A Framework for Applying the Principles of Depth Perception to Information Visualization

ZEYNEP CIPILOGLU YILDIZ, Bilkent University-Celal Bayar University
ABDULLAH BULBUL, Bilkent University
TOLGA CAPIN, Bilkent University

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 novel solution, we automate this process by proposing a framework that determines important depth cues for the input scene and the rendering methods to provide these cues. While determining the importance of the cues, we consider the user's tasks and the scene's spatial layout. The importance of each depth cue is calculated using a fuzzy logic–based decision system. Then, suitable rendering methods that provide the important cues are selected by performing a cost-profit analysis on the rendering costs of the methods and their contribution to depth perception. Possible cue conflicts are considered and handled in the system. We also provide formal experimental studies designed for several visualization tasks. A statistical analysis of the experiments verifies the success of our framework.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Color; shading; shadowing; texture*; I.3.8 [**Computer Graphics**]: Applications

General Terms: Algorithms, Experimentation, Human Factors
Additional Key Words and Phrases: Depth perception, depth cues, information visualization, cue combination, fuzzy logic

ACM Reference Format:

Cipiloglu Yildiz, Z., Bulbul, A., and Capin, T. 2013. A framework for applying the principles of depth perception to information visualization. ACM Trans. Appl. Percept. 10, 4, Article 19 (October 2013), 22 pages.
DOI: http://dx.doi.org/10.1145/2536764.2536766

1. INTRODUCTION

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 be apparent for an information visualization to be comprehensible.

It is clear that depth is an important component of visualization, and it should be handled carefully. Depth cues construct the core part of depth perception; the Human Visual Aystem (HVS) uses depth cues to perceive spatial relationships between objects. Nevertheless, using the depth property in 3D

Authors' addresses: Z. Cipiloglu Yildiz, Bilkent University - Computer Engineering Bilkent University Department of Computer Engineering Bilkent Ankara 06800 Turkey; email: zeynep.cipiloglu@cbu.edu.tr, tcapin@cs.bilkent.edu.tr.

DOI: http://dx.doi.org/10.1145/2536764.2536766

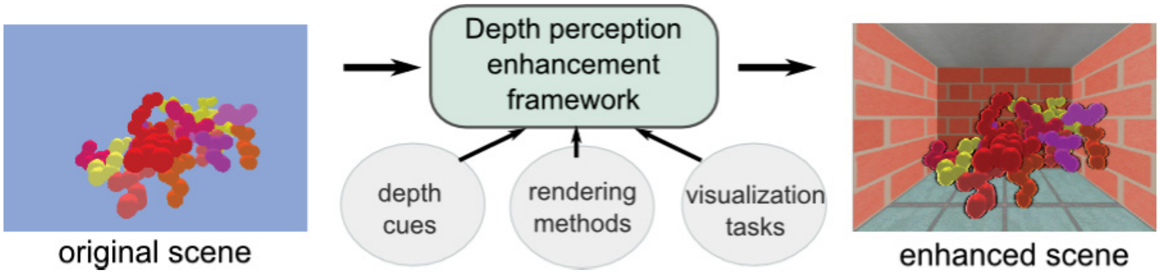


Fig. 1. Illustration of the proposed framework.

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. Therefore, a system that selects and combines appropriate depth cues and rendering methods in support of target visualization tasks would be beneficial.

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

The rest of the article is organized as follows: First, we provide an overview of the previous work on depth perception from the perspectives of the perception and information visualization fields. Then, we explain the proposed approach for automatically enhancing the depth perception of a given scene. Last, we provide the experiment design and our analysis of the results.

2. BACKGROUND

2.1 Depth Cues and Cue Combination

Depth cues can be categorized as pictorial, oculomotor, binocular, and motion-related, as illustrated in Figure 2 [Howard and Rogers 2008; Shirley 2002; Ware 2004]. The interaction between depth cues and how they are unified in the HVS 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* [Howard and Rogers 2008, Section 27.1].

In *cue averaging* models [Maloney and Landy 1989; Oruc et al. 2003], 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. According to the *cue specialization* model, different cues may

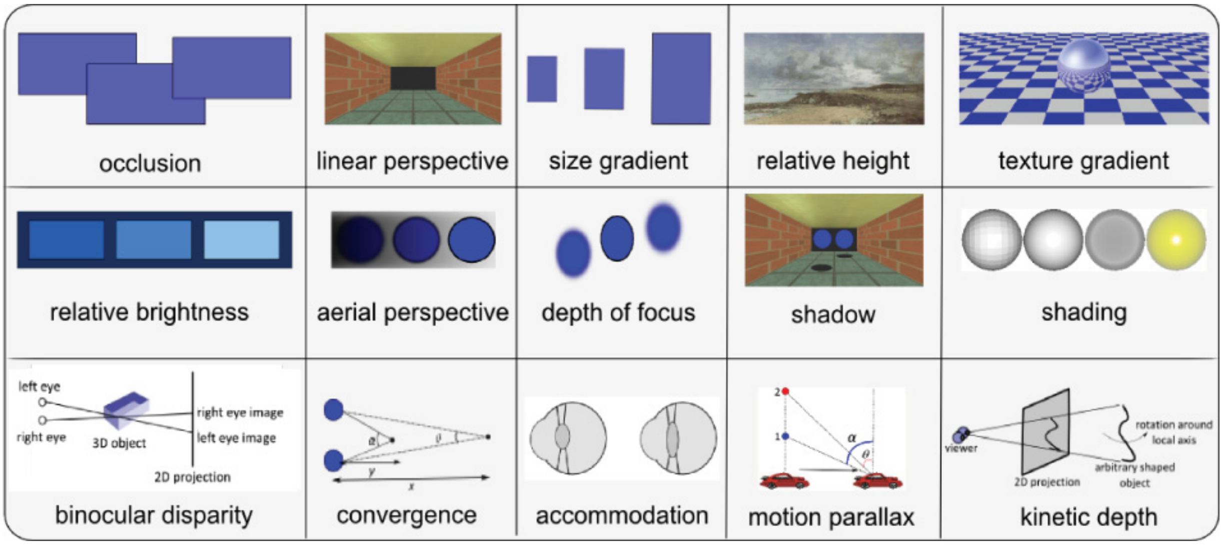


Fig. 2. Depth cues. Detailed explanations of these cues are available in Cipiloglu et al. [2010].

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 [Howard and Rogers 2008, Section 27.1]. Based on this model, several researchers consider the target task as an important factor in determining cues that enhance depth perception [Bradshaw et al. 2000; Schrater and Kersten 2000]. Ware presents a list of possible tasks and a survey of depth cues according to their effectiveness under these tasks [Ware 2004]. 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, whereas perspective becomes more effective for far distances [Howard and Rogers 2008, Section 27.1]. Cutting and Vishton [1995] 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.

Recent studies in the perception literature aim the incorporation of these models and have focused on a probabilistic model [Trommershäuser 2011; Knill 2005]. 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, the 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. [1999] 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. [1992]

explore the effects of pictorial cues and conclude that the strength of a cue is highly affected by the task.

2.2 Depth Perception for Information Visualization

Although there are a number of studies that investigate the perception of depth in scientific visualizations [Ropinski et al. 2006; Tarini et al. 2006], relatively few studies exist in the field of information visualization. The main reason behind this is that information visualization considers abstract datasets without inherent 2D or 3D semantics and therefore lacks a mapping of the data onto the physical screen space [Keim 2002]. 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 [Gershon 1997]. Some common 3D information visualization methods are cone trees [Robertson et al. 1991], data mountains [Robertson et al. 1998], and task galleries [Robertson et al. 2000]. We refer the reader to Card and Mackinlay [1997] and Sears and Jacko [2007] for a further investigation of these techniques.

Ware and Mitchell [2008] 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. First, since there are no parallel lines in graphs, perspective has no effect. Second, shading, texture gradient, and occlusion also have no effect unless the links are rendered as solid tubes. Third, shadow is distracting for large graphs. According to their experimental results, binocular disparity and motion together produce the lowest error rates.

A number of rendering methods, such as shadows, texture mapping, and fog, 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. Studies by Tarini et al. [2006], Weiskopf and Ertl [2002], and Swain [2009] 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 [2008, page 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 and dropping lines to ground plane [Ware 2004]. Moreover, Kersten et al. [1996] 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.

An earlier study by Cipiloglu et al. [2010] proposes a generic depth enhancement framework for 3D scenes; our article improves 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.

3. APPROACH

In this article, we present a framework for automatically selecting proper depth cues for a given scene and the rendering methods that provide these depth cues. To determine suitable depth cues for a given scene, we use a hybrid algorithm based on the *cue averaging*, *cue specialization*, and *range extension* models described in Section 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

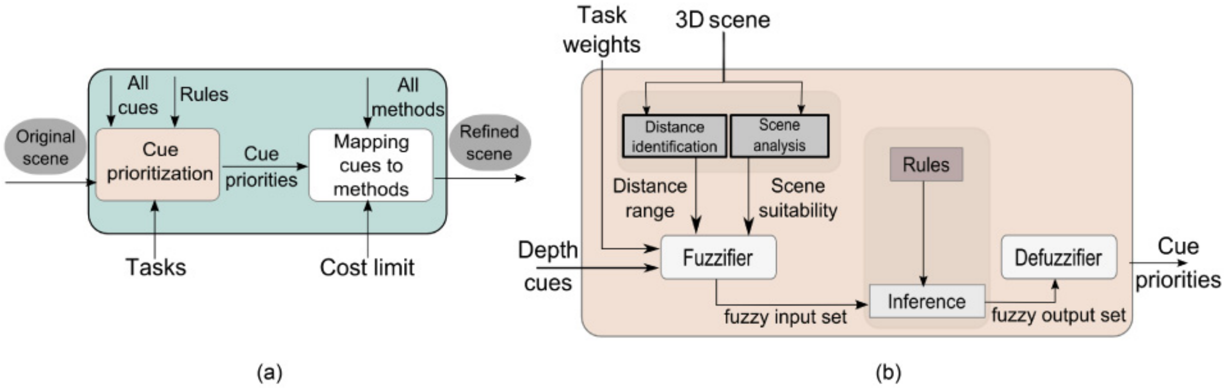


Fig. 3. (a) General architecture of the system. (b) Cue prioritization stage (corresponds to the pink box in part a).

the main factor in choosing which methods will provide the selected depth cues. Possible cue conflicts are also handled in the system.

We present the general architecture of our framework in Figure 3(a). 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 trade-off 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.

3.1 Cue Prioritization

The purpose of this stage (Figure 3(b)) 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 nonnumeric linguistic terms such as "strong" and "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 and distance, whose mathematical modeling is difficult with binary logic. Fuzzy logic has been successfully applied in modeling complex systems related to human intelligence, perception, and cognition [Brackstone 2000; Russell 1997].

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. Last, 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 [Ware 2004], we define the basic building blocks for the user's tasks as follows:

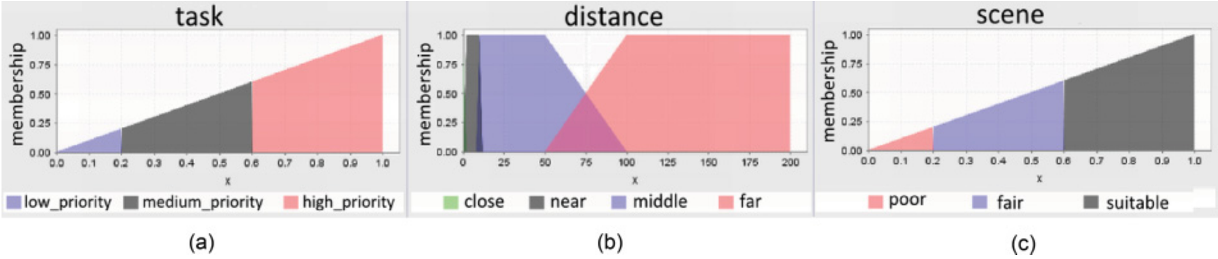


Fig. 4. Membership functions for fuzzification.

—*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, shown in Figure 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.

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 4(b)), which are constructed based on the distance range classification in Cutting and

Vishton [1995]. Based on these functions, the input variables for distance are labeled as *close*, *near*, *medium*, or *far* (Equation (1)).

$$
\begin{aligned}
\mu_{close}(x) &= -x/2+1, \quad x \in [0,2) \\
\mu_{near}(x) &= \begin{cases} x/2, & x \in [0,2) \\ 1, & x \in [2,10) \\ -x/2+6, & x \in [10,12] \end{cases} \\
\mu_{medium}(x) &= \begin{cases} x/2-4, & x \in [8,10) \\ 1, & x \in [10,50) \\ -x/50+2, & x \in [50,100] \end{cases} \\
\mu_{far}(x) &= \begin{cases} x/50-1, & x \in [50,100) \\ 1, & x \in [100,\infty) \end{cases}
\end{aligned}
\tag{1}
$$

where x is the crisp input value that corresponds to the absolute distance from the viewer and μ_{close}, μ_{near}, μ_{medium}, and μ_{far} are the functions for close, near, medium, and far, respectively.

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 [Howard and Rogers 2008, 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 earlier, 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 I shows 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 shown in Figure 4(c).

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 Howard and Rogers [2008], Ware [2004], Ware and Mitchell [2008], Hubona et al. [1999], and Wanger et al. [1992]. 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 shown in Table II. Table III 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 the online appendix.

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 nonfuzzy correspondences. This defuzzification is performed by the triangular and trapezoidal membership functions (Figure 5).

We use the Center of Gravity (COG) function as the defuzzification algorithm (Equation (2)):

$$U = \left(\int_{min}^{max} u\,\mu(u)\,du\right) \Big/ \left(\int_{min}^{max} \mu(u)\,du\right), \tag{2}$$

Table I. Scene Suitability Guidelines and Their Formulation in Our System

Depth Cue	Scene Suitability Guidelines	Heuristic
Occlusion	Occlusion is a weak cue for finding patterns of points in 3D, if the points are small [Ware and Mitchell 2008].	$scene_{occlusion} = \frac{numberOfLargePoints}{totalNumberOfPoints}$
Size gradient	If all the points have a constant and large size in a 3D scatter plot, weak depth information is obtained [Ware 2004].	$scene_{sizeGradient} = \begin{cases} 0.5, & all\ the\ points\ are\ large \\ 0, & otherwise. \end{cases}$
Relative height	The object sizes should not be too different from each other [Howard and Rogers 2008, Section 24.1.4].	$isOutlier(o) = \begin{cases} yes, & abs(size_o - avgSize) >= threshold \\ no, & otherwise. \end{cases}$ $scene_{relativeHeight} = 1 - \frac{numOfOutliers}{totalNumOfObjects}$
Shadow	If there is a large number of points in a 3D scatter plot, shadow will not help [Ware and Mitchell 2008].	$scene_{shadow} = min(1 - \frac{totalNumOfObjects}{threshold}, 0)$
	Objects should be slightly above the ground [Ware 2004].	$isSlightlyAbove(o) = \begin{cases} yes, & o_y - ground_y <= \frac{RoomHeight}{3} \\ no, & otherwise. \end{cases}$ $scene_{shadow} = \frac{numOfObjectsSlightlyAbove}{totalNumOfObjects}$
	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 [Ware 2004], [Howard and Rogers 2008, Section 24.4.2].	The light to produce shadows is appropriately located by the system; there is no need to check the scene according to this constraint.
Linear perspective	Perspective is weak if there is a large number of points in a 3D scatter plot [Ware and Mitchell 2008].	$scene_{perspective} = min(1 - \frac{totalNumOfObjects}{threshold}, 0)$
Binocular disparity	When the surfaces are textured, the effect of this cue is more apparent [Ware 2004].	$scene_{binocularDisparity} = \frac{numOfTransparentObjects}{totalNumOfObjects}$

Table II. Fuzzy Logic Operators

Operator	Operation	Fuzzy Correspondence
AND	$\mu_A(\mathrm{x})$ & $\mu_B(\mathrm{x})$	$\min\{\mu_A(\mathrm{x}), \mu_B(\mathrm{x})\}$
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 III. 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$_{binocular\_disparity}$ is *suitable* AND (minDistance is NOT *far* OR maxDistance is NOT *far*) AND tracing_data_path_in_3d_graph is *high_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$_{linear\_perspective}$ is *suitable* AND patterns_of_points_in_3d 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*

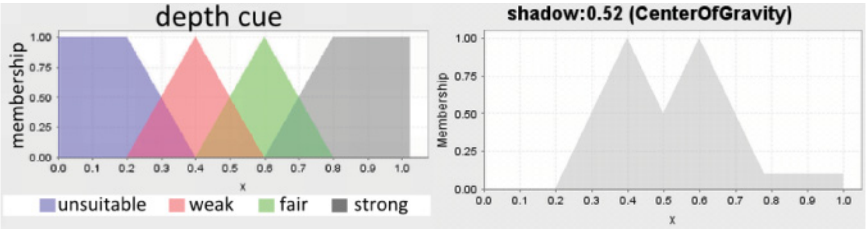


Fig. 5. Left: Membership functions for defuzzification. Right: A sample fuzzy output of the system for shadow depth cue.

Table IV. 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 [Ware 2004]
Relative brightness	Proximity luminance covariance [Ware 2004], fog [Akenine-Moller et al. 2008]
Aerial perspective	Fog, proximity luminance covariance, Gooch shading [Gooch et al. 1998]
Texture gradient	Texture mapping, bump mapping [Akenine-Moller et al. 2008], perspective projection, ground plane, room
Shading	Gooch shading, boundary enhancement [Luft et al. 2006], ambient occlusion [Bunnel 2004], texture mapping
Shadow	Shadow map [Akenine-Moller et al. 2008], ambient occlusion
Linear perspective	Perspective projection, ground plane, room, texture mapping
Depth of focus	Depth of field [Haeberli and Akeley 1990], multiview rendering [Dodgson 2005]
Accommodation	Multiview rendering [Ware et al. 1998]
Convergence	Multiview rendering [Ware et al. 1998]
Binocular disparity	Multiview rendering
Motion parallax	Face tracking [Bulbul et al. 2010], multiview rendering
Motion perspective	Mouse/keyboard-controlled motion

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 [Runkler and Glesner 1993].

The second plot in Figure 5 shows 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 COG of this region.

3.2 Mapping 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 article presents our implementation of the most common techniques, although the framework can be extended with new rendering methods. Table IV 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

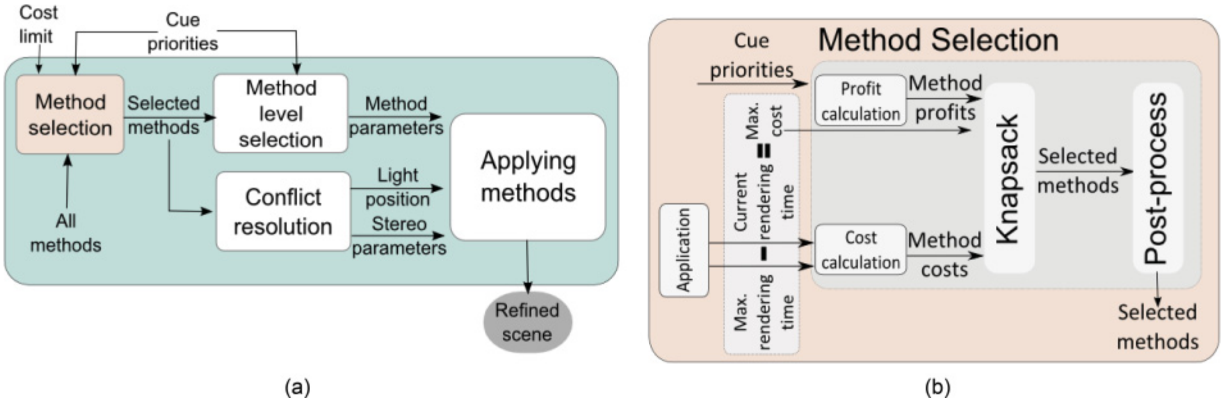


Fig. 6. (a) Cue to rendering method mapping stage. (b) Method selection step (corresponds to the pink box in (a)).

than one depth cue. For example, *texture mapping* enhances 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 6(a) 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 possible cue conflicts are avoided. If the *multiview 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 from cue conflicts. Last, the selected rendering methods are applied to the original scene with the determined parameters.

Method Selection. This step (Figure 6(b)) models the trade-off 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 (Equation (3)), based on the cue averaging model:

$$profit_i = \sum_{j \in C_i} c_{ij} \times p_j, \tag{3}$$

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 provides the same depth cue, they do not necessarily provide it in the same way and same strength. For instance, the *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 [Kersten et al. 2006], 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 Equation (4), which maximizes the total "gain" while keeping the total cost under the "maxCost":

$$\begin{aligned} Gain &= \sum_{i \in M} profit_i \times x_i \\ Cost &= \left(\sum_{i \in M} cost_i \times x_i \right) \leq maxCost, \end{aligned} \tag{4}$$

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 postprocessing 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, *multiview 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, *multiview 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.

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 (Equation (5)).

$$importance_i = \sum_{j \in C_i} c_{ij} \times p_j, \tag{5}$$

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 (Equation (6)).

$$level_i = \begin{cases} weak, & \lfloor 3 * importance_i \rfloor = 0 \\ moderate, & \lfloor 3 * importance_i \rfloor = 1 \\ strong, & \lfloor 3 * importance_i \rfloor = 2 \end{cases} \tag{6}$$

Table V 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 V. Rendering Methods with Levels

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

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 [Howard and Rogers 2008, 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 [Hoffman et al. 2008], discussion of which is beyond the scope of this article.

There are several methods to reduce the effect of convergence-accommodation cue conflict. Cyclopean scale is one method that 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 [Ware et al. 1998]. Note that the 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 *multiview 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 Equation (7) [Ware 2004], which calculates the separation in centimeters using the ratio of the nearest point to the farthest point of the scene objects. This dynamic calculation allows large eye separation for shallow scenes and smaller eye separation for deeper scenes.

$$VirtualEyeSep = 2.5 + 5.0 * (NearPoint/FarPoint)^2 \tag{7}$$

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.

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 IV; the effects of some of these methods are illustrated in Table VI via a protein model (except for face tracking).

Table VI. Some of the Rendering Methods in Our System

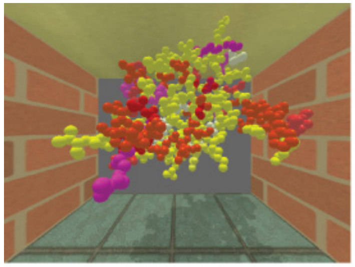
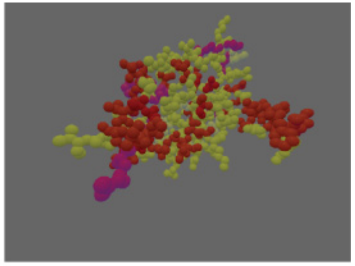
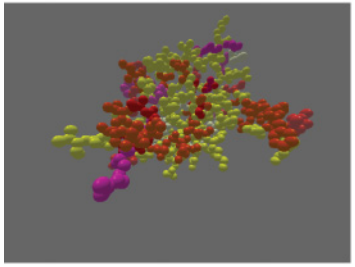
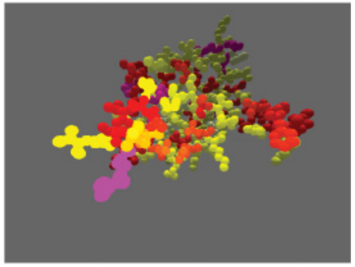
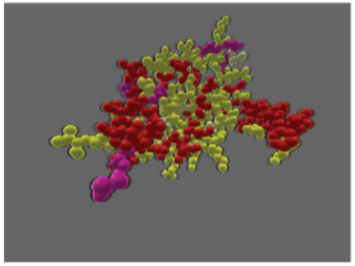
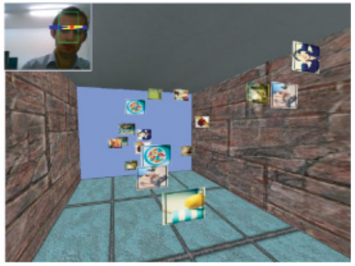
Rendering Method	Summary	Rendering Method	Summary
Shadow map 	Provided cues: Shadow. We perform a depth test from the light's point of view; the points that cannot pass the depth test should be in shadow [Akenine-Moller et al. 2008].	Gooch Shading 	Provided cues: Shading, aerial perspective. This is 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 [Gooch et al. 1998].
Fog 	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. $c_{final} = \mathrm{f} \times c_{surface} + (1-f) \times c_{fog}$	Proximity Luminance Covariance 	Provided cues: Relative brightness, aerial perspective. The luminance and saturation of the objects are changed according to their distance from the viewpoint. $\forall p \in P, \quad L'_p = \lambda \times eyeDist_p^2 \times L_p$
Boundary Enhancement 	Provided cues: Shading, occlusion. We enhance the color contrast in the scene's spatially important regions [Luft et al. 2006]. The difference between Gaussian filtered and original depth buffer values gives the spatial importance function (ΔD) . $\Delta D = G * D - D$ $\forall \ p \ \in \ P, \quad R_p, G_p, B_p \ = R_p, G_p, B_p + \Delta D_p \cdot \lambda$	Face Tracking 	Provided cues: Motion parallax, motion perspective. We use the face position to determine the viewpoint [Bulbul et al. 2010].

Table VII. Tasks in the User Experiments

Task	Tree Visualization	Graph Visualization	Scatter Plot Visualization	Surface Visualization
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

4. EXPERIMENTS

We performed four user experiments to evaluate the proposed technique for common visualization scenarios. These scenes (3D tree, graph, scatter plot, and surface visualization) represent most of the common tasks listed in Section 3.1. 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 VII summarizes the main tasks for each experiment.

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 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, which 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.

4.1 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 7(a)). 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 participants × DEPTH_CUE_SELECTION_TECHNIQUE × SCENE_COMPLEXITY = 270 trials.

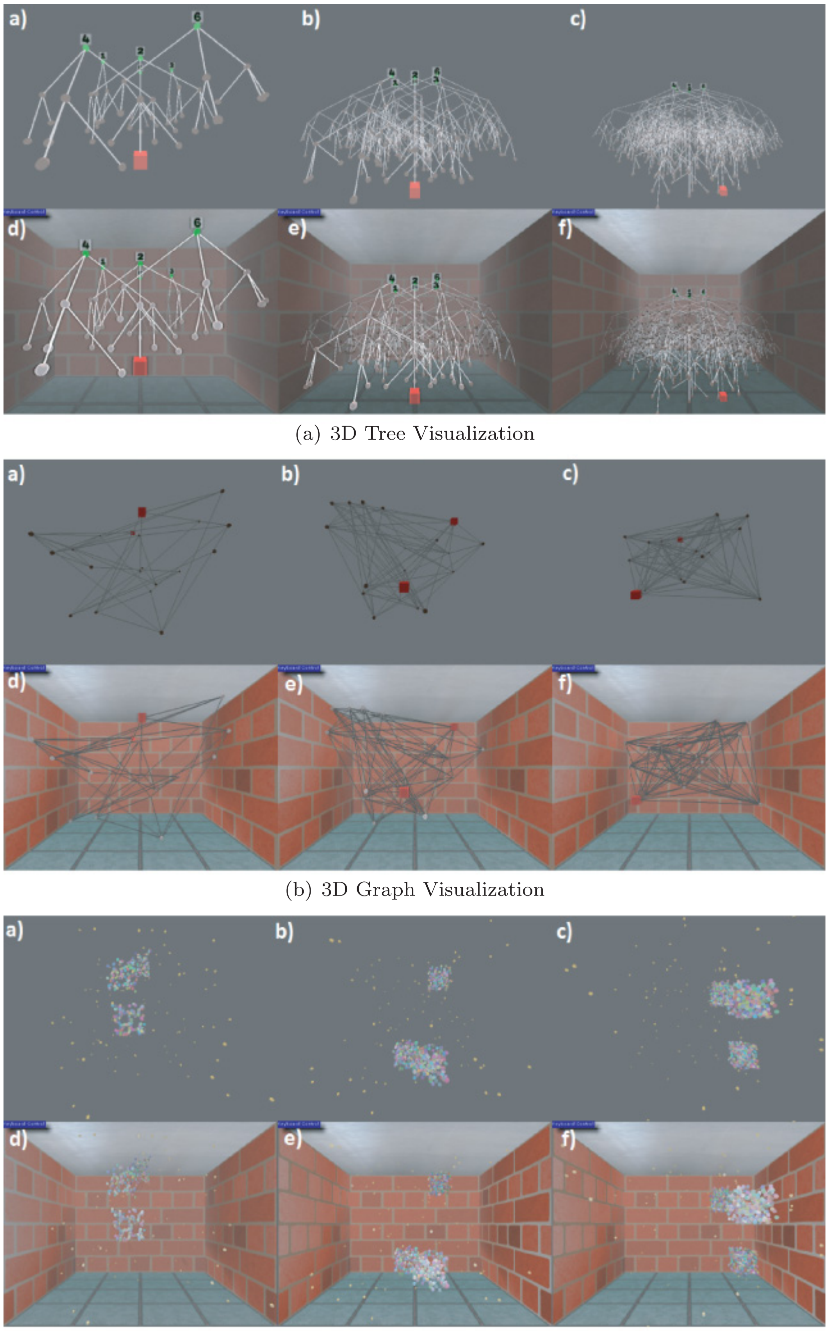
Results: For this experiment, the automatically selected methods were *proximity luminance, multiview, boundary enhancement, keyboard-controlled motion*, and *room* (Figure 7(a)). In this selection, the noteworthy methods are *multiview*,*boundary enhancement*, and *keyboard-controlled motion*. These selections are consistent with the inference in Ware and Mitchell [2008], 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 *multiview* 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 scene with LEVEL_1 complexity, the proposed framework selected the *shadow* method in addition to the above methods. This is comprehensible as the number of nodes in this level is low and thus there is more rendering budget. Second, in the most complex scene, the *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 8(a) 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

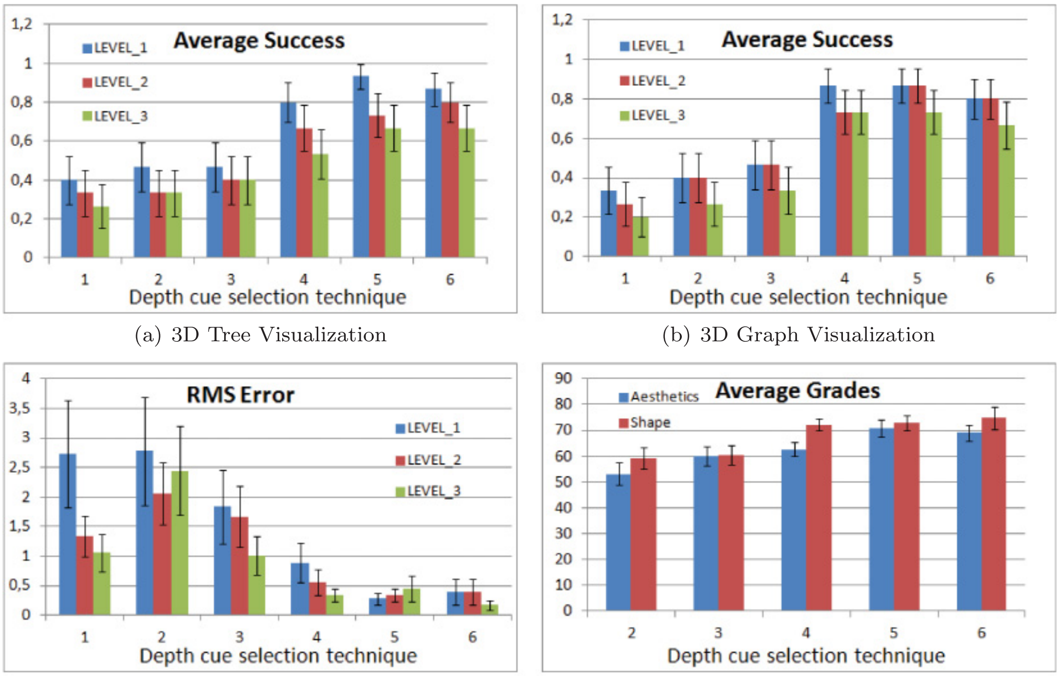


(a) 3D Tree Visualization

(b) 3D Graph Visualization

(c) 3D Scatter Plot Visualization

Fig. 7. Top rows: The scenes with basic cues. Bottom rows: The scenes with the automatically selected methods. (Multiview, face-tracking, and keyboard-controlled methods cannot be shown here.) Left to right: Scene complexity Level 1, Level 2, Level 3.



(a) 3D Tree Visualization (b) 3D Graph Visualization

(c) 3D Scatter Plot Visualization (d) 3D Surface Visualization

Fig. 8. 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.)

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, and so forth.

4.2 Experiment 2: 3D Graph Visualization

Procedure: Fifteen subjects were shown a 3D graph in which two randomly selected nodes were colored differently (Figure 7(b)). The task was to determine whether the selected nodes were linked by a path of length 2, as suggested in Ware and Mitchell [2008]. 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 ($D = 2 * |E|/\{|V| * (|V| - 1)\}$). 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 participants × DEPTH_CUE_SELECTION_TECHNIQUE × SCENE_COMPLEXITY = 270 trials.

Results: The automatically selected methods were *proximity luminance*, *multiview*, *boundary enhancement*, *keyboard-controlled motion*, and *room* (Figure 7(b)). The designer selected *multiview* 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 ($F(1, 14) = 13.27$, $p = 0.003 < 0.05$) for the DEPTH_CUE_SELECTION_TECHNIQUE on success rates. 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.

4.3 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 7(c)). 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 participants × DEPTH_CUE_SELECTION_TECHNIQUE × SCENE_COMPLEXITY = 324 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*, *multiview*, and *room* methods for each level.

We calculated the average error in user responses for the number of clusters using the RMS error (Equation (8)):

$$RMS(E) = \sqrt{(\Sigma_{i=1}^{|E|}(E_i - C_i)^2)/|E|}, \tag{8}$$

where E is the set of user responses and C is the set of correct answers. Root mean square errors are shown in Figure 8(c); 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.

4.4 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 9)—to represent surface plots in

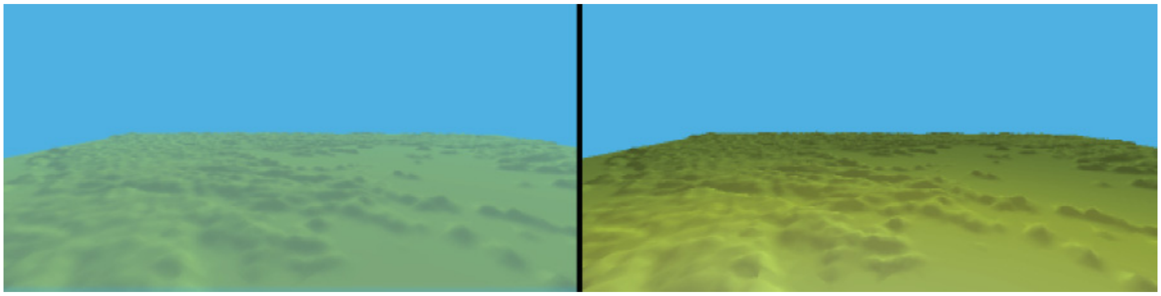


Fig. 9. Test scenes for 3D surface experiment. Left: A portion of the scene with basic cues. Right: A portion of the scene with the automatically selected methods. (Note that multiview and keyboard-controlled methods cannot be displayed here.)

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 participants × (DEPTH_CUE_SELECTION_TECHNIQUE-1) = 85 trials.

Results: *Multiview*, *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 methods and included *face tracking* instead of *multiview*.

Figure 8(d) 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.

5. CONCLUSION

We 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 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. Our framework can also be used for automatically enhancing the comprehensibility of the visualization, or as a component to suggest suitable rendering methods to application developers.

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.

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 image-based methods have recently been proposed to measure it in an image [Rosenholtz et al. 2005]. 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 for 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. Furthermore, during the realization of the depth cues, we favor making the data in the visualization comprehensible rather than realistic, as Ware [2008] also suggests. For instance, our method exaggerates the atmospheric contrast artificially to enhance the sense of depth. 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.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

ACKNOWLEDGMENTS

This work is supported by the Scientific and Technical Research Council of Turkey (TUBITAK, project number 110E029). We would also like to thank all those who participated in the experiments for this study.

REFERENCES

AKENINE-MOLLER, T., HAINES, E., AND HOFFMAN, N. 2008. Texturing. In *Real-Time Rendering* (3rd ed.). A. K. Peters, 147–200.

BRACKSTONE, M. 2000. Examination of the use of fuzzy sets to describe relative speed perception. *Ergonomics 43,* 4, 528–542.

BRADSHAW, M. F., PARTON, A. D., AND GLENNERSTER, A. 2000. The task-dependent use of binocular disparity and motion parallax information. *Vision Research 40,* 27, 3725–3734.

BULBUL, A., CIPILOGLU, Z., AND CAPIN, T. 2010. A color-based face tracking algorithm for enhancing interaction with mobile devices. *Visual Comput. 26,* 5, 311–323.

BUNNEL, M. 2004. *Dynamic Ambient Occlusion and Indirect Lighting*. Addison-Wesley.

CARD, S. AND MACKINLAY, J. 1997. The structure of the information visualization design space. In *Proceedings of the 1997 IEEE Symposium on Information Visualization (InfoVis'97).* IEEE, 92–99.

CIPILOGLU, Z., BULBUL, A., AND CAPIN, T. 2010. A framework for enhancing depth perception in computer graphics. In *Proceedings of the 7th Symposium on Applied Perception in Graphics and Visualization (APGV'10)*. ACM, New York, 141–148.

CUTTING, J. E. AND VISHTON, P. M. 1995. *Perception of Space and Motion. Handbook of Perception and Cognition* (2nd ed.). Academic Press, 69–117.

DODGSON, N. A. 2005. Autostereoscopic 3D displays. *Computer 38*, 31–36.

GERSHON, N. E. AND EICK, S. G. 1997. Information visualization applications in the real world. *IEEE Comp. Graph. Appl. 17,* 4, 66–70.

GOOCH, A., GOOCH, B., SHIRLEY, P., AND COHEN, E. 1998. A non-photorealistic lighting model for automatic technical illustration. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'98)*. ACM, New York, 447–452.

HAEBERLI, P. AND AKELEY, K. 1990. The accumulation buffer: Hardware support for high-quality rendering. *SIGGRAPH Comput. Graph. 24,* 4, 309–318.

HOFFMAN, D., GIRSHICK, A., AKELEY, K., AND BANKS, M. 2008. Vergence-accommodation conflicts hinder visual performance and cause visual fatigue. *J. Vis. 8,* 3, 33.

HOWARD, I. AND ROGERS, B. 2008. *Seeing in Depth*. Oxford University Press.

HUBONA, G. S., WHEELER, P. N., SHIRAH, G. W., AND BRANDT, M. 1999. The relative contributions of stereo, lighting, and background scenes in promoting 3D depth visualization. *ACM Trans. Comput.-Hum. Interact. 6,* 3, 214–242.

KEIM, D. 2002. Information visualization and visual data mining. *IEEE Trans. Vis. Comput. Graph. 8,* 1, 1–8.

KERSTEN, D., KNILL, D. C., MAMASSIAN, P., AND BÜLTHOFF, I. 1996. Illusory motion from shadows. *Nature 379*, 6560, 31.

KERSTEN, M., STEWART, J., TROJE, N. F., AND ELLIS, R. E. 2006. Enhancing depth perception in translucent volumes. *IEEE Trans. Vis. Comput. Graph.* 1117–1124.

KNILL, D. C. 2005. Reaching for visual cues to depth: The brain combines depth cues differently for motor control and perception. *J. Vis. 5,* 2, 103–115.

LUFT, T., COLDITZ, C., AND DEUSSEN, O. 2006. Image enhancement by unsharp masking the depth buffer. *ACM Trans. Graph. 25,* 3, 1206–1213.

MALONEY, L. T. AND LANDY, M. S. 1989. A statistical framework for robust fusion of depth information. In *Proceedings of SPIE Visual Communications and Image Processing IV*. William A. Pearlman, Philadelphia, PA, 1154–1163.

ORUC, I., MALONEY, T., AND LANDY, M. 2003. Weighted linear cue combination with possibly correlated error. *Vision Res. 43*, 2451–2468.

ROBERTSON, G., CZERWINSKI, M., LARSON, K., ROBBINS, D. C., THIEL, D., AND VAN DANTZICH, M. 1998. Data mountain: Using spatial memory for document management. In *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology (UIST'98)*. ACM, New York, 153–162.

ROBERTSON, G., VAN DANTZICH, M., ROBBINS, D., CZERWINSKI, M., HINCKLEY, K., RISDEN, K., THIEL, D., AND GOROKHOVSKY, V. 2000. The task gallery: A 3D window manager. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'00)*. ACM, New York, 494–501.

ROBERTSON, G. G., MACKINLAY, J. D., AND CARD, S. K. 1991. 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)*. ACM, New York, 189–194.

ROPINSKI, T., STEINICKE, F., AND HINRICHS, K. 2006. Visually supporting depth perception in angiography imaging. In A. Butz, B. Fisher, A. Krger, and P. Olivier, Eds., *Smart Graphics*, Lecture Notes in Computer Science Series, vol. 4073. Springer, Berlin, Germany, 93–104.

ROSENHOLTZ, R., LI, Y., MANSFIELD, J., AND JIN, Z. 2005. Feature congestion: A measure of display clutter. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'05)*. ACM, New York, 761–770.

RUNKLER, T. AND GLESNER, M. 1993. A set of axioms for defuzzification strategies towards a theory of rational defuzzification operators. In *Proceedings of the 2nd International IEEE Fuzzy Systems* Conference. IEEE, San Francisco, CA, 1161–1166.

RUSSELL, J. 1997. How shall an emotion be called. In R. Plutchik and H. R. Conte, Eds., *Circumplex Models of Personality and Emotions*. American Psychological Association, Washington, DC, 205–220.

SCHRATER, P. R. AND KERSTEN, D. 2000. How optimal depth cue integration depends on the task. *Int. J. Comput. Vision 40*, 73–91.

SEARS, A. AND JACKO, J. A. 2007. *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*. CRC Press.

SHIRLEY, P. 2002. *Fundamentals of Computer Graphics*. A. K. Peters, Ltd.

SWAIN, C. T. 2009. Integration of monocular cues to improve depth perception. US Patent 6,157,733.

TARINI, M., CIGNONI, P., AND MONTANI, C. 2006. Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Trans. Vis. Comput. Graph. 12*, 5, 1237–1244.

TROMMERSHÄUSER, J. 2011. Ideal-observer models of cute integration. In J. Trommershäuser, K. Körding and M. S. Landy, Eds., *Sensory Cue Integration*. Oxford University Press, 5–29.

WANGER, L. R., FERWERDA, J. A., AND GREENBERG, D. A. 1992. Perceiving spatial relationships in computer-generated images. *IEEE Comp. Graph. Appl. 12*, 44–58.

WARE, C. 2004. Space perception and the display of data in space. In *Information Visualization: Perception for Design*. Elsevier, 259–296.

WARE, C. 2008. Getting the information: Visual space and time. *Visual Thinking for Design*. Morgan Kaufmann, 87–106.

WARE, C., GOBRECHT, C., AND PATON, M. 1998. Dynamic adjustment of stereo display parameters. IEEE Trans. Syst. Man Cybern. A, Syst. Humans *28,* 1, 56–65.

WARE, C. AND MITCHELL, P. 2008. Visualizing graphs in three dimensions. *ACM Trans. Appl. Percept. 5,* 1, 1–15.

WEISKOPF, D. AND ERTL, T. 2002. A depth-cueing scheme based on linear transformations in tristimulus space. Tech. rep. TR–2002/08. Visualization and Interactive Systems Group, University of Stuttgart.

Received February 2012; revised September 2012; accepted May 2013