# Realtime 3D 360-Degree Telepresence With Deep-Learning-Based Head-Motion Prediction

Tamay Aykut, *Student Member, IEEE*, Jingyi Xu, *Student Member, IEEE*, and Eckehard Steinbach, *Fellow, IEEE*

***Abstract*— The acceptance and the dissemination of 360° telepresence systems are severely restricted by the appearance of motion sickness. Such systems consist of a client-side, where the user wears a head-mounted display, a server-side, which provides a 3D 360° visual representation of the remote scene, and a communication network in between. Due to the physically unavoidable latency, there is often a noticeable lag between head motion and visual response. If the sensory information from the visual system is not consistent with the perceived ego-motion of the user, and the emergence of visual discomfort is inevitable. In this paper, we present a delay-compensating 3D 360° vision system, which provides omnistereoscopic vision and a significant reduction of the perceived latency. We formally describe the underlying problem and provide an algebraic description of the amount of achievable delay compensation both for perspective and fisheye camera systems, considering all the three degrees of freedom. Furthermore, we propose a generic approach that is agnostic to the underlying camera system. In addition, a novel deep-learning-based head motion prediction algorithm is presented to further improve the compensation rate. Using the naive approach, where no compensation and prediction is applied, we obtain a mean compensation rate of 72.8% for investigated latencies from 0.1 to 1.0 s. Our proposed generic delay-compensation approach, combined with our novel deep-learning-based head-motion prediction approach, manages to achieve a mean compensation rate of 97.3%. The proposed technique also substantially outperforms prior head-motion prediction techniques, both for traditional and deep learning-based methods.**

***Index Terms*— 3D vision, deep learning, telepresence, remote reality, 3D 360-degree vision, omnistereoscopic vision.**

## I. Introduction

TELEPRESENCE systems such as the one shown in Fig. 1 allow a user to immerse herself/himself into a remote environment. The telerobot (here our MAVI platform [1]) is equipped with sensors which capture visual and auditory information about the distant space. Video and audio signals are exchanged over a communication network. Depending on the distance between the user and the operator, such communication networks introduce inevitable delays, which do not only lessen the immersive effect, but are also detrimental to the visual comfort. The amount of time needed to mirror the user's head-motion and display it onto the screen is denoted as the Motion-to-Photon (M2P) latency. If the M2P delay exceeds a certain threshold [2], the user will suffer from motion sickness and is prone to terminate the telepresence session in progress [3]–[5]. It is crucial that the sensory impressions from the visual system, the vestibular system, and the non-vestibular proprioceptors are in accordance with the user's perceived ego-motion as well as the user's expectations based on prior experiences [6]. To adhere to this premise, one feasible approach is to capture the entire scene at once and provide the user with instant visual feedback for the desired heading by changing the Region of Interest (ROI). In this way, the user is able to freely explore the distant environment while the M2P latency is kept low. There are already many consumer products that acquire and stream monoscopic 360° videos. 3D 360° vision, instead, still poses major challenges both in terms of the acquisition and streaming. Stereoscopic (3D) panoramas are highly favored as they provide separate image content from different vantage points for each eye, and thus allow the perception of depth information. 3D vision enhances

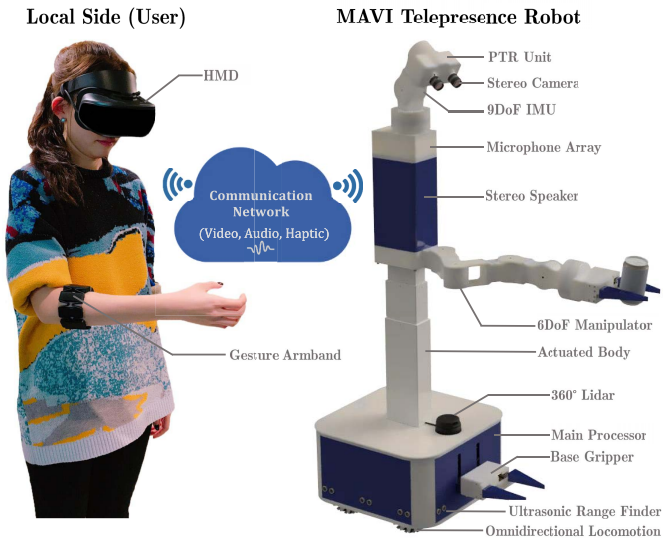


Fig. 1. Experimental telepresence setup, where the user controls our MAVI robot in a remote environment [7]. At the top of the MAVI platform a stereo camera system is combined with a 3DoF pan-tilt-roll unit to provide stereoscopic views in all possible directions.

Manuscript received August 6, 2018; revised December 27, 2018; accepted January 29, 2019. Date of publication February 4, 2019; date of current version March 11, 2019. This paper was recommended by Guest Editor T. Stockhammer. *(Corresponding author: Tamay Aykut.)*

The authors are with the Department of Electrical and Computer Engineering, Technical University of Munich, 80333 Munich, Germany (e-mail: tamay.aykut@tum.de; jingyi.xu@tum.de; eckehard.steinbach@tum.de).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JETCAS.2019.2897220

the level of immersion and can greatly improve task performance especially for indoor applications, as the human stereo vision is limited to 19m [8]. Previous works have shown that stereoscopic vision exhibits great advantages over monoscopic viewing [9], [10]. Under stereoscopic conditions, tasks were performed faster and with fewer errors [9], [10]. Prior art deployed catadioptric and multi-camera systems to create stereoscopic 360° panorama videos [11]–[16]. Beside the fact that most of them are bulky and expensive, the majority of these systems can also not guarantee realtime capability, a large stereoscopic budget, or depth perception in every viewing direction simultaneously.

In this work, we focus exclusively on the visual perception of our telepresence system and apply a standard two-camera stereo video setup and augment it with a three Degrees-of-Freedom (DoF) electro-mechanical Pan-Tilt-Roll Unit (PTR-U). Such a system is lean, realtime-capable, allows binocular vision in every viewing direction, and provides a large stereoscopic budget. Despite its valuable benefits, a PTR-U-based stereo-camera setup introduces further challenges, which need to be overcome. Among other things, these involve an optimal and smooth control as well as an additive delay originating from the hardware components. The hardware delay in particular increases the M2P latency and hence the mismatch between ego-motion and visual feedback, resulting in deteriorated visual comfort. To address this issue, we combine the advantages of a PTR-U-based stereo camera system and an immediately accessible, stereoscopic 360° visual representation of the remote scene by adopting the concept of the Delay Compensating Vision System (DCVS) [17], [18]. In [7], we conducted a subjective pilot study that revealed the superior performance of the DCVS compared to the naive method where no compensation is applied at all. Deploying the delay compensation approach helps to reduce motion sickness while maintaining the feeling of presence [7]. Conceptionally, the DCVS uses cameras with a larger field of view ($fov_c$) than the visual field of the user. The residual image content is leveraged for local delay compensation. A detailed description is provided in Section III. The DCVS concept was initially presented for perspective cameras where delay was compensated for horizontal motions. In [18], the DCVS concept was adopted and tailored particularly for fisheye cameras. In this work, we extend the DCVS for perspective cameras to 3DoF, refine the algebraic description of [18], and propose a generic, mathematically-founded delay-compensation approach, which can be used for both perspective and fisheye cameras. We further present the generic compensation rates as a metric to convey the achievable level of delay compensation. We improve the level of delay compensation by applying head-motion prediction using a novel deep neural network that is based on stacked Gated Recurrent Units (GRU) and convolution components to extract the most distinct features at different granularities. We trained our network on normalized differences rather than on absolute orientation values for generalization. The key contributions of this paper can be stated as follows:

- The DCVS approach in [17] compensates the delay for horizontal orientations using perspective cameras. We extend this approach to all 3DoFs and provide a mathematically-founded algebraic model to compute the degree of achievable delay-compensation.
- We further propose a novel generic delay-compensation model, which is agnostic to the underlying camera system and works for both perspective and fisheye cameras. We provide an algebraic specification of the delay that can be compensated for given parameters.
- The level of accessible latency compensation is additionally improved by a head-motion prediction algorithm that is based on a novel deep neural network, which shows superior performance compared to prior art. We use qualitative measures as well as an independent dataset to evaluate our proposed approach.

## II. Related Work

Capturing and streaming the entire (stereoscopic) 360° visual representation of the remote environment allows the user to casually explore the scene with a low M2P latency. Rather than waiting for the updated image frame, the ROI can be adjusted accordingly. To provide the user with an immersive 3D perception of the remote scene, an omnistereoscopic snapshot of the distant environment is highly preferred. Prior art shows that the flawless acquisition of stereoscopic full panorama videos in realtime remains unsolved. Promising approaches deploy either sequential acquisition [19], [20], catadioptric [11], [21], or multi-camera systems [12]–[16], [22]. For sequential acquisition, two cameras (usually line cameras) are mounted on a rig at a certain distance and spun to capture images from different vantage points for any viewing direction. Based on the concept of concentric mosaics [23], the sequential gathering results in high-quality omnistereoscopic videos. Due to its sequential nature, it is not applicable to dynamic environments and comes with a large overhead, which makes it challenging for embedded systems. Catadioptric systems, instead, can be used for dynamic environments. Aggarwal *et al.* [11] proposed a light-weight catadioptric system that combined a "coffee filter mirror" with a single camera to produce stereoscopic panoramas. Weissig *et al.* [21], instead, developed a bulky catadioptric system for large-scale applications. Multi-camera (or depth sensor) arrangements are a further method to capture stereoscopic video content [12]–[16], [22]. The underlying concept is often inspired by concentric mosaics, view interpolation, or depth-image-based rendering. Instead of spinning the cameras, multiple cameras are applied. Catadioptric and multi-camera systems have in common that they require a stitching process after capturing the imagery. In the absence of sufficient features in the scene, the stitching process is, however, deemed to be error-prone. Visualizing erroneous imagery on a Head-Mounted-Display leads to a magnification of the distorted footage and thus results in negative implications for the Quality of Experience (QoE), especially as

the content is shown to the eyes of the user from a very small distance. Besides that, these systems are characterized by a small stereoscopic budget as the inter-ocular distance is tightly bound to the mirror or multi-camera rig design. Current solutions further require costly peripheral equipment for the high computational demands, show limited realtime abilities, and can often provide only a constrained area of depth perception. The proposed generic DCVS does not need any stitching process and is hence free from stitching artifacts. Through the electro-mechanical PTR-U, we are able to provide 3D perception in every viewing direction.

Sending two high-resolution full-panorama videos, preferably at a high frame rate, does not only claim large portions of the communication capacity but also turns out to be redundant, as the user can never consume the whole image at once. Viewport-adaptive streaming approaches are investigated in literature to find the optimal trade-off between QoE and available resources [24]. Viewport-adaptive streaming is a top-down approach that takes for granted that a complete 360° visual representation is available for streaming. A subset of the entire 360° video content with an additional buffer margin is then selected, which is to be sent to the client. The design and the amount of the buffer size remains constant and lacks a concrete mathematical description. It is unclear, how much buffer is needed for the delay in question. Moreover, current acquisition and streaming solutions are mainly limited to monoscopic footage due to the previously mentioned challenges of acquiring omnidirectional stereo videos. We propose a solution that tackles both the acquisition of omnidirectional stereo videos and the streaming thereof. While viewport-adaptive approaches mostly follow a one-to-many on demand streaming methodology, we ensure online (live) telepresence with an on demand visual feedback for one specific user. A comparison of latest advances in viewport adaptive streaming and our proposed approach is a matter of further study. Our approach creates omnistereoscopic *vision on demand* in realtime with less overhead. We also provide a thorough algebraic description of the underlying problem. The resulting generic compensation rate can be leveraged (also for viewport-dependent streaming) to establish optimization techniques to determine the optimal required buffer area for the underlying conditions, such as the respective delay, the available image content, the size of the user's viewport, etc.

Head-motion prediction is considered as a valid and well studied approach to decrease the M2P latency. The applications of head-motion prediction are manifold. In 360° video streaming, state-of-the-art methods try to find the optimal portions of the image that should be sent in high quality, so as not to allocate precious transmission rates to regions of the image, which are not even shown to the user [25]. In former times, prediction methods were also investigated to compensate the local lag between head motion and display response [2], [4], [26]–[28]. Applied techniques were either based on regression methods (e.g (Weighted) Linear Regression) or on model-based extrapolation policies. Recent work investigated deep neural networks to forecast the head-motion after the present delay [7]. In this paper, we propose a novel deep network that is a composite of Gated Recurrent Units (GRUs) and convolution units. Our proposed network outperforms previous approaches and shows the best performance in compensating the delay.

## III. Delay Compensation

For an immersive telepresence experience, the user needs a 3D 360° visual representation of the remote scene. HMDs are often utilized to select the desired viewport more intuitively. To provide a delay-free and realistic visual impression, the Delay Compensation Vision System (DCVS) was first introduced in [17]. Rather than using a bulky and computationally complex multi-camera setup, a realtime capable stereo camera system is mounted on a Pan-Tilt Unit (PT-U) to mirror the user's head motion. Note that in our initial work, we captured only 2DoF of the user's head movement. In its raw setup, such a system introduces a severe lag between head motion and visual response, which leads to immediate visual discomfort. Beside the communication delay, we identified the following components, which contribute to the total latency $\tau$:

$$\tau = t_s + t_m + t_c + t_p + t_r + 2 \cdot t_n, \text{ with} \tag{1}$$

- $t_s$: Sampling rate of orientation sensor,
- $t_m$: Mechanical delay,
- $t_c$: Camera delay ($t_c = \frac{1}{f_c}$), with $f_c$ being the camera frame rate,
- $t_p$: Processing delay: rectification, encoding, and processing the captured images for transmission,
- $t_r$: Rendering delay: processes of extracting the frames from the received data, decoding them, and rendering them to the HMD,
- $t_n$: Network delay (one-way).

Eq. 1 shows that even for local teleoperation tasks there is an inherent delay (typically >200ms) that needs to be compensated. With the concept of DCVS, we are giving the user the impression of an instantaneous and uninterrupted visual telepresence experience. We decouple the head motion from the PTU movement and stream the user more imagery than is actually needed. By applying cameras with a larger field of view ($fov_{\mathrm{c}}$) compared to the size of the user's viewport own the HMD ($fov_{\mathrm{h}}$), we create a certain buffer zone. This residual image content is leveraged for local delay compensation until the updated frame at the desired head orientation arrives. In this way, we provide the user with an instantaneous visual response and arrange for the user's perception of ego-motion to be in accordance with the sensory inputs from the visual system, the vestibular system, and the non-vestibular proprioceptors mitigating the effect of cyber sickness [3]. In a traditional PTU-based camera system, the head motion would lead to a frozen image as the footage for the new head direction is not available immediately due to the latency present. We, however, first change the region of interest of our image data depending on the viewing direction, prior to updating the imagery. The degree of delay compensation is determined by the cache size and the difference between the HMD and the PT-U pan and tilt orientations. In [17], we proved the DCVS concept for pan rotations using perspective cameras. To increase the achievable level of delay

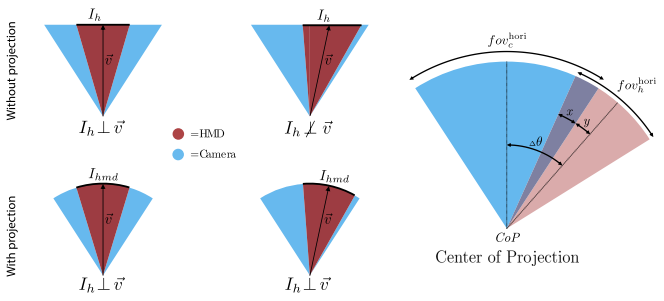


Fig. 2. Left: adapting the region-of-interest (ROI) is coupled with an expected perspective change. This issue is addressed by projecting the images onto a common cylinder. Right: if the user rotates the head excessively fast, a full compensation can no longer be guaranteed. The degree of compensation is expressed in the compensation rate and describes the ratio between available image content and the visual field of the user.

compensation, we next deployed equidistant fisheye cameras, considered 3DoFs thereof and provided a new metric for fisheye cameras that conveys the degree of achievable delay compensation [18]. In this paper, we extend the compensation particularly for perspective cameras to 3DoFs with a pan-tilt-roll unit, give a more detailed algebraic description for the delay compensation using equidistant fisheye cameras, and finally propose a novel generic approach, which is independent of the applied camera lens and can be applied to both perspective and fisheye cameras.

### A. Perspective Cameras

For the initial DCVS [17], the delay was compensated for horizontal head rotations, which are known to be the dominant ones. Capturing a larger horizontal field-of-view ($fov_c^{\text{hori}}$) than displaying to the user allows for a buffer $b^{\text{hori}}$, which is leveraged for local latency compensation [17]:

$$b^{\text{hori}} = \tfrac{1}{2}(fov_c^{\text{hori}} - fov_h^{\text{hori}}). \tag{2}$$

Fig. 2 demonstrates the simplified modes of operation. We adapt the ROI according to the new viewing direction $\vec{v}$ to enable instantaneous visual response. Simply changing the ROI for a new viewing direction does not consider the perspective change which occurs during head movements. Formally, this is associated with the fact that the new viewing direction $\vec{v}$ is not perpendicular to the available image plane $I_{\text{h}}$. We approach this issue by projecting the images onto a cylinder, where the viewing direction is perpendicular to the cylinder surface for every possible horizontal rotation (see Fig. 2).

Each pixel $(x, y)$ in the image $I_{\text{h}}$ is mapped onto the cylinder as follows:

$$x' = r \cdot \arctan\left(\frac{x}{r}\right), \tag{3}$$

$$y' = y \cdot \frac{r}{\sqrt{x^2 + r^2}}, \tag{4}$$

with $(x', y')$ being the projected cylinder coordinates and the radius $r = f$. The compensation rate $c_p$ was proposed as a metric to describe the achievable level of delay compensation for pan rotations and is described as the ratio of the available image content in the pan direction and the horizontal $fov_h^{\text{hori}}$ of the displayed image on the HMD [17]:

$$c_p = \frac{x}{fov_h^{\text{hori}}}, \tag{5}$$

with

$$x = \frac{fov_h^{\text{hori}}}{2} - y, \qquad y = \Delta\theta - \frac{fov_c^{\text{hori}}}{2}. \tag{6}$$

Inserting Eq. 6 into Eq. 5 yields the formula for the (pan) compensation rate $c_p$, which can be expressed as a function of the orientation change $\Delta\theta = |\theta_h - \theta_c|$ or the angular velocity in pan direction $\dot{\theta}_h$ [17]:

$$\begin{aligned} c_p &= \frac{\frac{1}{2}(fov_h^{\text{hori}} + fov_c^{\text{hori}}) - \Delta\theta}{fov_h^{\text{hori}}} \\ &= \frac{\frac{1}{2}(fov_h^{\text{hori}} + fov_c^{\text{hori}}) - \dot{\theta}_h \cdot \tau}{fov_h^{\text{hori}}}, \end{aligned} \tag{7}$$

where $c_p = 1$ corresponds to a full (100%) compensation:

$$c_p = \begin{cases} 1 & c_p \geq 1 \\ c_p & 0 \leq c_p < 1 \\ 0 & c < 0. \end{cases} \tag{8}$$

*1) Extension to 3DoF:* In the following, we extend the delay compensation ability of perspective cameras to 3DoFs. To take the perspective change into account, we project the image onto a sphere rather than a cylinder. The radius of the sphere $r_{\text{sph}} = f$ is set to be identical to the focal length of the camera. Each pixel pair $(x, y)$ is mapped onto the sphere with:

$$\begin{aligned} x' &= f \cdot \arctan\left(\frac{x}{f}\right), \\ y' &= f \cdot \arctan\left(\frac{y}{f}\right). \end{aligned} \tag{9}$$

The rotation vector $\vec{\xi}_{\{h,c\}} = [\theta_{\{h,c\}} \phi_{\{h,c\}} \psi_{\{h,c\}}]^{\text{T}}$ describes the angular rotation around the z-axis, x-axis and the y-axis for either the user's head motion (subscript: $h$) or the camera (subscript: $c$). $\Delta\vec{\xi} = [\Delta\theta, \Delta\phi, \Delta\psi]^{\text{T}}$ conveys the mismatch between the user's head state and the camera position with $\Delta\theta = |\theta_h - \theta_c|$, $\Delta\phi = |\phi_h - \phi_c|$, and $\Delta\psi = |\psi_h - \psi_c|$. The rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$ terms the combined rotation:

$$\mathbf{R} = \mathbf{R}_\psi \cdot \mathbf{R}_\phi \cdot \mathbf{R}_\theta. \tag{10}$$

While the individual rotations for pan and tilt are expressed as a revolution around their respective axes:

$$\mathbf{R}_\theta = \begin{bmatrix} \cos(\Delta\theta) & 0 & \sin(\Delta\theta) \\ 0 & 1 & 0 \\ -\sin(\Delta\theta) & 0 & \cos(\Delta\theta) \end{bmatrix}, \tag{11}$$

$$\mathbf{R}_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\Delta\phi) & -\sin(\Delta\phi) \\ 0 & \sin(\Delta\phi) & \cos(\Delta\phi) \end{bmatrix}, \tag{12}$$

we define the roll rotation to be a revolution around the optical axis as shown in Fig. 3. The direction vector $\vec{v}$ of the optical axis is defined by:

$$\vec{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \mathbf{R}_\phi \cdot \mathbf{R}_\theta \cdot \begin{bmatrix} 0 \\ 0 \\ f \end{bmatrix}. \tag{13}$$

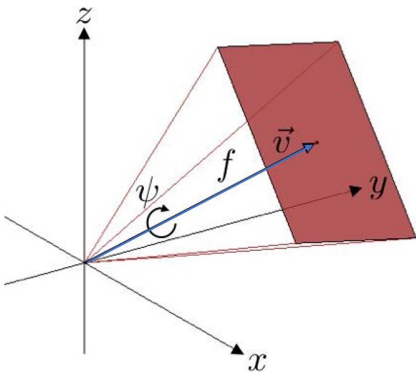


Fig. 3. In contrast to the pan and tilt rotations, which are described as rotations around their axes, the roll rotation is defined as the revolution around the optical axis. The optical axis can be any arbitrary axis through the origin and is described by its direction vector.

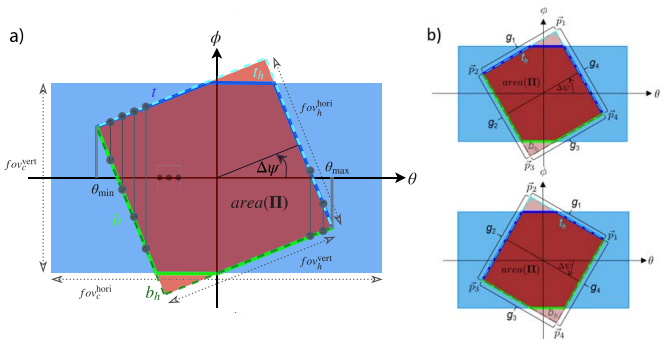


Fig. 4. Delay compensation for perspective cameras considering all 3DoFs. The level of delay compensation is determined within the spherical domain. a) Overview of the system description after projecting the images onto the common sphere. b) The direction of the roll rotation influences the computation of the auxiliary curves $t$ and $b$.

Therewith, the roll rotation matrix can be computed according to [29]:

$$\mathbf{R}_\psi = \cos(\Delta\psi) + (1-\cos(\Delta\psi)) \cdot \frac{\vec{v}\otimes\vec{v}}{|\vec{v}|^2} + \frac{\sin(\Delta\psi)}{|\vec{v}|} \cdot [\vec{v}]_\times = \begin{bmatrix} k_1\, v_x^2 + k_2 & k_1\, v_x v_y - k_3\, v_z & k_1\, v_x v_z + k_3\, v_y \\ k_1\, v_x v_y + k_3\, v_z & k_1\, v_y^2 + k_2 & k_1\, v_y v_z - k_3\, v_x \\ k_1\, v_x v_z - k_3\, v_y & k_1\, v_y v_z + k_3\, v_x & k_1\, v_z^2 + k_2 \end{bmatrix}, \tag{14}$$

with $k_1 = \frac{1-\cos(\Delta\psi)}{|\vec{v}|^2}$, $k_2 = \cos(\Delta\psi)$, and $k_3 = \frac{\sin(\Delta\psi)}{|\vec{v}|}$. After projecting the available image content onto the sphere, we change the ROI according to the current head movement. The amount of delay compensation is expressed by means of the available image content $area(\Pi)$ in proportion to the size of the image to be displayed:

$$c_{ptr} = \frac{area(\Pi)}{fov_h^{\text{hori}} \cdot fov_h^{\text{vert}}}. \tag{15}$$

Depending on how $fov_h^{\text{hori}}$ and $fov_h^{\text{vert}}$ are defined, the unit of $area(\Pi)$ is either in degrees or radian squared. When the desired $fov_h^{\text{hori}}$ is set, the vertical field of view of the HMD's image plane $fov_h^{\text{vert}}$ can be computed according to:

$$fov_h^{\text{vert}} = 2 \cdot \arctan\left(\frac{1}{ratio} \cdot \tan\left(\frac{fov_h^{\text{hori}}}{2}\right)\right), \tag{16}$$

where $ratio$ corresponds to the aspect ratio. $\Pi$ is a set that contains all available pixels in the image, which are accessible to be displayed on the HMD for the latest viewport. The computation of the $area(\Pi)$ for all 3DoFs is more challenging than for the 1D case, as a rectangular shape is not always preserved (see Fig. 4). After projecting the image content onto the sphere, we are able to calculate the overlapping area $\Pi$, which is available for visualization. Within the spherical coordinate system, we define two auxiliary curves $t$ and $b$ to compute $area(\Pi)$ as follows:

$$area(\Pi) = \int_{\theta_{min}}^{\theta_{max}} (t - b)\, d\theta. \tag{17}$$

We integrate over $\theta$ with a finite number of steps. At each step we assign the corresponding maximum value of $\phi$ to the top curve $t$ and likewise the minimum value thereof to the bottom curve $b$ (see Fig. 4a). If there is only one $\phi$ value available, we assign them to both curves. Depending on the present roll rotation, these curves can vary (see Fig. 4). To compute $t$ and $b$, we use two auxiliary curves $t_h$ and $b_h$, which are, analogously to $t$ and $b$, each constructed from two adjacent edge lines of the HMD frame:

$$t = \max\left(\min\left(t_h, \frac{fov_c^{\text{vert}}}{2}\right), -\frac{fov_c^{\text{vert}}}{2}\right), \tag{18}$$

$$b = \min\left(\max\left(b_h, -\frac{fov_c^{\text{vert}}}{2}\right), \frac{fov_c^{\text{vert}}}{2}\right). \tag{19}$$

Inserting these two equations into Eq. 17 results in:

$$area(\Pi) = \int_{\theta_{min}}^{\theta_{max}} \max\left(\min\left(t_h, \frac{fov_c^{\text{vert}}}{2}\right), -\frac{fov_c^{\text{vert}}}{2}\right) + - \min\left(\max\left(b_h, -\frac{fov_c^{\text{vert}}}{2}\right), \frac{fov_c^{\text{vert}}}{2}\right) d\theta. \tag{20}$$

Fig. 4b demonstrates how the determination of $t_h$ and $b_h$ changes depending on the current mismatch between the camera's roll rotation and that of the head. For $0° \leq \psi_c - \psi_h < 90°$ (see Fig. 4b (top)), $t_h$ and $t_b$ are defined as:

$$t_h = \begin{cases} g_1, & \theta \leq p_{1,\theta} \\ g_4, & \theta > p_{1,\theta}, \end{cases} \tag{21}$$

$$b_h = \begin{cases} g_2, & \theta \leq p_{3,\theta} \\ g_3, & \theta > p_{3,\theta}, \end{cases} \tag{22}$$

with $p_{i,\theta}$ being the first element of a corner point pair $\vec{p}_i = [p_{i,\theta}, p_{i,\phi}]^{\text{T}}$ of the HMD's image plane $\forall i \in \{1,2,3,4\}$. The bottom scheme of Fig. 4b illustrates the case where $-90° < \psi_c - \psi_h < 0°$. In that case, $t_h$ and $b_h$ are expressed as:

$$t_h = \begin{cases} g_2, & \theta \leq p_{2,\theta} \\ g_1, & \theta > p_{2,\theta}, \end{cases} \tag{23}$$

$$b_h = \begin{cases} g_3, & \theta \leq p_{4,\theta} \\ g_4, & \theta > p_{4,\theta}. \end{cases} \tag{24}$$

The lines $g_k$ can be computed with the following formula where $\vec{p}_i = [p_{i,\theta}, p_{i,\phi}]^T$ and $\vec{p}_j = [p_{j,\theta}, p_{j,\phi}]^T$ are the two

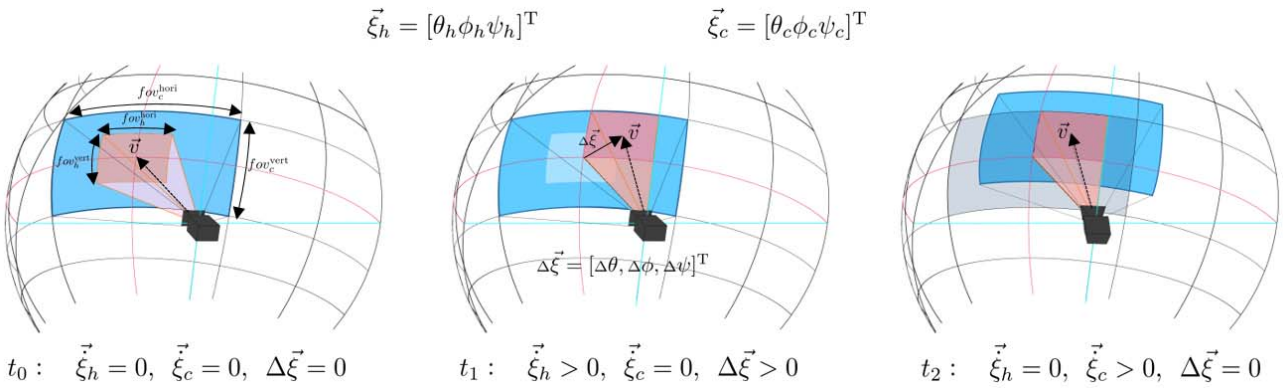


Fig. 5. The modes of operation of the generic delay-compensation approach. The captured images are first projected onto a common sphere to address the perspective change. The red viewport corresponds to the visual field of the user, the blue one visualizes the size of the captured image. If the user rotates the head, we change the ROI accordingly and leverage the residual image content for local delay compensation. In the meantime, the PTR-U position is adjusted accordingly and the updated image frame is sent.

end points of the line $g_k$ with $i, j, k \in \{1, 2, 3, 4\} \wedge i \neq j$

$$g_k = \frac{p_{j,\phi} - p_{i,\phi}}{p_{j,\theta} - p_{i,\theta}} \cdot \theta + \frac{p_{i,\theta} \cdot p_{j,\phi} - p_{j,\theta} \cdot p_{i,\phi}}{p_{i,\theta} - p_{j,\theta}}. \tag{25}$$

Therefore, the values of $\vec{p}_1$, $\vec{p}_2$, $\vec{p}_3$ and $\vec{p}_4$ are derived with:

$$\vec{p}_1 = \mathbf{R}_{\psi,2\mathrm{D}} \cdot \begin{bmatrix} fov_h^{\mathrm{hori}}/2 \\ fov_h^{\mathrm{vert}}/2 \end{bmatrix} + \begin{bmatrix} \Delta\theta \\ \Delta\phi \end{bmatrix}, \tag{26}$$

$$\vec{p}_2 = \mathbf{R}_{\psi,2\mathrm{D}} \cdot \begin{bmatrix} -fov_h^{\mathrm{hori}}/2 \\ fov_h^{\mathrm{vert}}/2 \end{bmatrix} + \begin{bmatrix} \Delta\theta \\ \Delta\phi \end{bmatrix}, \tag{27}$$

$$\vec{p}_3 = \mathbf{R}_{\psi,2\mathrm{D}} \cdot \begin{bmatrix} -fov_h^{\mathrm{hori}}/2 \\ -fov_h^{\mathrm{vert}}/2 \end{bmatrix} + \begin{bmatrix} \Delta\theta \\ \Delta\phi \end{bmatrix}, \tag{28}$$

$$\vec{p}_4 = \mathbf{R}_{\psi,2\mathrm{D}} \cdot \begin{bmatrix} fov_h^{\mathrm{hori}}/2 \\ -fov_h^{\mathrm{vert}}/2 \end{bmatrix} + \begin{bmatrix} \Delta\theta \\ \Delta\phi \end{bmatrix}. \tag{29}$$

The roll rotation matrix $\mathbf{R}_{\psi,2\mathrm{D}}$ in the 2D image plane is given as:

$$\mathbf{R}_{\psi,2\mathrm{D}} = \begin{bmatrix} \cos(\Delta\psi) & -\sin(\Delta\psi) \\ \sin(\Delta\psi) & \cos(\Delta\psi) \end{bmatrix}. \tag{30}$$

The bounds of integration $\theta_{min}$ and $\theta_{max}$ from Eq. 20 can be computed with:

$$\theta_{min} = \max\left(-\frac{fov_c^{\mathrm{hori}}}{2}, \min(p_{2,\theta}, p_{3,\theta})\right), \tag{31}$$

$$\theta_{max} = \min\left(\frac{fov_c^{\mathrm{hori}}}{2}, \max(p_{1,\theta}, p_{4,\theta})\right). \tag{32}$$

### *B. Equidistant Fisheye Cameras*

Similar to wide-angle perspective cameras, fisheye cameras can be deployed to capture a larger visual field than is displayed to the user. Although the modes of operation of delay-compensation are conceptionally identical, the computation of the achievable compensation rate varies. In contrast to perspective cameras, the image plane of (equidistant) fisheye cameras is spherical. In this section, we will briefly revisit the theoretical background proposed in [18] and elaborate on the algorithmic methodology, which boosts the computational performance.

The image plane of a fisheye camera is displayed as a hemisphere, as depicted in Fig. 6a. Therefore, the images are distorted and need to be corrected before they are visualized to the user. We correct the fisheye image content by mapping it from its spherical image plane onto a rectangle, which corresponds to the HMD's image plane. The radius of the hemisphere is the focal length $f$ of the fisheye camera. In an identical manner as for to perspective cameras, the reachable amount of delay compensation for fisheye cameras is expressed as the ratio between the image content, which is available for visualization at the present time, and the area of the HMD's image plane:

$$c_{ptr} = \frac{area(\Pi)}{l_h^{\mathrm{hori}} \cdot l_h^{\mathrm{vert}}}. \tag{33}$$

The variables $l_h^{\mathrm{hori}}$ and $l_h^{\mathrm{vert}}$ represent the length and width of the HMD's image plane and are derived as follows:

$$l_h^{\mathrm{hori}} = 2 \cdot f \cdot \tan(\frac{fov_h^{\mathrm{hori}}}{2}), \qquad l_h^{\mathrm{vert}} = 2 \cdot f \cdot \tan(\frac{fov_h^{\mathrm{vert}}}{2}). \tag{34}$$

To determine $area(\Pi)$ we need to calculate the edge and corner points $\Pi_E \subset \Pi$, which are the minimum number of points needed to calculate $area(\Pi)$. Hence, $\Pi_E$ contains the corner points and a set of points on the curved line, which result from a partial availability of image content with respect to the current head orientation. Fig. 6 illustrates the vertices of the HMD image plane $\vec{p}_m \ \forall m \in \{1, 2, 3, 4\}$, which are calculated as follows:

$$\vec{p}_1 = \mathbf{R} \cdot \begin{bmatrix} l_h^{\mathrm{hori}}/2 \\ l_h^{\mathrm{vert}}/2 \\ f \end{bmatrix}, \quad \vec{p}_2 = \mathbf{R} \cdot \begin{bmatrix} -l_h^{\mathrm{hori}}/2 \\ l_h^{\mathrm{vert}}/2 \\ f \end{bmatrix}, \tag{35}$$

$$\vec{p}_3 = \mathbf{R} \cdot \begin{bmatrix} -l_h^{\mathrm{hori}}/2 \\ -l_h^{\mathrm{vert}}/2 \\ f \end{bmatrix}, \quad \vec{p}_4 = \mathbf{R} \cdot \begin{bmatrix} l_h^{\mathrm{hori}}/2 \\ -l_h^{\mathrm{vert}}/2 \\ f \end{bmatrix}, \tag{36}$$

with $\mathbf{R}$ the overall rotation matrix as defined in Eq. 10. We introduce the auxiliary measure $h$ to define the permitted

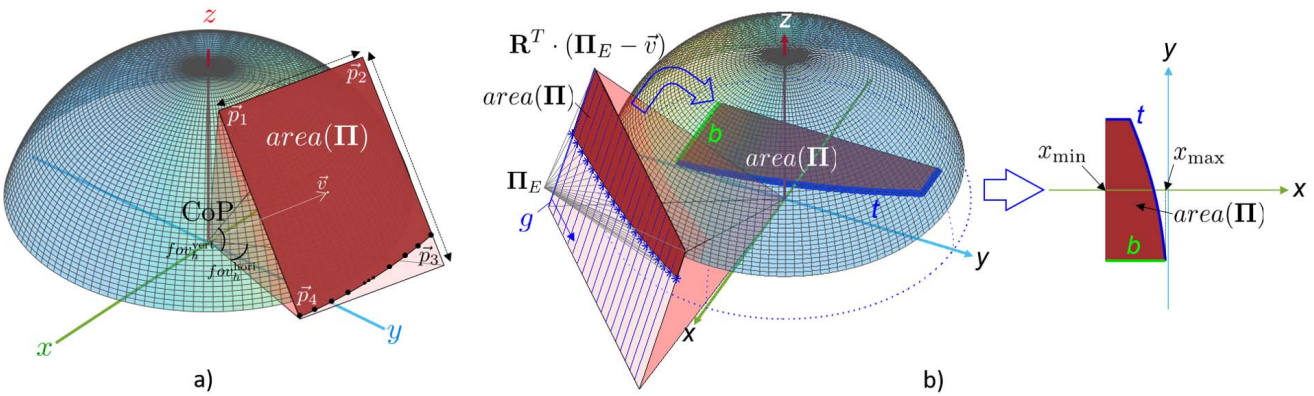


Fig. 6. a) Delay-compensation for fisheye cameras. Depending on the user's viewing direction, the ROI of the available image plane is adjusted accordingly. Depicted is the case where full compensation was not possible. The edge points of the overlapping area are determined to compute the achievable delay compensation. b) Transformation of the set $\Pi_E$ into the center of the xy-plane to ease the computation of $area(\Pi)$ with the integral over $(t-b)$ [18].

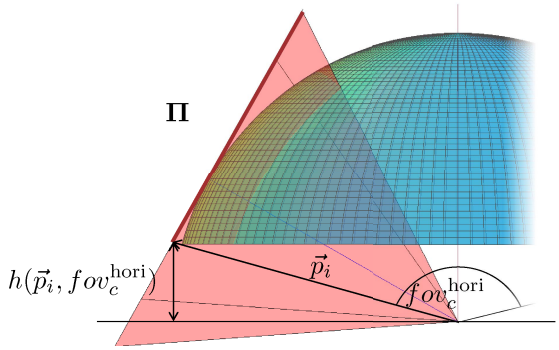


Fig. 7. The auxiliary measure $h$ is introduced to determine the minimum height $h(\vec{p}_i, fov_c^{\text{hori}})$ of an arbitrary image point $\vec{p}_i \in \mathbf{I}_h$ to be in $\Pi$ [18]. This measure holds permanently as we fixate the camera's location in the 3D world and rotate the scene instead.

height of an arbitrary image point $\vec{p}_i \in \mathbf{I}_h$ to be in $\Pi$:

$$h\left(\vec{p}_i, fov_c^{\text{hori}}\right) = |\vec{p}_i| \cdot \cos\left(\frac{fov_c^{\text{hori}}}{2}\right). \tag{37}$$

Fig. 7 shows the geometric relation to derive $h$. An image point $\vec{p}_i$ is classified to be in $\Pi$ if the following condition holds:

$$\Pi \cup \{\vec{p}_i\} \; \forall \vec{p}_i \in \mathbf{I}_h \iff p_{z,i} \geq h\left(\vec{p}_i, fov_c^{\text{hori}}\right). \tag{38}$$

We use the previous condition to identify the elements of $\Pi_E$. If all four vertices $\vec{p}_m$ meet this condition, we conclude that the compensation rate is equal to $c_{ptr} = 1$ and if none of the vertices fulfill this condition, a delay compensation is not feasable as $c_{ptr} = 0$. In all other cases the compensation rate lies between $]0, 1[$ and we need to compute the points on the curved line of the overlapping area, which can be seen in Fig. 6. We save computation time by approximating this curved line by means of 40 points. We determine these 40 points by constructing 40 lines $g$ (20 horizontal, 20 vertical) as depicted in Fig. 6b. These lines are uniformly distributed and are parallel to the edges of the HMD image plane. We thus compute them with two opposite points $\vec{q}_i, \vec{q}_j$, which are located on opposite edges of the HMD image plane:

$$g : \vec{x}_{ij} = \vec{q}_j + \lambda(\vec{q}_j - \vec{q}_i). \tag{39}$$

Hence, the points $\vec{q}_i$ and $\vec{q}_j$ are linear combinations of the corner points $\vec{p}_m \; \forall m \in \{1, 2, 3, 4\}$. For the horizontal lines $g^{\text{hori}}$ the points $\vec{q}_i$ and $\vec{q}_j$ are determined using:

$$\vec{q}_i = \vec{p}_1 + \mu \cdot (\vec{p}_2 - \vec{p}_1), \qquad \vec{q}_j = \vec{p}_4 + \mu \cdot (\vec{p}_3 - \vec{p}_4) \tag{40}$$

and for the vertical lines $g^{\text{vert}}$:

$$\vec{q}_i = \vec{p}_2 + \mu \cdot (\vec{p}_3 - \vec{p}_2), \qquad \vec{q}_j = \vec{p}_1 + \mu \cdot (\vec{p}_4 - \vec{p}_1), \tag{41}$$

where $\mu$ iterates from 0 to 1 with an arbitrarily small step size (here 20 steps). We leverage the lines $g$ to determine the points on the curved edge. These edge points are the last pixels available for visualization. The z-value of these points needs to be identical to $h$ as described in Eq. 38. Hence, we look for points $\vec{x}_{ij}$ on each line $g$, which satisfy the following equation:

$$\begin{aligned} x_{ij,z} &\stackrel{!}{=} h(\vec{x}_{ij}, fov_c^{\text{hori}}), \\ q_{i,z} + \lambda(q_{j,z} - q_{i,z}) &\stackrel{!}{=} h\left(\vec{q}_i + \lambda(\vec{q}_j - \vec{q}_i), fov_c^{\text{hori}}\right). \end{aligned} \tag{42}$$

We solve this equation for $\lambda$ by means of the Newton-Raphson method. Inserting $\lambda$ into Eq. 39 yields the desired edge points $\forall \lambda \in [0, 1]$. This range guarantees that the points lie inside the HMD image plane. The computed points now belong to the set $\Pi_E$ and can be utilized to calculate $area(\Pi)$. Similar to perspective cameras, we compute the overlapping area by integrating over the auxiliary curves $t$ and $b$:

$$area(\Pi) = \int_{x_{min}}^{x_{max}} (t - b)\, dx. \tag{43}$$

To ease the calculation, we transform the overlapping area into the center of the xy-plane as depicted in Fig. 6b and specify the curves $t$ and $b$ with the transformed edge and corner points in the set $\Pi_E$:

$$\Pi_E^{xy} = \mathbf{R}^{\text{T}} \cdot (\Pi_E - \vec{v}), \tag{44}$$

where the inverse of the rotation matrix $\mathbf{R}^{-1} = \mathbf{R}^{\text{T}}$ is simply its transpose, and $\vec{v}$ is the current viewing direction of the user,

**Algorithm 1** Assignment of the Points From Set $\Pi_E^{xy}$ to the Top Curve Set $T$ and/or the Bottom Curve set $B$

**for** all $\vec{p}_i \in \Pi_E^{xy}$ **do**
  **if** $p_{i,x} = x_{\min}$ **then** add point $\vec{p}_i$ to $X_{\min}$
  **if** $p_{i,x} = x_{\max}$ **then** add point $\vec{p}_i$ to $X_{\max}$
**if** size$(X_{\min}) = 1$ **then** $y_{\text{threshold}} = X_{\min,y}$
**else if** size$(X_{\max}) = 1$ **then** $y_{\text{threshold}} = X_{\max,y}$
**else if** size$(X_{\min}) \geq 2$ && size$(X_{\max}) \geq 2$ **then**
  $y_{\text{threshold}} = \min(\max(X_{\min,y}), \max(X_{\max,y}))$
**for** all $\vec{p}_i \in \Pi_E^{xy}$ **do**
  **if** $p_{i,y} \geq y_{\text{threshold}}$ **then** add point $\vec{p}_i$ to $T$
  **if** $p_{i,y} \leq y_{\text{threshold}}$ **then** add point $\vec{p}_i$ to $B$
sort points in $T$ according to their $x$-values
sort points in $B$ according to their $x$-values

as previously introduced in Eq. 13. The bounds of integration (see Eq. 43) can be computed as:

$$x_{\min} = \{x \in \mathbb{R} \mid \min\{p_{i,x}\}, \forall \vec{p}_i = (p_{i,x}, p_{i,y})^T \in \Pi_E^{xy}\},$$
$$x_{\max} = \{x \in \mathbb{R} \mid \max\{p_{i,x}\}, \forall \vec{p}_i = (p_{i,x}, p_{i,y})^T \in \Pi_E^{xy}\}. \quad (45)$$

To compute the curves $t$ and $b$, all points in $\Pi_E^{xy}$ have to be allocated to either the set $T$, which contains the points on the top curve $t$, or the set $B$, which contains the points on the bottom curve $b$, or both. Fig. 6 illustrates how the points are appropriately assigned. The process of allocating the points in $\Pi_E^{xy}$ to the sets $T$ and $B$ is presented as pseudocode in Algorithm 1, where $T$, $B$, $X_{\min}$ and $X_{\max}$ are sets of points. The curves $t$ and $b$ can be computed from the sets $T$ and $B$ by calculating the lines between neighboring points $\vec{p}_k$ and $\vec{p}_{k+1}$ with $\vec{p}_k, \vec{p}_{k+1} \in T$ or $\vec{p}_k, \vec{p}_{k+1} \in B$ as:

$$m = \frac{p_{k+1,y} - p_{k,y}}{p_{k+1,x} - p_{k,x}} \cdot x + \frac{p_{k,x}\, p_{k+1,y} - p_{k+1,x}\, p_{k,y}}{p_{k,x} - p_{k+1,x}}. \quad (46)$$

Therefore, the curves $t$ and $b$ consist of concatenated line segments $m$, which are accurate enough to precisely calculate $area(\Pi)$ with the integral in Eq. 43, and hence the present level of achievable delay compensation.

### C. Generic Delay Compensation

The two approaches above are explicitly tailored to the underlying camera system. In this section, we propose a generic algorithm for delay compensation, which is independent of the deployed camera system and can be dynamically applied for both perspective and fisheye cameras. We map the camera's image plane $\mathbf{I}_c$ and the HMD image plane $\mathbf{I}_h$ onto a common sphere and specify the overlapping area $\hat{\Pi}$ as the image content which is available for display at that time instance (see Fig. 5). The image planes mapped onto the sphere are denominated as $\hat{\mathbf{I}}_c$ and $\hat{\mathbf{I}}_h$, with their overlapping area $\hat{\Pi}$. The generic compensation rate can hence be expressed as:

$$c_{ptr} = \frac{area(\hat{\Pi})}{area(\hat{\mathbf{I}}_h)}. \quad (47)$$

We first compute the area of the rectangular HMD image plane, which is mapped onto the sphere. We operate in the spherical domain and apply the scalar surface integral [30] as follows:

$$area(\hat{\mathbf{I}}_h) = \iint_{\hat{\mathbf{I}}_h} \mathrm{d}\hat{\mathbf{I}}_h = \int_{\upsilon_{\min}}^{\upsilon_{\max}} \int_{u_{\min}(\upsilon)}^{u_{\max}(\upsilon)} \left\| \frac{\partial \vec{\varphi}}{\partial u} \times \frac{\partial \vec{\varphi}}{\partial \upsilon} \right\| \mathrm{d}u\, \mathrm{d}\upsilon, \quad (48)$$

with $u$ and $\upsilon$ being the polar and azimuth angles, respectively. $\vec{\varphi}$ is a parameterization of spherical coordinates [30] and is derived according to:

$$\vec{\varphi}(u, \upsilon) = \begin{bmatrix} r \cdot \sin(u) \cdot \cos(\upsilon) \\ r \cdot \sin(u) \cdot \sin(\upsilon) \\ r \cdot \cos(u) \end{bmatrix}. \quad (49)$$

The partial derivatives of the parameterization $\vec{\varphi}(u, \upsilon)$ are determined using:

$$\frac{\partial \vec{\varphi}}{\partial u} = \begin{bmatrix} r \cdot \cos(u) \cdot \cos(\upsilon) \\ r \cdot \cos(u) \cdot \sin(\upsilon) \\ -r \cdot \sin(u) \end{bmatrix}, \quad \frac{\partial \vec{\varphi}}{\partial \upsilon} = \begin{bmatrix} -r \cdot \sin(u) \cdot \sin(\upsilon) \\ r \cdot \sin(u) \cdot \cos(\upsilon) \\ 0 \end{bmatrix}. \quad (50)$$

By applying the cross product to these derivatives and taking the Euclidean norm, we obtain:

$$\left\| \frac{\partial \vec{\varphi}}{\partial u} \times \frac{\partial \vec{\varphi}}{\partial \upsilon} \right\| = r^2 \sin(u). \quad (51)$$

By inserting this result into Eq. 48, we get the final equation for the HMD's image plane area:

$$area(\hat{\mathbf{I}}_h) = \int_{\upsilon_{\min}}^{\upsilon_{\max}} \int_{u_{\min}(\upsilon)}^{u_{\max}(\upsilon)} r^2 \cdot \sin(u)\, \mathrm{d}u\, \mathrm{d}\upsilon. \quad (52)$$

To compute the bounds of integration of the integral in spherical coordinates, we leverage the corner points of the HMD image plane as derived in Eq. 35 and 36, and all points on the edges of the HMD image plane $\vec{p}_i \in \mathbf{I}_{h,E}$ between these corner points. We map these points onto a sphere with radius $r_{sph}$ according to:

$$\hat{\vec{p}}_i = \frac{\vec{p}_i}{|\vec{p}_i|} \cdot r_{sph}. \quad (53)$$

We first transform the points $\hat{\vec{p}}_i = (\hat{p}_{i,x}, \hat{p}_{i,y}, \hat{p}_{i,z})^T \in \hat{\mathbf{I}}_{h,E}$ from cartesian to spherical coordinates $\hat{\vec{p}}_i^{\,\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,\upsilon}^{\text{sph}})^T \in \hat{\mathbf{I}}_{h,E}^{\text{sph}}$ [30]:

$$\hat{p}_{i,u}^{\text{sph}} = \arccos\left( \frac{\hat{p}_{i,z}}{\sqrt{\hat{p}_{i,x}^2 + \hat{p}_{i,y}^2 + \hat{p}_{i,z}^2}} \right), \quad (54)$$

$$\hat{p}_{i,\upsilon}^{\text{sph}} = \arctan\left( \frac{\hat{p}_{i,y}}{\hat{p}_{i,x}} \right). \quad (55)$$

The radius $\hat{p}_{i,r}^{\text{sph}} = r_{sph}$ is identical to $r_{sph}$ as we have normalized the points $\vec{p}_i \in \mathbf{I}_{h,E}$ to the radius $r_{sph}$ (Eq. 53). The respective transformation of the HMD image plane into the spherical domain is visualized in Fig. 8.

The bounds of integration $\upsilon_{\min}$ and $\upsilon_{\max}$ can be calculated by the minimum or the maximum $\upsilon$-value of all points in $\hat{\mathbf{I}}_{h,E}^{\text{sph}}$:

$$\upsilon_{\min} = \{\upsilon \in \mathbb{R} \mid \min\{\hat{p}_{i,\upsilon}^{\text{sph}}\}, \forall \hat{\vec{p}}_i^{\,\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,\upsilon}^{\text{sph}})^T \in \hat{\mathbf{I}}_{h,E}^{\text{sph}}\},$$
$$\upsilon_{\max} = \{\upsilon \in \mathbb{R} \mid \max\{\hat{p}_{i,\upsilon}^{\text{sph}}\}, \forall \hat{\vec{p}}_i^{\,\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,\upsilon}^{\text{sph}})^T \in \hat{\mathbf{I}}_{h,E}^{\text{sph}}\}. \quad (56)$$

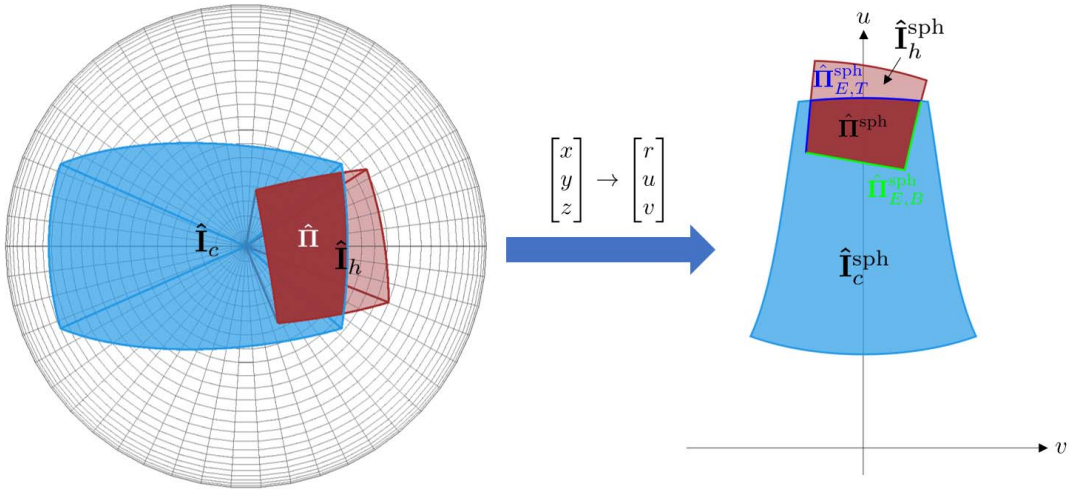


Fig. 8. Overview of the generic delay-compensation approach. Both the requested viewport of the user as well as the available image content are mapped onto the sphere to find the corresponding ROI. Depending on the user's head velocity, the available residual imagery, and the present delay a full compensation cannot always be ensured. To ease the computation of the achievable degree of compensation, we transform the pixels into the spherical domain and determine the overlapping area, which describes the image content that is available for visualization. The ratio of the overlapping area and the size of the viewport conveys the compensation rate.

Note that the bounds of integration $u_{\min}(v)$ and $u_{\max}(v)$ depend on the integration variable $v$ as described in Eq. 52, and hence need special attention. For this reason, we assign the points in $\hat{\mathbf{I}}_{h,E}^{\text{sph}}$ to either the top line $\hat{\mathbf{I}}_{h,T}^{\text{sph}}$ or the bottom line $\hat{\mathbf{I}}_{h,B}^{\text{sph}}$ of the overlapping area and sort them by their $x$-values as depicted in Fig. 8. For a given $v$ value, $u_{\min}(v)$ and $u_{\max}(v)$ are computed by searching for the corresponding point on the bottom or top line, respectively:

$$\begin{aligned} u_{\min}(v) &= \{u \in \mathbb{R} \mid \hat{p}_{i,v}^{\text{sph}} = v, \\ &\qquad \forall \hat{\vec{p}}_i^{\,\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,v}^{\text{sph}})^T \in \hat{\mathbf{I}}_{h,B}^{\text{sph}} \subset \hat{\mathbf{I}}_h^{\text{sph}}\}, \\ u_{\max}(v) &= \{u \in \mathbb{R} \mid \hat{p}_{i,v}^{\text{sph}} = v, \\ &\qquad \forall \hat{\vec{p}}_i^{\,\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,v}^{\text{sph}})^T \in \hat{\mathbf{I}}_{h,T}^{\text{sph}} \subset \hat{\mathbf{I}}_h^{\text{sph}}\}. \end{aligned} \tag{57}$$

In compliance with Eq. 52, we utilize these bounds to compute $area(\hat{\mathbf{I}}_h)$. The calculation of the overlapping area $\hat{\Pi}$ follows the concept of Eq. 52 except for the bounds of integration:

$$area(\hat{\Pi}) = \int_{v_{\min}^{\hat{\Pi}}}^{v_{\max}^{\hat{\Pi}}} \int_{u_{\min}^{\hat{\Pi}}(v)}^{u_{\max}^{\hat{\Pi}}(v)} r^2 \cdot \sin(u)\, \mathrm{d}u\, \mathrm{d}v. \tag{58}$$

To compute these integration bounds, we need to specify the edges of the overlapping area. This is done by making use of the camera's image plane. For perspective cameras, the image plane corresponds to a rectangle, which is mapped onto the sphere akin to the HMD's image plane. We therefore determine the edges of the camera image plane as described in Section III-A except that we use the camera's $fov_c$ instead of the HMD's. In a subsequent step, we map these points onto the sphere and transform them into the spherical domain as presented in Eqs. 53 to 55, which can be also seen in Fig. 8.

In case of equidistant fisheye cameras, the image plane is already spherical. We however need to adapt the radius of the sphere which we map the image planes onto, such that it is identical to the focal length of the fisheye camera $r_{sph} = f$. For equidistant fisheye cameras the images are circular by which $fov_c^{\text{hori}} = fov_c^{\text{vert}}$ is deemed valid. The edges of the spherical fisheye image plane $\vec{p}_i \in \hat{\mathbf{I}}_{c,E}$ are derived as follows:

$$\vec{p}_i = \mathbf{R} \cdot \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} f \cdot \sin(fov_c^{\text{hori}}/2) \\ 0 \\ f \cdot \cos(fov_c^{\text{hori}}/2) \end{bmatrix}. \tag{59}$$

$\forall \alpha \in [0, 2\pi)$. The points $\hat{\mathbf{I}}_{c,E}$ of the fisheye camera are transformed to spherical coordinates $\hat{\mathbf{I}}_{c,E}^{\text{sph}}$ with Eq. 54 and 55. We use these edges of the camera image plane and the edges of the HMD image plane $\hat{\mathbf{I}}_{h,E}^{\text{sph}}$ to identify the innermost edges of the overlapping area $\hat{\Pi}_E^{\text{sph}}$. In general, there are three cases which can occur: If the HMD image plane is located completely within the camera image plane, the compensation rate is considered to be 1. If it lies entirely outside of the camera image plane, the compensation rate is set to 0. In all other cases, $area(\hat{\Pi})$ is calculated using Eq. 58. The respective bounds of integration are again computed by means of $\hat{\Pi}_{E,T}^{\text{sph}}$, which corresponds to the set of points on the top line, and $\hat{\Pi}_{E,B}^{\text{sph}}$, which contains the set of points on the bottom line of the edges $\hat{\Pi}_E^{\text{sph}}$ of the overlapping area as can be seen in Fig. 8:

$$\begin{aligned} v_{\min}^{\hat{\Pi}} &= \{v \in \mathbb{R} \mid \min\{\hat{p}_{i,v}^{\text{sph}}\}, \quad \forall \hat{\vec{p}}_i^{\,\text{sph}} \in \hat{\Pi}_E^{\text{sph}}\}, \\ v_{\max}^{\hat{\Pi}} &= \{v \in \mathbb{R} \mid \max\{\hat{p}_{i,v}^{\text{sph}}\}, \quad \forall \hat{\vec{p}}_i^{\,\text{sph}} \in \hat{\Pi}_E^{\text{sph}}\}, \\ u_{\min}^{\hat{\Pi}}(v) &= \{u \in \mathbb{R} \mid \hat{p}_{i,v}^{\text{sph}} = v, \quad \forall \hat{\vec{p}}_i^{\,\text{sph}} \in \hat{\Pi}_{E,B}^{\text{sph}}\}, \\ u_{\max}^{\hat{\Pi}}(v) &= \{u \in \mathbb{R} \mid \hat{p}_{i,v}^{\text{sph}} = v, \quad \forall \hat{\vec{p}}_i^{\,\text{sph}} \in \hat{\Pi}_{E,T}^{\text{sph}}\}, \end{aligned} \tag{60}$$

where $\hat{\vec{p}}_i^{\,\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,v}^{\text{sph}})^{\text{T}}$. After quantifying the areas of $\hat{\Pi}$ and $\hat{\mathbf{I}}_{hmd}$, we specify the generic compensation rate with respect to Eq. 47.

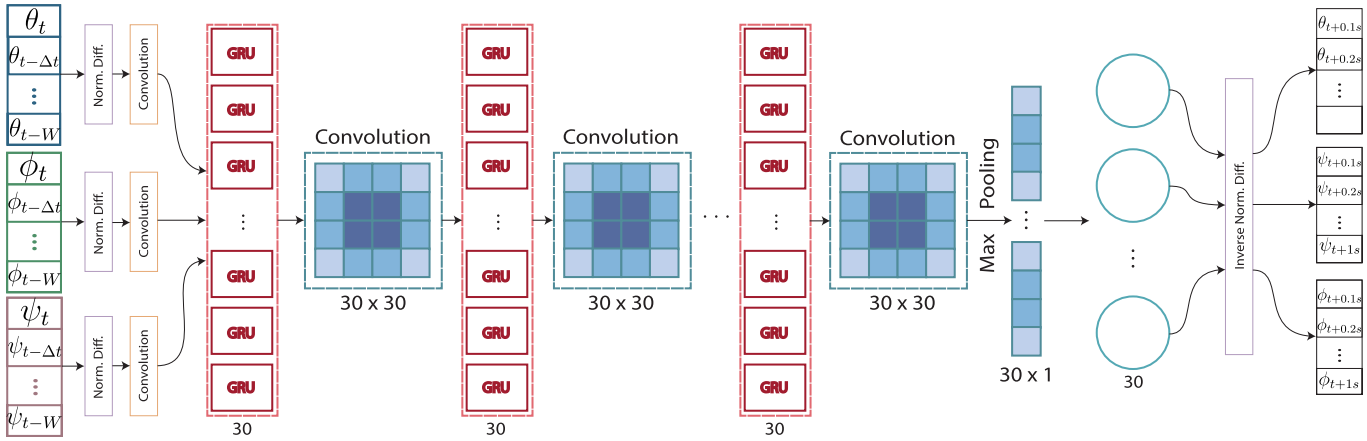


Fig. 9. Schematic overview of our proposed deep neural network for head-motion prediction. The input is first transformed to normalized difference to ensure generalization, and passed through a convolution unit that acts as a low-pass filter. Six sequential GRU and convolution units compute the most distinct features at different granularities. A subsequent max pooling layer downsamples the output of the last convolution layer und preserves the most significant features. After a dense feed-forward network, the differences are inverted to absolute values. The output predicts a whole sequence of future orientation values ($\vec{\xi}^{\text{pred}}_{t+n\cdot 100\text{ms}}$ $\forall n \in \mathbb{Z}, n = 1, ..., 10$).

## IV. Head Motion Prediction

The delay-compensation approach by itself works always only to a certain extent. Its merit is highly subject to the design parameters such as the overall delay, the provided visual field of the camera and the choice of the user viewport's size. If there is a sudden and fast head movement or an abrupt orientation change, it is unavoidable that the desired viewport will be outside of the buffer zone. In [7], we proposed a psycho-physically motivated velocity-based FoV adaptation approach to increase the buffer size - and hence the delay compensation capability - during fast head rotations, while keeping the visual comfort at the same level. With and without this adaptation technique, the non-compensable areas are either filled with black or artificially colored pixels, which is done by taking the color values of the outermost pixels of the viewport. In order to remedy such unsatisfactory tendencies, however, it would be desirable to know the future head motion after the current delay. In this way, we can send the prospective head orientation rather than the present one, so as to avoid unwanted phenomena. One widely accepted method is to apply head motion prediction techniques. Former methods rely either on traditional or neural network-based techniques. In this paper, we propose a deep neural network that is based on an interleaved structure of gated recurrent units and convolution components. GRUs belong to the group of recurrent neural networks (RNN) that share weights over time, and hence capture long-term dependencies. Initially, RNNs were lacking in practical relevance due to the vanishing and exploding gradient problem, until the introduction of long short-term memory (LSTM) units. In [7], the best performance for head-motion prediction was achieved with a network structure that is based on an interleaved structure of LSTM units and dense networks. We instead make use of dense GRUs combined with convolutional components to exploit local features at different granularities. The GRU is computationally more efficient, as it has fewer parameters and states than LSTM units. The proposed network is depicted in Fig. 9.

Our proposed network is fed with a sequence of pan, tilt, and roll orientation values. We consider the present and past values within a window $W$. We empirically discovered $W = 250$ms (last 20 values) to be a good value. We ensure the network's generalization capabilities by transforming the absolute orientation values into their normalized difference:

$$\hat{\vec{\xi}}_{\text{diff, t}} = \frac{\vec{\xi}_{\text{diff, t}}}{\max(|\vec{\xi}_{\text{diff}}|)}, \quad \text{with } \vec{\xi}_{\text{diff, t}} = \vec{\xi}_{\text{t}} - \vec{\xi}_{\text{t-}\Delta\text{t}}, \tag{61}$$

where $\Delta t = \frac{1}{f_{\text{IMU}}}$ is given by the Inertial Measurement Unit's (IMU's) frequency and describes the timesteps when new orientation data is provided. In this way, we train the network on the normalized differences rather than on the absolute values. Training solely on absolute orientation values led to erroneous predictions for other datasets. After computing the normalized differences, we added a convolution layer with a kernel size of ten as a sort of low-pass filter to reduce the noise. The core of the network consists of an interleaved structure of six dense GRUs each with 30 nodes and subsequent convolution units (kernel size 30x30), which compute the most distinct features. The successive max pooling layer acts as a discretization process and down-samples the output of the last convolution layer by keeping the most significant features. After passing the values through a dense feedforward network (FFN), we remap to the absolute orientation values as follows:

$$\vec{\xi}^{\text{pred}}_{\text{t}} = \vec{\xi}_{\text{t}} + \sum_{\text{k=t}-\tau/\Delta\text{t}}^{\text{t}} \hat{\vec{\xi}}^{\text{pred}}_{\text{diff, t}} \cdot \max(|\vec{\xi}_{\text{diff}}|). \tag{62}$$

Rather than predicting only a single value, our output forecasts a whole sequence of orientation values ($\vec{\xi}^{\text{pred}}_{t+n\cdot 100\text{ms}}$ $\forall n \in \mathbb{Z}, n = 1, ..., 10$).

The training procedure was configured for 1000 epochs, including an early-stopping technique (patience=10, min. delta=0) to avoid overfitting. The batch size was set to $2^9$. A learning-rate-decay scheduler was incorporated, to manually

optimize the learning process. The initial learning rate of $10^{-3}$ was decreased every 30 epochs by 70%. The Adam optimizer was used as the gradient-descent algorithm. For the FFN, we utilized Rectified Linear Units (ReLU's) as activation function. The proposed network was implemented in Keras and Tensorflow. We used a Core i7 (x64) machine for training and inference. The network yielded outputs in the range of single-digit microseconds, and thus proved its realtime inference abilities.

## V. Experiments

The proposed realtime-capable 3D vision system was implemented and mounted on our semi-autonomous MAVI telepresence platform [1] as depicted in Fig. 1. For reproducible and representative evaluation, however, we performed the validation on a stereoscopic 360° video sequence, which was kindly provided by the Fraunhofer HHI [31]. For both scenarios, we implemented a server/client-based setup, where the 3D 360° visual representation of the scene is (captured and) provided at the server side according to the desired head orientation. The client side records the user's head orientation, is responsible for selecting and rendering the correct viewport, and predicts prospective head-orientation values. We established both a TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) connection between the server and the client. We use a TCP connection for reliably streaming the visual content by means of the Real Time Streaming Protocol (RTSP). Both frames for the left and the right eye are stacked and compressed with the H.264/AVC codec [32]. We used the realtime capable *x264* implementation [33] of the H.264/AVC standard [32]. We selected the *baseline* profile and set the preset configuration to *ultrafast*. We further tuned the codec with the *zerolatency* and *fastdecode* configuration to optimize for fast encoding and low latency streaming as well as to disable computationally demanding filters such as the deblocking filter and CABAC (Context-Adaptive Binary Arithmetic Coding) to allow for faster decoding on devices with lower computational power. Due to the incongruity between the camera frame rate and the IMU's data-acquisition rate, we use UDP for transmitting the current head orientation quickly and efficiently, where a dropped packet has no perceivable impact. The communication delay was set constant for investigated latencies between 0-1s by means of a network emulator [34].

We used the LMT dataset [17], where 30 participants, aged between 22-40, watched 3 different 360° videos with changing dynamics in the respective scene. In total, the dataset consists of $30 \cdot 3 = 90$ subsets, each with an average video length of 120s. The IMU sensor, which was used to record the head motions, provided the filtered orientation data at a frequency of $f_{\text{IMU}} = 80\text{Hz}$. 81 profiles were used to train the deep network, while the remaining 9 profiles were utilized for validation. To prove the proposed concept's generalization abilities, we leveraged a completely independent dataset (called the "IMT dataset" in the following), which was recorded by the IMT group under dissimilar conditions and with different participants [35]. They registered the head-motion profile of 58 subjects where each of them watched five 360° videos of 70 seconds in length.

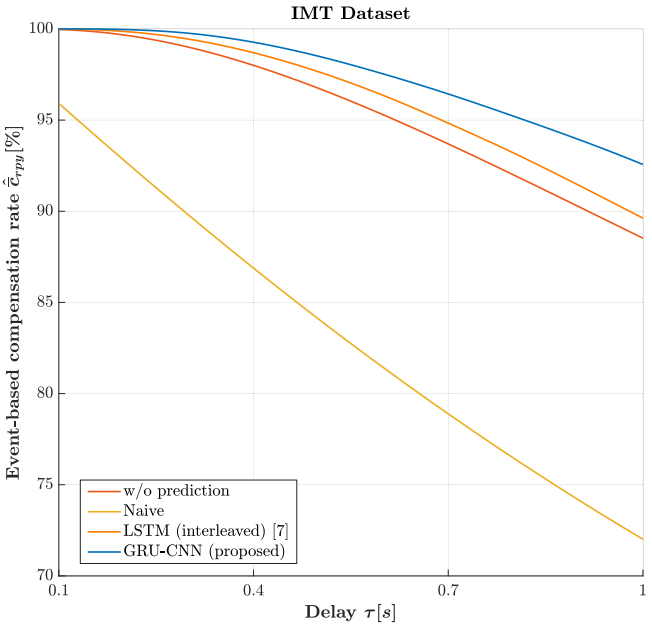


Fig. 10. The proposed generic compensation rate is leveraged as a metric to compare the level of achievable delay compensation for various methods. It is apparent that applying the proposed (generic) compensation approach results in a significant improvement compared to the *naive* approach where no compensation is present. Adopting head motion prediction techniques helps to improve the degree of latency compensation. Our proposed deep neural-network-based approach outperforms prior methods considerably.

We use the Mean Absolute Error (MAE), the root mean square error (RMSE) and the (event-based) mean compensation rate $\hat{\bar{c}}_{ptr}$ as qualitative measures to evaluate our approach and to compare it with state-of-the-art solutions. In [7], we proposed the event-based mean compensation rate as a measure that averages the mean compensation rate for all videos $V$, participants $P$, and available samples $S = \frac{t_{\text{end}}^{\text{V, P}} - t_{\text{start}}^{\text{V, P}}}{f_{\text{IMU}}}$ for a present delay $\tau$ where at least one of the head-rotation velocities is above the IMU's sensor noise ($\epsilon = 5°$):

$$\hat{\bar{c}}_{rpy} = \frac{1}{V \cdot P \cdot \sum_{k=1}^{S} \delta(a_k)} \cdot \sum_{i=1}^{P} \sum_{j=1}^{V} \sum_{k=1}^{S} \delta(a_k) \cdot c_{rpy}^{i,j,k}, \quad (63)$$

where the dirac delta function $\delta(a_k)$ returns one when its argument $(a_k)$ becomes zero:

$$a_k = \begin{cases} 0, & \text{if } \{|\dot{\theta}_k| \text{ or } |\dot{\phi}_k| \text{ or } |\dot{\psi}_k|\} \geq 0 + \epsilon \\ \neq 0, & \text{otherwise.} \end{cases} \quad (64)$$

We investigated varying latencies in the range $\tau \in [0.1 - 1s]$ and deployed wide-angle stereo cameras. Table I summarizes the utilized system attributes of the camera and the defined visual field of the user. The vertical $fov_h^{\text{vert}}$ of the user is determined with respect to the required aspect ratio (8:9) of the HMD ($ratio$):

$$fov_{\{h,c\}}^{\text{vert}} = 2 \cdot \arctan\left(\frac{1}{ratio} \cdot \tan\left(\frac{fov_{\{h,c\}}^{\text{hori}}}{2}\right)\right). \quad (65)$$

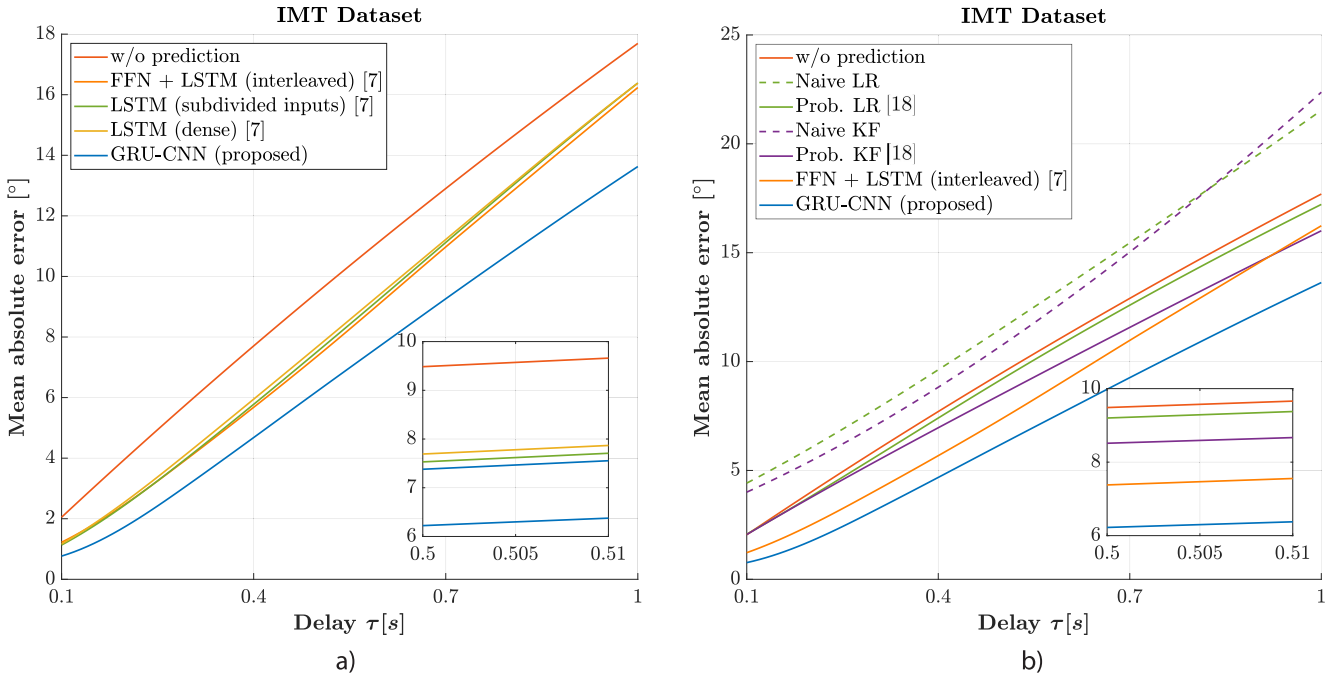


Fig. 11. Mean absolute error values are visualized for different head-motion prediction methods. Our proposed approach is shown in blue and proves to be a substantial improvement compared to prior art. a) We compare the mean absolute error of previous neural network-based methods to the *naive* approach, where no prediction is applied, and to our proposed method. b) We contrast the mean absolute error of our deep network with the best prior neural network, the *naive* approach, and with other representative traditional techniques.

TABLE I

SPECIFICATIONS OF THE CAMERA SYSTEM $fov^{\{\text{HORI, VERT, DIAG}\}}$ AND THE RESULTING DIAGONAL BUFFER SIZES USED FOR EVALUATION

| | hori | vert | diag | $b^{\text{diag}}$ |
|---|---|---|---|---|
| $fov_c$ | 100° | 60° | 105.88° | - |
| $fov_h$ | 60° | 66.01° | 81.98° | **11.94°** |

Table I shows that residual imagery is only provided in the horizontal direction. As the vertical field of view of the camera and the user are almost identical, a vertical buffer is almost nonexistent, and hence makes it a challenging task. A more generic and indicative measure is, however, the diagonal buffer size $b^{\text{diag}}$ that leverages the diagonal $fov^{\text{diag}}_{\{h,c\}}$:

$$fov^{\text{diag}}_{\{h,c\}} = 2 \cdot \arctan\left(\sqrt{\tan^2\left(\frac{fov^{\text{hori}}_{\{h,c\}}}{2}\right) + \tan^2\left(\frac{fov^{\text{vert}}_{\{h,c\}}}{2}\right)}\right). \tag{66}$$

The available diagonal buffer size for our scenario is presented in Table I and can be computed with:

$$b^{\text{diag}} = \tfrac{1}{2}(fov^{\text{diag}}_c - fov^{\text{diag}}_h). \tag{67}$$

## VI. DISCUSSION

We compare our approach to state-of-the-art traditional (non-deep-learning-based) and deep neural network-based methods. Traditional techniques have been investigated for years and usually apply either regression- or model-based extrapolation methods using past trajectories. One of the most representative techniques are (weighted) Linear Regression (LR), here denoted as *naive LR*, and the Kalman Filter-based extrapolation technique (*naive KF*). The KF is utilized to first estimate the optimal state of the head orientation. In a further step, a constant-acceleration motion model is assumed and adopted for extrapolation. In [18], a probabilistic error model was proposed, which improves the accuracy of existing methods such as the *naive LR* and the *naive KF* significantly. The forecasted value is weighted by its probability of being at that orientation value. Deploying deep neural networks for head-motion prediction were initially investigated in [7].

We evaluate all methods using the IMT dataset, to examine not only the accuracy and the level of achievable delay compensation but also the generalization abilities. In Fig. 11 and Fig. 12, we contrast the MAE and the RMSE of our proposed approach with the best three deep-learning-based approaches as well as to the most representative traditional non-deep-learning-based techniques. It can be seen that our proposed GRU-CNN-based network is able to substantially improve the MAE and RMSE compared to all existing methods, especially for larger delays.

We further investigate the degree of latency compensation by applying the (event-based) generic compensation rate as a valid metric, as is shown in Fig. 10. For the *naive LR*, where no compensation and prediction is applied, the generic compensation rate comes to 72.8%. Deploying the proposed delay-compensation approach together with the presented deep-learning-based head-motion prediction technique, we manage to obtain a mean (event-based) generic compensation rate of 97.3% although we deployed only a small buffer. The results demonstrate that applying the (generic)

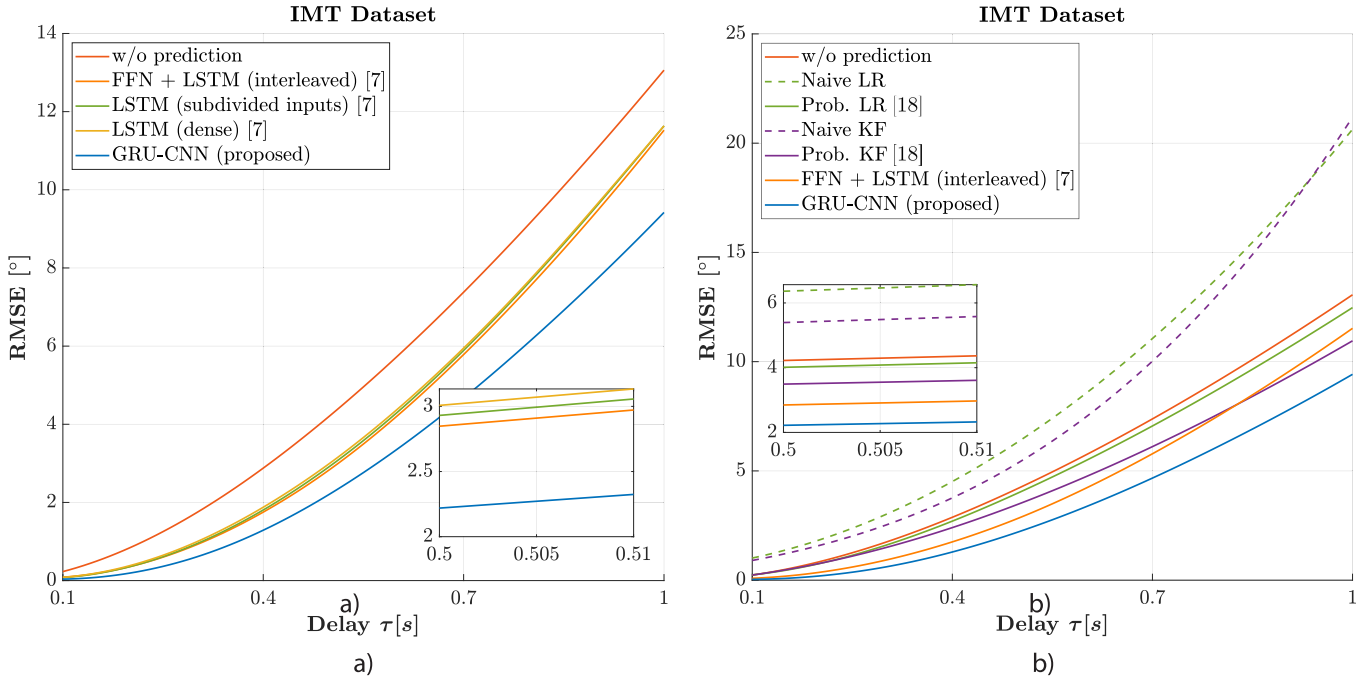


Fig. 12. The RMSE is demonstrated for prior art that investigated head-motion prediction. The blue curve represents our proposed approach. It can be seen that our proposed network exhibits the lowest RMSE values for all delay instances. a) We compare the root mean squared error of previous neural network-based methods to the *naive* approach, where no prediction is applied and to our proposed method. b) We contrast the root mean squared error of our deep network with the best prior neural network, the *naive* approach, and with other representative traditional techniques.

DCVS drastically improves the level of delay compensation even when using no prediction and only small buffers. While the best deep-learning-based approach in [7] was able to outperform prior techniques, our proposed method can further optimize and substantially enhance the level of delay compensation.

## VII. Conclusion

In this paper, we investigate the perceived realtime capability of telepresence systems that provide a 3D impression of a remote scene. Most applications of telepresence systems are impeded by the onset of motion sickness. Visual discomfort will lead to instant termination of the telepresence session. This is caused by an overly large motion-to-photon latency that describes the time needed to reflect the user's head motion onto the display. We propose a novel approach that provides the user with a stereoscopic 360° visual impression of the distant environment. The user perceives the acquisition and streaming of the 3D 360° visual content in realtime. We provide instant visual feedback for any casual head-motion and, thus, avoid the occurrence of visual discomfort. By following a *vision on demand* policy, we are able to exploit the benefits of a low-cost and lean stereoscopic vision system that is mounted on an electro-mechanical unit to mimic the user's head movements. By capturing a larger visual field than actually displayed to the user, we manage to leverage the residual image content for local delay compensation. We change the ROI according to the user's head orientation and update the footage as soon as the new imagery at the desired view direction is available. In this paper, we complete the algebraic description of our proposed algorithm for perspective and fisheye cameras considering all 3DoF of a head motion. We further propose a novel technique, which we refer to as "generic compensation", to apply the delay compensation approach and algebraically formulate the level of achievable delay compensation independent of the underlying camera system.

In a final step, we present a new deep-learning-based head-motion prediction network that outperforms existing methods significantly. We use qualitative measures such as the MAE, the RMSE, and the (generic) compensation rate to evaluate our approach. The results verify that applying even a small buffer can lead to a substantial enhancement of delay compensation when making use of our proposed deep network.

In future work, we plan to put further effort into extending the deep-learning-based head-motion prediction in different directions. Among others, we want to use the compensation rate itself as loss function to allow the network a certain error tolerance as long as the compensation rate is kept at its maximum. Besides that, it seems promising to incorporate the visual content into the prediction mechanism. For that, we aim to combine current IMU-based prediction method with a deep neural network architecture that estimates salient regions in the scene and forecasts where the user might look next.

In the long term, we further target to develop a similar delay compensation approach for manipulation tasks with our semi-autonomous MAVI telepresence platform. Within this scope, it is relevant to investigate the impact and emergence of motion sickness in the context of eye-hand coordination.

## References

[1] M. Karimi, T. Aykut, and E. Steinbach. (2018). "MAVI: A research platform for telepresence and teleoperation." [Online]. Available: https://arxiv.org/abs/1805.09447

[2] S. M. LaValle, A. Yershova, M. Katsev, and M. Antonov, "Head tracking for the Oculus Rift," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May/Jun. 2014, pp. 187–194.

[3] L. Tillman, *Motion Sickness*. New York, NY, USA: Academic, 1975.

[4] J.-R. Wu and M. Ouhyoung, "On latency compensation and its effects on head-motion trajectories in virtual environments," *Vis. Comput.*, vol. 16, no. 2, pp. 79–90, 2000.

[5] R. S. Allison, L. R. Harris, M. Jenkin, U. Jasiobedzka, and J. E. Zacher, "Tolerance of temporal delay in virtual environments," in *Proc. Virtual Reality*, Mar. 2001, pp. 247–254.

[6] M. A. Watson and F. O. Black, "The human balance system: A complex coordination of central and peripheral systems," Vestibular Disorders Assoc., Portland, OR, USA, Tech. Rep., 2008.

[7] T. Aykut, M. Karimi, C. Burgmair, A. Finkenzeller, C. Bachhuber, and S. Eckehard, "Delay compensation for a telepresence system with 3D 360° vision based on deep head motion prediction and dynamic FoV adaptation," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 4343–4350, Oct. 2018.

[8] R. S. Allison, B. J. Gillam, and E. Vecellio, "Binocular depth discrimination and estimation beyond interaction space," *J. Vis.*, vol. 9, no. 1, p. 10, 2009.

[9] D. Drascic, "Skill acquisition and task performance in teleoperation using monoscopic and stereoscopic video remote viewing," in *Proc. Hum. Factors Soc. Annu. Meeting*. Los Angeles, CA, USA: SAGE Publications, vol. 35, no. 19, 1991, pp. 1367–1371.

[10] J. Y. C. Chen, E. C. Haas, and M. J. Barnes, "Human performance issues and user interface design for teleoperated robots," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 37, no. 6, pp. 1231–1245, Nov. 2007.

[11] R. Aggarwal, A. Vohra, and A. M. Namboodiri, "Panoramic stereo videos with a single camera," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3755–3763.

[12] R. Anderson *et al.*, "Jump: virtual reality video," *ACM Trans. Graph.*, vol. 35, no. 6, p. 198, 2016.

[13] *Facebook Surround 360*. Accessed: Dec. 1, 2017. [Online]. Available: https://facebook360.fb.com/ facebook-surround-360/

[14] *Nokia OZO*. Accessed: Dec. 1, 2017. [Online]. Available: https://ozo.ia.com/eu/

[15] *Samsung–Project Beyond*. Accessed: Dec. 1, 2017. [Online]. Available: http://thinktankteam.info/be-yond/

[16] K. Matzen, M. F. Cohen, B. Evans, J. Kopf, and R. Szeliski, "Low-cost 360 stereo photography and video capture," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 148:1–148:12, Jul. 2017.

[17] T. Aykut, S. Lochbrunner, M. Karimi, B. Cizmeci, and E. Steinbach, "A stereoscopic vision system with delay compensation for 360° remote reality," in *Proc. Thematic Workshops ACM Multimedia*, 2017, pp. 201–209.

[18] T. Aykut, C. Burgmair, M. Karimi, J. Xu, and E. Steinbach, "Delay compensation for actuated stereoscopic 360Â° telepresence systems with probabilistic head motion prediction," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 2010–2018.

[19] S. Peleg and M. Ben-Ezra, "Stereo panorama with a single camera," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, Jun. 1999, pp. 395–401.

[20] C. Richardt, Y. Pritch, H. Zimmer, and A. Sorkine-Hornung, "Megastereo: Constructing high-resolution stereo panoramas," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 1256–1263.

[21] C. Weissig, O. Schreer, P. Eisert, and P. Kauff, "The ultimate immersive experience: Panoramic 3D video acquisition," in *Proc. Int. Conf. Multimedia Modeling*. Berlin, Germany: Springer, 2012, pp. 671–681.

[22] O. Schreer, P. Kauff, P. Eisert, C. Weissig, and J.-C. Rosenthal, "Geometrical design concept for panoramic 3D video acquisition," in *Proc. Proc. 20th Eur. Signal Process. Conf. (EUSIPCO)*, Aug. 2012, pp. 2757–2761.

[23] H.-Y. Shum and L.-W. He, "Rendering with concentric mosaics," in *Proc. 26th Annu. Conf. Comput. Graph. Interact. Techn.* Reading, MA, USA: Addison-Wesley, 1999, pp. 299–306.

[24] X. Corbillon, A. Devlic, G. Simon, and J. Chakareski, "Optimal set of 360° videos for viewport-adaptive streaming," in *Proc. ACM Multimedia Conf.*, 2017, pp. 943–951.

[25] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proc. 5th Workshop Things Cellular, Oper., Appl. Challenges*, 2016, pp. 1–6.

[26] R. Azuma and G. Bishop, "A frequency-domain analysis of head-motion prediction," in *Proc. 22nd Annu. Conf. Comput. Graph. Interact. Techn.*, 1995, pp. 401–408.

[27] R. H. Y. So and M. J. Griffin, "Experimental studies of the use of phase lead filters to compensate lags in head-coupled visual displays," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 26, no. 4, pp. 445–454, Jul. 1996.

[28] H. Himberg and Y. Motai, "Head orientation prediction: delta quaternions versus quaternions," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 39, no. 6, pp. 1382–1392, Dec. 2009.

[29] J. M. Van Verth and L. M. Bishop, *Essential Mathematics for Games and Interactive Applications: A Programmer's Guide*. Boca Raton, FL, USA: CRC Press, 2015.

[30] G. A. Korn and T. M. Korn, *Mathematical Handbook for Scientists and Engineers: Definitions, Theorems, and Formulas for Reference and Review*, Dover, 2000.

[31] *Fraunhofer Heinrich Hertz Institut*. Accessed: Jan. 2, 2019. [Online]. Available: https://www.hhi.fraunhofer.de/

[32] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[33] L. Merritt and R. Vanam. (2006). *x264: A High Performance H. 264/AVC Encoder*. [Online]. Available: http://neuron2.net/library/avc/overview_x264_v8_5.pdf

[34] M. Carbone and L. Rizzo, "Dummynet revisited," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 2, pp. 12–20, Apr. 2010. doi: 10.1145/1764873.1764876.

[35] X. Corbillon, F. De Simone, and G. Simon, "360° video head movement dataset," in *Proc. 8th ACM Multimedia Syst. Conf.*, Jun. 2017, pp. 199–204.


**Tamay Aykut** received the bachelor's and master's degrees in electrical engineering and information technology from the Technical University of Munich, Munich, Germany, in 2013 and 2016, respectively, where he is currently pursuing the Ph.D. degree. In 2016, he joined the Chair of Media Technology, Technical University of Munich, as a Research Associate. His research interests include artificial intelligence, mobile robotics, and the extensive field of computer vision.

**Jingyi Xu** received the B.Sc. degree in electrical engineering and information technology and the M.Sc. degree (Hons.) from the Technical University of Munich, Germany, in 2012 and 2014, respectively. In 2014, she joined the Chair of Media Technology, Technical University of Munich, as a Member of Research Staff. Her research interests focus on physics-based object grasping and manipulation with soft fingers.

**Eckehard Steinbach** received the Ph.D. degree from the Image Communication Group, University of Erlangen–Nuremberg, Germany, in 1999. He studied electrical engineering at the University of Karlsruhe, Germany, the University of Essex, U.K., and ESIEE in Paris. From 1994 to 2000, he was a member of research staff of the Image Communication Group, University of Erlangen–Nuremberg. From 2000 to 2001, he was a Post-Doctoral Fellow of the Information Systems Laboratory, Stanford University. In 2002, he joined the Department of Electrical Engineering and Information Technology, Technical University of Munich, Germany, where he is currently a Full Professor of Media Technology. His current research interests are in the areas of audio-visual-haptic information processing and communication, and networked and interactive multimedia systems.