A Framework for Enhancing Depth Perception in Computer Graphics

Zeynep Cipiloglu* Bilkent University
Abdullah Bulbul† Bilkent University
Tolga Capin‡ Bilkent University

Abstract

This paper introduces a solution for enhancing depth perception in a given 3D computer-generated scene. For this purpose, we propose a framework that decides on the suitable depth cues for a given scene and the rendering methods which provide these cues. First, the system calculates the importance of each depth cue using a fuzzy logic based algorithm which considers the target tasks in the application and the spatial layout of the scene. Then, a knapsack model is constructed to keep the balance between the rendering costs of the graphical methods that provide these cues and their contibution to depth perception. This cost-profit analysis step selects the proper rendering methods. In this work, we also present several objective and subjective experiments which show that our automated depth enhancement system is statistically (*$p < 0.05$*) better than the other method selection techniques that are tested.

Keywords: computer graphics, depth perception, perceptually-aware rendering, depth cues, cue combination

1 Introduction

3D rendering methods and display technologies such as head-mounted displays and autostereoscopic displays have advanced significantly in the past few years. This rapid development in the 3D technology also brings the problem of better visualization of the 3D content. People desire to see realistic 3D scenes, especially when they are playing games or watching 3D movies. Therefore, usage of the third dimension in an effective manner has become essential.

Providing correct depth information during the design of a 3D scene is very important; however, it is not easy for an application designer to deal with this additional issue. It requires understanding of real-life depth cues that are used to perceive the spatial relationships between the objects by the human visual system. Therefore, an automated system that aids the 3D application designer in improving the depth perception of an input 3D scene would be very beneficial.

To develop such a system, an algorithm that combines different depth cues and rendering methods is needed. Although the first approach is to provide all possible depth cues, this is not always the best solution. Providing all the cues may lead to problems such as high computational cost, unnecessary scene complexity, and cue conflicts. Hence, a system designed to enhance depth perception should consider the aspects such as the nature of the task, spatial layout of the scene and computational costs of the methods. A number of methods have been proposed to improve depth perception in 3D computer-generated imagery. However, these methods are generally limited and insufficient, since they are either proposed to operate on specific domains, or they do not provide a solution to unify different depth enhancement methods appropriately. Hence, a comprehensive system that combines existing depth enhancement methods properly according to the given scene is required.

In this work, we present a framework that automatically selects the proper depth enhancement methods for the given scene, depending on the task, spatial layout of the scene, and the costs of the rendering methods. The contributions of this study are as follows:

- A fuzzy logic based algorithm for automatically determining the proper depth cues for the given scene and task,
- A knapsack model for selecting proper depth enhancement methods, evaluating the cost and profit of these methods,
- A formal experimental study to evaluate the effectiveness of the proposed algorithms.

2 Background

In this section, we examine the depth cues and cue combination models from the perception point of view, and analyze the rendering methods used for enhancing depth perception in computer graphics.

2.1 Depth Cues and Cue Combination

Depth cues, which help the human visual system to perceive the spatial relationships between the objects, construct the core part of depth perception. These visual cues can be categorized as pictorial, oculomotor, binocular, and motion-related cues as illustrated in Tables 1, 2, 3, and 4 respectively, based on the studies [Howard and Rogers 2008], [Shirley 2002], and [Ware 2004].

How the human visual system unifies different sources of depth cues into a single knowledge is a widely-investigated topic. Many studies have investigated the interaction of different cues. There is not a single, accepted cue combination model, however. The mostly-accepted models of cue interaction are generally the variations of the following categories: *cue averaging, cue dominance, cue specialization, range extension,* and *probabilistic models* [Howard and Rogers 2008].

Most of the research on cue combination focuses on the *cue averaging* models, in which each cue is associated with a weight determining its reliability. The overall perception is obtained by summing up the individual depth cues multiplied by their weights. The studies by Maloney and Landy [1989] and Oruc et al. [2003] are example approaches based on the *cue averaging* model.

Cue dominance is a model proposed to consider cue conflicts. According to this model, if two depth cues provide conflicting information, one of them may be suppressed and the final percept may be based on the other cue [Howard and Rogers 2008].

Cue specialization models are based on the idea that different cues may be used for interpreting different components of a stimulus. Several researchers consider the target task as an important factor on determining the cues that enhance the depth perception [Bradshaw et al. 2000], [Schrater and Kersten 2000]. Ware presents a comprehensive list of the tasks and a survey of the depth cues according to their effectiveness under these tasks [Ware 2004]. For

*e-mail: zeynep@cs.bilkent.edu.tr
†e-mail:bulbul@cs.bilkent.edu.tr
‡e-mail:tcapin@cs.bilkent.edu.tr

APGV 2010, Los Angeles, California, July 23 – 24, 2010.
© 2010 ACM 978-1-4503-0248-7/10/0007 $10.00

Table 1: Pictorial Depth Cues

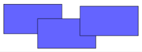
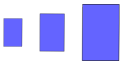
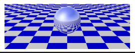
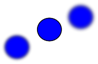
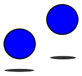
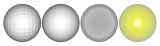
Depth Cue	Details
Occlusion 	If an object overlaps some part of the other, it is known that the blocked object is further. It only gives information about the order of the objects.
Linear Perspective	In real life, parallel lines seem converging, as they move away, towards the horizon.
Size Gradient 	The size of an object is inversely proportional to the distance from the viewer. Hence, larger objects seem closer to the viewer.
Relative Height	When the world is divided by a horizon; the objects closer to the horizon seem further under the horizon, and seem closer above the horizon. (Painting: "The Coast of Protrieux" by Eugene Boudin.)
Texture Gradient 	In textured surfaces, when the surface gets further away, the texture becomes smoother and finer.
Relative Brightness	The intensity level of an object varies with depth. Brighter objects are prone to be seen closer.
Aerial Perspective	Further objects seem hazy and bluish due to the scattering of the light in the atmosphere. Hence, aerial perspective increases the perceived distance. (Painting: "Near Salt Lake City" by Albert Bierstadt.)
Depth-of-focus 	Our eyes fixate on different objects in the world to bring them to sharp focus. The objects other than the object in the sharp focus seem blurry.
Shadow 	If the object is in shadow, it is further from the light source. Shadows of the objects on the ground facilitate the perception of the objects' relative positions by connecting them to the ground plane.
Shading 	Shading provides important information about the surface shape by enabling the observer to distinguish between convexities and concavities.

Table 2: Oculomotor Depth Cues

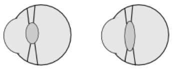
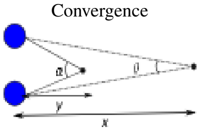
Depth Cue	Details
Accommodation 	The process of the distortion in the eye lens to fixate on a point is called accommodation. The amount accommodation that the eye lens performs to focus on an object varies with depth.
Convergence 	It is the fixation of the eyes towards a single location in order to maintain a single binocular vision. The increase in the convergence angle indicates that the fixation point comes closer ($\alpha > \theta, y < x$).

Table 3: Binocular Depth Cues

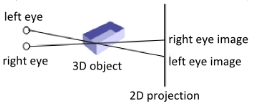
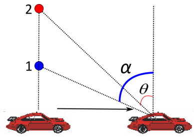
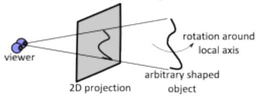
Depth Cue	Details
Binocular Disparity  left eye right eye 3D object right eye image left eye image 2D projection	Left and right eyes look at the world from slightly different angles, which results in slightly different retinal images. This provides binocular vision.

Table 4: Motion Related Depth Cues

Depth Cue	Details
Motion Parallax 	As the user moves his eyes side to side, the images of the closer objects move more in the visual field than those of further objects. This is because the angular speed of an object is inversely related to the distance from the viewer ($z_1 < z_2$, $\alpha > \theta$).
Kinetic Depth  viewer rotation around local axis arbitrary shaped object 2D projection	The overall shape of an arbitrary object can be perceived better when it rotates around its local axis, since the ambiguities due to the projection from 3D to 2D are resolved with the rotation.

instance, according to his investigations, perspective is a strong cue when the task is "judging the relative positions"; while it becomes ineffective for the task "tracing data paths in 3D graphs". As an another example, stereoscopic viewing and kinetic depth together significantly increased the accuracy when the task is "tracing data paths in 3D graphs" [Ware and Mitchell 2008].

According to the *range extension* model, different cues may be effective in different ranges. For example, binocular disparity is a strong cue in the near distances, while perspective becomes more effective at far distances [Howard and Rogers 2008]. In this sense, 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 human visual system to different depth cues in each range.

Lastly, Bulthoff and Yuille [1996] present a *probabilistic* approach to estimate the cue interactions by considering various prior assumptions of the human visual system on the scene and material attributes, using a Bayesian framework. An example prior assumption is that human visual system assumes that light is stationary and comes from left-above [Howard and Rogers 2008].

2.2 Rendering Methods for Depth Enhancement

Based on the depth cues and principles discussed in the previous subsection, 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.

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 a commonly-used method to make the height between the object and the ground more visible [Ware 2004]. 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 technique for enabling the user to judge the sizes of unknown objects easier [Ware 2004].

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 [Haeberli and Akeley 1990]. *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, Dosher 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 [Ware 2004].

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. For instance in Bunnel's [2004] 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 [Gooch et al. 1998], [Rheingans and Ebert 2001]. 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 [Nienhaus and Doellner 2003], [Markosian et al. 1997]. An image-space approach is proposed by Luft et al. [2006] 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 [Dodgson 2005]. Rendering on 3D displays is an active topic in itself [Bulbul et al. 2010b].

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. [2010a] 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.

Other: There are also studies that combine multiple depth enhancement methods. Tarini et al. [2006] propose a system for enhanced visualization of molecular data. In this work, ambient occlusion and edge cueing schemes are applied for molecular visualization. Weiskopf and Ertl [2002] developed a more comprehensive depth cueing framework based on the principles of color vision. In this study, only color properties such as intensity and saturation are employed for providing depth cues by transforming the color values according to the distance. The literature survey indicates that there is a lack of comprehensive framework for uniting different methods of depth enhancement.

3 Approach

We propose a framework for automatically selecting the proper depth cues for the given scene and the rendering methods that provide these depth cues. While automatically selecting the suitable cues and rendering methods for the given scene, we consider the following factors: the distance of the objects in the scene, the user's tasks in the application, the spatial layout of the scene, and the costs of the rendering methods. Hence, our algorithm can be considered as a hybrid of the *cue averaging*, *cue specialization*, and *range extension* models of cue combination described in Section 2.1.

The general architecture of the automatic depth enhancement process can be seen in Figure 1. Our approach first determines the priority of each depth cue based on the task, distance of the objects, and scene attributes using fuzzy logic. The next stage is to select the suitable rendering methods that provide the cues which are determined as high priority in the previous stage. In this stage, we consider the costs of the methods and try to solve the cost and cue priority trade-off. After selecting the proper rendering methods, we apply these methods to the given scene and produce a refined scene with a better depth perception.

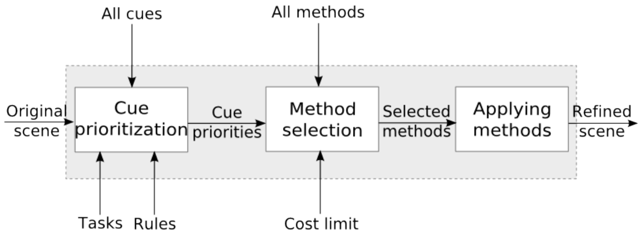


Figure 1: General Architecture of the System.

3.1 Cue Prioritization

The aim of this stage is to determine the important depth cues for the given scene. This stage analyzes the user's task and the spatial layout of the scene; and a priority value, which represents the effectiveness of that cue for the given scene, is assigned to each depth cue. The general architecture of this stage is shown in Figure 2.

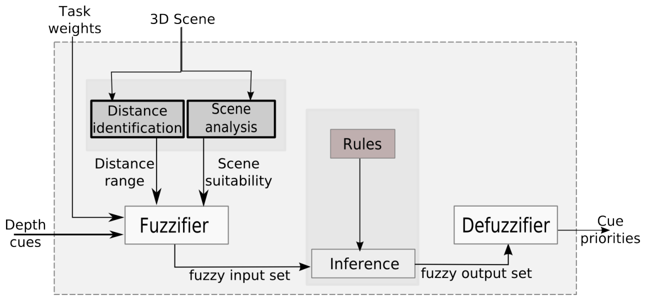


Figure 2: Fuzzy Cue Prioritization Stage.

The system maintains a cue priority vector which stores a priority value, in the range of (0, 1), for each depth cue. At the end of this stage, cue priority values are calculated which show the strength of the corresponding cue for the given scene.

To calculate the cue priorities, we have selected fuzzy logic as the decision making method due to a number of reasons. Firstly, fuzzy logic has been used successfully to model complex systems such as human intelligence, perception, and cognition [Brackstone 2000], [Russell 1997]. Secondly, the problem of combining different depth cues depends on many factors such as task, distance, and etc. Fuzzy logic systems provide a robust solution for this kind of multi-input systems whose mathematical modeling is difficult.

Fuzzification

In fuzzy logic, linguistic variables such as *age, temperature* are used, instead of numerical values. The goal of this step is to represent the variables linguistically to activate the rules defined in terms of linguistic variables. In this step, numerical variables are converted to fuzzy set of variables.

Task weights are the first input 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 user's tasks as follows:

- Judging the relative positions of objects in space
- Reaching for objects
- Surface target detection
- Tracing data paths in 3D graphs

- Finding patterns of points in 3D space
- Judging the “up” direction
- The aesthetic impression of 3D space (Presence)

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 tasks. In our algorithm, a fuzzy linguistic variable between 0 and 1 is kept for each task. These values correspond to the weights of the tasks in the application and 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 which divide the region into three (Figure 3). Using these membership functions and the task weights, each task is labeled as “low_priority”, “medium_priority”, or “high_priority” to be used in the rule base.

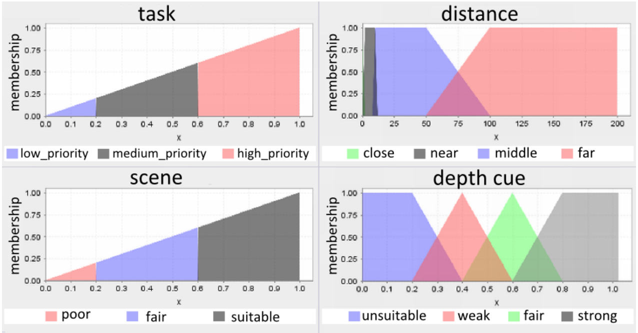


Figure 3: Membership functions.

Distance of the objects to the user is another input to the system, as *range extension* is a cue combination model that constructs 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 viewpoint, and mapped to the range 0-100.

To fuzzify these variables, we use the trapezoidal membership functions (Figure 3), which are constructed based on the distance range classification in [Cutting and Vishton 1995]. Based on these functions, input variables for distance are labeled as “close”, “near”, “middle”, or “far” (Eq. 1).

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

where *x* is the crisp input value which 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 may affect the behaviors of different cues in different ways. For instance, if there are a large number of points in a 3D scatter plot, cast shadows do not contribute to the depth perception [Ware 2004].

To handle these scene specific parameters in our system, we define another input linguistic variable “scene”, for each depth cue. Initially, the scene is assumed to be suitable for each depth cue. Then, the scene is analyzed separately for each depth cue and if there is an inhibitive situation similar to the cases described above, the “scene” value for that cue is penalized. For instance, according to Madison et al., cast shadows give the best results when the objects are slightly above the ground plane [Ware 2004]. To handle this situation, we count the number of objects which are slightly above the ground plane in the scene, and we calculate the $scene_{shadow}$ as the ratio of the number of objects that are slightly above the ground plane to the total number of objects in the scene. The “scene” values are fuzzified as “poor”, “fair”, or “suitable” using the piecewise linear membership function (Figure 3).

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. For each depth cue, there is 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 5. Table 6 contains sample rules used to evaluate the priority values of different depth cues. Current rule base consists of 106 rules in total.

Table 5: Fuzzy logic operators used in the evaluation of the rules

Operator	**Operation**	**Fuzzy Correspondance**
AND	μ_A(x) & μ_B(x)	min{ μ_A(x), μ_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 6: Sample fuzzy rules

IF scene is *suitable* AND tracing_data_path_in_3d_graph is *high_priority* THEN shadow is *weak*

IF scene is *suitable* AND (minDistance is *far* OR maxDistance is *far*) THEN aerial_perspective is *strong*

IF scene is *suitable* AND (minDistance is NOT *far* OR maxDistance is NOT *far*) AND asthetic_impression is *low_priority* THEN binocular_disparity is *strong*

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 3). As the defuzzification algorithm, we use the “center of gravity” (COG) function in Eq. 2.

$$U = \frac{\int_{min}^{max} u\,\mu(u)\,du}{\int_{min}^{max} \mu(u)\,du} \qquad (2)$$

where *U* is the result of defuzzification, *u* is the output variable, μ is the membership function after inference, *min* and *max* are the lower and upper limits for defuzzification, respectively [fcl 1997].

In Figure 4, a sample demonstration of the overall fuzzy cue prioritization stage for the shadow depth cue is illustrated. At the end of this stage, priority values for each depth cue are produced.

3.2 Method Selection

After the cue prioritization, the next stage is to support the cues with high priority, using proper rendering methods. However, there are different depth enhancement methods that provide the same cue, as well as methods that can provide multiple cues at the same time.

Table 7 shows the depth cues and the rendering methods we have implemented to provide these cues. In the table, some of the methods are labeled as “helper”. This means that these methods do not provide the corresponding depth cue directly, however they either

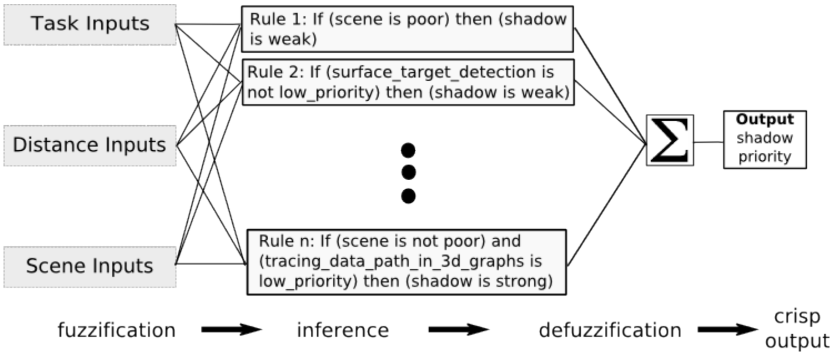

Task Inputs
Distance Inputs
Scene Inputs
Rule 1: If (scene is poor) then (shadow is weak)
Rule 2: If (surface_target_detection is not low_priority) then (shadow is weak)
Rule n: If (scene is not poor) and (tracing_data_path_in_3d_graphs is low_priority) then (shadow is strong)
Σ
Output shadow priority
fuzzification
inference
defuzzification
crisp output

Figure 4: Demonstration of the fuzzy cue prioritization stage on shadow depth cue.

increase the effect of the cue or there is a dependency between the rendering methods that provide this cue. For instance, *perspective projection* does not provide *texture gradient* itself; however, when the surface is textured, it increases the effect of *texture gradient* cue.

Table 7: Rendering methods corresponding to the depth cues

Depth Cues	**Depth Enhancement Methods**
Size gradient	Perspective projection
Relative height	Perspective projection, dropping lines to ground
	Helper: ground plane
Relative brightness	Proximity luminance, fog
Aerial perspective	Fog, proximity luminance, Gooch shading
Texture gradient	Texture mapping, bump mapping
	Helper: perspective projection, ground plane, room
Shading	Gooch shading, boundary enhancement, ambient occlusion, texture mapping
Shadow	Shadow map, ambient occlusion
	Helper: ground plane, room
Linear perspective	Perspective projection
	Helper: ground plane, room, texture mapping
Depth of focus	Depth-of-field, multi-view rendering
Accommodation	Multi-view rendering
Convergence	Multi-view rendering
Binocular disparity	Multi-view rendering
Motion parallax	Face tracking, multi-view rendering
Motion perspective	Mouse/keyboard controlled motion

The architecture for method selection stage is shown in Figure 5. The inputs to the system are the cue priority vector from the previous stage, current frame rate in frames per second (FPS) from the application, and a target FPS set by the user. The FPS values are used to calculate the maximum cost (Eq. 3).

$$maxCost = currentFPS - targetFPS \tag{3}$$

The core part of this stage is modeling the trade-off between the cost and profit of a depth enhancement method as a *Knapsack* problem. According to this approach, a "profit" and a "cost" value are assigned to each depth enhancement method. "Profit" is used to quantify the contribution of a method to the enhancement of depth perception in the given scene and calculated as a weighted sum of the priorities of the depth cues provided by this method (Eq. 4), based on the "cue averaging" model:

$$profit_i = \sum_{j \in C_i} c_j \times p_j \tag{4}$$

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_j is a constant that represents how much method *i* contributes to the cue *j*.

We calculate the "cost" of a rendering method as the reduction in the current FPS caused by this method. We define a cost reduction table which keeps a FPS reduction value (R_i) in percentages for each method. These values are obtained empirically, as the average reduction in FPS due to the corresponding method, through different scenes. Then, the cost of each rendering method is calculated using this table and the current FPS at run time (Eq. 5).

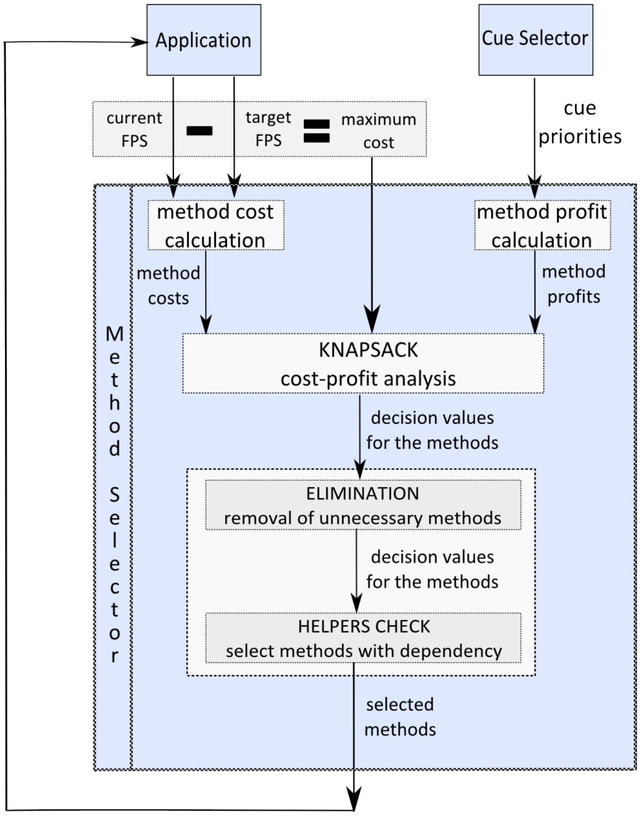

Application
Cue Selector
current FPS
target FPS
maximum cost
cue priorities
Method Selector
method cost calculation
method profit calculation
method costs
method profits
KNAPSACK cost-profit analysis
decision values for the methods
ELIMINATION removal of unnecessary methods
decision values for the methods
HELPERS CHECK select methods with dependency
selected methods

Figure 5: Method Selection Stage.

$$cost_i = R_i \times currentFPS/100 \tag{5}$$

The Knapsack problem in Eq. 6, which maximizes the total "gain" while keeping the total cost under the "maxCost", is solved using the dynamic programming approach:

$$Gain = \sum_{i \in M} profit_i \times x_i$$

$$Cost = (\sum_{i \in M} cost_i \times x_i) \leq maxCost \quad (6)$$

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

At the end of the cost-profit analysis step, we obtain the decision values of each depth enhancement method. It is possible to use these values directly as the final decisions; however, we apply two more steps to improve the quality of the system.

The purpose of the elimination step is to eliminate additional cost by unselecting some of the methods that provide only the cues that are already provided by other methods. For example, although the main purpose of *multi-view rendering* is providing *binocular disparity*, it also creates the *depth-of-focus* effect. Hence, there is no need to increase the rendering cost with *depth-of-field* method which only provides *depth-of-focus* cue, if a more "profitable" method is already selected. For such kind of methods, see Table 7.

Another post-processing step is checking the use of helpers, in which the methods that are labeled as "helper" for the corresponding method in Table 7 are checked and selected if they are not already selected. For instance, if *shadow mapping* method is selected but *ground plane* is not enabled, this step selects *ground plane* and updates the total cost and profit accordingly.

The above procedure is repeated multiple times to obtain a more accurate estimation in the final FPS. The number of passes is bounded by a *threshold* value. Three passes generally result in accurate estimations. Note that other cost limitations can also be taken into account, such as memory requirements. It is also possible to extend the system to consider multiple limitations at the same time, using multiply constrained knapsack problem.

3.3 Methods for Enhancing Depth Perception

After receiving the decision values of the rendering methods, the last step is to apply these methods to the given scene. Our current implementation supports the rendering methods shown in Table 7; only the important ones are explained in this section.

Shadow Map: In our framework, shadow is obtained by using shadow maps. In this method, a depth test is performed from the light's point of view and the points that cannot pass the depth test should be in shadow.

Fog: We implemented a fog rendering method to provide the aerial perspective cue, in which the final color of each pixel (c_{final}) is interpolated between the surface color ($c_{surface}$) and the fog color (c_{fog}) according to the fog factor (*f*) which depends on the distance from the viewpoint (Eq. 7).

$$c_{final} = f \times c_{surface} + (1 - f) \times c_{fog} \quad (7)$$

Proximity Luminance: This method changes the luminance of the objects according to their distance from the viewpoint to provide relative brightness and aerial perspective cues. We first convert the color value from RGB space to HSL space, modify the luminance value according to the pixel's distance using Eq. 8, and convert the modified color back to RGB space.

$$\forall p \in P, \quad L'_p = \lambda \times eyeDist_p^2 \times L_p \quad (8)$$

where *P* is the set of all pixels, L_p is the current luminance value of pixel *p*, L'_p is the modified luminance value of pixel *p*, $eyeDist_p$ is the distance of pixel *p* to the viewpoint, and λ is a constant that determines the strength of the method.

Boundary Enhancement: We enhance the important edges using the depth buffer based method [Luft et al. 2006]. According to this method, the derivative of the depth values is used to calculate the "spatial importance" function which indicates the spatially important areas in the scene. This function (ΔD) is calculated as the difference between the original (*D*) and the Gaussian filtered (*G*) depth buffer (Eq. 9).

$$\Delta D = G * D - D \quad (9)$$

Here, the $*$ operator stands for convolution. After calculating the spatial importance function, the color contrast of the whole image is modified by adding the spatial importance value (ΔD_p) multiplied by a constant (λ) to each color channel (R_p, G_p, B_p) (Eq. 10).

$$\forall p \in P, \quad R_p, G_p, B_p = R_p, G_p, B_p + \Delta D_p \cdot \lambda \quad (10)$$

Face Tracking: We use the face tracking system by Bulbul et al. [Bulbul et al. 2010a] to provide motion parallax. The face position is used in the application to determine the viewpoint.

Multi-view Rendering: We provide binocular and oculomotor cues using multi-view rendering which is obtained by a 9-view 3D lenticular display. The scenes rendered from different viewpoints are combined using the interlacing operations for the display.

4 Experimental Study

The success of the proposed system was evaluated by several experiments. In this paper, we selected two important and common tasks: "judging relative positions" and "surface target detection (shape perception)" from those listed in Section 3.1. For the first task, we performed both an objective and a subjective experiment, while we tested the second task based on a subjective study.

4.1 Objective Experiment

Subjects: We performed the objective experiment on 14 subjects: 9 males and 5 females with a mean age of 24.4. All the subjects have self-reported normal or corrected vision. They were voluntary graduate or undergraduate students with computer science background. They were not informed about the purpose of the experiment.

Procedure: An experimental setup similar to the one in Wanger's study [Wanger 1992] was used. Subjects were given a scene with a randomly positioned test object in a region whose boundaries were indicated visually and asked to estimate the z position (between 0-50) of the given object (Figure 6). There was no time limit and they entered their estimations using a slider-like widget (Figure 6).

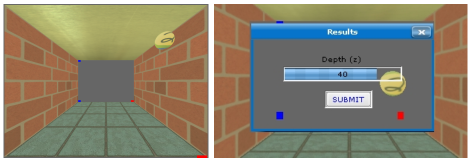


Figure 6: Left: The scene used in the objective experiment. (Red bars and blue bars show the boundaries in z and y, respectively.) Right: Submission of the results.

The above procedure was repeated for five different conditions. In the first case there were no depth cues in the scene other than the perspective projection which is indicated by the lengths of the limit indicator bars shown in Figure 6. In the second test case, which rendering methods will be used were decided randomly at run time. The third case was also a random selection with a cost limit. In other words, each method was applied randomly only if it did not decrease the frame rate under the given cost limit. The cost limit was the same as the one used in automatic selection case. The fourth case was the application of all the depth enhancement methods and in the last case, the methods that will be applied to the scene were chosen using our algorithm.

Results: RMS error for each test case is calculated using Eq. 11, where T is the set of estimations and R is the set of real positions.

$$RMS(T) \quad = \quad \sqrt{\frac{\Sigma_{i=1}^{|T|}(T_i - R_i)^2}{|T|}} \qquad (11)$$

Figure 7 shows the RMS errors for each test case. As shown in the figure, our algorithm gives the best results with RMS error of only 3.1%. Hence, it is even better than applying all the methods. A possible reason for this result is that applying all the methods may cause cue conflicts and confuse the subjects. Also, when all the methods are applied, the frame rate decreases and this situation distracts the user. Therefore, we consider the third case as the strongest competitor of our case because of the cost limit. The results show that our algorithm results in more than 2 times better estimations in depth, compared to the third case. We also performed a *paired samples t-test* on the experimental data. We compared the results of each test case to the results of our case and this statistical analysis show that the difference between our algorithm and the other selection techniques is statistically significant ($p < 0.05$).

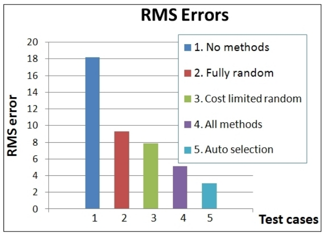


Figure 7: RMS errors for objective depth judgement.

4.2 Subjective Experiments

We have also performed subjective tests for the tasks "depth perception" and "shape perception" separately.

Subjects: For the depth judgement task, 21 subjects (17 males, 4 females) with a mean age of 24.2 were participated in the experiment; and 11 subjects (9 males, 2 females) whose mean age is 24.1, evaluated the scenes for the shape judgement task. The subjects were among the voluntary graduate and undergraduate students who have self-reported normal or corrected vision. They were not informed about the purpose of the experiment.

Procedure: For each task, the subjects were shown a scene (Figure 8) and asked to grade the given scene between 0 and 100. At the beginning, the subjects were informed about the procedure and the grading criteria. They were told that they should evaluate the ease of understanding the relative distances between the objects in the scene, for the depth judgement; and the perception of the shapes (curvatures, convexity/concavity, etc.) of the objects, for the shape judgement. At first, the scene without any cues was shown to the subjects and they were told that the grade of this scene is 50 and they graded the other test cases by comparing them to the original scene. Test cases were the same with the objective experiment.

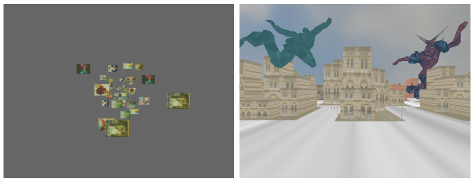


Figure 8: The scenes (without cues) for depth judgement (left) and shape judgement (right) tasks.

Results: Our automatic selection framework suggested the methods keyboard control, room, multi-view rendering, and proximity luminance for the depth judgement task; while boundary enhancement, face tracking, bump mapping, Gooch shading, shadow, and proximity luminance were suggested for the shape judgement task. Here, the remarkable point is that shape-from-shading and structure-from-motion cues are dominant among the selected methods for shape judgement task. The average grades for each task is shown in Figure 9. Our algorithm has the highest grades for both tasks: about 87 for depth and 73 for shape judgement. Since some of the error bars are overlapping, we performed a *paired samples t-test* for the results, which showed that our algorithm is statistically ($p < 0.05$) better than the other selection techniques for both tasks.

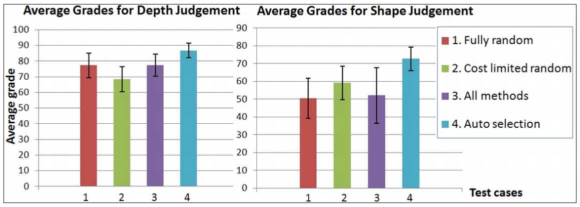


Figure 9: Results for subjective depth (left) and shape (right) judgement tasks. (Error bars show the 95% confidence intervals.)

5 Conclusion

In this work, we have presented a framework that proposes methods for enhancing the depth perception in a given 3D scene. Our algorithm automatically decides on the important depth cues using fuzzy logic and selects the rendering methods which provide these cues based on the Knapsack problem. In this depth enhancement framework, we consider several factors: the target tasks, spatial layout of the scene, and the costs of the rendering methods. Our framework can either be used for automatically enhancing the depth perception of a given scene, or as a component that suggests suitable rendering methods to application developers. Figure 10 shows several examples of the results of our system.

We evaluated our system using objective and subjective experimental studies for depth and shape judgement tasks. According to the results of the objective experiment for depth judgement, average RMS error of our system is only 3.1%. In addition, subjective experiments show that the scenes which are enhanced using our algorithm have the highest scores among other test cases for both tasks, with a statistically significant ($p < 0.05$) difference (Figure 9). On the other hand, the main limitation of our system is not considering the cue conflicts and the effects of animation.

One future direction is to perform more comprehensive experiments such as testing other tasks and comparing with different selection methods. Moreover, the rule base should be extended and more rendering methods should be implemented. Another idea is to compare the behaviour of our system for different multi-view technologies.

Acknowledgements

The authors are supported by the EC FP7-213349 All 3D Imaging Phone project and TUBITAK. Also, we would like to thank all the

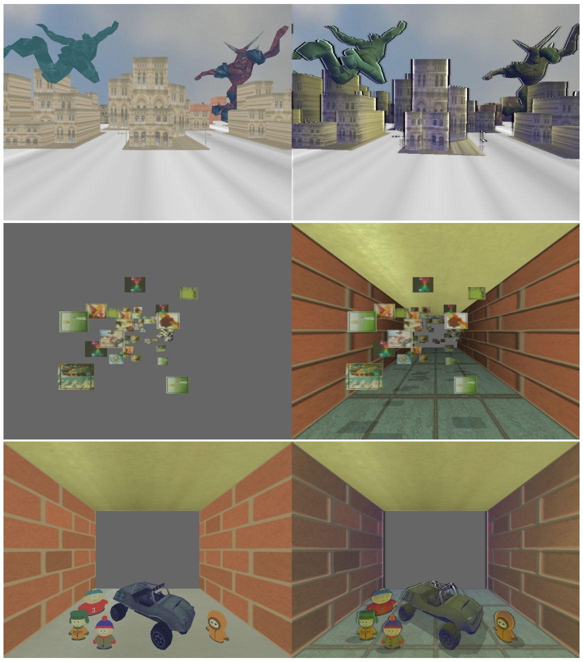


Figure 10: Left: Original scene. Right: The scene with enhanced depth perception.

participants of the experiments in this work.

References

Akenine-Moller, T., Haines, E., and Hoffman, N. 2008. *Real-Time Rendering*, third ed. A. K. Peters, ch. 6.

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. *The Visual Computer 26*, 5, 311–323.

Bulbul, A., Cipiloglu, Z., and Capin, T. 2010. A perceptual approach for stereoscopic rendering optimization. *Computers & Graphics 34*, 2, 145 – 157.

Bulthoff, H. H., and Yuille, A. 1996. *A Bayesian Framework for the Integration of Visual Modules*.

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

Cutting, J. E., and Vishton, P. M. 1995. *Perception of Space and Motion. Handbook of Perception and Cognition*, second ed. Academic Press, 69–117.

Dodgson, N. A. 2005. Autostereoscopic 3d displays. *Computer 38*, 31–36.

1997. Fuzzy control programming. Tech. rep., International Electrotechnical Commision.

Gooch, A., Gooch, B., Shirley, P., and Cohen, E. 1998. A non-photorealistic lighting model for automatic technical illustration. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 447–452.

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

Howard, I., and Rogers, B. 2008. *Seeing in Depth*. Oxford University Press.

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 *SPIE Visual Commmunications and Image Processing IV*, vol. 1199, 1154–1163.

Markosian, L., Kowalski, M. A., Goldstein, D., Trychin, S. J., Hughes, J. F., and Bourdev, L. D. 1997. Real-time nonphotorealistic rendering. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 415–420.

Nienhaus, M., and Doellner, J. 2003. Edge-enhancement - an algorithm for real-time non-photorealistic rendering. *Journal of WSCG*, 346–353.

Oruc, I., Maloney, T., and Landy, M. 2003. Weighted linear cue combination with possibly correlated error. *Vision Research 43*, 2451–2468.

Rheingans, P., and Ebert, D. 2001. Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Transactions on Visualization and Computer Graphics 7*, 253–264.

Russell, J. 1997. How shall an emotion be called. *Circumplex models of personality and emotions*, 205–220.

Schrater, P. R., and Kersten, D. 2000. How optimal depth cue integration depends on the task. *International Journal of Computer Vision 40*, 73–91.

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

Tarini, M., Cignoni, P., and Montani, C. 2006. Ambient occlusion and edge cueing for enhancing real time molecular visualization. *Visualization and Computer Graphics, IEEE Transactions on 12*, 5 (Sept.-Oct.), 1237–1244.

Wanger, L. R., Ferwerda, J. A., and Greenberg, D. A. 1992. Perceiving spatial relationships in computer-generated images. *IEEE Computer Graphics and Applications 12*, 44–58.

Wanger, L. 1992. The effect of shadow quality on the perception of spatial relationships in computer generated imagery. In *SI3D '92: Proceedings of the 1992 symposium on Interactive 3D graphics*, ACM, New York, NY, USA, 39–42.

Ware, C., and Mitchell, P. 2008. Visualizing graphs in three dimensions. *ACM Trans. Appl. Percept. 5*, 1, 1–15.

Ware, C. 2004. *Information Visualization: Perception for Design*. Elsevier, ch. 8.

Weiskopf, D., and Ertl, T. 2002. A depth-cueing scheme based on linear transformations in tristimulus space. Tech. rep., Visualization and Interactive Systems Group University of Stuttgart.