VoxAR: Adaptive Visualization of Volume Rendered Objects in Optical See-Through Augmented Reality

Saeed Boorboor, Matthew S. Castellana, Yoonsang Kim, Zhutian Chen, Johanna Beyer, Hanspeter Pfister, *Fellow, IEEE*, and Arie E. Kaufman, *Fellow, IEEE*

***Abstract*—We present VoxAR, a method that facilitates an effective visualization of volume-rendered objects in optical see-through head-mounted displays (OST-HMDs). The potential of augmented reality (AR) to integrate digital information into the physical world provides new opportunities for visualizing and interpreting scientific data. However, a limitation of OST-HMD technology is that due to its holographic nature, the rendered pixels of a virtual object interfere with the physical world, making it challenging to perceive the augmented virtual information accurately. In this work, we address this challenge by presenting a two-step approach. First, VoxAR determines an appropriate placement of the volume-rendered object in the real-world scene by evaluating a set of spatial and environmental objectives, managed as a set of user-selected preferences and pre-defined constraints. We achieve a real-time solution by implementing the objectives using a GPU shader language. Next, VoxAR adapts the colors of the volume-rendered object based on the real-world placement region. To this end, we introduce a novel method that optimally adjusts the input transfer function colors such that the resulting volume rendering has pixel colors discernible with respect to the background while simultaneously maintaining the perceptual relationship between the modified colors and its mapping to data intensity values. Finally, we present an assessment of our approach through objective evaluations and subjective user studies.**

***Index Terms*—Adaptive Visualization, Situated Visualization, Augmented Reality, Volume Rendering.**

I. Introduction

THE transformative ability of augmented reality (AR) to fuse the digital world of bits with the physical world of atoms has provided new opportunities to visualize 3D spatial scientific data. Over the decades, there have been significant advances in methods and techniques for virtually presenting information to users. However, in contrast to virtual reality (VR) and mixed reality (MR) technologies, optical see-through (OST) AR has only been sparsely adopted for scientific visualization [1]. This can be attributed to a fundamental challenge in OST-AR design: augmenting virtual content onto the user's optical field-of-view (FoV) blends the projected virtual object with the physical environment. As such, it is possible to inaccurately perceive the rendered pixels of the virtual content, which is a critical limitation in scientific visualization. The full potential of AR is realized when virtual data is effectively visualized with respect to its referent in the physical space. Towards this goal, methods have been developed for the intuitive placement [2], [3], visibility enhancement [4]–[7], and color correction [8], [9] of virtual objects rendered in OST head-mounted displays (HMDs), albeit as separate objectives [10].

A fundamental utility of volume rendering visualization is exploring and interpreting volume data. Typically, this is achieved using transfer functions (TFs) that map the intrinsic values of the data to a spectrum of optical properties, such as color and opacity. In terms of visual perception, most existing AR techniques are not adequately designed to address the challenges of visualizing volume-rendered objects. This motivated us to design VoxAR – a method for augmenting volume-rendered objects in OST-HMDs that adapts to real-world surroundings and adjusts TF colors to effectively visualize scientific data. Specifically, VoxAR adopts a two-fold approach: first, it determines an optimal position for displaying a virtual object in the user's FoV, based on user-defined preferences, and second, it adjusts the colors of the TF to distinguish the rendered volume from the background.

The placement of a virtual object in the scene can significantly impact data understanding and decision-making [11]. Unlike a controlled desktop setting, the real-world serves as the virtual canvas for AR visualizations. Due to its dynamic nature, spatial locations of virtual objects cannot be pre-calculated, and must be determined in-situ. Existing frameworks and toolkits for MR [3], [12] adaptively place virtual objects in the scene by solving a set of rules and user-defined semantic preferences, such as distance between the virtual object and its physical referent, its position from the center of the FoV, and surface magnetism. To improve the visual perception of data in OST-AR, we additionally evaluate environmental factors from the scene, such as light sources, that flush the displayed projection from the HMD, and the perceptual color difference between the real-world backdrop and TF colors. VoxAR is designed using GPU shaders that evaluate all candidate 3D spatial locations in the FoV in parallel and solves an optimal location that best satisfies a composite set of user-defined semantic preferences, minimizing environmental limitations.

Following placement, the colors of the input TF may require further adjustment so that the pixel colors of the volume-rendered object are discernible with respect to the background region. Existing solutions to alleviate the blending of virtual objects with the physical world can be broadly categorized into contrast enhancement [4], [5], [9] and re-colorization [13] techniques. However, unlike the nature of the virtual objects

Boorboor, Castellana, Kim, and Kaufman are with the Department of Computer Science, Stony Brook University. Boorboor and Castellana have equal contributions. Chen, Beyer, and Pfister are with John A. Paulson School of Engineering and Applied Sciences, Harvard University.

Manuscript received April 19, 2021; revised August 16, 2021.

addressed in most existing techniques, where colors broadly indicate the presence of an attribute, the visual output from volume rendering enables reasoning and understanding about the data attributes through a perceptual mapping of defined TF colors to those of the rendered output. In VoxAR, we introduce a novel approach to search for an optimal shift in the original TF that enhances the color perception of the resultant volume-rendered object against the real-world background while maintaining the visual characteristics for mapping the data attributes. Rather than post-processing, we perform the TF function enhancement prior to the volume rendering pipeline. Specifically, we have designed an objective function that shifts the input TF in the CIE $L^*a^*b^*$ (CIELAB) space such that it satisfies a set of constraints designed to (1) maximize the visual color difference between the TF spectrum and the background colors, (2) maintain color properties similar to the input TF, and (3) ensure a valid $L^*a^*b^*$ to RGB-space transformation. Given the 3D search space, we use CMA-ES [14] as a solver to find the TF color intervals with minimum objective cost.

VoxAR is developed as an end-to-end system in Unity3D game engine for Microsoft Hololens2 OST-HMD. We demonstrate our results by using examples of volume datasets with TF presets from widely used volume rendering applications. To evaluate the effectiveness of our approach, we conduct user studies and show that VoxAR significantly enhances a user's ability to perceive and analyze volume-rendering visualizations in AR.

In summary, We define our contributions as follows:

- an end-to-end OST-AR framework for the placement and adaptive visualization of volume-rendered objects,
- a real-time implementation for solving an optimal placement based on user preferences while minimizing color overlap with the real-world surroundings,
- a novel method to adjust the TF color based on background colors while preserving the perceptual mapping between volume data attributes and the input TF.

II. Related Work

In Sec. II-A, we review existing works on adaptive virtual object placement and systems that have formalized the goal as a constraint system, and in Sec. II-B we discuss techniques that improve color perception in AR, and specifically OST-AR by either performing contrast enhancement or re-colorization of the virtual object. Exploring these related works, we observe that, to the best of our knowledge, (1) placement and color constraints have not been effectively explored as a coupled problem, and (2) these works do not satisfactorily address the challenges of volume rendering visualization.

A. Visualization and Object Placement

In context-aware mixed reality, the virtual experience dynamically adapts to context-specific information. The information considered as context can vary significantly depending on the desired goals of the application. For example, contextual information can be derived from depth information: Google's DepthLab [15] utilizes depth information to create believable interactions with the environment. AR objects can be placed behind or physically interact with real-world ones. Microsoft's FLARE [16] analyzes a scene to generate location-specific AR layouts based on detected geometry and surfaces. Position or location-based data can also provide context. Systems for situated visualizations, such as SemanticAdapt [17] and RagRug [18] modify virtual objects based on semantic associations with real-world physical objects. VoxAR utilizes a context-aware approach, generating scene-specific candidate locations for placement based on color and depth information, and by evaluating spatial and environmental objectives managed as a set of user-selected preferences and pre-defined constraints.

Many mainstream AR authoring systems provide the ability to specify objectives for AR objects to determine their behavior automatically. Microsoft's Mixed Reality Toolkit [19] includes a number of solvers that compute the position and orientation of AR objects based on which kinds of solvers are attached (e.g., surface magnetism, constant view size, etc.) and a predetermined algorithm for how to evaluate them. Unity MARS [2] uses Reasoning APIs to collect information about the scene and extract it into a higher-level database, where database objects have special semantic information stored as *traits*. Users can construct and attach placement-related goals called *conditions* to AR objects. These conditions evaluate real-world placement locations in the database based on how well the corresponding traits satisfy the conditions. Evangelista et al. have developed Adaptive User Interfaces Toolkit (AUIT) [3] that allows users to create and attach goals to objects to determine their position and orientation. Instead of providing a database, it allows users to customize all aspects of the placement process. Users can hook into existing abstracted data sources, write their own objectives for AR objects, determine what kinds of solvers to use and when to trigger them, and how to transition between changing states. Each of these tools can handle goals relating to transform-related properties of AR objects. In this work, we extend the scope of the placement objectives to address the visual properties of the object or the scene. For instance, in our case, the transfer function used to render the virtual object and factors such as avoiding occlusion, textured backgrounds, and light sources. Our approach provides a novel technique for incorporating such image-based objectives alongside spatial objectives, taking advantage of specialized shaders for increased efficiency.

B. Visualization Color Enhancement

Solutions to alleviate color-blending can be divided into hardware [20], [21] and software-based methods. For the scope of this paper, we describe software-based techniques here. Existing works that address color blending in OST-HMDs [22] can be broadly categorized as color correction and visibility improvement solutions. Color correction involves sensing background colors and subsequently subtracting them from the colors of the virtual content [8]. However, such compensation typically results in a decrease in brightness. For improved color contrast, Hincapi et al. [9] have developed

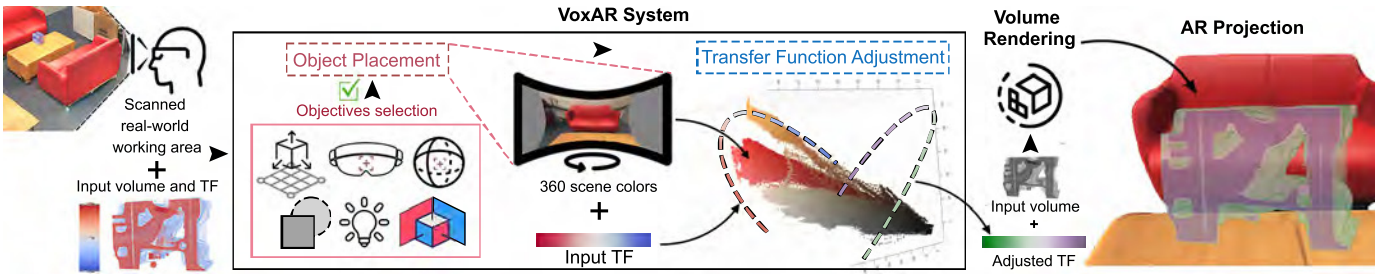

Fig. 1. An illustration of the VoxAR pipeline. For a volume to be rendered using an input TF function, in a user's real-world FoV, VoxAR first determines an optimal placement. This is evaluated based on an objectives selection set, which the users opt for from our formulated list. The placement scene is then used to adjust the input TF such that the resultant volume rendering facilitates an effective visualizing experience.

SmartColor, a real-time algorithm that performs background subtraction in the CIELAB space using GPU shaders. The work is primarily for text visibility, and a real-time solution is achieved by discretizing the color space, which does not adequately capture the full TF spectrum. For more of an adaptive approach, Fukiage et al. [7] have introduced a framework for measuring the visibility metric needed to predict the visibility of semi-transparent virtual objects against any background. A major limitation of this work regarding TF colors is that their approach only analyzes luminance and does not consider color-opponent channels. This is more so because their algorithm is focused on solving for 50% transparency, while in our case, a volume-rendered object can have varying opacity values. A recent work closest to our goals is that of Zhang et al. [4]. In this work, the authors present a constraint-based system that aims to preserve the contrast between virtual objects and the background and maintain consistency with the original displayed color. However, their approach is threshold-intensive. That is to say, for effective results, the right hue thresholds need to be adjusted, especially for varying backgrounds and TF colors.

While there have been works to improve the quality of volume rendering in video see-through AR [23], [24], to the best of our knowledge, no work has sufficiently addressed the challenge of color correction and visibility improvement for volume rendering visualization in OST-AR. Most approaches address the blending challenge as a post-processing problem, whereas we have designed VoxAR to solve the TF color optimization as a step prior to the rendering process. Moreover, for techniques that solve color enhancement in the CIELAB color space, we noticed that most works assume a valid and correct projection of their solution in the CIELAB space to RGB, whereas VoxAR ensures this validity and perceptually meaningful conversion as part of its constraint. Conclusively, in reviewing works for AR, VoxAR is a novel approach that combines spatial placement and color adjustment as complementary solutions for achieving an effective immersive visualization experience.

III. VoxAR Design and Workflow

We introduce a two-step method that precedes the volume rendering pipeline and the augmentation of its result in the real-world. Initially, by evaluating spatial and environmental objectives, managed as user-selected preferences and pre-defined constraints, VoxAR finds an appropriate position for placing the virtual object in the real-world (Sec. IV). Next, it adjusts the input TF colors such that the pixels of the resultant volume-rendered object are discernable against the background. Importantly, this adjustment attempts to best preserve the perceptual mapping between the data attributes and the colors assigned in the input TF (Sec. V). An overview of the VoxAR pipeline is illustrated in Fig. 1.

The above-mentioned method is limited to a static FoV and scene. However, one key utility of AR applications lies in their ability to interact with virtual objects while navigating within the real-world environment. Constantly adapting positions and colors during data visualization and analysis can risk introducing inconsistencies in data perception. Thus, to support an AR system with changing FoV, following initial placement, VoxAR performs the TF optimization based on the background colors surrounding the provided position. Subsequently, it continues to evaluate the placement objective score with an additional constraint of maintaining color discernibility of the adjusted TF atop the updated background. If the objective score falls below a threshold, an alternative optimal position is suggested to the user. To this end, for initialization, the user is first required to scan a working area. This allows the system to generate a 3D scene model and evaluate its semantics.

IV. VoxAR Visualization Object Placement

For a given scene instance, VoxAR solves the placement of the virtual object in the real-world by (1) determining sets of candidate locations, (2) analyzing the candidate locations with regard to the user-selected objectives, and finally, (3) placing the object at the location which *best* satisfies the objectives.

A. Determining Candidate Locations for Placement

The search space for placing a virtual object in the physical space can be reduced to a finite subset of visually distinguishable, semantically meaningful, and environmentally favorable locations in 3D space, which can then be evaluated based on user preferences and constraints. VoxAR uses two distinct categories of groups of candidate locations: *surface magnetism* [25], [26] and *discretized 3D*. On top of this, each object must be defined with a location and orientation. For placement in discretized 3D, we evaluate regular, discrete locations within a bounding volume of the working area, using a fixed orientation. However, for surface magnetism,

the orientation of the object is heavily dependent on varying surface normals.

To achieve a real-time system, we introduce *placement maps*, a texture-based data structure that stores real-world spatial information for each *valid* pixel in the scene. A placement map comprises three textures with the following attributes:

- **Validity map:** a binary score to indicate whether the pixel needs to be evaluated.
- **Position map:** if the pixel is valid, the corresponding 3D position of the pixel, mapped to the texture (r, g, b) tuple.
- **Rotation map:** if the pixel is valid, the quaternion of the pixel normal, mapped to the texture (r, g, b, a) tuple.

Representing this information as shader textures allows multiple positions for multiple objectives to be evaluated in parallel on a GPU.

Using the inverse projection matrix of the AR HMD camera I, the placement maps for surface magnetism and discretized 3D are generated as follows. For surface magnetism, a single placement map PM_{surf} consists of:

- position map $POS_{surf}[x, y] = I * (x, y, depth(x, y))$
- rotation map $ROT_{surf}[x, y] = orient(normal(x, y))$
- validity map $VM_{surf}[x, y] = is\_surface(x, y)$

where *depth* and *normal* are externally calculated depth and normal information of a given pixel using the initially scanned 3D model of the scene, *orient* returns a quaternion-based orientation determined from a provided normal, and *is_surface* is a binary boolean based on externally calculated surface detection information.

In contrast, discretized 3D creates multiple placement maps from a user-specified bounding volume derived from the camera frustum. Given a discrete set of camera-space values along the camera positive Z axis $\{z \mid z = \text{near}+k\cdot\text{interval}, k \in \mathbb{Z}, \text{near} \leq z \leq \text{far}\}$, where near, far, and interval can be adjusted, and single quaternion Q representing an orientation in 3D space, a set of placement maps $PM_{3D} = \{PM_{z_i} \mid \forall z_i \in z, \exists PM_{z_i}\}$ is generated where:

- position map $POS_{z_i}[x, y] = I * (x, y, z_i)$
- rotation map $ROT_{z_i}[x, y] = (Q.x, Q.y, Q.z, Q.w)$
- validity map $VM_{z_i}[x, y] = 1$

In effect, we create a discrete 3D grid of points via placement maps, with width and height resolution corresponding to the device FoV, and an adjustable depth resolution based on $near$, far, and $interval$. Higher depth resolutions consider more areas in 3D space, at the expense of overall system performance.Although the discretized 3D placement maps come from the same 3D volume, each placement map is evaluated by objectives independently of all others.

B. Defining the placement objectives

We have identified and implemented a set of objectives, discussed later in this section, based on our review of adaptive placement-related objectives [2], [3], [19], [25]. The choice of objectives was determined by how well they contribute to the goal of AR volume visualization, with a primary focus on volume visibility and a secondary focus on semantically meaningful placement. We have additionally designed a new color objective for the perceptual contrast between the TF and real-world background colors.

Below, we describe the VoxAR placement objectives and formulate the scores for each objective. Unless otherwise specified, the output of the objective is a score $\in [1, 0]$, using the following equation:

$$Score(x, y, \lambda) = \begin{cases} 1, & \text{if } \lambda \text{ and } valid \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

O1 Surface magnetism allows users to either associate a virtual object against a horizontal or vertical surface, or set it to be placed anywhere in the 3D space. Given the set of placement map types S, where $S = \{3D, surface\}$, the score for each pixel is determined using Eq. 1 where λ is true if the type of the corresponding placement map has been expressed as a user preference.

O2 Point proximity allows users to tether a virtual object to a 3D object or position in the scene, with radius r_O. For a specified 3D point in the real-world, (x_p, y_p, z_p), to place the virtual object in its proximity at a maximum distance d_p, the score is calculated from Eq. 1 where $\lambda = \|(x_p, y_p, z_p) - \text{pixel2world}(x, y)\| \leq d_p$. The $\text{pixel2world}(x, y)$ provides a 3D position using the placement map's position map.

O3 Center of screen projection allows users to prefer viewing the virtual object projected toward the center of their FoV. For a device screen space center, (x_s, y_s), and a scaling factor, s, the score map for this constraint is calculated using:

$$Score(x, y, s) = e^{-\alpha}, \quad \alpha = \frac{\|(x, y) - (x_s, y_s)\|}{s} \quad (2)$$

O4 Color discernibility is our novel objective that maximizes the perceptual difference between TF colors and the colors in the real-world scene. Given an input TF, we first generate a set T by uniformly sampling the colors along its spectrum and projecting them in the CIELAB color space. Subsequently, for the corresponding color of the placement map pixel, also projected in the CIELAB space, p_{Lab}, we first find the point t_{min} in T with minimum Euclidean distance from p_{Lab}. Then, the objective score for pixel (x, y) is calculated using the CIELAB ΔE_{00}, which measures the perceptual color difference between two colors, in this case, t_{min} and p_{Lab}. After this per-pixel metric has been calculated per-pixel, the result is averaged using the object-sized 2D kernel described previously.

O5 Visibility ensures the virtual object does not collide or get occluded by scene objects. For the 3D point of a pixel location, $pixel2world(x, y)$, its score is determined by utilizing our object-sized 2D kernel to place the bounding box at the point and performing and, through uniformly sampled pixels, performing oriented ray-cube intersection tests. With regards to Eq. 1, λ is defined as the condition where the real-world depth value derived from the camera

depth texture is determined to be outside the oriented bounding box based on the intersection test results.

O6 **Environmental** avoids the placement of the virtual object against challenging environmental conditions (e.g., light sources, which can cause flushing of the OST-HMD projection). For a detected light source visible in the scene, all of its pixels are set to 0.

Users can define any combination of objectives. For each objective, a weight and a constraint level must be specified. The weight assigns relative importance to the objective score, whereas the constraint is a binary class that categorizes the objective as *requirement* or *preference*. As such, a *required* objective must always be met. In other words, if the placement map has a validity value of 0, that pixel will not be considered for placement, regardless of other objectives. Moreover, we identify objectives **O5** and **O6** as hard constraints since they impact the effectiveness of the visualization and, therefore, should always be considered with minimum pre-defined weights.

Each candidate location in each placement map is evaluated for each objective in terms of how well placing the volume there would satisfy the objective. This produces a per-pixel score based on each objective. The scores are stored in separate *score map* textures. The storage format of placement maps and score maps facilitates an easy visual understanding of how different stages of the placement component interact.

Some objectives, such as **O2**, evaluate the placement map on a per-pixel basis. Other image-related objectives, such as **O4**, require more complex evaluations; since rendered volumes generally take up multiple pixels onscreen, a 2D screen-space bounding box around the projected volume should be evaluated. To accomplish this, we utilize custom shaders that, for each pixel being evaluated, calculate the size of the bounding box of the AR object, centered at the pixel and oriented to the corresponding rotation, to define a 2D kernel. As a result, the final objective score for the center pixel is determined by evaluating and averaging the scores of all pixels that fall within the kernel.

C. Final placement

All generated score maps are grouped together according to the corresponding placement map they were generated from, regardless of the objective that generated them. Considered collectively, each score map group thus has a complete evaluation of the individual placement map with respect to all objectives. To determine which pixel location is locally optimal within each score map group, the score maps *SM* are aggregated according to the following equation:

$$SM_{agg}[x,y] = \begin{cases} 0 & \text{if } \exists SM_i((SM_i[x,y]=0) \wedge H_i) \\ \sum_{i=1}^{n} SM_i[x,y] * w_i & \text{otherwise} \end{cases} \tag{3}$$

where SM_{agg} is the aggregated score map, (x, y) is the current pixel location under consideration, SM_i is an objective-specific score map, w_i is the corresponding objective weight, and H_i is 1 if the objective for SM_i is classified as *hard* and 0 otherwise. This eliminates any pixel that does not satisfy the hard constraint; the values of all remaining pixels are based on a weighted sum of the scores at corresponding pixel locations in the original score maps.

Once this aggregation has been completed, the locally optimal pixel location score of every aggregated score map is determined and compared. The aggregated score map with the globally optimal pixel location has the corresponding location in its placement map queried for its position and rotation, which are used to place the AR object.

D. Re-evaluating changing FoV

A single placement evaluation is insufficient for high-quality results with changing Fov and scenes throughout an AR session. Continuous re-evaluation is needed to ensure that the user-selected objectives are consistently met.

To achieve this, after initial placement, VoxAR re-examines the existing placement every second by calculating an aggregate objective score based on the location where the object appears in the current FoV. In the case where the object is outside the updated FoV, objectives **O1**, **O4**, **O5**, and **O6** maintain their most recent scores, **O3** returns 0, and **O2** is re-calculated.

To avoid spontaneous repositioning of the object, VoxAR determines a new optimal position once the current position objective score drops below a defined threshold. It is important to note here that to maintain the perceptual mapping between the TF and the volume data attributes, VoxAR does not re-evaluate the TF colors (explained in Sec. V). Thus, during the re-evaluation, **O4**, color discernibility, becomes a *requirement* constraint.

When a new position is evaluated, the user receives a visual indication in the form of a bounding box as a suggestion. If the user accepts, the AR object will be moved to the new location and, subsequently, re-evaluated. Contrarily, the suggestion may move if the location is no longer good or disappear if the prior location score significantly improves.

V. VoxAR Transfer Function Adjustment

In the direct volume rendering model, TFs emphasize or classify features of interest within data by mapping intrinsic data attributes to optical properties, such as color and opacity [27]. For the scope of this work, we consider the case of 1D TFs that map scalar data intensity values to color and opacity. Typically, 1D TFs are defined using a set of control points with assigned color and opacity values, which are then linearly interpolated to construct a complete spectrum (Fig. 2(a)). As a result, given a data value, s, $\text{TF}(s) \mapsto (\mathbf{C^3}, \alpha)$, where $\mathbf{C^3} \in (R, G, B)$ and α is opacity.

When deployed in an OST-HMD, the visual similarity between colors in the real-world background and any color along the interpolated TF spectrum can interfere with the optically semi-transparent rendering of the virtual object. To avoid this, we formulate a constrained optimization for TF adjustment that aims to ensure: (1) the colors in the adjusted TF spectrum are perceptually distinguishable from the background, and (2) the color control points of the adjusted TF reflect visual characteristics similar to those of the input TF.

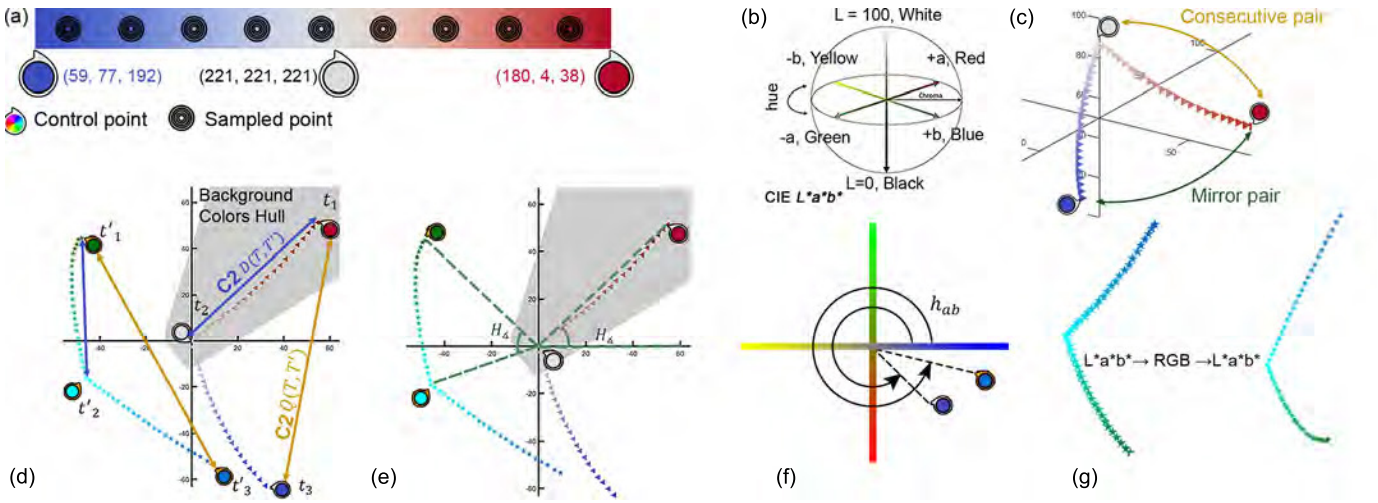

Fig. 2. (a) illustrates the linear interpolation of colors in a TF from its control points. (b) is a diagram of the CIELAB colorspace. (c) demonstrates the mapping of the TF in (a) from RGB to CIELAB. The following illustrate the TF adjustment constraint calculations: (d) the intersection of the adjusted control points with the background region (**C1**), and the ΔE calculation between consecutive pairs and mirror pairs (**C2**), (b) hue separation (**C2**), (c) hue measurement (**C3**), and (d) valid projections between color spaces (**C4**).

A. *Defining the solution space*

To obtain a visually meaningful solution, we solve our formulated optimization in the CIELAB color space [28]. CIELAB is a device-agnostic 3D space modeled to represent colors as perceived by the human eye. Specifically, CIELAB expresses colors as a measure of perceptual lightness, L^*, *red*-*ness*-to-*greenness*, a^*, and *blueness*-to-*yellowness*, b^*. Based on this representation, the distance between 3D color values corresponds approximately to the change humans see between colors. This is quantified using ΔE_{00} [29]:

$$\Delta E_{00} = \sqrt{\left(\frac{\Delta L}{S_L}\right)^2 + \left(\frac{\Delta C}{S_C}\right)^2 + \left(\frac{\Delta H}{S_H}\right)^2 + R_T \frac{\Delta C}{S_C}\frac{\Delta H}{S_H}} \tag{4}$$

where ΔL, ΔC, and ΔH are the differences in the lightness, chroma, and hue of two colors in the CIELAB space, and S_L, S_C, and S_H are the scaling factors that correct for differences in the range and slope of the L, C, and H scales. R_T is a term that accounts for the interaction between chroma and hue differences. For simplicity, this paper will refer to ΔE_{00} as ΔE.

Moreover, to balance the effect of the size of visual objects on color appearance [30], Stone et al. [31] have developed a model that provides a minimum scaling factor for the $L^*a^*b^*$ axes that enables an effective discernibility of colors more than 50% of the time. In our implementation, we consider a visual angle of $1/3°$ and scale the function's interval by 3, as suggested by Gramazio et al. [32].

B. *Optimization constraints*

We now describe the constraints we have formulated for the TF adjustment optimization problem. The constraints determine a penalty cost for each adjusted TF candidate, which the solver (described later in this section) attempts to minimize to obtain a final result. As an initial step, RGB values from the pixels of the real-world scene and the TF color control points are projected into the CIELAB space. Using the projected points, each candidate TF is evaluated as discussed below. Fig. 2(d)-(g) illustrates the computation of the constraints.

C1 Background discriminability. The primary goal for adjusting the TF is to minimize color interference with the real-world background. Therefore, for the set of colors in the adjusted TF candidate, T', and the background, B, we define this constraint, I, as:

$$I(T,B) = \begin{cases} kE_D, & \text{if } \Delta E(T',B) \leq d \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where E_D is the constraint penalty score and k is a weighted factor, explained below. We choose $d = 11.5$ for the ΔE bound as a scaled *just-noticeable difference* (JND) measure to cater to the possibility of low opacity TF mapping in addition to the semi-transparent projection. Existing works [33], [34] have quantified an empirical benchmark for the minimum perceptual color difference as 2.3, commonly termed JND. Based on our initial experiments and pilot user studies, we empirically determined a scaled factor of JND $\times 5$.

To support changing FoV around the initial placement, a histogram of all colors in the working region is computed. This is achieved by pivoting a virtual 360°omnidirectional camera at the placement position and generating a panoramic scene texture. Naturally, there are many unique colors in the entire scene, and it can become challenging to find visually non-intersecting TF colors. To effectively reduce the number of background colors, we convert the panorama into superpixels of 10% of the unique colors in the scene texture. Moreover, we noticed that background pixels that have a low representation of color distribution in the scene or are physically located far from the object position make it difficult for the solver to converge to a solution. Thus, we represent $b \in B$ as a function of its L*a*b representation, superpixel size, and distance from the placement position. That is to say, b is represented as $(L^*, b^*, a^*, \sum^n (s_n \times d_n))$, where n is the color frequency, s is

the superpixel size, and d is the euclidean distance between the physical position corresponding to the center of the superpixel and the object placement position.

To reduce the performance overhead of comparing all the background and adjusted TF points, we define a convex hull bounding the background points. Therefore, k in Eq. 5 is the normalized frequency value $\times$ of normalized distance value of the vertex closest to the TF point intersecting with the hull.

C2 Perceptual characteristics of the TF control points. To preserve the visual characteristics of the input TF (T), we have identified three attributes to compare in candidate TFs: (1) perceptual color difference, (2) hue separation, and (3) $L^*a^*b^*$ congruence. We formulate this constraint, P, as:

$$P(T,T') = w_d D(T,T') + w_a A(T,T') + w_q Q(T,T') \quad (6)$$

with w_d, w_a, and w_q as weights for each attribute constraint.

The first term, $D(T,T')$, measures the perceptual color difference between control point pairs:

$$D(T,T') = \sum_{i=1}^{n-1} \mid \Delta E(t_i, t_{i+1}) - \Delta E(t'_i, t'_{i+1}) \mid \quad (7)$$

where n is the number of control points in the TFs, $t \in T$, and $t' \in T'$.

The second term, $A(T,T')$ maintains a measure of hue separation between the control points by comparing the angles between consecutive pairs on the a^*b^* plane:

$$\begin{aligned} A(T,T') &= \sum_{i=1}^{n-1} \mid H_{\measuredangle}(t_i, t_{i+1}) - H_{\measuredangle}(t'_i, t'_{i+1}) \mid \\ H_{\measuredangle}(c_1, c_2) &= \cos^{-1}\left(\frac{c_1 \cdot c_2}{\|c1\|\|c2\|}\right) \end{aligned} \quad (8)$$

Finally, we noticed that due to consecutive pair-wise comparisons, in some instances, the solver would optimize the cost by interleaving the shape of the TF curve in such a way that it would satisfy the constraints. Therefore, to preserve the TF global curvature, we additionally check for its congruence by performing mirror comparisons of the control points. We define the third term Q, that checks for congruence, as:

$$\begin{aligned} Q(T,T') = &\sum_{i=1}^{n/2} \mid \Delta E(t_i, t_{n-i+1}) - \Delta E(t'_i, t'_{n-i+1}) \mid \\ &+ \sum_{i=1}^{n/2} \mid H_{\measuredangle}(t_i, t_{n-i+1}) - H_{\measuredangle}(t'_i, t'_{n-i+1}) \mid \end{aligned} \quad (9)$$

C3 Similarity to original color tone. For some applications, it may be important that the adjusted TF retains the *color tone* or *hueness* of the input TF. Therefore, we formulate this constraint, S, to adjust to a user-defined weight, w_s, as:

$$\begin{aligned} S(T,T') &= \sum_{i}^{n} s(t_n, t'_n), \\ s(c,c') &= \begin{cases} e^{\Delta h_{ab}(c', \lambda_h)}, & \text{if } \Delta h_{ab}(c', \lambda_h) \geq \lambda_h \\ 0, & \text{otherwise} \end{cases} \\ h_{ab}(c) &= \arctan\left(\frac{c_{b^*}}{c_{a^*}}\right) \end{aligned} \quad (10)$$

where $\lambda_h = w_s h_{ab}(t_n)$, h_{ab} is the hueness measured in the a^*b^* space, and Δh_{ab} is the absolute difference between two colors.

C4 CIELAB to RGB projection. Due to the difference in 3D gamut sizes, not all CIELAB values have a valid RGB projection. Moreover, not considering gamma correction, projecting the optimization solution from a continuous CIELAB space to a discrete RGB space, may lose perceptual color differentiation on the device. Therefore, to constrain the solution to have a valid and equally effective adjusted TF in the RGB space, we define V as:

$$V(T'_{rgb}) = w_{pr} \sum_{i=1}^{n} Pr(t'_{rgb,\,i}) + w_{jnd} \sum_{i=1}^{n-1} J\left(t'_{rgb,\,i}, t'_{rgb,\,i+1}\right) \quad (11)$$

$$Pr(c) = \begin{cases} 0, & \text{if } (0,0,0) \leq (r,g,b) \leq (1,1,1) \\ 1, & otherwise \end{cases} \quad (12)$$

$$\begin{aligned} J(c_1, c_2) &= \sum_{i=1}^{k-1} f(\text{lerp}(c_1, c_2, i), \text{lerp}(c_1, c_2, i+1)) \\ f(a,b) &= \begin{cases} 1, & \Delta E(\text{lab2rgb}(a), \text{lab2rgb}(b)) \leq k \text{ JND} \\ 0, & otherwise \end{cases} \end{aligned} \quad (13)$$

where T'_{rgb} is the set of control points from the candidate T' projected in the RGB color space. Since TFs are a continuous interpolation of control points, we formulate Eq. 13 to uniformly sample RGB values in T'_{rgb}, using a sampling frequency k, and maintain a reasonable JND value along the spectrum when reprojected back to the CIELAB space.

C. Solving the TF adjustment final objective

Using the constraints formulated above, VoxAR solves for an adjusted TF, TF_{adj} by minimizing the following objective:

$$\begin{aligned} \left[\hat{\mathbf{R}}, \hat{\mathbf{T}}\right] = \text{argmin}_{\mathbf{R},\mathbf{T}} [&I(T', B) + P(T, T') \\ &+ V(T'_{rgb}) + S(T, T')] \end{aligned} \quad (14)$$

$$T' = \left[\mathbf{RT}\right] T \quad , \quad T = \left[t_1, t_2, ..., t_n\right]^\top$$

$$\text{TF}_{adj} = \left[\hat{\mathbf{R}}, \hat{\mathbf{T}}\right] T \quad (15)$$

where $\mathbf{R} \in (\theta_{L^*}, \theta_{a^*}, \theta_{b^*})$ and $\mathbf{T} \in (L^*, a^*, b^*)$ are rotation and translation matrices, respectively. Essentially, we solve for a rotation and translation that would optimally transform the input TF control points in the CIELAB space, such that the adjusted TF would satisfy the formulated constraints. Given the large search space and possible solutions, we use Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [14] as the solver for our objective function. CMA-ES is an evolutionary algorithm commonly used for global optimization of non-linear functions. It is particularly effective in high-dimensional search spaces and can handle noisy and non-convex optimization problems.

At each iteration, CMA-ES generates a population of candidate $\mathbf{R}$ and $\mathbf{T}$ according to a multivariate normal distribution

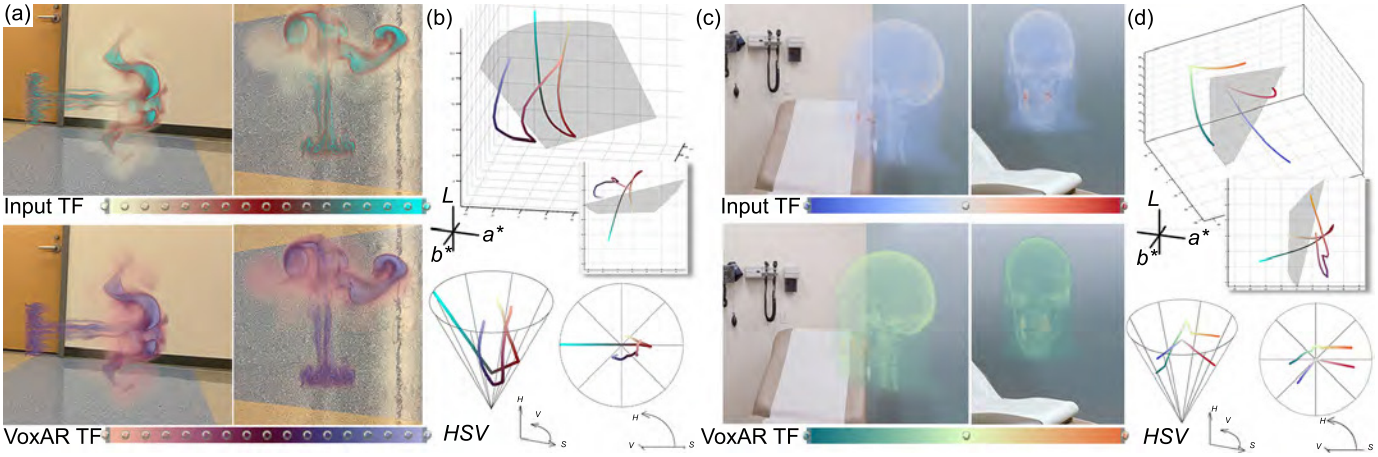

Fig. 3. VoxAR TF adjustment results. (a) and (c) show FoVs of the Jet volume rendered using Haze-Cyan input TF (*top*) and our adjust VoxAR TF *(Bottom)*, and the Skull volume rendered using the Diverging input TF (*top*) and our adjust VoxAR TF *(Bottom)*, respectively. (b) and (d) show plots of the input and VoxAR TF in the CIELAB (*top*) and HSV (*bottom*) color space.

with a mean vector and covariance matrix updated based on the history of successful candidate solutions. Specifically, the candidates from a population are used to transform the input TF, which we have referred to as T', and are evaluated using the objective function in Eq. 14. The best candidate solutions are then selected to form the next generation. The mean and covariance matrix of the multivariate normal distribution are updated based on the selected candidate solutions to bias the search towards promising regions of the search space. This process is repeated until the maximum number of iterations or a desired level of convergence is met. Thus, CMA-ES uses a combination of random search and adaptation of the search distribution to explore the search space efficiently and converge to an optimal solution.

VI. Implementation

Our end-to-end VoxAR system uses Unity3D [35] game engine and Microsoft HoloLens2 as the OST-HMD medium. For volume rendering, we utilize VTK's holographic remote rendering feature [36]. Since placement objectives are implemented using shaders, they can be evaluated on the GPU. However, our current implementation solves the TF adjustment algorithm on the CPU. Thus, after receiving a placement result from the HMD, VoxAR solves the adjusted TF on a compute server and passes the result to VTK for volume rendering. Using the Microsoft Mixed Reality Toolkit (MRTK) holographic remoting feature [37], the volume-rendered result is sent to the HMD over a wireless network. Furthermore, the coupled MRTK and VTK system allows users to perform basic volume interactions – in our case, rotation – which is then communicated to VTK for re-rendering and, subsequently, reprojection in AR. VoxAR assumes certain features, such as depth information and spatial mapping, to be obtained using the HMD's API (such as MRTK). Additionally, many AR toolkits are capable of surface detection, classifying specific surfaces as walls, floors, tables, and more.

VII. Results and Evaluation

We now discuss the results of VoxAR, including findings from a user study we conducted to assess the system. Although the results, as seen through an OST-HMD, cannot be shown as images, we use the MRTK additive shader on the HoloLens Mixed Reality Capture to exhibit the visual quality of the rendered semi-transparent pixels. For this paper, we use the following volume datasets and standard TF presets:

Volume dataset, with sizes:

- Jet heptane gas simulation undergoing combustion (Jet), $[300 \times 300 \times 300]$
- Skull CT (Skull), $[256 \times 256 \times 256]$
- Engine, $[256 \times 256 \times 128]$

Input TFs, with control points :

- Red-White-Blue (Diverging), 3 control points:
- Haze-Cyan, 17 control points:
- Continuous Viridis, 256 control points:

For all TF adjustments, we used a large value of $k = 10$ and $w_{pr} = 10$ for **C1** background discernibility and **C4** CIELAB to RGB projection weights, respectively, to avoid invalid solutions that may need manual correction. All other weights, w_d, w_a, w_q, and w_{jnd}, were set to 0.5. We let the CMA-ES iterate over the optimization for a maximum of 20 iterations. We refer the reader to a mixed-reality video capture of our results in the supplementary material.

In Fig. 3 we demonstrate examples of VoxAR for scientific visualization. Fig. 3(a) *top* shows the Jet volume rendered using the Haze-Cyan TF in a hallway. Following placement – using objectives **O3** center screen and **O1** anywhere in 3D – VoxAR adjusts the TF such that the gas volume at the lighter end of the input TF, blending with the wall and floor, becomes visually contrasting. The resultant volume-rendered object is shown in Fig. 3(a) *bottom*. Fig. 3(b) *top* shows the input and VoxAR TFs projected in the CIELAB space. The gray hull

represents colors in the real-world, captured in 360°centered at the VoxAR placement position. It can be seen here that VoxAR optimizes the TF to avoid intersection with the real-world colors while maintaining the perceptual shape of the input TF. Moreover, Fig. 3(b) *bottom* shows the HSV projection of the input and VoxAR TFs. Extending this example to situated visualization applications, where experts may wish to analyze simulation data connected to its physical referent or location, VoxAR aids in effectively placing the virtual volume in the scene, based on the provided objectives, and study the data without interference with the background colors.

Fig. 3(c) *top* and *bottom* demonstrate the visualization of a human skull CT volume, using an input diverging TF and the VoxAR adjusted TF, respectively, projected on the wall of a medical examination room, using surface magnetism objective **O1**. VoxAR TF alleviates the visually intersecting colors of the wall with the resultant volume-rendering. Here, we also demonstrate **C3**, hue similarity constraint. Using $\lambda_h = 20$, it can be seen in Fig. 3(d) that VoxAR TF minimally adjusted the input TF hues in order to avoid background color intersection. VoxAR can facilitate the integration of AR-based medical visualization applications, specifically in scenarios where medical experts may want to observe and project data in the surroundings or refer to the patient without a blocking video-see-through device. Moreover, the VoxAR TF adjustment algorithm to preserve the perceptual mapping of colors to data attributes while avoiding background interference facilitates the sensitive need to visualize medical data as accurately as possible.

Furthermore, we compare our technique with the most recent work in AR color enhancement by Zhang et al [4] (using $\lambda_E = 0.4$, as suggested in the publication). Fig. 4(a) shows a synthetic volume of shapes with data intensities spread uniformly across the data ranges, rendered using the Haze-Cyan TF. While [4] aids in recovering the shapes blended in the background, highlighted using the dotted annotation in (b), their algorithm does not retain color consistency of the TF, as pointed out using the arrow. This is because [4] performs a pixel-wise operation of the virtual object against its corresponding background color. By designing an algorithm that precedes the volume-rendering step VoxAR, in contrast, takes a more wholesome approach and determines an optimized color spectrum by evaluating simultaneously all the background colors. The improvement in both the color contrast and color consistency using VoxAR is shown in Fig. 4(c).

Next, in Fig. 5 we demonstrate the results of the VoxAR placement for a small working region with varying colors, a surface, and a point defined at the center of the scene with proximity assigned to cover the area. The evaluation was conducted for an input volume with a TF overlapping with the background, shown in Fig. 5(a). Fig. 5(b) shows that for objectives **O4** color discernibility and **O1**, VoxAR places the volume on the surface and in front of the green background, away from the overlapping yellow color. In contrast, for **O2** point proximity and **O1** anywhere in 3D, VoxAR places the volume in front of the blue background, as shown in Fig. 5(c).

Finally, in Fig. 6, we demonstrate VoxAR placement updates for changing FoV. For a changing FoV from the green FoV to the blue FoV in Fig. 6(a), (b) shows the initial placement of the volume for the green FoV. As the user moves to the blue FoV, the placement score drops below a defined threshold of 80% of the original score, and a new *optimal* placement is suggested to the user using a bounding box shown in Fig. 6(c). On performing a pinching gesture, the volume is updated to the new position.

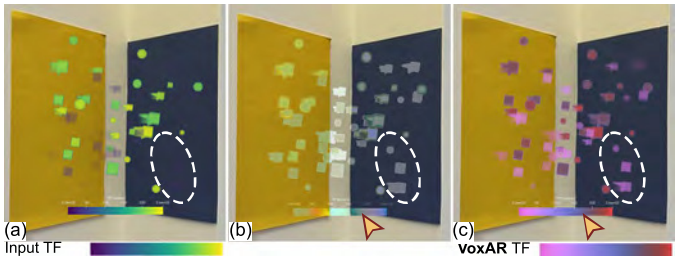

Fig. 4. For an input TF in (a), (b) is the result using [4], and (c) is the VoxAR result. The white dotted annotation shows that both [4] and VoxAR improve visibility against the background, however, VoxAR maintains the TF color consistency, as seen on the color bar pointed by the arrow.

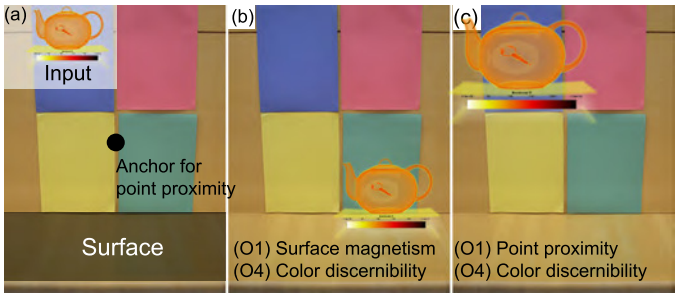

Fig. 5. Evaluation of VoxAR surface placement. For an input TF to be positioned in a scene with colored placeholders and a surface shown in (a), (b) and (c) demonstrate the results based on the objectives provided.

A. User Study

For our user study, we recruited 16 participants: 11 males and 5 females, aged between 19 and 35 (Mean: 27.6±4.7). No participants were color-blind. Since this work is specifically for volume rendering visualization, we required the participants to have an understanding of volume rendering and TFs.

a) Experiment Design: The study was carried out in two sequential parts. First, each participant was asked to place the Engine volume in a simulated 3D scene using one of two predefined objective combinations (**Part I**). The Engine dataset was modified to contain cube and sphere volume primitives (explained later in this section). We divided the setup so two participants had the same objective combination. Moreover, we ensured that the 3D scene and objective combination resulted in similar difficulty for each setup.

Next, we computed an adjusted TF for the user-configured placement and a VoxAR placement using the same objectives. As a result, we generated four scenarios:

S1 User-configured placement + input TF, (UP+OTF)
S2 User-configured placement + its VoxAR TF, (UP+VTF)
S3 VoxAR placement + input TF, (VP+OTF)
S4 VoxAR placement + VoxAR TF, (VP+VTF)

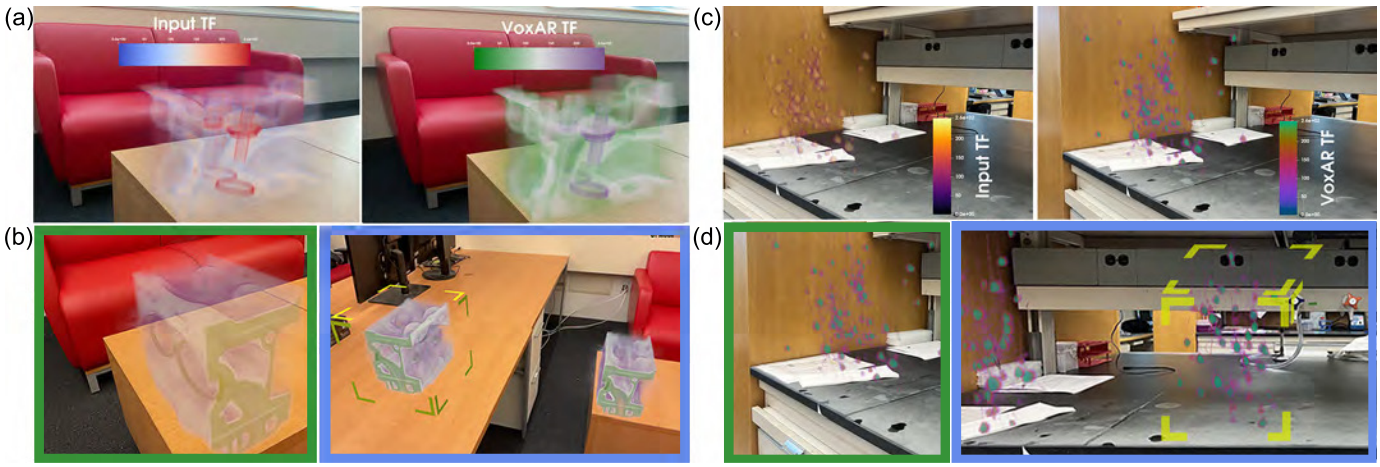

Fig. 6. Updating placement as the user moves from the green to blue FoV.

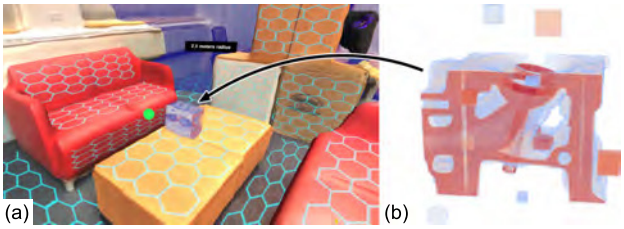

Fig. 7. Screenshot of the visual cues provided to the participants for **Part I**. The transparent 'honeycomb' texture marks valid surface areas, the purple region indicates out-of-bound, and the green dot indicates screen center.

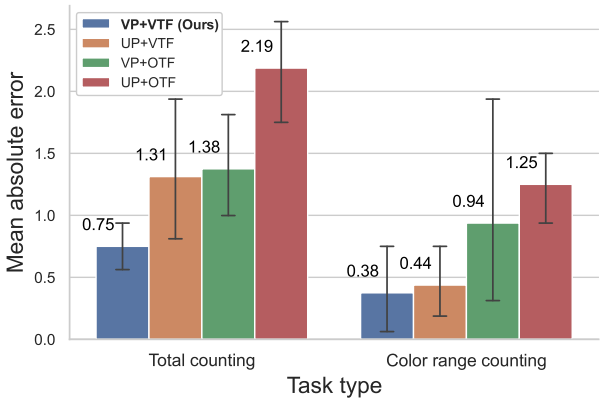

Fig. 8. Comparison of mean Absolute Error between **T1** and **T2** for all scenarios (95% confidence interval).

For each scenario, participants performed two tasks (**Part II**):

T1 Count the number of volumetric primitives of type X.

T2 Count the number of volumetric primitives of type Y that have an intensity value in the range I.

Part I: Each participant was shown a photogrammetry-reconstructed [38] 3D scene in Unity and asked to find an optimal placement for a volume that is either on (1) a surface area or (2) anywhere in the view 2.5m from a given point, and for both, placed as close to the center of the FoV as possible. To aid the participants with the objectives, we provided 3D visuals in the scene as shown in Fig. 7 (a).

Part II: We adopt a within-subject design with two independent variables. Specifically, for each participant, we presented four scenarios, **S1** to **S4**, and asked them to complete tasks **T1** and **T2** for all the scenarios. To avoid learning, the Engine volume was modified for each scenario to include cube and sphere volumes of randomized frequency (with a total cube+volume count of 10), positions, sizes ranging between $10 \times 10 \times 10$ and $20 \times 20 \times 20$ voxels, and intensity values between 0 - 255 (see Fig. 7 (b) for an example). We used the R2B diverging TF as input for **S1** and **S3**, and for ease of identifying intensity ranges in **T2**, we binned the TF colors into five uniform-sized bins. Moreover, for counterbalancing our findings, each participant was presented **S1** to **S4** in a random order.

Before starting **Part II**, we first performed eye and color calibration of the HoloLens, followed by a warm-up session to help the participants familiarize themselves with using the HMD. During warm-up, they were shown a different volume and TF and were asked to practice the hand-gesture-based rotation interactivity. At the start of each trial, the participants were seated where the pre-defined FoV was measured for **Part I** and were asked to respond to the tasks *"as accurately and efficiently as possible."* Based on our current implementation of the VoxAR system, the participants were only allowed to rotate the volume. After every trial, the participants were asked a series of qualitative questions.

b) User Study Results: We present an evaluation of VoxAR by analyzing the quantitative and qualitative responses for **T1** and **T2**, for all scenarios, based on Absolute Error and Task Completion Time. For our analysis, we define **S1** (UP + OTF) as the baseline condition and use it to compare with (UP+VTF), (VP+ OTF), and (VP+VTF).

Absolute Error: We define absolute error as the average measure of how much the answers of the participants differ from the correct answer. Fig. 8 shows a plot of this measure for each scenario, averaged over the total number of trials. Based on this result, we can see that the end-to-end VoxAR technique, placement optimization followed by TF adjustment, significantly decreases the mean absolute error. That is to say, for both tasks, the participants were able to perform the data

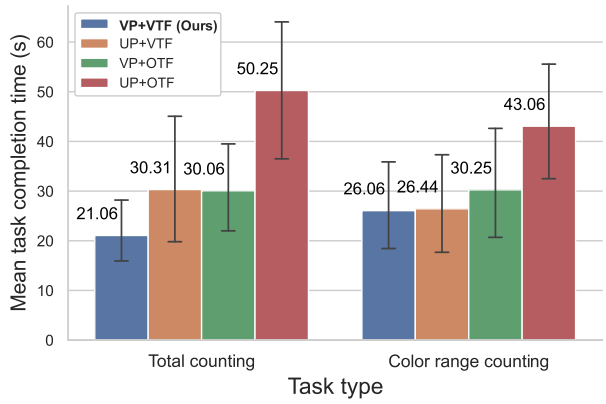

Fig. 9. Comparison of mean task completion time between **T1** and **T2** for all scenarios (95% confidence interval).

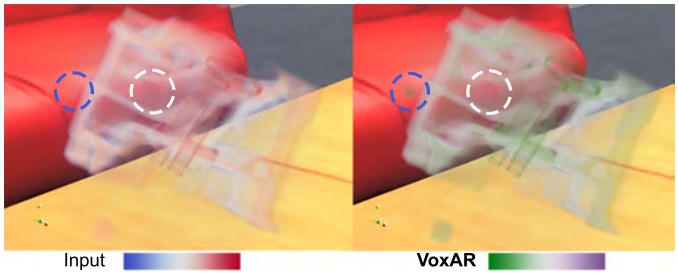

Fig. 10. An example of instances where VoxAR TF aided users to better perceive data, compared to the input TF.

analysis tasks more accurately. Compared to UP+OTF, VoxAR reduces the mean absolute error by 65.8% and 69.6% for **T1** and **T2**, respectively. An application of Friedman's test confirmed that there is a significant effect on the recognition of an element of volumetric objects: Q=16.9; p¡.001 for **T1** and (Q=12.8; p¡.01 for **T2**. Pairwise comparisons using the Nemenyi post-hoc test indicate that the difference between the baseline (UP+OTF) and ours (VP+VTF; VoxAR) in both of the tasks is significant (p=0.001; p=0.021, respectively). In observing the reasons for the difference in visualization effectiveness, we noticed that, as shown in Fig. 10, the background blending made the participants prone to missing smaller primitives (as marked by the blue ring). Moreover, given the similarity of the red TF color with the background, most users misconceived the hole in the volume as a primitive (marked by the white ring). However, using the VoxAR TF, participants were able to deduce that the appearance of the background color represents a hole. The findings also confirm the idea that using either of the VoxAR components, placement optimization or TF adjustment, can improve data perception in OST-AR.

Task Completion Time Next, we measure the task completion time of the two tasks over the total number of trials, as shown in Fig. 9. The results show that VoxAR reduces the time taken to complete each task by 58.1% and 39.5%, on average, respectively. An application of Friedman's test shows that there is a significant effect on the completion time (Q=24.5; p¡.001 for **T1**) (Q=13; p¡.005 for **T2**). Pairwise comparisons using the Nemenyi post-hoc test indicate that the difference between UP+OTF and ours, VP+VTF and VoxAR, is significant (p=0.001; p=0.007, respectively).

c) Subjective feedback: To collect findings for perceived performance, effort, and the certainty of the two tasks, we asked qualitative questions based on the Semantic differential scale [39], at the end of each task. Each question consisted of ratings ranging from 0 to 5, and was anchored by bipolar adjectives. A higher rating indicated that the participant was more confident in their abilities or had a higher positive response towards the condition. The results did not show a uniform tendency across questions. However, the lowest rated condition among all participants was consistently UP+OTF (Mean: 3.3 ± 1.2). The mean ratings of our full method (VP+VTF) was 3.67, and the other two conditions (UP+VTF and VP+OTF) were rated similarly (3.65, 3.79, respectively).

B. System Performance

For all results, we used an Intel Xeon Bronze 3106 CPU with 64GB of RAM and an Nvidia Quadro RTX 6000 as the remote render server. On average, VoxAR placement achieved framerates in excess of the HoloLens target framerate of 60Hz [40]. Considering surface placement alone, VoxAR achieves $>$ 80fps, dependent on placeable surfaces in view. For 3D placement, we achieve 70-80fps, and considering both, we achieve 53-65fps. The performance of VoxAR regarding surface placement is more stable, given that the system only needs a single placement map. Performance regarding 3D placement is more variable as it is negatively correlated with the resolution of the 3D space considered.

The VoxAR TF adjustment is CPU-based and has an $O(n)$ time complexity, depending on the number of TF control points. The diverging TF with 3 control points took 5s, whereas the Viridis TF, which is a continuous TF and has the maximum number of control points (255) took 15s to optimize. As TF adjustment occurs only once, the system is capable of real-time performance after initial placement.

VIII. Conclusion, Limitations, and Future Work

This paper presented VoxAR, a two-step approach for enhancing volume rendering visualization in OST-HMDs. Our method combines spatial and environmental constraints with user preferences to find an optimal placement for the volume at runtime. Once placed, it adjusts the input TF to improve its visual distinctiveness against the real-world background, when rendered, while also maintaining the perceptual mapping between the data attributes and the input TF colors. Furthermore, we have provided a solution to extend the VoxAR method for changing FoV, thus supporting a key utility of AR applications to allow interactivity in the real-world. To evaluate VoxAR, we have demonstrated potential applications and compared our technique with a recent work in OST-AR color enhancement. We also carried out a user study, and its findings suggest that VoxAR facilitates effective and efficient user performance when conducting volume rendering-related comprehension tasks in OST-AR.

Naturally, VoxAR is not free from limitations. It addresses a novel challenge in AR-based scientific visualization. Some limitations are hardware-related. For instance, OST-HMDs cannot project dark colors, such as black, thus limiting the lower luminance range of TF color options. Additionally, many OST-AR devices do not contain a dedicated GPU, thus affecting our shader-driven performance and needing to offload our volume rendering pipeline. To this end, we intend to examine newer devices that allow visor-dimming features and have dedicated GPUs.

At the technique level, although we address changing FoV, further challenges for dynamic scenes need to be addressed. Specifically, converging to a solution becomes difficult as the color spectrum in the background surrounding the initial placement broadens. As such, we aim to investigate the possibility of dynamically adapting the TF colors for immediate yet changing backgrounds in such a way that the TF update would minimally affect the perceptual mapping of its colors to the data attributes. This also implies studying the impact of such changes on data analysis and reasoning during real-time and dynamic visualization adaptation. Moreover, VoxAR does not address high-frequency textures or colors gathered from artifacts due to specular reflection or transparent objects like windows and glass. Finally, we plan to develop a more collaborative version of the system, allowing for multiple users and multiple volumes.

ACKNOWLEDGMENTS

This research was supported in part by the National Science Foundation awards IIS2107224, IIS2107328, IIS1901030, NCS-FO2124179, OAC1919752, ICER1940302, and an IBM-SUNY grant 2106.

REFERENCES

[1] M. Mathur, J. M. Brozovich, and M. K. Rausch, "A brief note on building augmented reality models for scientific visualization," *Finite Elements in Analysis and Design*, vol. 213, p. 103851, 2023.

[2] Unity Technologies, "Advanced workflows for AR developers: Unity MARS," https://unity.com/products/unity-mars, 2023, online; accessed 15-March-2023.

[3] J. a. M. Evangelista Belo, M. N. Lystbæk, A. M. Feit, K. Pfeuffer, P. Kán, A. Oulasvirta, and K. Grønbæk, "AUIT – the adaptive user interfaces toolkit for designing xr applications," in *ACM Symposium on User Interface Software and Technology*, 2022.

[4] Y. Zhang, R. Wang, Y. Peng, W. Hua, and H. Bao, "Color contrast enhanced rendering for optical see-through head-mounted displays," *IEEE Transactions on Visualization & Computer Graphics*, vol. 28, no. 12, pp. 4490–4502, 2022.

[5] Y. Itoh, M. Dzitsiuk, T. Amano, and G. Klinker, "Semi-parametric color reproduction method for optical see-through head-mounted displays," *IEEE Transactions on Visualization & Computer Graphics*, vol. 21, no. 11, pp. 1269–1278, 2015.

[6] K.-H. Lee and J.-O. Kim, "Visibility enhancement via optimal two-piece gamma tone mapping for optical see-through displays under ambient light," *Optical Engineering*, vol. 57, no. 12, p. 123104.

[7] T. Fukiage, T. Oishi, and K. Ikeuchi, "Visibility-based blending for real-time applications," in *IEEE International Symposium on Mixed and Augmented Reality*, 2014, pp. 63–72.

[8] C. Weiland, A.-K. Braun, and W. Heiden, "Colorimetric and photometric compensation for optical see-through displays," in *Universal Access in Human-Computer Interaction: Intelligent and Ubiquitous Interaction Environments Conference.* Springer, 2009, pp. 603–612.

[9] J. David Hincapié-Ramos, L. Ivanchuk, S. K. Sridharan, and P. P. Irani, "SmartColor: Real-Time Color and Contrast Correction for Optical See-Through Head-Mounted Displays," *IEEE Transactions on Visualization & Computer Graphics*, vol. 21, no. 12, pp. 1336–1348, 2015.

[10] A. Erickson, K. Kim, G. Bruder, and G. F. Welch, "A Review of Visual Perception Research in Optical See-Through Augmented Reality," in *International Conference on Artificial Reality and Telexistence and Eurographics Symposium on Virtual Environments*, F. Argelaguet, R. McMahan, and M. Sugimoto, Eds. The Eurographics Association, 2020.

[11] K. Marriott, F. Schreiber, T. Dwyer, K. Klein, N. H. Riche, T. Itoh, W. Stuerzlinger, and B. H. Thomas, *Immersive Analytics*. Springer, 2018, vol. 11190.

[12] D. Lindlbauer, A. M. Feit, and O. Hilliges, "Context-aware online adaptation of mixed reality interfaces," in *ACM Symposium on User Interface Software and Technology*, 2019, p. 147–160.

[13] E. Marino, F. Bruno, and F. Liarokapis, "Color harmonization, deharmonization and balancing in augmented reality," *Applied Sciences*, vol. 11, no. 9, 2021.

[14] N. Hansen, *The CMA Evolution Strategy: A Comparing Review*. Springer, 2006, pp. 75–102.

[15] R. Du, E. Turner, M. Dzitsiuk, L. Prasso, I. Duarte, J. Dourgarian, J. Afonso, J. Pascoal, J. Gladstone, N. Cruces, S. Izadi, A. Kowdle, K. Tsotsos, and D. Kim, "DepthLab: Real-Time 3D Interaction with Depth Maps for Mobile Augmented Reality," in *ACM Symposium on User Interface Software and Technology*, 2020, p. 829–843.

[16] R. Gal, L. Shapira, E. Ofek, and P. Kohli, "Flare: Fast layout for augmented reality applications," in *IEEE International Symposium on Mixed and Augmented Reality*, 2014, pp. 207–212.

[17] Y. Cheng, Y. Yan, X. Yi, Y. Shi, and D. Lindlbauer, "SemanticAdapt: Optimization-based adaptation of mixed reality layouts leveraging virtual-physical semantic connections," in *ACM Symposium on User Interface Software and Technology*, 2021, p. 282–297.

[18] P. Fleck, A. Sousa Calepso, S. Hubenschmid, M. Sedlmair, and D. Schmalstieg, "RagRug: A toolkit for situated analytics," *IEEE Transactions on Visualization & Computer Graphics*, pp. 1–1, 2022.

[19] Microsoft, "Microsoft Mixed Reality Toolkit v2.8.3," https://github.com/Microsoft/MixedRealityToolkitUnity/releases, 2022, online; accessed 1-March-2023.

[20] K. Kiyokawa, Y. Kurata, and H. Ohno, "An optical see-through display for mutual occlusion of real and virtual environments," in *IEEE and ACM International Symposium on Augmented Reality*, 2000, pp. 60–67.

[21] O. Cakmakci, Y. Ha, and J. P. Rolland, "A compact optical see-through head-worn display with occlusion support," in *IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE, 2004, pp. 16–25.

[22] J. L. Gabbard, J. E. Swan, J. Zedlitz, and W. W. Winchester, "More than meets the eye: An engineering study to empirically examine the blending of real and virtual color spaces," in *IEEE Virtual Reality Conference*. IEEE, 2010, pp. 79–86.

[23] O. Kutter, A. Aichert, C. Bichlmeier, J. Traub, S. Heining, B. Ockert, E. Euler, and N. Navab, "Real-time volume rendering for high quality visualization in augmented reality," in *Augmented Environments for Medical Imaging including Augmented Reality in Computer-aided Surgery*, 2008, pp. 104–113.

[24] Y. Chu, X. Li, X. Yang, D. Ai, Y. Huang, H. Song, Y. Jiang, Y. Wang, X. Chen, and J. Yang, "Perception enhancement using importance-driven hybrid rendering for augmented reality based endoscopic surgical navigation," *Biomedical optics express*, vol. 9, no. 11, pp. 5205–5226, 2018.

[25] W. Luo, A. Lehmann, H. Widengren, and R. Dachselt, "Where should we put it? Layout and placement strategies of documents in augmented reality for collaborative sensemaking," in *Proceedings of ACM CHI Conference on Human Factors in Computing Systems*, 2022.

[26] B. Lee, X. Hu, M. Cordeil, A. Prouzeau, B. Jenny, and T. Dwyer, "Shared surfaces and spaces: Collaborative data visualisation in a co-located immersive environment," *IEEE Transactions on Visualization & Computer Graphics*, vol. 27, no. 02, pp. 1171–1181, 2021.

[27] P. Ljung, J. Krüger, E. Groller, M. Hadwiger, C. D. Hansen, and A. Ynnerman, "State of the art in transfer functions for direct volume rendering," *Computer Graphics Forum*, vol. 35, no. 3, pp. 669–691, 2016.

[28] ISO/CIE 11664-4, "Colorimetry—Part 4: CIE 1976 L*a*b* colour space," *European Standard*, 2019.

[29] M. R. Luo, G. Cui, and B. Rigg, "The development of the cie 2000 colour-difference formula: Ciede2000," *Color Research & Application*, vol. 26, no. 5, pp. 340–350, 2001.

[30] M. D. Fairchild, *Color Appearance Models*. John Wiley & Sons, 2013.

[31] M. Stone, D. A. Szafir, and V. Setlur, "An engineering model for color difference as a function of size," in *Color and Imaging Conference*. Society for Imaging Science and Technology, 2014, pp. 253–258.

[32] C. C. Gramazio, D. H. Laidlaw, and K. B. Schloss, "Colorgorical: Creating discriminable and preferable color palettes for information visualization," *IEEE Transactions on Visualization & Computer Graphics*, vol. 23, no. 1, pp. 521–530, 2017.
[33] M. Mahy, L. Van Eycken, and A. Oosterlinck, "Evaluation of Uniform Color Spaces Developed after the Adoption of CIELAB and CIELUV," *Color Research & Application*, vol. 19, no. 2, pp. 105–121, 1994.
[34] D. A. Szafir, M. Stone, and M. Gleicher, "Adapting color difference for design," in *Color and Imaging Conference*, vol. 2014. Society for Imaging Science and Technology, 2014, pp. 228–233.
[35] Unity Technologies, "Unity3D," https://unity3d.com/unity/, 2023, online; accessed 1-March-2023.
[36] Kitware, "Holographic Remoting: Stream VTK to the HoloLens 2," https://www.kitware.com/stream-vtk-to-the-hololens-2/, online; accessed 30-March-2023.
[37] Microsoft, "Holographic remoting — MRTK2," https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/features/tools/holographic-remoting?view=mrtkunity-2022-05, online; accessed 30-March-2023.
[38] M. Queisner, M. Pogorzhelskiy, C. Remde, J. Pratschke, and I. M. Sauer, "VolumetricOR: A new approach to simulate surgical interventions in virtual reality for training and education," *Surgical Innovation*, vol. 29, no. 3, pp. 406–415, 2022.
[39] J. G. Snider and C. E. Osgood, *Semantic differential technique; a sourcebook*. Aldine Publishing Company, 1969.
[40] Microsoft, "App quality criteria – Mixed Reality," https://learn.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/app-quality-criteria-overview, 2022, online; accessed 1-March-2023.

Saeed Boorboor is currently pursuing a Ph.D. degree in Computer Science at Stony Brook University. He received his B.S. in Computer Science from the School of Science and Engineering, Lahore University of Management Sciences, Pakistan. His research interests include immersive analytics, scientific visualization, medical imaging, and computer graphics.

Matthew Castellana is currently pursuing a Ph.D. degree in Computer Science at Stony Brook University. He received his B.S. in Computer Science and B.E. in Computer Systems and Engineering from Rensselaer Polytechnic Institute. His research interests include virtual and augmented reality, novel VR/AR peripherals and interfaces, remote collaboration, graphics, and related areas.

Yoonsang Kim is currently pursuing a Ph.D. degree in Computer Science at Stony Brook University. He received his Bachelor's degree in Computer Science from Soongsil University, South Korea. His research interests include multi-user scenarios, platform/device-agnostic scenarios in XR, AR privacy and security, and computer graphics.

Zhutian Chen is a postdoc at the Visual Computing Group at Harvard University. Before joining Harvard, he was a postdoc at University of California San Diego, and a Ph.D. student at Hong Kong University of Science and Technology. His interests are in Information Visualization, Human-Computer Interaction, and Augmented Reality.

Johanna Beyer received the PhD degree in computer science with the University of Technology Vienna, Austria in 2009. He is currently a research associate with Visual Computing Group, Harvard University. Before joining Harvard, she was a postdoctoral fellow with Visual Computing Center, KAUST. Her research interests include scalable methods for visual abstractions, large-scale volume visualization, and immersive analytics.

Hanspeter Pfister is currently An Wang professor of computer science with the John A. Paulson School of Engineering and Applied Sciences, Harvard University. His research interests include visual computing, including intersection of scientific visualization, information visualization, computer graphics, and computer vision and spans a wide range of topics, including biomedical image analysis and visualization, image and video analysis, and visual analytics in data science.

Arie E. Kaufman is a Distinguished Professor of Computer Science, Director of Center of Visual Computing, and Chief Scientist of Center of Excellence in Wireless and Information Technology at Stony Brook University. He served as Chair of Computer Science Department, 1999-2017. He has conducted research for >40 years in visualization, VR and graphics and their applications, and published >350 refereed papers. He was the founding Editor-in-Chief of IEEE TVCG, 1995-98. He is an IEEE Fellow, ACM Fellow, National Academy of Inventors Fellow, recipient of IEEE Visualization Career Award (2005), and inducted into Long Island Technology Hall of Fame (2013) and IEEE Visualization Academy (2019). He received his Ph.D. in Computer Science from Ben-Gurion University, Israel (1977).