International Conference on Artificial Reality and Telexistence
Eurographics Symposium on Virtual Environments (2022)
J.-M. Normand and H. Uchiyama (Editors)

FoReCast: Real-time Foveated Rendering and Unicasting for Immersive Remote Telepresence

Y. T. Tefera [1,2], D. Mazzanti[1], S. Anastasi[3], D. G. Caldwell[1], P. Fiorini[2], and N. Deshpande[1] †

[1]Istituto Italiano di Tecnologia (IIT), Via Morego, 30, 16163, Genova, Italy
[2]Department of Computer Science, University of Verona, Via Strada Le Grazie 15, Verona, Italy
[3] Istituto Nazionale per l'Assicurazione contro gli Infortuni sul Lavoro (INAIL), P.le Pastore 6, 00144 Rome, Italy

Abstract

Rapidly growing modern virtual reality (VR) interfaces are increasingly used as visualization and interaction media in 3D telepresence systems. Remote environments scanned using RGB-D cameras and represented as dense point-clouds are being used to visualize remote environments in VR in real-time to increase the user's immersion. To this end, such interfaces require high quality, low latency, and high throughput transmission. In other words, the entire system pipeline from data acquisition to its visualization in VR has to be optimized for high performance. Point-cloud data particularly suffers from network latency and throughput limitations that negatively impact user experience in telepresence. The human visual system provides an insight into approaching these challenges. Human eyes have their sharpest visual acuity at the center of their field-of-view, and this falls off towards the periphery. This visual acuity fall-off was taken as an inspiration to design a novel immersive 3D data visualization framework to facilitate the processing, transmission, and visualization in VR of dense point-clouds. The proposed FoReCast framework, shows significant reductions in latency and throughput, higher than 60% in both. A preliminary user study shows that the framework does not significantly affect the user quality of experience in immersive remote telepresence.

CCS Concepts
• ***Human-centered computing*** → ***Visualization design and evaluation methods; Mixed / augmented reality;***

1. Introduction

Immersive Remote Telepresence (IRT) systems have recently received increased interest due in no small measure to the ready availability of good quality, low-cost, consumer-grade RGB-D sensors and immersive virtual reality (VR) devices [ZSG*18]. The visualization of dense point-clouds using VR headsets in real-time, can allow immersive visualization by an observer, perceiving the colour and the 3D profile of a real-world remote environment simultaneously [SKH*19, MK16]. This is the key distinguishing factor from traditional telepresence systems, which rely on mono- or stereo-video feedback and suffer from limitations in terms of fixed or non-adaptable camera viewpoints, occluded views of the remote space, etc. [CHB07, KYIS14]. Nevertheless, the increased data footprint in processing dense point-clouds in real-time imposes hard constraints regarding latency, throughput, and the visual quality for the user [SKS*19, RWF*20]. For instance, latency and low quality have been shown to reduce the sense of presence and provoke cybersickness [MRWB03, SNL20]. For many applications, including remote inspection and disaster response, these constraints are further exacerbated since the scene is *a priori* unknown and should be visualized in real-time from the 3D input data (e.g., depth maps, point-clouds). *IRT* therefore presents the challenge of appropriately managing the typical data flow from remote data acquisition, processing, streaming, compression, decoding, and visualization to the user, while allowing optimal visual quality [OERF*16, RWF*20].

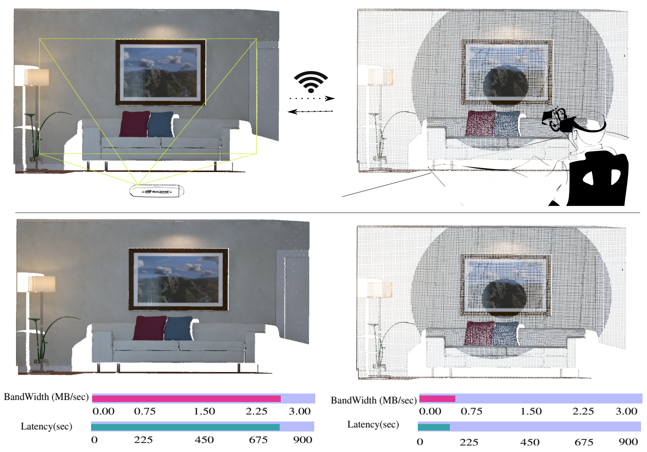


Figure 1: *TOP: shows the FoReCast framework, where remotely acquired point-cloud is streamed and rendered in real-time. BOTTOM: Bandwidth and Latency numbers for - left: without foveation; and right: applying the FoReCast framework.*

† This research was conducted in collaboration with the Italian National Worker's Compensation Authority (INAIL), under the "Robot Teleoperativo 2" project.

DOI: 10.2312/egve.20221278

In this paper, the human visual system provides the inspiration to address the coupling between the 3D data acquisition and its visualization inside VR. Specifically, the concept of *foveation* (or, *gaze-contingent*) [Hen05], which predicates that the human visual system has the highest visual acuity at the center of its field-of-view, and the acuity falls off towards the periphery. This concept has been used by gaze-contingent rendering techniques for displays with fixed focal distances, such as desktop monitors or stereo displays [GFD*12,PSK*16,TLZ*18]. In this work, we use this acuity fall-off to facilitate the sampling, compression, streaming (unicasting), and rendering of dense point-clouds for a remote user, thereby reducing the amount of data to be transmitted and the consequent latency. The *FoReCast* framework, utilizes the user's fixation point to divide the acquired point-cloud into concentric conical regions of progressively reducing resolution away from the center. As a result, this approach presents a re-thinking of the *IRT* processes - it includes the user's perspective in optimizing the data flow and quality of the remote scene visualized at the user, exclusive to the user in VR from a first-person perspective. Figure 1-top demonstrates the concept of the *FoReCast* framework, while -bottom shows the crux of the problem being addressed in this paper. The remote scene is captured using an RGB-D camera, the acquired point-cloud is streamed to the user site, and it is visualized immersively inside the VR head-mounted display (HMD). As can be seen, when the point-cloud is streamed and rendered as-captured, the bandwidth and latency numbers are very high. In contrast, the same point-cloud, streamed and rendered using the *FoReCast* framework, shows significant reductions on both counts. It is important to note that as an image, Fig. 1 can only depict the VR scene from a third-person perspective. Since, the *FoReCast* framework is optimized for the user's first-person perspective, the quality of the foveated rendering can be judged only when the user is inside VR. Therefore, evaluating the framework through focused user studies is a crucial part of this ongoing research.

2. Related Work and Contribution

Our proposed approach relies on diverse fields including gaze tracking, real-time 3D data processing, compression, streaming, foveation, and efficient rendering for immersive telepresence. The most relevant recent approaches to *IRT* are briefly discussed here.

A. Immersive Remote Telepresence: Researchers have long seen the advantages of using 3D VR environments in telepresence [Bej96, MB97]. In recent investigations, Maimone et al. [MF11] were among the first to investigate a telepresence system offering fully dynamic, real-time 3D scene capture and viewpoint flexibility using a head-tracked stereo 3D display. Orts-Escolano et al. [OERF*16] presented "holoportation" doing high quality 3D reconstruction for small, fixed-sized regions of interest. Authors in [FCG*17,MK16] present remote exploration telepresence systems for large- and small-scale regions of interest with reconstruction and real-time streaming of 3D data. In [WSKK20, SKS*19], the authors have taken this idea further with simultaneous immersive live telepresence for multiple users for remote robotic telepresence and collaboration. Furthermore, VR-based immersive interfaces for robotic teleoperation have gained a lot of traction, where models of the remote robots are combined with real-time point-cloud, real-time stereo video, and gesture tracking inside VR [PBAR15, KSE*17, LFR18, TSLM17, WRU*18, RWF*20, NMB*21]. These approaches have benefited from commercially available HMDs that include eye-trackers are the Fove-0, Varjo VR-1, PupilLabs Core, and the HTC Vive Pro Eye [SNW*21]. At the same time, efficient compression and representation techniques, e.g., point-cloud compression, signed distance fields are being investigated intensively to address the large memory and bandwidth requirements for such 3D data [WSKK20,HPKG08,KBR*12,SK06,MBC16].

B. Foveated Rendering: Guenter et al. [GFD*12] presented one of the first foveated rendering techniques to accelerate graphics computation. Stengel et al. [SGEM16] proposed gaze-contingent rendering that only shades visible features of the image while cost-effectively interpolating the remaining features. The work by Bruder et al. [BSB*19] used a sampling mask computed based on visual acuity fall-off using the Linde-Buzo-Gray algorithm. In the commercial domain, VR headsets are exploiting foveated rendering for increased realism and reduced graphical demands [Cha21].

2.1. Contributions

Drawing on the advancements in the above fields, this article presents our work *FoReCast*. This is the first attempt, to the best of our knowledge, towards utilizing the concept of foveation for sampling and rendering of real-time dense 3D point-cloud data for immersive visualization. The following contributions are proposed:

1. *Main:* A novel approach for foveated rendering of real-time, remote 3D data, for immersive visualization in VR, i.e., differentially sampling, unicasting, and rendering of real-time dense point-clouds in VR, exploiting the human visual system.
2. *Supplementary:*
   a. A method for unicasting of partitioned point-cloud and combining it at the user site using GPU and CPU parallelization.
   b. A new volumetric point-cloud density based peak signal-to-noise ratio (PSNR) metric to evaluate the proposed approach.
   c. A preliminary user study with 24 subjects to evaluate the impact of the proposed approach on perceived visual quality.

3. Human Visual Acuity and Foveation

Figure 2-A shows the photoreceptor (cones and rods) distribution in the human retina and the retinotopic organization for vision. The cone density is highest in the central region of the retina, and reduces monotonically to a fairly even density into the peripheral retina region. This distribution underpins the concept of *Foveation*, and helps define the idea of visual acuity. Retinal eccentricity implies the angle at which the light from the image gets focused on the retina. With the photoreceptors' monotonically reducing density, approximating the retina as being formed of discrete concentric regions, where the density of the photoreceptors is inversely proportional to the eccentricity angles [QCF*19], helps simplify the concept of *Foveation*. Table 1 gives an example of such an approximation with six retinal regions [QCF*19].

A. Visual Acuity: Visual acuity is quantitatively represented in terms of *minimum angle of resolution* (MAR, measured in arcminutes) [Wey58,GFD*12,SRJ11]. MAR accounts for the number of

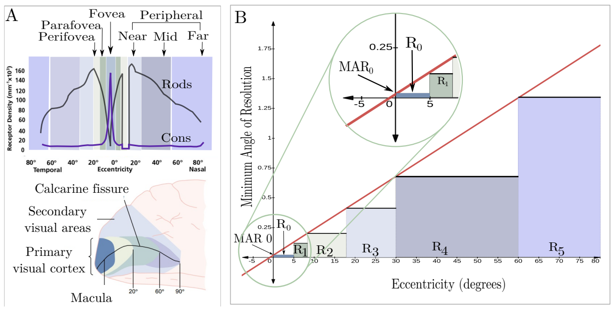


Figure 2: *A) Photoreceptor distribution in the retina and retinotopic organization. B) Minimum Angle of Resolution (MAR) against Eccentricity for the retinal regions* $(R0-R5)$*, based on Eq.1.*

Table 1: *Human retinal regions and their sizes in diameter and eccentricity angle (example derived from [QCF*19]).*

	Region	Diameter (mm)	Eccentricity e (°)
R_0	Fovea	1.5	5°
R_1	ParaFovea	2.5	8°
R_2	PeriFovea	5.5	18°
R_3	Near Peripheral	8.5	30°
R_4	Mid Peripheral	14.5	60°
R_5	Far Peripheral	26	$> 60°$

neurons allocated to process the information from the visual field, as a function of the eccentricity [CR74]. This relation between MAR and eccentricity can be approximated as a linear model, which has been shown to closely match the anatomical features of the eye [Wey58, GFD*12, SRJ11, BSB*19].

$$MAR = mE + MAR_0 \tag{1}$$

Here MAR_0 is the intercept, which signifies the smallest resolvable eccentricity angle for humans, and m is the slope of the linear model. MAR_0 for a healthy human varies between 1 and 2 arcminutes, i.e., $1/60°$ -to- $1/30°$ ($1°$ = 60 arcminutes). Authors in [GFD*12] experimentally determined the values of m based on observed image quality, ranging between 0.022 to 0.034. Figure 2-B captures this linear relationship of Eq. 1, showing how visual acuity degrades as a function of eccentricity, represented with a piece-wise constant approximation, i.e., each retinal region has a distinct constant MAR value [GFD*12].

Foveating a 3D point-cloud implies introducing concentric regions in it that correspond to the retinal fovea regions. The regions are centered on the human eye gaze direction, each of them having a specific radius and its associated visual acuity, i.e., rendering quality. The following section describes how this is done practically.

4. Real-time 3D Data Acquisition and Foveated Sampling

The 3D data acquisition module acquires the RGB-D images from an RGB-D cameras, e.g., Intel RealSense, ZED stereo camera and project each pixel into 3D as point-cloud map. The point-cloud is represented using an unordered list of surfels [WLSM*15], where each surfel has a position $\mathbf{p} \in \mathbb{R}^3$, a normal $\mathbf{n} \in \mathbb{R}^3$, a colour $\mathbf{c} \in \mathbb{R}^3$, a weight $w \in \mathbb{R}$, a radius $r \in \mathbb{R}$, an initialization timestamp t_0, and a current timestamp t. The camera's intrinsic matrix $\mathbf{K}$ is defined by: (i) the focal lengths f_x and f_y, (ii) a principal point in the image (c_x, c_y), and (iii) the radial and tangential distortion coefficients k_1, k_2 and p_1, p_2 respectively. The domain of the image space in the incoming RGB-D frame is defined as $\Omega \subset \mathbb{N}^2$, with the colour image $\mathbf{C}$ having pixel colour $\mathbf{c} : \Omega \to \mathbb{N}^3$, and the depth map $\mathbf{D}$ having pixel depth $d : \Omega \to \mathbb{R}$. Given $\mathbf{K}$, the 3D back-projection of a pixel $\mathbf{u}_i = [x_i, y_i]^T \in \Omega$ for a given depth value $d(\mathbf{u}_i) \in \mathbf{D}$ is defined as $\mathbf{p}_i(\mathbf{u}_i, d(\mathbf{u}_i)) = \mathbf{K}^{-1}[\mathbf{u}_i, 1]^T d(\mathbf{u}_i)$. Over all pixels $\mathbf{u}_i$, this converts the RGB-D frame into a point-cloud map, $\mathcal{M}$.

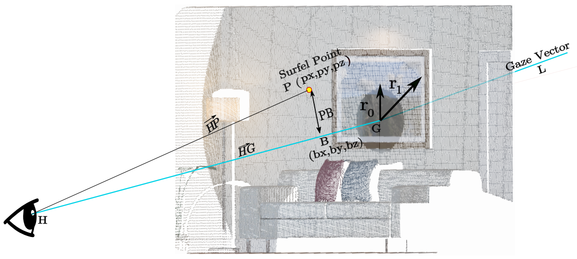


Figure 3: *Map partitioning - the surfel point* $\mathbf{P}(px, py, pz)$ *is classified using the ray* $\mathbf{L}$ *cast from the point of origin* $\mathbf{H}(hx, hy, hz)$*.*

A. Map Partitioning: The foveation model projects the retinal fovea regions into the point-cloud map $\mathcal{M}$, thereby partitioning it into concentric regions. It is then resampled to approximate the monotonically decreasing visual acuity in the foveation model, termed *foveated sampling*.

The center of the eye gaze is used as a point of origin $\mathbf{H}(hx, hy, hz)$. To partition $\mathcal{M}$ into N retinal regions, a ray is cast from $\mathbf{H}(hx, hy, hz)$. This ray, i.e., the *gaze vector* $\mathbf{L} \in \mathbb{R}^3$ is extended up to the last point of intersection $\mathbf{G}(gx, gy, gz)$ with the surfel map. The N foveation regions are now structured around $\mathbf{L}$. With point-cloud data, the concentric regions are conical volumes, with their apex at $\mathbf{H}(hx, hy, hz)$ (Fig. 3), with increasing radii away from $\mathbf{H}$. Algorithm 1, which is implemented in the GPU for faster processing, details how the radii are calculated based on d^{vi} for each surfel. To assign each surfel in $\mathcal{M}$ to a particular region $\mathcal{M}_n \, \forall n \in \{0...N\}$, the shortest distance, i.e., the perpendicular distance between the surfel and $\mathbf{L}$ is used. As shown in Fig. 3, the shortest distance from the surfel $\mathbf{P}(px, py, pz)$ to the ray $\mathbf{L}$ is the perpendicular $\mathbf{PB} \perp \mathbf{L}$, where $\mathbf{B}(bx, by, bz)$ is a point on $\mathbf{L}$. $\|PB\|$ can be obtained using the the projection of $\vec{HP}$ on $\mathbf{L}$, i.e., the cross product of $\vec{HP}$ and $\vec{HG}$, normalized to the length of $\vec{HG}$, $\|PB\|_{\mathbf{P}} = \frac{\|\vec{HP} \times \vec{HG}\|}{\|\vec{HG}\|}$. Algorithm 1 assigns surfel $\mathbf{P}_i$ to the region $\mathcal{M}_n$.

B. Foveated Point-cloud Sampling: The partitioned global map $\mathcal{M}$, with the region-assigned surfels, is converted into a point-cloud library (PCL) data structure, $\mathcal{P}_n$ for each $\mathcal{M}_n$ region. This conversion is sped up using the CUDA-OpenGL interoperability implementation. The CUDA implementation is used for map partitioning and foveated sampling, while OpenGL draws the point-cloud for texture and parameter initialization. CUDA and OpenGL both run on the GPU and share data through common memory in the

Algorithm 1: Map Partitioning Algorithm

```
Input: $\mathcal{M}$            /* Map to be partitioned */
       $\mathbf{L}$             /* Gaze direction vector */
    $e_{0 \ldots N}$  /* Eccentricity angles for $N$ regions */
foreach surfel $P_i$ in the map $\mathcal{M}$ do
    $\mathbf{B} \leftarrow \mathbf{proj}_{\mathbf{L}}^{P_i}$   /* projection of point $\mathbf{P}_i$ on $\mathbf{L}$ */
    $d^{vi} \leftarrow \|\vec{HB}\|$     /* distance between H and B */
    $d \leftarrow \mathbf{PB} \perp \mathbf{L}$            /* shortest distance */
    for $j$=1 to $N$ do
        $r_j \leftarrow \tan(e_j) * d^{vi}$  /* compute the radii $r_j$ */
    end
    /* put $P_i$ into the maps $\mathcal{M}_0$, ..., $\mathcal{M}_n$ */
    if $d < r_0$ then
        $\mathcal{M}_0 \leftarrow \mathbf{P}_i$;
    else if $d > r_0$ AND $d \leq r_1$ then
        $\mathcal{M}_1 \leftarrow \mathbf{P}_i$;
    ⋮
    else
        $\mathcal{M}_n \leftarrow \mathbf{P}_i$
    end
end
```

framebuffer. To implement the foveated sampling, the $\mathbb{R}^3$ space of each $\mathcal{P}_n$ region needs to be further partitioned into an axis-aligned regular grid of cubes, i.e., *voxelization*, as shown in Figure 4. After voxelization, the down-sampling of points follows the foveation model - the voxels in the fovea region of the PCL are the most dense, and this density progressively reduces towards the peripheral regions. This voxelization and down-sampling is a three-step process: First, calculating the volume of the voxel grid for each region by simply calculating the point-cloud distribution for that region, $[(x_{n,min}, x_{n,max}), (y_{n,min}, y_{n,max}), (z_{n,min}, z_{n,max})]$. Second, calculating the voxel size, $\mathfrak{v}$, is a more involved process. Consider the voxelization of the central fovea region, $\mathcal{P}_0$. From sec. 3, the smallest resolvable angle for a healthy human, i.e., MAR_0 is 1 arcminute, i.e., 0.016667°. Eq. (2) gives the smallest resolvable object length,

$$\mathfrak{l} = d^{vi} * \tan(MAR_0) \tag{2}$$

Eq. (2) itself could provide the optimum voxel size, $\mathfrak{v}$. The important consideration here is the value of d^{vi}. In alg. 1, a d^{vi} value for each point is calculated. In contrast, here in order to down-sample the region based on the voxelization, we calculate one d^{vi} value for the entire $\mathcal{P}_0$ region, approximated as the distance from the eye (point of gaze origin) to the 3D centroid, $\mathfrak{pc}_0$, of the point-cloud in the fovea region.

$$\begin{aligned} d_0^{vi} &= \mathbf{d}(\mathbf{H}, \mathfrak{pc}_0) \\ \mathfrak{v}_0 &= d_0^{vi} * \tan(MAR_0) \end{aligned} \tag{3}$$

, where **H** is the eye gaze origin, and $\mathfrak{v}_0$ is the voxel size for the $\mathcal{P}_0$ region.

Once the voxelization of region $\mathcal{P}_0$ is finalized, for the subsequent concentric regions from $\mathcal{P}_1$ to $\mathcal{P}_n$, the voxel sizes are correlated with the linear MAR relationship in Fig. 2-B. Eq. (4) shows that as the eccentricity angle of the regions increases, so do the voxel sizes.

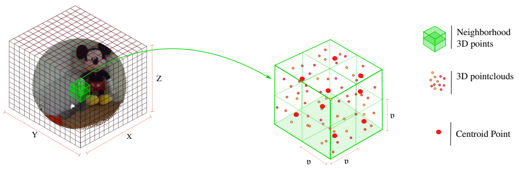


Figure 4: *PCL voxelization - the point-cloud inside each voxel is approximated by its centroid in that voxel.*

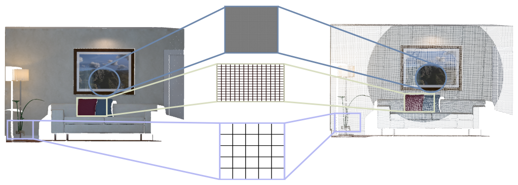


Figure 5: *Foveated point-cloud sampling example showing the different voxel grid sizes for the different regions.*

$$\begin{aligned} MAR_n &= m \cdot E_n + MAR_0 \\ \mathfrak{v}_n &= \frac{MAR_n}{MAR_{n-1}} * \mathfrak{v}_{n-1} \end{aligned} \tag{4}$$

The increasing voxel size away from the fovea region implies that when the third step, i.e., down-sampling, is applied, the approximation of the point-cloud within a voxel is done over progressively dense voxels. This down-sampling done by approximating the point-cloud within each voxel with its 3D centroid. Figure 4 shows the centroid approximation of the point-cloud, while Fig. 5 shows the sample voxel grids for the different regions. This three-step foveated sampling and compression process is computationally expensive, and the OpenMP parallel programming method is used to achieve real-time performance.

5. The *FoReCast* framework

The overall goal is to accelerate the processing, transmission, and rendering of real-time dense point-clouds remotely in VR, while maintaining a rich visual experience for the user. To that end, the *FoReCast* framework, seen in Figure 6, comprises a server-client architecture that encapsulates the foveation methodology. It is divided into three major parts: the **user site**, the **remote site**, and a **packetization and communication network** between them.

A. User Site: The user site manages the: (1) decoding and rendering of the streamed maps, (2) tracking of the eye-gaze and HMD pose, and (3) real-time transfer of this information to the remote site. To visualize and explore the streamed maps, a VR-based interface is designed using the Unreal Engine (UE), which serves as the immersive VR environment for the user. The user site and the remote site are independent environments with their respective reference frames. Therefore, a change-of-bases transformation is necessary to transform the obtained gaze-direction and HMD pose from the UE environment to the remote site environment (Linux, OpenGL), and vice versa, as shown in Figure 7. Likewise, the received point-cloud needs to be visualized in UE at the user site,

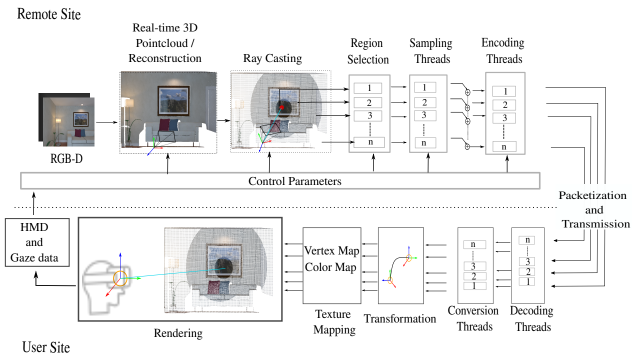


Figure 6: *Schema of the proposed Foveated Rendering framework.*

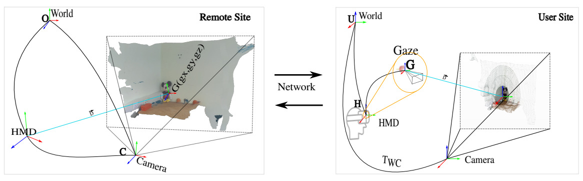


Figure 7: *Schematic showing the coordinate systems.*

and positioned appropriately based on the remote site camera pose, through a similar transformation.

At the user site, to render the received real-time point-cloud data, the following modules were developed, as seen in Figure 6:

1. **a real-time parallel streamer and decoder:** The decoding module includes the state-of-the-art point-cloud codec algorithm from [MBC16]. However, this algorithm uses an octree subdivision and occupancy coding for point-clouds, which is computationally expensive for dense point-clouds. In this work, we modify this algorithm to enable parallel compression and streaming for each foveated region over a TCP socket using Boost ASIO, which is synchronized through a time-stamp.
2. **conversion system:** to convert each decoded point-cloud region $\mathcal{P}_n$ into a texture for visualization in UE, while also transforming the reference frame of the received data.
3. **rendering system:** to transfer the textures to the UE GPU shaders for real-time rendering, based on [Kra19].

B. Remote Site: The system consists of modules for acquisition, point-cloud conversion, sampling, and streaming, as shown in Fig. 6. Figure 7shows the reference frames of the remote site. During real-time point-cloud conversion, a parallel module receives information about the head pose ${}^{U}\mathbf{H}$ and the gaze direction ${}^{U}\vec{D}$ from the user site, and both have to undergo a coordinate transformation, as stated earlier.

C. Communication network: Between the remote and user sites, a custom point-cloud streaming pipeline was implemented using the Boost ASIO cross-platform C++ library for network and low-level I/O programming. ROSbridge [CJO*17] was used to communicate the head pose ${}^{U}\mathbf{H}$, the gaze direction vector ${}^{U}\vec{D}$, and the pose of the remote camera ${}^{O}\mathbf{P}$, among other things.

D. Framework Implementation: The proposed framework is implemented on the following software and hardware components.

Figure 8: *Sample frames from three of the four evaluation datasets.*

1. User site: The HTC Vive Pro Eye VR headset, which comes with a built-in Tobii Eye Tracking system, that has accuracy of $0.5^\circ - 1.1^\circ$ with a trackable FOV of 110°. The PC is running Windows 10 operating system with Nvidia GeForce GTX 1080 graphics card.
2. Remote site: The ZED stereoscopic camera served as the RGB-D camera with a resolution of 1280 X 720. The computing unit consists of an MSI GE63 Raider laptop with Intel Core i7-8750H CPU @ 2.20 GHz, 12 cores, and an Nvidia GP104M Graphics card running Ubuntu operating system.
3. Communication: an Ethernet LAN connection was used between the server and the client, passing through a NightHawk Pro Gaming (SX10) 10 Gbit/s switch.

6. Experiment Design and Evaluation Metrics

The experiment design focuses on a thorough evaluation of the *FoReCast* framework using datasets, online and acquired, using defined experimental conditions and benchmarking against defined metrics.

A. Datasets: For the evaluation, four datasets were used and sample images are shown in Figure 8. Two of these were online synthetic datasets of static environments [HWMD14], consisting of a living room (**LIV**) and an office scene (**OFF**), both provided with the ground truth. Additional dynamic scene with a moving balloon (**BAL**) is captured in a lab area (inspired by TUM dataset [PBL*19]).

B. Experimental Conditions: Three test foveation conditions were created, each having a different combination of the six regions mentioned in sec. 3, going from high performance gain to high visual quality.

- **F1:** The point-cloud has four partitions - Fovea, Parafovea, Perifovea, and the rest. The progressive foveated sampling in the regions follows Eq. (4). For the *rest* of the point-cloud region, it is sampled using the voxel sizes for the *Far Peripheral* region.
- **F2:** has five partitions - Fovea, Parafovea, and Perifovea, *Near Peripheral*, and then the rest, with a similar sampling strategy as F1.
- **F3:** includes all six partitions as seen in Table 1 - Fovea, Parafovea, Perifovea, Near-, Mid-, and the Far Peripheral regions, with the corresponding sampling strategy.

In addition to the three conditions above, two further conditions are created to represent the two ends of the sampling scale, i.e., full-sampling and no-sampling.

- **F0:** To simulate the approach of fully down-sampling a point-cloud to allow the least streaming costs, the whole point-cloud is down-sampled with the voxel size of the *Near Peripheral* region in Eq. (4).

- **FREF:** We used the the state-of-the-art point-cloud compression method from [MBC16] as a base reference condition, against which all the other conditions F0 - F3 are compared. In this condition the visual field is left untouched and the *FoReCast* framework is not applied.

C. Evaluation Metrics: The following objective and subjective metrics were used to evaluate the *FoReCast* framework:

1. Data transfer rate: measured as an overall value between the user and remote sites, using the network data packet analysis tool, Wireshark [San17].
2. Latency: The end-to-end latency is composed of the sub-components in the framework: (1) at the remote site - data acquisition (log-read RGB-D images), ray-casting, conversion (surfels into PCL data structure), sampling, and encoding; and (2) at the user site - decoding, conversion, and rendering. In addition, the pre-specified latency includes: (1) the eye tracker - around $8ms$ ($120\ Hz$); (2) the ROSbridge network to communicate the gaze pose to the remote site - $10ms$ ($100\ Hz$).
3. PSNR metric: The reduction of points results in degradation of the visual quality on the peripheral region when rendered to the user. To quantify this, the *Point-to-Point* Peak Signal-to-Noise Ratio based geometry quality metric is a frequently used measure of distortion [DR17, GMRMT05]. It is deemed insufficient though, as it does not consider the underlying surfaces represented by the point-clouds when estimating the distortion [TOF*17]. Further, it can be sensitive to size differences and noise when calculating the peak signal estimation. A new *volumetric density based* PSNR metric, is proposed, which utilizes two volumetric densities for the data under consideration: (1) the general volumetric density (proposed in CloudCompare [GM11]), computed using the number of neighbors $N^v_{\mathcal{P}}$ for each point $\mathfrak{p}$ in the point-cloud $\mathcal{P}$ that lie inside a spherical volume v, as seen in Eq. 5. Figure 9 visualizes the concept, where the foveated point-cloud shows higher density around the fovea region and lower density in the peripheral regions; and (2) its maximum volumetric density as the peak signal. For the peak signal, the volumetric density is calculated with the k-nearest neighbor approach, as seen in Eq. (6), to account for the distribution of the density across the point-cloud and avoid any skew in the values due to sensor noise.

$$\mathbf{vd} = \frac{1}{N_{\mathcal{P}}} \sum_{\forall \mathfrak{p} \in \mathcal{P}} \frac{N^v_{\mathcal{P}}}{\frac{4}{3} \cdot \pi \cdot R^3} \tag{5}$$

$$\begin{aligned} \mathbf{vd}^k_{\mathfrak{p} \in \mathcal{P}_1} &= \frac{1}{k} \sum_{i=1}^{k} \frac{N^v_{\mathcal{P}_1}}{\frac{4}{3} \cdot \pi \cdot R^3}, \\ \mathbf{vd}^{max}_{\mathcal{P}_1} &= \max_{\forall p \in P_1} \left(\mathbf{vd}^k_{\mathfrak{p}} \right) \end{aligned} \tag{6}$$

The value of $k = 10$ was found experimentally and the choice depends on the input dataset and the resolution, as data with more depth measurement errors will likely perform better when the value of k is higher. The value of radius R, for consistency, is estimated by averaging the voxel sizes across all the foveated regions in the F3 condition (Fig. 9). For the symmetric density difference calculation, for every point $\mathfrak{p}$ in the reference (original) point-cloud $\mathcal{P}_1$ (FREF), the closest corresponding point in the degraded cloud $\mathfrak{p}_{nn} \in \mathcal{P}_2$ (F0-F3) is found. The density

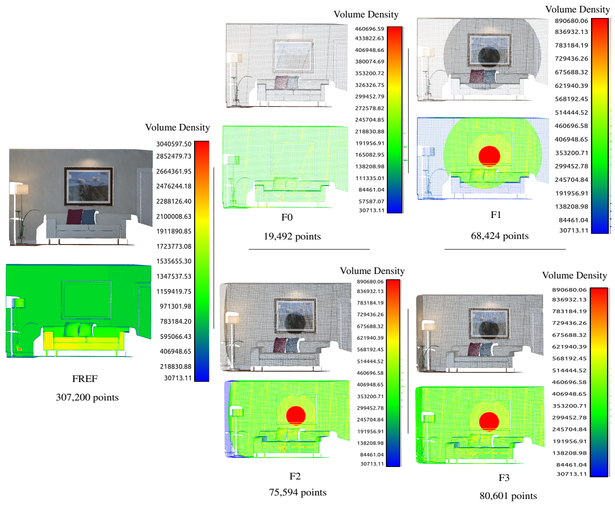


Figure 9: *The foveated conditions and the colour-coded map for the volumetric density estimation for the* ***LIV*** *dataset using Eq. (5) and plotted by CloudCompare [GM11] using radius of R = 0.019809.*

$\mathbf{vd}_{\mathcal{P}_1}$ and $\mathbf{vd}_{\mathcal{P}_2}$ are then estimated using Eq. (5). The density-based PSNR is calculated as a ratio of the maximum density of a reference point-cloud **FREF** to the symmetric root-mean-square (rms) difference in the general densities. Eq. (7) provides the equations; $N_{\mathcal{P}_1}$ is the number of points in region $\mathcal{P}_1$.

$$\begin{aligned} \mathbf{vd}^{rms}(\mathcal{P}_1, \mathcal{P}_2) &= \sqrt{\frac{1}{N_{\mathcal{P}_1}} \sum_{i=1}^{N_{\mathcal{P}_1}} \left[\mathbf{vd}^i_{\mathcal{P}_1} - \mathbf{vd}^i_{\mathcal{P}_2} \right]^2} \\ \mathbf{vd}^{sym}(\mathcal{P}_1, \mathcal{P}_2) &= max\left(\mathbf{vd}^{rms}(\mathcal{P}_1, \mathcal{P}_2)), \mathbf{vd}^{rms}(\mathcal{P}_2, \mathcal{P}_1) \right) \\ PSNR_{\mathrm{vd}} &= 10 \cdot log_{10} \frac{(\mathbf{vd}^{max}_{\mathcal{P}_1})^2}{(\mathbf{vd}^{sym}(\mathcal{P}_1, \mathcal{P}_2))^2} \end{aligned} \tag{7}$$

4. *Visual Quality of Experience* (QoE) user study: to assess the impact of the *FoReCast* framework on quality of experience, we put the following two research questions to guide the study (adapted from the research work [WRK*16]), **RQ1**: Can subjects differentiate between scenes with varying graphical contexts, streamed with and without the proposed system? and **RQ2**: How do different combinations of the foveated regions impact subjective quality of experiance? In this case, the stimulus is the real-time point-cloud (foveated vs. non-foveated) which is the independent variable and the ability to notice quality degradation is the dependent variable. Using the Double Stimulus Impairment Scale (DSIS) study approach [Int20] with the **LIV** dataset, subjects were first presented with the FREF condition, followed by a 3-second pause, and one of the altered conditions (F0-F3) following immediately after, in a randomized manner. Both FREF and altered conditions had 450 frames and were shown for 35 seconds before and after the 3 seconds pause. The subjects were then asked to rate the second presented

stimulus on a 5-point scale [Int20], on whether the alteration was: (5) imperceptible; (4) perceptible, but not annoying; (3) slightly annoying; (2) annoying; and (1) very annoying. The arithmetic mean opinion score (MOS) was calculated for each condition.

5. *Visual Motion Following* (VMF) user study: to assess the effect of peripheral quality loss on the tracking accuracy when visualizing moving objects in a dynamic remote scene, using the **BAL** dataset seen in Fig. 8-a. Subjects were asked to fixate their gaze on the moving balloon and follow its motion trajectory as closely as possible; the balloon trajectory and eye-gaze trajectory in 3D were used to calculate the trajectory root-mean-square-error (RMSE).

For the user study, 24 subjects (9 females and 15 males) participated in the study, aged 21 to 35 years. All subjects had a 20/20 or corrected vision, and the eye-tracker was calibrated for all subjects. Based on the ITU-T [Int20] recommendation, subjects were made familiar with the experimental setup using the **OFF** dataset, with the VR headset and the gesture controller devices. Each subject performed two trials for each test condition, within the QoE and VMF experiments. In addition, the experiment was designed as a between-subjects study to remove a carryover effect from one experimental condition to another and the experimental conditions were presented in a pseudo-random order.

7. Results and Discussion

Following a recommendation by [BMFE19], five randomized HMD positions with varying distances to the center of the datasets were used for the objective metrics evaluation. Four hundred frames were tested for each HMD position from each dataset, for the objective metrics.

A. Data transfer rate: Table 2 shows the relative reduction in number of points per frame and table 3 reports the average bandwidth required for streaming and the relative percentage reduction in bandwidth as compared to the FREF condition. MBytes/sec. Whereas The mean bandwidth required for the F1 condition gives an average 61% reduction as compared with FREF. The numbers are similar for the F2 condition, an average 61% reduction, and F3 offers a lower 55% reduction. Statistical t-test analysis showed that these reductions are significant at 95% CI (p-values<<0.05). Within the 3 conditions, although F1 is the most advantageous, the difference among the 3 reductions is not statistically significant (p-value = 0.3). On the other hand, as expected, the foveation conditions perform worse than the F0 condition, which offers the highest bandwidth reduction, up to 81%.

B. Latency reduction: The mean latency values for framework are listed in Table. 3. Again, the foveation conditions offer between 60% (F3) and 67% (F1) speedup over the FREF condition. However, they also perform worse than the F0 condition, between 20% and 35% slower. The speedups (and slowdowns) are statistically significant, p-values<<0.05. A more detailed system component level evaluation is seen in Table 4. The most time-consuming elements are related to data conversion and compression. As expected, the numbers show an upward trend from F0 to FREF. It is noted that this trend is not linear - latency increases at a greater rate with increasing point-cloud density.

Table 2: *Mean number of points per frame.*

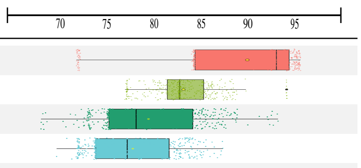
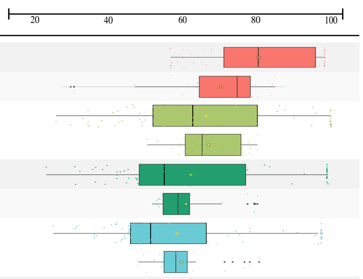
	BAL	OFF	LIV	Reduction(%)
F0	39K	18K	20K	
F1	45K	51K	49K	
F2	65K	54K	69K	
F3	66K	56K	73K	
FREF	252K	307K	307K	

Table 3: *Compressed bandwidth (MBytes/sec; top row) and latency (ms; bottom row)*

	BAL	OFF	LIV	Reduction(%)
F0	0.78	0.49	0.25	
	198	196	190	
F1	0.80	1.02	0.50	
	226	224	223	
F2	0.97	1.03	0.55	
	235	240	242	
F3	1.03	1.22	0.74	
	256	256	257	
FREF	1.82	1.78	1.32	
	562	622	618	

C. PSNR metric: Figure 10 illustrates the volumetric density based PSNR metric, that helps objectively discriminate among the test conditions in terms of the costs they impose on the visual quality. In all cases, the F0 PSNR is significantly worse (p-value<<0.05), which negates the bandwidth and latency advantages it offers. The foveation conditions offer progressively better PSNR values, averaging 69.5 dB (F1), 70 dB (F2), and 71.6 dB (F3), over the 4 datasets. The F3 PSNR is significantly better than F1 (p-value<<0.05), but not F2 (p-value = 0.64).

D. QoE metric: Figure 11 shows the MOS, averaged over the 24 subjects. It is seen that all three foveation conditions have their MOS > 3. For the F1 and F2 conditions, the foveation is certainly perceptible, but it may not hinder the users' experience, since the perceived degradation is only 'slightly annoying' (F1) or 'not annoying'. With an MOS > 4, the F3 condition shows that subjects are not able to easily perceive the degradation, and even if they do, it is 'not annoying'. The F0 condition has an MOS < 3, implying

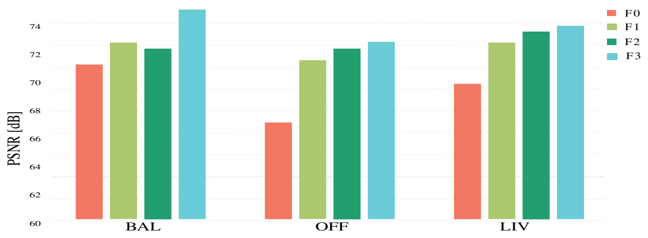


Figure 10: *Volumetric density based PSNR for all experimental conditions and data-sets.*

Table 4: *Sample comparison of averaged system components execution times per frame for the* ***OFF*** *and* ***LIV*** *datasets.*

	Component	OFF F_0	OFF F_1	OFF F_2	OFF F_3	OFF F_{REF}	LIV F_0	LIV F_1	LIV F_2	LIV F_3	LIV F_{REF}
Remote	RGB-D reader	4	4	4	4	4	4	4	4	4	4
	Partitioning	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3
	Conversion	46	59	43	51	56	54	53	47	46	56
	Sampling	21	22	23	25	0	30	21	22	22	0
	Encoding	60	88	114	111	319	104	73	109	119	396
User	Decoding	25	36	46	47	121	47	38	59	58	158
	Conversion	0.7	1.3	1.8	2	7	1	1.2	1.6	1.6	6
	Rendering	18	18	18	18	18	14	14	14	14	14
Total(ms)		175	228	250	258	525	254	204	257	265	634

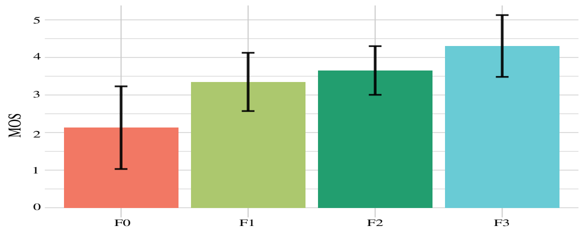


Figure 11: *Mean MOS and SD (error bars) for the QoE metric.*

the degradation can be annoying for subjects, which further negates the benefits it offers on the other metrics.

E. VMF metric: After removing outliers beyond the 5^{th} and 95^{th} percentile thresholds (using the MATLAB "quantile" function), 4595 "log-normally" distributed data points, i.e., error calculated between the ground-truth and eye-gaze trajectories, were available in each condition over all the subjects. The Two-way Students' T-test was used to compare the means (of the log) of the data distributions across the conditions. As seen in Fig. 12, the results reveal an "inverted bell-curve" phenomenon across F0-to-FREF. The smallest error is for condition F2. The T-test showed that the improvement offered in the F2 condition is statistically significant (p-values$<<$0.0), when compared to all other conditions at the 95% CI.

F. Discussion: The five metrics analyzed here offer a cost-benefit understanding of the tested conditions, i.e., the benefits in bandwidth and latency vs. the costs in PSNR, QoE, and VMF. For instance, the F0 condition, as expected, offers the most benefit for bandwidth and latency, but the costs in PSNR, QoE, and VMF are the highest. Whereas the FREF condition is the ideal in terms of PSNR and QoE, the overall analysis demonstrates that the foveated conditions together provide the optimal cost-benefit ratio, as compared to both F0 and FREF conditions. The perceived degradations are seen to not significantly impact QoE. A deeper analysis shows that the F3 condition performs significantly better in the benefit metrics, while its costs are not significantly worse than FREF. As expected, the F1 condition falls at the lower end within the 3 conditions, but still offers significantly higher benefit on latency and bandwidth. The F2 condition offers a good cost-benefit compromise between the two conditions. Here, the flexibility of the *FoReCast* framework offers a key advantage. Real-time usage requirements and a user-selectable approach can allow users to choose among the three conditions and switch among them as required.

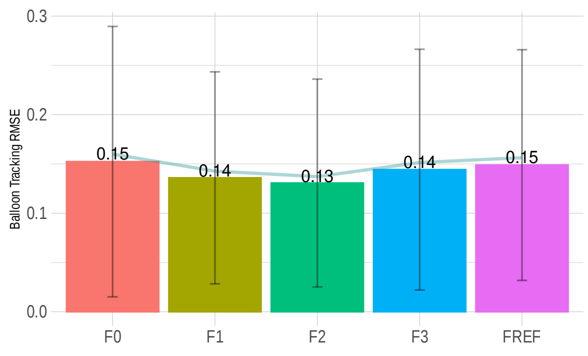


Figure 12: *VMF RMSE and SD (error bars) for the* ***BAL*** *dataset.*

G. Limitations of the FoReCast framework: As noted earlier, we have not found any commercial or research approaches that focus on foveation of real-time remotely captured 3D point-cloud rendering and unicasting in VR. Therefore, evaluating the proposed approach vis-á-vis state-of-the-art was not possible. The devised metrics helped understand the impact of *FoReCast* on *IRT*. Even so, the framework suffers two important limitations:

1. Bottleneck: in the process of conversion, partitioning, and sampling, especially at high resolutions, as seen in Tables 2 and 4. The implemented GPU and CPU parallelization, through OpenMP and CUDA helps significantly. But this also requires more computing resources at the remote site. The framework has to evolve in a way as to offload this processing either to the user site or in an independent cloud server, so as to allow thin remote clients.
2. Distortion and aliasing: introduced due to discontinuities at region boundaries and over-sampling in the peripheral regions. These are more noticeable in the peripheral regions, and can be distracting, although the QoE score does not reflect that. Future studies will include anti-aliasing techniques such as the work proposed in [SKW19].

8. Conclusions and Future Work

This work presented a novel foveated remote rendering and streaming pipeline, the *FoReCast* framework, that utilizes the acuity fall-off in human visual systems. The approach facilitates the processing, transmission, and rendering of dense point-clouds while simultaneously reducing throughput requirements and maintaining the visual experience for the users. The main contribution of this paper shows that by exploiting the human visual system, remotely acquired dense point-cloud data can be presented to a user in a foveated way. Validation experiments demonstrated significant reductions in latency and throughput, higher than 60% in both. The PSNR metric allowed to discriminate among the foveated conditions for objective quality assessment, showing the overall advantages of the framework. Preliminary user trials demonstrated that foveated rendering of dense point-clouds in VR does not have a significant negative impact on quality of experience, in static or dynamic scenes.

Future investigations will focus on addressing the limitations in the approach. A comprehensive user study will help situate the *FoReCast* framework in terms of usability and user experience in real-world environments.

References

[Bej96] BEJCZY A. K.: Virtual Reality in Robotics. In *1996 IEEE Conference on Emerging Technologies and Factory Automation. ETFA '96* (1996), pp. 7–15 vol.1. 2

[BMFE19] BRUDER V., MÜLLER C., FREY S., ERTL T.: On evaluating runtime performance of interactive visualizations. *IEEE transactions on visualization and computer graphics 26*, 9 (2019), 2848–2862. 7

[BSB*19] BRUDER V., SCHULZ C., BAUER R., FREY S., WEISKOPF D., ERTL T.: Voronoi-Based Foveated Volume Rendering. In *EuroVis (Short Papers)* (2019), pp. 67–71. 2, 3

[Cha21] CHARLTON A.: What is Foveated Rendering? Explaining the VR technology key to lifelike realism, May 11 2021. [Accessed: 05-Sep-2021]. 2

[CHB07] CHEN J. Y. C., HAAS E. C., BARNES M. J.: Human Performance Issues and User Interface Design for Teleoperated Robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 37*, 6 (Nov 2007), 1231–1245. 1

[CJO*17] CRICK C., JAY G., OSENTOSKI S., PITZER B., JENKINS O. C.: ROSbridge: ROS for non-ROS users. In *Robotics Research*, Christensen H., Khatib O., (Eds.). Springer, 2017, pp. 493–504. 5

[CR74] COWEY A., ROLLS E. T.: Human Cortical Magnification Factor and its Relation to Visual Acuity. *Experimental Brain Research 21* (1974), 447–454. 3

[DR17] 3DG M., REQUIREMENTS: *Call for Proposals for Point Cloud Compression V2*. Tech. rep., MPEG 3DG and Requirements, Hobart, AU, 2017. 6

[FCG*17] FAIRCHILD A. J., CAMPION S. P., GARCÍA A. S., WOLFF R., FERNANDO T., ROBERTS D. J.: A Mixed Reality Telepresence System for Collaborative Space Operation. *IEEE Transactions on Circuits and Systems for Video Technology 27* (2017), 814–827. 2

[GFD*12] GUENTER B., FINCH M., DRUCKER S., TAN D., SNYDER J.: Foveated 3D Graphics. *ACM Transactions on Graphics (TOG) 31*, 6 (2012), 1–10. 2, 3

[GM11] GIRARDEAU-MONTAUT D.: Cloudcompare-open source project. *OpenSource Project 588* (2011). 6

[GMRMT05] GIRARDEAU-MONTAUT D., ROUX M., MARC R., THIBAULT G.: Change Detection on Point Cloud Data Acquired with a Ground Laser Scanner. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 36* (01 2005). 6

[Hen05] HENDRICKSON A.: Organization of the adult primate fovea. In *Macular Degeneration*, Penfold P. L., Provis J. M., (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 1–23. doi:10.1007/3-540-26977-0_1. 2

[HPKG08] HUANG Y., PENG J., KUO C.-C. J., GOPI M.: A generic scheme for progressive point cloud coding. *IEEE Transactions on Visualization and Computer Graphics 14*, 2 (2008), 440–453. 2

[HWMD14] HANDA A., WHELAN T., MCDONALD J. B., DAVISON A. J.: A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM. In *IEEE Intl. Conf. on Robotics and Automation, ICRA* (Hong Kong, China, May 2014), pp. 1524–1531. 5

[Int20] INTERNATIONAL TELECOMMUNICATION UNION: *Recommendation ITU-T P.919: Subjective test methodologies for 360º video on head-mounted displays*. ITU, 2020. 6, 7

[KBR*12] KAMMERL J., BLODOW N., RUSU R. B., GEDIKLI S., BEETZ M., STEINBACH E.: Real-time Compression of Point Cloud Streams. In *2012 IEEE International Conference on Robotics and Automation* (2012), IEEE, pp. 778–785. 2

[Kra19] KRAFT V.: *Efficient Rendering of Massive and Dynamic Point Cloud Data in State-of-the-art Graphics Engines*. Master's thesis, Universität Bremen, 2019. 5

[KSE*17] KRUPKE D., STARKE S., EINIG L., ZHANG J., STEINICKE F.: Prototyping of Immersive HRI Scenarios. In *Proc. 20th Intl. Conf. on CLAWAR 2017* (Oct. 2017), pp. 537–544. 2

[KYIS14] KAMEZAKI M., YANG J., IWATA H., SUGANO S.: A basic framework of virtual reality simulator for advancing disaster response work using teleoperated work machines. *Journal of Robotics and Mechatronics 26*, 4 (2014), 486–495. doi:10.20965/jrm.2014.p0486. 1

[LFR18] LIPTON J. I., FAY A. J., RUS D.: Baxter's Homunculus: Virtual Reality Spaces for Teleoperation in Manufacturing. *IEEE Robotics and Automation Letters 3*, 1 (2018), 179–186. 2

[MB97] MILGRAM P., BALLANTYNE J.: Real World Teleoperation via Virtual Environment Modeling. In *Intl. Conf. on Artificial Reality and Tele-existence (ICAT'97)* (Dec. 3-5 1997). 2

[MBC16] MEKURIA R., BLOM K., CESAR P.: Design, implementation, and evaluation of a point cloud codec for tele-immersive video. *IEEE Transactions on Circuits and Systems for Video Technology 27*, 4 (2016), 828–842. 2, 5, 6

[MF11] MAIMONE A., FUCHS H.: Encumbrance-free telepresence system with real-time 3d capture and display using commodity depth cameras. In *10th IEEE International Symposium on Mixed and Augmented Reality* (2011), IEEE, pp. 137–146. 2

[MK16] MOSSEL A., KRÖTER M.: Streaming and Exploration of Dynamically Changing Dense 3D Reconstructions in Immersive Virtual Reality. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* (2016), IEEE, pp. 43–48. 1, 2

[MRWB03] MEEHAN M., RAZZAQUE S., WHITTON M., BROOKS F.: Effect of Latency on Presence in Stressful Virtual Environments. In *IEEE Virtual Reality, 2003. Proceedings.* (2003), pp. 141–148. 1

[NMB*21] NACERI A., MAZZANTI D., BIMBO J., TEFERA Y. T., PRATTICHIZZO D., CALDWELL D. G., MATTOS L. S., DESHPANDE N.: The *Vicarios* Virtual Reality Interface for Remote Robotic Teleoperation. *Journal of Intelligent & Robotic Systems 101*, 80 (2021). 2

[OERF*16] ORTS-ESCOLANO S., RHEMANN C., FANELLO S., CHANG W., KOWDLE A., DEGTYAREV Y., KIM D., DAVIDSON P. L., KHAMIS S., DOU M., TANKOVICH V., LOOP C., CAI Q., CHOU P. A., MENNICKEN S., VALENTIN J., PRADEEP V., WANG S., KANG S. B., KOHLI P., LUTCHYN Y., KESKIN C., IZADI S.: Holoportation: Virtual 3D Teleportation in Real-Time. In *29th Annual Symposium on User Interface Software and Technology (UIST)* (New York, NY, USA, 2016), Association for Computing Machinery, p. 741–754. 1, 2

[PBAR15] PEPPOLONI L., BRIZZI F., AVIZZANO C. A., RUFFALDI E.: Immersive ROS-integrated Framework for Robot Teleoperation. In *2015 IEEE Symposium on 3D User Interfaces (3DUI)* (2015), pp. 177–178. 2

[PBL*19] PALAZZOLO E., BEHLEY J., LOTTES P., GIGUÈRE P., STACHNISS C.: ReFusion: 3D Reconstruction in Dynamic Environments for RGB-D Cameras Exploiting Residuals. In *IEEE/RSJ IROS* (2019). 5

[PSK*16] PATNEY A., SALVI M., KIM J., KAPLANYAN A., WYMAN C., BENTY N., LUEBKE D., LEFOHN A.: Towards Foveated Rendering for Gaze-tracked Virtual Reality. *ACM Transactions on Graphics (TOG) 35*, 179 (2016), 1–12. 2

[QCF*19] QUINN N., CSINCSIK L., FLYNN E., CURCIO C. A., KISS S., SADDA S. R., HOGG R., PETO T., LENGYEL I.: The clinical relevance of visualising the peripheral retina. *Progress in Retinal and Eye Research 68* (2019), 83–109. doi:https://doi.org/10.1016/j.preteyeres.2018.10.001. 2, 3

[RWF*20] ROSEN E., WHITNEY D., FISHMAN M., ULLMAN D., TELLEX S.: Mixed Reality as a Bidirectional Communication Interface for Human-Robot Interaction. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2020), pp. 11431–11438. doi:10.1109/IROS45743.2020.9340822. 1, 2

[San17] SANDERS C.: *Practical packet analysis: Using Wireshark to solve real-world network problems*. No Starch Press, 2017. 6

[SGEM16] STENGEL M., GROGORICK S., EISEMANN M., MAGNOR M.: Adaptive Image-space Sampling for Gaze-contingent Real-time Rendering. *Computer Graphics Forum 35*, 4 (2016), 129–139. 2

[SK06] SCHNABEL R., KLEIN R.: Octree-based Point-Cloud Compression. *Eurographics Symposium on Point-Based Graphics 6* (2006), 111–120. 2

[SKH*19] STOTKO P., KRUMPEN S., HULLIN M. B., WEINMANN M., KLEIN R.: SLAMCast: Large-Scale, Real-Time 3D Reconstruction and Streaming for Immersive Multi-Client Live Telepresence. *IEEE Transactions on Visualization and Computer Graphics 25*, 5 (2019), 2102–2112. 1

[SKS*19] STOTKO P., KRUMPEN S., SCHWARZ M., LENZ C., BEHNKE S., KLEIN R., WEINMANN M.: A VR System for Immersive Teleoperation and Live Exploration with a Mobile Robot. In *IEEE/RSJ IROS* (Nov 2019), pp. 3630–3637. doi:10.1109/IROS.2012.6386012. 1, 2

[SKW19] SCHÜTZ M., KRÖSL K., WIMMER M.: Real-time continuous level of detail rendering of point clouds. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)* (2019), pp. 103–110. doi:10.1109/VR.2019.8798284.

[SNL20] STAUFFERT J.-P., NIEBLING F., LATOSCHIK M. E.: Latency and Cybersickness: Impact, Causes, and Measures. A Review. *Frontiers in Virtual Reality 1* (2020), 31. 1

[SNW*21] STEIN N., NIEHORSTER D. C., WATSON T., STEINICKE F., RIFAI K., WAHL S., LAPPE M.: A comparison of eye tracking latencies among several commercial head-mounted displays. *i-Perception 12*, 1 (2021), 2041669520983338. 2

[SRJ11] STRASBURGER H., RENTSCHLER I., JÜTTNER M.: Peripheral Vision and Pattern Recognition: A Review. *Journal of Vision 11*, 5 (2011), 13–13. 2, 3

[TLZ*18] TAN G., LEE Y.-H., ZHAN T., YANG J., LIU S., ZHAO D., WU S.-T.: Foveated Imaging for Near-eye Displays. *Optics Express 26*, 19 (2018), 25076–25085. 2

[TOF*17] TIAN D., OCHIMIZU H., FENG C., COHEN R., VETRO A.: Geometric Distortion Metrics for Point Cloud Compression. In *2017 IEEE International Conference on Image Processing (ICIP)* (2017), pp. 3460–3464. doi:10.1109/ICIP.2017.8296925. 6

[TSLM17] THEOFANIDIS M., SAYED S. I., LIOULEMES A., MAKEDON F.: VARM: Using Virtual Reality to Program Robotic Manipulators. In *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments - PETRA 2017* (2017), pp. 215–221. 2

[Wey58] WEYMOUTH F. W.: Visual Sensory Units and the Minimal Angle of Resolution. *American Journal of Ophthalmology 46*, 1 (1958), 102–113. 2, 3

[WLSM*15] WHELAN T., LEUTENEGGER S., SALAS-MORENO R., GLOCKER B., DAVISON A.: ElasticFusion: Dense SLAM without a Pose Graph. In *Robotics: Science and Systems* (2015). 3

[WRK*16] WEIER M., ROTH T., KRUIJFF E., HINKENJANN A., PÉRARD-GAYOT A., SLUSALLEK P., LI Y.: Foveated real-time ray tracing for head-mounted displays. *Computer Graphics Forum 35* (10 2016), 289–298. doi:10.1111/cgf.13026. 6

[WRU*18] WHITNEY D., ROSEN E., ULLMAN D., PHILLIPS E., TELLEX S.: ROS Reality: A Virtual Reality Framework Using Consumer-Grade Hardware for ROS-Enabled Robots. In *Proc. IEEE IROS 2018* (Oct. 1-5 2018), pp. 5018–5025. 2

[WSKK20] WEINMANN M., STOTKO P., KRUMPEN S., KLEIN R.: Immersive VR-based Live Telepresence for Remote Collaboration and Teleoperation. In *Wissenschaftlich-Technische Jahrestagung der DGPF* (2020), pp. 391–399. 2

[ZSG*18] ZOLLHÖFER M., STOTKO P., GÖRLITZ A., THEOBALT C., NIESSNER M., KLEIN R., KOLB A.: State of the Art on 3D Reconstruction with RGB-D Cameras. *Computer Graphics Forum 37*, 2 (2018), 625–652. 1