mesh sequences into an intermediate volumetric representation, to be able to apply image-space operations. We construct a 3D volume for each frame, where we store the luminance values of the vertices at each voxel. The values of the empty voxels are determined by linear interpolation.

Using such a spatiotemporal volume representation provides an important flexibility as we get rid of the connectivity problems and it allows us to compare meshes with different number of vertices. Moreover, the input model is not restricted to be a triangle mesh; volumetric representation enables the algorithm to be applied on other representations such as point-based graphics. Another

advantage is that the complexity of the algorithm is not much affected by the number of vertices.

To obtain the spatiotemporal volume, we first calculate the Axis Aligned Bounding Box (AABB) of the mesh. To prevent inter-frame voxel correspondence problems, we use the overall AABB of the mesh sequences. We use the same voxel resolution for both test and reference mesh sequences. Determining the suitable resolution for the voxels is critical since it highly affects the accuracy of the results and the time and memory complexity of the algorithm. At this point, we use a heuristic to calculate the resolution at each dimension, in proportion to the length of the bounding box in the corresponding dimension. According to this heuristic, in Eq. 3.3, we calculate the volume resolution as W , H , and D in x , y , and z dimensions respectively. We analyze the effect of the *minResolution* parameter in this equation on the accuracy, in the results section.

$$\begin{aligned}
 \mathit{minLength} &= \min(\mathit{width}_{AABB}, \mathit{height}_{AABB}, \mathit{depth}_{AABB}) \\
 w &= \lfloor \mathit{width}_{AABB} / \mathit{minLength} \rfloor \\
 h &= \lfloor \mathit{height}_{AABB} / \mathit{minLength} \rfloor \\
 d &= \lfloor \mathit{depth}_{AABB} / \mathit{minLength} \rfloor \\
 W &= w \times \mathit{minResolution} \\
 H &= h \times \mathit{minResolution} \\
 D &= d \times \mathit{minResolution}
 \end{aligned} \tag{3.3}$$

At the end of this step, we obtain a 3D spatial volume for each frame, which in turn constructs a 4D (3D+time) representation for both reference and test mesh sequences. We call this structure *spatiotemporal volume*. Also, an index structure is maintained to keep the voxel indices of each vertex. The rest of the method operates on this 4D spatiotemporal volume.

In the following steps, we do not use the full spatiotemporal volume for performance related concerns. We define a time window as suggested by Myszkowski et al. [47, p. 362]. According to this heuristic, we only consider a limited number of consecutive frames to compute the visible difference prediction map of a specific frame. In other words, to calculate the probability map for the i^{th} frame, we

process the frames between $i - \lfloor tw/2 \rfloor$ and $i + \lfloor tw/2 \rfloor$, where tw is the length of the time window. We empirically set it as $tw = 3$.

Velocity Estimation. Since our method also has time dimension, we need the vertex velocities in each frame. Using an index structure, we compute the voxel displacement of each vertex (D_i) between consecutive frames ($\Delta D_i = \|p_{it} - p_{i(t-1)}\|$) where p_{it} denotes the voxel position of vertex i at frame t). The remaining empty voxels inside the bounding box are assumed to be static.

Then we calculate the velocity of each voxel at each frame (v in *deg/sec*), using the pixel resolution (ppd in *pixels/deg*) and frame rate (FPS in *frames/sec*) with Eq. 3.4. We assume default viewing parameters of $0.5m$ viewing distance and *19-inch* display with 1600×900 resolution, while calculating ppd in Eq. 3.4. This is then adapted with N_1 frames to reduce the erroneous computations (Eq. 3.5).

$$v_{it} = \frac{\Delta D_i}{ppd} \times FPS \quad (3.4)$$

$$v'_{it} = \frac{v_{i(t-1)} + v_{it} + v_{i(t+1)}}{3} \quad (3.5)$$

Lastly, it is crucial to compensate for smooth pursuit eye movements to be used in spatiotemporal sensitivity calculations. This will allow us to handle temporal masking effect where high-speed motion hides the visibility of distortions. The following equation (Eq. 3.6) describes a motion compensation heuristic proposed by Daly [54].

$$v_R = v_I - \min(0.82v_I + v_{min}, v_{max}) \quad (3.6)$$

where v_R is the compensated velocity, v_I is the physical velocity, v_{min} is the drift velocity of the eye (0.15 deg/sec), v_{max} is the maximum velocity that the eye can track efficiently (80 deg/sec). According to Daly [54], the eye tracks all objects in the visual field with an efficiency of 82 %. We adopt the same efficiency value for our spatiotemporal volume. However, if the visual attention map is available, it is also possible to substitute this map as the tracking efficiency [97].

3.1.3 Perceptual Quality Evaluation

In this section, the main steps of the perceptual quality evaluation system are explained in detail.

Amplitude Compression. Daly [29] proposes a simplified local amplitude non-linearity model as a function of pixel location, which assumes perfect local adaptation (Section 2.1.2.1). We have adapted this nonlinearity to our spatiotemporal volume representation (Eq. 3.7).

$$\frac{R(x, y, z, t)}{R_{max}} = \frac{L(x, y, z, t)}{L(x, y, z, t) + c_1 L(x, y, z, t)^b} \quad (3.7)$$

where x, y, z , and t are voxel indices, $R(x, y, z, t)/R_{max}$ is the normalized response, $L(x, y, z, t)$ is the value of the voxel, $b = 0.63$ and $c_1 = 12.6$ are empirically set constants. Voxel values are compressed by this amplitude nonlinearity.

Channel Decomposition. We have adapted the Cortex Transform [29] which is described in Section 2.1.2.2, on our spatiotemporal volume with a small exception. A 3D model is not assumed to have a specific orientation at a given time, in our method. For this purpose, we exclude fan filters that are used for orientation selectivity from the Cortex Transform adaptation. Therefore, in our cortex filter implementation we use Eq. 3.8 instead of Eq. 2.6 with only *dom* filters (Eq. 2.2). These band-pass filters are demonstrated in Figure 3.3.

$$B^k = \begin{cases} dom_k & \text{for } k = 1 \dots K - 1 \\ baseband & \text{for } k = K \end{cases} \quad (3.8)$$

We perform cortex filtering in the frequency domain by applying Fast Fourier Transform (FFT) on the spatiotemporal volume and multiplying this with the cortex filters that are constructed in the frequency domain. We obtain K frequency bands at the end of this step. Each frequency band is then transformed back to the spatial domain using Inverse Fourier Transform. This process is illustrated in Figure 3.4.

Global Contrast. The sensitivity to a pattern is determined by its contrast rather than its intensity [98]. Contrast in every frequency channel is computed

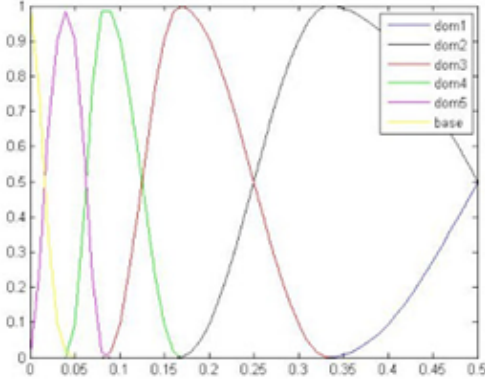


Figure 3.3: Difference of Mesa (DOM) filters. (x-axis: spatial frequency in *cycles/pixel*, y-axis: response)

according to the global contrast definition with respect to the mean value of the whole channel, given in Eq. 3.9 [98, 47].

$$C^k = \frac{I^k - \text{mean}(I^k)}{\text{mean}(I^k)} \quad (3.9)$$

where C^k is the spatiotemporal volume of contrast values and I^k is the spatiotemporal volume of luminance values in channel k .

Contrast Sensitivity. Filtering the input image with the Contrast Sensitivity Function (CSF) constructs the core part of the VDP-based models (Section 2.1.2.3). Since our model is for dynamic meshes, we use the spatiovelocity CSF (Figure 2.5b) which describes the variations in visual sensitivity as a function of both spatial frequency and velocity, instead of the static CSF used in the original VDP.

Our method handles temporal distortions in two ways: First, smooth pursuit compensation handles temporal masking effect which refers to the loss of sensitivity due to high speed. Secondly, we use spatiovelocity CSF in which contrast sensitivity is measured according to the velocity, instead of static CSF.

Each frequency band is weighted with the spatiovelocity CSF which is given in Eq. 3.10 [54, 53]. One input to the CSF is per voxel velocities in each frame, estimated in preprocessing; and the other input is the center spatial frequency of

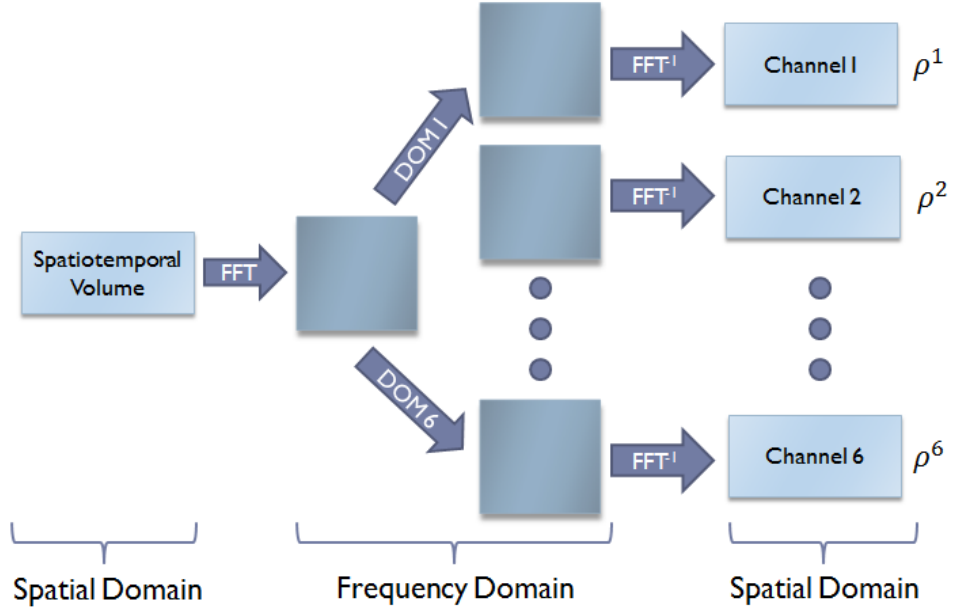


Figure 3.4: Frequency domain filtering in Cortex Transform.

each frequency band.

$$CSF(\rho, v) = c_0(6.1 + 7.3 \left| \log\left(\frac{c_2 v}{3}\right) \right|^3) \times c_2 v (2\pi c_1 \rho)^2 \times \exp\left(-\frac{4\pi c_1 \rho (c_2 v + 2)}{45.9}\right) \quad (3.10)$$

where ρ is the spatial frequency in *cycles/degree*, v is the velocity in *degrees/second*, and $c_0 = 1.14, c_1 = 0.67, c_2 = 1.7$ are empirically set coefficients. A more principled way would be to obtain these parameters through a parameter learning method.

Error Pooling. All the previous steps are applied on the reference and test animations. At the end of these steps, we obtain K channels for each mesh sequence. We take the difference of test and reference pairs for each channel and the outputs go through a psychometric function that maps the perceived contrast (C') to detection probability using Eq. 3.11 [43]. After applying the psychometric function, we combine each band using the probability summation formula (Eq. 3.12) [43].

$$P(C') = 1 - \exp(-|C'|^3) \quad (3.11)$$

$$\hat{P} = 1 - \prod_{k=1}^K (1 - P^k) \quad (3.12)$$

The resulting \hat{P} is a 4D volume that contains the detection probabilities per voxel. It is then straightforward to convert this 4D volume to per vertex probability map for each frame, using the index structure (Section 3.1.2). Lastly, to combine the probability maps of each frame into a single map, we take the average of all frames per vertex. This gives us a per vertex visible difference prediction map for the animated mesh.

Summary of the Method. The overall process is summarized in Eq.3.13 in which \mathcal{F} denotes the Fourier Transform, \mathcal{F}^{-1} denotes the Inverse Fourier Transform, and L_T and L_R are spatiotemporal volumes for test and reference mesh sequences, respectively. ρ^k is the center spatial frequency of channel k and V_T and V_R contain the voxel velocities corresponding to L_T and L_R respectively.

$$\begin{aligned} C_{T,R}^k &= \text{Contrast}(\text{Channel}_{T,R}^k) \times \text{CSF}(\rho^k, V_{T,R}) \\ \text{Channel}_{T,R}^k &= \mathcal{F}^{-1}[\mathcal{F}(AC_{T,R}) \times \text{DOM}^k] \\ AC_{T,R} &= \text{AmplitudeCompression}(L_{T,R}) \\ P^k &= P(C_T^k - C_R^k) \\ P &= 1 - \prod_{k=1}^K (1 - P^k) \end{aligned} \quad (3.13)$$

3.2 Mesh-based Approach

Due to the limitations such as computational complexity of the voxel-based approach, we also propose a fully mesh-based method as an alternative. In this method, we benefit from the eigen-decomposition of a mesh since eigenvalues are identified as natural vibrations of a mesh [99, Chapter 4] and hence, they are directly related to the geometric quality of the mesh.

3.2.1 Overview

In this mesh-based approach, almost the same steps in the voxel-based approach exist with several adaptations for 3D. The method is applied on the mesh vertices, not on the spatiotemporal volume representation. However, this introduces a restriction for the reference and test meshes to have the same number of vertices, since the computations are done per vertex. The steps of the method are displayed in Figure 3.5. Only the different parts of the algorithm will be described in detail.

3.2.2 Preprocessing

In the preprocessing step, illumination calculation and vertex velocity estimation are performed as in the voxel-based approach. The spatiotemporal volume construction step is not executed this time, since the rest of the algorithm will operate on the mesh vertices. As an additional step, *Manifold Harmonics Basis (MHB)* are computed and stored to feed the *Channel Decomposition* step of the new approach.

Calculation of MHBs. Calculation of MHBs is a costly operation since it requires eigen-decomposition of the mesh Laplacian. Fortunately, once they are computed; there is no need to recalculate them. Therefore, we calculate and store the MHBs in the preprocessing step.

For a triangle mesh of n vertices, a function basis H^k , called MHB is calculated. The k^{th} element of the MHB is a piecewise linear function with values H_i^k defined at i^{th} vertex of the surface, where $k = 1 \dots m$ and $i = 1 \dots n$ [100]. MHB is computed as the eigenvectors of discrete Laplacian of $\bar{\Delta}$ whose coefficients are given in Eq. 3.14.

$$\bar{\Delta}_{ij} = -\frac{\cot\beta_{ij} + \cot\beta'_{ij}}{\sqrt{|v_i^*||v_j^*|}} \quad (3.14)$$

where β_{ij} and β'_{ij} are two angles opposite to edge defined by vertices i and j , v^* refers to the circumcentric dual of simplex v , and $|\cdot|$ denotes the simplex volume.

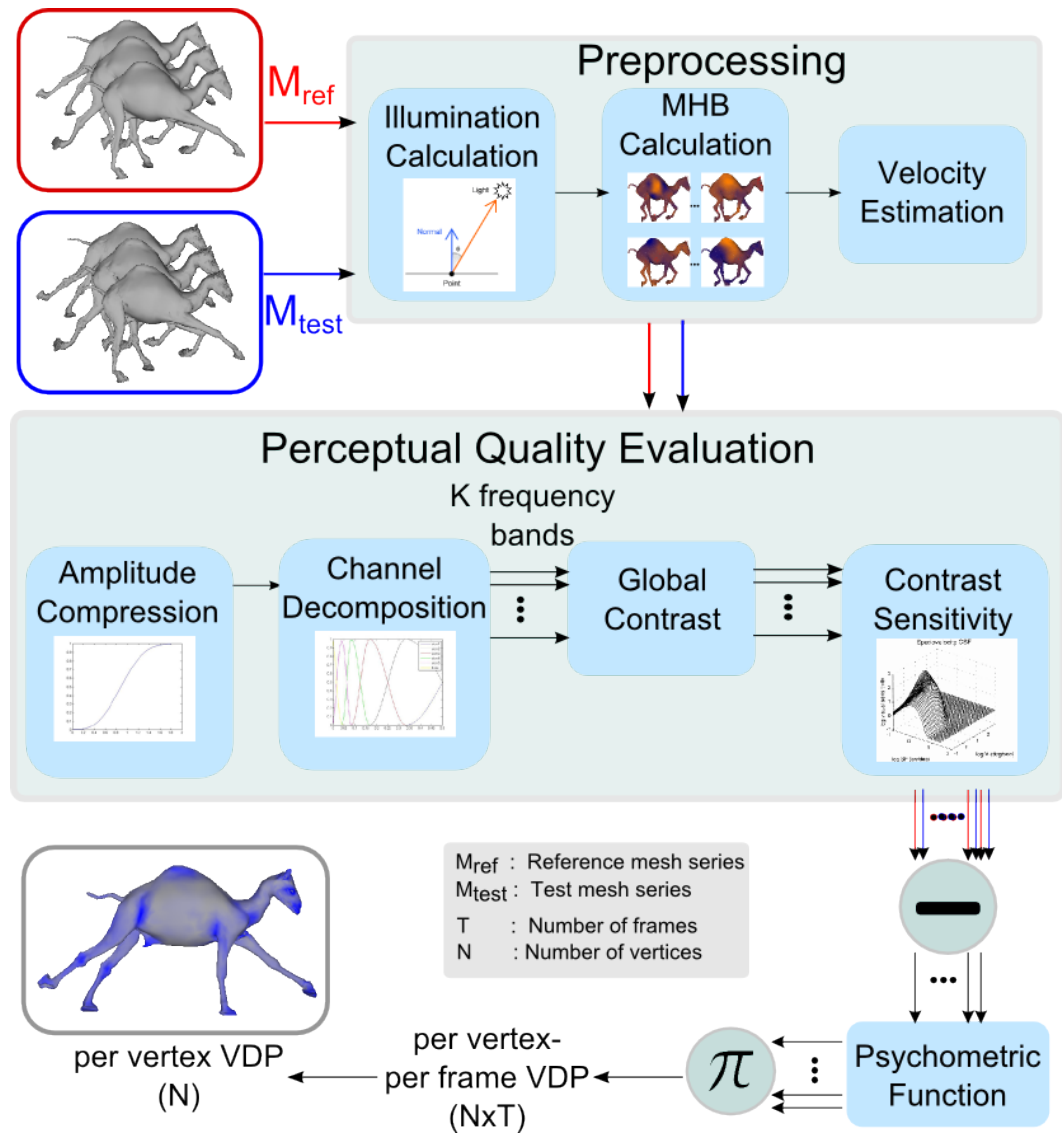


Figure 3.5: Method overview for the mesh-based approach.

3.2.3 Perceptual Quality Evaluation

All the steps of this method are similar to the corresponding steps in the voxel-based approach. Therefore, only *Channel Decomposition* step which is totally different from its counterpart in the previous approach will be explained in detail. Other steps follow the same procedures described in the previous section, with the exception that equations are applied per vertex instead of per voxel, since they are not applied on the spatiotemporal volume.

Channel Decomposition. The most important distinction of the mesh-based approach lies in the *Channel Decomposition* step. In this step of the voxel-based approach, we use *Cortex Transform* which filters the spatiotemporal volume with *DoM (Difference Of Mesa)* filters in the frequency domain. *Manifold Harmonics* can be considered as the generalization of Fourier analysis to surfaces of arbitrary topology [100]. Hence, we employ *Manifold Harmonics* for applying DoM filter on the mesh and obtain 6 frequency channels as in the voxel-based approach.

Figure 3.6 demonstrates the processing pipeline of *Manifold Harmonics*. In step A of this pipeline, *Manifold Harmonics Basis* is calculated for the given triangle mesh of N vertices, which is already performed in the preprocessing step in our implementation.

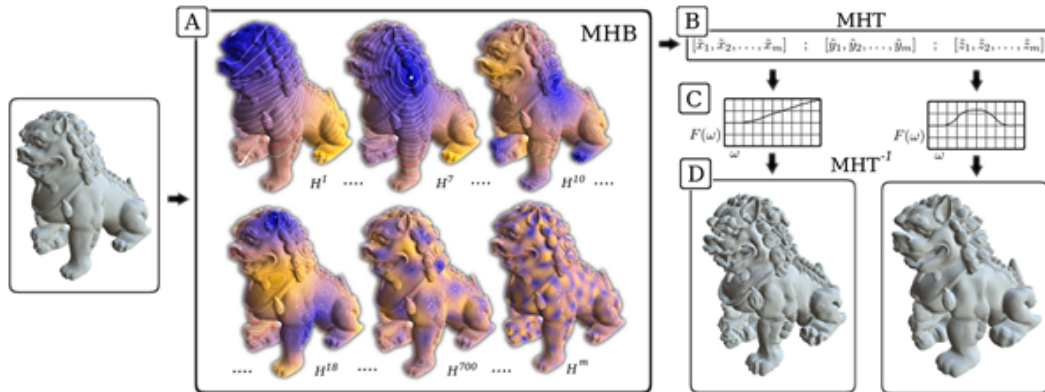


Figure 3.6: Pipeline for Manifold Harmonics Transform. (Image from [100], © 2008 John Wiley and Sons, reprinted with permission.)

Step B illustrates the transformation of the geometry into frequency space by the help of *Manifold Harmonics Transform (MHT)*, which corresponds to projecting x into MHB by solving for the coefficients \tilde{x}_k given in Eq. 3.15. In step C, frequency space filtering is performed by multiplying the coefficients calculated in MHT step by a frequency space filter $F(w)$ (Eq. 3.17). Lastly, the mesh is transformed back to the geometric space using *inverse Manifold Harmonics Transform (MHT⁻¹)*. In its simplest form, MHT^{-1} is performed using Eq. 3.16; however if a filtering is performed, we use Eq. 3.17 to obtain filtered values denoted by x_i^F . For a more detailed explanation of MHT, please refer to [100] and [99, Chapter 4].

$$\tilde{x}_k = \sum_{i=1}^N x_i D_{ii} H_i^k \quad (3.15)$$

$$x = \sum_{k=1}^m \tilde{x}_k H^k \quad (3.16)$$

$$x_i^F = \sum_{k=1}^m F(w_k) \tilde{x}_k H_i^k \quad (3.17)$$

One can discover the similarity between the processing pipeline of *Manifold Harmonics* and frequency domain filtering used in the voxel-based approach (Figure 3.4). MHT and MHT^{-1} correspond to *Fourier* and *Inverse Fourier Transform*, respectively. We construct DoM filters displayed in Figure 3.3 to be used as the frequency space filters ($F(w)$).

Note that the notation in Figure 3.6 was given assuming that the geometry of the mesh will be filtered. It is also possible to filter other attributes of the mesh. For instance, for filtering the color values, we need to replace x , y , z values with r , g , b values. Figure 3.7 exemplifies both geometry (a) and color filtering (b). In our case, we need filtering the color values of the mesh with DoM filters.

Figure 3.8 depicts the six frequency channels generated by applying *Cortex Transform* on an image, while Figure 3.9 includes the outputs of the *Channel Decomposition* step of the mesh-based approach for the hand mesh. One can notice the parallelism between these results as the frequency decreases from channel 1 to 6 and more details are captured in the high frequency bands.



Figure 3.7: Filtering different attributes of a mesh with MHT (Image from [101], courtesy of Bruno Levy, reprinted with permission).

3.3 Summary and Discussion

In this chapter, we propose two approaches for evaluating the visual quality of 3D animated mesh sequences. Both of these approaches follow a similar pipeline with Daly’s VDP method [29] and they have several pros and cons. The distinctions of the metrics proposed in this chapter from the current metrics can be listed as follows: Firstly, our metrics can handle dynamic meshes in addition to the static meshes. Secondly, we produce a per-vertex error map instead of a global quality value per-mesh, which allows to guide perceptual geometry processing applications. Lastly, the proposed metrics are not application specific.

In the first approach, which we refer as voxel-based, we construct an intermediate data structure called spatiotemporal volume and apply the HVS models on this structure. By the help of this volumetric representation, it is possible to measure the quality of 3D meshes with different connectivities. However, processing the spatiotemporal volume requires costly operations.

We also propose a mesh-based approach as an alternative to the voxel-based approach. In this method, a similar procedure is applied on the mesh vertices, not on the spatiotemporal volume. Although this approach improves the computational speed, it restricts the reference and test meshes to have the same number of vertices, as the computations are performed per vertex.

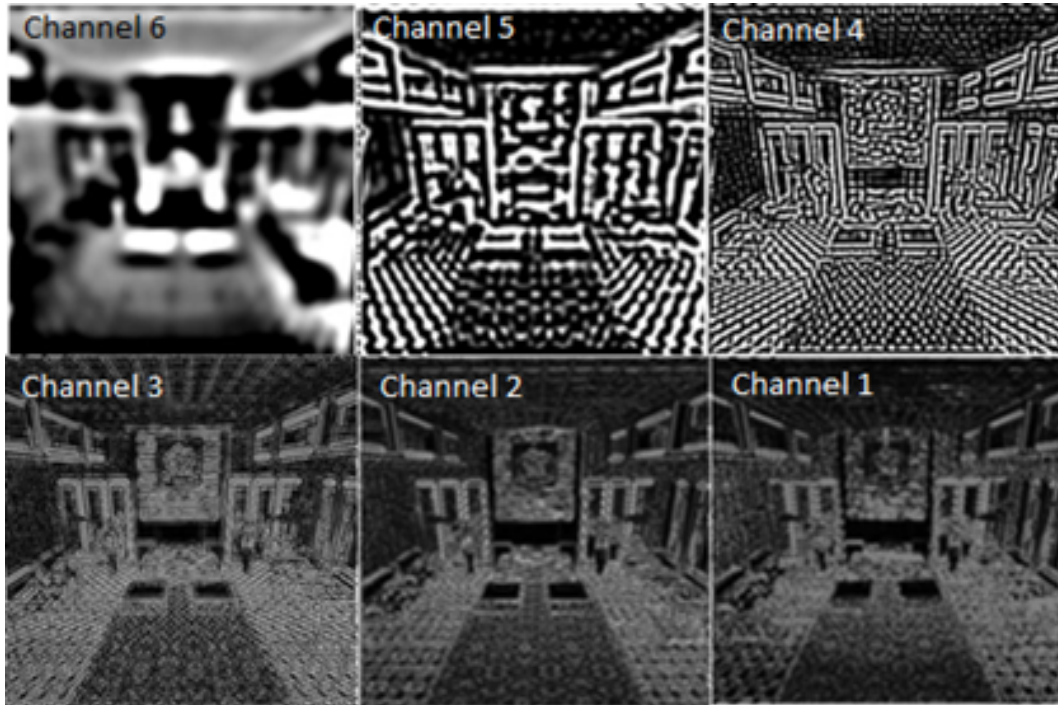


Figure 3.8: Application of Cortex Transform on an image (Image courtesy of Karol Myszkowski).

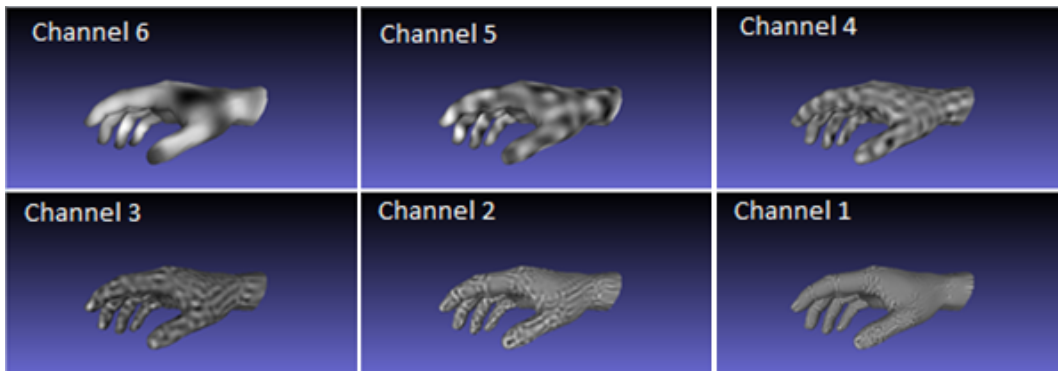


Figure 3.9: Output of the Channel Decomposition step for the hand mesh.

Both of the methods provided in this chapter follow a bottom-up procedure including low-level HVS mechanisms that entail adjusting many parameters. For that reason, we also suggest a novel method which utilizes machine learning techniques in Chapter 4. Detailed performance analysis for all the VQA methods proposed in this thesis can be found in Chapter 6.

Chapter 4

Learning the Visual Quality of Static Meshes

We have also developed a machine learning based approach for evaluating the visual quality of 3D static meshes. In this study, our aim is to learn a global perceptual quality metric for triangle meshes using the quality ratings gathered from real observers. We deliberate that this is an elegant way since modeling human visual system processes is a tedious task and requires tuning many parameters, especially in 3D. We employ crowdsourcing methodology for collecting data of quality evaluations.

4.1 Overview

We propose an objective perceptual distance metric for assessing the global visual quality of 3D static meshes. As an alternative to the classical bottom-up approaches, we suggest a data-driven approach in which a quality metric is directly learned from observer evaluations.

Figure 4.1 illustrates the pipeline for the proposed method. First of all, using

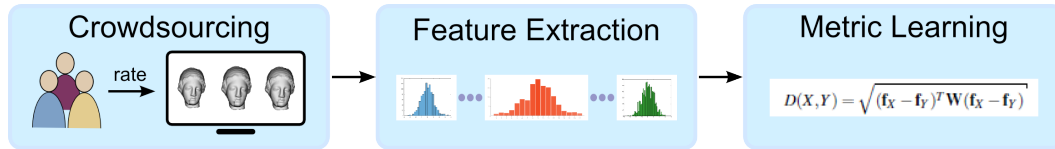


Figure 4.1: Method overview.

crowdsourcing, we collect comparative evaluations of 3D meshes from human observers. Then we extract several descriptive features from the 3D meshes used in the experiment. Lastly, we define a simple distance function and learn the weights of the extracted features on this function through optimization.

4.2 Data Collection through Crowdsourcing

According to our methodology, we first need to collect user evaluations. The most common way of this process is to utilize online crowdsourcing platforms. We chose *Amazon Mechanical Turk (AMT)*¹ as our crowdsourcing platform due to its prevalence, efficiency, and sound documentation. We benefit from the AMT command line tools², which offer a simple and efficient interface to the AMT library.

Using AMT services, one can easily design and conduct simple user tests each of which is called *Human Intelligence Task (HIT)*. AMT supplies much functionality for performing user experiments which allow displaying images and videos. However, it does not provide built-in functionality to show 3D meshes, which is crucial for our experiments. Therefore, we constructed a framework which can display 3D meshes interactively on the web browser by directing AMT server to external pages running *WebGL*³ and *Javascript 3D Library*⁴.

¹<https://www.mturk.com/mturk/welcome>

²<https://requester.mturk.com/developer/tools/clt>

³<http://get.webgl.org/>

⁴<http://threejs.org/>

4.2.1 Data

We used LIRIS/EPFL general purpose dataset [15] for training our model. In this dataset, there are 88 models, between 40K and 50K vertices, which were generated from 4 reference objects: *Armadillo*, *RockerArm*, *Dinosaur*, and *Venus* (Figure 4.2). Two types of distortion, noise addition and smoothing, were applied with different strengths at four locations: on the whole model, on smooth areas, on rough areas, and on intermediate areas. The dataset also provides mean opinion scores (MOS) from 12 observers and 7 static metric results for these models.



Figure 4.2: Meshes used in the AMT experiment.

In our AMT experiments, as the original meshes have high number of vertices, we used 50% simplified versions of the models, with boundary preserving constraint, to prevent possible loading overhead on the client browsers. We applied the *quadric edge collapse decimation* method in MeshLab [102], for simplifying the meshes. Appendix A.1 includes pointer to the actual data we used.

4.2.2 Experiment Design

It is known that human observers are better at providing relative comparisons than making absolute judgments. In our experiments, we preferred triplet design in which three meshes from the same object type with different distortions are presented. The task of the viewer is then to select which of the meshes is more similar to the reference undistorted mesh (displayed in Panel A), in terms of visual quality (Figure 4.3). At the top of the HIT page, we provide a list of guidelines to the subjects explaining that they should consider the spatial distortions in the mesh surface while judging the visual quality. We have opted for a forced choice design with only two options; we have not presented a “None of them” or “Both of them” option since such options are highly abused by lazy turkers who want to make money without effort.

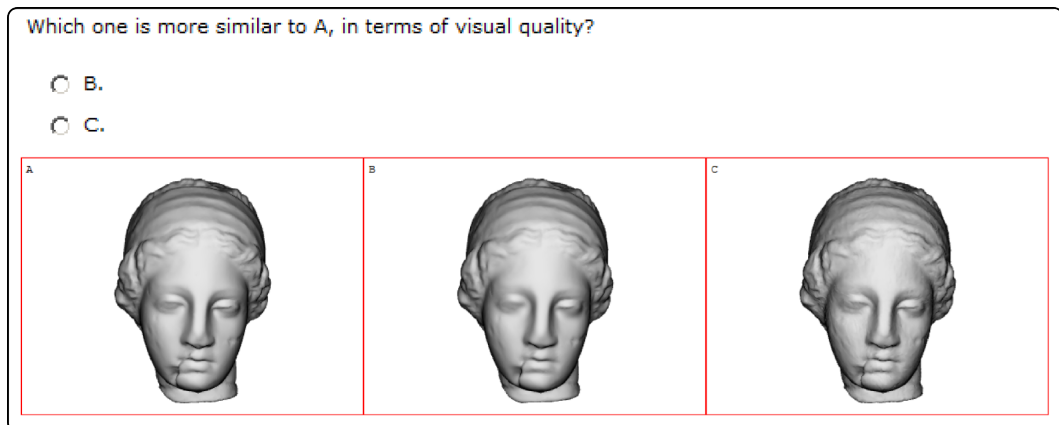


Figure 4.3: Screenshot from our AMT experiment.

The user is able to rotate, zoom in/out, and translate the models and the

user interaction is simultaneous for three models. Panel A always contains the reference model without distortions. This generates 210 triplets for each of the object category and hence 840 triplets in total. We asked two query triplets in each HIT, one of which is a control question with an obvious answer. The duration of each HIT, for which we paid \$0.01, was approximately 3 minutes. Each triplet was evaluated by at least five users, at the end of the experiment.

4.2.3 Reliability Check for the Crowdsourced Data

Data gathered through online crowdsourcing platforms are prone to some reliability concerns; because we do not have full control over the response collection process as it is performed remotely. Thus, in order to assure the reliability of the collected data, we employ several design issues in our AMT experiments, following the suggestions in [103]. Our precautions for the purpose of reliability can be summarized as follows:

- First of all, each user has to take a training session with obvious answers, which facilitates the learning of the test procedure for the user. The users are allowed to proceed to the actual test, only if they answer all the training questions correctly. This training session was easily implemented through the “qualification” facility of the AMT library.
- Secondly, each HIT contains one control question with an obvious answer. If the user fails to answer the control question correctly, that HIT is rejected.
- Lastly, a response time check is performed to identify sloppy participants. In this regard, if the response time for a HIT is shorter than a threshold value, that HIT is also rejected.
- If a user has three or more rejected HITs, we regard him as unreliable and do not include his responses in the final dataset.

At the end of the experiment, 6 subjects were blocked among 134 unique subjects participated in the experiment and the ratio of the rejected HITs over

the total submitted HITs is about 3%.

4.3 Feature Extraction

The main purpose in this step is to extract several features that describe the geometry of the meshes. We have implemented the following geometric attributes that are widely-used for visual quality calculations, in our method. All these attributes are per-vertex, except the global roughness value and eigenvalues. Normalized histograms were calculated for each of the listed per-vertex attribute. Number of histogram bins was defined as 10, empirically. Stacking all these attributes together generates a feature vector of size 201.

- **Geometric position.** Vertex positions with their x, y , and z coordinates are used as simple attributes.
- **Vertex normals.** We have also comprised the vertex normals (x, y , and z coordinates) in the feature vector, as they are important attributes of the mesh geometry and used in several simplification algorithms.
- **Curvatures.** Surface curvature is considered to be related to the visual quality of the mesh, in the literature. Minimum (κ_1), maximum (κ_2), mean $((\kappa_1 + \kappa_2)/2)$, and Gaussian ($\kappa_1 \times \kappa_2$) curvature fields are estimated as in [104]. According to this definition, curvature tensor T for every vertex v for the neighborhood B , approximated by a geodesic disk around this vertex, is calculated as below.

$$T(v) = \frac{1}{|B|} \sum_{edges\ e} \beta(e) |e \cap B|^{-e} \tau_e \quad (4.1)$$

where $|B|$ is the surface area over which the tensor is estimated, $\beta(e)$ is the signed angle between the normals of the faces incident to edge e , $|e \cap B|$ is the length of the intersection of edge e with the region B , and $^{-e}$ is a unit vector in the same direction with e . Eigendecomposition of the tensor field T is used to estimate the minimum and maximum curvatures.

- **Curvature directions.** We have also included the directions of the minimum and maximum curvatures (eigenvectors associated with the minimum and maximum eigenvalues of the curvature tensor field in Eq. 4.1) as x, y, z coordinates, in our feature vector.
- **Shape index and curvedness.** *Shape index* is defined to capture the structure of a surface and it is based on the scaling invariance [105]. For instance, we describe all the spheres as of the same shape regardless of their size.

In conjunction with the shape index notion, *curvedness* refers to the amount of surface curvature. These fields are computed based on the minimum and maximum curvatures, as in Eq. 4.2 and Eq. 4.3, respectively.

$$ShapeIndex = 2/\pi \times \arctan [(\kappa_2 + \kappa_1)/(\kappa_2 - \kappa_1)] \quad (4.2)$$

$$Curvedness = \sqrt{(\kappa_1^2 + \kappa_2^2)}/2 \quad (4.3)$$

- **Surface roughness.** Local roughness for each vertex is defined as the absolute value of the Laplacian of the discrete Gaussian curvature [22]. Firstly, mesh Laplacian matrix is calculated as in Eq. 4.4, with cotangent weights.

$$D_{ij} = \frac{\cot(\beta_{ij}) + \cot(\beta'_{ij})}{2}, \text{ for } j \in N_i^{(V)} \quad (4.4)$$

$$D_{ii} = - \sum_j D_{ij}$$

where $N_i^{(V)}$ is the neighborhood of v_i , and β_{ij} and β'_{ij} are the two angles opposite to the edge constructed by v_i and v_j . Then the local roughness at each vertex is defined as in Eq. 4.5, where GC denotes the discrete Gaussian curvature.

$$LR_i = \left| GC_i + \frac{\sum_{j \in N_i^{(V)}} D_{ij} \cdot GC_j}{D_{ii}} \right| \quad (4.5)$$

Lastly, global roughness is obtained through the accumulation of the local roughness values.

- **Eigenvalues.** As mentioned previously, eigenvalues correspond to the natural vibrations of the mesh surface [99]. We include the first 10 eigenvalues of the mesh Laplacian, in our final feature vector.
- **Mesh Saliency.** Mesh saliency values calculated according to the method by Lee et al. [18], which uses the center-surround Gaussian operator on the mean curvature field, are also included in the initial attempts of our method. However, they are excluded from the final implementation as they do not provide an improvement in the accuracy while saliency calculations amplify the computational cost.

4.4 Metric Learning

Based on the feature vector definition in the previous section and training data gathered through crowdsourcing, we formulate our problem as an instance of metric learning [106]. Metric learning has many successful applications such as measuring illustration style similarity for clip arts [107], 3D shape style similarity [108], compatibility for 3D furniture models [109], and style similarity for infographics design [110]. As a general approach in these studies, an objective function, based on a logistic formulation which expects more noise for relative comparisons with less clear answers, is defined and minimized [107].

More precisely, given two meshes (X and Y) to be compared, let f_X and f_Y be their feature vectors respectively. We define the weighted Euclidean distance between them as in Eq. 4.6. Our goal is then to learn the weights on the diagonal of W , in such a way that the likelihood of observing the training data is maximized.

$$D(X, Y) = \sqrt{(f_X - f_Y)^T W (f_X - f_Y)} \quad (4.6)$$

Given a triplet of meshes $\langle A, B, C \rangle$, we model the probability that the user selects B as more similar to A than C by a sigmoid function (Eq. 4.7).

$$P_{BC}^A = \frac{1}{1 + \exp(D(A, B) - D(A, C))} \quad (4.7)$$

Learning is performed by *Maximum A Posteriori (MAP)* estimation which is acquired by minimizing the objective function in Eq. 4.8, over the set of all training triplets T . The second term in the equation is L_1 regularization term which is added for the purpose of obtaining a sparse feature vector.

$$-\sum_T \log(P_{BC}^A) + \lambda \|diag(W)\|_1 \quad (4.8)$$

This non-linear unconstrained optimization problem is solved by *Sequential Quadratic Programming (SQP)* implementation in Matlab [111], as one of the state-of-the-art numerical solutions for nonlinear optimization. Coefficient λ in Eq. 4.8 is the regularization weight and experimentally set to 0.1, in our implementation. The optimization procedure is initialized by small random weights.

Before the optimization, in the collected data, we have observed that some of the responses are not discriminative in the sense that disagreement between the subjects is high. Such kind of responses do not improve the learning process and introduces computational overhead. Hence, we preprocessed the data collected from the AMT experiment to remove the non-discriminative responses (i.e. 2 of the responses are B and 3 of them are C, or vice versa). About 23% of the tuples were eliminated at the end of this process.

The data collected in the experiment has been randomly split into two parts: 60 % of the data for each mesh category is used for training the model and remaining 40 % of the data is reserved for testing purposes. It took approximately 45 minutes to converge our optimization procedure, in a 3.3 GHz PC.

The optimization ends up with 32 non-zero weights among 201 features, as plotted in Figure 4.4. Too small weights (< 0.02) were also set to 0, manually. Weights of the features related to maximum curvature, mean curvature, Gaussian curvature, curvedness, roughness, and eigenvalues are not shown in the figure because they have almost zero weights. As a remark, this does not mean that zero-weight attributes are totally useless for mesh quality; there could be many different settings of the features that produce a similar result.

We refer the readers to Appendix A.2 for a publicly available dataset including

meshes, data obtained from the crowdsourcing experiment, and learned feature weights.

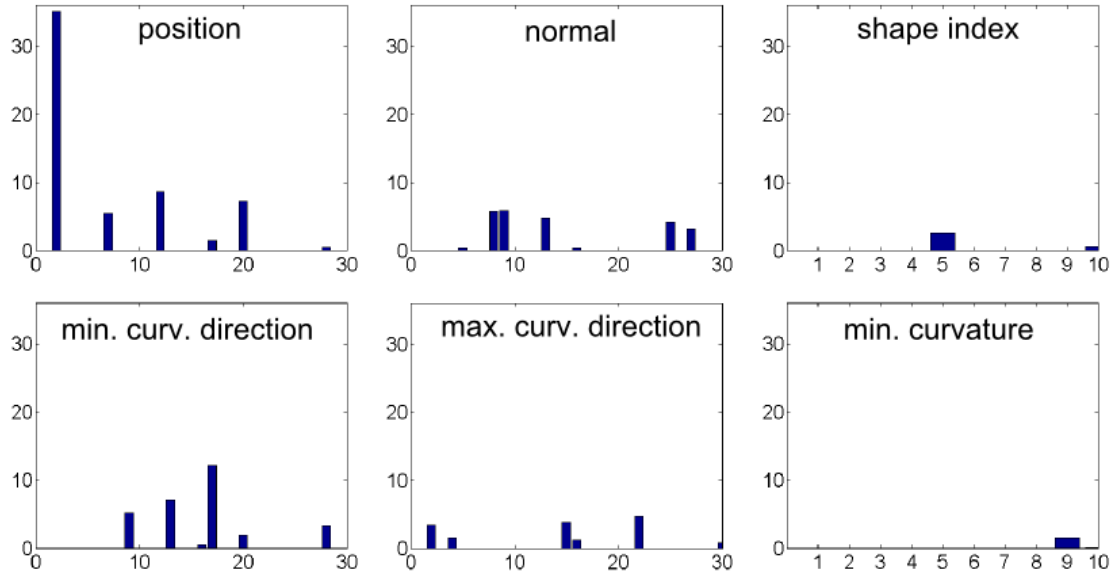


Figure 4.4: Learned weights of the feature vector. (x-axis: Index of the feature, y-axis: weight of the feature.)

4.5 Summary and Discussion

During the implementation of the bottom-up VQA approach proposed in Chapter 3, we have reached the idea that developing a bottom-up model for visual quality evaluation of 3D triangulated meshes is a difficult process. First of all, human visual system is not fully explored and all the theoretical models for explaining the visual perception are designed for 2D. As a result, adapting the models originally devised for 2D on 3D realm is really challenging and requires carefully tweaking a number of parameters.

In contrast, a machine learning approach could be a neater solution for obtaining a perceptual error metric whose parameters are directly learned from ratings of the human observers. Fortunately, the increase in the prevalence of crowdsourcing tools facilitates the data gathering process and leverages the incorporation of

human observers into computation.

Despite the efficiency of the proposed method, there are several limitations. In Section 4.2.3 we have already explained the reliability concerns and our preventive actions. Although we believe that these precautions minimize the bias in the collected data, they may not be sufficient. For instance, we can foresee that a diverse range of viewing parameters and display properties are used in the experiments by the subjects. Since these parameters have significant effect on the perception of visual quality, we are planning to expand this method by applying a multi-scale approach in which features are extracted for several simplification levels of the original mesh. This will improve the robustness of the algorithm by incorporating different levels of detail.

Finally, feature extraction is an important step and may influence the accuracy of the metric rigorously. Thus, new features should be investigated and their effects on the accuracy should be experimented. We also intend to exploit deep learning methods to automatically extract mesh descriptors, instead of using the manually extracted features. Lastly, the model should be trained by a more diverse set of meshes conveying several distortion types, with more participants.

Chapter 5

Enhancing the Perceived Depth Quality

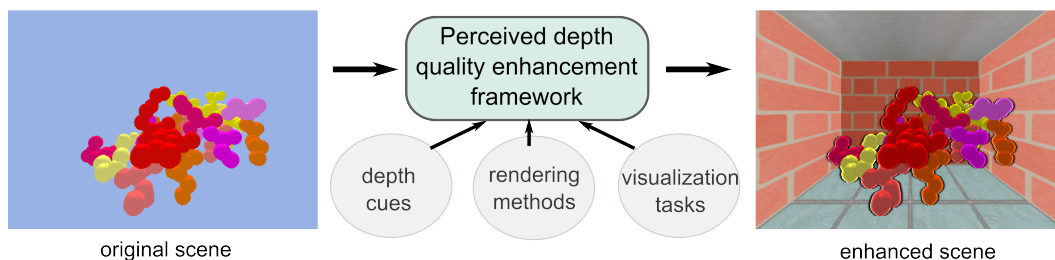


Figure 5.1: Illustration of the proposed framework. (Image from [59], © 2013 ACM, reprinted with permission.)

In this chapter, we propose a framework (Figure 5.1) that automatically selects proper depth enhancement methods for a given scene, depending on the task, spatial layout, and the costs of the rendering methods. The algorithm uses fuzzy logic for determining the significance of different depth cues, and the knapsack algorithm for modeling the tradeoff between the cost and profit of a depth enhancement method.¹

An earlier study by Cipiloglu et al. [58] proposes a generic framework for enhancing the perceived depth quality in 3D scenes; the proposed method improves

¹The figures and text in this chapter are reprinted from [59] with the permission of ACM, with license number 3967280834473.

this framework with a more flexible rendering method selection, adapts the solution for information visualization, and presents a formal experimental study to evaluate the effectiveness of the proposed solution. The contributions of this study are as follows:

- A framework for improving comprehensibility by employing the principles of depth perception in computer graphics and visualization,
- A fuzzy logic based algorithm for determining proper depth cues for a given scene and visualization task,
- A knapsack model for selecting proper depth enhancement methods by evaluating the cost and profit of these methods,
- A formal experimental study to evaluate the effectiveness of the proposed algorithms.

5.1 Overview

We present an algorithm for automatically selecting proper depth cues and the rendering methods that provide these depth cues for a given scene. To determine suitable depth cues for a given scene, we use a hybrid algorithm based on the *cue averaging*, *cue specialization*, *range extension*, and *cue conflict* models described in Section 2.2.1. In this hybrid model, we consider several factors for selecting proper depth cues: the user’s task in the application, the distance of the objects in the scene from the viewpoint, and other scene properties. In addition, we consider the costs of the rendering methods as the main factor in choosing which methods will provide the selected depth cues.

We present the general architecture of our framework in Figure 5.2. The approach first determines the priority of each depth cue based on the user’s task, distance of the objects, and scene properties using fuzzy logic. The next stage maps the high-priority depth cues to the suitable rendering methods that provide

these cues. In this stage, our framework considers the costs of the methods and solves the cost and cue priority tradeoff using a knapsack model. After selecting the proper rendering methods, the last stage applies these methods to the given scene and produces a refined scene.

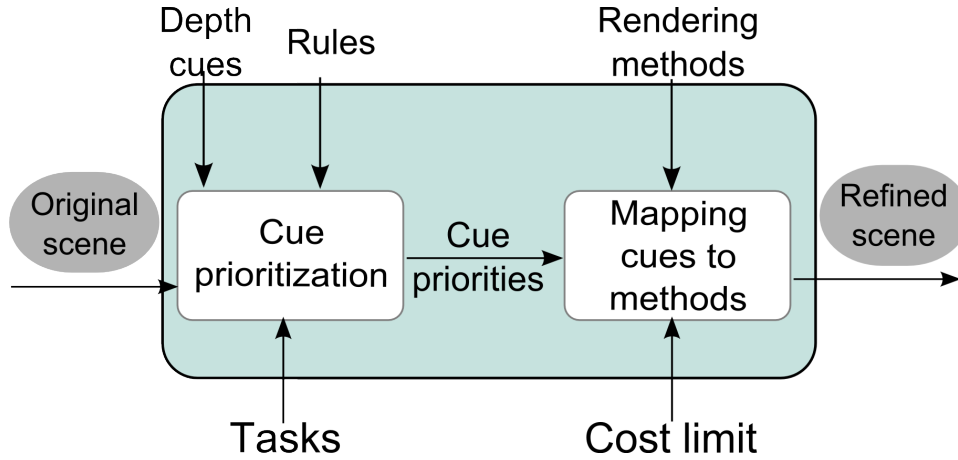


Figure 5.2: General architecture of the system. (Image from [59], © 2013 ACM, reprinted with permission.)

5.2 Cue Prioritization

The purpose of this stage (Figure 5.3) is to determine the depth cues that are appropriate for a given scene. This stage takes as input user’s tasks, the 3D scene, and the considered depth cues. At the end of the stage, a priority value in the range of $[0,1]$ is assigned to each depth cue, which represents the effectiveness of that depth cue for the given scene and task.

In this step, we use fuzzy logic for reasoning. Fuzzy logic is suitable for this task because of its effectiveness in expressing uncertainty in applications that make extensive use of linguistic variables. Because the effects of different depth cues for different scenes are commonly expressed in non-numeric linguistic terms such as “strong”, “effective”, the fuzzy logic approach provides an effective solution for this problem. Furthermore, the problem of combining different depth cues depends on many factors, such as task, distance, etc., whose mathematical

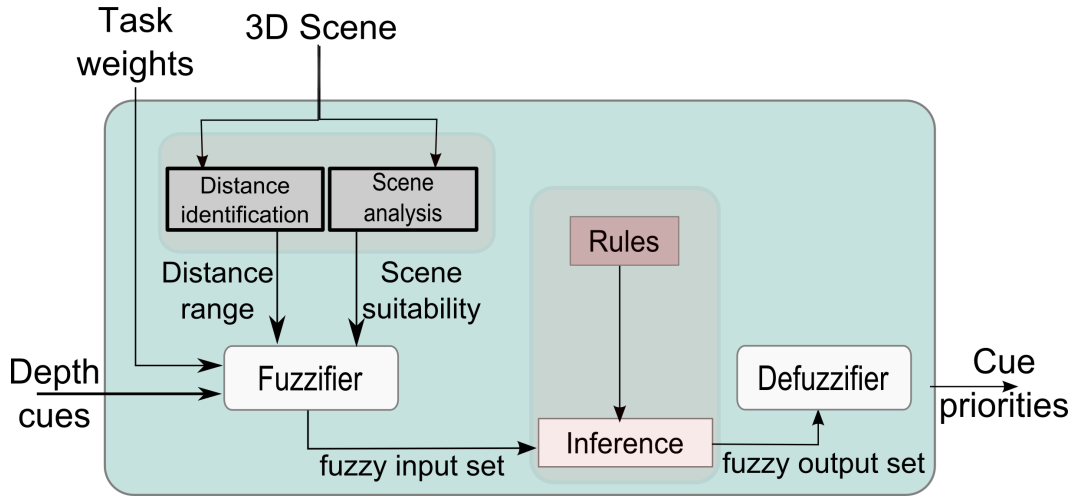


Figure 5.3: Cue prioritization stage. (Image from [59], © 2013 ACM, reprinted with permission.)

modeling is difficult with binary logic. Fuzzy logic has been successfully applied in modeling complex systems related to human intelligence, perception, and cognition [112, 113, 114, 115].

5.2.1 Fuzzification

The goal of this step is to represent the input parameters in linguistic form by a set of fuzzy variables. This step has three types of input variables, related to the user’s task, objects’ distance, and scene properties. First, linguistic variables are defined for each input variable. Then, linguistic terms that correspond to the values of the linguistic variables are defined. Lastly, membership functions are constructed to quantify the linguistic terms.

Task weights are the first input parameters to this stage, based on the *cue specialization* model. These weights represent the user’s task while interacting with the application. Following Ware’s user task classification [57], we define the basic building blocks for the user’s tasks as follows:

- *Judging the relative positions of objects:* In a 3D scene, it is important to

- interpret the relative distances of objects.
- *Reaching for objects*: In interactive applications, it is necessary to allow the user to reach for objects.
 - *Surface target detection*: The proper use of visual cues is important for visualizing the shapes and the surface details of the objects.
 - *Tracing data paths in 3D graphs*: Presenting the data in a 3D graph effectively requires efficient use of 3D space and visual cues.
 - *Finding patterns of points in 3D space*: Interpreting the positions of the points in a 3D scatter plot potentially requires more effort than 2D.
 - *Judging the “up” direction*: In real life, gravity and the ground help us to determine direction; however, an artificial environment generally lacks such cues. Hence, in computer-generated images, it is important to provide visual cues to help determine direction.
 - *The aesthetic impression of 3D space (presence)*: To make the user feel that he is actually present in the virtual environment, the aesthetic impression of the scene is important.
 - *Navigation*: Good visualization allows us to navigate the environment easily, to explore the data.

The weights of the tasks vary depending on the application. For example, in a graph visualization tool, the user’s main task is *tracing data paths in 3D graphs*, whereas in a CAD application, *judging the relative positions* and *surface target detection* are more important. In our algorithm, a fuzzy linguistic variable between 0 and 1 is utilized for each task. These values correspond to the weights of the tasks in the application and are initially assigned by the application developer using any heuristics he desires.

Fuzzification of the task-related input variables is obtained by piecewise linear membership functions, plotted in Figure 5.4 (a). Using these functions and the task weights, each task is labeled as “low_priority”, “medium_priority”, or “high_priority”, to be used in the rule base. The membership functions $\mu_{low_priority}$, $\mu_{medium_priority}$, and $\mu_{high_priority}$ convert x , the crisp input value that corresponds to the weight of the task, into the linguistic terms *low_priority*,

medium_priority, and *high_priority*, respectively.

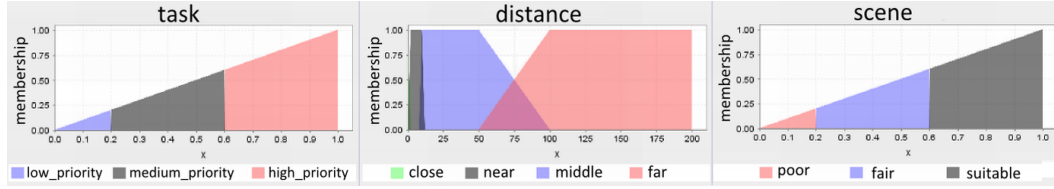


Figure 5.4: Membership functions for fuzzification. (Image from [59], © 2013 ACM, reprinted with permission.)

Distance of the objects from the viewpoint is the second input parameter to the system, as the *range extension* cue combination model is part of our hybrid model. To represent the distance range of the objects, two input linguistic variables, “minDistance” and “maxDistance”, are defined. These values are calculated as the minimum and maximum distances between the scene elements and the view point, and mapped to the range [0-100]. To fuzzify these variables, we use the trapezoidal membership functions (Figure 5.4 (b)), which are constructed based on the distance range classification in [64]. Based on these functions, the input variables for distance are labeled as *close*, *near*, *middle*, or *far*.

The spatial layout of the scene is the third input parameter, which affects the priority of depth cues. For instance, relative height cue is effective for depth perception if the object sizes are not too different from each other [55, Section 24.1.4].

To represent the effect of scene properties, we define an input linguistic variable, “scene_c”, between 0 and 1 for each depth cue *c*. Initially, the value of scene_c is 1, which means that the scene is assumed to be suitable for each depth cue. To determine the suitability value for each cue, the scene is analyzed separately for each depth cue and if there is an inhibitory situation similar to the cases described above, the scene_c value for that cue is penalized. If there are multiple constraints for the same depth cue, the minimum of the calculated values is accepted as the scene_c value.

Table 5.1 demonstrates the scene analysis guidelines that we adapted from the

literature and the list of heuristics used by our system to measure the suitability of the scene for each depth cue. After calculating the scene_c values using the heuristics as described in the table, these values are fuzzified as *poor*, *fair*, or *suitable* with the piecewise linear membership function displayed in Figure 5.4(c).

5.2.2 Inference

The inference engine of the fuzzy logic system maps the fuzzy input values to fuzzy output values using a set of IF-THEN rules. Our rule base is constructed based on a literature survey of experimental studies on depth perception, including [55, 57, 90, 67, 68]. Each depth cue has a different set of rules. According to the values of the fuzzified input variables, the rules are evaluated using the fuzzy operators listed in Table 5.2. Table 5.3 contains sample rules used to evaluate the priority values of different depth cues. The current rule base consists of 121 rules which are available in Appendix A.3.

5.2.3 Defuzzification

The inference engine produces fuzzy output variables with values “strong”, “fair”, “weak”, or “unsuitable” for each depth cue. These fuzzy values should be converted to non-fuzzy correspondences. This defuzzification is performed by the triangular and trapezoidal membership functions (Figure 5.5).

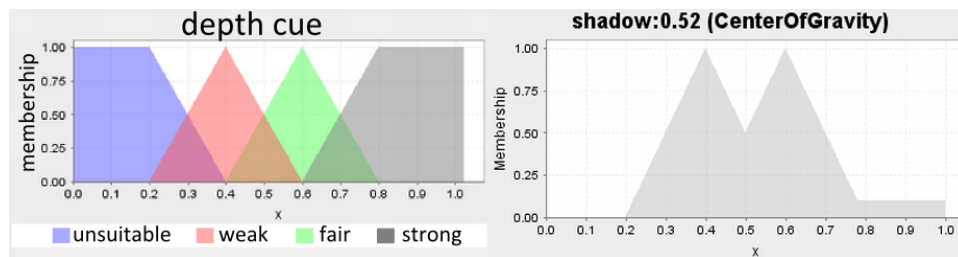


Figure 5.5: Left: Membership functions for defuzzification. Right: A sample fuzzy output of the system for shadow depth cue. (Image from [59], © 2013 ACM, reprinted with permission.)

Table 5.1: Scene suitability guidelines and their formulation in our system.

Scene Suitability Guidelines	Heuristic
<i>Occlusion</i> is a weak cue for finding patterns of points in 3D, if the points are small [90].	$scene_{occlusion} = \frac{numberOfLargePoints}{totalNumberOfPoints}$
If all the points have a constant and large size in a 3D scatter plot, weak depth information is obtained through <i>size gradient</i> cue [57].	$scene_{sizeGradient} = \begin{cases} 0.5, & \text{all the points are large} \\ 0, & \text{otherwise.} \end{cases}$
The object sizes should not be too different from each other, for <i>relative height</i> cue to be effective [55, Section 24.1.4].	$isOutlier(o) = \begin{cases} \text{yes}, & \text{abs}(size_o - avgSize) \geq thr \\ \text{no}, & \text{otherwise.} \end{cases}$ $scene_{relativeHeight} = 1 - \frac{numOfOutliers}{totalNumOfObjects}$
If there is a large number of points in a 3D scatter plot, <i>shadow</i> will not help [90]. Objects should be slightly above the ground [57]. For better perception, a simple lighting model with a single light source is required, with the light coming from above and from one side and infinitely distant [57], [55, Section 24.4.2].	$scene_{shadow} = \min(1 - \frac{totalNumOfObjects}{threshold}, 0)$ $isSlightlyAbove(o) = \begin{cases} \text{yes}, & o_y - ground_y \leq \frac{roomHeight}{3} \\ \text{no}, & \text{otherwise.} \end{cases}$ $scene_{shadow} = \frac{numOfObjectsSlightlyAbove}{totalNumOfObjects}$ The light to produce shadows is appropriately located by the system; there is no need to check the scene according to this constraint.
<i>Perspective</i> is weak if there is a large number of points in a 3D scatter plot [90].	$scene_{perspective} = \min(1 - \frac{totalNumOfObjects}{threshold}, 0)$
When the surfaces are textured, the effect of <i>binocular disparity</i> cue is more apparent [57].	$scene_{binocularDisparity} = \frac{numOfTexturedObjects}{totalNumOfObjects}$

Table 5.2: Fuzzy logic operators.

Operator	Operation	Fuzzy Correspondence
AND	$\mu_A(x) \& \mu_B(x)$	$\min\{ \mu_A(x), \mu_B(x) \}$
OR	$\mu_A(x) \parallel \mu_B(x)$	$\max\{ \mu_A(x), \mu_B(x) \}$
NOT	$\neg\mu_A(x)$	$1 - \mu_A(x)$

Table 5.3: Sample fuzzy rules.

IF scene_{aerial_perspective} is *suitable* AND (minDistance is *far* OR maxDistance is *far*) THEN aerial_perspective is *strong*

IF scene_{binocular_disparity} is *suitable* AND (minDistance is NOT *far* OR maxDistance is NOT *far*) AND aesthetic_impression is *low_priority* THEN binocular_disparity is *strong*

IF scene_{kinetic_depth} is *suitable* AND surface_target_detection is *high_priority* THEN kinetic_depth is *strong*

IF scene_{linear_perspective} is *suitable* AND tracing_data_path_in_3d_graph is *high_priority* THEN linear_perspective is *weak*

IF scene_{motion_parallax} is *suitable* AND (minDistance is NOT *far* OR maxDistance is NOT *far*) AND navigation is *high_priority* THEN motion_parallax is *strong*

IF scene_{shading} is *suitable* AND surface_target_detection is *high_priority* THEN shading is *strong*

IF scene_{shadow} is *suitable* AND tracing_data_path_in_3d_graph is *high_priority* THEN shadow is *weak*

We use the “center of gravity” (COG) function as the defuzzification algorithm:

$$U = \left(\int_{min}^{max} u \mu(u) du \right) / \left(\int_{min}^{max} \mu(u) du \right) \quad (5.1)$$

where U is the result of defuzzification, u is the output variable, μ is the membership function after inference, and min and max are the lower and upper limits for defuzzification, respectively [116].

The second plot in Figure 5.5 exemplifies the result of a sample run of the system for the shadow depth cue. In the figure, shaded regions belong to the fuzzy output of the system. This result is defuzzified using the COG algorithm and the final priority value for the shadow cue is calculated as the center of gravity of this region.

5.3 Mapping Selected Depth Cues to Rendering Methods

After determining the cue priority values, the next issue is how to support these cues using rendering methods. Numerous rendering methods have been proposed for visualization applications, and providing an exhaustive evaluation of these techniques is a considerable task. Therefore, this study presents our implementation of the most common techniques, although the framework can be extended with new rendering methods.

Table 5.4 presents the depth cues and rendering methods in our system. Note that there are generally a number of rendering methods that correspond to the same depth cue. For example, the *linear perspective* cue can be conveyed by *perspective projection*, inserting a *ground plane*, or *texture mapping* of the models. On the other hand, the same rendering method may provide more than one depth cue. For instance, *texture mapping* enhances both the *texture gradient* and *linear perspective* cues.

The problem in this stage is to find the rendering methods that provide important depth cues with the minimum possible cost. Figure 5.6 illustrates this mapping stage. The inputs to the system are the cue priority values from the previous stage and a cost limitation from the user. This stage consists of four internal steps: first, suitable rendering methods are selected based on a cost-profit analysis. Then, the parameters of the selected rendering methods are determined according to the cue weights. The third step is cue conflict resolution, in which

Table 5.4: Rendering methods for providing the depth cues.

Depth Cues	Rendering Methods
Size gradient	Perspective projection
Relative height	Perspective projection, dropping lines to ground, ground plane [57]
Relative brightness	Proximity luminance covariance [57], fog [117]
Aerial perspective	Fog, proximity luminance covariance, Gooch shading [72]
Texture gradient	Texture mapping, bump mapping [117], perspective projection, ground plane, room
Shading	Gooch shading, boundary enhancement [76], ambient occlusion [71], texture mapping
Shadow	Shadow map [117], ambient occlusion
Linear perspective	Perspective projection, ground plane, room, texture mapping
Depth of focus	Depth-of-field [70], multi-view rendering [77]
Accommodation	Multi-view rendering [118]
Convergence	Multi-view rendering [118]
Binocular disparity	Multi-view rendering
Motion parallax	Face tracking [119], multi-view rendering
Motion perspective	Mouse/keyboard-controlled motion

possible cue conflicts are avoided. If the *multi-view rendering* method is among the selected methods, a stereo refinement process is performed in this step to determine the stereo camera parameters that promote 3D perception and mitigate eye strain due to cue conflicts. Last, the selected rendering methods are applied to the original scene with the determined parameters.

5.3.1 Method Selection

This step (Figure 5.7) models the tradeoff between the cost and profit of a depth enhancement method. Cue priorities from the previous stage, current rendering time in milliseconds per frame from the application, and a maximum allowable rendering time from the user are taken as input. Then the maximum cost is calculated as the difference between the maximum and current rendering times.

We formulate the method selection decision as a *knapsack problem* in which each depth enhancement method is assigned a “cost” and a “profit”. Profit corresponds to a method’s contribution to depth perception and is calculated as the weighted sum of the priorities of the depth cues provided by this method

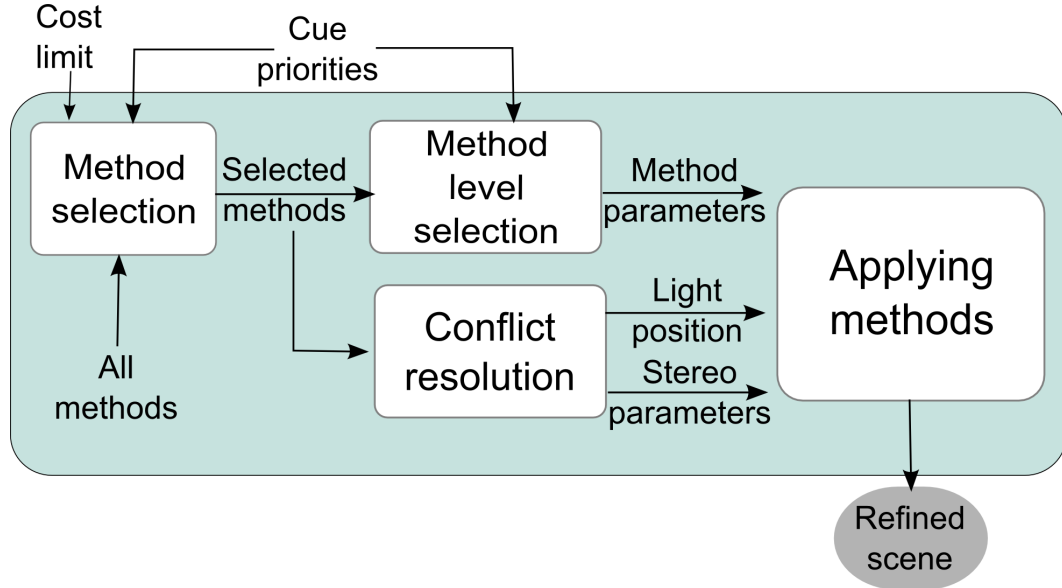


Figure 5.6: Cue to rendering method mapping stage. (Image from [59], © 2013 ACM, reprinted with permission.)

(Eq. 5.2), based on the “cue averaging” model:

$$profit_i = \sum_{j \in C_i} c_{ij} \times p_j \quad (5.2)$$

where C_i is the set of depth cues provided by method i ; p_j is the priority value of cue j ; and c_{ij} is a constant. The functionality of c_{ij} is as follows: If more than one rendering method provide the same depth cue, they do not necessarily provide it in the same way and same strength. For instance, *aerial perspective* cue can be provided by *fog*, *proximity luminance*, and *Gooch shading* methods; but the effects of these methods are different. With increasing distance, the *fog* and *proximity luminance* methods reduce contrast and saturation, whereas *Gooch shading* generates blue shift. We handle these differences by assigning heuristic weight (c_{ij}) between $[0, 1]$ to each method, which determine the contribution of method i to cue j . For example, since *aerial perspective* is generally simulated with the reduction in contrast [120], we assign higher weights to *fog* and *proximity luminance* methods than *Gooch shading* for this cue.

A method’s cost is calculated as the increase in a frame’s rendering time due to this method. Using the dynamic programming approach, we solve the knapsack problem in Eq. 5.3, which maximizes the total “gain” while keeping the total cost

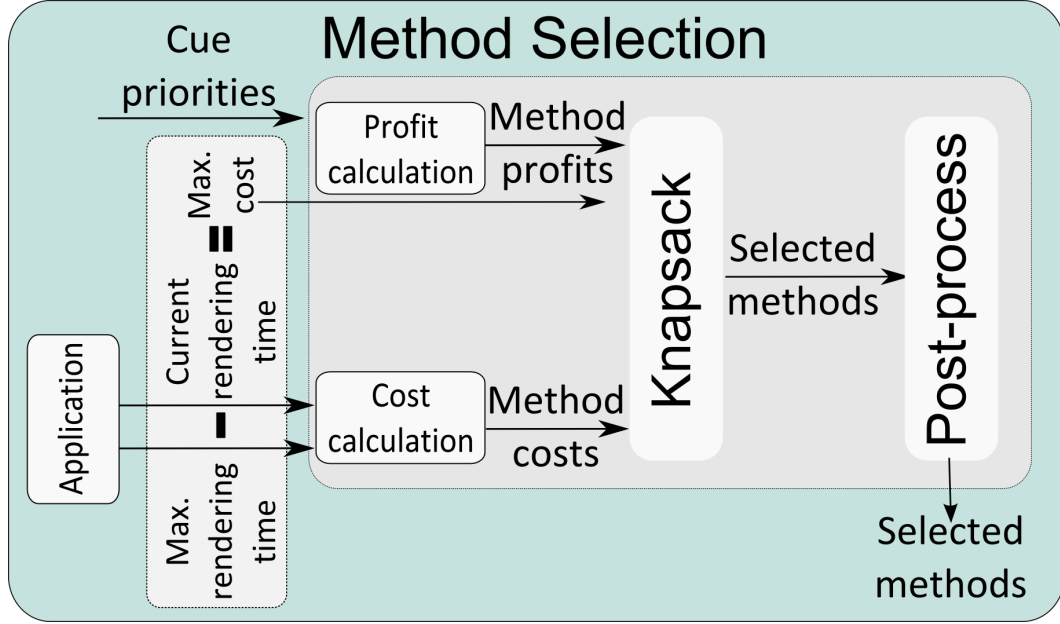


Figure 5.7: Method selection step. (Image from [59], © 2013 ACM, reprinted with permission.)

under the “maxCost”:

$$\begin{aligned}
 Gain &= \sum_{i \in M} profit_i \times x_i \\
 Cost &= \left(\sum_{i \in M} cost_i \times x_i \right) \leq maxCost
 \end{aligned} \tag{5.3}$$

where M is the set of all methods; $maxCost$ limits the total cost; $profit_i$ is the profit of method i ; $cost_i$ is the cost of method i ; and $x_i \in \{0, 1\}$ is the solution for method i and indicates if method i will be applied.

After this cost-profit analysis, we apply a post-processing step to the selected methods. First, we eliminate unnecessary cost caused by the methods that only provide cues already given by more profitable methods. For instance, *multi-view rendering* provides the *depth-of-focus* and *binocular disparity* cues. Hence, if the *depth-of-field* method, which only provides the *depth-of-focus* cue is selected, we can deselect it because a more profitable method *multi-view rendering* is also selected. Second, we check for dependencies between methods. As an example, if *shadow mapping* is selected without enabling *ground plane*, we select *ground plane* and update the total cost and profit accordingly.

5.3.2 Method Level Selection

In our framework, several rendering methods can be controlled by parameters. If a rendering method is essential for improving depth perception, depending on the cues it provides, it should be applied with more strength. In certain cases, applying the methods in full strength will not increase the rendering cost (e.g. *proximity luminance* strength). Even in these cases, we allow for choosing the parameters for less strength, because if all the methods are very apparent, some of them may conceal the visibility of others, thus leading to possible cue conflicts. Moreover, it is suitable to provide a logical mapping between the cue weights calculated in the previous stage and the method strengths. In this regard, we calculate the importance of each method as the weighted linear combination of the priority values of the cues provided by the corresponding method (Eq. 5.4).

$$importance_i = \sum_{j \in C_i} c_{ij} \times p_j \quad (5.4)$$

where C_i is the set of all depth cues provided by method i ; p_j is the priority value of cue j ; and c_{ij} is a constant between 0 and 1 that represents method i 's contribution to cue j .

These method importance values are used to determine the strength of the application of that method. To disallow an excessive number of modes for method parameters, we select from predefined method levels according to the importance:

$$level_i = \begin{cases} weak, & [3 \times importance_i] = 0 \\ moderate, & [3 \times importance_i] = 1 \\ strong, & [3 \times importance_i] = 2 \end{cases} \quad (5.5)$$

Table 5.5 shows the list of rendering methods that can be applied in different levels and the parameters that control the strength of these methods. Note that changing these parameters does not affect the rendering costs of the methods. Therefore, we apply this level selection step after the cost-profit analysis.

Table 5.5: Rendering methods with levels.

Rendering Method	Strength Parameter
Proximity luminance covariance	λ : The strength of the method. (See Table 5.6.)
Boundary enhancement	λ : The strength of the method. (See Table 5.6.)
Fog	<i>Fog type</i> : Linear, exponential, or exponential squared.
Gooch shading	<i>Mixing factor</i> : The amount that determines in what proportion the gooch color will be mixed with the surface color.
Depth of field	<i>Blur factor</i> : The strength of the blur filter.

5.3.3 Cue Conflict Avoidance and Resolution

We also aim to resolve possible cue conflicts. Identifying conflicts, however, is not straightforward. There are several cases in which we may encounter cue conflicts.

If the light position is not correct, the shadow cue may give conflicting depth information. For this reason, we locate the point light source in the left-above of the scene, since the HVS assumes that light originates from left-above [55, Section 24.4.2].

Another important and problematic cue conflict is convergence-accommodation. In the HVS, the accommodation mechanism of the eye lens and convergence are coupled to help each other. In 3D displays, however, although the apparent distances to the scene objects are different, all objects lie in the same focal plane: the physical display. This conflict also results in eye strain. In most current screen-based 3D displays, it is not possible to completely eliminate the convergence-accommodation conflict; extra hardware and different display technologies are needed [121], discussion of which is beyond the scope of this work.

There are several methods to reduce the effect of convergence-accommodation cue conflict. Cyclopean scale is one method and it provides a simple solution, scaling the virtual scene about the midpoint between the eyes to locate the nearest part of the scene just behind the monitor [118]. Note that cyclopean scale does not change the overall sizes of the objects in the scene, because it scales the scene around a single viewpoint. In this way, the effect of accommodation-convergence conflict is lessened, especially for the objects closer to the viewpoint. This technique also reduces the amount of perceived depth; to account for this,

we decrease the profit of the *multi-view rendering* method in proportion to the cyclopean scale amount.

Another possible stereoscopic viewing problem is diplopia, which is generally caused by incorrect eye separation. The apparent depth of an object can be adjusted with different eye separation values. As the virtual eye separation amount increases, perceived stereoscopic depth decreases. We adjust virtual eye separation through Eq. 5.6 [57], which calculates the separation in centimeters using the ratio of the nearest point to the furthest point of the scene objects. This dynamic calculation allows large eye separation for shallow scenes and smaller eye separation for deeper scenes.

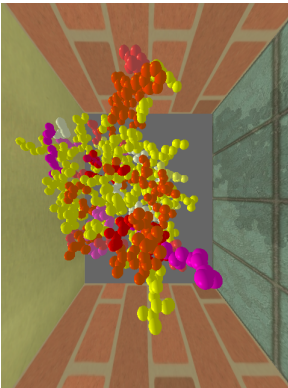
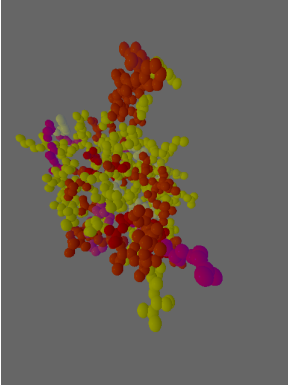
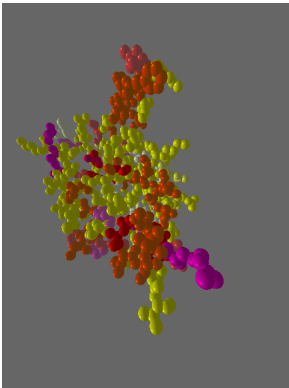
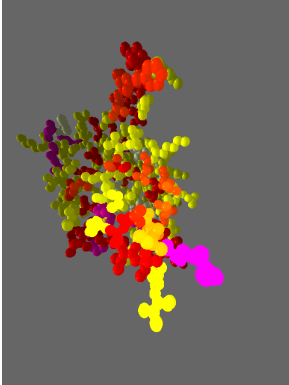
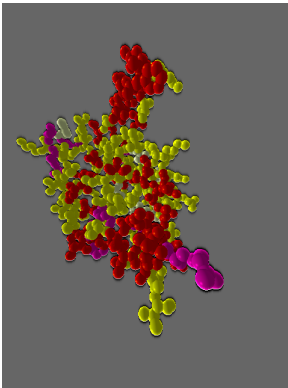
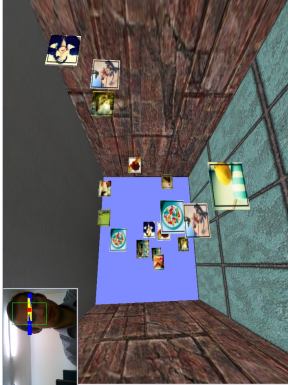
$$VirtualEyeSep = 2.5 + 5.0 \times (NearPoint/FarPoint)^2 \quad (5.6)$$

As the output of this step, we produce the position of the shadow-caster light and stereo-viewing parameters: virtual eye separation and cyclopean scale factor, to be used in the method application step. In this fashion, we avoid possible cue conflicts and decrease the problems with stereoscopic viewing such as eye strain. As a further improvement to our system, attention-aware disparity control model proposed by Celikcan et al. [122], which optimizes the stereo camera parameters to promote viewing comfort, can be adapted for our framework.

5.3.4 Rendering Methods

After determining suitable rendering methods, we apply them to the scene with the calculated parameters. Our current implementation supports the methods in Table 5.4; the effects of some of these methods are illustrated in Table 5.6 via a protein model (except for face tracking).

Table 5.6: Some of the rendering methods in our system.

Rendering Method	Summary	Rendering Method	Summary
Shadow map 	<p>Provided cues: Shadow. We perform a depth test from the light’s point of view; the points that cannot pass the depth test should be in shadow [117].</p>	Gooch Shading 	<p>Provided cues: Shading, aerial perspective. This is a non-photorealistic shading model in which the color of the object is interpolated between cool colors and warm colors according to the light direction [72].</p>
Fog 	<p>Provided cues: Aerial perspective, relative brightness. The final color of each pixel is interpolated between the surface color and the fog color according to their distance from the viewpoint.</p> $c_{final} = f \times c_{surface} + (1 - f) \times c_{fog}$	Proximity Luminance Covariance 	<p>Provided cues: Relative brightness, aerial perspective. The luminance and saturation of the objects are changed according to their distance from the viewpoint.</p> $\forall p \in P, L'_p = \lambda \times eyeDist_p^2 \times L_p$
Boundary Enhancement 	<p>Provided cues: Shading, occlusion. We enhance the color contrast in the scene’s spatially important regions [76]. The difference between Gaussian filtered and original depth buffer values gives the spatial importance function (ΔD).</p> $\Delta D = G * D - D$ $\forall p \in P, R_p, G_p, B_p = R_p, G_p, B_p + \Delta D_p \cdot \lambda$	Face Tracking 	<p>Provided cues: Motion parallax, motion perspective. We use the face position to determine the viewpoint [119].</p>

5.4 Summary and Discussion

The proposed framework can be used for automatically enhancing the comprehensibility of the visualization, or as a component to suggest suitable rendering methods to application developers. Our framework is generic and can easily be extended by changing the existing rule base. Thus, it can be used for experimenting with several different depth cue combinations and new rendering methods.

In the current implementation, we consider the cost in terms of rendering time because we target interactive visualization applications. Another cost consideration concerns motion parallax, which is an important cue for visualization tasks and requires real-time rendering of the methods. It is also possible to consider alternative cost metrics, such as memory requirements or visual clutter measurements. Visual clutter is particularly undesirable for visualization tasks, and several methods have recently been proposed to measure it in a visualization [123, 124, 125].

In addition, extending the system to use the multiple constrained knapsack problem will allow considering multiple cost limitations at the same time. Additionally, the current system is designed to operate globally, which means that the objects in the scene are not analyzed individually. It may be suitable to extend the system to consider each object in the scene individually and apply depth enhancement methods to only the objects that need them.

We use a heuristic approach for analyzing the scene suitability for each depth cue. Developing a probabilistic approach for a more principled formulation of the guidelines in the scene analysis step may yield better results. Likewise, due to the nature of fuzzy logic, we use membership functions that are manually defined. Although we use the findings from empirical studies while defining these functions, tuning may be required for a better fit. Furthermore, during the realization of the depth cues, we favor making the data in the visualization comprehensible rather than realistic, as Ware [90] suggests. For instance, our method exaggerates the atmospheric contrast artificially to enhance the sense of depth.

Chapter 6

Evaluation

In this chapter we present the experimental validation of the proposed methods in three sections. In the first section, user evaluation of the VQA metrics described in Chapter 3 is given. The second section contains the experimental results for the machine learning based VQA method in Chapter 4 and results of the perceived depth quality enhancement framework proposed in Chapter 5 are elaborated in the last section.

6.1 Visual Quality Assessment of Dynamic Meshes

6.1.1 User Experiment

We conducted subjective user experiments to evaluate the fidelity of our voxel-based visual quality metric. In this section, we explain the experimental design and analyze the results. The subjective evaluation results in this study are publicly available as supplementary material (See Appendix A.1).

Table 6.1: Information about the meshes.

	<i>Camel</i>	<i>Elephant</i>	<i>Hand</i>	<i>Horse</i>
# <i>vertices</i>	21885	42321	7997	8431
# <i>faces</i>	43777	84638	15853	16853
# <i>frames</i>	42	48	45	48

6.1.1.1 Data

We used four different mesh series in the experiments. The original versions of these animated meshes (Figure 6.1) are obtained from public datasets [126] and [127]; and information about these meshes are given in Table 6.1. The animations are continuously repeated and the playback frame rate is 60 frames/second for the sequences. For the modified versions of the animated meshes, we applied random vertex displacement filter on each frame of the reference meshes, using MeshLab tool [102]. The only parameter of this filter is the maximum displacement which we set as 0.1. The vertices are displaced with a vector whose normal is bounded by this value. This corresponds to adding random noise on the mesh vertices.

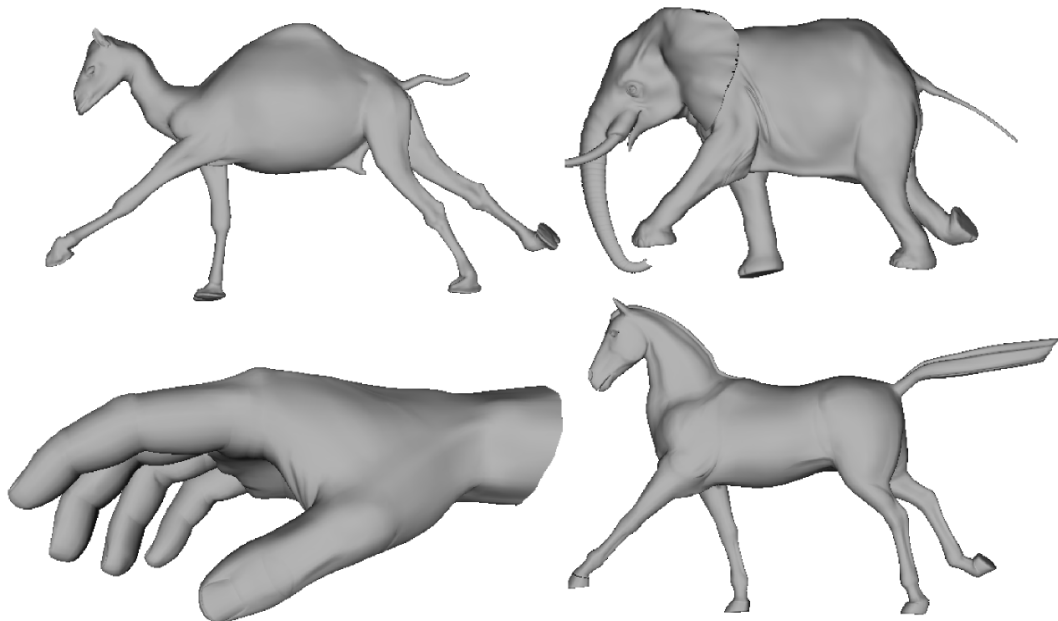


Figure 6.1: Sample frames from the reference animations.

6.1.1.2 Experimental Design

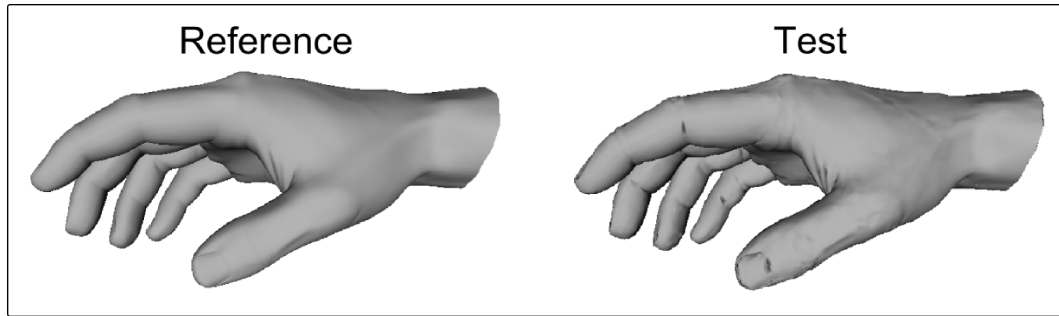
In this experiment, our aim is to measure the correlation between the subjective evaluation and the proposed metric results. The subjects in the experiment evaluated the perceived quality of the animated meshes by marking the perceived distortions on the mesh. For the experiment setup, we used simultaneous double stimulus for continuous evaluation (SDSCE) methodology among the standards listed in [6]. According to this design, presenting both stimuli simultaneously eliminates the need for memorization.

Task. In the experiments, we used two displays; one for viewing the animations and the other for evaluation. In the viewing screen (Figure 6.2a), both the reference and test meshes were shown in animation and the interaction (rotating and zooming) was simultaneous.

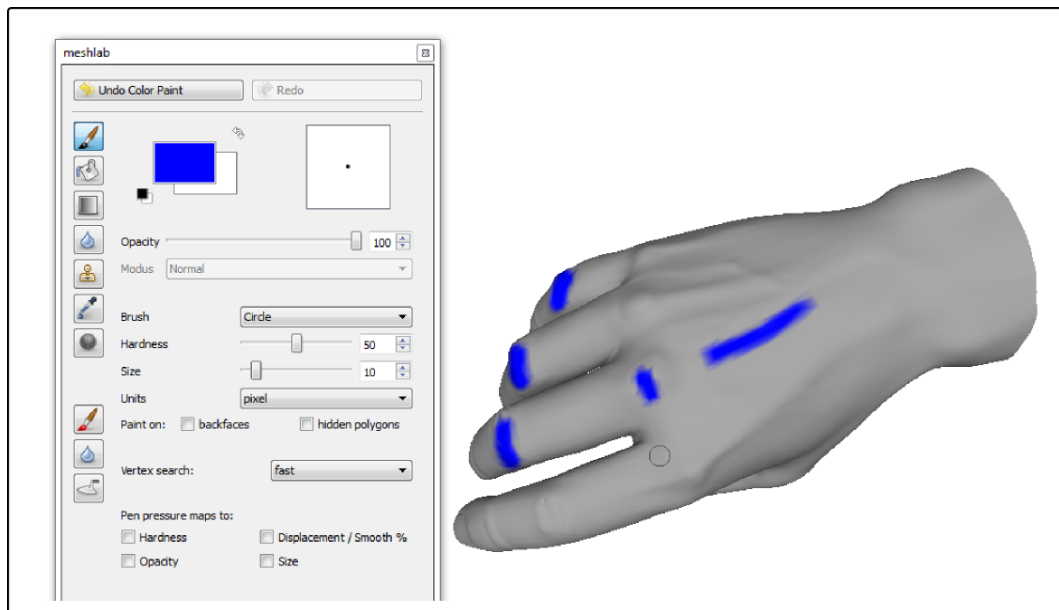
In the evaluation screen (Figure 6.2b), a marking tool with tip intensity was supplied to the user. The user’s task was to mark the visible distortions. The task of annotation would be very difficult if it was performed on dynamic state. Therefore, the users marked the visible distortions on a single static frame, selected manually (frames in Figure 6.1). One may argue that marking the distortions on static state may introduce bias. We try to minimize this effect in two ways. First of all, the annotation was done on a sample frame of the reference animation instead of the modified animation. In this way, the distortions were never seen in static mode by the observers. Secondly, the user was still able to view both of the animations and manipulate the view-point simultaneously in the viewing screen, during the evaluation. This eliminates the necessity for memorization.

At the beginning of the experiments, subjects were given the following instruction: “A distortion on the mesh is defined as the spatial artifacts, compared to the reference mesh. Consider the relative scale of distortions and mark the visible distortions accordingly, using the intensity tool.”

Setup. The environment setup in the experiments has a significant impact on



(a) Viewing Screen.



(b) Evaluation Screen.

Figure 6.2: Experimental setup.

the results. Therefore, the parameters such as lighting, materials, stimuli order should be carefully designed [6]. We explain each parameter below.

- *Viewing Parameters:* The observers viewed the stimuli on a 19-inch display from 0.5m away the display.
- *Lighting:* We use a stationary left-above, center directed lighting [40].
- *Materials and Shading:* To prevent highlighting effects and accentuate distortions unpredictably, we used Gouraud shading in the experiments. Moreover, we used meshes without texture.

- *Animation and Interaction:* Free-viewpoint was enabled to the viewers for interaction. Furthermore, since inspection of the mesh during paused state was contradictory to the purpose of the experiment, two different displays were used and the evaluation of the mesh was conducted on one of the screens while the animation is ongoing on the other screen.
- *Stimuli order:* Each modified and reference mesh combination was presented in a random order allowing for more accurate comparisons. In other words, there was not a specific ordering of the meshes and subjects were also able to pause their evaluation and continue whenever they want.

Subjects. Twelve subjects with various levels of computer experience participated in the experiment. All of the subjects evaluated every animated mesh in the experiment.

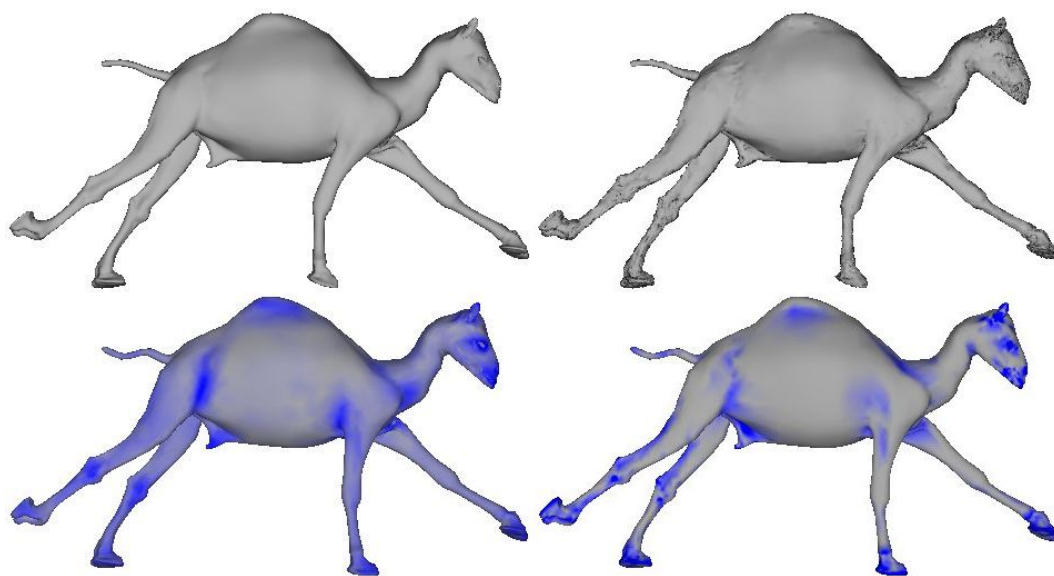
6.1.1.3 Results and Discussion

The mesh frames that were marked by the subjects were stored as vertex color maps. To unify the responses of each subject for each mesh, we calculate a mean subjective response using Eq. 6.1.

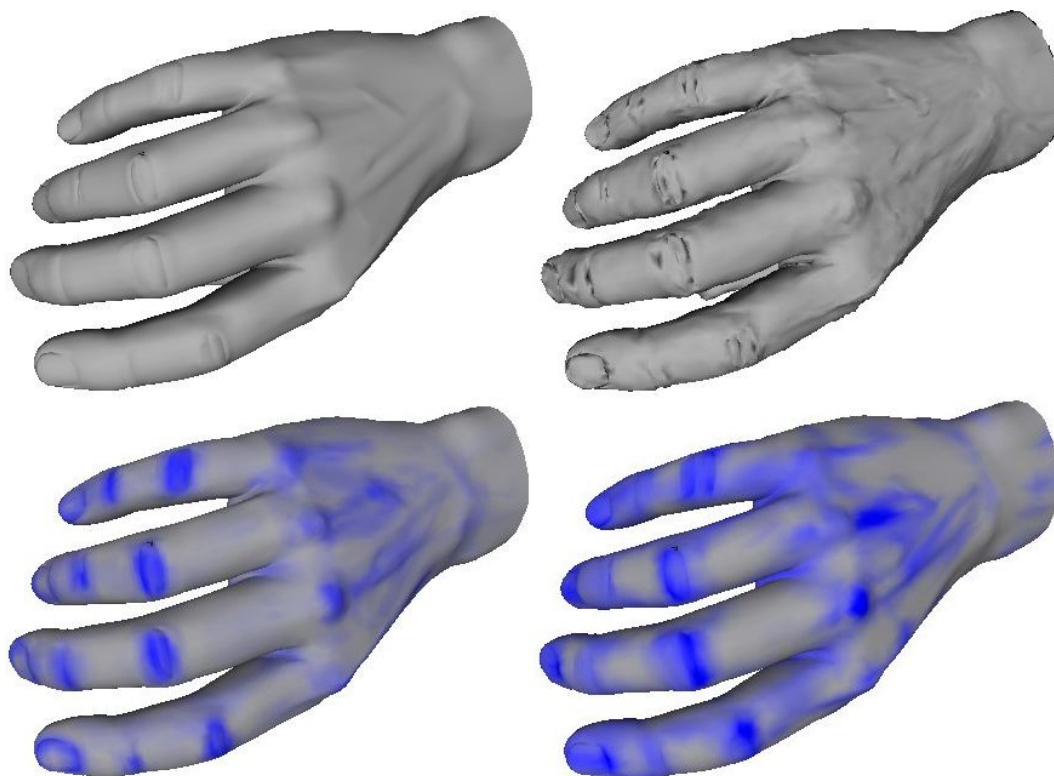
$$\mu(v_{i,M'}) = \frac{\sum_{s=1}^N R_s(v_{i,M'})}{N} \quad (6.1)$$

where N is the number of subjects who evaluated the mesh M' , $R_s(v_{i,M'})$ represents the given response to a single vertex v_i , mesh M' and subject s combination. Figure 6.3a and 6.3b portray sample results from the experiment along with the reference and modified mesh pair and the output of our algorithm.

Next, we compare the mean subjective responses with our proposed method’s predictions. For this purpose, we use two common methods for correlation: Pearson linear correlation coefficient (r) for prediction accuracy, and Spearman rank order correlation coefficient (ρ) for monotonicity between the mean subjective response and estimated metric output [12].



(a) Camel mesh.



(b) Hand mesh.

Figure 6.3: *Top-left*: reference mesh, *top-right*: modified mesh, *bottom-left*: mean subjective response, *bottom-right*: estimated visual response. Blue regions in the mean subjective response and estimated response maps demonstrate the high perceptual differences.

Table 6.2: Pearson (r) and Spearman (ρ) correlation coefficients for each mesh.

	<i>Pearson_r</i>	<i>Spearman_ρ</i>	<i>Strength</i>
<i>Camel</i>	84%	83%	High
<i>Elephant</i>	59%	65%	Modest
<i>Hand</i>	72%	71%	High
<i>Horse</i>	71%	70%	High
<i>Overall</i>	71%	72%	High

Notice that correlation coefficients vary in the range of $[-1,1]$ and a negative coefficient indicates a negative correlation while positive coefficient means a positive correlation. While interpreting the correlation analysis, we used the categorization in [128], where correlation coefficients (in absolute value) which are ≤ 0.35 are considered as low or weak correlations, $0.36 \leq r, \rho \leq 0.67$ modest or moderate correlations, and $0.68 \leq r, \rho \leq 1$ strong or high correlations.

While measuring the correlation, we considered the limitations of the paint tool, in which subjects may unintentionally mark some region nearby the region they actually target. To reduce the effect of this problem, we followed the approach used in image/video quality assessment validations where image or video frame is divided into a regular grid and the comparison is done tile by tile [43]. Based on this idea, we grouped the nearby vertices and find the correlation based on the average intensity of these regions. We asked a designer to segment the mesh manually using a paint-based interface, although any available mesh segmentation technique could also be used for this purpose [129]. The designer was instructed to create about 50 segments for each model.

Table 6.2 includes the correlation coefficients for each mesh and when all the samples are combined (overall). Both Pearson and Spearman correlation analysis give consistent results. However, Spearman’s correlation could be more reliable in our case, because a darker mark in user responses indicates a higher distortion; yet, it is a subjective issue to decide on which intensity corresponds to which distortion amount. Hence, finding a correlation between the rank orders of the vertices rather than the absolute color values is more appropriate.

As the table indicates, the average correlation is about 70%, which can be considered as a promising result for the field of local dynamic mesh quality assessment. Correlation coefficients for *Camel*, *Hand*, and *Horse* meshes are high, while *Elephant* mesh exhibits a moderate correlation.

One important issue that affects the results negatively is that the subjects tend to evaluate only certain views of the meshes. Eight of the subjects reported that they had generally marked the meshes from the side views. In addition, since the meshes are known objects, visual attention principles may have come into play and our metric does not reflect this mechanism.

6.1.2 Comparison to State-of-the-art Techniques

6.1.2.1 Comparison to Static Metrics

It is required to compare the performance of our method with the current state-of-the-art techniques. However, there is no publicly available dataset including the user evaluations for dynamic meshes, to the best of our knowledge. Therefore, we compared our metric to the static metrics using the public LIRIS/EPFL general purpose dataset [15]. The description of the dataset can be found in Section 4.2.1.

Since our method is also applicable for static meshes, we ran our algorithm on these models by setting velocities to 0. Although our aim is to produce a 3D map as output, to be able to compare our metric to the other techniques, we used the average of the vertex probabilities in the output map as the overall score of the mesh quality. These scores are in the range of 0-1 and a high score indicates that the distortions on this mesh are highly visible.

Figure 6.4 includes several examples from the *Venus* model. MOS values of the highly noisy objects in (b) and (c) are higher than the smoothed object in (d). This is intuitive as the smooth model seems less distorted than the noisy object. Our metric conforms to this situation since the metric outputs for (b) and (c) are higher than the output for (d). According to the subjective evaluations,

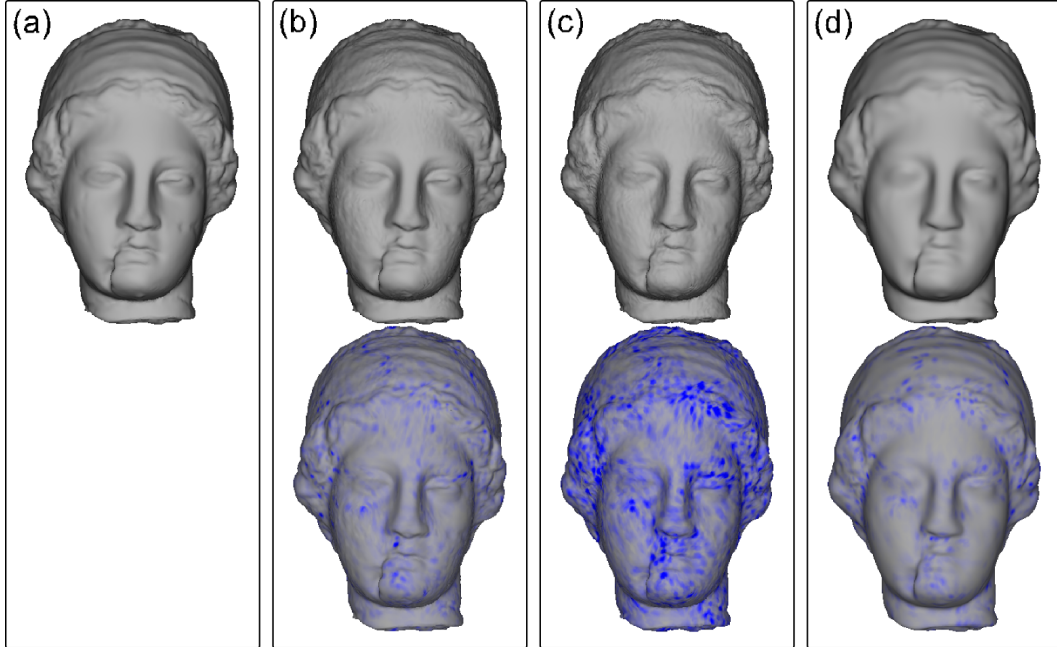


Figure 6.4: Top row: original models. Bottom row: Metric outputs. (a) Original model (b) High noise on smooth regions (MOS = 8.80, MSDM = 0.64, Our metric = 0.69) (c) High noise on the whole object (MOS = 9.40, MSDM = 0.70, Our metric = 0.85), (d) High smoothing on the whole object (MOS = 8.10, MSDM = 0.58, Our metric = 0.54).

model in (c) exhibits the highest distortion as also reflected by our method. Our results show similarity between the results of the MSDM metric as well.

Figure 6.5 provides MOS vs. our metric estimation plots for each object in the dataset. Spearman correlation coefficients between MOS values and each of the provided metric results were also calculated as listed in Table 6.3. We haven't included the results for pure geometric metrics RMS and Hausdorff Distance since they are quite low. According to these results, our metric well correlates with the subjective responses and it is superior to the most of the static metrics.

6.1.2.2 Comparison to Dynamic Metrics

Perceptual error metrics designed for dynamic meshes to date that we are aware of are [27] and [28]. However, dynamic mesh datasets of [27] and [28] provide only

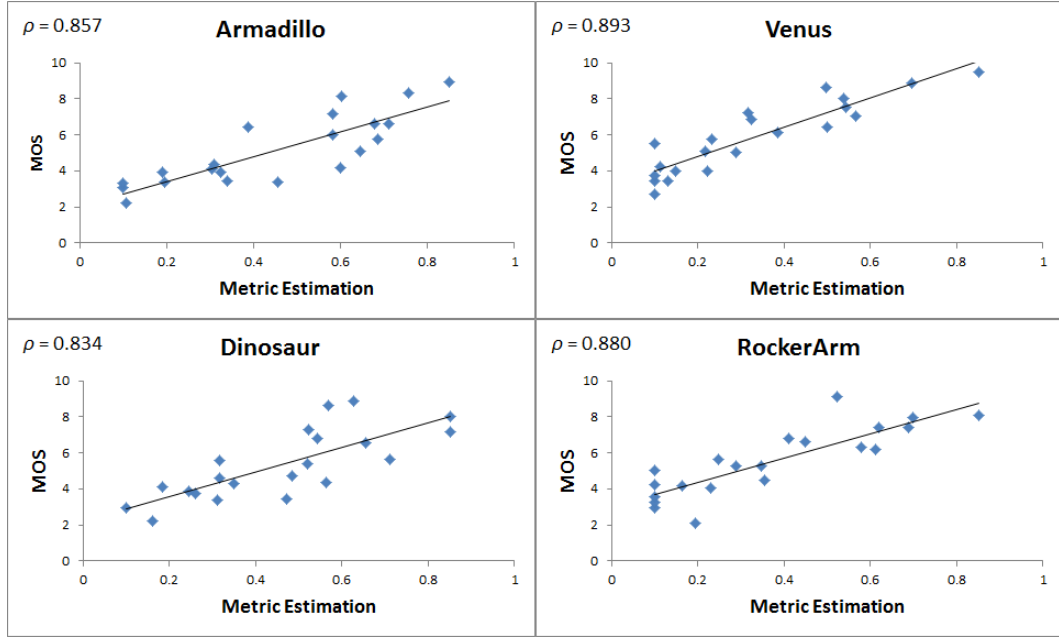


Figure 6.5: Subjective MOS vs. our metric estimation for each model. Spearman correlation coefficients and trendlines are also displayed.

Table 6.3: Spearman correlation coefficients for each model and metric (highest values are marked with bold font).

	<i>Armadillo</i>	<i>Venus</i>	<i>Dinosaur</i>	<i>RockerArm</i>	<i>Mean</i>
<i>Our metric</i>	86%	89%	79%	88%	86%
<i>MSDM</i> [15]	84%	86%	70%	88%	82%
<i>3DWPM2</i> [20]	71%	26%	47%	29%	43%
<i>3DWPM1</i> [20]	64%	68%	59%	85%	69%
<i>GL1</i> [13]	68%	91%	5%	2%	42%
<i>GL2</i> [14]	76%	89%	22%	18%	51%

one frame per animation and this is not sufficient for our metric to be applied on these datasets. Our metric also differs from these metrics in two ways: First we do not require the test and reference meshes to be the same connectivity; for example, the test mesh could be a simplified version of the reference mesh, with a different number of vertices. Moreover, they are not directly comparable to our method since we produce a 3D map of local visible distortions as output, while they give a global error per dynamic mesh. Even though they also generate a 3D map in the interim steps and accumulate it to a single value, we do not have access to those interim steps. Hence, although developing a single error value per dynamic mesh is out of our purpose, to be able to compare our metric, we unified our 3D map into a single score by averaging the error values of each vertex. Then we performed a second user experiment, following a similar design in [27].

In this experiment, we produced three modification levels per dynamic mesh given in Table 6.1, resulting in 12 animations. Using the MeshLab [102] tool, we applied random vertex displacement filter by varying the maximum displacement parameter, which is set as 0.1, 0.2, and 0.3 for levels 1, 2, and 3, respectively.

During the experiments, given the non-modified animation as reference, the subjects were asked to assign a score of 0, 1, 2, or 3 to the modified animation. In this evaluation scheme, 0 means that there is no perceptible difference between the reference and test animations. Evaluations of ten subjects were combined by calculating the mean opinion score (MOS) per modified mesh. Then the correlation between the metric outputs and MOS values was calculated.

MOS vs. metric estimation plot in Figure 6.6 reveals an almost linear relationship. Pearson and Spearman correlation coefficients for each mesh are also listed in Table 6.4. Although the meshes used in the experiments are different; considering that the correlation coefficients in [27] varies between 0.92 and 0.98, our results are comparable to the state-of-the-art. We see that the correlation is very high ($> 90\%$) in this second experiment. This is because assigning an overall score to the given dynamic mesh is an easier task than marking the locations that are perceived different. The main purpose of this study is to produce a 3D map of visible distortions rather than generating an overall quality estimation per mesh.

Table 6.4: Pearson (r) and Spearman (ρ) correlation coefficients for each mesh.

	<i>Pearson$_r$</i>	<i>Spearman$_\rho$</i>
<i>Camel</i>	93%	94%
<i>Elephant</i>	94%	97%
<i>Hand</i>	95%	94%
<i>Horse</i>	98%	95%
<i>Mean</i>	92%	88%

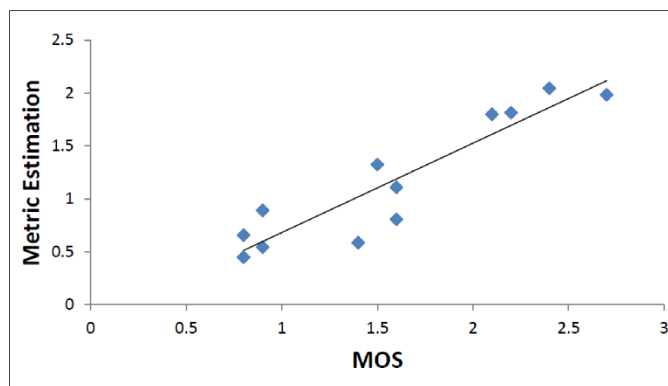


Figure 6.6: Subjective testing results vs. metric estimation.

6.1.3 Performance Evaluation

6.1.3.1 Resolution of the Spatiotemporal Volume

The resolution of the spatiotemporal volume at each dimension affects the success of our method. In order to investigate this effect, we also performed several runs of our algorithm with varying voxel resolutions and calculated correlation coefficients for each run. We changed the *minResolution* parameter in Eq. 3.3, which determines the length of the spatiotemporal volume at each dimension, in proportion to the length of the bounding box of the mesh.

Figure 6.7 plots the correlation coefficients with respect to the *minResolution*

Table 6.5: Effect of the *minResolution* parameter on the correlation strengths of each mesh.

	30	60	90	120	150
<i>Camel</i>	Weak	Modest	High	High	High
<i>Elephant</i>	Weak	Weak	Weak	Modest	Modest
<i>Hand</i>	Weak	Modest	High	High	High
<i>Horse</i>	Modest	High	High	High	High

parameter in Eq. 3.3. The plot includes the mean results of all the meshes. We see that the correlation is very low when *minResolution* is 10. Then it starts to increase rapidly with the increasing resolution to a certain extent. After a while, for about *minResolution* > 50, the increase rate drops. For *minResolution* > 100, mean correlation settles to the band of 0.6–0.7 and increasing the resolution no further improves the accuracy.

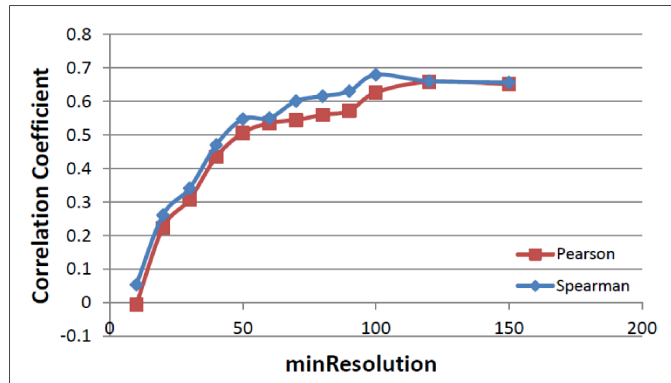


Figure 6.7: Effect of the *minResolution* parameter on the mean correlation.

Table 6.5 lists the strength of the correlation with respect to the *minResolution* parameter, for each mesh. One can observe that the correlation coefficients generally increase with the increasing resolution. When the resolution is too small, too many vertices fall in a single voxel, thus the result is not accurate. As the resolution gets higher, estimation is more accurate but the computational cost also increases. Moreover, incrementing the resolution does not improve the performance radically after a certain value.

According to our experiments, we drew a new heuristic to calculate the

minResolution parameter. It is not desired to have too small resolution that allows many vertices to fall into the same voxel. So we aim to distribute the vertices to different voxels as much as possible. We start with the assumption that vertices are distributed homogeneously. We also know that a mesh is generally represented with the vertices located on the surface and inside of the mesh is empty. Hence, we can assume that vertices are located on the facets of the bounding box. More conservatively, we take the facet of the AABB with the minimum area and obtain a resolution that allows distributing all the N vertices of the mesh to this facet homogeneously. For this purpose, we first calculate the proportions of the facets of the AABB (w, h , and d in Eq. 3.3). Then we can express each dimension as a function of some constant k (such that wk, hk, dk). If we select the minimum two of these dimensions as min_1 and min_2 , we can distribute N vertices to the facet of minimum area with $k = \sqrt{N/(min_1 \times min_2)}$. We can then substitute this k value as the *minResolution* parameter.

This heuristic results in the following approximate *minResolution* values for *Camel*, *Elephant*, *Hand*, and *Horse* meshes respectively: 100, 200, 90, and 60. According to Table 6.5, these values provide high correlations.

In summary, the resolution of the spatiotemporal volume has a significant impact on the estimation accuracy and computational cost of our method. Our heuristic to calculate the resolution of the volume works well. Alternatively, a more intelligent algorithm that considers the distribution and density of the vertices along the mesh bounding box could produce better estimations.

6.1.3.2 Processing Time

We monitored the processing time of our algorithm on a 3.3 GHz PC. As mentioned before, the resolution of the spatiotemporal volume, namely *minResolution* parameter in Eq. 3.3, determines the running time of our method. Figure 6.8 displays the change in the running time of our metric (without preprocessing) per frame, with respect to the *minResolution* parameter. Note that in our method, frames of the animation can be processed in parallel. Hence, processing time of

the animation is determined by the processing time of one frame. The figure implies that processing time changes in proportion to the cube of the *minResolution* parameter, expectedly.

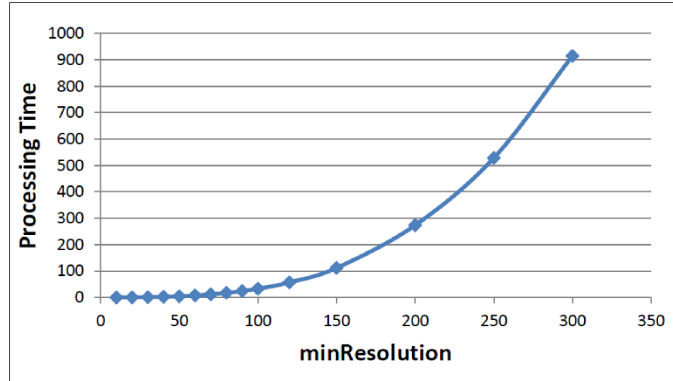


Figure 6.8: Processing time (in seconds) of one frame with respect to the *minResolution* parameter.

Table 6.6 includes the approximate processing times for several meshes, along with their vertex count and *minResolution* parameter calculated according to our heuristic described in Section 6.1.3.1. In the table, the column with title “Time (VB)” refers to the voxel-based approach, while the last column titled with “Time (MB)” is for the mesh-based approach which will be discussed in the next section. As the table indicates, our metric cannot be used in real-time applications in its current form. However, it is possible to improve the performance by processing the spatiotemporal volume on GPU or employing more efficient data structures which process only the non-empty voxels. Another improvement possibility is to use lookup tables for CSF and Difference of Mesa (dom) filters, instead of calculating them on-the-fly.

Table 6.6: Processing times (in seconds) for several meshes.

<i>Mesh</i>	<i>#Vertices</i>	<i>minResolution</i>	<i>Time (VB)</i>	<i>Time (MB)</i>
<i>Horse</i>	8K	60	8	15
<i>Camel</i>	21K	100	33	46
<i>Elephant</i>	42K	200	274	91
<i>Venus</i>	100K	300	915	244

Table 6.7: Pearson (r) and Spearman (ρ) correlation coefficients for each mesh. (Mesh-based approach is shown with bold font.)

	$Pearson_r$	$Spearman_\rho$
<i>Camel</i>	84% - 87%	83% - 81%
<i>Elephant</i>	59% - 58%	65% - 63%
<i>Hand</i>	72% - 69%	71% - 73%
<i>Horse</i>	71% - 71%	70% - 70%
<i>Overall</i>	71% - 71%	72% - 72%

6.1.4 Evaluation of the Mesh-based Approach

Lastly, we evaluate the performance of the mesh-based approach proposed in Section 3.2, in comparison to our voxel-based approach which is analyzed in detail in the previous subsections. Table 6.7 involves Pearson and Spearman correlation coefficients as a measure for the accuracy of the method. In this table, we replicate the correlation results of the voxel-based approach which is already given in Table 6.2, for convenience. In the table, the first columns in each correlation method list the results of the voxel-based approach; while the second columns (with bold font) include the coefficients for the mesh-based approach. These results demonstrate that both the voxel- and mesh-based approaches achieve good and almost the same correlation values.

We also measured the running time of the mesh-based approach, for several meshes. Measurements are performed on a 3.3 GHz PC again. As depicted from Figure 6.9, running time of the algorithm is almost proportional to the number of vertices of the mesh; which is reasonable because the algorithm operates per vertex. Furthermore, processing times for several meshes are also listed in the last column of Table 6.6.

We know that the running time depends on the number of voxels for the voxel-based approach and it depends on the number of vertices in the mesh-based approach. Keeping this observation in mind, we see that the voxel-based approach runs faster for the *horse* and *camel* animations, although the number of vertices is less than the number of voxels in these cases. This is due to the

manifold harmonics calculations in the mesh-based approach. However, the share of these calculations diminishes as the number of voxels gets much higher than the number of vertices.

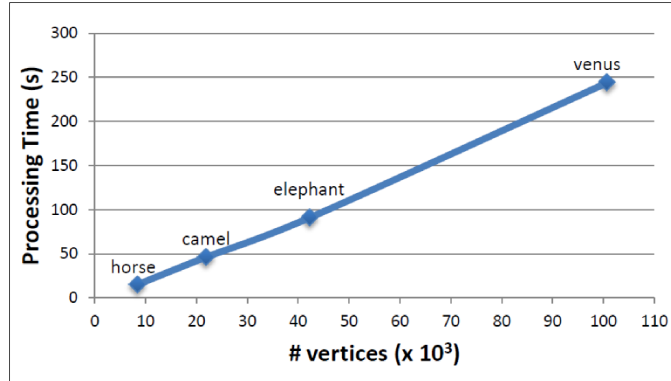


Figure 6.9: Processing time (in seconds) of one frame for several meshes.

Concisely, both the voxel- and mesh-based approaches produce comparable results from the accuracy perspective. On the other hand, we can deduce from the computational time measurements that our mesh-based VQA method is more efficient than the voxel-based method for large meshes (i.e. $\# vertices > 25K$). Nevertheless, it is important to remind that mesh-based approach confines the reference and test meshes to have the same number of vertices. Thus, for large meshes without connectivity changes, mesh-based approach is more preferable because of its efficiency, with the expense of Manifold Harmonics Basis calculations as a preprocessing.

It is also important to note a design issue in the MHB calculations of the mesh-based approach. Calculation of the MHBs for meshes with high number of vertices is a problem due to its space complexity. In order to overcome this problem, cotangent weight and delta matrices in Eq. 3.14 are stored as sparse matrices which enables a significant amount of reduction in the memory space.

We have also adopted the idea in [130], where eigen-decomposition is performed on the simplified version of the mesh and the results are mapped to the original mesh using a kd-tree structure, for mesh saliency calculations. Following this process, for large meshes ($\#vertices > 25K$, in our implementation), the

Channel Decomposition step (Section 3.2.3) is applied on the simplified versions of the meshes and the results are expanded to the original size using a kd-tree representation. For mesh simplification, we employ the *quadric edge collapse decimation* method of MeshLab’s implementation [102], with boundary-preserving option is set.

As a last remark, our primary concern was not the processing time while developing these metrics. Therefore, we have not aimed at a real-time algorithm since there is still application area for offline geometry processing operations. Yet, the performance of the proposed algorithms can further be improved by employing more efficient implementation methods and data structures, which is beyond the scope of our aim.

6.2 Learning the Visual Quality of Static Meshes

In this section, we evaluate the success of our machine learning based VQA metric described in Chapter 4. To evaluate the success of our data-driven VQA metric, we have calculated *prediction accuracy* which measures how well a distance metric predicts the preferences of the human observers. We compare our metric to several state-of-the-art metrics by computing their prediction accuracy values.

To define the prediction accuracy formally, let tuple t collected from the crowdsourcing experiment, be in the form of $\langle A, B, C, q \rangle$; where A is the reference mesh, B and C are the distorted test meshes to be compared, and q is the query response as a binary variable with 1 indicating that B was selected as more similar to A and 0 indicating that C was selected as more similar to A . Given the set of testing tuples T , prediction accuracy (PA_d) is computed as the percentage of correct predictions for variable q , when a specific metric d is used as the decision maker (Eq. 6.2).

$$PA_d = 100 \times \frac{\sum_{t \in T} \delta_{q_t s_t}}{|T|} \quad (6.2)$$

where δ is the Kronecker delta defined as below and s_t is the metric decision for tuple t , determined according to Eq. 6.4.

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \quad (6.3)$$

$$s_{\langle A, B, C, q \rangle} = \begin{cases} 0, & d(A, B) > d(A, C) \\ 1, & d(A, B) < d(A, C) \\ 0, & d(A, B) = d(A, C) \quad \&\& \quad q = 0 \\ 1, & d(A, B) = d(A, C) \quad \&\& \quad q = 1 \end{cases} \quad (6.4)$$

As stated previously, each tuple is evaluated by at least five users in our experiment. According to this design, binary variable q can be determined in two ways. The first option is to process the *raw* tuples, while the second alternative is to construct a single tuple for each unique query triplet by determining the value of q according to the *majority* response for that triplet.

Table 6.8 includes the prediction accuracies for our metric and several other state-of-the-art methods. In the table, *our metric_M* refers to our metric whose parameters are learned according to the majority rating and *our metric_R* denotes the metric learned through the raw tuples. Although the results depict that raw method performs slightly better than the majority method, they both achieve higher performance than the other metrics. Therefore, majority method is more preferable because it suggests an improvement for the computational time of the optimization process, as the number of tuples obtained according to the majority decision is about five times (the minimum number of users evaluated each triplet) less than the number of raw tuples.

6.3 Perceived Depth Quality Enhancement Method

We performed four user experiments to evaluate the proposed technique for common visualization scenarios. These scenes (3D tree, graph, scatter plot, and

Table 6.8: Prediction accuracy of each metric for each mesh (highest values are marked with bold font).

	<i>Armadillo</i>	<i>Venus</i>	<i>Dinosaur</i>	<i>RockerArm</i>	<i>Mean</i>
<i>Our metric_M</i>	81%	89%	87%	93%	88%
<i>Our metric_R</i>	81%	90%	86%	95%	88%
<i>MSDM</i> [15]	83%	85%	76%	85%	82%
<i>3DWPM1</i> [20]	73%	76%	74%	87%	78%
<i>3DWPM2</i> [20]	76%	60%	69%	65%	68%
<i>GL1</i> [13]	74%	88%	55%	54%	68%
<i>GL2</i> [14]	78%	87%	60%	58%	71%

surface visualization) represent most of the common tasks listed in Section 5.2. In the experiments, we measure the success rates as an indicator of the comprehensibility of the scenes. We also investigated the scalability of the proposed method with different scene complexity levels. Table 6.9 summarizes the main tasks for each experiment.

Table 6.9: Tasks in the experiments (*Exp.#1*: Tree Visualization, *Exp.#2*: Graph Visualization, *Exp.#3*: Scatter Plot Visualization, *Exp.#4*: Surface Visualization).

<i>Task</i>	<i>Exp.#1</i>	<i>Exp.#2</i>	<i>Exp.#3</i>	<i>Exp.#4</i>
Judging the relative positions of objects	High	High	High	High
Reaching for objects	Medium	Medium	Medium	Medium
Detecting surface target	Low	Low	Low	High
Tracing data paths in 3D graphs	High	High	Medium	Low
Finding patterns of points in 3D space	Medium	Medium	High	Low
Judging the up direction	Medium	Medium	Medium	Medium
The aesthetic impression of 3D space	Medium	Medium	Medium	High
Navigation	High	High	High	High

Hypotheses: The experiment design is based on the following hypotheses:

- **H1.** The proposed framework increases accuracy, thus comprehensibility, in the performed visualization tasks, when compared to base cases where basic cues are available or depth cues are randomly selected.
- **H2.** The proposed framework allows results as accurate as the “gold standard” case, where all depth cues are available or depth cues are selected by

an experienced designer.

- **H3.** The proposed framework is scalable in terms of scene complexity.

Subjects: Graduate/undergraduate students in engineering and who had no advanced knowledge of depth perception, signed up for the experiment. All participants self-reported normal or corrected-to-normal vision.

Design: We used a repeated measures design with two independent variables. The first independent variable was DEPTH_CUE_SELECTION_TECHNIQUE. In the experiments, we used six different selection methods. In the first method, NO_METHODS, the original scene was rendered using Gouraud shading and perspective projection, with no further depth enhancement methods. The second method, RANDOM, determined if a depth enhancement method would be applied randomly at run-time with no cost constraint. The third case, COST_LIMITED_RANDOM, was random selection with a cost limit; whether a method would be applied was decided randomly as long as it would not increase the rendering time above the given cost limit. We used the same cost limit for this case and the automatic selection case. The fourth method, ALL_METHODS, applied all depth enhancement methods with no total cost constraint. In the fifth method, CUSTOM, an experienced designer in our research group selected the most suitable depth cues for the given scenes. In the last method, AUTO, depth enhancement methods were determined using the proposed framework. Among these selection methods, ALL_METHODS and CUSTOM cases can be considered the “gold standard”.

The second independent variable was SCENE_COMPLEXITY, and was applied to measure the scalability of the proposed method. There were three levels in the first three experiments: LEVEL_1, LEVEL_2, and LEVEL_3, with the scene becoming more complex from LEVEL_1 to LEVEL_3. We used different metrics for scene complexity in each experiment, which will be explained in the related sections.

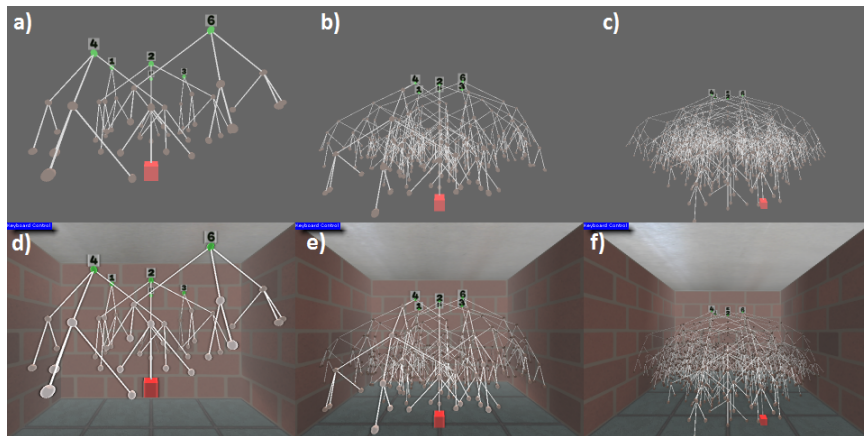
The presentation order of the technique was counterbalanced across participants. The procedure was explained to the subjects at the beginning. After a short training session, subjects began the experiment by pressing a button when they felt ready. Written instructions were also available in the experiments.

6.3.1 Experiment 1 - 3D Tree Visualization

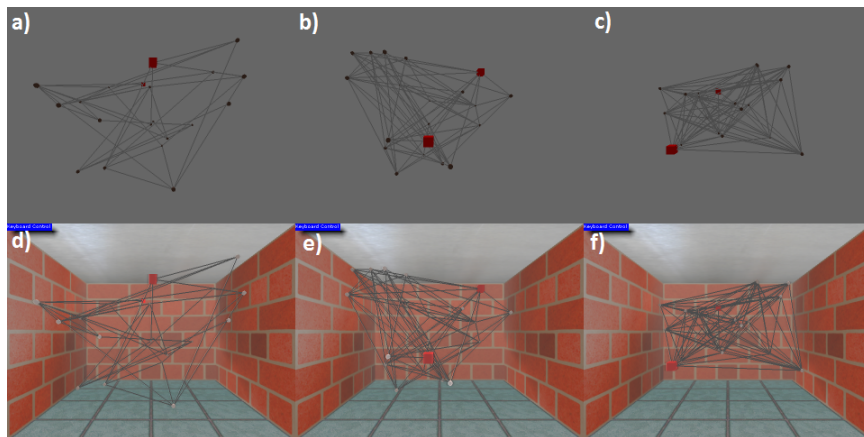
Procedure: Fifteen subjects were asked to find the root of a given leaf node in a randomly generated 3D tree (Figure 6.10a). The test leaf was also determined randomly at each run and displayed with a different color and shape. A forced-choice design was used, and the subjects selected an answer from 1 to 6 on the given form. There were six trees in each run. As the scene complexity measure, we used tree height, set to 2, 4, and 5 in LEVEL_1, LEVEL_2, and LEVEL_3, respectively. In total, the design of the experiment resulted in: $15 \text{ participants} \times \text{DEPTH_CUE_SELECTION_TECHNIQUE} \times \text{SCENE_COMPLEXITY} = 270 \text{ trials}$.

Results: For this experiment, the automatically selected methods were *proximity luminance*, *multi-view*, *boundary enhancement*, *keyboard-controlled motion*, and *room* (Figure 6.10a). In this selection, the noteworthy methods are *multi-view*, *boundary enhancement*, and *keyboard-controlled motion*. These selections are consistent with the inference in [90], which concludes that binocular disparity coupled with motion parallax gives the best results for tracing data paths in 3D graphs. *Boundary enhancement* is helpful in the sense that occlusions between the links become apparent and it is easier to follow the paths. *Room* was also selected because there is an available cost budget and it enhances the linear perspective. For the CUSTOM case, however, our designer selected the *multi-view* and *keyboard-controlled motion* methods.

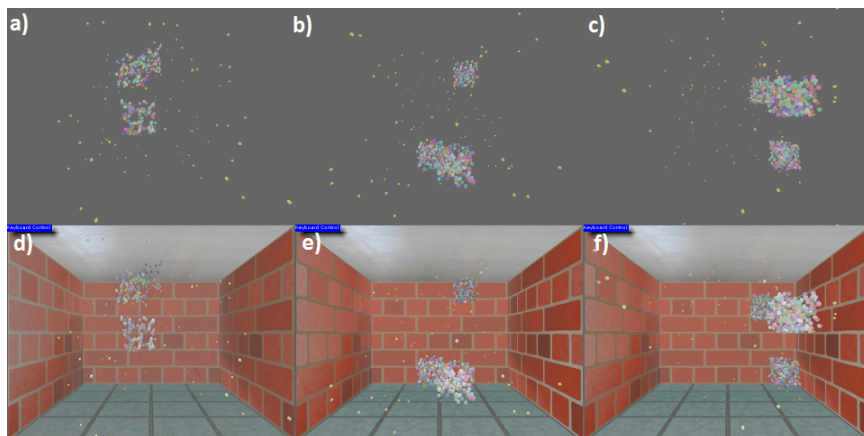
The scene complexity level did not have a vivid effect on the selected rendering methods except in two ways: First, in the LEVEL_1 scene, the proposed framework selected the *shadow* method in addition to the above methods. This is



(a) 3D Tree Visualization.



(b) 3D Graph Visualization.



(c) 3D Scatter Plot Visualization.

Figure 6.10: Top rows: The scenes with basic cues. Bottom rows: The scenes with the automatically selected methods. (Multi-view, face tracking, keyboard-control methods cannot be shown here.) Left to right: Scene Complexity Level 1, Level 2, Level 3. (Image from [59], © 2013 ACM, reprinted with permission.)

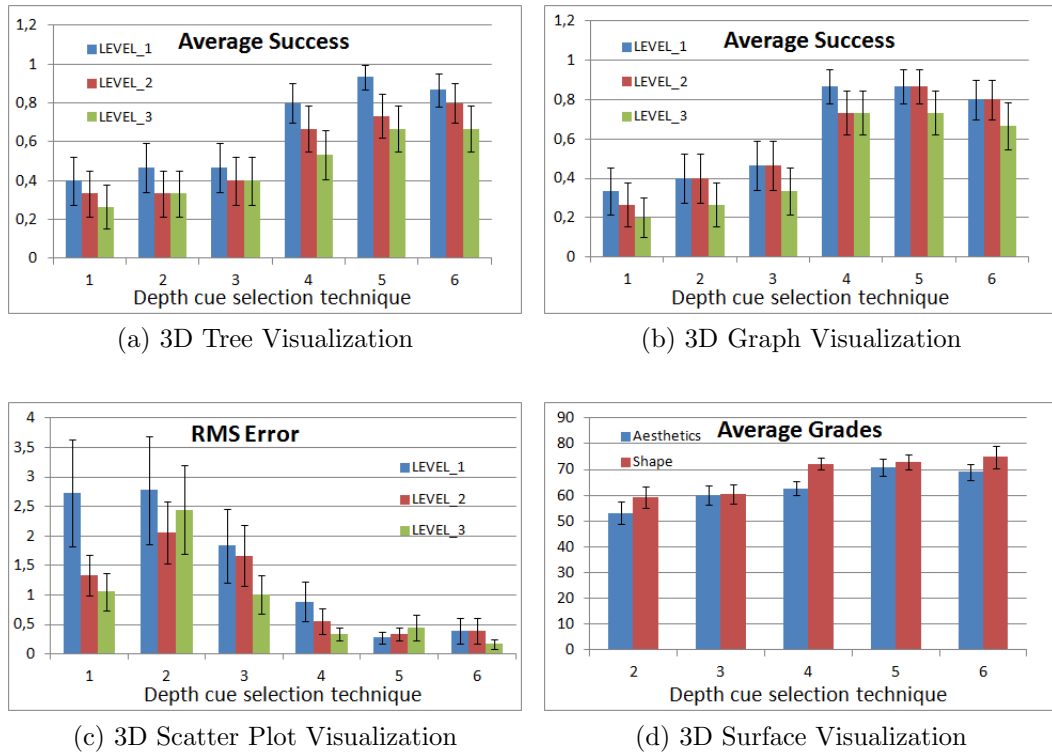


Figure 6.11: Experiment results. (Depth cue selection techniques: 1 - No methods, 2 - Random, 3 - Cost-limited random, 4 - All methods, 5 - Custom, 6 - Auto selection. Error bars show the 95 % confidence intervals.) (Image from [59], © 2013 ACM, reprinted with permission.)

comprehensible as the number of nodes in this level is low and thus there is more rendering budget. Second, in the most complex scene, *boundary enhancement* method was not selected because this method increases the rendering cost.

The success rate was calculated as the percentage of viewers who found the correct root. Figure 6.11a shows the experimental results. The figure shows that ALL_METHODS, CUSTOM, and AUTO cases had the highest success rates and that the success rates of these methods decrease as the scene becomes more complex, as expected.

Using the repeated measures ANOVA on the experimental results, we found a significant effect ($F(1, 14) = 10.66, p = 0.006 < 0.05$) for the

DEPTH_CUE_SELECTION_TECHNIQUE on success rate. A pairwise comparison revealed significant differences ($p < 0.001$) between the AUTO method and three of the selection methods: NO_METHODS (mean success rate: 0.33), RANDOM (mean success rate: 0.37), and COST_LIMITED_RANDOM (mean success rate: 0.42). Pairwise comparisons showed no significant difference between our method (AUTO) and remaining methods (ALL_METHODS, CUSTOM). Moreover, no significant interaction was observed between the independent variables.

These results show that the proposed framework enhances the success rates and produces a more comprehensible scene in the 3D tree visualization task. Furthermore, the results are comparable to gold standard cases (custom selection and applying all methods with no cost restriction). With the proposed method, the success rates are considerable for each scene complexity level (Level 1: 86 %, Level 2: 80 %, Level 3: 66 %). These results suggest that our framework is scalable in terms of scene complexity, defined by the height of the tree, although other scene complexity measures need to be tested, such as the number of trees, occlusion level, tree layout, etc.

6.3.2 Experiment 2 - 3D Graph Visualization

Procedure: Fifteen subjects were shown a 3D graph in which two randomly selected nodes were colored differently (Figure 6.10b). The task was to determine whether the selected nodes were linked by a path of length 2, as suggested in [90]. The subjects selected “YES” or “NO” from the result form. In the experiments, we kept the total number of nodes fixed as 20. The scene complexity was determined according to the graph density, which is the ratio of the number of edges and the number of possible edges ($D = 2 \times |E| / \{|V| \times (|V| - 1)\}$). The density levels were selected as 0.25, 0.50, and 0.75 for LEVEL_1, LEVEL_2, and LEVEL_3, respectively. In total, the design of the experiment resulted in: $15 \text{ participants} \times \text{DEPTH_CUE_SELECTION_TECHNIQUE} \times \text{SCENE_COMPLEXITY} = 270 \text{ trials}$.

Results: The automatically selected methods were: *proximity luminance*, *multi-view*, *boundary enhancement*, *keyboard-controlled motion*, and *room* (Figure 6.10b). The designer selected *multi-view* and *keyboard-controlled motion* in the CUSTOM case for each level of scene complexity.

According to the repeated measures ANOVA, we found a significant effect for the DEPTH_CUE_SELECTION_TECHNIQUE on success rates ($F(1, 14) = 13.27, p = 0.003 < 0.05$). Pairwise comparisons revealed significant differences ($p < 0.001$) between the AUTO method (mean success rate: 0.75) and three of the selection methods: NO_METHODS (mean success rate: 0.26), RANDOM (mean success rate: 0.35), and COST_LIMITED_RANDOM (mean success rate: 0.42). Furthermore, pairwise comparisons between the AUTO technique and the remaining methods (CUSTOM, ALL_METHODS) revealed no significant difference, suggesting similar performance of our method to the gold standard case. The interaction between DEPTH_CUE_SELECTION_TECHNIQUE and SCENE_COMPLEXITY was not found to be significant.

These results further show that the proposed framework also facilitates the 3D graph visualization task. The results are comparable to gold standard cases (custom selection and applying all methods) in each level of scene complexity. This is an implication for the scalability of our method.

6.3.3 Experiment 3 - 3D Scatter Plot Visualization

Procedure: Eighteen subjects were asked to find the number of equally sized natural clusters among random noise points in a 3D scatter plot (Figure 6.10c). A forced-choice design was used. This time, the scene complexity variable was the number of nodes in a cluster (LEVEL_1: 100, LEVEL_2: 200, LEVEL_3: 300). There were three to seven clusters in the given scenes. In total, the design of the experiment resulted in: $18 \text{ participants} \times \text{DEPTH_CUE_SELECTION_TECHNIQUE} \times \text{SCENE_COMPLEXITY} = 324 \text{ trials}$.

Results: In this test, *proximity luminance*, *keyboard-controlled motion*, *face tracking*, and *room* methods were selected automatically in all levels. In the first level of scene complexity, *shadow* was also selected. Custom selection included *keyboard-controlled motion*, *multi-view*, and *room* methods for each level.

We calculated the average error in user responses for the number of clusters using the RMS error:

$$RMS(E) = \sqrt{(\sum_{i=1}^{|E|} (E_i - C_i)^2) / |E|} \quad (6.5)$$

where E is the set of user responses and C is the set of correct answers. Root mean square errors are displayed in Figure 6.11c; the smallest errors were obtained using the AUTO and CUSTOM methods. The errors were also small in ALL_METHODS case. Another observation is that the error rate decreases as the number of points in a cluster increases. One possible reason for this result is that as the number of points increases, the patterns in a cluster become more obvious and the clusters approach the appearance of a solid shape.

A repeated measures ANOVA test showed the significant ($F(1,17) = 10.28, p = 0.005 < 0.05$) effect of DEPTH_CUE_SELECTION_TECHNIQUE on the results. Pairwise differences AUTO - NO_METHODS, AUTO - RANDOM, AUTO - COST_LIMITED_RANDOM are also significant ($p < 0.001$). On the other hand, the differences between AUTO method and other methods (ALL_METHODS and CUSTOM) are not significant according to the statistical analysis. According to the ANOVA analysis, there is no significant interaction between the independent variables.

6.3.4 Experiment 4 - 3D Surface Visualization

Procedure: Seventeen viewers evaluated the scenes subjectively. They were shown the scene with basic cues and told that the grade of this scene was 50. Then, they were asked to grade the other scenes between 0 and 100 in comparison to the first scene. There were two grading criteria: shape and aesthetics.

The subjects were informed about the meanings of these criteria at the beginning: “While reporting the shape grades, evaluate the clarity of surface details like curvatures, convexities, etc. For the aesthetic criterion, assess the general aesthetic impression and overall quality of the images.” We used a terrain model – with 5000 faces and 2600 vertices (Figure 6.12) – to represent surface plots in visualization. In this experiment, the scene complexity was not tested and only one level was used. In total, the design of the experiment resulted in: $17 \text{ participants} \times (\text{DEPTH_CUE_SELECTION_TECHNIQUE}-1) = 85 \text{ trials}$.

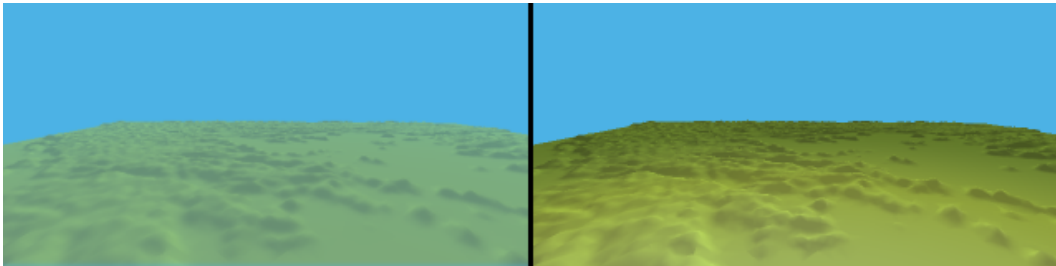


Figure 6.12: Test scenes for 3D surface experiment. (Left: A portion of the scene with basic cues, Right: A portion of the scene with the automatically selected methods. Note that multi-view and keyboard control methods cannot be displayed here.) (Image from [59], © 2013 ACM, reprinted with permission.)

Results: *Multi-view, Gooch shading, proximity luminance, bump mapping, shadow, boundary enhancement, and keyboard-controlled motion* methods were selected automatically. The designer selected the *fog* method in addition to the above methods, and included *face tracking* instead of *multi-view*.

Figure 6.11d shows the average grades of each test case for the “shape” and “aesthetics” criteria. As the plot indicates, the proposed method results in comparable grades for both criteria to the custom selection. It can be concluded that these selections are also appropriate since *shape-from-shading* and *structure-from-motion* cues are available in the selected methods.

The one-way repeated measures ANOVA on the experimental results found a significant effect for the DEPTH_CUE_SELECTION_TECHNIQUE on the shape grades ($F(4, 16) = 19.08, p = 0.00023 < 0.05$). Further pairwise comparisons

showed that pairwise differences between RANDOM and AUTO, and between COST_LIMITED_RANDOM and AUTO are also statistically significant ($p < 0.001$). These results suggest that the proposed method gives better subjective evaluation of the surface shapes. Furthermore, pairwise comparisons between the AUTO technique and the remaining methods (CUSTOM, ALL_METHODS) revealed no significant difference, suggesting similar performance of our method to the gold standard case.

The ANOVA test for the aesthetics grades also showed that the effect of the depth cue selection method is significant ($F(4, 16) = 26.38, p = 0.00061 < 0.05$). Pairwise comparisons showed that differences RANDOM-AUTO and COST_LIMITED_RANDOM-AUTO are also statistically significant ($p < 0.001$). However, pairwise comparisons also show statistically significant difference ($p = 0.001 < 0.05$) between the proposed method and ALL_METHODS. This is understandable because applying all methods generates visual clutter, which affects the perception of aesthetics. In addition, most of the subjects reported that lower frame rates distracted them and that they considered this situation while evaluating the aesthetics criterion. Another possible reason for this result is that some of the methods may interact with each other when applied together; however this requires further investigation and analysis of cue conflicts. Conversely, there is no statistically significant difference between the proposed method and the CUSTOM case.

Chapter 7

Conclusion

The ultimate goal while generating a 3D scene using computer graphics techniques is to provide high quality and comprehensible contents with tolerable cost. Since the users evaluate the quality of a scene based on their perception, it is a good idea to employ the principles of visual perception in order to generate scenes that are perceived as of high quality, while saving rendering cost. To that end, we first develop a perceptual quality metric for measuring the visibility of distortions on static and dynamic 3D meshes. We also suggest a system for automatically enhancing the perceived depth quality in a 3D scene, to improve the comprehensibility of the visualization.

One of our aims in this thesis is to provide a general-purpose visual quality metric for dynamic triangle meshes since it is a costly process to accomplish subjective user evaluations. For this purpose, we propose a full-reference perceptual quality estimation method based on the well-known VDP approach by Daly [29]. Our approach accounts for both spatial and temporal sensitivity of the HVS. As the output of our algorithm, we obtain a 3D probability map of visible distortions. According to our formal experimental study, our perceptually-aware quality metric produces promising results.

The most significant distinction of our method is that it handles animated 3D

meshes; since most of the studies in the literature omit the effect of temporal variations. Our method is independent of connectivity, shading, and material properties; which offers a general-purpose quality estimation method that is not application-specific. It is possible to measure the quality of 3D meshes that are distorted by a modification method which changes the connectivity or number of vertices of the mesh. The algorithm can also account for static meshes. The proposed method is even applicable to the scenes containing multiple dynamic or static meshes. More importantly, the representation of the input mesh is not limited to triangle meshes and it is possible to apply the method on point-based surface representation. Lastly, we provide an open dataset including subjective user evaluation results for 3D dynamic meshes.

The main drawback of our method is the computational complexity due to 4D nature of the spatiotemporal volume. However, we overcome this problem to some extent by using a time window approach which processes a limited number of consecutive frames. Furthermore, a significant amount of speed-up may be obtained by processing the spatiotemporal volume in GPU.

As a future work, we aim to perform a more comprehensive user study, investigating the effects of several parameters. Another possible research direction is to integrate visual attention and saliency mechanism to the system.

Another contribution of this thesis is a method for estimating the perceived quality of a static mesh using a machine learning pipeline, in which crowdsourced data is used while learning the parameters of a distance metric that best fits the human perception. To the best of our knowledge, this is the first attempt for a VQA metric that utilizes crowdsourcing tools. As the initial attempt for such a VQA method, our metric handles the global visual quality of static meshes only, in a full-reference scenario. Nevertheless, we have plans to extend this idea for evaluating the local visibility of distortions and also considering animated meshes. A similar approach can be applied even for no-reference quality assessment.

Lastly, since depth is an important component of visualization, we also propose a framework to determine the suitable rendering methods that make the

spatial relationship of the objects apparent, thus enhance the comprehensibility in a given 3D scene. In this framework, we consider several factors, including the target task, scene layout, and the costs of the rendering methods. This framework develops a hybrid model of the existing cue combination models: *cue averaging*, *cue specialization*, *range extension*, and *cue conflict*. In the framework, important depth cues for a given scene are determined based on a fuzzy logic system. Next, the problem of which rendering methods will be used for providing these important cues in computer-generated scenes is solved using a knapsack cost-profit analysis.

The proposed solution was verified using formal experimental studies, and statistical analysis shows the effective performance of our system for common information visualization tasks, including graph, tree, surface, and scatter plot visualization. The results reveal a somewhat better tree and graph visualization than surface plots, which is likely due to the strength of the depth cues and rendering methods used for graph visualization than for surface or scatter plot visualization.

Although the system performs well for the tested environments, it has several limitations. First, the system should be validated for other scene complexity measures. The requirement for the framework to access the complete structure of the scene is another limitation. The rule base requires frequent updates to incorporate new findings in the perception research.

To sum up, although we provide several methods for applying the principles of visual perception in computer graphics, there is still room for research in perceptually-based computer graphics techniques such as perceptual geometry processing, no-reference quality metrics, visual attention models, etc. We believe that our study gives some insight for the usage of perception in computer graphics and visualization applications.

Bibliography

- [1] M. Reddy, “Perceptually optimized 3D graphics,” *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 68–75, 2001.
- [2] K. Cater, A. Chalmers, and G. Ward, “Detail to attention: exploiting visual tasks for selective rendering,” in *ACM International Conference Proceeding Series*, vol. 44, pp. 270–280, 2003.
- [3] C. O’Sullivan, S. Howlett, Y. Morvan, R. McDonnell, and K. O’Conor, “Perceptually adaptive graphics,” *Eurographics State of the Art Reports*, vol. 4, pp. 1–24, 2004.
- [4] J. Hasic, A. Chalmers, and E. Sikudova, “Perceptually guided high-fidelity rendering exploiting movement bias in visual attention,” *ACM Transactions on Applied Perception (TAP)*, vol. 8, no. 1, p. 6, 2010.
- [5] C. Healey and J. Enns, “Attention and visual memory in visualization and computer graphics,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 7, pp. 1170–1188, 2012.
- [6] A. Bulbul, T. K. Çapın, G. Lavoué, and M. Preda, “Assessing Visual Quality of 3-D Polygonal Models,” *IEEE Signal Processing Magazine*, vol. 28, no. 6, pp. 80–90, 2011.
- [7] M. Corsini, M.-C. Larabi, G. Lavoué, O. Petřík, L. Váša, and K. Wang, “Perceptual metrics for static and dynamic triangle meshes,” in *Computer Graphics Forum*, vol. 32, pp. 101–125, Wiley Online Library, 2013.

- [8] G. Lavoué and R. Mantiuk, “Quality assessment in computer graphics,” in *Visual Signal Quality Assessment*, pp. 243–286, Springer, 2015.
- [9] P. Cignoni, C. Rocchini, and R. Scopigno, “Metro: Measuring Error on Simplified Surfaces.,” *Comput. Graph. Forum*, vol. 17, no. 2, pp. 167–174, 1998.
- [10] D. P. Luebke, *Level of detail for 3D graphics*. Morgan Kaufmann, 2003.
- [11] Z. Wang, H. R. Sheikh, and A. C. Bovik, “No-reference perceptual quality assessment of jpeg compressed images,” in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 1, pp. I–477, IEEE, 2002.
- [12] W. Lin and C.-C. Jay Kuo, “Perceptual visual quality metrics: A survey,” *Journal of Visual Communication and Image Representation*, vol. 22, no. 4, pp. 297–312, 2011.
- [13] Z. Karni and C. Gotsman, “Spectral compression of mesh geometry,” in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 279–286, ACM Press/Addison-Wesley Publishing Co., 2000.
- [14] O. Sorkine, D. Cohen-Or, and S. Toledo, “High-pass quantization for mesh encoding,” in *Symposium on Geometry Processing*, pp. 42–51, Citeseer, 2003.
- [15] G. Lavoué, E. D. Gelasca, F. Dupont, A. Baskurt, and T. Ebrahimi, “Perceptually driven 3D distance metrics with application to watermarking,” in *Optics & Photonics*, pp. 63120L–63120L, International Society for Optics and Photonics, 2006.
- [16] G. Lavoué, “A multiscale metric for 3D mesh visual quality assessment,” in *Computer Graphics Forum*, vol. 30, pp. 1427–1437, Wiley Online Library, 2011.
- [17] S.-J. Kim, S.-K. Kim, and C.-H. Kim, “Discrete differential error metric for surface simplification,” in *Computer Graphics and Applications*, pp. 276–283, IEEE, 2002.

- [18] C. Lee, A. Varshney, and D. Jacobs, “Mesh saliency,” in *ACM SIGGRAPH*, pp. 659–666, ACM, 2005.
- [19] E. D. Gelasca, T. Ebrahimi, M. Corsini, and M. Barni, “Objective evaluation of the perceptual quality of 3D watermarking,” in *IEEE International Conference on Image Processing*, vol. 1, pp. I–241, IEEE, 2005.
- [20] M. Corsini, E. Gelasca, T. Ebrahimi, and M. Barni, “Watermarked 3-D mesh quality assessment,” *IEEE Transactions on Multimedia*, vol. 9, no. 2, pp. 247–256, 2007.
- [21] Y. Pan, L. I. Cheng, and A. Basu, “Quality metric for approximating subjective evaluation of 3-D objects.,” *IEEE Transactions on Multimedia*, vol. 7, no. 2, pp. 269–279, 2005.
- [22] K. Wang, F. Torkhani, and A. Montanvert, “A fast roughness-based approach to the assessment of 3D mesh visual quality,” *Computers & Graphics*, vol. 36, no. 7, pp. 808–818, 2012.
- [23] F. Torkhani, K. Wang, and J.-M. Chassery, “A curvature-tensor-based perceptual quality metric for 3D triangular meshes,” *Machine Graphics and Vision*, vol. 23, no. 1-2, pp. 59–82, 2014.
- [24] L. Dong, Y. Fang, W. Lin, and H. S. Seah, “Perceptual quality assessment for 3D triangle mesh based on curvature,” *IEEE Transactions on Multimedia*, vol. 17, no. 12, pp. 2174–2184, 2015.
- [25] G. Nader, K. Wang, F. Hetroy-Wheeler, and F. Dupont, “Just noticeable distortion profile for flat-shaded 3D mesh surfaces.,” *IEEE transactions on visualization and computer graphics*, 2015.
- [26] J. Guo, V. Vidal, A. Baskurt, and G. Lavoué, “Evaluating the local visibility of geometric artifacts,” in *Proceedings of the ACM SIGGRAPH Symposium on Applied Perception*, pp. 91–98, ACM, 2015.
- [27] L. Vasa and V. Skala, “A perception correlated comparison method for dynamic meshes,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 2, pp. 220–230, 2011.

- [28] F. Torkhani, K. Wang, and J.-M. Chassery, “Perceptual quality assessment of 3D dynamic meshes: Subjective and objective studies,” *Signal Processing: Image Communication*, vol. 31, pp. 185–204, 2015.
- [29] S. J. Daly, “Visible differences predictor: an algorithm for the assessment of image fidelity,” in *Proceedings of SPIE/IS&T Symposium on Electronic Imaging: Science and Technology*, pp. 2–15, International Society for Optics and Photonics, 1992.
- [30] P. Longhurst and A. Chalmers, “User Validation of Image Quality Assessment Algorithms,” in *Proceedings of the Theory and Practice of Computer Graphics*, (Washington, DC, USA), pp. 196–202, IEEE Computer Society, 2004.
- [31] J. Lubin, *A Visual Discrimination Model for Imaging System Design and Evaluation*. World Scientific, 1995.
- [32] B. Li, G. W. Meyer, and R. V. Klassen, “Comparison of two image quality models,” in *Photonics West Electronic Imaging*, pp. 98–109, International Society for Optics and Photonics, 1998.
- [33] M. R. Bolin and G. W. Meyer, “A perceptually based adaptive sampling algorithm,” in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’98, (New York, NY, USA), pp. 299–309, ACM, 1998.
- [34] B. Watson, A. Friedman, and A. McGaffey, “Measuring and predicting visual fidelity,” in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’01, (New York, NY, USA), pp. 213–220, ACM, 2001.
- [35] M. Ramasubramanian, S. N. Pattanaik, and D. P. Greenberg, “A perceptually based physical error metric for realistic image synthesis,” in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’99, (New York, NY, USA), pp. 73–82, ACM Press/Addison-Wesley Publishing Co., 1999.

- [36] G. Ramanarayanan, J. Ferwerda, B. Walter, and K. Bala, “Visual equivalence: towards a new standard for image fidelity,” in *Proceedings of ACM SIGGRAPH*, SIGGRAPH '07, (New York, NY, USA), ACM, 2007.
- [37] J. A. Ferwerda, P. Shirley, S. N. Pattanaik, and D. P. Greenberg, “A model of visual masking for computer graphics,” in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, (New York, NY, USA), pp. 143–152, ACM Press/Addison-Wesley Publishing Co., 1997.
- [38] S. Albin, G. Rougeron, B. Peroche, and A. Tremeau, “Quality image metrics for synthetic images based on perceptual color differences,” *Image Processing, IEEE Transactions on*, vol. 11, pp. 961–971, Sept. 2002.
- [39] L. Dong, Y. Fang, W. Lin, C. Deng, C. Zhu, and H. S. Seah, “Exploiting entropy masking in perceptual graphic rendering,” *Signal Processing: Image Communication*, vol. 33, pp. 1–13, 2015.
- [40] B. E. Rogowitz and H. E. Rushmeier, “Are image quality metrics adequate to evaluate the quality of geometric objects?,” in *Photonics West 2001-Electronic Imaging*, pp. 340–348, International Society for Optics and Photonics, 2001.
- [41] I. Cleju and D. Saupe, “Evaluation of supra-threshold perceptual metrics for 3d models,” in *Proceedings of the 3rd Symposium on Applied Perception in Graphics and Visualization*, APGV '06, (New York, NY, USA), pp. 41–44, ACM, 2006.
- [42] G. Lavoué, M. C. Larabi, and L. Vasa, “On the efficiency of image metrics for evaluating the visual quality of 3D models,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, pp. 1987–1999, Aug 2016.
- [43] T. O. Aydin, M. Čadík, K. Myszkowski, and H.-P. Seidel, “Video quality assessment for computer graphics applications,” in *ACM Transactions on Graphics (TOG)*, vol. 29, p. 161, ACM, 2010.
- [44] P. Burt and E. Adelson, “The laplacian pyramid as a compact image code,” *IEEE Transactions on Communications*, vol. 31, no. 4, pp. 532–540, 1983.

- [45] N. Ahmed, T. Natarajan, and K. R. Rao, “Discrete cosine transform,” *IEEE Transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.
- [46] A. B. Watson, “The cortex transform: rapid computation of simulated neural images,” *Computer Vision, Graphics, and Image Processing*, vol. 39, no. 3, pp. 311–327, 1987.
- [47] K. Myszkowski, P. Rokita, and T. Tawara, “Perception-based fast rendering and antialiasing of walkthrough sequences,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 6, no. 4, pp. 360–379, 2000.
- [48] R. Snowden, R. J. Snowden, P. Thompson, and T. Troscianko, *Basic Vision: An Introduction to Visual Perception*, ch. Spatial Vision. Oxford University Press, 2012.
- [49] F. W. Campbell and J. Robson, “Application of fourier analysis to the visibility of gratings,” *The Journal of Physiology*, vol. 197, no. 3, p. 551, 1968.
- [50] C. Blakemore and F. W. Campbell, “On the existence of neurones in the human visual system selectively sensitive to the orientation and size of retinal images.,” *The Journal of Physiology*, vol. 203, pp. 237–260, July 1969.
- [51] P. G. Barten, *Contrast sensitivity of the human eye and its effects on image quality*, vol. 21. SPIE Optical Engineering Press Washington, 1999.
- [52] J. Mannos and D. Sakrison, “The effects of a visual fidelity criterion of the encoding of images,” *IEEE Transactions on Information Theory*, vol. 20, no. 4, pp. 525–536, 1974.
- [53] D. Kelly, “Motion and vision. ii. stabilized spatio-temporal threshold surface,” *Journal of the Optical Society of America*, vol. 69, no. 10, pp. 1340–1349, 1979.
- [54] S. Daly, “Engineering observations from spatiovelocity and spatiotemporal visual models,” *Human Vision and Electronic Imaging III*, vol. 3299, pp. 180–191, 1998.

- [55] I. Howard and B. Rogers, *Seeing in Depth*. Oxford University Press, 2008.
- [56] P. Shirley, *Fundamentals of Computer Graphics*. A. K. Peters, Ltd., 2002.
- [57] C. Ware, *Information Visualization: Perception for Design*, ch. 8. Elsevier, 2004.
- [58] Z. Cipiloglu, A. Bulbul, and T. Capin, “A framework for enhancing depth perception in computer graphics,” in *Proceedings of the 7th Symposium on Applied Perception in Graphics and Visualization*, APGV ’10, (New York, NY, USA), pp. 141–148, ACM, 2010.
- [59] Z. Cipiloglu Yildiz, A. Bulbul, and T. Capin, “A framework for applying the principles of depth perception to information visualization,” *ACM Transactions on Applied Perception*, vol. 10, pp. 19:1–19:22, Oct. 2013.
- [60] L. T. Maloney and M. S. Landy, “A statistical framework for robust fusion of depth information,” in *Proceedings of the Advances in Intelligent Robotics Systems Conference*, pp. 1154–1163, International Society for Optics and Photonics, 1989.
- [61] I. Oruc, T. Maloney, and M. Landy, “Weighted linear cue combination with possibly correlated error,” *Vision Research*, vol. 43, pp. 2451–2468, 2003.
- [62] M. F. Bradshaw, A. D. Parton, and A. Glennerster, “The task-dependent use of binocular disparity and motion parallax information,” *Vision Research*, vol. 40, no. 27, pp. 3725 – 3734, 2000.
- [63] P. R. Schrater and D. Kersten, “How optimal depth cue integration depends on the task,” *International Journal of Computer Vision*, vol. 40, pp. 73–91, 2000.
- [64] J. E. Cutting and P. M. Vishton, *Perception of Space and Motion. Handbook of Perception and Cognition*, pp. 69–117. Academic Press, second ed., 1995.
- [65] J. Trommershäuser, *Sensory Cue Integration*, ch. 1 - Ideal Observer Models of Cue Integration. Oxford University Press, USA, 2011.

- [66] D. C. Knill, “Reaching for visual cues to depth: The brain combines depth cues differently for motor control and perception,” *Journal of Vision*, vol. 5, no. 2, pp. 103–115, 2005.
- [67] G. S. Hubona, P. N. Wheeler, G. W. Shirah, and M. Brandt, “The relative contributions of stereo, lighting, and background scenes in promoting 3d depth visualization,” *ACM Transactions on Computer-Human Interaction*, vol. 6, no. 3, pp. 214–242, 1999.
- [68] L. R. Wanger, J. A. Ferwerda, and D. A. Greenberg, “Perceiving spatial relationships in computer-generated images,” *IEEE Computer Graphics and Applications*, vol. 12, pp. 44–58, 1992.
- [69] M. Zannoli, R. A. Albert, A. Bulbul, R. Narain, J. F. O’Brien, and M. Banks, “Correct blur and accommodation information is a reliable cue to depth ordering,” *Journal of Vision*, vol. 14, no. 10, pp. 138–138, 2014.
- [70] P. Haeberli and K. Akeley, “The accumulation buffer: Hardware support for high-quality rendering,” *ACM SIGGRAPH*, vol. 24, no. 4, pp. 309–318, 1990.
- [71] M. Bunnell, *Dynamic Ambient Occlusion and Indirect Lighting*. Addison Wesley, 2004.
- [72] A. Gooch, B. Gooch, P. Shirley, and E. Cohen, “A non-photorealistic lighting model for automatic technical illustration,” in *SIGGRAPH ’98: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, (New York, NY, USA), pp. 447–452, ACM, 1998.
- [73] P. Rheingans and D. Ebert, “Volume illustration: Nonphotorealistic rendering of volume models,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 7, no. 3, pp. 253–264, 2001.
- [74] L. Markosian, M. A. Kowalski, D. Goldstein, S. J. Trychin, J. F. Hughes, and L. D. Bourdev, “Real-time nonphotorealistic rendering,” in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 415–420, ACM Press/Addison-Wesley Publishing Co., 1997.

- [75] M. Nienhaus and J. Döllner, “Edge-enhancement-an algorithm for real-time non-photorealistic rendering,” in *Proceedings of the International Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision*, 2003.
- [76] T. Luft, C. Colditz, and O. Deussen, “Image enhancement by unsharp masking the depth buffer,” *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 1206–1213, 2006.
- [77] N. A. Dodgson, “Autostereoscopic 3D displays,” *Computer*, vol. 38, pp. 31–36, 2005.
- [78] A. Bulbul, Z. Cipiloglu, and T. Capin, “A perceptual approach for stereoscopic rendering optimization,” *Computers & Graphics*, vol. 34, no. 2, pp. 145–157, 2010.
- [79] A. Bulbul, Z. Cipiloglu, and T. Capin, “A color-based face tracking algorithm for enhancing interaction with mobile devices,” *The Visual Computer*, vol. 26, no. 5, pp. 311–323, 2010.
- [80] T. Ropinski, F. Steinicke, and K. Hinrichs, “Visually supporting depth perception in angiography imaging,” in *Smart Graphics* (A. Butz, B. Fisher, A. Krger, and P. Olivier, eds.), vol. 4073 of *Lecture Notes in Computer Science*, pp. 93–104, Springer Berlin / Heidelberg, 2006.
- [81] M. Tarini, P. Cignoni, and C. Montani, “Ambient occlusion and edge cueing for enhancing real time molecular visualization,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, pp. 1237–1244, Sept.-Oct. 2006.
- [82] B. Preim, A. Baer, D. Cunningham, T. Isenberg, and T. Ropinski, “A Survey of Perceptually Motivated 3D Visualization of Medical Image Data,” *Computer Graphics Forum*, vol. 35, pp. 501–525, June 2016.
- [83] D. Keim, “Information visualization and visual data mining,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 1–8, 2002.

- [84] S. G. Gershon, N. E., “Information visualization applications in the real world,” *IEEE Computer Graphics and Applications*, vol. 17, no. 4, pp. 66–70, 1997.
- [85] G. G. Robertson, J. D. Mackinlay, and S. K. Card, “Cone trees: animated 3D visualizations of hierarchical information,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Reaching Through Technology*, CHI '91, (New York, NY, USA), pp. 189–194, ACM, 1991.
- [86] G. Robertson, M. Czerwinski, K. Larson, D. C. Robbins, D. Thiel, and M. van Dantzich, “Data mountain: using spatial memory for document management,” in *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology*, UIST '98, (New York, NY, USA), pp. 153–162, ACM, 1998.
- [87] G. Robertson, M. van Dantzich, D. Robbins, M. Czerwinski, K. Hinckley, K. Ridsen, D. Thiel, and V. Gorokhovskiy, “The task gallery: a 3D window manager,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '00, (New York, NY, USA), pp. 494–501, ACM, 2000.
- [88] S. K. Card and J. Mackinlay, “The structure of the information visualization design space,” in *Proceedings of the IEEE Symposium on Information Visualization*, pp. 92–99, IEEE, 1997.
- [89] A. Sears and J. A. Jacko, *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*. CRC, 2007.
- [90] C. Ware and P. Mitchell, “Visualizing graphs in three dimensions,” *ACM Transactions on Applied Perception*, vol. 5, no. 1, pp. 1–15, 2008.
- [91] J. Staib, S. Grottel, and S. Gumhold, “Enhancing scatterplots with multi-dimensional focal blur,” *Computer Graphics Forum*, vol. 35, no. 3, pp. 11–20, 2016.

- [92] K. W. Hall, C. Perin, P. G. Kusalik, C. Gutwin, and S. Carpendale, “Formalizing emphasis in information visualization,” *Computer Graphics Forum*, vol. 35, no. 3, pp. 717–737, 2016.
- [93] D. Weiskopf and T. Ertl, “A depth-cueing scheme based on linear transformations in tristimulus space,” tech. rep., Visualization and Interactive Systems Group University of Stuttgart, 2002.
- [94] C. T. Swain, “Integration of monocular cues to improve depth perception,” Dec. 5 2000. US Patent 6,157,733.
- [95] C. Ware, *Visual Thinking for Design*, ch. 5 - Visual Space and Time. Morgan Kaufmann, 2008.
- [96] D. Kersten, D. C. Knill, P. Mamassian, and I. Bühlhoff, “Illusory motion from shadows,” *Nature*, vol. 379, no. 31, 1996.
- [97] H. Yee, S. Pattanaik, and D. P. Greenberg, “Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments,” *ACM Transactions on Graphics (TOG)*, vol. 20, no. 1, pp. 39–65, 2001.
- [98] R. Eriksson, B. Andren, and K. E. Brunnstroem, “Modeling the perception of digital images: A performance study,” in *Proceedings of the Photonics West’98 Electronic Imaging*, pp. 88–97, International Society for Optics and Photonics, 1998.
- [99] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy, *Polygon Mesh Processing*. CRC press, 2010.
- [100] B. Vallet and B. Lévy, “Spectral geometry processing with manifold harmonics,” *Computer Graphics Forum*, vol. 27, no. 2, pp. 251–260, 2008.
- [101] B. Vallet and B. Lévy, “Spectral geometry processing with manifold harmonics,” tech. rep., 2007.
- [102] P. Cignoni, M. Corsini, and G. Ranzuglia, “Meshlab: an open-source 3D mesh processing system,” *ERCIM News*, pp. 45–46, April 2008.

- [103] Y. Gingold, A. Shamir, and D. Cohen-Or, “Micro perceptual human computation for visual tasks,” *ACM Transactions on Graphics (TOG)*, vol. 31, no. 5, p. 119, 2012.
- [104] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun, “Anisotropic polygonal remeshing,” *ACM Transactions on Graphics (TOG)*, vol. 22, no. 3, pp. 485–493, 2003.
- [105] J. J. Koenderink and A. J. van Doorn, “Surface shape and curvature scales,” *Image and Vision Computing*, vol. 10, no. 8, pp. 557–564, 1992.
- [106] B. Kulis, “Metric learning: A survey,” *Foundations and Trends in Machine Learning*, vol. 5, no. 4, pp. 287–364, 2012.
- [107] E. Garces, A. Agarwala, D. Gutierrez, and A. Hertzmann, “A similarity measure for illustration style,” *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, p. 93, 2014.
- [108] Z. Lun, E. Kalogerakis, and A. Sheffer, “Elements of style: learning perceptual shape style similarity,” *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, p. 84, 2015.
- [109] T. Liu, A. Hertzmann, W. Li, and T. Funkhouser, “Style compatibility for 3D furniture models,” *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, p. 85, 2015.
- [110] B. Saleh, M. Dontcheva, A. Hertzmann, and Z. Liu, “Learning style similarity for searching infographics,” in *Proceedings of the 41st Graphics Interface Conference*, pp. 59–64, Canadian Information Processing Society, 2015.
- [111] J. Nocedal and S. Wright, *Numerical Optimization*. Springer Science & Business Media, 2006.
- [112] J. Russell, “How shall an emotion be called,” *Circumplex Models of Personality and Emotions*, pp. 205–220, 1997.
- [113] M. Brackstone, “Examination of the use of fuzzy sets to describe relative speed perception,” *Ergonomics*, vol. 43, no. 4, pp. 528–542, 2000.

- [114] A. Lotfi, F. King-Sun, T. Kokichi, and S. Masamichi, “Fuzzy sets and their applications to cognitive and decision processes,” *Academic Press Rapid Manuscript Reproduction*, 2008.
- [115] M. Smithson, “Fuzzy sets and fuzzy logic in the human sciences,” in *Fuzzy Logic in Its 50th Year*, pp. 175–186, Springer, 2016.
- [116] T. Runkler and M. Glesner, “A set of axioms for defuzzification strategies towards a theory of rational defuzzification operators,” in *Proceedings of the 2nd IEEE International Conference on Fuzzy Systems*, vol. 2, pp. 1161–1166, 1993.
- [117] T. Akenine-Moller, E. Haines, and N. Hoffman, *Real-Time Rendering*, ch. 6. A. K. Peters, third ed., 2008.
- [118] C. Ware, C. Gobrecht, and M. Paton, “Dynamic adjustment of stereo display parameters,” *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 28, pp. 56–65, Jan 1998.
- [119] A. Bulbul, Z. Cipiloglu, and T. Capin, “A color-based face tracking algorithm for enhancing interaction with mobile devices,” *The Visual Computer*, vol. 26, no. 5, pp. 311–323, 2010.
- [120] M. Kersten, J. Stewart, N. F. Troje, and R. E. Ellis, “Enhancing depth perception in translucent volumes,” *IEEE Transactions on Visualization and Computer Graphics*, pp. 1117–1124, 2006.
- [121] D. Hoffman, A. Girshick, K. Akeley, and M. Banks, “Vergence-accommodation conflicts hinder visual performance and cause visual fatigue,” *Journal of Vision*, vol. 8, no. 3, p. 33, 2008.
- [122] U. Celikkan, G. Cimen, E. B. Kevinc, and T. Capin, “Attention-aware disparity control in interactive environments,” *The Visual Computer*, vol. 29, no. 6, pp. 685–694, 2013.
- [123] R. Rosenholtz, Y. Li, J. Mansfield, and Z. Jin, “Feature congestion: a measure of display clutter,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 761–770, ACM, 2005.

- [124] R. Rosenholtz, Y. Li, and L. Nakano, “Measuring visual clutter,” *Journal of Vision*, vol. 7, no. 2, pp. 17–17, 2007.
- [125] Y. Meng, H. Zhang, M. Liu, and S. Liu, “Clutter-aware label layout,” in *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis)*, pp. 207–214, IEEE, 2015.
- [126] R. W. Sumner and J. Popović, “Deformation transfer for triangle meshes,” in *ACM Transactions on Graphics (TOG)*, vol. 23, pp. 399–405, ACM, 2004.
- [127] I. Wald, “Utah 3D animation repository.” <http://www.sci.utah.edu/~wald/animrep/>.
- [128] R. Taylor, “Interpretation of the correlation coefficient: a basic review,” *Journal of Diagnostic Medical Sonography*, vol. 6, no. 1, pp. 35–39, 1990.
- [129] X. Chen, A. Golovinskiy, and T. Funkhouser, “A benchmark for 3D mesh segmentation,” *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 28, no. 3, 2009.
- [130] R. Song, Y. Liu, R. R. Martin, and P. L. Rosin, “Mesh saliency via spectral processing,” *ACM Transactions on Graphics (TOG)*, vol. 33, no. 1, p. 6, 2014.

Appendix A

Data

A.1 Supplemental Material for Animated Mesh Quality Assessment

Supplementary material consisting of the subjective user evaluation results described in Section 6.1.1 can be downloaded from the following link: http://cs.bilkent.edu.tr/~zeynep/phd_data/DynamicMeshVQA.zip.

The supplemental material includes the mesh files in *off* format and has the following directories:

- *Metric output* directory includes the results of our error metric proposed in Chapter 3 for each mesh used in the experiment.
- *Reference* directory includes the original mesh frames.
- *Test* directory includes the mesh frames, obtained by modifying the original frames as described in Section 6.1.1.1.
- *User responses* directory includes the user evaluations of twelve subjects and the mean subjective responses.

A.2 Supplemental Material for Learning the VQA of 3D Meshes

Supplemental material regarding the crowdsourcing experiment described in Chapter 4 and its evaluation in Section 6.2 can be obtained via the following link: http://cs.bilkent.edu.tr/~zeynep/phd_data/Learning.VQA.zip.

The zip file contains the following data:

- Mesh files used in the crowdsourcing experiment. (See Section 4.2.1 for details.)
- Tuples gathered from the crowdsourcing experiment. We provide all the tuples in raw format, how we process these tuples is explained in Section 4.4.
- Learned feature weights are also given (Section 4.4). Note that these weights are obtained through the raw method described in Section 6.2.

A.3 Supplemental Material for Perceived Depth Quality Enhancement Framework

Supplemental material to the perceived depth quality enhancement framework proposed in Chapter 5 can be found in: http://cs.bilkent.edu.tr/~zeynep/phd_data/DepthPerception.zip.

The file contents are as listed below:

- A simple fuzzy logic tutorial to construct a background for the readers who are unfamiliar with the topic.
- List of 121 fuzzy rules used in the inference engine of the system. (Section 5.2.2)
- Several videos to demonstrate the usage and results of the proposed method.