Technische Universität München
Fakultät für Elektrotechnik und Informationstechnik
Lehrstuhl für Medientechnik

Towards Immersive Telepresence: Stereoscopic 360-degree Vision in Realtime

Tamay Aykut, M.Sc.

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:		Prof. Dr.-Ing. Norbert Hanik
Prüfer der Dissertation:	1.	Prof. Dr.-Ing. Eckehard Steinbach
	2.	apl. Prof. Dr.-Ing. Walter Stechele

Die Dissertation wurde am 13.06.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 28.07.2019 angenommen.

Abstract

The technological advances in immersive telepresence are greatly impeded by the challenges that need to be met when mediating the realistic feeling of presence in a remote environment to a local human user. A server-/client-based architecture can be built to allow human users to immerse themselves into a distant environment upon request. Video, audio, and haptic signals are exchanged over a communication network. In particular, the mediation of omnidirectional visual information is a critical factor that has a sizable impact on the perceived quality of experience. Physically unavoidable delays cause a noticeable lag between head-motion and visual response. Head-mounted displays are widely deployed as an assistive device to increase the level of immersion. The time needed to reflect the human's motion onto the screen is denoted as the motion-to-photon latency. A mismatch between the sensory information from the visual system and the perceived ego-motion of the user provokes the emergence of motions sickness and thus limits the mainstream acceptance and dissemination of such telepresence systems.

Providing a stereoscopic 360° visual representation of the distant scene further fosters the level of realism and greatly improves task performance. Omnistereoscopic vision allows the user to perceive depth information and sense the distant scene in 3D. The realtime acquisition and streaming of omnidirectional vision in stereoscopy is a discerning cutting-edge research topic and still poses major challenges. State-of-the-art technology has primarily developed catadioptric or multi-camera systems to address this issue. Current solutions are bulky, not realtime-capable, and tend to produce erroneous image content due to the stitching processes involved, which are prone to perform poorly for texture-less scenes.

The work presented in this manuscript takes a *vision on-demand* bottom-up approach and creates stereoscopic scene information upon request. A two-camera setup is deployed to mimic the human vision system and augment it with a three degree-of-freedom actuation unit to be able to mirror the user's head-motion. A buffer-based delay-compensation approach is proposed that deploys fisheye cameras to acquire a wider visual field than the viewport size of the user, creating extra image content around the viewport's margin. The additional buffer is created to avoid frozen images despite large end-to-end delays. Instantaneous visual feedback is provided by leveraging the peripheral image content for local delay-compensation until the updated image frame arrives. The delay-compensation approach is extended with a velocity-based field-of-view adaptation technique to decline the onset of motion sickness while maintaining the level of presence. The user's visual field is thereby temporarily reduced for head rotation velocities that exceed a defined threshold.

The final component of the delay-compensation approach is an innovative viewport prediction paradigm. Proper head-motion prediction is performed by adopting a late-fusion strategy that uses a head-orientation-based deep network to predict prospective head-positions. Two deep architectures are presented that leverage spatio-temporal scene information and upgrade the forecasting capabilities. The late-fusion policy is robust against poor lighting conditions or other external disturbances. A generic compensation rate is proposed as a new type of device-agnostic metric to convey the achievable level of delay-compensation. Results show superior performance compared to related work. A mean compensation rate as high as 99.99 % is achieved for investigated latencies between 0.1 s to 1 s.

Kurzfassung

Die Verbreitung und Akzeptanz von Technologien, die realitätsnahe und immersive Telepräsenz ermöglichen, sind zur Zeit dieser Arbeit stark limitiert. Fundamentale Problemstellungen müssen zunächst gelöst werden um die Immersion des Nutzer in die entfernte Umgebung zu erhöhen. Server-/Client-basierte Architekturen werden meist aufgesetzt um eine stabile Kommunikation zwischen dem Nutzer und dem in einer entfernten Umgebung positionierten Teleoperator aufzubauen. Die Übertragung von visuellen, auditiven und haptischen Signalen vermittelt dem Nutzer das realistische Gefühl der Anwesenheit. Besonders der Transfer von visuellen Daten kann einen gravierenden Einfluss auf den Immersionsgrad und die empfundene Präsenz haben. Head Mounted Displays (HMDs) werden meist verwendet um die Wahrnehmung der entfernten Umgebung intuitiver und realitätsnäher zu gestalten. Jedoch führt die unvermeidbare Latenz zwischen dem Server- und Clientsystem zu einer spürbaren Verzögerung zwischen Kopfbewegung und optischer Resonanz. Die Zeit, die benötigt wird eine ausgeführte Kopfbewegung auf dem Display des HMD zu reflektieren wird als die motion-to-photon (M2P) Latenz bezeichnet. Unstimmigkeiten zwischen den sensorischen Informationen des visuellen Systems und der empfundenen Eigenbewegung können im schlimmsten Fall zu Bewegungskrankheit (Motion Sickness) führen und damit das Ende des Telepräsenzerlebnisses bedeuten.

Erwiesener Maßen kann der Immersionsgrad durch das Bereitstellen von stereoskopischen Aufnahmen erhöht werden. Stereoskopie ermöglicht dem Nutzer den räumlichen Eindruck von Tiefe indem den beiden Augen Bilder aus zwei verschiedenen Perspektiven dargestellt werden. Die omnidirektionale Erfassung von stereoskopischen Aufnahmen in Echtzeit ist ein intensiv erforschtes Themengebiet, das noch mit erheblichen Problemstellungen zu kämpfen hat. Der Stand der Technik präsentierte katadioptrische und Multi-Kamera Systeme um diese Herausforderungen zu adressieren. Aktuelle Lösungsansätze sind teuer, sperrig und rechenintensiv. Der Einsatz von mehreren Kameras erfordert das Zusammenführen von partiell überlappenden Sichtfeldern, die v.a. bei Szenen mit geringen Strukturen (meist) fehlerhaft sind.

Die vorliegende Arbeit verfolgt im Vergleich zu gängigen Lösungsnansätzen eine andere Strategie. Statt omnidirektionale Stereoaufnahmen als Ganzes zu erfassen, werden Teilbereiche erst bei Bedarf stereoskopisch aufgezeichnet. Hierfür wird ein Zwei-Kamera System mit einem elektro-mechanischem Aktuator gekoppelt. Das mechatronische System verfügt über drei Freiheitsgrade und ist in der Lage die Kopfbewegungen eines menschlichen Nutzers zu imitieren. Die M2P Latenz ist bei solch einem System besonders kritisch. Um dem

entgegenzuwirken und dennoch die Vorteile eines schlanken Stereosystems ausnutzen zu können, wird ein innovativer Lösungsansatz vorgestellt, der die empfundene Latenz signifikant reduziert. Die Basis des Kompensationsalgorithmus geht aus einem Puffer-basierten Ansatz hervor. Fischaugenobjektive werden eingesetzt um ein größeres Sichtfeld der Szene abzudecken als für die Darstellung des Nutzers benötigt wird. Die dadurch zusätzlich generierten Bildinformation können für eine lokale Kompensation verwendet werden. Sobald der Nutzer seinen Kopf bewegt, kann eine instantane visuelle Resonanz erfolgen. Der Ansatz wird durch eine psychophysisch motivierte Technik erweitert, die das Sichtfeld bei schnelleren Kopfbewegungen kurzzeitig reduziert. Dieses Vorgehen mildert die Entstehung von Motion Sickness bei gleichbleibender empfundener Präsenz. Ein weiterer wesentlicher Bestandteil des Kompensationsalgorithmus ist ein auf künstlicher Intelligenz basierendes Verfahren zur Vorhersage von zukünftigen Kopfbewegungen. Die Voraussage besteht aus drei separat trainierten tiefen Netzen, die erst zu einem späten Zeitpunkt fusioniert werden. Als Grundlage wurde ein tiefes Netz entwickelt, das in der Vergangenheit liegende Orientierungen als Eingabe verwendet. Erweitert wird dieses Modell mit zwei weiteren Prädiktoren, die die räumlichen und zeitlichen Informationen der Szene extrahieren und mit dem orientierungsdaten-basierten Ansatz fusionieren. Die Kompensationrate wird als eine neue Metrik eingeführt um den Grad der Kompensationsfähigkeit zu quantifizieren. Der vorgestellte Ansatz weist im Vergleich zum Stand der Technik signifikante Verbesserungen auf und erzielt durchschnittliche Kompensationsraten von bis zu 99.99 % für untersuchte Latenzzeiten von 0.1 s – 1 s.

Acknowledgements

The work presented in this dissertation was carried out as a member of the academic staff at the Chair of Media Technology (LMT) at the Technical University of Munich. Many people have supported me, personally as well as professionally, during the past years.

First of all, I would like to express my deep gratitude to my supervisor Prof. Eckehard Steinbach. It has been an honor to be part of his team. He provided me with endless support and crucial remarks that helped to shape my final dissertation. His professional attitude and enthusiasm for my conducted research was contagious and inspiring.

Furthermore, I would like to thank Prof. Walter Stechele for agreeing to become the second examiner and Prof. Norbert Hanik for chairing the thesis committee.

Many thanks go also to our industry partners, Dr. Felix Reinshagen, Dr. Rastin Pries, and Dr. Marco Hoffmann for their support during our interesting and fruitful collaborations.

My sincere appreciation is also devoted to my former and current colleagues at LMT. The group has been a source of friendships as well as good advice and collaboration. They have not only helped me professionally through their continued support, but also made my time at LMT special and enjoyable on a personal level. I am particularly grateful to Dr. Clemens Schuwerk and Dr. Nicolas Alt for kick-starting my research with their experience and knowledge. My thanks also go to Mojtaba Karimi who fundamentally contributed to the great success of the MAVI telepresence platform that we built from scratch during many sleepless nights. I highly value my colleagues and students at LMT, I feel however obliged to highlight a few people. Particular thanks go to Jingyi Xu, Dr. Dominik van Opdenbosch, Dr. Christoph Bachhuber, and Dmytro Bobkov for interesting discussions and important impulses for my research. I am also grateful to all my students, especially Christoph Burgmair, Basak Gülecyüz and Stefan Lochbrunner, who contributed to our publications.

My appreciation also goes to the professional administrative support provided by Marta Giunta, Simon Krapf, and Martina Schmid. Special thanks are devoted to my mentor, Dr. Martin Maier, for his ongoing spiritual support and great mentorship throughout my Ph.D.

Last but not least, I would like to express my deepest gratitude to my family, in particular to my beloved parents Arife Güler and Turgay Aykut, my brother Timucin Aykut, Andrea Wojner and my two best friends Emin Rizovic and Christoph Nowak, who always encouraged me and believed in my abilities. Without their endless support and valuable advice I would not be where I am today.

Tamay Aykut, Munich, Mai 17, 2019

Contents

Notation

Abbreviations

Abbreviation	Description	Definition
2D	Two-dimensional	page 7
3D	Three-dimensional	page 7
AUC	Area under the ROC Curve	page 88
AVC	Advanced video coding	page 27
CABAC	Context-adaptive binary arithmetic coding	page 39
CCD	Charge-coupled device	page 8
CMP	Cube map projection	page 27
CNN	Convolutional neural network	page 97
DASH	Dynamic adaptive streaming over HTTP	page 29
DCVS	Delay-compensation vision system	page 33
(D)IBR	(Depth) Image-based rendering	page 17
DoF	Degree-of-freedom	page 4
ERP	Equirectangular projection	page 27
FFN	Feed-forward neural network	page 81
GPU	Graphics processing unit	page 78
GRU	Gated Recurrent Unit	page 5
HEVC	High-efficiency video coding	page 27
HMD	Head-mounted-display	page 3
IDP	Inter-pupillary distance	page 23
IMU	Inertial measurement unit	page 35
INV	Involvement	page 67
IPQ	Igroup presence questionnaire	page 67
KF	Kalman filter	page 72
LR	Linear regression	page 72
LSTM	Long short-term memory	page 31
M2P	Motion-to-photon	page 3
MAE	Mean absolute error	page 39
MAVI	Machine vision and interaction	page 33
MOS	Mean opinion score	page 67
MSE	Mean square error	page 81
MVP	Multi-viewpoint	page 20
PID	Proportional–integral–derivative	page 77
PRES	General "sense of being there"	page 67
PSNR	Peak-signal-to-noise-ration	page 104
PTR-U	Pan-tilt-roll-unit	page 4
PT-U	Pan-tilt-unit	page 33
QEC	Quality emphasized center	page 29
QER	Quality emphasized region	page 29

Abbreviation	Description	Definition
QoE	Quality-of-experience	page 2
QP	Quantization parameter	page 101
REAL	Experienced Realism	page 67
ReLU	Rectified linear units	page 83
RMSE	Root mean square error	page 39
RNN	Recurrent neural networks	page 83
ROC	Receiver operating characteristics	page 88
RTSP	Realtime streaming protocol	page 38
SP	Spatial presence	page 67
SSQ	Simulator sickness questionnaire	page 67
SVP	Single-viewpoint	page 20
TCP	Transmission control protocol	page 38
UDP	User datagram protocol	page 38
VR	Virtual reality	page 22
WS-PSNR	Weighted-to-spherically-uniform PSNR	page 104

Scalars and vectors

x	Scalar
$\boldsymbol{x}$	Vector
$\boldsymbol{X}$	Matrix
$\|x\|$	Absolute value of scalar x
$\|\boldsymbol{x}\|$	Euclidean norm of vector $\boldsymbol{x}$

Subscripts and superscripts

x_w	Signal x associated with the world coordinate system
x_c	Signal x associated with the camera coordinate system
$x_{c,i}$	Signal x associated with the coordinate system of camera i
x'	Projected image point of x
$\bar{x}$	Mean of x
$\hat{x}$	Estimated/predicted value of x

Symbols

f	Focal length
fov	Field-of-view

Chapter 1

Introduction

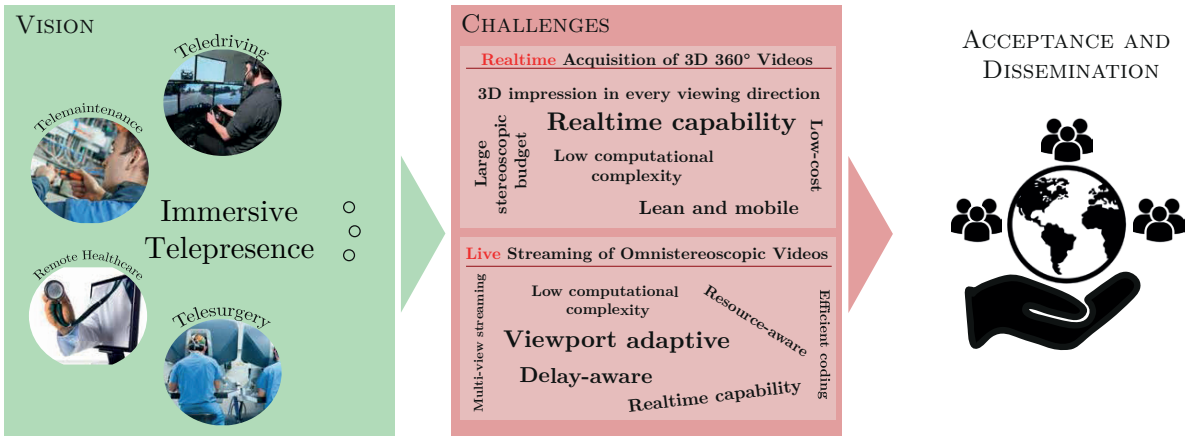


Figure 1.1: Overview of potential applications that greatly benefit from high-fidelity technologies that target immersive telepresence. The focus of this work is put on the mediated perception of visual information of a remote scene. Besides the communication delay, there exist various other latencies that contribute to the overall latency and strongly affect the immersive experience of the user. Especially omnidirectional stereoscopic vision, which allows the perception of depth and has positive implications on task performance, is a challenging issue that needs to be addressed to provide high-fidelity telepresence. Some of the accompanying challenges are grouped and highlighted to underline the problematic path of such technologies to be accepted and widely spread around the globe.

Immersive telepresence describes the ability to virtually transport a human from one local place to another in an instant. Provisioning such technology in high-fidelity finds promising usage in numerous forthcoming applications such as teledriving, telemaintenance, remote healthcare, minimally invasive surgery, and medical assistance as depicted in Figure 1.1. Telepresence allows the human to enlarge his/her workspace to remote locations bridging obstacles such as distance or danger, which is particularly valuable for rescue or exploration missions in hazardous environments. Video, audio, and, in case of remote manipulation capabilities, haptic signals are transmitted over a communication network. Multiple studies exposed a strong correlation between the feeling of presence and the quality level of visual, auditory, and haptic data [10]–[12]. Especially, the visual response turned out to be of great significance. An advanced sense of presence is achieved for high-quality visual representations of the remote scene [12], [13].

That is why this thesis places particular emphasis on the exchange of visual data for telep-

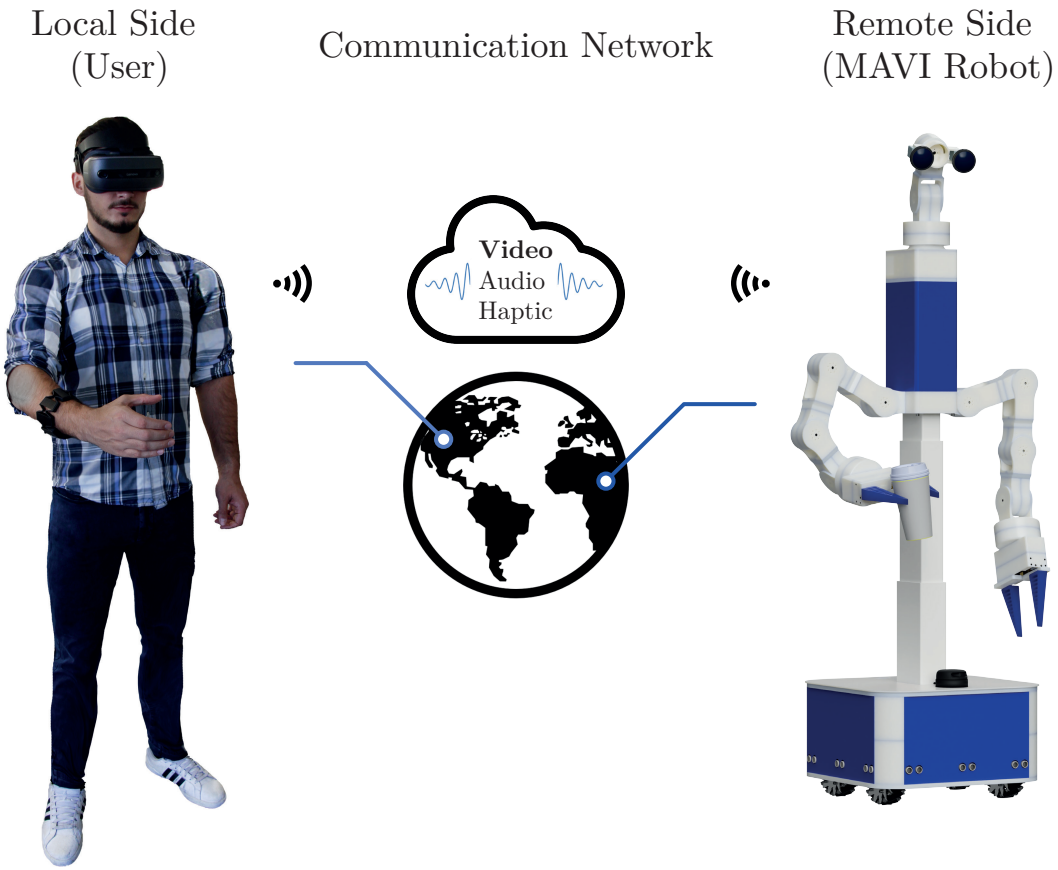


Figure 1.2: Overview of the telepresence scenario that is designed to investigate the mediated perception of visual data. A server/client-based setup is built to establish a connection between a user and the MAVI telepresence robot [3] as teleoperator. The sensor head of the MAVI telepresence platform is developed and used to provide a omnidirectional stereoscopic view of the remote scene. A three degree-of-freedom pan-tilt-roll-unit augments the two-camera setup to mirror the head motions of the user at the local side.

resence systems, which tends to be the bottleneck of such systems. For practical experiments, the MAVI robot platform [3] is developed and equipped with sensors, which capture visual and auditory information about the remote environment. A telepresence scenario is created as is shown in Figure 1.2. The overall goal is to mediate the visual perception of a temporally or spatially distant real environment to a human user being at another location. Immersive telepresence is accomplished by establishing a connection between a master- (i.e. the user) and a client-system (here the MAVI telepresence platform), which exchange video and audio signals over a communication network. The bidirectional transmission of data is part of a global control loop between the user and the remote operator and burdens stiff requirements on the communication network. Such a communication introduces ineluctable delays, which are subject to the distance between the user and the operator. The end-to-end latency decisively influences the stability of such a system and heavily impairs the user's quality-of-experience (QoE). It does not only lessen the immersive effect but is also detrimental to the visual comfort. While delayed audio feedback may attenuate the sense of presence, a stalled visual response actively provokes the emergence of motion sickness, especially when

received through head-mounted-displays (HMDs). Motion sickness is the main reason why the acceptance and dissemination of omnidirectional (360°) telepresence systems is sharply limited. The time required to reflect the human's motion entirely and display the corresponding view onto the screen is referred to as the motion-to-photon (M2P) latency. If the M2P latency exceeds a threshold value of about 20 ms [14], the user is prone to suffer from motion sickness and is likely to interrupt the telepresence process [15]–[17]. It is of vital significance that the perceived ego-motion of the user complies with the sensory impressions from the visual system, the vestibular system, and the non-vestibular proprioceptors [18]. If these demands are not fulfilled correctly and the user's expectation based on prior observations does not comply with these inputs, it is unavoidable that the user will experience visual discomfort and hence suffer from motion sickness and nausea [15], [17], [19]. Smart solutions are highly appreciated to prevent such phenomena.

One approach to address these issues is to record and send a 360°×180° video of the entire scene. Receiving the image content through an HMD provides the user with instantaneous visual feedback and hence a low M2P latency given the immediate access to the entire visual representation of the distant environment. The user is then able to freely explore the remote scene without concerning about the present delay. There already exist many consumer products that acquire and stream monoscopic 360° videos. The same image content is thereby displayed to both eyes on the HMD. The immersive experience, however, significantly enhances when providing stereoscopic (3D) vision [20], [21]. Stereoscopic vision provides separate images from different vantage points for each eye. The different perspectives facilitate the perception and sensation of depth information within the scene. Binocular vision is known to promote task performance, particularly for indoor applications given the fact that the human stereo vision is limited to approximately 19 m [22]. Prior art proved that stereoscopic vision shows great advantages over monoscopic viewing [20], [21]. Task performance was conducted faster and less erroneous under stereoscopic viewing conditions [20], [21]. The flawless acquisition and streaming of omnistereoscopic panorama video poses major challenges and remains an open research topic. These challenges are the reason for the hampered acceptance and dissemination of 3D vision-enabling telepresence systems. Previous work mainly deployed catadioptric systems or multi-camera arrangements. These systems tend to be bulky, expensive, and often far away from being realtime capable. Precise calibration and positioning are required to conduct correct stitching of partial scene views. The quality and realism of the computationally demanding stitching process is highly subject to the features and structures within the scene and is hence prone to erroneous outputs. Flawed stitching might result in critical distortions within the footage, which are even magnified due to the HMD's lens structure and the small display-to-eye distance. Not only the acquisition but also the streaming of omnistereoscopic footage needs special attention. Sending two complete 360° video sequences does not only claim large parts of the communication capacity but also appears to be dispensable as large peripheral segments outside of the current viewport are not even shown to the user. Viewport adaptive, delay-, and resource-aware streaming strategies are therefore highly favored.

The state-of-the-art addressed these challenges through a top-down approach by first capturing a full stereoscopic snapshot of the remote scene and then processing the image content for viewport selection. The work presented in this manuscript takes an opposite approach. A minimum amount of two cameras are deployed and augmented with an electro-mechanical pan-tilt-roll-unit (PTR-U) as part of MAVI's sensor head as shown in Figure 1.2. In this way, a *vision on-demand* strategy is opted for, which reduces the computational and financial burden substantially. The PTR-U can mimic the three degrees-of-freedom (DoF) of the human's head motion, which is continuously recorded through an orientation sensor embedded in the HMD and provides a $360^\circ \times 180^\circ$ visual impression of the distant environment. Depending on the network delay though, it takes a certain amount of time to replicate the head motion and send the updated image frames back to the user. The accumulation of latencies within the whole processing pipeline causes incongruities between ego-motion and visual response. The delay between the head-motion and the visual response was introduced as the M2P latency and identified as detrimental to the visual comfort when exceeding a certain threshold. Merely deploying an augmented stereo-camera setup is thus not sufficient. The resulting sizable M2P latency provokes the emergence of motion sickness and the discontinuation of the telepresence session in question. Notwithstanding, such a hardware system comes along with desirable properties. Its lean and mobile structure allows binocular vision in every viewing direction in realtime and benefits from a large stereoscopic budget. To be able to exploit the beneficial features of such a system, sophisticated algorithmic solutions are highly desired to overcome the QoE-limiting M2P latency.

Within the scope of this work, a modified PTR-U-based two-camera setup is designed and upgraded with an artificial intelligence-based delay-compensation algorithm that allows the provision of instantaneous visual feedback for highly immersive telepresence. The proposed delay-compensation approach relies on three individual components.

The first part is a buffer-based delay-compensation technique that is premised on the acquisition of a larger visual field than is actually needed for display. Equidistant fisheye cameras are deployed that have a wider field-of-view (fov_c) compared to the HMD's viewport size (fov_h). Picturing only a subset of the captured footage facilitates the residual image content to be used for local delay-compensation until the updated frame eventually arrives. The so-called *compensation rate* is introduced as a qualitative measure to convey the achievable level of delay-compensation and is highly subject to the available buffer size and the present delay in question.

The second algorithmic component describes a velocity-based dynamic field-of-view adaptation technique that is motivated by the characteristics of the human eye. The field-of-view is thereby temporarily reduced during rapid head rotations and increased again for slower motions. Pixel extrapolation techniques are investigated to artificially fill the peripheral of the viewport. Subjective studies confirm that the proposed approach does not affect the feeling of presence. Instead, it positively impinges on the achievable degree of delay-compensation and supports the reduction of motion sickness.

The third element is built upon a novel viewport prediction paradigm. Rather than sending the current head position, the prospective orientation after the present delay is estimated and sent to the remote camera system. A probabilistic and various machine learning-based head-motion prediction techniques are developed and investigated. The final, best performing prediction methodology is based on a deep fusion network that uses not only the head orientation data but also spatio-temporal information within the scene as a source of information for proper viewport forecasting. The networks use an interleaved structure of stacked Gated Recurrent Units (GRUs) and convolution layers to extract the most distinct features at different granularities. Two head-motion datasets are leveraged for proper evaluation based on qualitative measures. Merging all individual solutions to one delay-compensation algorithm achieves a mean delay-compensation rate of **99.99 %** for investigated communication delays between 0.1 s to 1 s substantially improving the visual comfort of the user.

The key contributions of the work presented in this manuscript can be summarized as follows:

- A customized **sensor head** is developed that deploys two high-resolution, equidistant fisheye cameras in a parallel configuration attached to a 3 DoF electro-mechanical unit that is able to precisely mirror the user head's pan, tilt, and roll rotations.

- A geometric, **buffer-based delay-compensation** approach is presented that leverages the wider field-of-view of the fisheye cameras to capture a larger field of vision than is actually displayed on the HMD to the user. Extra image content is thereby created around the displayed viewport. The residual imagery is exploited for local delay-compensation until the updated frame arrives to ensure instantaneous visual response for the requested head position. A generic, mathematically-founded delay-compensation model is presented that is agnostic to the underlying camera system and works both for perspective and fisheye cameras. The so-called delay compensation rate is introduced as a novel metric to reflect the achievable level of delay-compensation.

- A psycho-physically motivated **velocity-based field-of-view adaptation** technique is proposed to further increase the degree of retrievable delay-compensation. Depending on the current head-motion velocity, the displayed visual field of the user is temporarily decreased. The adaptive constriction process momentarily enlarges the buffer size, which results in a higher level of latency compensation, especially for fast rotations. This approach is motivated by the characteristics of the human eye. Subjective studies are conducted to prove its validity.

- A probabilistic and various **machine-learning-based head-motion prediction** methods are proposed to further optimize the degree of delay compensation. Multiple deep neural network architectures are applied to improve the forecasting accuracy of prospective head orientations. Densely sampled head-motion data is merged with spatial and temporal information within the scene employing saliency and motion maps, respectively, for superior viewport prediction. A late-fusion strategy is presented that is robust against external, disruptive factors and is able to perform superbly even for

lousy lighting conditions. The proposed prediction paradigm excels the state-of-the-art and yields remarkable mean compensation rates of **99.99 %** for investigated delays ranging from **0.1 s to 1 s**.

Thesis outline

Chapter 2 introduces the theoretical background needed to understand the proposed concepts and the decisions made within the scope of this work. The proposed methodology is confronted with related work. The main advantages are discussed and compared to the state-of-the-art.

The experimental framework is presented in **Chapter 3**. The developed prototype of MAVI's sensor head along with its hardware specifications is briefly revisited to better understand the presented methodologies. A subsequent section is devoted to present the deployed head motion datasets as well as the utilized metrics, which facilitate a reproducible validation process.

Chapter 4 introduces the delay-compensation approach and establishes a generic, algebraic formulation of the compensation rate metric that is agnostic to the underlying camera system and is valid both for perspective and (equidistant) fisheye cameras.

Chapter 5 analyzes the effects of the velocity-based, temporary reduction of the visual field on the subjective sense of presence as well as the retrievable degree of delay-compensation. The mathematical model of the delay-compensation approach is algebraically modified to incorporate the dynamic field-of-view adaptation technique.

The deep-learning-based, semantic viewport prediction approach that fuses head orientation data with spatio-temporal scene information is detailed in **Chapter 6**. The proposed deep fusion paradigm is compared to prior art and evaluated by means of qualitative measures for real head-motion profiles.

Chapter 7 concludes this work by summarizing the most distinct outcomes of this manuscript, discussing their limitations, and briefly broaching some potential conceptual improvements for future work.

Parts of this manuscript have been published in international peer-reviewed scientific journals [1], [2] and conferences [4]–[6].

Chapter 2

Background and Related Work

This chapter introduces the relevant background needed to better understand the proposed methodologies within the scope of this work. Prior art that technically relates to this work is surveyed and discussed. Section 2.1 presents the basics of projective view geometry and covers topics from image synthesis to omnistereoscopic panorama acquisition. Assets and drawbacks thereof are briefly debated to better comprehend the strategical decisions made for this work. Section 2.5 deals with state-of-the-art methodologies that aim to optimally process, stream, and visualize immersive visual content. The last section delves into the area of head-motion/viewport prediction techniques and highlights their limitations and applicability for telepresence applications.

2.1 Projective View Geometry

2.1.1 Perspective Projection

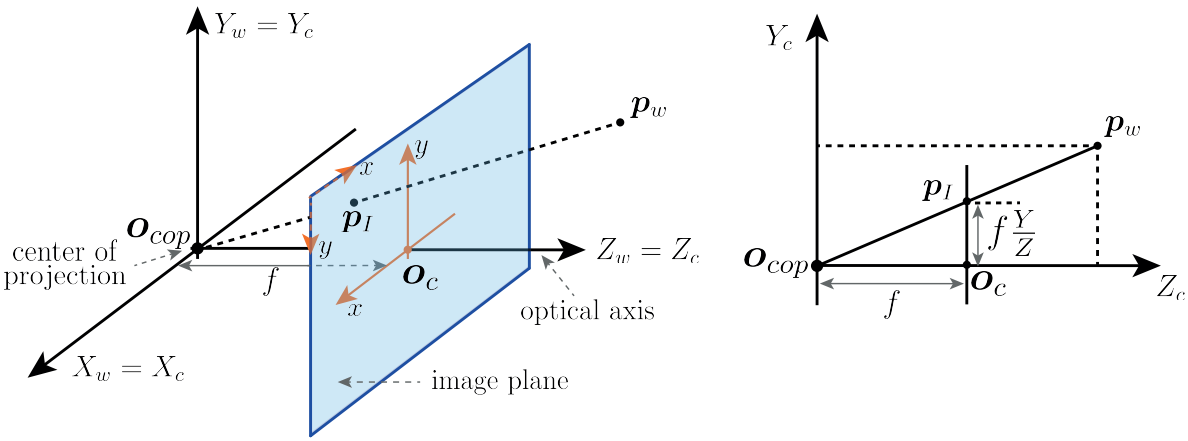


Figure 2.1: Left: Schematic modeling of perspective projection of a 3D point $\boldsymbol{p}_W$ in the world coordinate system to a 2D point $\boldsymbol{p}'$ in the image plane. Right: A 2D section of the projection scheme is shown to visualize the relation between 2D image points and 3D world points.

The overall goal of any camera projection scheme for image formation is to map a three-dimensional (3D) scene to a two-dimensional (2D) representation thereof. The perspective projection, also called central projection, is based on the well-known pinhole model that as-

sumes a lens-less, tiny aperture ignoring effects such as focus and lens thickness [23]. For every 3D point $\boldsymbol{p}_w = [X \ Y \ Z]^T$ lying within the view frustum of the camera, the corresponding ray passes through the center of projection $\boldsymbol{o}_{cop}$, which is located at the origin of the camera's coordinate system and intersects thereby with a 2D point $\boldsymbol{p}' = [x \ y]^T$ on the image plane $\boldsymbol{I}$. The camera coordinate system and the reference world coordinate frame are aligned to simplify the geometric modeling of the perspective projection and the visualization of a 2D slice thereof as is shown in Figure 2.1. The image plane is always at a certain distance away from $\boldsymbol{o}_{cop}$. This distance is known as the *effective focal length* f. The optical axis is usually defined to be the Z-axis of the camera coordinate system. The optical axis hits the image plane at the image center $\boldsymbol{o}_c = [o_x \ o_y]^T$, which is referred to as the *principle point*. The geometric system description visualized in Figure 2.1 allows us to define the projective mapping of a 3D world point $\boldsymbol{p}_w$ to a 2D image representation (up to scale) as follows:

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} \mapsto \lambda \begin{pmatrix} x \\ y \end{pmatrix} = \lambda \begin{pmatrix} f \cdot \frac{X_c}{Z_c} \\ f \cdot \frac{Y_c}{Z_c} \end{pmatrix}, \tag{2.1}$$

with $\lambda = Z_c$ being a scalar value representing the (unknown) depth value of the point $\boldsymbol{p}_w$. The homogeneous coordinates for the world and image points are used to express the perspective projection as a linear mapping in terms of a matrix-vector multiplication:

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} f \cdot X_c \\ f \cdot Y_c \\ Z_c \end{pmatrix} = Z_c \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\boldsymbol{P}} \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix}. \tag{2.2}$$

The transformation matrix $\boldsymbol{P}$ can be decomposed into the *intrinsic parameter matrix* $\boldsymbol{K}$, also called the *calibration matrix*, and the *canonical projection matrix* $\boldsymbol{P}_\pi$, which performs the actual perspective projection [23]:

$$\boldsymbol{P} = \boldsymbol{K}\boldsymbol{P}_\pi, \quad \text{with} \quad \boldsymbol{K} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix}, \quad \text{and} \quad \boldsymbol{P}_\pi = \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} = [\boldsymbol{I} \mid \boldsymbol{0}]. \tag{2.3}$$

In its current definition, the pinhole camera model assumes an ideal camera with perfectly squared pixel dimensions. In charge-coupled device (CCD) cameras, however, pixels are not necessarily equally scaled in both axial directions [24]. To account for non-square pixels, the scale factors s_x and s_y are introduced to specify the number of pixels per unit distance in the $x-$ and $y-$ direction [23], [24]. In the case of non-rectangular pixels, a further scaling term s_θ, referred to as the *skew factor*, can be included.

The origin of the image plane is currently located at the principle point $\boldsymbol{o}_c$. It is a commonly accepted convention though to redefine the image's origin to be at the upper left corner as depicted in Figure 2.1. This translatory adaptation can directly be incorporated into the camera's intrinsic parameter matrix with respect to the scaled pixel coordinates. The general form of the calibration matrix, particularly for CCD cameras, can thus be derived as

follows:

$$\boldsymbol{K} = \begin{bmatrix} s_x & s_\theta & o_x \\ & s_y & o_y \\ & & 1 \end{bmatrix} \cdot \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} = \begin{bmatrix} s_x f & s_\theta f & o_x \\ & s_y f & o_y \\ & & 1 \end{bmatrix}. \tag{2.4}$$

2.1.2 Rigid Body Motion

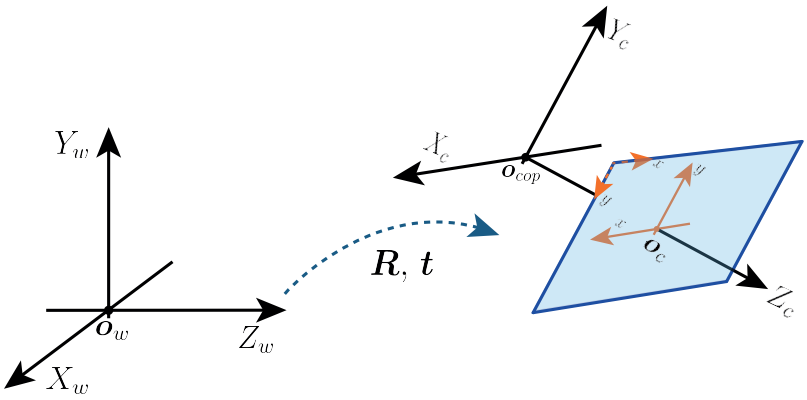


Figure 2.2: Rigid body transformation (rotation and translation) of the camera frame with respect to the world coordinate system (reference frame).

The camera's coordinate frame is not always co-aligned with the world coordinate system. In real-world applications, the camera's location and orientation changes within the 3D space. This displacement is expressed as a rigid body motion considering a transformation that consists of a rotation $\boldsymbol{R}$ and a translation $\boldsymbol{t}$ as is depicted in Figure 2.2:

$$\boldsymbol{p}_c = \boldsymbol{R} \cdot \boldsymbol{p}_w + \boldsymbol{t}, \tag{2.5}$$

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}. \tag{2.6}$$

Expressing both the rotation and the translation in terms of the homogeneous representation allows the transformation $\boldsymbol{T}$ to be defined as a single matrix-vector multiplication:

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix}}_{\boldsymbol{T}} \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}. \tag{2.7}$$

There exist various representations to parameterize rotation matrices. The most common schemes are Euler angles, quaternions, and the Rodrigues formula. Euler angles are characterized by three angles denoted as the yaw θ, the pitch ϕ, and the roll ψ angle. Any orientation is composed of three individual rotations around the respective axis of the coordinate system. The overall rotation matrix is obtained by multiplying the individual rotations

matrices. The three rotation matrices for yaw θ, pitch ϕ, and roll ψ are defined as follows:

$$\boldsymbol{R}_\theta = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}, \boldsymbol{R}_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix}, \boldsymbol{R}_\psi = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.8}$$

The overall rotation can then be described as:

$$\boldsymbol{R} = \boldsymbol{R}_\psi \cdot \boldsymbol{R}_\phi \cdot \boldsymbol{R}_\theta, \tag{2.9}$$

with $\boldsymbol{R}$ being orthogonal ($\boldsymbol{R}\boldsymbol{R}^T = \boldsymbol{I}$). Changing the order of the matrix multiplication leads to different rotations. Euler angles do not consistently provide a unique description of rotations. The ambiguous character is caused by the so-called *gimbal lock*, where a degree-of-freedom is lost.

Three-dimensional rotations can also be described by unit quaternions, which are a generalization of complex numbers and do not suffer from the problem of gimbal lock [23]. A unit quaternion $\boldsymbol{q}$ can be represented in several ways:

$$\boldsymbol{q} = w + q_1 i + q_2 j + q_3 k, \tag{2.10}$$

$$\boldsymbol{q} = (w \;\; q_1 \;\; q_2 \;\; q_3)^T, \tag{2.11}$$

$$\boldsymbol{q} = (s, \boldsymbol{v}), \text{with} \tag{2.12}$$

$$s = w \quad \text{and} \quad \boldsymbol{v} = (q_1 \;\; q_2 \;\; q_3)^T. \tag{2.13}$$

Rotations with quaternions are simple to compute, as a multiplication of any two quaternions is analogous to a multiplication of two complex numbers given that the multiplication $ij = -ji$ is anticommutative [24]. A vector $\boldsymbol{p}_q = (0, \boldsymbol{p})$ is rotated by a quaternion $\boldsymbol{q}$ by multiplying $\boldsymbol{q}$ and its inverse $\boldsymbol{q}^{-1} = (s, -\boldsymbol{v})$ from both sides:

$$\boldsymbol{p}'_q = \boldsymbol{q} \cdot \boldsymbol{p}_q \cdot \boldsymbol{q}^{-1}. \tag{2.14}$$

The Rodrigues rotation formula is an alternative way to represent a 3D rotation. It is an efficient way to rotate a 3D vector $\boldsymbol{x}$ by an angle α around a given axis $\boldsymbol{k}$ [25]:

$$\boldsymbol{R} \cdot \boldsymbol{x} = \cos(\alpha) \cdot \boldsymbol{x} + (1 - \cos(\alpha)) \cdot \frac{\boldsymbol{k} \otimes \boldsymbol{k}}{||\boldsymbol{k}||^2} \cdot \boldsymbol{x} + + \frac{\sin(\alpha)}{||\boldsymbol{k}||} \cdot [\boldsymbol{k}]_\times \cdot \boldsymbol{x}, \tag{2.15}$$

2.1.3 Projective Geometry

The benefit of homogeneous coordinates is that they allow us to express sequential computation steps as one matrix-vector multiplication. All processing phases including the rigid body motion, the perspective projection, and the camera intrinsics can be summarized into a single matrix $\boldsymbol{M}$:

$$Z\boldsymbol{p}' = Z \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \boldsymbol{K} \cdot \boldsymbol{P}_\pi \cdot \boldsymbol{T} \cdot \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \boldsymbol{K} \cdot [\boldsymbol{R} \,|\, \boldsymbol{t}] \cdot \boldsymbol{p}_w = \boldsymbol{M} \cdot \boldsymbol{p}_w. \tag{2.16}$$

The perspective projection and the Euclidean transformation are often combined and written as $[\boldsymbol{R} \,|\, \boldsymbol{t}]$.

2.1.4 Stereo View Geometry

Stereo camera configurations are frequently used in many applications such as 3D reconstruction or visual odometry as it offers a rich description of the 3D environment in a passive manner. Depth information or image pair correspondences can easily be computed by means of triangulation. Stereoscopic cameras are also applied in virtual or remote reality applications to provide the human user the perception of depth by the use of binocular vision. Due to its significance in visual computing, a brief overview of stereo camera setups and the mathematical relation thereof is given.

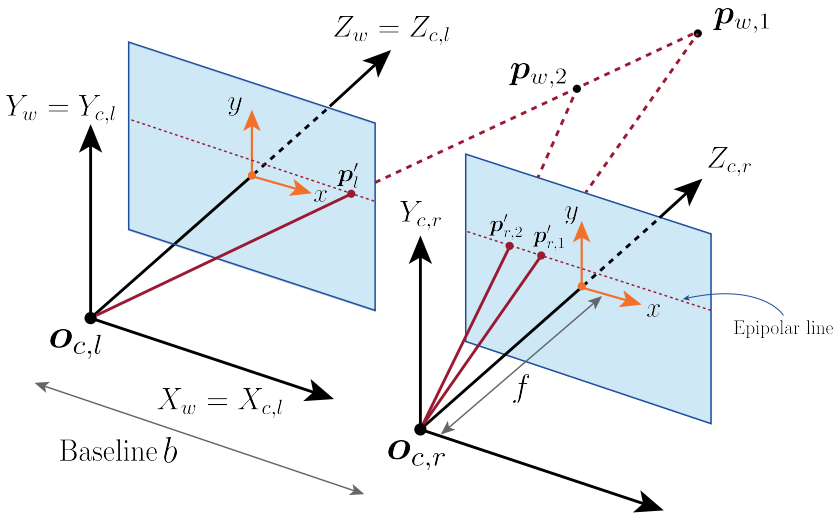


Figure 2.3: Canonical stereo configuration. The left and right camera are parallel to each other and are translated in X-direction by the baseline b. Any point in the 3D space that is projected to both images lies on a horizontal line (epipolar line) with $y_l = y_r$.

The general motion of two cameras with respect to the world coordinate system and the relative motion between them is described via a rigid body motion. The origin of the left camera is set to be located at the world coordinate system:

$$\boldsymbol{p}_{c,l} = \begin{pmatrix} X_{c,l} \\ Y_{c,l} \\ Z_{c,l} \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{R}_{c,l} & \boldsymbol{t}_{c,l} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \cdot \boldsymbol{p}_w = \boldsymbol{p}_w. \tag{2.17}$$

Figure 2.4 shows a particular case of a stereo camera arrangement assuming a parallel configuration with a pure translation in X-direction. The distance between the two cameras is denoted as the baseline b. The relative motion of the right camera with respect to the left one can hence be described as follows:

$$\boldsymbol{p}_{c,r} = \begin{bmatrix} \boldsymbol{R}_{c,r} & \boldsymbol{t}_{c,r} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \cdot \boldsymbol{p}_w = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \cdot \boldsymbol{p}_{c,l} = \begin{pmatrix} X_{c,l} - b \\ Y_{c,l} \\ Z_{c,l} \\ 1 \end{pmatrix}. \tag{2.18}$$

Assuming two ideal cameras without a change in the internal parameters, the relationship between the two projected image points (pixels) that represent the same point in the 3D space

can be computed with:

$$\boldsymbol{p}'_r = \boldsymbol{K} \cdot \boldsymbol{P}_\pi \cdot \boldsymbol{p}_{c,r} = \boldsymbol{p}'_l + \begin{pmatrix} -fb & 0 & 0 \end{pmatrix}^T, \tag{2.19}$$

$$Z_w \begin{pmatrix} x_r \\ y_r \\ 1 \end{pmatrix} = Z_w \begin{pmatrix} x_l \\ y_l \\ 1 \end{pmatrix} + \begin{pmatrix} -fb \\ 0 \\ 0 \end{pmatrix}. \tag{2.20}$$

These equations can be used to calculate the depth value (Z_w) of points in the 3D environment that can be seen by both cameras:

$$Z_w = \frac{bf}{x_l - x_r} = \frac{bf}{D}, \quad \text{with } y_r = y_l. \tag{2.21}$$

D is denoted as the disparity and describes the difference in pixel location of an object that

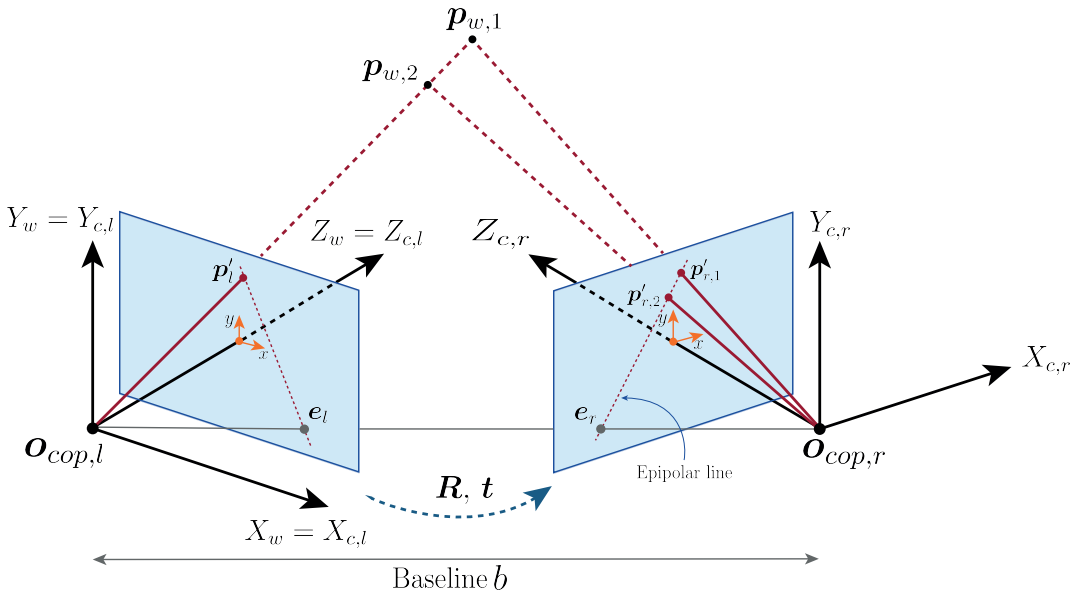


Figure 2.4: Schematic overview of a general stereo camera configuration. The cameras are no longer parallel to each other. The relative transformation consists of rotation and translation. Any point in the 3D space that is projected onto both images lies on the epipolar lines, which are no longer horizontal.

is seen by the left and the right camera. The disparity is inversely proportional to the depth value. Pixel correspondences are hence needed to compute the depth. The canonical stereo configuration depicted in Figure 2.3 benefits from an efficient matching as the search for point correspondences can be restricted to horizontal lines (see Equation (2.21) with $y_r = y_l$) rather than covering the entire image plane.

These relations account, however, for the special case, where the cameras are ideal and purely translated in X-direction, which does not hold for real-world applications. A more generic policy can be deduced by considering the intrinsic parameters of the cameras and an unrestricted rigid body motion between them. Figure 2.4, for instance, shows a stereo setup with inward-looking cameras, which induce larger overlapping views. By considering such a setup, the calibrated camera models can be stated as:

$$\boldsymbol{M}_l = \boldsymbol{K}_l \left[\boldsymbol{I} \,|\, \boldsymbol{0} \right], \quad \boldsymbol{M}_r = \boldsymbol{K}_r \left[\boldsymbol{R} \,|\, \boldsymbol{t} \right]. \tag{2.22}$$

The term $\boldsymbol{K}_{\{l,r\}}$ denotes the calibration matrix for the left and the right camera, respectively. Image point correspondences need to be found to calculate the depth values using triangulation. Any image pair correspondence ($\boldsymbol{p}'_l$, $\boldsymbol{p}'_r$) that represents the same point $\boldsymbol{p}_w$ in the 3D space needs to satisfy the following relation [26]:

$$\boldsymbol{p}'^T_l \boldsymbol{F} \boldsymbol{p}'_r = 0, \tag{2.23}$$

with $\boldsymbol{F}$ being the fundamental matrix. The intrinsic and extrinsic parameters of the camera models specified in Equation (2.22) can be used to compute $\boldsymbol{F}$ according to [24]:

$$\boldsymbol{F} = [\boldsymbol{K}_r \boldsymbol{t}]_\times \boldsymbol{K}_r \boldsymbol{R} \boldsymbol{K}_l^{-1} = \boldsymbol{K}_r^{-T} \boldsymbol{R} \boldsymbol{K}_l^T \left[\boldsymbol{K}_l \boldsymbol{R}^T \boldsymbol{t}\right]_\times . \tag{2.24}$$

The mapping from one image point to the other can then be expressed as [24]:

$$\boldsymbol{p}'_r = \boldsymbol{K}_r \boldsymbol{R} \boldsymbol{K}_l^{-1} \boldsymbol{p}'_l + \frac{1}{Z_w} \boldsymbol{K}_r \boldsymbol{t}. \tag{2.25}$$

However, the computational cost for stereo matching algorithms that are supposed to work for arbitrary camera configurations is high. The correspondence search can cover the whole image plane. The most common method is to apply *stereo rectification* to reduce the search space for image point correspondences from two dimensions to one. This is done by artificially transforming the present stereo configuration to the parallel setup shown in Figure 2.4, which is referred to as the *canonical stereo setup* [24]. The reduction to a horizontal line search results in an enormous speed-up and simplification of the depth calculation. After rescaling the images to account for different focal lengths, the depth can be easily computed according to Equation (2.21). State-of-the-art stereo rectification approaches include [27]–[29].

2.1.5 Camera Distortions

The camera model introduced in Subsection 2.1.1 is based on the pinhole model for ideal cameras without distortions. Real cameras, however, are equipped with lenses that come with geometrical distortions due to imperfections in the design and the assembly thereof. A distorted image point (x_d, y_d) can be expressed as the sum of distortion-free image coordinates (x, y) and distortion values $(\mathcal{D}_x, \mathcal{D}_y)$ in $x-$ and $y-$direction, respectively:

$$x_d = x + \mathcal{D}_x, \tag{2.26}$$

$$y_d = y + \mathcal{D}_y. \tag{2.27}$$

The prior art revealed a wide range of methodologies to model lens distortions [30]–[33]. One common way is to categorize the geometrical distortions into radial, tangential (or decentering), and thin prism distortions [31].

Radial distortion in $x-$ and $y-$ direction ($\mathcal{D}_{x,rad}, \mathcal{D}_{y,rad}$) comes from the erroneous radial curvature of the lens and can be described with [31]:

$$\mathcal{D}_{x,rad} = x(K_1 r^2 + K_2 r^4), \tag{2.28}$$

$$\mathcal{D}_{y,rad} = y(K_1 r^2 + K_2 r^4), \tag{2.29}$$

with K_1 and K_2 being the radial distortion coefficients. r is the distance from the undistorted point (x, y) to the center of radial distortion (principal point). Figure 2.5 illustrates the effect of radial symmetric distortion. Negative radial distortions are denoted as Barrel and positive as Pincushion distortions, respectively.

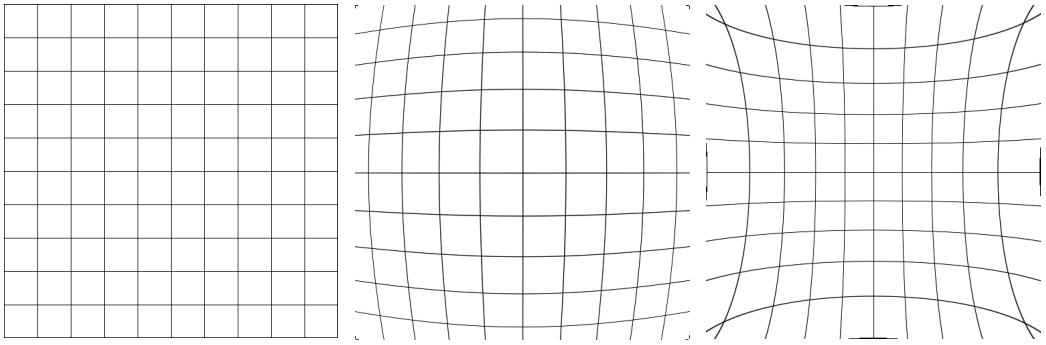


Figure 2.5: The apparent effect of radial distortions on a grid image (left). The barrel distortion (middle) is a convex distortion that bends off-centered straight lines towards the edges of the image. The barrel distortion usually occurs when capturing images with wide-angle or fisheye lenses. The pincushion distortion (right) behaves contrary to the barrel distortion. Off-centered straight lines are bent inwards. The pincushion distortion is a concave aberration and mainly occurs in telephoto lenses.

Tangential distortion ($\mathcal{D}_{x,tan}, \mathcal{D}_{y,tan}$) arises from the decentering of the lens and other optical elements. The decentering coefficients P_1 and P_2 are used to express the tangential distortion as [31]:

$$\mathcal{D}_{x,tan} = P_1(3x^2 + y^2) + 2P_2xy, \tag{2.30}$$

$$\mathcal{D}_{y,tan} = P_2(3y^2 + x^2) + 2P_1xy. \tag{2.31}$$

Thin prism distortion ($\mathcal{D}_{x,pris}, \mathcal{D}_{y,pris}$) is caused by the imperfect lens design/manufacturing or the camera assembly (e.g., a minor tilt of lens components or the image sensor) [31] and is defined as:

$$\mathcal{D}_{x,pris} = S_1(x^2 + y^2), \tag{2.32}$$

$$\mathcal{D}_{y,pris} = S_2(x^2 + y^2), \tag{2.33}$$

where S_1 and S_2 are the coefficients of the thin prism distortion.

Combining the three distortion types leads to a total lens distortion of a camera that can be expressed as [31]:

$$\mathcal{D}_x = S_1(x^2 + y^2) + 3P_1x^2 + P_1y^2 + 2P_2xy + K_1x(x^2 + y^2), \tag{2.34}$$

$$\mathcal{D}_y = S_2(x^2 + y^2) + 2P_1xy + P_2x^2 + 3P_2y^2 + K_1y(x^2 + y^2). \tag{2.35}$$

2.1.6 Fisheye Camera Models

Perspective cameras are typically considered ideal. Straight lines in the real world are largely preserved in the image. Subsection 2.1.5 describes how to deal with undesirable but inevitable effects that are caused in real world camera systems. The radial distortion (barrel

distortion) is mainly dominant when applying fisheye lenses compared to the pinhole camera model. Fisheye optics are particularly favorable for applications where large portions of the scene need to be imaged in a single snapshot. Assuming an ideal pinhole model, where straight lines in the scene are mapped into straight lines on the image plane, a large field-of-view would result in a large projected image as is shown in the geometric illustration of Table 2.1 (a). This phenomenon can be mathematically confirmed when regarding the (undistorted) projected radial distance from the principal point on the image plane $r_u = \sqrt{x_u^2 + y_u^2}$ for a perspective projection:

$$r_u = f \tan(\eta), \tag{2.36}$$

where η represents the incident angle of the projected ray to the optical axis of the camera. A perspective camera lens that covers a hemispherical (180°×180°) field-of-view produces an infinite large image and is hence not applicable. The prior art designed and investigated fisheye projection functions such that large areas of the 3D scene can be mapped onto the image plane at the cost of a radial distortion [34]–[36]. These projection functions along with the mathematical descriptions that are needed to convert between the perspective image space and the corresponding fisheye image space are summarized in Table 2.1. The fisheye cameras used for the present work follow the *equidistant* fisheye projection convention. The radial distance $r_d = \sqrt{x_d^2 + y_d^2}$ to the distorted image point $\boldsymbol{p}'_{sph}$ by means of the equidistant projection is directly proportional to the angle of the incident ray η:

$$r_d = f\eta. \tag{2.37}$$

The length of the radial distance r_d is thereby identical to the length of the arc segment between the principal point $\boldsymbol{o}_c$ and the projected point $\boldsymbol{p}_{sph}$ on the projection sphere (see Table 2.1). Inserting Equation (2.36) into the equidistant projection function allows us to determine the equidistant fisheye representative of a perspective image point with:

$$r_d = f \arctan\left(\frac{r_u}{f}\right). \tag{2.38}$$

The inverse thereof is defined to convert fisheye image points back to perspective ones according to:

$$r_u = f \tan\left(\frac{r_d}{f}\right). \tag{2.39}$$

Fisheye lenses are usually designed and manufactured to follow one of the projection models compiled in Table 2.1. Recalling the previously mentioned imperfect sequence of manufacturing processes and the permitted level of tolerance limits, minor deviations from the ideal projection functions are unavoidable. Various distortion models have been investigated and proposed to account for deviations in fisheye lenses, some of which can be found in [36]–[40]. Table 2.1 gives an overview of the most frequently applied ones.

2.2 View Synthesis

The previously introduced concepts describe the low-level basics and instruments that are used to accomplish the overall goal of creating an omnidirectional (stereoscopic) snapshot

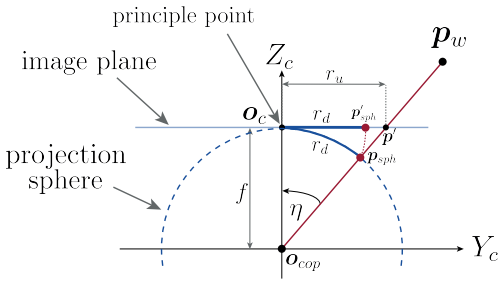
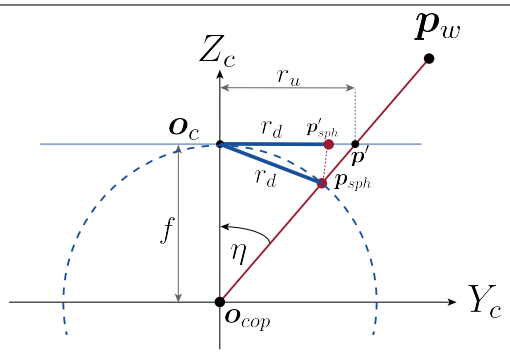
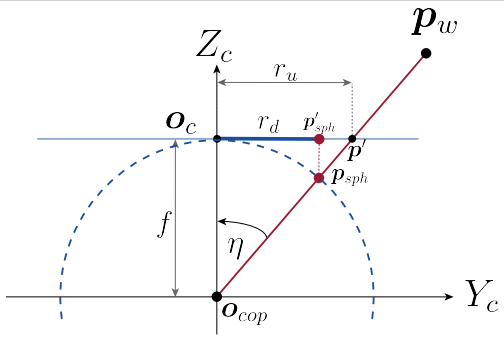
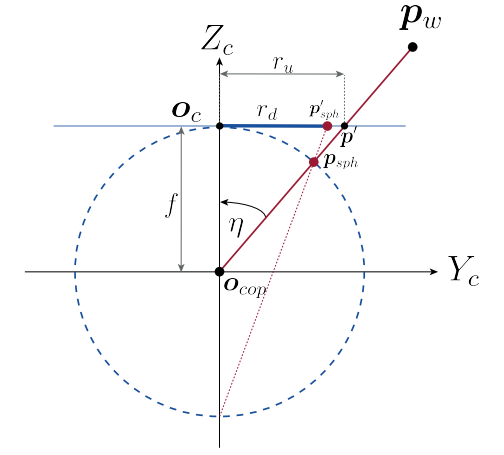
Projection type	Geometric illustration	Mathematical description
Equidistant projection (a)		$r_d = f\eta = f \arctan\left(\frac{r_u}{f}\right)$ $r_u = f \tan\left(\frac{r_d}{f}\right)$
Equisolid projection (b)		$r_d = 2f \sin\left(\frac{\eta}{2}\right) = 2f \sin\left(\frac{\arctan\left(\frac{r_u}{f}\right)}{2}\right)$ $r_u = f \tan\left(2 \arcsin\left(\frac{r_d}{2f}\right)\right)$
Orthographic projection (c)		$r_d = f \sin(\eta) = f \sin\left(\arctan\left(\frac{r_u}{f}\right)\right) = \frac{r_u}{\sqrt{\left(\frac{r_u}{f}\right)^2+1}}$ $r_u = \left(\frac{r_d}{\sqrt{1-\left(\frac{r_d}{f}\right)^2}}\right)$
Stereographic projection (d)		$r_d = 2f \tan\left(\frac{\eta}{2}\right) = 2f \tan\left(\frac{\arctan\left(\frac{r_u}{f}\right)}{2}\right) = \frac{r_u}{1-\left(\frac{r_u}{2f}\right)^2}$ $r_u = f \tan\left(2 \arctan\left(\frac{r_d}{2f}\right)\right)$

Table 2.1: Mathematically-founded and geometrically illustrated summary of state-of-the-art fisheye projection models. These involve (a) equidistant, (b) equisolid, (c) orthographic, and (d) stereographic projection models [36]. This work uses the equidistant projection model as a mathematical foundation when deploying fisheye cameras.

of a 3D (remote) environment. There exist different ways of producing photorealistic visual representations of a 3D scene both for static and dynamic scenes. (Depth) Image-based rendering ((D)IBR) is a simplified approach that leverages images as the primary source of information to address this problem. Chan *et al.* [41] introduced an IBR continuum that outlines a wide range of techniques for novel view synthesis that have been proposed in literature. Figure 2.6 sorts these methods with respect to the amount of geometric information and image data that is needed in the synthesis process. Despite its continuous character, the representations can coarsely be classified into three categories [41]: 1) rendering without geometry, 2) rendering with implicit geometry, and 3) rendering with explicit geometry, which will be briefly discussed in the following.

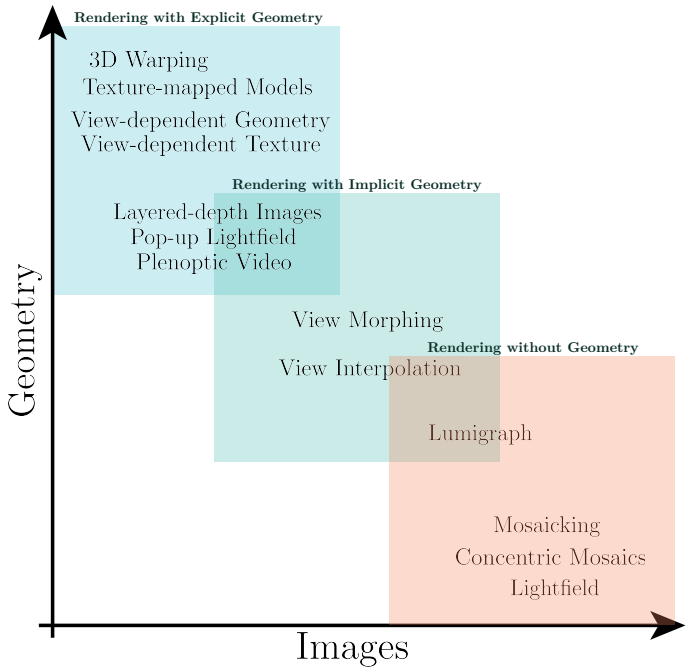


Figure 2.6: Schematic overview of the image-based rendering continuum according to [41]. The more images are available, the less geometry of the environment is needed for novel view synthesis, and vice-versa.

2.2.1 Rendering Without Geometry

In the absence of scene geometry, images are the dominant substrate for IBR. The less geometry is available, the more images are needed for view synthesis as was elucidated in Figure 2.6. Representative techniques that render without geometry are premised on the *plenoptic function* [42].

Plenoptic Function The 7D plenoptic function records the intensity of the light rays passing through a camera center located at (X, Y, Z) for every possible angle (θ, ϕ) and wavelength λ at every time instance t [42] as visualized in Figure 2.7 (a):

$$\mathcal{P}_7 = \mathcal{P}(X, Y, Z, \theta, \phi, \lambda, t). \tag{2.40}$$

Plenoptic Modeling McMillan and Bishop firstly introduced the *plenoptic modeling* (see Figure 2.7 (b)), which assumes a static environment with non-changing light conditions waiving the two variables λ and t [43]:

$$\mathcal{P}_5 = \mathcal{P}(X, Y, Z, \theta, \phi). \tag{2.41}$$

2D Panorama Additionally fixating the viewpoint of the camera to a specific location, as visualized in Figure 2.7 (c), allows to define the simplest variation of the plenoptic function referred to as 2D *panorama*:

$$\mathcal{P}_2 = \mathcal{P}(\theta, \phi), \tag{2.42}$$

which is parameterized in angular coordinates [44]. A 2D panorama covers the complete $360° \times 180°$ space and can be cylindrical [45] or spherical [46]. A regular image, which records only a subset of the entire scene at a fixed viewpoint, can be treated as an incomplete plenoptic sample of $\mathcal{P}_2$ [47]. A 2D panorama can hence be constructed by capturing a certain number of plenoptic samples (depending on the field-of-view). This procedure is known as image mosaicking and leverages a sequence of image samples to construct panoramas. Each input image needs to be projected onto the common (cylindrical or spherical) surface and adequately aligned. Global bundle adjustment algorithms are usually applied as countermeasures for the accumulated registrations errors [48]. If the images do not share a common center of projection, visual artifacts will occur and result in intensity discrepancies and image distortions. Deghosting techniques are required for optimal local alignments [49].

Light Fields The measurement of the 5D plenoptic modeling in real applications is very challenging; especially for concave regions where the light leaving one point might be blocked by another one. Restricting the region-of-interest to locations that lie outside the convex hull of an object results in light rays that remain constant from one point location to the other [50] as depicted in Figure 2.7 (d). The 5D plenoptic function contains then a one dimensional redundancy leaving a 4D function that was introduced as the 4D *light field* and is defined as the radiance along with rays in empty space [51]:

$$\mathcal{P}_4 = \mathcal{P}(u, v, s, t). \tag{2.43}$$

The 4D light field is parameterized by the intersection of rays with two parallel planes (u, v) and (s, t) of the bounding box of an object that is assumed to be the simplified convex hull thereof [47].

Concentric Mosaics Limiting the range of motion to regions outside the convex hull of objects helpes to reduce the plenoptic function by one dimension. Inspired by this course of action, a 3D parameterization approach was proposed, coined as *concentric mosaics*, that restricts the camera motion to planar concentric circles [52]. Concentric mosaics are created by composing slit images (usually taken with line/slit-scan cameras) that are sampled at different locations along each circle. Three parameters are used to index all input image rays:

$$\mathcal{P}_3 = \mathcal{P}(r, \eta, \varphi), \tag{2.44}$$

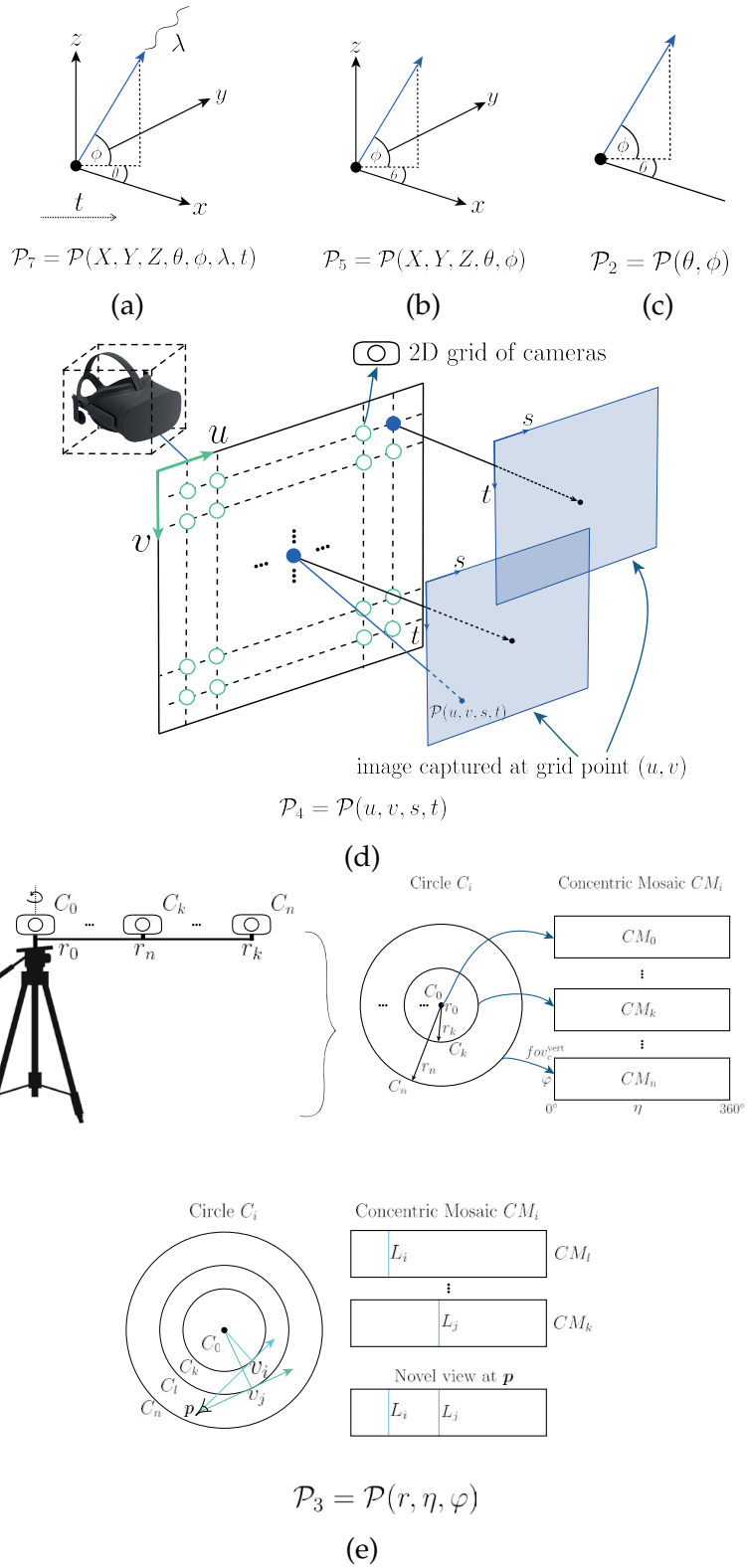


Figure 2.7: Figurative overview of the 7D plenoptic function and the dimension-reduced variations thereof: (a) 7D plenoptic function, (b) 5D plenoptic modeling, (c) 2D panorama, (d) 4D light fields, and (e) concentric mosaics.

where r is the radius, η the rotation angle, and φ the vertical elevation. Novel views can be rendered within the circular region by combining and interpolating previously captured rays. Depth corrections are required to account for the vertical distortions present in the rendered image [52]. Figure 2.7 (e) outlines the working principle of concentric mosaics and illustrates its view synthesis process.

2.2.2 Rendering with Implicit Geometry

Rendering with implicit geometric information refers to techniques where the geometry is not directly available [44]. 3D information is deducted from positional correspondences (features) across visually overlapping images and computed by means of projective geometry. Approaches that can be assigned to this category comprise (joint) view interpolation [53], [54], view morphing [55], and transfer methods, which are based on geometric constraints between image pairs or trifocal tensors [47], [56], [57].

2.2.3 Rendering with Explicit Geometry

Representations with explicit geometry have direct access to the 3D information of the scene, either in the form of depth values or 3D coordinates [44]. 3D warping belongs to this category, where novel views are rendered by means of pixel information of reference images in the near vicinity. These pixel values are projected to their correct 3D locations and reprojected onto the image from the novel viewpoint [58]. Occlusions, incomplete pixel information, and depth discontinuities can induce the emergence of holes in the output image [59]. Hole-filling is often approached by in-painting methods and is a recent topic of research [60]–[63].

Further rendering techniques that leverage explicit geometry knowledge are layered depth images, which store lists of depth and color values for each ray intersecting with the environment [64], and view-dependent texture maps, which are widely used in computer graphics [47], [65].

2.3 Acquisition of Monoscopic Panoramas

2D panoramas represent the simplest version of the plenoptic function. The acquisition of monoscopic panorama images/videos is considered state-of-the-art and can be classified into 1) single-viewpoint (SVP) and 2) multi-viewpoint (MVP) panoramic footage.

Single-viewpoint Panoramas Omnidirectional images that share a common center of projection analogous to the central projection are denoted as single-viewpoint panoramas. SVPs can be created by either using catadioptric systems or rotating cameras. Catadioptric systems combine refraction and reflection in optical systems using lenses (dioptrics) and specially designed mirrors (catoptrics). Hyperbolic or parabolic mirrors are developed to capture the 360° view of the scene with a single snapshot [66], [67] as is shown in Figure 2.8 (a). Incomplete plenoptic samples can also be utilized to produce a panorama image as is described in

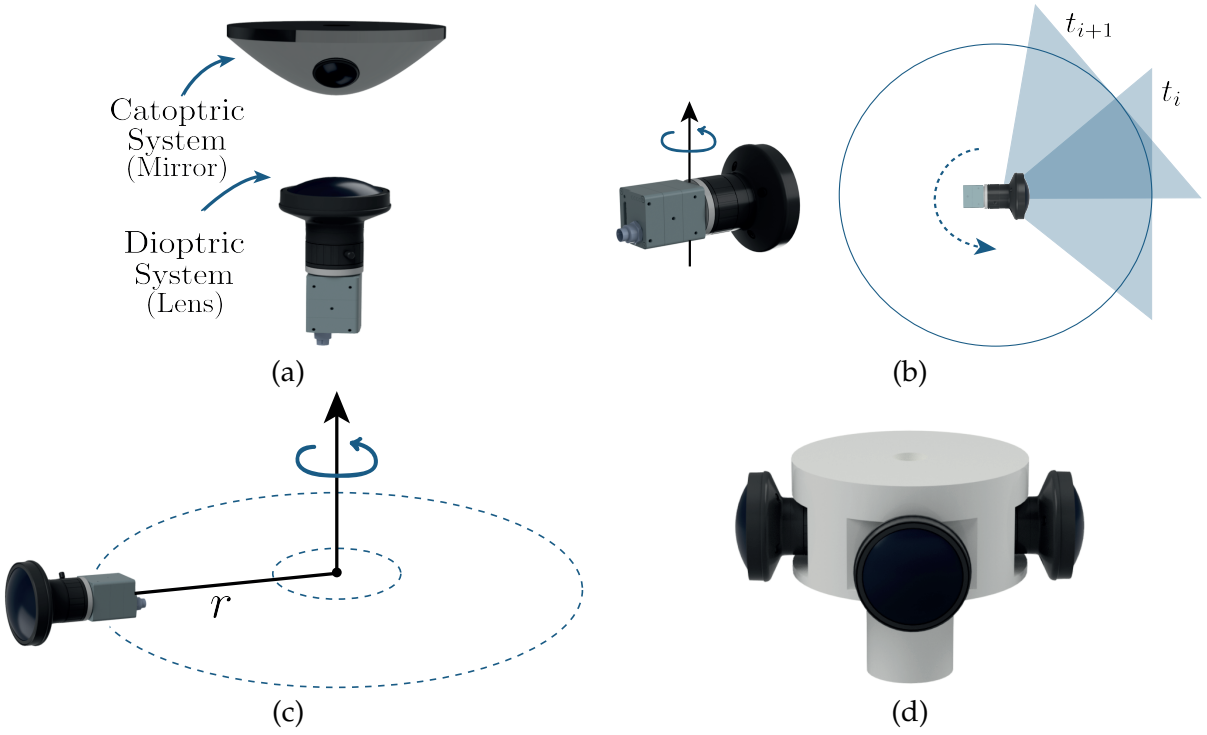


Figure 2.8: Compilation of state-of-the-art acquisition methods to create single- or multi-viewpoint panoramas. (a) Catadioptric systems combine hyperbolic or parabolic mirrors with wide-angle cameras to capture the scene in a single snapshot. (b) Single-viewpoint panoramas can also be created by rotating a single camera around its nodal point and stitching partially overlapping views. (c) Multi-viewpoint panoromas can be produced by rotating an off-centered camera system and stitching either view-overlapping images or view slits with a high sampling rate. (d) MVPs can also be approximated by deploying a multi-camera system.

Subsection 2.2.1. The samples are created by rotating a camera around its nodal point and acquiring a number of either view-overlapping images at multiple perspectives or vertical slit images by using slit-scan photography techniques [68], [69]. The respective mosaicking process is visualized in Figure 2.8 (b). The rotation around the nodal point is essential to ensure the center of projections to be spatially collocated. SVP cannot be modeled with multiple cameras due to the limitation of their physical dimensions.

Multiple-viewpoint Panoramas MVPs can be constructed by both rotating an off-centered camera [70] or adopting a multiple-camera arrangement as illustrated in Figure 2.8 (c) and (d). The displacement of the center-of-projections is deliberately chosen to be (often) centrally symmetric [71]. Multi-perspective slit images can be used to approximate a concentric mosaics-based approach by limiting the range of motion to one single viewing circle. Another well-studied strategy is to acquire off-centered image footage and leverage computer vision techniques for optimal stitching [72].

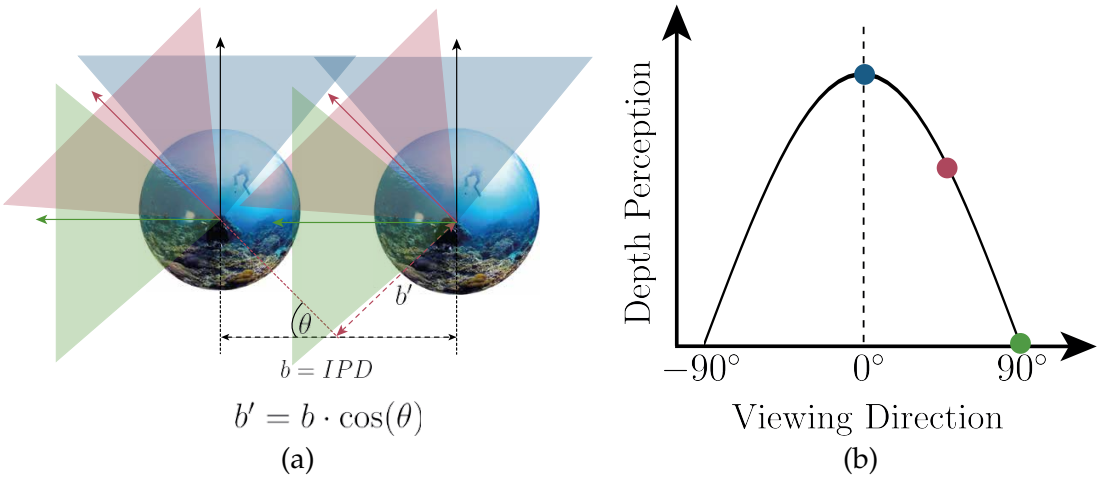


Figure 2.9: The challenge of capturing omnistereoscopic visual representations. (a) Simply using two parallel omnidirectional panorama cameras is not sufficient to provide stereoscopic perception equally in all viewing direction. The baseline b, which roughly coincides with the humans inter-pupillary eye distance (IPD), constitutes the depth-perceiving sensibility. (b) Any viewing direction dissimilar to the forward view decreases the effective baseline and hence the intensity of depth perception.

2.4 The Challenge of Omnistereoscopic Vision

While capturing and streaming monoscopic 360° video content in realtime is considered state-of-the-art, this is not yet the case for omnistereoscopic photography. The acquisition and processing of omnidirectional stereoscopic panoramas is a cutting-edge research field that is highly favored in virtual reality (VR) systems. For the display of stereoscopic panoramas on fully immersive VR systems, a binocular view of the (distant) environment needs to be rendered for the current viewing direction. The sensation and perception of depth is established whenever two synchronized views with horizontal parallax are provided with respect to the inter-ocular distance of the human user's eye. Simply deploying two omnidirectional cameras in a parallel, offset configuration, where each camera is to be assigned to one eye, is not sufficient as is demonstrated in Figure 2.9. The depth perception in such a configuration cannot equally be ensured in all viewing directions. The proper 3D sensation is only guaranteed for gaze directions that are (nearly) perpendicular to the camera's mounting ($X-$) axis. The effective baseline decreases thereby with $\cos(\theta)$ until zero depth perception is reached for viewing directions that are co-aligned with the $X-$axis. Besides the heterogeneous depth allocation, there exists also a conflict of viewing directions as each camera is visible in the other view. More sophisticated solutions are needed to address this issue.

A comprehensive classification and review of possible acquisition methods and models for the rendering of stereoscopic panoramas both for static and dynamic scenes are discussed in [69] and [73]. The prior art shows that the flawless realtime acquisition of stereoscopic panorama videos remains unsolved. Conceivable configurations are based on sequential acquisition [74], [75], the design of catadioptric systems [76], [77], or the deployment of a multi-camera arrangement [78]–[83].

Sequential Acquisition Sequential acquisition of stereoscopic panoramas is usually performed by rotating camera systems and conceptually leads to the best results. Most of these systems are based on the concept of concentric mosaics [52]. Two outwards facing cameras are installed back-to-back at a certain distance on a rig to produce panoramic views for each eye. The camera system is spun to capture images from different vantage points for any viewing direction. The sequential gathering results in high-quality omnistereoscopic images. Due to its sequential nature, however, most of the proposed solutions do not apply to dynamic environments. Along with its massive overhead of data needed to render binocular views, it is unsuitable for embedded systems, especially for telepresence applications. Konrad *et al.* [84] managed to deploy proper hardware to enable live streaming with 5 fps. Utilizing more line cameras and fine-tuning the settings can boost the realtime performance up to 16.67 fps and 32 fps, respectively [84]. The large amount of data is processed by a local computer to render the line images and remove the distortions seamlessly.

Catadioptric Systems Catadioptric systems extend the visual field of cameras by using tailored mirrors. First studies deployed catadioptric systems in a vertical configuration. The coaxial stereo configuration was used to compute the depth information and produce a 3D reconstructed model of the environment in realtime. Recent work deployed catadioptric systems for human stereo-vision [76]–[78]. The advantage of these systems is that they are usually not limited to static scenes and can also be utilized in dynamic environments. Schreer *et al.* [78] presented a system arrangement of twelve high-resolution cameras combined with a large mirror structure. Six planar mirror sectors are arranged as a hexagonal frustum. Each of these segments reflect a partial view of the scene on a stereo camera pair. The hexagonal, star-like configuration is, however, detrimental to the minimum depth-sensation distance. The inter-axial distance of adjoining stereo-camera systems is larger than the baseline of one two-camera pair making the stitching process of partially overlapping views challenging. The work presented in this manuscript enforces omnistereoscopic vision to provide immersive telepresence particularly for indoor scenarios, where events occur in the near vicinity of the remote operator. Depth sensation should also be provided for close objects, which makes the presented approach not applicable for the targeted application. This limitation is particularly critical when performing pinpoint manipulative tasks. In addition, catadioptric systems necessitate precise calibration to achieve proper stitching results. It is hence not clear if such systems are robust and reliable enough when attached to a moving platform on bumpy grounds.

A light-weight catadioptric sensor for omnistereoscopic footage production was proposed by Aggarwal *et al.* [77], who used a single camera in combination with a specifically drafted "coffee filter mirror". The system was designed to create stereoscopic scene snapshot for an inter-pupillary distance (IDP) of 6.5 cm. The stereoscopic budget is greatly limited, as the IDP is highly subject to the system design and can not be changed dynamically. The presented system benefits from its computational and financial lightness compared to related work. One critical drawback of this system is, however, the unavoidable stitching artifacts, which are clearly visible. Flawed image content presented on HMDs is a QoE-limiting factor,

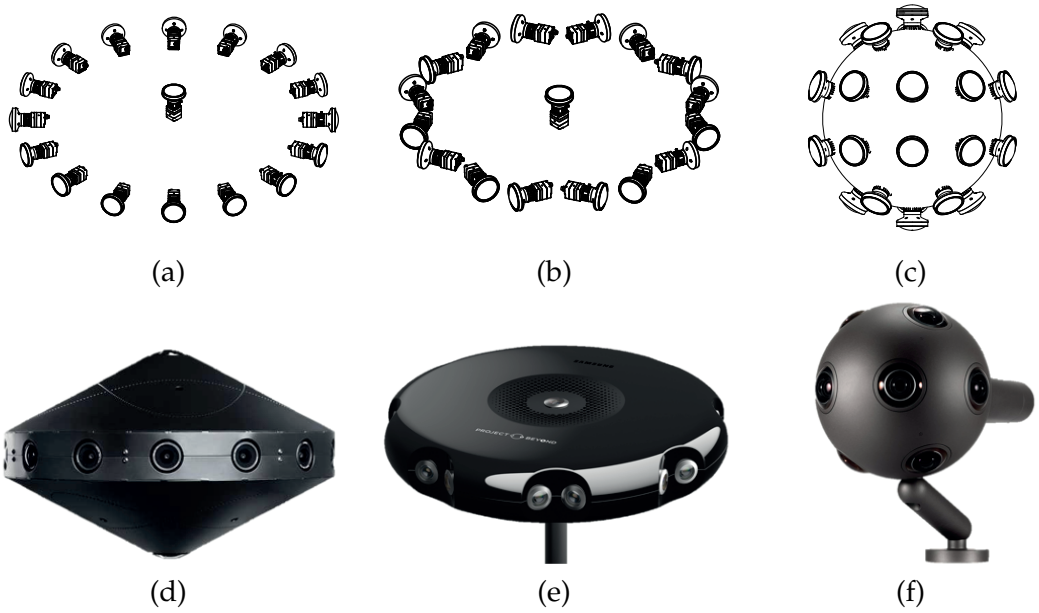


Figure 2.10: (a)-(c) illustrate multi-camera configurations for the acquisition of omnistereoscopic visual data. (d)-(f) represent their implementation as consumer products (reproduced from [5])

as the videos are magnified and displayed to the user from a minimal distance.

Multi-camera Systems Multi-camera systems are often positioned in a radial or spherical arrangement to acquire omnistereoscopic visual content both for static and dynamic scenes. Gurrieri *et al.* [69] give a profound outline of multi-camera configurations and rendering techniques that are presented in literature. The most promising and representative camera assemblies are illustrated in Figure 2.10 (a) - (c). Consumer products were implemented and released on the basis of these techniques such as Samsung's Project Beyond [82] (Figure 2.10 (d)), Facebook's Surround 360 [80] (Figure 2.10 (e)), or Nokia's Ozo [81] (Figure 2.10 (f)). There exist much more products such as the 3D system Jaunt [85] or Google's Jump [79], which put the presented multi-camera configurations into practice.

The rendering policy of stereoscopic panoramas is highly dependent on the cameras' arrangements and often inspired by an approximation of concentric mosaics, view interpolation between adjacent cameras for each eye, or depth image-based rendering techniques. Partial views are captured for each eye and mosaicked to one omnidirectional visual snapshot of the scene by means of state-of-the-art stitching methodologies. The physical dimensions limit the number of employable cameras and thus influence the quality of the final output. The more cameras are used, the better the stitching quality becomes, but at the same time the higher the amount of data that needs to be processed. In addition to the camera's housing dimension that might limit the number of deployed cameras, the self-occlusion between adjacent cameras needs to be taken into account [69]. Mounting a wide range of camera modules with small field-of-views would affect the minimum distance of depth perception.

The stereoscopic budget is a measure for the level of flexibility of changing the interpupillary distance and is severely limited in such multi-camera-based architectures. The

multi camera's assembly design presets the inter-ocular distance implicitly to one specific value. Facebook, for instance, realeased its Facebook Surround 360 system in 2016, which deployed 14 wide-angle camera modules mounted in a radial horizontal configuration with one fishseye lens facing upwards and two facing downwards for a complete spherical coverage [80]. The computational complexity given by the tremendous data overhead is tough. With a transfer rate of 17 Gb/s of acquired image footage, this systems is far-away from being realtime-capable. The amount of data can only be handled in a post-processing step. Processing as such involves intrinsic image correction, bundle-adjusted mutual extrinsic camera corrections to compensate misalignments in the camera orientation, and optical flow calculations between adjacent cameras to compute the left-right eye disparity [80]. A subsequent process finalizes the view synthesis both for the left and right eye. The enormous computational complexity implicitly infers the required hardware peripherals that are needed for the stitching process. A similar solution is presented by Google's Jump project [79]. A comparable multi-camera arrangement, similar to Figure 2.10 (a) is adopted to create omnistereoscopic footage. The time needed to mosaick a single frame is in the domain of minutes. Even with a parallel computation over 1000 cores, one hour of footage was processed in around ten hours. This elucidates the challenge of facilitating stereoscopic panoramas in realtime.

Another shortcoming of the multiple-camera assemblies illustrated in Figure 2.10 (a) and (b) arises from the limited depth sensation abilities. Proper depth perception is only ensured for a horizontal radial belt around the cameras. Above and beneath this radial belt, that is when looking up- or downwards, depth perception is not properly facilitated, as these regions are filled with monoscopically acquired image content.

Both multi-camera and catadioptric architectures necessitate a computationally complex stitching procedure subsequent to the acquisition. The stitching grade is highly subject to the features present within the scene. Insufficient features result in flawed image content, which is deemed to be a QoE-limiting factor when displayed on head-mounted displays considering their magnification characteristics and minimal eye-to-display distance. The architecture of both approaches presets the available IDP and thus limits the stereoscopic budget. Costly peripheral hardware equipment is often required to withstand sizable computational burdens, making such solutions inappropriate for applications that need to perform in realtime.

One objective of this work is to address the previously discussed challenges. In Chapter 4, a delay-compensation vision system is presented that does not require a mosaicking procedure and is thus free from stitching artifacts. The proposed two-camera system is augmented with an electro-mechanical unit to provide depth sensation in any desired viewing direction. Along with its large stereoscopic budget, the proposed approach is lean, realtime-capable, and at a budget price.

2.5 Visual Communication

The acquisition of panorama videos is only the initial step needed to establish an immersive VR experience. While local applications would start to render the VR content on the screen

(a)

(b)

Figure 2.11: (a) Equirectangular image projection of an omnidirectional scene snapshot and its equivalent (b) cupe map projection.

or HMD, this is not yet the case for remote reality or telepresence applications. They provide the visual information of the remote scene and allow human users to immerse themselves into distant or inaccessible environments. A typical server-/client-based processing pipeline of such a remote reality system can be divided roughly into the 1) acquisition, 2) preprocessing, 3) compression, and 4) display of 360° images.

2.5.1 Acquisition

The recording of monoscopic and stereoscopic panorama videos were thoroughly discussed in Section 2.4. Due to the challenge of capturing omnistereoscopic footage, especially in realtime, the focus of related work is mainly put on monoscopic visual content acquisition.

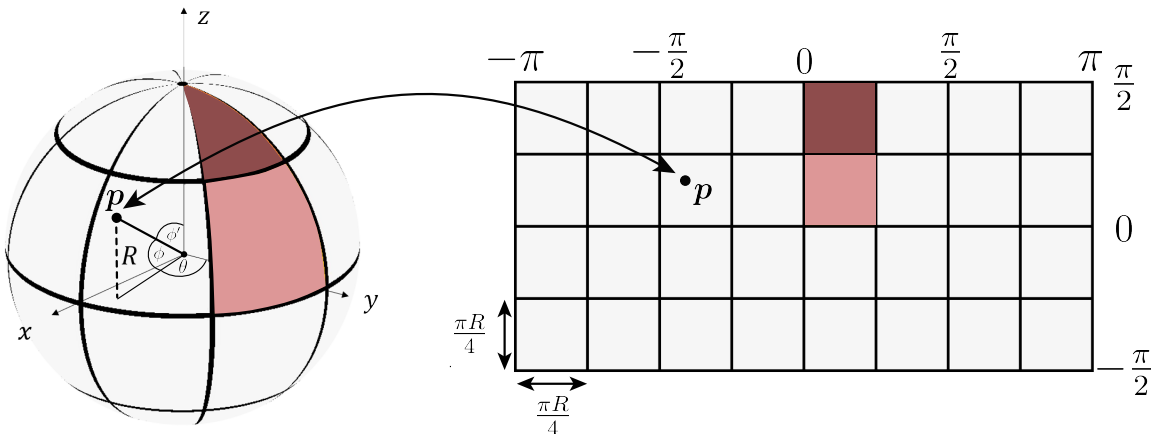


Figure 2.12: The equirectangular image format is obtained by unfolding the spherical image. The sphere is sampled into horizontal circles, which are then projected as horizontal lines on the 2D image plane. The equirectangular projection scheme is characterized by oversampling towards the pole regions. Dividing the ERP in a tile grid results in a mismatch between the area size of tiles in the ERP plane and its corresponding spherical tile regions.

2.5.2 Preprocessing

Preprocessing is a far-reaching term that may address many computational steps after the image acquisition and before the encoding. In the context of 360° video streaming, the preprocessing step can be seen as a preparatory phase for the encoding procedure. 2D Panoramas are usually projected onto a sphere (or cylinder) as is described in Section 2.2.1 and are hence denoted as spherical videos. Current image/video compression standards such as *H.264/MPEG-4 Advanced Video Coding (AVC)* [86] and its successor *H.265 High-Efficiency video Coding (HEVC)* [87], however, do not offer a direct feature to encode in the spherical domain. To still exploit the highly reliable and widely compatible image coding standards, the images are projected on a planar surface by means of a 2D projection such as the *equirectangular projection (ERP)* [88], the *cube map projection (CMP)* [89], or the *Rhombic dodecahedron* [90]. A shortcoming of these projections are the introduced artifacts caused by oversampling, undersampling, distortions and/or discontinuous boundaries. The ERP and CMP, both pictured in Figure 2.11 (a) and (b), are the most prominent ones and will be briefly described in the following.

Equirectangular Projection The equirectangular mapping is the simplest and hence most widely encountered sphere-to-plane projection technique. The spherical image with radius r is unwrapped and stretched to a 2D rectangular plane of size $(2\pi r, \pi r)$. This is accomplished by sampling the sphere into circles of latitudes (horizontal circles) and projecting them to horizontal lines on the 2D image plane. The main projection artifact of ERPs is due to the oversampling around the pole regions, where lines need to be substantially more stretched to fit onto the rectangular image plane compared to lines in the proximity of the equator area. The high amount of redundant pixels towards the pole regions increases the surface area by 57 % resulting in an enlarged bitrate requirement [91]. Although there exist more

sophisticated and efficient projection techniques, ERP is the preferred and widely supported projection method given its low complexity. Figure 2.12 demonstrates the equirectangular mapping from a sphere to a 2D image plane. The horizontal and vertical coordinates of the image plane correspond to the spherical longitude (θ) and latitude (ϕ' or ϕ) values. A Cartesian 3D point $\boldsymbol{p} = (x, y, z)$ on the sphere with radius r must only be transformed into polar coordinates with respect to the geometric arrangement sketched in Figure 2.12:

$$\theta = \arctan\left(\frac{y}{x}\right), \tag{2.45}$$

$$\phi' = \arccos\left(\frac{z}{r}\right) \quad \text{or} \quad \phi = \arcsin\left(\frac{z}{r}\right), \tag{2.46}$$

$$\text{with} \quad r = \sqrt{x^2 + y^2 + z^2}. \tag{2.47}$$

Depending on the followed convention, either ϕ' or ϕ can be used to convey the latitude value, respectively. ϕ' and ϕ have a direct relation to each other ($\phi = \frac{\pi}{2} - \phi'$) according to:

$$\sin(\phi) = \cos(\frac{\pi}{2} - \phi) = \cos(\phi'). \tag{2.48}$$

Cube Map Projection The cube map projection does not require a warping step. The spherical image is first projected onto the six faces of a cube and thus consists of six independent perspective projections as depicted in Figure 2.11 (b). The faces are then unfolded and arranged into a 2D image plane. Compared to the ERP, CMP does not introduce redundant pixels and is characterized by less distortion.

2.5.3 Compression

Adopting the sphere-to-plane projection technique allows the user to leverage common coding standards that are compatible across multiple systems. The streaming of high-resolution panorama videos is bandwidth-consuming and hence requires efficient video codec standards. The H.264/AVC [86] and its successor H.265/HEVC [87] are the most prominent and supported ones. Both standards exhibit significantly lower bitrates compared to previous standards. H.265/HEVC is particularly beneficial for high-resolution videos. A detailed comparison can be found in [92].

2.5.4 Display

After encoding the 360° videos, they are transmitted over a communication network. The received video data is decoded and prepared for display. HMDs are often deployed to provide a fully immersive experience. They usually have one or two small displays, which are augmented with lenses. These lenses aim to project the screens, which are located very close to the eyes, to a distance of a few meters away, where the video content can be viewed comfortably. A shortcoming thereof is, however, that they introduce a Pincussion distortion (see Figure 2.5), chromatic aberrations, and other artifacts, which need to be accounted for in software. The images are thereby digitally corrected by pre-distorting them with a Barrel distortion. The most prominent representatives are the Oculus Rift [93], HTC Vive [94], and Samsung's Gear VR [95].

2.5.5 Viewport-dependent Streaming

Encoding and streaming the entire 360° video at a single quality level without any spatial variance is referred to as *monolithic* streaming. In case of omnistereoscopic vision, two complete high-resolution panorama videos, preferably at high frame rate, need to be sent over the communication network. Following this policy 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 when watched through an HMD. The state-of-the-art focused on viewport-dependent streaming strategies to reduce the required transmission rate. Viewport regions are streamed at higher qualities than peripheral areas. The overall objective is to find the optimal trade-off between QoE and available resources [96].

One way to do this is to apply a pyramid-based viewport dependent projection scheme as was proposed by Facebook and presented in [97]. Their approach pre-selects 30 viewports of the 360° video content and modifies them as per the pyramid projection. The base of the pyramid is allocated for the corresponding viewport and maintains its full resolution. The quality is then gradually decreased with respect to the distance to the base. The server stores each viewport with five different resolutions. Their approach manages to reduce the file size and thus the data rate consumption by 80 %. The downside of this methodology is the relatively high storage requirement at the server as 150 different streams of the same content need to be stored.

Another promising approach was proposed by Corbillon *et al.* [98], who create multiple representations of the same video content at the server with predetermined *quality emphasized regions (QER)* and with different global quality levels. Each QER has a *quality emphasis center (QEC)*, which is used to select the representation to be streamed to the user. The QEC that is closest to the viewpoint is chosen as the current representation and sent to the user with respect to the available transmission rate akin to *DASH (Dynamic Adaptive Streaming over HTTP)* [99]. The client selects thereby the appropriate representation according to the current viewport, the requested image resolution, and the available transmission capacity. The necessitated storage resources at the server side are noticeably lower compared to [97].

An alternative way of approaching rate-limited 360° video streaming applications draws upon a spatial tiling of the footage [100]–[103]. Each tile is treated and encoded/decoded independently. Tile-based streaming has lower storage requirements when compared to viewport-adaptive projection and streaming methodologies. Bitrate-adaptation affects each tile independently and hence allows for more flexibility. Only the desired tiles are streamed to the client. Prior works deployed rate-distortion optimization techniques with spatial tiling and performed superior compared to monolithic streaming [103].

The state-of-the-art deploys in general two common methods for the tile selection process by either sending only the tiles, which are within the current viewport of the user, or by transmitting the tiles within the viewport with high quality and the peripheral ones with low quality. The latter one streams the entire 360° visual content and hence accounts for the strict latency requirements of HMD-based VR systems by providing immediate visual feedback for head-motions. A downside of tile-based streaming is its reduced coding efficiency. Smaller tiles ensure high streaming bitrate savings but at the cost of higher compression

losses [101].

Sending two complete panoramas, preferably at high-resolution and high frame rate requires large transmission capacities, although a considerable portion of the footage will remain unused. Only the content of the user's viewing direction needs to be rendered onto the HMD, which corresponds to a subset of the full panorama.

Related work investigated viewport-adaptive streaming techniques to find the optimal trade-off between quality-of-experience and available network resources. Viewport-adaptive streaming technologies can be categorized as top-down approaches that take the availability of omnidirectional image contents for granted. A portion of the scene snapshot along with a complementary buffer margin is extracted according to the desired gaze direction and sent to the client-side. The amount of required buffer size is unknown and lacks a detailed algebraic description. A shortcoming of related work is that most acquisition and streaming solutions are mainly limited to monoscopic image content considering the previously introduced challenges of creating omnistereoscopic footage. State-of-the-art viewport-adaptive approaches mostly follow a one-to-many on-demand streaming methodology.

The study presented in this manuscript approaches both the recording and streaming of omnistereoscopic videos and facilitates a live, first-person-view telepresence experience with less overhead. The root issues are mathematically modeled and profoundly described. The compensation rate is introduced as a device-agnostic metric to reflect the realizable degree of delay-compensation. Its generic character makes it integrable for viewport-dependent streaming techniques. Optimization problems can be formulated to identify the minimum amount of buffer region required to provide immediate access to visual content throughout the whole telepresence session. System parameters such as the respective delay, the available image content, or the size of the user's viewport need to be considered and incorporated into the optimization framework.

2.6 Viewport Prediction

The latency between the server and the client, which is caused by the communication delay and other contributors, induces a lag between the head-motion and the corresponding visual response (M2P-latency). Viewport or head-motion prediction is a valid method to further improve the performance of streaming applications. Rather than sending the current head orientation value, the prospective head motion is predicted with respect to the present end-to-end delay. Estimating and sending the future head position helps to decrease the M2P-latency.

The research in head motion prediction has already started in the early '90s. The objective then was to compensate the local lag between head motion and display response, which is caused by the time needed to track the user's head and render the visual content onto the HMD [14], [16], [104]–[107]. Even nowadays, it takes at least around 30 ms to 50 ms to render imagery to an HMD as was investigated in [108]. The delay compensation of these methods hence aimed to compensate latencies in the range of 10 ms to 100 ms. There were two

widely used best practices to accomplish head motion forecasting. One way was to exclusively consider the past head trajectory of a user and fit a first-order polynomial employing Linear Regression (LR) [109]. Other groups approached this issue by using filters such as the (Extended) Kalman Filter [110] or Particle Filter [111] to first obtain an optimal state estimate of the current head position in terms of head orientation, angular velocity, and acceleration. In a subsequent step, the estimated state representation is employed in a motion model for linear or polynomial extrapolation. The advantage of the first-order polynomial fitting technique is given by its low-complexity and precision for homogeneous and smooth motions. It suffers, however, from erroneous prediction and tends to overshoot for quick motions and abrupt orientation changes. The benefit of applying filters that optimally estimate the current head orientation, velocity, and acceleration and inserting them into a motion model is their robustness and reliability during quick fluctuations. Shortcomings of such techniques result from the computational complexity and sensitivity to noise. The precision and reliability strongly depends thereby on the accuracy of the deployed motion model.

Another recent field of research relevant in this context is 360° video streaming as was introduced in Section 2.5. Sending the entire panorama videos proved to claim large parts of the transmission capacity. A considerable portion of the imagery remains unused as only the user's viewport is rendered onto the HMD. Prior art decided to send only a segment of the omnidirectional footage and leverage prediction techniques to forecast successive segments for smooth and non-disturbing view transitions. Prospective head motions were predicted akin to the previously described methods such as (weighted) linear regression or averaging of past orientation values within a window of interest [112], [113]. The objective was now to compensate the entire delay between the server and client, which ranged from 0.1 s to 1 s.

These methods solely concentrate on the orientation data of the head movements. Another way of approaching this issue is to leverage the image content itself to detect regions within the scene the user might be interested in. Saliency maps, also referred to as attention maps, are created to indicate the visual attractiveness and eye fixation likelihoods of the captured scene. Further incorporating the temporal component into the prediction paradigm helps to improve the overall performance. Mavlankar *et al.* [114] adopted an auto-regressive moving average model to estimate the velocity of the viewport center and used motion vectors and navigation trajectories for viewport prediction. Recent approaches employ supervised learning techniques using (deep) neural networks to improve the fixation prediction [115]–[118]. Xu *et al.* [119] combined a head trajectory encoder module, which is premised on a Long Short-Term Memory (LSTM) network, and a saliency encoder module, which represents the Inception-ResNet-V2 module [120]. The inputs of the modules are the equirectangular projected footage and the viewport saliency and motion maps. Both modules are merged with fully connected dense layers to predict the displacement in the head trajectory. The combination of these two modules outperforms the individual components of only using head orientation, saliency maps, or the motion maps.

As opposed to the state-of-the-art, the study presented in this thesis puts special emphasis on the sensory data as the main source of information rather than putting head orientation data on a level with spatial and temporal scene information. A prediction paradigm is pro-

posed that performs superior head-motion prediction independent of textures and features available within the scene. Saliency maps and motion maps are leveraged as an additional information source to further optimize the forecasting. A probabilistic and several machine learning-based head-motion prediction algorithms are presented, which aim to optimize the achievable degree of delay-compensation. Various deep neural networks are proposed and evaluated. The best deep network is based on stacked Gated Recurrent Units and convolution layers to extract the most distinct features at different granularities. The proposed network is realtime-capable and excels the state-of-the-art.

2.7 Chapter Summary

This chapter introduced the most relevant background needed to better comprehend the reasoning discussed and the concepts proposed in this work. The first part addressed the basics of projective geometry and view synthesis, which are then used to detail the acquisition strategies for the creation of panoramic images. The transition from monoscopic to stereoscopic image acquisition is made to emphasize the challenges of acquiring stereoscopic $360°$ footage. State-of-the-art technologies that approach this issue are presented and thoroughly debated. Further topics that thematically relate to the overall goal of immersive telepresence, such as visual communication and viewport prediction, are further introduced and discussed. The prior arts' advantages and drawbacks are compared to the properties of the proposed delay-compensation vision system that uses only a minimum amount of two cameras mounted on a pan-tilt-roll-unit to mimic the user's head-motion and applies a novel delay-compensation paradigm for immersive telepresence.

Chapter 3

Experimental Setup and Evaluation Metrics

This chapter briefly revisits the structural basics of the MAVI telepresence platform that is used to test the algorithms developed in this study. For reproducible and comparable validations, two head motion datasets are used and described in the following. Qualitative measures are further introduced that are used as metrics to compare different approaches.

Parts of the work presented in this thesis have been published in [1], [2], [4]–[6].

3.1 MAVI Telepresence Platform

The approaches and algorithms presented in this study are implemented and tested on the MAVI (MAchine Vision and Interaction) research platform [3]. The semi-autonomous, low-cost MAVI robot is made of (mainly) self-designed 3D printed parts. It is characterized by a four-mecanum-wheel-based omnidirectional locomotion system and exhibits high manipulation capabilities each of which being at a budget price. Most relevant for this dissertation is its realtime-capable 3D 360° vision system, which is able to provide 3D omnidirectional vision in realtime by means of the algorithms and techniques proposed in this work. Figure 3.1 (a) illustrates the main components and dimensions of the MAVI platform and its vision system. The conceptual design and development of the delay compensation vision system (DCVS) underwent multiple iterations. The first version deployed a linearly actuated monocular camera mounted on a pan-tilt-unit (PT-U) to mimic the 2 DoF of the user's head movement. The second version extended the first version by a low-cost stereo-camera system using two web-cameras. Version 3 added the roll rotation as another degree-of-freedom and applied professional industrial cameras. The latest version, being a pan-tilt-roll-unit, considers all 3 DoF and employs high-resolution fisheye cameras for a large spatial coverage of the remote scene. All four versions are depicted in Figure 3.1 (b). The technical specifications of the PTR-U and the cameras are summarized in Table 3.1.

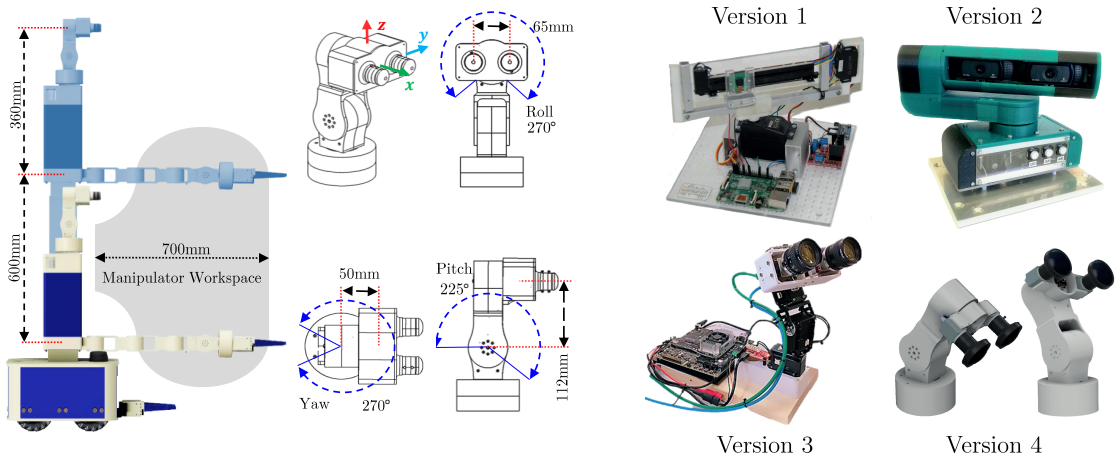


Figure 3.1: (a) Side-view of the developed MAVI telepresence platform showing the working spaces of the sensor head and the manipulator (adopted from [1]). (b) Evolving history of the designed sensor head as a delay-compensation vision system. The final version deploys two fisheye cameras to capture a larger visual field compared to the user's viewport size. The binocular vision system is mounted on a pan-tilt-roll-unit that is able to mimic the 3 DoF of a user's head-motion.

	Hardware specifications of the DCVS	
2K Ximea Camera	Part number	MC050CG-SY
	Resolution	5 MP 2464 × 2056 pixels
	Frame rate	76 fps
	Sensor model	Sony IMX250 LQR-C
	Sensor size	2/3"
	Readout method	Global shutter
	Data interface	USB 3.1 Gen 1
Fisheye lens	Part number	FE185C086HA-1
	Focal length (mm)	2.7
	Focus	Fixed
	Field-of-view (HxV)	185° × 140°35′
	Mount	C
Servo motors	Part number	Dynamixel MX-106T
	Microcontroller unit	St Cortex-M3 (STM32F103C8 @ 72 MHz, 32 bit)
	Position sensor	Contactless absolute encoder (12 bit, 360°)
	Motor	Maxon
	Baud rate	8000 bps ∼ 4.5 Mbps
	Resolution	0.088°
	Stall torque	8.4 N m @ 12 V, 5.2 A
	No load speed	45 rpm @ 12 V
	Link (Physical)	TTL Level Multi Drop Bus

Table 3.1: Hardware components and specifications for the delay compensation vision system consisting of the stereo-camera setup and the actuated pan-tilt-roll-unit.

The omnidirectional 3D vision system is built with a binocular camera setup, which is mounted on an actuated 3 DoF pan-tilt-roll-unit to be able to properly mimic the user's head motion. The concepts and algorithms introduced in this thesis are used to reduce the perceived delay when receiving the visual information of the remote environment the robot is located at. The pan-tilt-roll-unit follows the yaw, pitch, and roll Euler convention. Any desired head orientation triggered by the user is first converted to the quaternion representation. The correct conversion of an arbitrary head orientation q, expressed as quaternion $q = \begin{bmatrix} q_x & q_y & q_z & q_w \end{bmatrix}^T$, to the corresponding euler angles θ_c, ϕ_c, and ψ_c subject to the PTR-U design is obtained with:

$$\theta_c = \text{atan2}\left(-2\left(q_x q_y - q_w q_z\right), q_w^2 + q_x^2 - q_y^2 - q_z^2\right), \tag{3.1}$$

$$\phi_c = \text{asin}\left(2\left(q_x q_z + q_w q_y\right)\right), \tag{3.2}$$

$$\psi_c = \text{atan2}\left(-2\left(q_y q_z - q_w q_x\right), q_w^2 - q_x^2 - q_y^2 + q_z^2\right). \tag{3.3}$$

The baseline between the cameras, which corresponds to the interpupillary distance, is set to 6.5 cm but can be adjusted by about +/-1 cm according to the user's eye anatomy.

3.1.1 Head-motion Datasets

The MAVI telepresence platform was mainly deployed for demonstration and testing purposes. However, for solid and comparable validation of the proposed delay-compensation algorithms, two independent datasets with real head-motion user profiles are utilized.

3.1.1.1 LMT Dataset

The LMT dataset was specifically recorded for this work and includes real head-motion profiles of 30 participants ($P = 30$). The subjects were aged between 22 to 40 and watched three dissimilar (monoscopic) 360° video sequences with changing dynamics in the respective scene. The video sequences that are used for the subjective study are briefly summarized and described in Table 3.2. The immersive experience was increased by equipping the participants with HMDs when watching the panorama video content. The customized HMD was outfitted with an inertial measurement unit (IMU) to record the head orientation values. The IMU sensor provided filtered orientation data at a frequency of $f_{\text{IMU}} = 80\,\text{Hz}$. The entire dataset contains $30 \cdot 3 = 90$ subsets, each with an average video sequence length of 120 s. Considering the IMU's sampling rate, approximately 864,000 data points are at hand for experimental validation.

Youtube ID	Name	Content description	Resolution	Frame rate	Offset
gen0NgJjry4	Buckingham Palace	Buckingham Palace expedition. Guided tour through some of the Palace's state rooms.	2560×1440	30 fps	116 s
_YnXw93oU70	Formula E	Formula E's race highlights from Buenos Aires.	2560×1440	30 fps	4 s
7T57kzGQGto	Lion King	*Circle of Life* in 360° | The Lion King on Broadway.	2560×1440	30 fps	115 s

Table 3.2: Specification and description of the utilized 360° video sequences to which the participants were exposed to during the subjective study using the LMT dataset. The head-motion profiles of 30 participants were recorded while exposing them to three video sequences with varying dynamics in the scene for approximately 120 s. An IMU was mounted on the HMD to capture the filtered orientation data at a sampling rate of 80 Hz.

3.1.1.2 IMT Dataset

The second dataset that is utilized for experimental validation is provided by [121] and is referred to as the IMT dataset in the remainder of this thesis. The IMT dataset was recorded with completely different participants and video sequences under dissimilar conditions. Corbillon *et al.* [121] captured the head-motion profiles of 58 subjects, while each of which was watching five 360° video sequences for approximately 70 s. These video sequences are briefly described and specified in Table 3.3.

The IMT dataset represents the head movements of each participant as quaternions. A head-motion is only recorded if a motion occurs that is differing from the previous one. Due to its event-based character, the IMT dataset does not follow a constant sampling rate. A pre-processing step is thus required to obtain a constant sampling frequency and to convert the quaternions to Euler angles. Bi-linear interpolation is deployed to provide a sampling rate of 80 Hz that is alike to the one in the LMT dataset.

Youtube ID	Name	Content description	Resolution	Framerate	Offset
8lsB-P8nGSM	Rollercoaster	Rollercoaster. Rapid moving camera fixed in front of a moving rollercoaster. Field-of-view is following the rollercoaster trail.	3840×2048	30 fps	65 s
CIw8R8thnm8	Timelapse	Timelapse of city streets. Fixed camera, clear horizon with a lot of fast moving people/cars, many scene cuts. Focus expected along the equator line.	2.560×1.440	30 fps	4 s
sJxiPiAaB4k	Paris	Guided tour of Paris. Static camera with some smooth scene cuts. Focus expected along the equator line	2.560×1.440	60 fps	0 s
2OzlksZBTiA	Diving	Diving scene. Slowly moving camera, no clear horizon. No main focus expected within the sphere.	2.560×1.440	30 fps	40 s
s-AJRFQuAtE	Venice	Virtual aerial reconstruction of Venice. Slowly moving camera. No main focus expected within the sphere.	2.560×1.440	25 fps	0 s

Table 3.3: Specification and description of the utilized 360° video sequences, which are displayed to the participants during the subjective study using the IMT dataset. The head motion profiles of 58 participants were recorded while exposing them to five video sequences for approximately 70 s (reproduced from [121]).

3.1.2 Omnistereoscopic Offline Footage

Subjective experiments that addressed omnistereoscopic vision were conducted by exploiting an exisiting 3D 360° footage that was kindly provided by the Fraunhofer HHI [122]. Such a dataset helps to execute meaningful and reproducible comparision. A snapshot of the footage is depicted in Figure 3.2. The stereoscopic panorama videos show an everyday scene in an apartment where people are talking and interacting with each other.

Figure 3.2: Snapshot of a rendered video frame in normal mode (top) and the HMD mode (bottom), where the footage is pre-distorted (HMD lens) such that it can be visualized onto an HMD distortion-free. The omnistereoscopic video sequence is kindly provided by the Fraunhofer HHI [122].

For both the (offline) omnistereoscopic footage and the live telepresence sessions using the MAVI robot platform, a server-/client-based setup was implemented, where the 3D 360° visual representation of the scene is provided at the server-side according to the requested head orientation of the user. The client-side records the user's head motion positions, is responsible for selecting and rendering the correct viewport, and predicts prospective head positions. Both a TCP (transmission control protocol) and a UDP (user datagram protocol) connection between the server and the client are established to exchange visual and sensory information. The TCP connection is used to reliably stream the visual content employing the realtime streaming protocol (RTSP). Both frames for the left and the right eye are stacked and compressed with the H.264/AVC codec [86]. The *x264* implementation [123] of the H.264/AVC standard [86] is deployed for realtime performance. The *baseline* profile is selected and the preset configuration is set to *ultrafast*. The codec is further tuned with the *zerolatency* and *fastdecode* configuration to optimize for fast encoding and low latency streaming. Computationally demanding filters such as the deblocking filter and the CABAC

(context-adaptive binary arithmetic coding) are disabled to allow for fast decoding on devices with lower computational power. Due to the incongruity between the camera frame rate and the IMU's data acquisition rate, UDP is used to independently transmit the current head orientation quickly and efficiently. Due to the high frequency of IMU data, a dropped packet of orientation data has no perceivable impact on the quality of experience. The communication delay is set constant for investigated latencies between 0.1 s to 1 s with a step size of 0.1 s employing a network emulator [124].

3.1.3 Evaluation Metrics

Three qualitative measures are deployed to assess the quality and performance of the algorithms proposed in this work. The mean absolute error (MAE) and the root mean square error (RMSE) are used to determine the accuracy of predicted orientation values $\hat{\boldsymbol{\xi}}_{h,\boldsymbol{\Gamma}}(t+\tau)$ given an arbitrary head-motion predictor $\boldsymbol{\Gamma}(\boldsymbol{\xi}_h(t),\tau)$ with respect to the correct values $\boldsymbol{\xi}_h(t+\tau)$ for a present end-to-end latency τ. The assessment is done both for the LMT and the IMT dataset by considering the amount of all videos (V), participants (P), and the samples for each video sequences (S) according to:

$$\mathrm{MAE}(\boldsymbol{\Gamma}(\boldsymbol{\xi}_h(t),\tau)) = \frac{1}{V\cdot P\cdot S}\cdot\sum_{i=1}^{P}\sum_{j=1}^{V}\sum_{k=1}^{S}\left|\hat{\boldsymbol{\xi}}_{h,\boldsymbol{\Gamma}}^{i,j,k}-\boldsymbol{\xi}_h^{i,j,k}\right|, \tag{3.4}$$

$$\mathrm{RMSE}(\boldsymbol{\Gamma}(\boldsymbol{\xi}_h(t),\tau)) = \sqrt{\frac{1}{V\cdot P\cdot S}\cdot\sum_{i=1}^{P}\sum_{j=1}^{V}\sum_{k=1}^{S}\left(\hat{\boldsymbol{\xi}}_{h,\boldsymbol{\Gamma}}^{i,j,k}-\boldsymbol{\xi}_h^{i,j,k}\right)^2}, \tag{3.5}$$

$$\text{with}\quad S = \frac{1}{f_{\mathrm{IMU}}}\cdot\left(t_{\mathrm{end}}^{V,P}-t_{\mathrm{start}}^{V,P}\right). \tag{3.6}$$

The achievable degree of delay compensation is further assessed by harnessing the so-called *(delay) compensation rate* c, which is algebraically derived in Chapter 4. The compensation rate is a scalar value between 0 – 1 (0 – 100 %) and denotes the retrievable degree of delay-compensation. A compensation rate of 1 (100 %) corresponds to a full compensation. Similarly to the MAE and RMSE, the mean compensation rate $\bar{c}$ is computed over all persons, video sequences, and samples to compare different approaches:

$$\bar{c} = \frac{1}{V\cdot P\cdot S}\cdot\sum_{i=1}^{P}\sum_{j=1}^{V}\sum_{k=1}^{S}c^{i,j,k}. \tag{3.7}$$

One drawback of this method is that the degree of compensation is averaged for all available samples; even for situations where no motion happens. A more meaningful metric is given by the so-called *event-based compensation rate*, which will accumulate only those values, where a motion greater than a threshold value ϵ occurs. The event-based compensation rate $\hat{\bar{c}}$ is quantified as:

$$\hat{\bar{c}} = \frac{1}{V\cdot P\cdot\sum_{k=1}^{S}\delta(a_k)}\cdot\sum_{i=1}^{P}\sum_{j=1}^{V}\sum_{k=1}^{S}\delta(a_k)\cdot c^{i,j,k}, \tag{3.8}$$

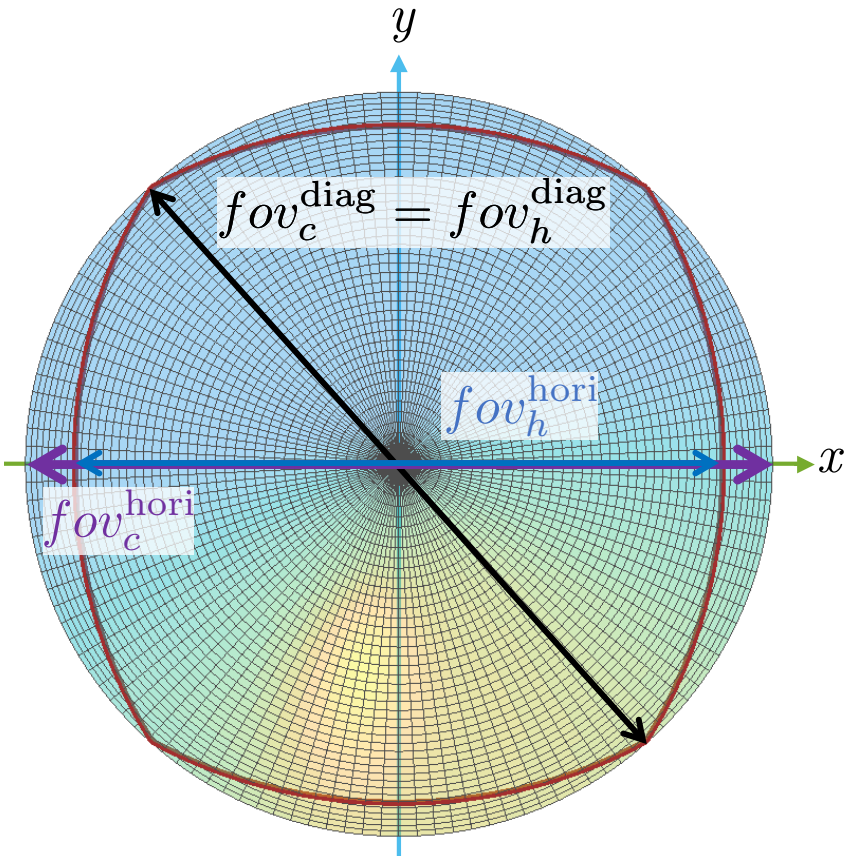


Figure 3.3: Illustration of why the use of diagonal field-of-views is more indicative and suitable for the determination of the true amount of available buffer region compared to the horizontal and vertical ones. The front view of the fisheye image plane is displayed as circular image surface. The borders of the HMD image plane that are mapped onto the fisheye image plane are depicted as dark red lines. Using the horizontal field-of-view indicates a buffer size greater than zero. Leveraging, however, the diagonal ones, it is obvious that the buffer equals zero (adopted from [6] © 2018 IEEE).

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

$$a_k = \begin{cases} 0, & \text{if } \{|\dot{\theta}_k| \text{ or } |\dot{\phi}_k| \text{ or } |\dot{\psi}_k|\} \geq 0 + \epsilon \\ \neq 0, & \text{otherwise.} \end{cases} \tag{3.9}$$

In this way, only those compensation rates are taken into account, where at least one of the head rotation values is above the IMU's sensor noise ($\epsilon = 5°$).

3.1.4 Buffer Sizes

The delay compensation paradigm comprises three individual components. A fundamental principle is to deploy a camera that captures imagery with a larger visual field than is actually displayed to the user. The size of the formed buffer region has a significant impact on the achievable level of delay-compensation. The amount of buffer is computed by means of the cameras and user's field-of-view. An intuitive approach would be to use either the horizontal or vertical field-of-views of the user and the camera, respectively, to express the

amount of available extra content. The horizontal and vertical field-of-views of the camera (fov_c^{hori}, fov_c^{vert}) are usually dictated by the underlying camera sensor. The horizontal and vertical field-of-views of the user (or the HMD) (fov_h^{hori}, fov_h^{vert}) are design parameters and can be modified upon request. Within the scope of this work, the horizontal field-of-view offered to the user varied between 60 ° to 120 °, while keeping a ratio of 8:9 to be in accordance with the HMD's expected aspect ratio ($ratio$). The corresponding vertical field-of-view of the HMD can then be deduced as:

$$fov_h^{\text{vert}} = 2 \cdot \arctan\left(\frac{1}{ratio} \cdot \tan\left(\frac{fov_h^{\text{hori}}}{2}\right)\right). \tag{3.10}$$

Figure 3.3 illustrates why the horizontal and vertical field-of-views are not suitable for the determination of the available buffer size. For instance, a horizontal fov_h^{hori} of the user less than the fov_c^{hori} of an equidistant fisheye camera does imply a larger buffer size. However, a more indicative measure is given by the diagonal field-of-view fov_h^{diag}, which can be computed as:

$$fov_h^{\text{diag}} = 2 \cdot \arctan\left(\sqrt{\tan^2\left(\frac{fov_h^{\text{hori}}}{2}\right) + \tan^2\left(\frac{fov_h^{\text{vert}}}{2}\right)}\right). \tag{3.11}$$

Equation 3.11 can also be utilized to compute the diagonal field-of-view of perspective cameras. This computation is void for equidistant fisheye cameras where $fov_c^{\text{vert}} = fov_c^{\text{hori}} = fov_c^{\text{diag}}$.

Considering these facts, the diagonal buffer b^{diag} is chosen as a more informative measure to constitute the available amount of buffer. The diagonal buffer b^{diag} can be computed as:

$$b^{\text{diag}} = \tfrac{1}{2}(fov_c^{\text{diag}} - fov_h^{\text{diag}}). \tag{3.12}$$

Table 3.4 illustrates some example buffer sizes for different HMD viewport sizes and a fisheye camera with $fov_c^{\text{vert}} = fov_c^{\text{hori}} = fov_c^{\text{diag}} = 150°$.

Subscript {}	$\boldsymbol{fov}_{\{\}}^{\text{hori}}$ [°]	$\boldsymbol{fov}_{\{\}}^{\text{vert}}$ [°]	$\boldsymbol{fov}_{\{\}}^{\text{diag}}$ [°]	$\boldsymbol{b}^{\text{diag}}$ [°]
Camera c	150	150	150	-
HMD/User h	60	66.01	81.98	**34.01**
	90	96.73	112.80	**18.60**
	120	125.67	138.03	**5.98**

Table 3.4: Investigated horizontal, vertical, and diagonal field-of-views for the camera and the HMD's viewport size

3.2 Chapter Summary

This chapter introduced the experimental setup that is used to evaluate the proposed approaches and compare them to the state-of-the-art. After a general overview of the developed semi-autonomous MAVI telepresence robot platform, special emphasis is placed on its sensor head, a binocular camera system mounted on a 3 DoF actuated, electro-mechanical component that is used to mirror the user's head movements. While this setup serves for demonstration purposes, it is of great importance to evaluate the proposed concepts with qualitative measures for a fair comparison. Within the scope of this work, the mean absolute error, the root mean square error, and the event-based compensation rate are used as metrics to prove the general validity of the presented methods. Two independently recorded datasets are employed that provide real head-motion trajectories of numerous user's which are exposed to different video sequences with varying dynamics in the scene.

Chapter 4

Buffer-based Delay-compensation

This chapter introduces the theory behind the buffer-based delay-compensation approach and provides a mathematically-founded description of the compensation rate, which is a novel metric to convey the degree of delay-compensation.

Parts of the work presented in this chapter have been published in [1], [2], [6].

4.1 Problem Statement

Providing the user an immersive telepresence experience necessitates a 3D 360° visual representation of the remote scene. HMDs are often deployed as an assistive device to extract the desired viewport more intuitively. The lag between head-motion and visual response, which is particularly critical for telepresence applications, is a QoE-limiting factor that might result in motion sickness and nausea. To provide a delay-free and realistic visual impression, it is of vital importance to keep the motion-to-photon latency as low as possible. Rather than using a bulky and computationally complex multi-camera setup, a realtime-capable stereo-camera system is selected and mounted on a pan-tilt-(roll-)unit to mirror the user's head-motion. The range of motion was initially restricted to only 2 DoF. In its raw setup, such a system introduces a severe lag between head-motion and visual response, which leads to immediate visual discomfort. In addition to the communication delay, there exist further stalling elements that are added to the end-to-end latency τ:

$$\tau = \underbrace{t_s + t_m + t_c + t_p + t_r}_{\text{inherent delay}} + 2 \cdot t_n, \quad \text{with} \tag{4.1}$$

- t_s: Sampling rate of the orientation sensor in the HMD,
- t_m: Mechanical delay of the PTR-U,
- t_c: Camera delay ($t_c = \frac{1}{f_c}$), with f_c being the camera's frame rate,
- t_p: Processing latency: rectification, encoding, and processing of the captured images prior to streaming,
- t_r: Rendering stall: processes of extracting the frames from the received image data, decoding them, and rendering them onto the HMD,
- t_n: One-way communication network delay.

The aggregation of latencies outlined in Equation (4.1) demonstrates that the communication delay is not the only lag that needs to be accounted for. Even for local streaming sessions there exist an accumulation of latencies, referred to as inherent delay, that are typically in the range of 150 ms to 200 ms. The buffer-based delay-compensation technique is presented as a remedy to support the user with an omnistereoscopic visual impression despite the presence of QoE-harming latencies. Immediate access to visual content is ensured by deploying cameras that cover a larger visual field (fov_{c}) than is called for by the HMD's viewport (fov_{h}). A defined buffer margin is thereby created around the requested viewport. The head-motion is decoupled from the electro-mechanical actuation unit to be able to leverage the extra image content for local delay-compensation until the update image frames arrive. This technique links each requested gaze direction with an instantaneous visual response and ensures thereby the user's perception of ego-motion to match with the sensory inputs from the visual system, the vestibular system, and the non-vestibular proprioceptors mitigating the emergence of motion sickness [15]. In a traditional pan-tilt-(roll)-unit-based two-camera setup, there is an unavoidable discrepancy between head-motion and visual feedback. The viewport would remain frozen given the fact that the requested imagery at the new head position is not available yet due to the present delay. The technique applied in this work, decouples the head-motion from the actuation and first changes the region-of-interest (ROI) according to the requested viewing direction prior to updating the visual information. The achievable level of delay-compensation is highly subject to the head-motion velocity, the provided buffer size, and the end-to-end latency. This chapter first illustrates the proposed delay-compensation technique for 1D, horizontal motions and gradually extends it to all three dimensions of the adopted pan-tilt-roll-unit. The compensation model is algebraically described both for perspective and fisheye cameras. The chapter concludes with a new type of generic approach that is agnostic to the deployed camera system and works thus both for perspective and fisheye cameras.

4.2 Perspective Cameras

This section introduces the conceptual basis of the buffer-based delay-compensation approach, particularly tailored for horizontal head-motions, which are known to be the most dominant ones (see Chapter 6). Assuming a camera system that records a broader horizontal visual field (fov_c^{hori}) than is originally needed to visualize the image content on the HMD's viewport (fov_h^{hori}) allocates a buffer b^{hori} that can be used for local delay-compensation:

$$b^{\mathrm{hori}} = \tfrac{1}{2}(fov_c^{\mathrm{hori}} - fov_h^{\mathrm{hori}}). \tag{4.2}$$

Figure 4.1 illustrates the concept of the buffer exploitation process. Immediate visual response is facilitated by adapting the region-of-interest with respect to the requested viewing direction $\boldsymbol{v}$. Merely shifting the region-of-interest does not account for the perspective change that befalls when moving the head. The reason for this is that the requested gaze direction $\boldsymbol{v}$ is not perpendicular to the present image content $\boldsymbol{I}_{\mathrm{h}}$. This issue is approached by first mapping the visual data onto a common cylinder that ensures the gaze direction to be

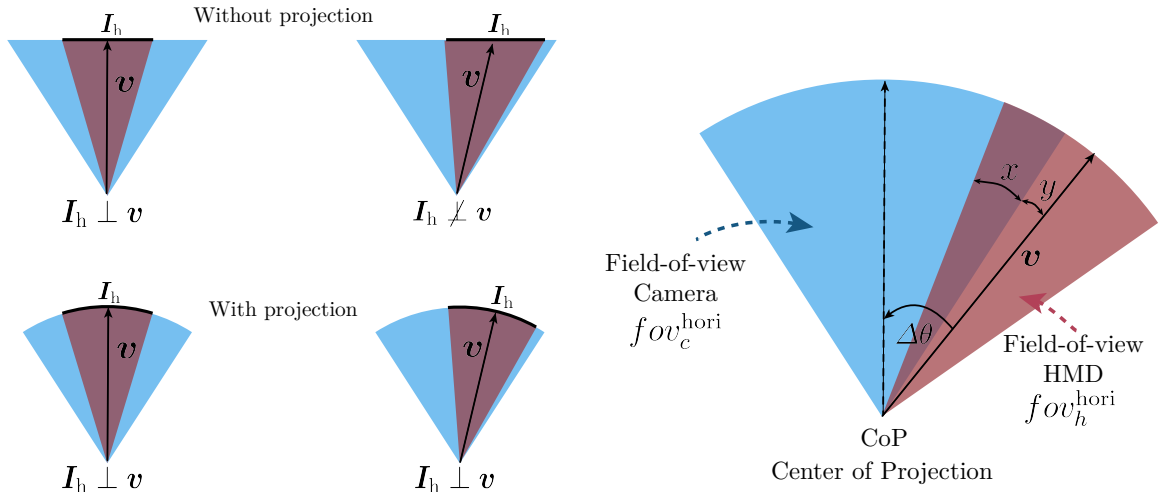


Figure 4.1: Left: The region-of-interest is shifted according to the request viewing direction. The eventuated perspective change is addressed by mapping the images onto a shared cylinder surface. Right: The buffer-based delay-compensation can not always ensure a full compensation, especially for fast head rotations. The retrievable level of delay-compensation is expressed as the ratio between accessible image content and the size of the user's visual field (reproduced from [2] © 2019 IEEE).

perpendicular to the cylinder surface for all conceivable horizontal motion as is visualized in Figure 4.1.

Any pixel (x, y) in the image $\boldsymbol{I}_\text{h}$ is projected onto the common cylinder surface according to:

$$x' = r \cdot \arctan\left(\frac{x}{r}\right), \tag{4.3}$$

$$y' = y \cdot \frac{r}{\sqrt{x^2 + r^2}}, \tag{4.4}$$

with (x', y') being the projected cylinder coordinates and $r = f$ the cylinder radius. The compensation rate c_p is introduced as a measure to express the reachable degree of delay-compensation associated with horizontal (pan) motions. It is defined as the ratio of the accessible image content with respect to the pan direction and the user's horizontal field-of-view (fov_h^{hori}) that is required to display the footage onto the HMD:

$$c_p = \frac{x}{fov_h^{\text{hori}}}, \quad \text{with} \tag{4.5}$$

$$x = \frac{fov_h^{\text{hori}}}{2} - y, \quad \text{and} \quad y = \Delta\theta - \frac{fov_c^{\text{hori}}}{2}. \tag{4.6}$$

Inserting Equation (4.6) into Equation (4.5) yields the formula for the (pan) compensation rate c_p, which can be described with respect to the change in orientation $\Delta\theta = |\theta_h - \theta_c|$ or the angular velocity in pan direction $\dot{\theta}_h$:

$$\begin{aligned} c_p &= \frac{\frac{1}{2}(fov_h^{\text{hori}} + fov_c^{\text{hori}}) - \Delta\theta}{fov_h^{\text{hori}}} \\ &= \frac{\frac{1}{2}(fov_h^{\text{hori}} + fov_c^{\text{hori}}) - \dot{\theta}_h \cdot \tau}{fov_h^{\text{hori}}}. \end{aligned} \tag{4.7}$$

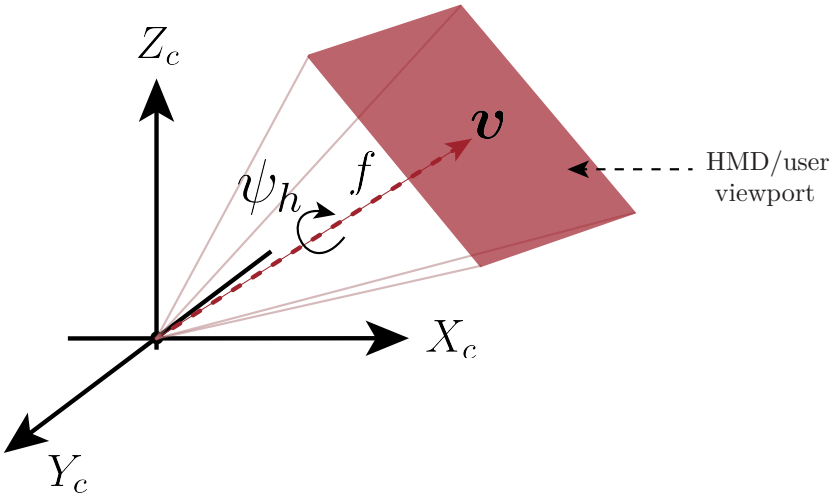


Figure 4.2: After determining the pan and tilt motion by means of Euler angles, the Rodriguez formula is used to independently express the roll rotation as a revolution around the optical axis. It is implicitly defined by the viewing direction of the user and can be any arbitrary axis that goes through the origin.

The degree of delay-compensation c^* can then be quantified as:

$$c^* = \begin{cases} 1 & c_p \geq 1 \\ c_p & 0 \leq c_p < 1 \\ 0 & c_p < 0, \end{cases} \tag{4.8}$$

where $c^* = 1$ corresponds to a full (100 %) compensation.

4.2.1 Extension to 3 DoF

This section upgrades the delay-compensation model to 3 DoF with respect to perspective cameras. Instead of using a shared cylinder as a common surface, the image data is mapped onto a sphere to address the unavoidable perspective change. Any pixel pair within the captured image (x, y) can be projected onto the common sphere, whose radius $r_{\text{sph}} = f$ is defined to equal the camera's focal length f:

$$\begin{aligned} x' &= f \cdot \arctan\left(\frac{x}{f}\right), \\ y' &= f \cdot \arctan\left(\frac{y}{f}\right). \end{aligned} \tag{4.9}$$

The orientation vector $\boldsymbol{\xi}_{\{h,c\}} = [\theta_{\{h,c\}} \;\; \phi_{\{h,c\}} \;\; \psi_{\{h,c\}}]^T$ describes the angular rotations around the $Y-$axis, $X-$axis and the $Z-$axis for either the user's head-motion (subscript: h) or the camera system (subscript: c). $\Delta\boldsymbol{\xi} = [\Delta\theta \;\; \Delta\phi \;\; \Delta\psi]^T$ expresses the disparity between the user's head-motion and the camera's orientation as per:

$$\Delta\theta = |\theta_h - \theta_c|, \tag{4.10}$$

$$\Delta\phi = |\phi_h - \phi_c|, \tag{4.11}$$

$$\Delta\psi = |\psi_h - \psi_c|. \tag{4.12}$$

The angular deviations are used to express the relative rotation $\boldsymbol{R}$ that is needed to reach the requested head orientation of the user. The rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times3}$ is the product of the individual rotation matrices of the pan, tilt, and roll revolutions:

$$\boldsymbol{R} = \boldsymbol{R}_{\psi} \cdot \boldsymbol{R}_{\phi} \cdot \boldsymbol{R}_{\theta}. \tag{4.13}$$

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

$$\boldsymbol{R}_{\theta} = \begin{bmatrix} \cos(\Delta\theta) & 0 & \sin(\Delta\theta) \\ 0 & 1 & 0 \\ -\sin(\Delta\theta) & 0 & \cos(\Delta\theta) \end{bmatrix}, \tag{4.14}$$

$$\boldsymbol{R}_{\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\Delta\phi) & -\sin(\Delta\phi) \\ 0 & \sin(\Delta\phi) & \cos(\Delta\phi) \end{bmatrix}, \tag{4.15}$$

the roll rotation is set to be a revolution around the optical axis, as shown in Figure 4.2. The viewing direction $\boldsymbol{v}$ of the optical axis is thereby subject to the present pan and tilt rotations and can be expressed as:

$$\boldsymbol{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \boldsymbol{R}_{\phi} \cdot \boldsymbol{R}_{\theta} \cdot \begin{bmatrix} 0 \\ 0 \\ f \end{bmatrix}. \tag{4.16}$$

To enforce the roll rotation to be a revolution around the optical axis, its rotation matrix needs to be computed according to [125]:

$$\boldsymbol{R}_{\psi} = \cos(\Delta\psi) + (1 - \cos(\Delta\psi)) \cdot \frac{\boldsymbol{v} \otimes \boldsymbol{v}}{|\boldsymbol{v}|^2} + \frac{\sin(\Delta\psi)}{|\boldsymbol{v}|} \cdot [\boldsymbol{v}]_{\times} \tag{4.17}$$

$$= \begin{bmatrix} k_1 v_x^2 + k_2 & k_1 v_x v_y - k_3 v_z & k_1 v_x v_z + k_3 v_y \\ k_1 v_x v_y + k_3 v_z & k_1 v_y^2 + k_2 & k_1 v_y v_z - k_3 v_x \\ k_1 v_x v_z - k_3 v_y & k_1 v_y v_z + k_3 v_x & k_1 v_z^2 + k_2 \end{bmatrix}, \quad \text{with} \tag{4.18}$$

$$k_1 = \frac{1 - \cos(\Delta\psi)}{|\boldsymbol{v}|^2}, \quad k_2 = \cos(\Delta\psi), \quad \text{and} \quad k_3 = \frac{\sin(\Delta\psi)}{|\boldsymbol{v}|}. \tag{4.19}$$

After mapping the accessible imagery onto the common sphere, the region-of-interest is adapted with respect to the requested head orientation to ensure visual response in an instant. The ratio between the size of available image content $area(\boldsymbol{\Pi})$ and the user viewport's size constitutes the degree of delay-compensation:

$$c_{ptr} = \frac{area(\boldsymbol{\Pi})}{fov_h^{\text{hori}} \cdot fov_h^{\text{vert}}}. \tag{4.20}$$

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

$$fov_h^{\text{vert}} = 2 \cdot \arctan\left(\frac{1}{ratio} \cdot \tan\left(\frac{fov_h^{\text{hori}}}{2}\right)\right), \tag{4.21}$$

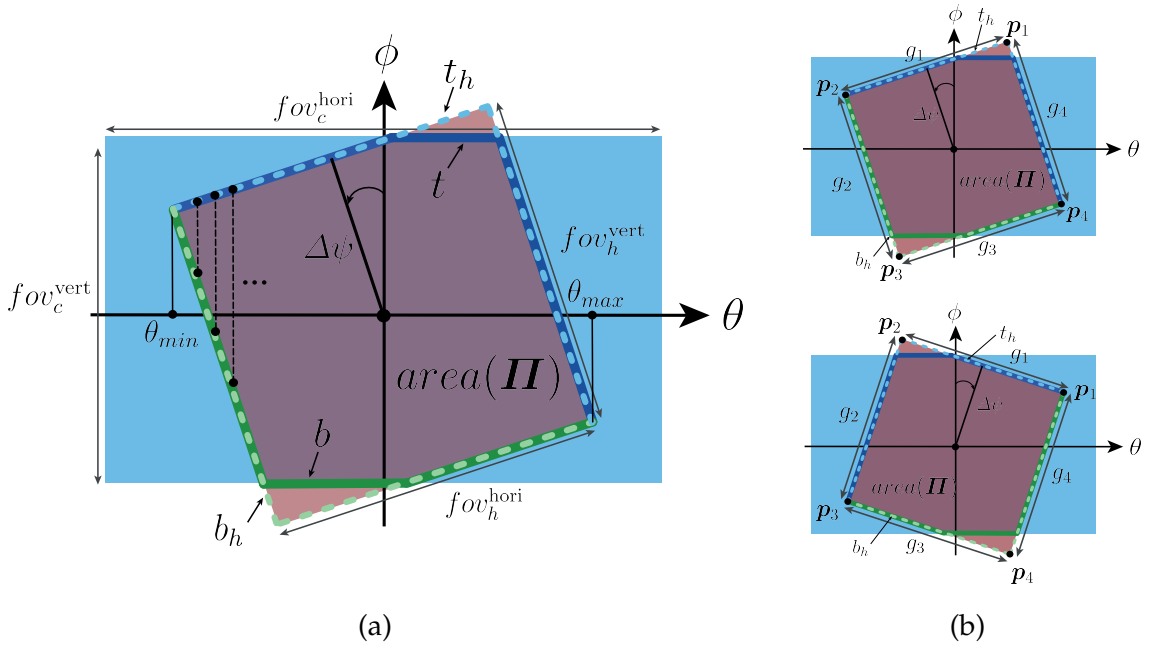


Figure 4.3: The delay-compensation model is shown for a perspective camera with respect to the 3 DoF of the user's head motion. The amount of reachable delay-compensation is visualized in spherical coordinates. (a) The accessible image content (blue) is demonstrated and described with respect to the requested viewing direction and its corresponding visual field after mapping them onto the common sphere. A scenario is depicted where only a partial compensation is obtained. (b) The construction of the auxiliary curves t and b is subject to the direction of the roll rotation and may change for different configurations (reproduced from [2] © 2019 IEEE).

where $ratio$ corresponds to the HMD's display aspect ratio. $\boldsymbol{\Pi}$ is the set of pixels that contains all available pixels in the image, which are accessible to be displayed on the HMD for the latest viewing direction. Considering all 3 DoF makes the computation of the $area(\boldsymbol{\Pi})$ more challenging than for the 1D case, as a rectangular shape is not always preserved (see Figure 4.3).

After projecting the image content onto the sphere, the overlapping $area(\boldsymbol{\Pi})$ can be calculated to describe the number of pixels that are available to be displayed. Within the spherical coordinate system, two auxiliary curves t and b are introduced to ease the computation of $area(\boldsymbol{\Pi})$ as follows:

$$area(\boldsymbol{\Pi}) = \int_{\theta_{min}}^{\theta_{max}} (t - b)\, d\theta. \tag{4.22}$$

A finite number of steps is used to integrate over θ. At each step, the corresponding maximum value of ϕ is assigned to the top curve t and likewise the minimum value thereof to the bottom curve b as is shown in Figure 4.3 (a). If there is only a single ϕ value available, then it is assigned to both curves. Depending on the present roll rotation, these curves can vary (see Figure 4.3). To compute t and b, two additional auxiliary curves t_h and b_h are used, which are, analogously to t and b, each constructed from two adjacent edge lines of the HMD frame:

$$t = \max\left(\min\left(t_h, \frac{fov_c^{\text{vert}}}{2}\right), -\frac{fov_c^{\text{vert}}}{2}\right), \tag{4.23}$$

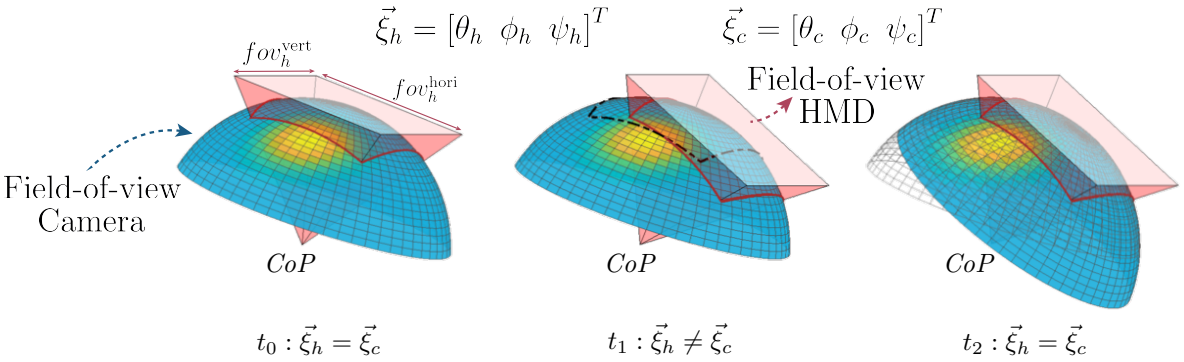


Figure 4.4: Overview of the generic buffer-based compensation approach with regard to the 3 DoF of the user's head motion. Cameras with an extended visual field (blue) are deployed to create a buffer margin around the user's viewport (red). The captured data is mapped onto a sphere to account for the perspective change. The region-of-interest is subsequently shifted according the user's latest head orientation. The extra image content is leveraged for local delay-compensation until the updated frame is received.

$$b = \min\left(\max\left(b_h, -\frac{fov_c^{\text{vert}}}{2}\right), \frac{fov_c^{\text{vert}}}{2}\right). \tag{4.24}$$

Inserting these two equations into Equation (4.22) allows us to reformulate the computation of $area(\mathbf{\Pi})$ to be:

$$\begin{aligned} area(\mathbf{\Pi}) = \int_{\theta_{min}}^{\theta_{max}} & \max\left(\min\left(t_h, \frac{fov_c^{\text{vert}}}{2}\right), -\frac{fov_c^{\text{vert}}}{2}\right) + \\ & - \min\left(\max\left(b_h, -\frac{fov_c^{\text{vert}}}{2}\right), \frac{fov_c^{\text{vert}}}{2}\right) \, d\theta. \end{aligned} \tag{4.25}$$

Figure 4.3 (b) demonstrates that the specification of t_h and b_h depends on the present disparity between the camera's and HMD's roll rotation. For $0° \leq \psi_c - \psi_h < 90°$ (see Figure 4.3 (b) (top)), t_h and t_b are defined as:

$$t_h = \begin{cases} g_1, & \theta \leq p_{1,\theta} \\ g_4, & \theta > p_{1,\theta} \end{cases}, \tag{4.26}$$

$$b_h = \begin{cases} g_2, & \theta \leq p_{3,\theta} \\ g_3, & \theta > p_{3,\theta} \end{cases}, \tag{4.27}$$

with $p_{i,\theta}$ being the first element of a corner point pair $\boldsymbol{p}_i = [p_{i,\theta} \ \ p_{i,\phi}]^{\text{T}}$ of the HMD's image plane $\forall i \in \{1, 2, 3, 4\}$. The bottom scheme of Figure 4.3 (b) illustrates the case where $-90° < \psi_c - \psi_h < 0°$. In this case, t_h and b_h are expressed as:

$$t_h = \begin{cases} g_2, & \theta \leq p_{2,\theta} \\ g_1, & \theta > p_{2,\theta}, \end{cases} \tag{4.28}$$

$$b_h = \begin{cases} g_3, & \theta \leq p_{4,\theta} \\ g_4, & \theta > p_{4,\theta}. \end{cases} \tag{4.29}$$

The lines g_k can be computed with the following formula where $\boldsymbol{p}_i = [p_{i,\theta}\ \ p_{i,\phi}]^T$ and $\boldsymbol{p}_j = [p_{j,\theta}\ \ p_{j,\phi}]^T$ are the two end points of the line g_k with $i, j, k \in \{1, 2, 3, 4\} \wedge i \neq j$:

$$g_k = \frac{p_{j,\phi} - p_{i,\phi}}{p_{j,\theta} - p_{i,\theta}} \cdot \theta + \frac{p_{i,\theta} \cdot p_{j,\phi} - p_{j,\theta} \cdot p_{i,\phi}}{p_{i,\theta} - p_{j,\theta}}. \tag{4.30}$$

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

$$\boldsymbol{p}_1 = \boldsymbol{R}_{\text{2D}}(\Delta\psi) \cdot \begin{bmatrix} fov_h^{\text{hori}}/2 \\ fov_h^{\text{vert}}/2 \end{bmatrix} + \begin{bmatrix} \Delta\theta \\ \Delta\phi \end{bmatrix}, \tag{4.31}$$

$$\boldsymbol{p}_2 = \boldsymbol{R}_{\text{2D}}(\Delta\psi) \cdot \begin{bmatrix} -fov_h^{\text{hori}}/2 \\ fov_h^{\text{vert}}/2 \end{bmatrix} + \begin{bmatrix} \Delta\theta \\ \Delta\phi \end{bmatrix}, \tag{4.32}$$

$$\boldsymbol{p}_3 = \boldsymbol{R}_{\text{2D}}(\Delta\psi) \cdot \begin{bmatrix} -fov_h^{\text{hori}}/2 \\ -fov_h^{\text{vert}}/2 \end{bmatrix} + \begin{bmatrix} \Delta\theta \\ \Delta\phi \end{bmatrix}, \tag{4.33}$$

$$\boldsymbol{p}_4 = \boldsymbol{R}_{\text{2D}}(\Delta\psi) \cdot \begin{bmatrix} fov_h^{\text{hori}}/2 \\ -fov_h^{\text{vert}}/2 \end{bmatrix} + \begin{bmatrix} \Delta\theta \\ \Delta\phi \end{bmatrix}. \tag{4.34}$$

The roll rotation matrix $\boldsymbol{R}_{\text{2D}}(\Delta\psi)$ in the 2D image plane is given as:

$$\boldsymbol{R}_{\text{2D}}(\Delta\psi) = \begin{bmatrix} \cos(\Delta\psi) & -\sin(\Delta\psi) \\ \sin(\Delta\psi) & \cos(\Delta\psi) \end{bmatrix}. \tag{4.35}$$

The bounds of integration θ_{min} and θ_{max} from Equation (4.25) are then computed with:

$$\theta_{min} = \max\left(-\frac{fov_c^{\text{hori}}}{2},\ \min(p_{2,\theta},\, p_{3,\theta})\right), \tag{4.36}$$

$$\theta_{max} = \min\left(\frac{fov_c^{\text{hori}}}{2},\ \max(p_{1,\theta},\, p_{4,\theta})\right). \tag{4.37}$$

4.3 Equidistant Fisheye Cameras

Fisheye cameras are another alternative to wide-angle perspective cameras to acquire wider fields of vision. The delay-compensation's working principle does conceptually not differ for fisheye cameras but necessitates a tailored model to mathematically describe the reachable degree of compensation. The image plane of fisheye cameras is spherical compared to the rectangular one of perspective cameras. This section aims to provide the theoretical foundations needed to compute the delay-compensation rate when using fisheye cameras. Figure 4.5 (a) illustrates the hemispherical image plane of fisheye cameras. The resulting image is distorted and requires a correction step prior to display. The images captured with fisheye cameras can be corrected by projecting them from its spherical image plane onto a rectangle, which corresponds to the HMD's image plane. The radius of the hemisphere is set to be equal to the focal length f of the fisheye camera. The mathematical model needed to

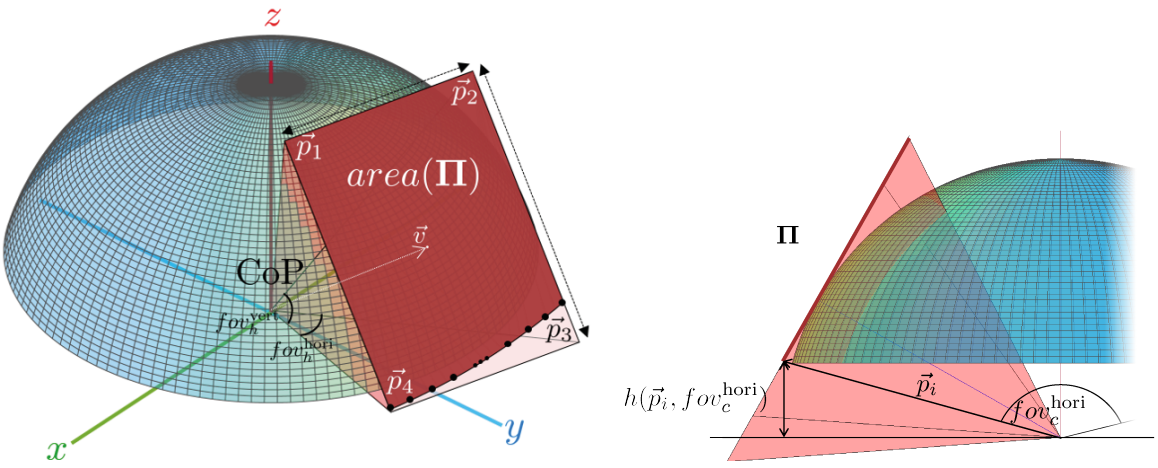


Figure 4.5: (a) The camera's and user's visual field are depicted for a scenario where only partial delay-compensation is achieved. Only the overlapping portions of the camera's image plane and the user's viewport can be used for display (dark red). Their edge points need to be specified to be able to assess the reachable degree of delay-compensation. (b) The auxiliary measure h is introduced to determine the minimum height $h(\boldsymbol{p}_i, fov_c^{\text{hori}})$ of an arbitrary image point $\boldsymbol{p}_i \in \boldsymbol{I}_h$ to be in $\mathbf{\Pi}$ [6]. This measure holds on a permanent basis as the camera's location is fixated in the 3D world. The scene, instead, is rotated to obtain the requested viewing direction (reproduced from [6] © 2018 IEEE).

describe the retrievable amount of delay-compensation for fisheye camera remains, conceptually, the same. The compensation rate is still defined as the ratio between the accessible amount of image content and the HMD's viewport size:

$$c_{ptr} = \frac{area(\mathbf{\Pi})}{l_h^{\text{hori}} \cdot l_h^{\text{vert}}}. \tag{4.38}$$

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

$$l_h^{\text{hori}} = 2 \cdot f \cdot \tan\left(\frac{fov_h^{\text{hori}}}{2}\right), \tag{4.39}$$

$$l_h^{\text{vert}} = 2 \cdot f \cdot \tan\left(\frac{fov_h^{\text{vert}}}{2}\right). \tag{4.40}$$

The edge and corner points are the minimum numbers of points needed to determine $area(\mathbf{\Pi})$. These points are summarized in the set $\mathbf{\Pi}_E \subset \mathbf{\Pi}$. The set $\mathbf{\Pi}_E$ contains the corner points and a sparse set of points on the curved line as shown in Figure 4.5 (a), which result from a partial availability of accessible image content with respect to the current head orientation. Figure 4.5 illustrates the 3D vertices of the HMD image plane $\boldsymbol{p}_m \; \forall m \in \{1, 2, 3, 4\}$, which are calculated as follows:

$$\boldsymbol{p}_1 = \boldsymbol{R} \cdot \begin{bmatrix} l_h^{\text{hori}}/2 \\ l_h^{\text{vert}}/2 \\ f \end{bmatrix}, \quad \boldsymbol{p}_2 = \boldsymbol{R} \cdot \begin{bmatrix} -l_h^{\text{hori}}/2 \\ l_h^{\text{vert}}/2 \\ f \end{bmatrix}, \tag{4.41}$$

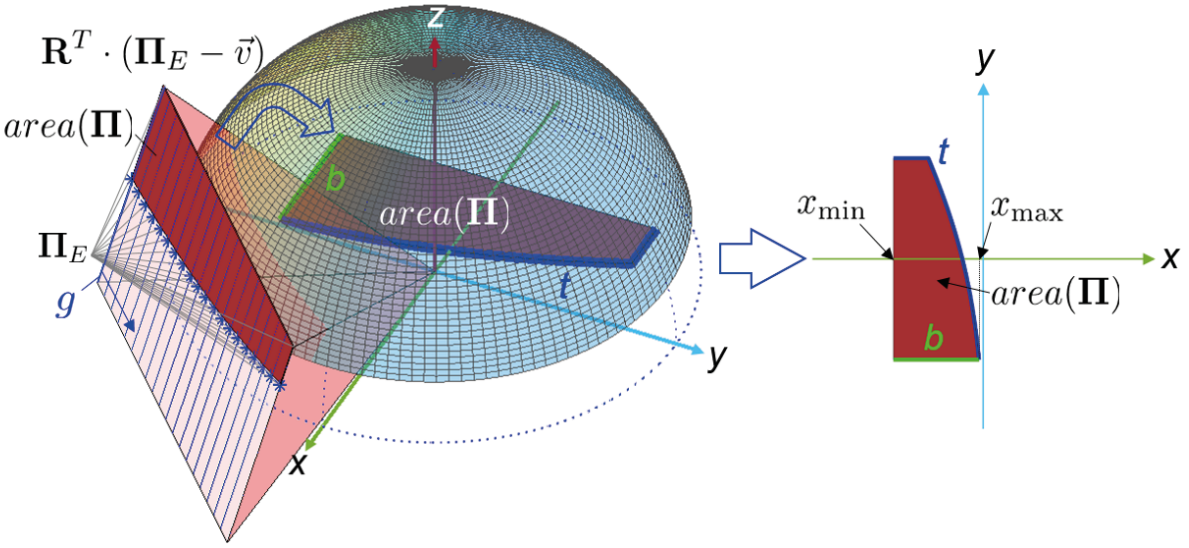


Figure 4.6: The specification of $area(\mathbf{\Pi})$ considering the integral over $(t-b)$ is simplified by mapping the set $\mathbf{\Pi}_E$ into the center of the $XY-$plane (reproduced from [6] © 2018 IEEE).

$$\boldsymbol{p}_3 = \boldsymbol{R} \cdot \begin{bmatrix} -l_h^{\text{hori}}/2 \\ -l_h^{\text{vert}}/2 \\ f \end{bmatrix}, \quad \boldsymbol{p}_4 = \boldsymbol{R} \cdot \begin{bmatrix} l_h^{\text{hori}}/2 \\ -l_h^{\text{vert}}/2 \\ f \end{bmatrix}, \tag{4.42}$$

with $\boldsymbol{R}$ being the overall rotation matrix as is defined in Equation (4.13). The auxiliary measure h is introduced to define the permitted height of an arbitrary image point $\boldsymbol{p}_i \in \boldsymbol{I}_h$ to be in $\mathbf{\Pi}$:

$$h\left(\boldsymbol{p}_i, fov_c^{\text{hori}}\right) = |\boldsymbol{p}_i| \cdot \cos\left(\frac{fov_c^{\text{hori}}}{2}\right). \tag{4.43}$$

Figure 4.5 shows the geometric derivation of h. An image point $\boldsymbol{p}_i$ is classified to be in $\mathbf{\Pi}$ if the following condition holds:

$$\mathbf{\Pi} \cup \{\boldsymbol{p}_i\}\ \forall \boldsymbol{p}_i \in \boldsymbol{I}_h \iff p_{i,z} \geq h\left(\boldsymbol{p}_i, fov_c^{\text{hori}}\right). \tag{4.44}$$

Equation (4.44) is used as the condition to identify the elements of $\mathbf{\Pi}_E$. If all four vertices $\boldsymbol{p}_m$ meet this condition, the compensation rate is equal to $c_{ptr} = 1$. If none of the vertices fulfill this condition, delay compensation is not feasible as $c_{ptr} = 0$. The compensation rate lies for all other cases between $]0, 1[$ and the points on the curved line of the overlapping area need to be computed, which can be seen in Figure 4.5. The computation time is reduced by approximating this curved line by sampling it to 40 points. These 40 points are determined by constructing 40 lines g (20 horizontal, 20 vertical) as depicted in Figure 4.5 (b). These lines are uniformly distributed and are parallel to the edges of the HMD image plane. Two opposite points $\boldsymbol{q}_i, \boldsymbol{q}_j$, which are located on opposite edges of the HMD image plane, are used to compute these lines according to:

$$g : \boldsymbol{x}_{ij} = \boldsymbol{q}_j + \lambda(\boldsymbol{q}_j - \boldsymbol{q}_i). \tag{4.45}$$

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

$$\boldsymbol{q}_i = \boldsymbol{p}_1 + \mu \cdot (\boldsymbol{p}_2 - \boldsymbol{p}_1), \qquad \boldsymbol{q}_j = \boldsymbol{p}_4 + \mu \cdot (\boldsymbol{p}_3 - \boldsymbol{p}_4). \tag{4.46}$$

The vertical lines g^{vert} are retrieved by:

$$\boldsymbol{q}_i = \boldsymbol{p}_2 + \mu \cdot (\boldsymbol{p}_3 - \boldsymbol{p}_2), \qquad \boldsymbol{q}_j = \boldsymbol{p}_1 + \mu \cdot (\boldsymbol{p}_4 - \boldsymbol{p}_1), \tag{4.47}$$

where μ iterates from 0 to 1. The step size is a design parameter and can be changed upon request. Empirical experiments showed that a step size of 20 steps is more than sufficient. The lines g are designed to quantify the points on the curved edge. These edge points are the last pixels accessible for display. The $z-$values of these points have to equal h as is demanded in Equation (4.44). It is thus needed to find points $\boldsymbol{x}_{ij}$ on each line g that satisfy the following condition:

$$\begin{aligned} x_{ij,z} &\stackrel{!}{=} h(\boldsymbol{x}_{ij}, fov_c^{\text{hori}}), \\ q_{i,z} + \lambda(q_{j,z} - q_{i,z}) &\stackrel{!}{=} h\Big(\boldsymbol{q}_i + \lambda(\boldsymbol{q}_j - \boldsymbol{q}_i), fov_c^{\text{hori}}\Big). \end{aligned} \tag{4.48}$$

Equation (4.48) is solved for λ through the Newton-Raphson method. Inserting λ into Equation (4.45) 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 $\boldsymbol{\Pi}_E$ and can be utilized to calculate $area(\boldsymbol{\Pi})$. The overlapping area can then be computed similarly to perspective cameras by integrating over the auxiliary curves t and b as follows:

$$area(\boldsymbol{\Pi}) = \int_{x_{min}}^{x_{max}} (t - b)\, \mathrm{d}x. \tag{4.49}$$

To ease the calculation, the overlapping area is transformed into the center of the XY-plane as depicted in Figure 4.5 (b). The curves t and b with the transformed edge and corner points in the set $\boldsymbol{\Pi}_E$ are specified with:

$$\boldsymbol{\Pi}_E^{xy} = \boldsymbol{R}^{\mathrm{T}} \cdot (\boldsymbol{\Pi}_E - \boldsymbol{v}), \tag{4.50}$$

where the inverse of the rotation matrix $\boldsymbol{R}^{-1} = \boldsymbol{R}^{\mathrm{T}}$ is simply its transpose, and $\boldsymbol{v}$ is the current viewing direction of the user, as is introduced in Equation (4.16).

The bounds of integration needed to solve Equation (4.49) can be computed as:

$$\begin{aligned} x_{\text{min}} &= \{x \in \mathbb{R} \,|\, \min\{p_{i,x}\}, \forall \boldsymbol{p}_i = (p_{i,x}, p_{i,y})^T \in \boldsymbol{\Pi}_E^{xy}\}, \\ x_{\text{max}} &= \{x \in \mathbb{R} \,|\, \max\{p_{i,x}\}, \forall \boldsymbol{p}_i = (p_{i,x}, p_{i,y})^T \in \boldsymbol{\Pi}_E^{xy}\}. \end{aligned} \tag{4.51}$$

The curves t and b are deduced from the points in $\boldsymbol{\Pi}_E^{xy}$, which are assigned 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 to both. Figure 4.5 aims to visualize the allocation process. The algorithmic loop applied to assign the points in $\boldsymbol{\Pi}_E^{xy}$ to the sets T and B is presented as pseudocode in Algorithm 1, where T, B, X_{min} and X_{max} are defined as sets of points. The sets T and B

Algorithm 1: Assignment of the points from set $\mathbf{\Pi}_E^{xy}$ to the top curve set T and/or the bottom curve set B

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

are then utilized to specify the curves t and b. This is done by computing the lines m between neighboring points $\boldsymbol{p}_k$ and $\boldsymbol{p}_{k+1}$, with $\boldsymbol{p}_k, \boldsymbol{p}_{k+1} \in T$ or $\boldsymbol{p}_k, \boldsymbol{p}_{k+1} \in B$ as:

$$m = \frac{p_{k+1,y} - p_{k,y}}{p_{k+1,x} - p_{k,x}} \cdot x + \frac{p_{k,x}p_{k+1,y} - p_{k+1,x}p_{k,y}}{p_{k,x} - p_{k+1,x}}. \tag{4.52}$$

Therefore, the curves t and b consist of concatenated line segments m, which are precise enough to compute $area(\mathbf{\Pi})$ with the integral in Equation (4.49). Equation (4.38) is then revisited to compute the present degree of reachable delay-compensation.

4.4 Generic Delay-compensation

The two approaches above are specifically tailored to the camera module in use. This section presents a generic delay-compensation model that is agnostic to the deployed camera system and can be flexibly applied for both perspective and fisheye cameras. The camera's image plane $\boldsymbol{I}_c$ and the HMD's image plane $\boldsymbol{I}_h$ are therefore mapped onto a common sphere. The area $\mathbf{\Pi}$ that is mutually shared between both image planes is defined as the image content that is accessible for display as is visualized in Figure 4.4. The projected image planes are

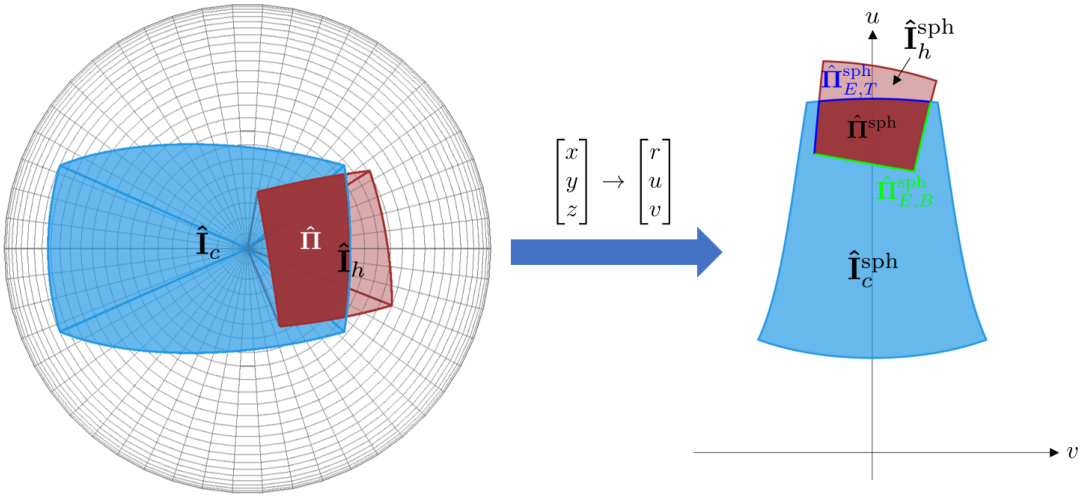


Figure 4.7: The system description is illustrated for the generic delay-compensation approach in the case of partial compensation abilities. Both the captured image data and the requested viewport are projected onto the sphere. The region-of-interest is shifted accordingly. To quantify the amount of available image content for display, the overlapping area (dark red) needs to be specified. The pixels are then converted into the spherical domain to simplify the computation of the overlapping area (adopted from [2] © 2019 IEEE) .

termed as $\hat{\boldsymbol{I}}_c$ and $\hat{\boldsymbol{I}}_h$ along with the shared (overlapping) area $\hat{\mathbf{\Pi}}$. The amount of achievable delay-compensation is then expressed in terms of the generic compensation rate c_{ptr} given by:

$$c_{ptr} = \frac{area(\hat{\mathbf{\Pi}})}{area(\hat{\boldsymbol{I}}_h)}. \tag{4.53}$$

After mapping the images onto the common sphere, the area of the rectangular HMD image plane can be computed in the spherical domain by applying the scalar surface integral according to [126]:

$$area(\hat{\boldsymbol{I}}_h) = \iint_{\hat{\boldsymbol{I}}_h} \mathrm{d}\hat{\boldsymbol{I}}_h = \int_{v_{\min}}^{v_{\max}} \int_{u_{\min}(v)}^{u_{\max}(v)} \left\| \frac{\partial \boldsymbol{\varphi}}{\partial u} \times \frac{\partial \boldsymbol{\varphi}}{\partial v} \right\| \mathrm{d}u\, \mathrm{d}v, \tag{4.54}$$

with u and v being the polar and azimuth angles, respectively. $\boldsymbol{\varphi}$ is a parameterization of spherical coordinates [126] and is derived with:

$$\boldsymbol{\varphi}(u,v) = \begin{bmatrix} r \cdot \sin(u) \cdot \cos(v) \\ r \cdot \sin(u) \cdot \sin(v) \\ r \cdot \cos(u) \end{bmatrix}. \tag{4.55}$$

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

$$\frac{\partial \boldsymbol{\varphi}}{\partial u} = \begin{bmatrix} r \cdot \cos(u) \cdot \cos(v) \\ r \cdot \cos(u) \cdot \sin(v) \\ -r \cdot \sin(u) \end{bmatrix}, \frac{\partial \boldsymbol{\varphi}}{\partial v} = \begin{bmatrix} -r \cdot \sin(u) \cdot \sin(v) \\ r \cdot \sin(u) \cdot \cos(v) \\ 0 \end{bmatrix}. \tag{4.56}$$

Applying the cross product to these derivatives and taking the Euclidean norm results in:

$$\left\| \frac{\partial \boldsymbol{\varphi}}{\partial u} \times \frac{\partial \boldsymbol{\varphi}}{\partial v} \right\| = r^2 \sin(u). \tag{4.57}$$

By inserting this result into Equation (4.54), the final equation for the HMD's image plane area can be obtained as:

$$area(\hat{\boldsymbol{I}}_h) = \int_{v_{\min}}^{v_{\max}} \int_{u_{\min}(v)}^{u_{\max}(v)} r_{\text{sph}}^2 \cdot \sin(u)\, \mathrm{d}u \mathrm{d}v. \tag{4.58}$$

To compute the bounds of integration of the integral in spherical coordinates, the corner points of the HMD image plane are leveraged as derived in Equations (4.41) and (4.42), as well as all points on the edges of the HMD image plane $\boldsymbol{p}_i \in \boldsymbol{I}_{h,E}$ between these corner points. These points are mapped onto a sphere with radius r_{sph}:

$$\hat{\boldsymbol{p}}_i = \frac{\boldsymbol{p}_i}{|\boldsymbol{p}_i|} \cdot r_{\text{sph}}. \tag{4.59}$$

The points $\hat{\boldsymbol{p}}_i = (\hat{p}_{i,x}, \hat{p}_{i,y}, \hat{p}_{i,z})^T \in \hat{\boldsymbol{I}}_{h,E}$ are first transformed from cartesian to spherical coordinates $\hat{\boldsymbol{p}}_i^{\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,v}^{\text{sph}})^T \in \hat{\boldsymbol{I}}_{h,E}^{\text{sph}}$ [126]:

$$\hat{p}_{i,u}^{\text{sph}} = \arccos\left(\frac{\hat{p}_{i,z}}{\sqrt{\hat{p}_{i,x}^2 + \hat{p}_{i,y}^2 + \hat{p}_{i,z}^2}} \right), \tag{4.60}$$

$$\hat{p}_{i,v}^{\text{sph}} = \arctan\left(\frac{\hat{p}_{i,y}}{\hat{p}_{i,x}} \right). \tag{4.61}$$

The radius $\hat{p}_{i,r}^{\text{sph}}$ is identical to r_{sph} ($\hat{p}_{i,r}^{\text{sph}} = r_{\text{sph}}$) as the points $\boldsymbol{p}_i \in \boldsymbol{I}_{h,E}$ are normalized to the radius r_{sph} as per Equation (4.59). The respective transformation of the HMD image plane into the spherical domain is visualized in Figure 4.7.

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

$$\begin{aligned} v_{\min} &= \{v \in \mathbb{R} \mid \min\{\hat{p}_{i,v}^{\text{sph}}\}, \forall \hat{\boldsymbol{p}}_i^{\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,v}^{\text{sph}})^T \in \hat{\boldsymbol{I}}_{h,E}^{\text{sph}}\}, \\ v_{\max} &= \{v \in \mathbb{R} \mid \max\{\hat{p}_{i,v}^{\text{sph}}\}, \forall \hat{\boldsymbol{p}}_i^{\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,v}^{\text{sph}})^T \in \hat{\boldsymbol{I}}_{h,E}^{\text{sph}}\}. \end{aligned} \tag{4.62}$$

Note that the bounds of integration $u_{\min}(v)$ and $u_{\max}(v)$ depend on the integration variable v as described in Equation (4.58), and hence need special attention. For this reason, the points in $\hat{\boldsymbol{I}}_{h,E}^{\text{sph}}$ are assigned to either the top line $\hat{\boldsymbol{I}}_{h,T}^{\text{sph}}$ or the bottom line $\hat{\boldsymbol{I}}_{h,B}^{\text{sph}}$ of the overlapping area and sorted by their x-values as depicted in Figure 4.7. 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:

$$\begin{aligned} u_{\min}(v) = \{u \in \mathbb{R} \mid \hat{p}_{i,v}^{\text{sph}} = v, \\ \forall \hat{\boldsymbol{p}}_i^{\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,v}^{\text{sph}})^T \in \hat{\boldsymbol{I}}_{h,B}^{\text{sph}} \subset \hat{\boldsymbol{I}}_h^{\text{sph}}\}, \\ u_{\max}(v) = \{u \in \mathbb{R} \mid \hat{p}_{i,v}^{\text{sph}} = v, \\ \forall \hat{\boldsymbol{p}}_i^{\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,v}^{\text{sph}})^T \in \hat{\boldsymbol{I}}_{h,T}^{\text{sph}} \subset \hat{\boldsymbol{I}}_h^{\text{sph}}\}. \end{aligned} \tag{4.63}$$

These bounds are then used to compute $area(\hat{\boldsymbol{I}}_h)$ with respect to Equation (4.58). The calculation of the overlapping $area(\hat{\boldsymbol{I}}_h)$ resembles conceptually Equation (4.58) except for the bounds of integration:

$$area(\hat{\boldsymbol{\Pi}}) = \int_{v_{\text{min}}^{\hat{\Pi}}}^{v_{\text{max}}^{\hat{\Pi}}} \int_{u_{\text{min}}^{\hat{\Pi}}(v)}^{u_{\text{max}}^{\hat{\Pi}}(v)} r_{\text{sph}}^2 \cdot \sin(u)\, \mathrm{d}u\, \mathrm{d}v. \tag{4.64}$$

To compute these integration bounds, the edges of the overlapping area need to be specified. 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 analogous to the HMD's image plane. The edges of the camera image plane are, therefore, determined as described in Section 4.2, except that the camera's fov_c is used instead of the HMD's. In a subsequent step, these points are mapped onto the sphere and transformed into the spherical domain as presented in Equations (4.59) - (4.61), which can be also seen in Figure 4.7.

In the case of equidistant fisheye cameras, the image plane is already spherical. The radius of the sphere, where the image planes are mapped onto, needs to be adapted. It is set to be identical to the focal length of the fisheye camera $r_{\text{sph}} = f$. 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 $\boldsymbol{p}_i \in \hat{\boldsymbol{I}}_{c,E}$ are derived as follows:

$$\boldsymbol{p}_i = \boldsymbol{R} \cdot \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} f \cdot \sin(fov_c^{\text{hori}}/2) \\ 0 \\ f \cdot \cos(fov_c^{\text{hori}}/2) \end{bmatrix}, \quad \forall \alpha \in [0, 2\pi). \tag{4.65}$$

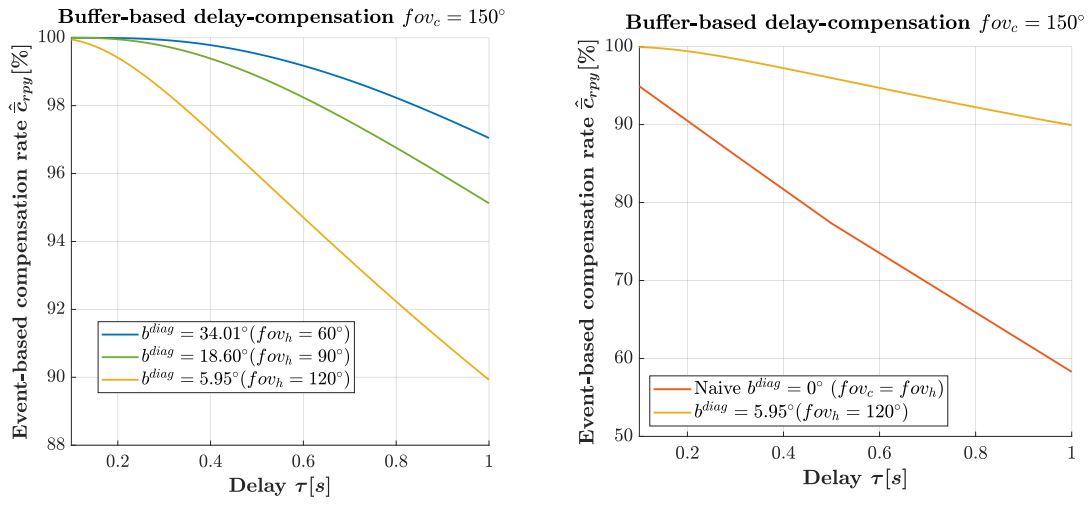
The points $\hat{\boldsymbol{I}}_{c,E}$ of the fisheye camera are transformed to spherical coordinates $\hat{\boldsymbol{I}}_{c,E}^{\text{sph}}$ with Equations (4.60) and (4.61).

The edges of the camera's and HMD's image plane $\hat{\boldsymbol{I}}_{h,E}^{\text{sph}}$ are leveraged to specify the inner edges of their mutual area $\hat{\boldsymbol{\Pi}}_E^{\text{sph}}$. In general, there are three cases which can occur: If the HMD image plane location is entirely within the camera image plane, the compensation rate is considered to be 1. If it lies completely outside of the camera image plane, the compensation rate is set to 0. In all other cases, $area(\hat{\boldsymbol{\Pi}})$ needs to be computed by means of Equation (4.64). The respective bounds of integration are calculated using $\hat{\boldsymbol{\Pi}}_{E,T}^{\text{sph}}$, which corresponds to the set of points on the top line, and $\hat{\boldsymbol{\Pi}}_{E,B}^{\text{sph}}$, which contains the set of points on the bottom line of the edges $\hat{\boldsymbol{\Pi}}_E^{\text{sph}}$ of the overlapping area as illustrated in Figure 4.7:

$$\begin{aligned} v_{\text{min}}^{\hat{\Pi}} &= \{v \in \mathbb{R} \mid \min\{\hat{p}_{i,v}^{\text{sph}}\}, \forall \hat{\boldsymbol{p}}_i{}^{\text{sph}} \in \hat{\boldsymbol{\Pi}}_E^{\text{sph}}\}, \\ v_{\text{max}}^{\hat{\Pi}} &= \{v \in \mathbb{R} \mid \max\{\hat{p}_{i,v}^{\text{sph}}\}, \forall \hat{\boldsymbol{p}}_i{}^{\text{sph}} \in \hat{\boldsymbol{\Pi}}_E^{\text{sph}}\}, \\ u_{\text{min}}^{\hat{\Pi}}(v) &= \{u \in \mathbb{R} \mid \hat{p}_{i,v}^{\text{sph}} = v, \forall \hat{\boldsymbol{p}}_i{}^{\text{sph}} \in \hat{\boldsymbol{\Pi}}_{E,B}^{\text{sph}}\}, \\ u_{\text{max}}^{\hat{\Pi}}(v) &= \{u \in \mathbb{R} \mid \hat{p}_{i,v}^{\text{sph}} = v, \forall \hat{\boldsymbol{p}}_i{}^{\text{sph}} \in \hat{\boldsymbol{\Pi}}_{E,T}^{\text{sph}}\}, \end{aligned} \tag{4.66}$$

where $\hat{\boldsymbol{p}}_i{}^{\text{sph}} = (\hat{p}_{i,r}^{\text{sph}}, \hat{p}_{i,u}^{\text{sph}}, \hat{p}_{i,v}^{\text{sph}})^{\text{T}}$. After quantifying the areas of $\hat{\boldsymbol{\Pi}}$ and $\hat{\boldsymbol{I}}_{\text{h}}$, the computation of the generic compensation rate can be deduced from Equation (4.53).

4.5 Results



(a) (b)

Figure 4.8: (a) An equidistant fisheye camera is used to capture a large field-of-view of $fov_c = 150°$. The respective view size of the user is varied according to $fov_h = \{60°, 90°, 120°\}$. The diagonal buffer is used as a qualitative measure to convey the available amount of extra image content. The results confirm that higher buffer sizes result in advanced delay-compensation abilities. This tendency can be seen through their respective mean compensation rates that were computed by means of the IMT dataset. (b) The buffer-based delay-compensation exhibits substantial improvements compared to the naive version where no compensation is applied at all; even for buffer sizes being as small as 5.95°.

The generic compensation rate is a valid metric to convey the achievable level of delay-compensation. The delay-compensation ability, however, is highly dependent on the amount of buffer, and hence the available set of image pixels that are ready to be displayed for a requested viewing direction. Figure 4.8 (a) demonstrates the influence of the buffer size on the degree of delay-compensation. The IMT dataset is used to plot the (event-based) mean delay-compensation rate for investigated delays between 0.1 s to 1 s. The diagonal buffer portion is used as a measure to quantify the available amount of extra image content, as is debated in Chapter 3. The high-resolution video content is assumed to be recorded with an equidistant fisheye camera that has a visual field of $fov_c = 150°$. The (horizontal) field-of-view of the HMD is changed in discrete steps from 60 ° to 120 ° (step size 30°). Figure 4.8 confirms the expected behavior, where larger buffer sizes achieve higher levels of delay-compensation. Figure 4.8 (b) is additionally plotted to compare the buffer-based delay-compensation performance with the naive version, where no compensation is applied at all. It is remarkable to see that already a little buffer of 5.95° shows great improvements compared to the naive method. Even for a delay of 1 s, mean compensation rates of around 90 % are reachable compared to the 57 % of the naive method.

4.6 Chapter Summary

This chapter introduced the concept behind the delay-compensation vision system, which aims to compensate the perceived latency between ego-motion and visual response. To avoid the emergence of motion sickness it is indispensable that the sensory information is in accordance with the expected visual impressions. The buffer-based delay-compensation approach is proposed as a technique to facilitate instantaneous visual feedback by providing pre-captured visual information. A greater visual field is captured than is actually needed for the user's viewport. The residual footage can be accessed immediately when the user's head is rotated. The amount of buffer is thereby decisive about the degree of achievable delay-compensation. Different buffer sizes are investigated to confirm this hypothesis. The compensation rate is proposed as a generic, device-agnostic metric to convey the achievable level of delay-compensation. Results verify that a significant increase in delay-compensation is obtained when applying the buffer-based delay-compensation approach compared to the naive case where no compensation is adopted at all.

Chapter 5

Dynamic Field-of-view Adaptation

The theory behind the delay-compensating vision system is introduced in Chapter 4. Its conceptional modes of operation are illustrated in Figure 4.4 (a). Wide-angle or equidistant fisheye cameras are used to capture a larger field-of-view than is displayed to the user. By using only a subregion of the camera's visual field, the residual imagery can be leveraged for local delay-compensation until the updated frame acquired at the present head position arrives. In this chapter, the delay-compensation paradigm is extended by a dynamic, velocity-based field-of-view adaptation technique. The objective is to improve the achievable degree of delay compensation, which is particularly sensitive to fast head rotations, without losing the feeling of presence.

Parts of the work presented in this chapter have been published in [1].

5.1 The Impact of Changing the User's Field-of-view

Previous work discovered a strong correlation between the sensed degree of presence and the emergence of motion sickness [127], [128] as described in the following. The selection process of a proper visual field is thereby a critical factor. Enlarged visual fields proved to enhance the perceived feeling of presence [129], but also stimulate more substantial portions of the peripheral vision. The peripheral vision is known to be particularly sensitive to stimuli induced by fast motion and thus tends to be the reason for the onset of motion sickness [130]. Lessening the field-of-view, instead, infers the decline of motion sickness, but simultaneously downgrades the feeling of presence [131]. The objective of the work presented in this chapter is to maintain the degree of presence while decreasing the visually induced motion sickness. A dynamic field-of-view adaptation technique is presented that temporarily constrains the visual field with respect to the current head-motion velocity. Circular and asynchronous rectangular field-of-view restriction policies are proposed. Minor research has been dedicated before to investigate the visual field's influence on the user's sensation. Prior art either reduced the field-of-view throughout the whole streaming exposure or changed it in isolated threads [132]. Comparable research that examined field-of-view adaptation is presented in [132], [133]. Fernandes *et al.* [132] changed the visual field with respect to translatory speed of a joypad controller, which is mainly useful for screen-based applications. The work presented in this chapter, however, exploits the filtered data values from the

Figure 5.1: Overview of different field-of-view limitation policies compared to the original one as a reference. Both the circular and asynchronous restriction strategies are presented as a snapshot for a fast motion with a high $\mathcal{R}_{\text{max}}$ for the sake of clarity. The remaining parts of the visual field are either filled with black pixels or artificially extrapolated color values, which are less discernible during fast motions. A fading layer is superimposed to smooth the restriction boundaries.

orientation sensor that is embedded in the HMD to adapt the user's visual field more naturally. Two velocity-based circular and (asynchronous) rectangular adaptation policies are proposed. Kala *et al.* [133], in contrast, leveraged scene information to modify the user's line of vision [133]. Their technique is particularly interesting for virtual environments, where simulator sickness might emerge through vection. This work, however, targets real telepresence applications, where the onset of vection is not very likely. The objective is to reduce the visually induced motion sickness that is triggered by the temporal lag between ego-motion and visual feedback.

5.2 Velocity-based Field-of-view Adaptation

The adaptive field-of-view modification technique, which is a function of the to current head-motion velocity, aims to lessen the emergence of motion sickness during quick head motions. The level of compensation is in this way noticeably improved, particularly for rapid direction changes and fast head-motions. A circular and an asynchronous rectangular field-of-view adaptation policy are introduced that are both dependent on the directed head-motion velocity. The peripherals of the constrained viewport portions are either filled with black or artificially colored pixels, which are generated by extrapolating the outermost color values of the visual field as is demonstrated in Figure 5.1. A subsequent fading layer is superimposed to smooth the transitions. The resulting compensation rate $\widetilde{c}_{ptr}$, conveying the reachable degree of delay-compensation, can be generalized and reformulated to:

$$c_{ptr} = \frac{area(\mathbf{\Pi})}{l_h^{\text{hori}} \cdot l_h^{\text{vert}}} \quad \Rightarrow \quad \widetilde{c}_{ptr} = \frac{area(\mathbf{\Pi})}{area(\boldsymbol{I}_h)}, \tag{5.1}$$

with:

$$l_h^{\text{hori}} = 2 \cdot f \cdot \tan\left(\frac{fov_h^{\text{hori}}}{2}\right), \tag{5.2}$$

$$l_h^{\text{vert}} = 2 \cdot f \cdot \tan\left(\frac{fov_h^{\text{vert}}}{2}\right), \tag{5.3}$$

which depend on the displayed horizontal and vertical field-of-views $\left(fov_h^{\text{hori}}, fov_h^{\text{vert}}\right)$ and refer to the width and height of the original image plane of the HMD. $\mathbf{\Pi}$ represents the set of all image points $\boldsymbol{p}_i \in \mathbf{\Pi}$ that are available to be displayed.

As the image plane is temporarily restricted, $area(\boldsymbol{I}_h)$ is used to denote the dynamically changing area of the HMD's image plane that is to be considered for the present head velocity. The auxiliary measure h, which is introduced in Equation (4.43), is utilized to determine the permitted height of an arbitrary image point $\boldsymbol{p}_i \in \boldsymbol{I}_h$ to be in $\mathbf{\Pi}$. During quick head rotations the amount of image points $\boldsymbol{p}_i$ that need to be taken into account gets confined. $\dot{\boldsymbol{\xi}}_{h,\text{th}}$ [rad/s] is denoted as the threshold rotation velocity that needs to be exceeded to initiate the field-of-view adaptation. $\dot{\boldsymbol{\xi}}_{h,\text{max}}$ [rad/s] terms the maximum rotation speed after which the maximum restriction $\mathcal{R}_{max} \in [0, 1]$ is reached. $\mathcal{R}_{max} = 0.2$, for instance, constitutes a maximum viewport restriction of 20 %; that is that at least 80 % of the user's viewport contains visual information even when the head is moved very fast. The amount of restriction $\mathcal{R}(\dot{\boldsymbol{\xi}}_h)$ as a function of the current head velocity $\dot{\boldsymbol{\xi}}_h$ is expressed as:

$$\mathcal{R}(\dot{\boldsymbol{\xi}}_h) = \mathcal{R}_{\text{max}} \cdot \min\left\{1, \max\left\{0, f_{\{\text{exp, lin}\}}(\dot{\boldsymbol{\xi}}_h)\right\}\right\}, \tag{5.4}$$

where $f_{\{\text{exp, lin}\}}(\dot{\boldsymbol{\xi}}_h)$ represents a linear or exponential adaptation behavior, which are both visualized in Figures 5.2 and 5.3. The linear and exponential adaptation functions are mathematically described as follows:

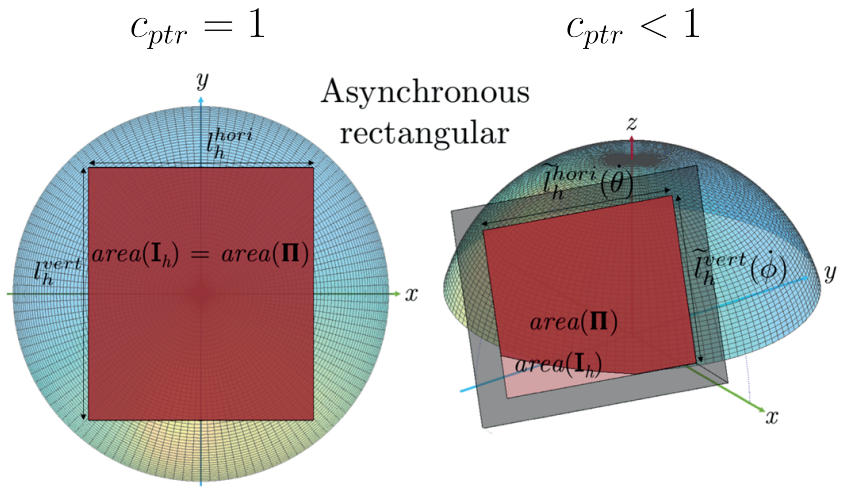
$$f_{\text{lin}}(\dot{\boldsymbol{\xi}}_h) = \frac{\left|\dot{\boldsymbol{\xi}}_h\right| - \dot{\boldsymbol{\xi}}_{h,\text{th}}}{\dot{\boldsymbol{\xi}}_{h,\text{max}} - \dot{\boldsymbol{\xi}}_{h,\text{th}}}, \tag{5.5}$$

$$f_{\text{exp}}(\dot{\boldsymbol{\xi}}_h) = \frac{\mathrm{e}^{\dot{\boldsymbol{\xi}}_h^2} - \mathrm{e}^{\dot{\boldsymbol{\xi}}_{h,\text{th}}^2}}{\mathrm{e}^{\dot{\boldsymbol{\xi}}_{h,\text{max}}^2} - \mathrm{e}^{\dot{\boldsymbol{\xi}}_{h,\text{th}}^2}}. \tag{5.6}$$

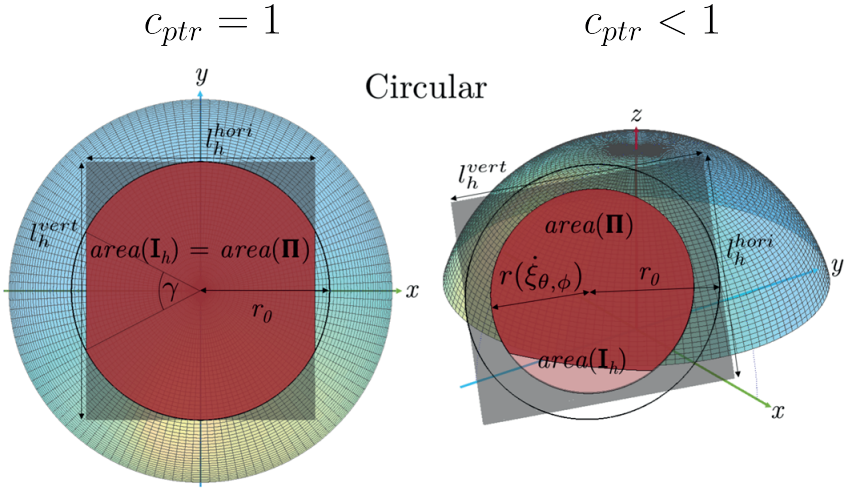
After determining the amount of restriction, it is essential to define the velocity-dependent geometric design of the visual field. An asynchronous rectangular and a circular restriction policy are presented, which are explained in the following in more detail.

5.2.1 Asynchronous Rectangular Restriction

The asynchronous rectangular restriction technique constrains the visual field independently in horizontal and vertical direction. The directional degree of restriction is thereby highly correlated to the angular pan and tilt velocities $(\dot{\theta}, \dot{\phi})$. Constraining the roll rotation leads to visually discomforting effects and is hence neglected for the field-of-view adaptation. The



(a)

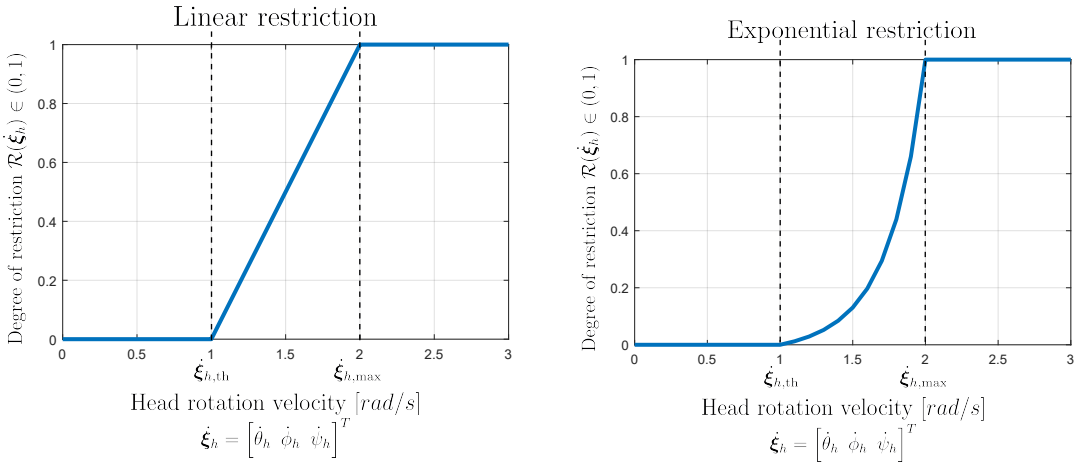


(b)

Figure 5.2: The illustrated schemes show the operating principles of the two presented velocity-based field-of-view adaptation techniques. The optimal case (full delay-compensation) is juxtaposed the case, where only partial delay-compensation is achieved. (a) Asynchronous rectangular restriction policy. The vertical and horizontal visual fields are treated decoupled from each other and are individually adapted with regard to the present pan and tilt rotation speed, respectively. (b) Circular restriction method. The HMD's viewport size is constrained to a circular region with respect to the current angular pan and tilt velocity. The amount of field-of-view restriction is subject to the joint pan and tilt velocity $\dot{\xi}_{\theta,\phi}$, respectively (reproduced from [1] © 2018 IEEE).

area of the HMD's adaptive image plane is therefore expressed as a function of the present angular pan and tilt velocities:

$$area(\boldsymbol{I}_h) = \widetilde{l}_h^{\text{hori}}(\dot{\theta}) \cdot \widetilde{l}_h^{\text{vert}}(\dot{\phi}). \tag{5.7}$$



(a) (b)

Figure 5.3: Overview of the two presented velocity dependencies for the dynamic field-of-view adaptation technique. (a) Linear dependency between the head rotation velocity and the amount of visual limitation. The visual field is not constrained until a threshold value $\dot{\boldsymbol{\xi}}_{h,\text{th}}$ is exceeded and takes effect up to a maximum value of $\dot{\boldsymbol{\xi}}_{h,\text{max}}$. (b) The exponential velocity-dependency can be described analogously. The threshold value helps to be resistant to noise. Besides that, visual deviations within the field-of-view are easier detected by the users when the head motion is slow compared to fast motions. The maximum value is important not to lose the visual field entirely.

The length and width of the HMD's image plane change dynamically and are subject to the amount of restriction:

$$\widetilde{l}_h^{\text{hori}}(\dot{\theta}) = (1 - \mathcal{R}(\dot{\theta})) \cdot l_h^{\text{hori}}, \tag{5.8}$$

$$\widetilde{l}_h^{\text{vert}}(\dot{\phi}) = (1 - \mathcal{R}(\dot{\phi})) \cdot l_h^{\text{vert}}. \tag{5.9}$$

The modified set of image points $\widetilde{\boldsymbol{p}}_i$ that is utilized to calculate the set of pixels $\boldsymbol{\Pi}$ that are now available to be displayed results to:

$$\widetilde{\boldsymbol{p}}_i = \boldsymbol{R} \cdot \left[\pm \frac{\widetilde{l}_h^{\text{hori}}(\dot{\theta})}{2} \;\; \pm \frac{\widetilde{l}_h^{\text{vert}}(\dot{\phi})}{2} \;\; f \right]^T, \tag{5.10}$$

with $\boldsymbol{R}$ being the rotation matrix that describes the current head orientation.

5.2.2 Circular Restriction

The circular restriction technique takes the joint angular velocity of pan and tilt rotations into account:

$$\dot{\xi}_{\theta,\phi} = \sqrt{\dot{\theta}^2 + \dot{\phi}^2}. \tag{5.11}$$

The dynamically changing area of the HMD's image plane thereby amounts to:

$$area(\boldsymbol{I}_h) = r(\dot{\xi}_{\theta,\phi})^2 \cdot (\pi - \gamma + \sin(\gamma)), \tag{5.12}$$

where γ is the angle of the excluded circular segments as is shown in Figure 5.2 (b). $r(\dot{\xi}_{\theta,\phi})$ alters with respect to the current joint head motion velocity:

$$r(\dot{\xi}_{\theta,\phi}) = (1 - \mathcal{R}(\dot{\xi}_{\theta,\phi})) \cdot r_0. \tag{5.13}$$

r_0 corresponds to the original size of the circle without any adaptation and is termed as:

$$r_0 = \frac{fov_h^{\text{vert}}}{2}. \tag{5.14}$$

$area(\mathbf{\Pi})$ can then be computed by considering the constrained set of available image points after making a radial round analysis $\forall \alpha \in [0, 2\pi)$:

$$\widetilde{\boldsymbol{p}}_i = \boldsymbol{R} \cdot \begin{bmatrix} r(\dot{\xi}_{\theta,\phi}) \cos(\alpha) & r(\dot{\xi}_{\theta,\phi}) \sin(\alpha) & f \end{bmatrix}^T, \tag{5.15}$$

with:

$$\left| r(\dot{\xi}_{\theta,\phi}) \cos(\alpha) \right| \leq \frac{l_h^{\text{hori}}}{2} \quad \wedge \quad \left| r(\dot{\xi}_{\theta,\phi}) \sin(\alpha) \right| \leq \frac{l_h^{\text{vert}}}{2}. \tag{5.16}$$

The image points $\boldsymbol{p}_i$ that are taken into account for the dynamic field-of-view adaptation are modified to $\widetilde{\boldsymbol{p}}_i$. Figure 5.2 (b) illustrates the constrained set of image points that need to be considered. The accessible amount of image content $area(\mathbf{\Pi})$ that can be leveraged for display can then be deduced in analogy to Chapter 4.

The merit of the adaptive field-of-view modification is conditional on the selected values for $\dot{\boldsymbol{\xi}}_{h,\text{th}}$, $\dot{\boldsymbol{\xi}}_{h,\text{max}}$, and $\mathcal{R}_{\text{max}}$. These values are etiological design parameters that have significant implications on the achievable level of delay-compensation and the degree of experienced visual comfort. They can be changed depending on the target application or the task at hand. The objective of this work is to provide high level of delay-compensation and visual comfort simultaneously. A pilot pre-study is therefore conducted to find suitable trade-off parameters. For that group of participants, the following value selections yielded the best results:

$$\dot{\boldsymbol{\xi}}_{h,\text{th}} = \begin{bmatrix} 1\,\text{rad/s} & 1\,\text{rad/s} & - \end{bmatrix}^T, \tag{5.17}$$

$$\dot{\boldsymbol{\xi}}_{h,\text{max}} = \begin{bmatrix} 2\,\text{rad/s} & 2\,\text{rad/s} & - \end{bmatrix}^T, \tag{5.18}$$

$$\mathcal{R}_{\text{max}} = \{0.2, 0.4\}. \tag{5.19}$$

Consequently, these parameters are used for experimental validation.

5.3 Results

Subjective experiments were performed to assess the virtue and efficacy of the proposed approach in terms of the perceived level of presence, simulator sickness, and the overall opinion score. All potential subjects first underwent a screening procedure to remove outliers. $N = 13$ healthy participants (female $= 31\,\%$, male $= 69\,\%$, average age $= 25.38$, sd $= 5.18$) with normal or corrected-to-normal visual acuity were then selected to join the pilot study. To be able to provide reproducible and conclusive outcomes, a stereoscopic $360°$ video footage was employed that was kindly provided by the Fraunhofer HHI [122]. Each subject was exposed to the virtual scene six times for 45 s. The adaptation policy was changed at each iteration pursuant to Figures 5.1, 5.2, and 5.3. The participants were asked to find artificially

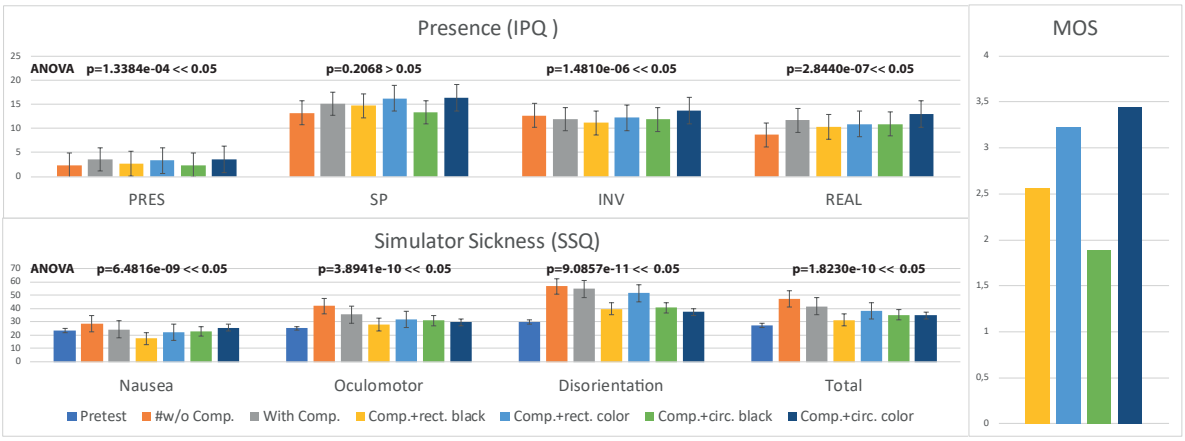


Figure 5.4: Subjective experiments to assess the degree of presence, simulator sickness and the overall opinion (MOS) with $\dot{\boldsymbol{\xi}}_{h,\text{th}} = [1\,\text{rad/s}\ \ 1\,\text{rad/s}\ \ -]^T$, $\dot{\boldsymbol{\xi}}_{h,\text{max}} = [2\,\text{rad/s}\ \ 2\,\text{rad/s}\ \ -]^T$. Compared to the naive approach, where no compensation is applied, the realization of the delay-compensation approach results in a significant improvement. The dynamic field-of-view adaptation reduces motion sickness while maintaining or, in some cases, even improving the feeling of presence (adopted from [1] © 2018 IEEE).

generated objects, whose position, size, and color were changed randomly, to implicitly foster dynamic head-motions and to avoid passiveness. After each session, the subjects were asked to fill questionnaires to examine the experienced degree of presence and simulator sickness, as well as to rank their overall opinion.

The Igroup Presence Questionnaire (IPQ) [134] was employed to quantify the perceived degree of presence. The IPQ is a very reliable self-report questionnaire (Cronbach's α = 0.87) that is composed of 14 items in total and based on a seven-point Likert scale (0-6). It determines three different subscales of presence – Spatial Presence (SP) , Involvement (INV) , Experienced Realism (REAL)) – and one additional item (the general "sense of being there" (PRES)) [134].

The Simulator Sickness Questionnaire (SSQ) [135] is deployed to measure the symptoms of simulator sickness. The SSQ is an established measure that comprises 16 items rated on a four-point Likert scale (0-3) and assesses the symptoms of three primary subscales: nausea, oculomotor, and disorientation. A total score is computed to quantify the aggregated severity.

In the end, the participants were further asked to rate their general opinion about the adopted field-of-view adaptation technique. To this end, the participants were asked to rate the field-of-view reduction compared to the reference field-of-view utilizing the Mean Opinion Score (MOS). The Absolute Category Rating scale was incorporated to map the ratings from *Bad* to *Excellent* to scalar values between 1 and 5. The final MOS value is a single rational number and is calculated as the arithmetic mean over all individual ratings.

The results from the subjective experiments by means of the applied questionnaires are illustrated in Figure 5.4. The participants perceived the adoption of the delay-compensation approach, in general, as more pleasant compared to the naive method, where no compensation is employed at all. This tendency is further confirmed by the mean IPQ and SSQ out-

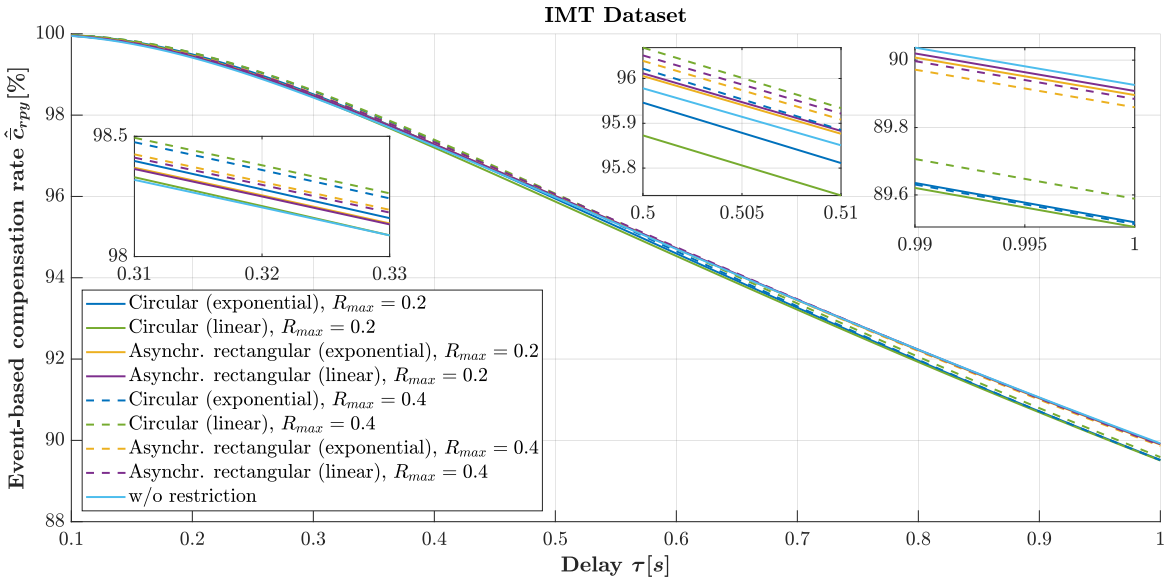


Figure 5.5: The performance of velocity-based linear and exponential dependency models are validated for the circular and the asynchronous rectangular restriction technique. The maximum restriction was varied from $\mathcal{R}_{\text{max}} = 0.2$ to $\mathcal{R}_{\text{max}} = 0.4$ to investigate their impact on the degree of delay-compensation. The mean compensation rate is used as quantitative measure to convey the achievable level of delay-compensation. A stronger field-of-view limitation leads, as was expected, to higher delay-compensation capabilities. The circular restriction technique with linear velocity-dependency is superior up to a latency of 0.5 s. Higher latencies benefit from the deployment of the rectangular restriction method.

comes, which show an increased level of presence. The results further support the claimed hypothesis that the dynamic field-of-view adaptation is capable of reducing the effect of simulator sickness while maintaining the feeling of presence. Statistical significance was verified with the ANOVA test. It is interesting to see that a field-of-view restriction technique partially leads to an even improved feeling of presence. This is particularly evident for the colorized restriction approaches. This outcome is likewise confirmed by the MOS values. The users enjoyed especially the adaptation technique, where the peripherals of the restricted viewport are filled with artificially colored pixel values.

Quantitative measures are further applied to compare the adaptation technique's degree of reachable delay-compensation to the adaption-less buffer-based delay-compensation technique that is introduced in Chapter 4. The IMT dataset is used to compute the mean compensation rates of the presented field-of-view adaptation techniques as is shown in Figure 5.5. The circular and asynchronous rectangular restriction methods are opposed to each other both for the linear and exponential velocity dependency model. The delay-compensation abilities are further investigated for maximum restriction values $\mathcal{R}_{\text{max}} = 0.2$ and $\mathcal{R}_{\text{max}} = 0.4$.

The results show that any kind of field-of-view adaptation technique is able to outperform the adaptation-less delay-compensation approach for latencies between 0 s to 0.5 s. The higher the latency becomes, the less superiority does the adaptation scheme show. The asyn-

chronous rectangular restriction design reveals better performance than the circular restriction methodologies. Figure 5.5 confirms that increasing the maximum restriction value $\mathcal{R}_{\max}$ from to 0.2 to 0.4 enhances the delay-compensation abilities as expected. However, these positive tendencies are only valid for maximum delays up to 0.5 s. Its merit for higher delays is not generalizable considering the random course of their compensation rates for latencies larger than 0.5 s. Further increasing the maximum restriction value to achieve better delay-compensation capabilities is not recommended as a too small viewport size is likely to deteriorate the feeling of presence.

5.4 Chapter Summary

The study presented in this chapter aimed to utilize the properties of the human visual system in order to achieve the following goals:

- improve the delay-compensation capability also for fast head-motions and abrupt direction changes,
- maintain or improve the feeling of presence,
- avoid or decrease the emergence of motion/simulator sickness.

These issues were approached by exploiting the properties of the human eye. Only a minor part of the eye provides high-acuity vision. The remaining portions are referred to as the peripheral vision, where the perception and discrimination of details, colors, and shapes are limited. During fast head movements, this phenomenon is even amplified. A dynamic, velocity-based field-of-view adaptation policy is presented that temporarily limits the visual field with regard to the current angular rotation speed. Asynchronous rectangular and circular restriction techniques are utilized to deploy the adaptation. The amount of restriction is thereby highly correlated to the velocity of the user's head rotation. A linear and an exponential dependency model are demonstrated and mathematically described. The presented approach does not only help to improve the delay-compensation ability but also decreased the emergence of simulator sickness while maintaining the feeling of presence. However, its positive implications on the delay-compensation abilities are only evident for delays within a range of 0 s to 0.5 s. Qualitative measures and subjective experiments are used to confirm the presented hypothesis.

Chapter 6

Semantic Viewport Prediction

The buffer-based delay-compensation technique is an effective method that works well for homogeneous and smooth motions. The achievable degree of delay-compensation is highly dependent on the system parameters such as the end-to-end delay, the camera's field-of-view, and the HMD's viewport size. In the case of rapid head motions or sudden orientation changes, it might happen that the requested viewport is outside of the buffer region. The psycho-physically motivated field-of-view adaptation technique is presented to enlarge the buffer region - and hence the delay-compensation abilities - particularly for fast head-motions. The end-to-end delay constitutes thereby the reachable level of delay-compensation. In the case of partial compensation, the unavailable image regions are either filled with black or artificially-generated color pixels, which is accomplished by extrapolating the color values of the outermost pixels of the viewport. Despite its positive impact on the delay-compensation performance, it is not possible to provide a full compensation throughout the entire telepresence experience. In order to remedy such unsatisfactory events, it is desirable to estimate the future head position with respect to the latency in question. In this way, the prospective head orientation can be sent to the server rather than the present one.

Parts of the work presented in this chapter have been published in [1], [2], [6].

6.1 Reliability of Head-motion Prediction

The head-motion profiles of all users and video sequences within the LMT dataset are exploited to investigate the validity of head-motion prediction. The time sequences of the pan, tilt, and roll orientations are used to investigate the autocorrelation function values of their delayed copies being in a range from 0 s to 2 s. Figure 6.1 shows the respective correlation values and clearly demonstrates that there is a strong correlation of the head orientations for delays less than 0.4 s. In particular, the pan orientation exhibits large correlation values of 0.7, even after 2 s. The reliability of the tilt and roll positions tends to be substantially more challenging for large delays. Figure 6.1 confirms that head-motion prediction is a logical approach to further improve the achievable level of delay-compensation.

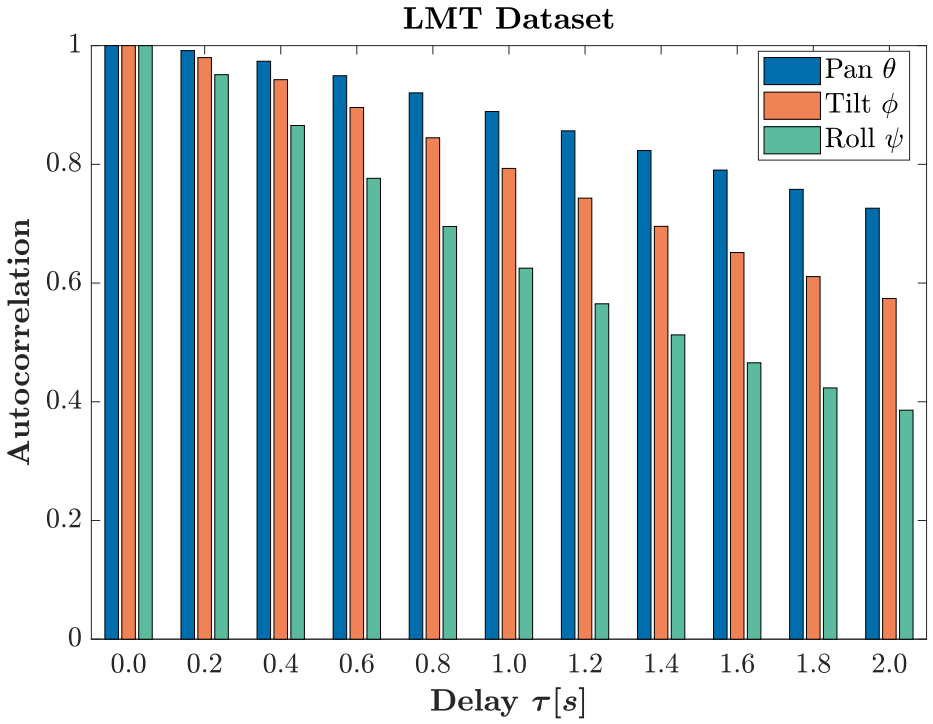


Figure 6.1: The LMT dataset is used to illustrate the autocorrelation function values of the pan, tilt, and roll rotations. The autocorrelation function values are used to present the correlation of the head motions with a delayed copy of themselves, being in a range of 0 s to 2 s. The horizontal (pan) rotations exhibit the most significant correlation values, even for high latencies.

6.2 Deterministic Prediction

Previous work mainly applied traditional methods such as linear regression or filter-based approaches. The (weighted) linear regression (LR) and the Kalman Filter-based (KF) extrapolation methods are introduced as representative state-of-the-art techniques.

6.2.1 Linear Regression (LR)

Linear regression models have two parameters (κ, υ) and estimate the data from a simple linear model. Each individual future head position (after the delay τ) can then be forecast according to:

$$\hat{\theta}_{h,\mathrm{LR}}(t+\tau) = \kappa_\theta(t) \cdot (t+\tau) + \upsilon_\theta(t), \tag{6.1}$$

$$\hat{\phi}_{h,\mathrm{LR}}(t+\tau) = \kappa_\phi(t) \cdot (t+\tau) + \upsilon_\phi(t), \tag{6.2}$$

$$\hat{\psi}_{h,\mathrm{LR}}(t+\tau) = \kappa_\psi(t) \cdot (t+\tau) + \upsilon_\psi(t). \tag{6.3}$$

The matrix-vector notation is used to concisely denote the prediction model as:

$$\hat{\boldsymbol{\xi}}_{h,\mathrm{LR}}(t+\tau) = \begin{bmatrix} \kappa_\theta(t) & \upsilon_\theta(t) \\ \kappa_\phi(t) & \upsilon_\phi(t) \\ \kappa_\psi(t) & \upsilon_\psi(t) \end{bmatrix} \cdot \begin{bmatrix} t+\tau \\ 1 \end{bmatrix} \tag{6.4}$$

$$= \begin{bmatrix} \boldsymbol{\kappa}(t) & \boldsymbol{\upsilon}(t) \end{bmatrix} \cdot \begin{bmatrix} t+\tau & 1 \end{bmatrix}^T. \tag{6.5}$$

All past orientation values within a window W are used to model the first-order polynomial function for viewport prediction. Any two data points are sufficient to solve for the values of the two parameters. Due to the random error given in the orientation values, each pair of data samples leads to a different result. The least squares estimation technique is applied to take all past data values within W into account and provide a solution that minimizes the deviation of all observed data samples. The linear regression method computes the parameters $\boldsymbol{\kappa}(t)$ and $\boldsymbol{\upsilon}(t)$ by minimizing the sum of squares of the residuals $\boldsymbol{res}(t)$:

$$\boldsymbol{res}(t) = \sum_{i=t-W}^{t} (\boldsymbol{\xi}_h(i) - \hat{\boldsymbol{\xi}}_{h,\mathrm{LR}}(i))^2 = \sum_{i=t-W}^{t} \boldsymbol{e}_i^2, \tag{6.6}$$

according to:

$$\frac{\boldsymbol{res}(t)}{\partial \boldsymbol{\kappa}} \stackrel{!}{=} \mathbf{0}, \quad \frac{\boldsymbol{res}(t)}{\partial \boldsymbol{\upsilon}} \stackrel{!}{=} \mathbf{0}, \tag{6.7}$$

where $\boldsymbol{e}_i = (\boldsymbol{\xi}_h(i) - \hat{\boldsymbol{\xi}}_{h,\mathrm{LR}}(i))$ represent the observed residuals for the ith observation. The optimal parameters that minimize the deviation of the observed data values are then obtained with [136]:

$$\boldsymbol{\kappa}(t) = \frac{\sum_{i=t-W}^{t} (i-\bar{t}) \cdot (\boldsymbol{\xi}_h(i) - \bar{\boldsymbol{\xi}}_h(t))}{\sum_{i=t-W}^{t} (i-\bar{t})^2}, \text{ and} \tag{6.8}$$

$$\boldsymbol{\upsilon}(t) = \bar{\boldsymbol{\xi}}_h(t) - \boldsymbol{\kappa}(t) \cdot \bar{t} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \text{ with} \tag{6.9}$$

$$\bar{t} = \frac{\sum_{i=t-W}^{t} i}{W/\Delta t} \quad \text{and} \quad \bar{\boldsymbol{\xi}}_h(t) = \frac{\sum_{i=t-W}^{t} \boldsymbol{\xi}_h(i)}{W/\Delta t}. \tag{6.10}$$

6.2.2 Kalman Filter-based Extrapolation (KF)

Another widely applied technique is to first use a filter that optimally estimates the head-motion's state $\boldsymbol{\chi}_{h,\mathrm{KF}}(t)$ and then employ motion models to predict future positions. The Kalman filter is used in this study to estimate the orientation, angular velocity, and accelera-

tion values of the head-motion:

$$\boldsymbol{\chi}_{h,\mathrm{KF}}(t) = \begin{bmatrix} \theta_{h,\mathrm{KF}}(t) & \dot{\theta}_{h,\mathrm{KF}}(t) & \ddot{\theta}_{h,\mathrm{KF}}(t) \\ \phi_{h,\mathrm{KF}}(t) & \dot{\phi}_{h,\mathrm{KF}}(t) & \ddot{\phi}_{h,\mathrm{KF}}(t) \\ \psi_{h,\mathrm{KF}}(t) & \dot{\psi}_{h,\mathrm{KF}}(t) & \ddot{\psi}_{h,\mathrm{KF}}(t) \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{\xi}_{h,\mathrm{KF}}(t) & \dot{\boldsymbol{\xi}}_{h,\mathrm{KF}}(t) & \ddot{\boldsymbol{\xi}}_{h,\mathrm{KF}}(t) \end{bmatrix}^T. \tag{6.11}$$

The state-space model $\boldsymbol{\chi}_{h,\mathrm{KF}}(t+1)$ and the measurement model $Y(t)$ are given by:

$$\boldsymbol{\chi}_{h,\mathrm{KF}}(t+1) = \boldsymbol{A} \cdot \boldsymbol{\chi}_{h,\mathrm{KF}}(t) + \boldsymbol{B} \cdot \boldsymbol{U}(t) + \boldsymbol{G} \cdot \boldsymbol{W}(t), \tag{6.12}$$

$$\boldsymbol{Y}(t) = \boldsymbol{C} \cdot \boldsymbol{\chi}_{h,\mathrm{KF}}(t) + \boldsymbol{D} \cdot \boldsymbol{U}(t) + \boldsymbol{V}(t), \text{ with} \tag{6.13}$$

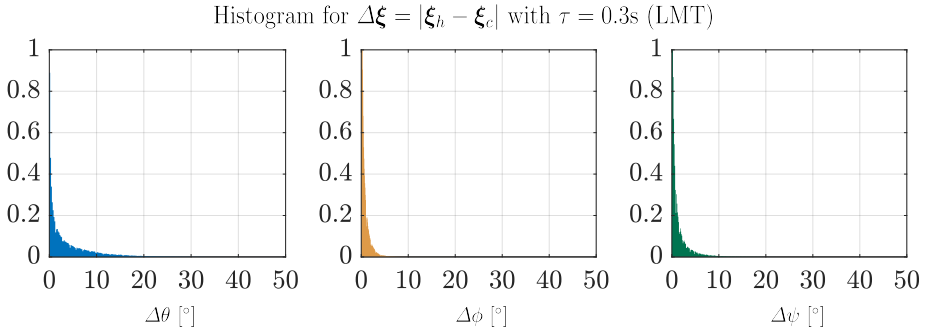
$$\boldsymbol{A} = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}(\Delta t)^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}. \tag{6.14}$$

Since the head orientation is a non-deterministic motion, the control input $\boldsymbol{U}(t)$ is unknown and thus set to $\boldsymbol{U}(t) = \boldsymbol{0}$; such that $\boldsymbol{B}$ and $\boldsymbol{D}$ are also implicitly set to $\boldsymbol{0}$. $\boldsymbol{W}(t)$ and $\boldsymbol{V}(t)$ are introduced as a random (white) disturbance and measurement noise to the model, respectively. $\boldsymbol{G}$ represents the process noise gain matrix and relates the process noise to the state variable. It is commonly set to $\boldsymbol{G} = \boldsymbol{I}$. $\boldsymbol{C}$ is the measurement gain matrix and takes the orientation measurements of the HMD into account. Assuming the head-motion's acceleration to be constant within each time step $\Delta t = 12.5\,\mathrm{ms}$, the following motion model is applied for predicting the future head orientation for an end-to-end delay of τ:

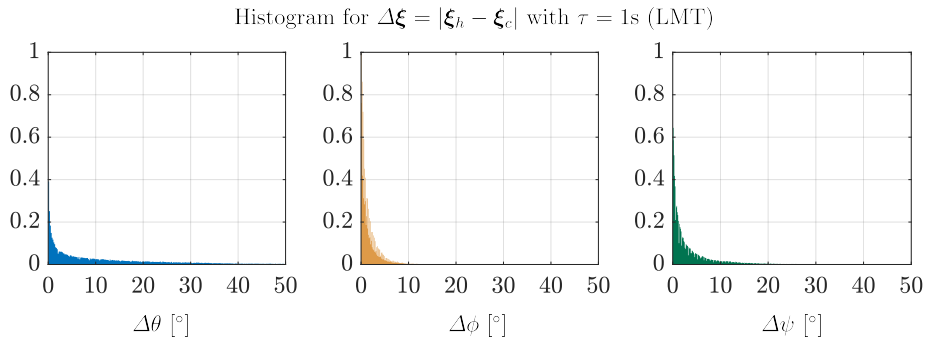
$$\hat{\boldsymbol{\xi}}_{h,\mathrm{KF}}(t+\tau) = \boldsymbol{\chi_h}_{,\mathrm{KF}} \cdot \begin{bmatrix} 1 \\ \tau \\ \frac{1}{2}\tau^2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\xi}_{h,\mathrm{KF}} & \dot{\boldsymbol{\xi}}_{h,\mathrm{KF}} & \ddot{\boldsymbol{\xi}}_{h,\mathrm{KF}} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ \tau \\ \frac{1}{2}\tau^2 \end{bmatrix}. \tag{6.15}$$

6.3 Probabilistic Head-motion Prediction

The prior art mainly applied either filter-based extrapolation techniques or linear regression models to forecast the head position. The level of achievable delay compensation can be greatly improved when deploying reliable viewport prediction. The LMT dataset is used to plot the absolute angular deviation for delays τ between 0.1 s to 1 s to observe any relevant patterns that might help to further improve the prediction accuracy. The histograms thereof are visualized in Figure 6.2 for $\tau = 0.3\,\mathrm{s}$ and $\tau = 1\,\mathrm{s}$ and show the horizontal motions (pan rotations) to be the most dominant ones with more fluctuations throughout the whole dataset. Table 6.1 shows the mean angular errors of the 90th, 99th, and 99.9th percentile for pan, tilt, and roll rotations considering all video sequences and participants in the LMT dataset for $\tau = 0.3\,\mathrm{s}$. It can be observed that horizontal motions necessitate larger buffer sizes to provide full compensation than tilt and roll rotations. Tilt rotations, instead, exhibit the least variations. The larger the delay is, the greater the deviations become. It is obvious that viewport prediction is necessary for proper delay-compensation. A closer examination of the histograms of the (absolute) angular deviations reveals that they show a similar pattern for



(a)



(b)

Figure 6.2: The histogram plots are shown for the absolute rotation errors with respect to end-to-end latenciess of (a) $\tau = 0.3\,\mathrm{s}$ and (b) $\tau = 1\,\mathrm{s}$ using the LMT dataset. The figures confirm that the horizontal motion appears to be the most dominant one as its distribution is significantly more spread compared to the tilt and roll rotations. This is visible both for small and large delays.

	Angular deviation for		
Percentiles	Pan ($\Delta\theta_h$) [°]	Tilt ($\Delta\phi_h$) [°]	Roll ($\Delta\psi_h$) [°]
$\mathbf{P_{90}}$	14.75	2.06	4.82
$\mathbf{P_{99}}$	35.50	5.50	14.31
$\mathbf{P_{99.9}}$	61.19	10.94	28.31

Table 6.1: The LMT dataset is leveraged to list the 90th, 99th, and the 99.9th percentile for the absolute angular deviations of pan, tilt, and roll rotations for $\tau = 0.3\,\mathrm{s}$. The numbers confirm that the horizontal movements are the most dominant ones. However, the roll rotations also seem to alter more than the tilt rotations.

all three DoFs. The errors for pan, tilt, and roll can all be approximated by a Gaussian distribution. In this context, an innovative data-driven head-motion prediction policy is proposed that is based on a probabilistic Gaussian error model.

The core idea behind the proposed approach is to employ an arbitrary predictor $\boldsymbol{\Gamma}(\boldsymbol{\xi}_h(t))$ that takes (at least) the current head orientations ($\boldsymbol{\xi}_h(t)$) as input to forecast the prospective

head position with respect to the present delay $\hat{\boldsymbol{\xi}}_{h,\boldsymbol{\Gamma}}(t+\tau)$:

$$\hat{\boldsymbol{\xi}}_{h,\boldsymbol{\Gamma}}(t+\tau) = \begin{bmatrix} \hat{\theta}_{h,\boldsymbol{\Gamma}}(t+\tau) & \hat{\phi}_{h,\boldsymbol{\Gamma}}(t+\tau) & \hat{\psi}_{h,\boldsymbol{\Gamma}}(t+\tau) \end{bmatrix}^T = \boldsymbol{\Gamma}(\boldsymbol{\xi}_h(t)). \tag{6.16}$$

Gaussian distributions are assumed for the previously described error distributions as illustrated in Figure 6.2 for $\tau = 0.3\,\mathrm{s}$ and $\tau = 1\,\mathrm{s}$. The probability density functions are fitted for the orientation deviations of each head direction. The mean $\boldsymbol{\mu}(\tau)$ and the standard deviation $\boldsymbol{\sigma}(\tau)$ parameters are trained for all 3 DoF and all delays between 0 s to 2 s with a step size of 0.1 s. The resulting solution spaces for $\boldsymbol{\mu}(\tau)$ and $\boldsymbol{\sigma}(\tau)$ as a function of the delay τ are then defined as:

$$\boldsymbol{\mu}(\tau) = \begin{bmatrix} \mu_\theta(\tau) & \mu_\phi(\tau) & \mu_\phi(\tau) \end{bmatrix}^T = \boldsymbol{\rho} \cdot \tau + \boldsymbol{\zeta}, \tag{6.17}$$

$$\boldsymbol{\sigma}(\tau) = \begin{bmatrix} \sigma_\theta(\tau) & \sigma_\phi(\tau) & \sigma_\phi(\tau) \end{bmatrix}^T = \boldsymbol{\eta} \circ \ln(\boldsymbol{\tau} + \boldsymbol{\gamma}) + \boldsymbol{\vartheta}, \tag{6.18}$$

with $\boldsymbol{\tau} = \tau \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$ and " $\circ$ " being the Hadamard product indicating an element-wise multiplication. The corresponding parameters are empirically determined as:

$$\boldsymbol{\rho} = \begin{bmatrix} -2.5 & 0.5 & 0.4 \end{bmatrix}^T \times 10^{-4}, \tag{6.19}$$

$$\boldsymbol{\zeta} = \begin{bmatrix} -92.1 & 10.1 & -116.2 \end{bmatrix}^T \times 10^{-4}, \tag{6.20}$$

$$\boldsymbol{\eta} = \begin{bmatrix} -0.3 & 0.0 & -0.1 \end{bmatrix}^T \times 10^{3}, \tag{6.21}$$

$$\boldsymbol{\gamma} = \begin{bmatrix} 45.2 & 2.8 & 8.2 \end{bmatrix}^T, \tag{6.22}$$

$$\boldsymbol{\vartheta} = \begin{bmatrix} 1.4 & 0.3 & 0.8 \end{bmatrix}^T \times 10^{3}. \tag{6.23}$$

$$\tag{6.24}$$

The solution spaces for the trained mean and standard deviation values as a function of the present delay are depicted in Figure 6.3. While the trained solution values for the tilt and roll rotations are similar to each other, the trained parameters for the pan rotation are significantly different. Assuming statistical independence of the individual head movement directions allows us to separately define the error probability density functions as:

$$\mathbb{P}\left(\hat{\boldsymbol{\xi}}_{h,\boldsymbol{\Gamma}}(t+\tau), \boldsymbol{\xi}_h(t), \tau\right) = \begin{bmatrix} P\left(\hat{\theta}_{h,\boldsymbol{\Gamma}}(t+\tau), \theta_h(t), \tau\right) \\ P\left(\hat{\phi}_{h,\boldsymbol{\Gamma}}(t+\tau), \phi_h(t), \tau\right) \\ P\left(\hat{\psi}_{h,\boldsymbol{\Gamma}}(t+\tau), \psi_h(t), \tau\right) \end{bmatrix}, \tag{6.25}$$

$$\text{with } P(\alpha, \beta, \tau) = \frac{1}{\sqrt{2\pi} \cdot \sigma_\beta(\tau)} \cdot \exp\left(-\frac{((\alpha - \beta) - \mu_\beta(\tau))^2}{2 \cdot \sigma_\beta(\tau)^2} \right). \tag{6.26}$$

The resulting probabilistic head-motion predictor $\hat{\boldsymbol{\xi}}_{h,\mathrm{prob}}(t+\tau)$, which forecasts the prospective head position for the present latency τ, is then specified as:

$$\hat{\boldsymbol{\xi}}_h^{\mathrm{prob}}(t+\tau) = \boldsymbol{\xi}_h(t) + \mathbb{P}\left(\hat{\boldsymbol{\xi}}_{h,\boldsymbol{\Gamma}}(t+\tau), \boldsymbol{\xi}_h(t), \tau\right) \circ \left(\hat{\boldsymbol{\xi}}_{h,\boldsymbol{\Gamma}}(t+\tau) - \boldsymbol{\xi}_h(t)\right). \tag{6.27}$$

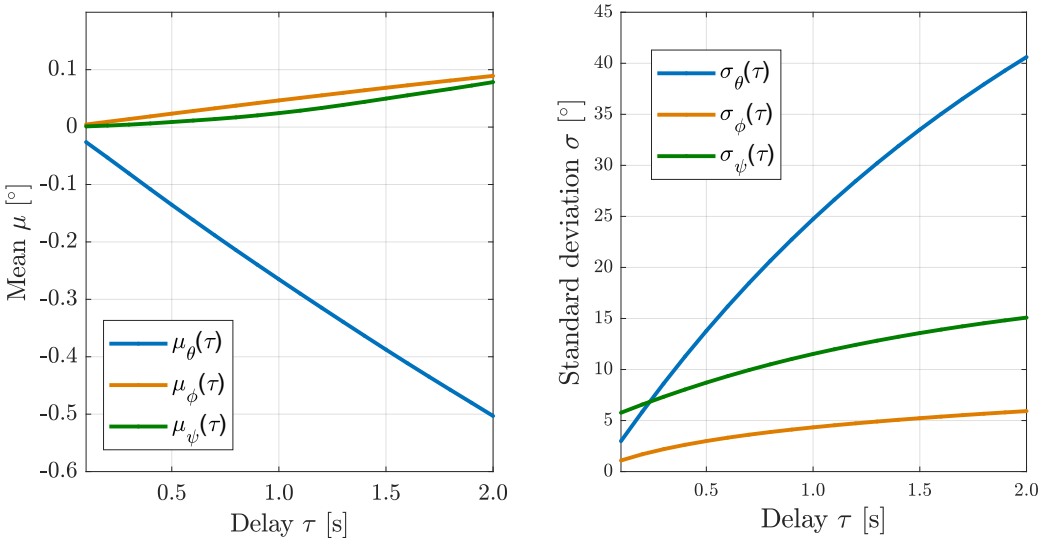


Figure 6.3: The solution spaces for the trained mean and standard deviation values are showcased with regard to their respective head rotation direction. The plots demonstrate that the dominant pan rotation exhibits noticeably different solution values compared to the trained values for tilt and roll rotations, which appear to be quite similar.

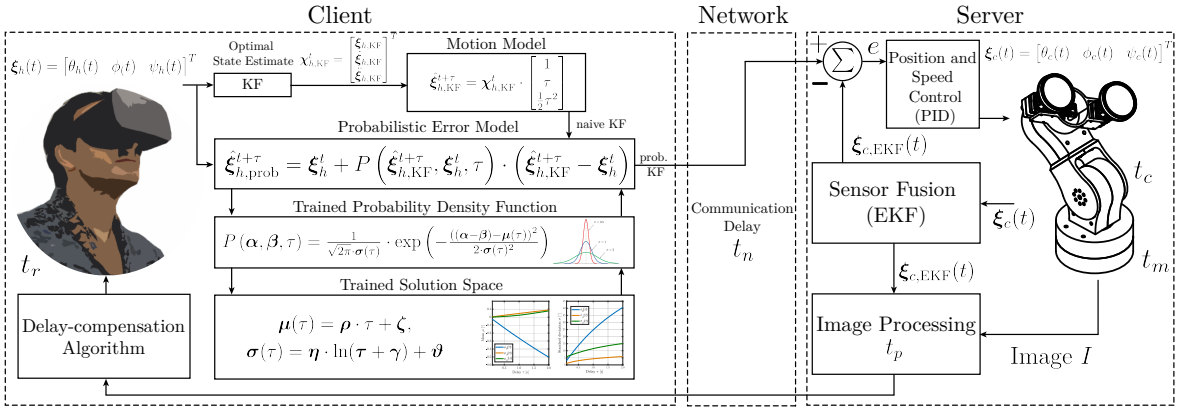


Figure 6.4: Overview of the complete telepresence processing pipeline that is needed for robust head-motion prediction in terms of the probabilistic modification approach presented here.

Figure 6.4 illustrates how the probabilistic head-motion prediction technique is implemented for the sensor head of the MAVI telepresence platform. The head orientation data is first acquired on the client-side in form of quaternions. After converting the quaternions into the Euler angle representation, the Kalman filter is used to obtain optimal state estimates. The estimated orientation, velocity, and acceleration values of the head movements are then fed into the constant acceleration motion model for extrapolation with respect to the current end-to-end delay. The previously derived solution spaces are used to modify the predicted values by means of the probabilistic error model. The trained error models countervail potential overshoots by weighing the predicted value with its probability to be at that position. The modified estimated orientation values are then sent to the server to anticipate the prospective motion. A proportional-integral-derivative (PID) controller is integrated to

ensure smooth motions of the PTR-U. Image processing steps are further incorporated to fasten the image processing pipeline. The debayering procedure, for instance, is offloaded to the graphics processing unit (GPU) for low-delay streaming followed by viewport selection, lens distortion removal, and efficient encoding. A more detailed description is presented in Chapter 3.

6.3.1 Results

To quantify and highlight the benefit of the probabilistic weighing technique, the MAE, the RMSE, and the compensation rate metrics, which are introduced in Chapter 3, are used as qualitative measures. The mean and standard deviation parameters of the probabilistic error model are trained with the complete LMT dataset. The weighing technique are investigated for the linear regression and the KF-based extrapolation methods as these are introduced as state-of-the-art representatives in Sections 6.2.1 and 6.2.2. The resulting mean absolute values and root mean square error values for the LMT dataset are shown in Figure 6.5 (a) and (c). The labels *Naive LR* and *Naive KF* correspond to the original prediction without a probabilistic weighing. *Prob. LR* and *Prob. KF* indicate the adoption of the proposed probabilistic error model. It is apparent that the probability-based adaptation approach leads to a fundamental error reduction both for the MAE and the RMSE values.

This tendency is also reflected in the achievable degree of delay-compensation. Figure 6.5 (e) depicts the compensation rates of the probabilistic head-motion predictors compared to their naive versions as well as to the naive case where no prediction and no compensation is deployed at all. The probabilistic error model helps to significantly improve the level of delay-compensation. A sizable improvement is particularly visible for large delays. The final predictor, as is defined in Equation (6.27), takes an initial prediction method as a basis (e.g., LR) and weighs the estimated orientation values with its trained probabilities to be at that position. The higher the end-to-end latency between the client and the server is, the less trust is given to the initial predictor and the more weight is assigned to the current position. In this way, large overshoots are avoided, which would lead to huge discrepancies between the predicted and true values. The probabilistic error model-based prediction can thus be interpreted as a more passive, secure way of forecasting prospective head-motions. All measures clearly show a substantial improvement of the probability-based approach compared to their naive variants. Nevertheless, the results also show that the naive version cannot be infinitely improved. The outcomes indicate a saturation behavior for the level of performance boost that can be expected, as the probabilistic versions, both for the linear regression and the KF-based extrapolation technique, show only minor performance differences between each other.

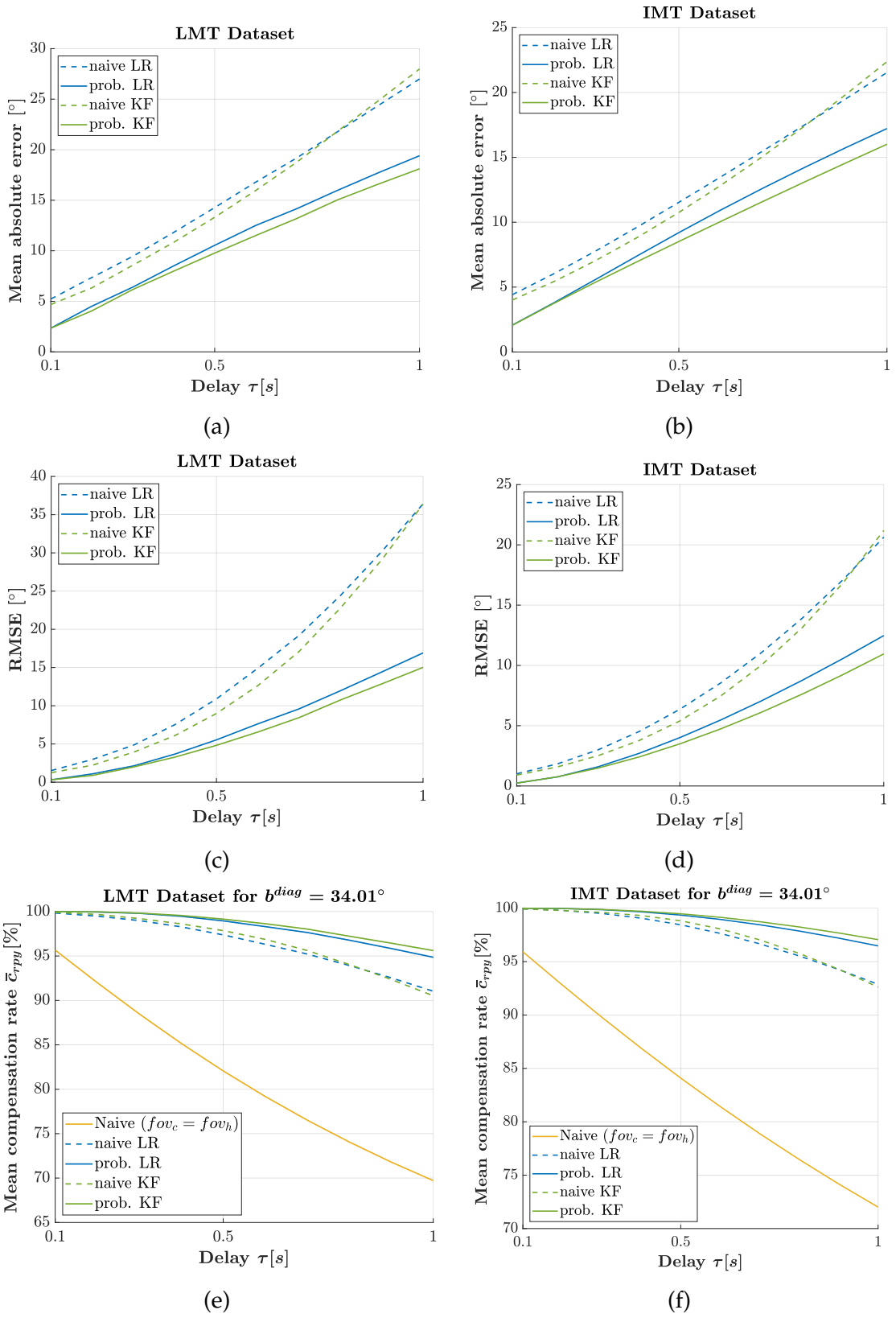


Figure 6.5: Comparison of representative state-of-the-art head-motion prediction methods, as well as their probabilistic versions. The mean absolute error, the root mean square error, and the compensation rate are used as metrics to quantify the prediction accuracy and the retrievable degree of delay-compensation. All methods are investigated for both the LMT ((a), (c), (e)) and the IMT ((b), (d), (f)) dataset to examine their general validity. The results verify the additional benefit that is gained by the use of the probabilistic weighing technique presented here (reproduced from [6] © 2018 IEEE).

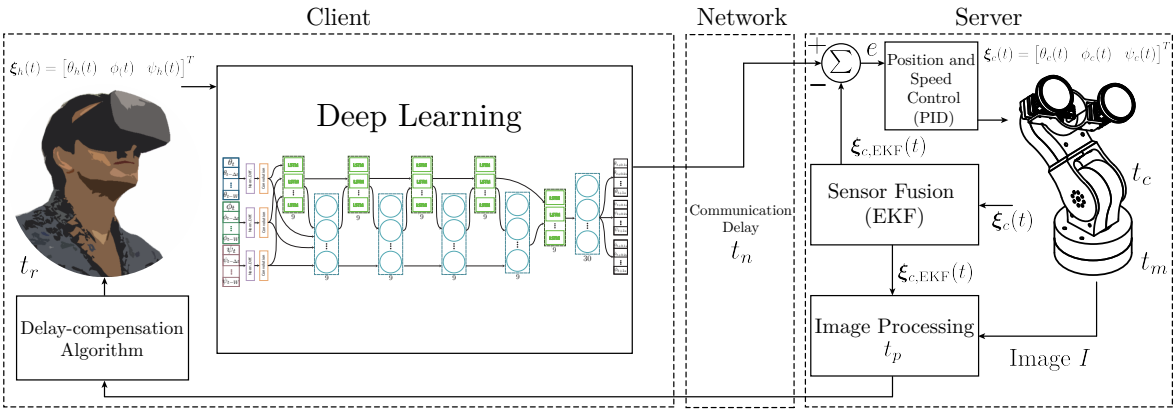


Figure 6.6: Schematic overview of the processing pipeline required to perform proper head motion prediction. The probabilistic head motion prediction approach is substituted by artificial intelligence that is supposed to learn the behavior of the user in order to forecast prospective head orientations accurately.

The parameters needed for the probabilistic error model are trained by means of the LMT dataset. The approach presented here was additionally evaluated on the separate IMT dataset to prove its general validity. Their MAE, RMSE and compensation rate values are depicted in Figure 6.5 (b), (d), and (f). The cross-validation with the IMT dataset proves that the probability-based adaptation policy performs equally well for the entirely different IMT dataset. The probabilistically weighted versions once again demonstrate a substantial performance boost as opposed to their naive versions. Similar behavior is noticeable for the compensation rate, where the probabilistic error models lead to a better degree of latency compensation.

6.4 Deep Learning-based Head-motion Prediction

The probability-based head-motion prediction approach exhibits great improvement compared to the prior art. As touched upon in the previous section, performance saturation is identified when analyzing the results. To further improve the achievable degree of delay-compensation, deep learning-based techniques are first investigated on head-motion data for viewport prediction. Figure 6.6 illustrates how the intensive filter-based state estimation and prediction procedure is replaced by an end-to-end deep learning-based prediction framework.

Various architectures ranging from simple dense feed-forward networks (FFNs), and deep recurrent neural networks such as the well-established Long Short-Term Memory (LSTM) [137] to more sophisticated joint structures are developed and investigated as shown in Figure 6.7. The LMT dataset is separated into a training set using 81 head-motion profiles of 27 participants ($P = 27$), a validation set with three profiles of one user ($P = 1$), and a test set containing six profiles of two participants ($P = 2$). The deep head-motion predictors presented here are also cross-validated on the independent IMT dataset to prove their generalization capabilities. The IMT dataset was recorded with different participants and contains

sensory information acquired under dissimilar conditions.

The input of the network consists of the current pan $\theta(t)$, tilt $\phi(t)$, and roll $\psi(t)$ orientations as well as their past trajectories within a certain time window W. $W = 0\,\mathrm{ms}$ would imply that only the current pan, tilt, and roll orientations are considered. Empirical experiments revealed that a window of $W = 250\,\mathrm{ms}$ delivers the best results. The time step size is set to $\Delta t = 12.5\,\mathrm{ms}$. The networks thus receive the 20 last sensory data for each orientation direction. Taking the orientation values directly worked sufficiently for the LMT dataset, which the network was trained for. Testing these networks on the IMT dataset, however, led to very erroneous inferences. In order to remedy such unsatisfactory tendencies and provide a fully generalized end-to-end solution that is supposed to work for any arbitrary head orientation sequence, a data pre-processing step is added. The orientation values are first scaled to be in the range of $-180\,^\circ$ to $180\,^\circ$. Rather than feeding the absolute orientation values into the network, the inputs are further subdivided into their respective orientation groups and the differences are taken into account instead:

$$\boldsymbol{\xi}_{\text{diff}}(t) = \boldsymbol{\xi}(t) - \boldsymbol{\xi}(t - \Delta t). \tag{6.28}$$

The differences are then normalized to be in the range of -1 to 1 to produce a generalized representation of the input data:

$$\tilde{\boldsymbol{\xi}}_{\text{diff}}(t) = \frac{\boldsymbol{\xi}_{\text{diff}}(t)}{\max\left(|\boldsymbol{\xi}_{\text{diff}}|\right)}. \tag{6.29}$$

This step is essential to ensure a generalization for other datasets. Omitting this step leads to erroneous inferences as is discussed before. The processed data is then fed into a 1D convolutional layer with a kernel size of ten, which acts as a low pass filter. The idea is to be robust against fast fluctuations and noise, which is a severe issue and even amplified when considering the differences of the orientations. Each of the network structures illustrated in Figure 6.7 is trained using the mean absolute error as loss function (L1-norm), which showed superior performance compared to the mean squared error (MSE) (L2-norm). The output of the deep networks (except for the dense feed-forward neural network (FFN)) predicts a whole course of future orientations from 0.1 s to 1 s with a step size of 0.1 s ($\boldsymbol{\xi}^{\text{pred}}_{t+n\cdot 0.1\text{s}}\ \forall n \in \mathbb{Z}, n = 1, ..., 10$). However, the actual output of the network is a sequence of future normalized differences $\hat{\tilde{\boldsymbol{\xi}}}_{\text{diff}}(t)$. The remapping of normalized differences to absolute orientation values is accomplished with:

$$\hat{\boldsymbol{\xi}}(t) = \boldsymbol{\xi}(t) + \sum_{k=t-\tau/\Delta t}^{t} \hat{\tilde{\boldsymbol{\xi}}}_{\text{diff}}(k) \cdot \max\left(|\boldsymbol{\xi}_{\text{diff}}|\right). \tag{6.30}$$

To the best of our knowledge, this is the first study that applies deep learning for head-motion prediction. The initial strategy is, therefore, to first examine simple dense feed-forward networks (Figure 6.7(e)), although typically not applied for time series prediction, to gain experience about the prediction behavior for head-motion data samples. The network is shown in Figure 6.7 (e). Given the fact that the way of feeding the input data to the network does not change for different latency values, the network has difficulties in learning for different delays. The output of the dense FFN depicted in Figure 6.7 (e) is a single orientation

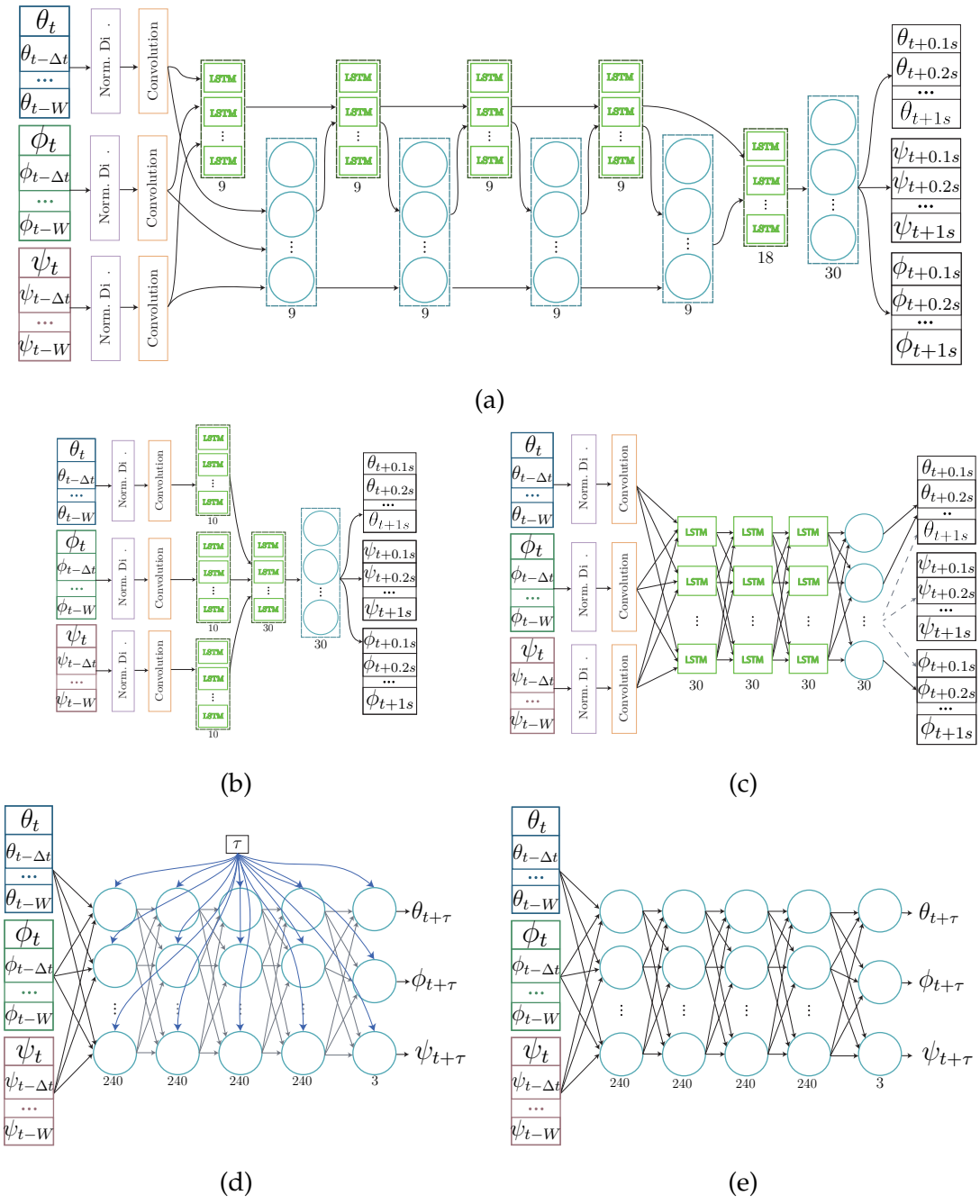


Figure 6.7: The deep learning-based head-motion prediction technique is applied to IMU-based head-motion orientation values for the first time. This figure gives an overview of the developed and investigated deep neural architectures: (a) Interleaved structure of LSTMs and dense feed-forward networks, (b) subdivided LSTM layers for each orientation direction that are then fused with another LSTM layer and a final dense FFN, (c) fully connected LSTM network, (d) fully connected dense feed-forward network with delay shortcuts, and (e) fully connected dense feed-forward network (reproduced from [1] © 2018 IEEE).

value for the future pan, tilt, and roll rotation. The network has to recognize the present delay with respect to the input values and predict the future position accordingly. The performance of this kind of deep structure is poor and hence not applicable for time-varying

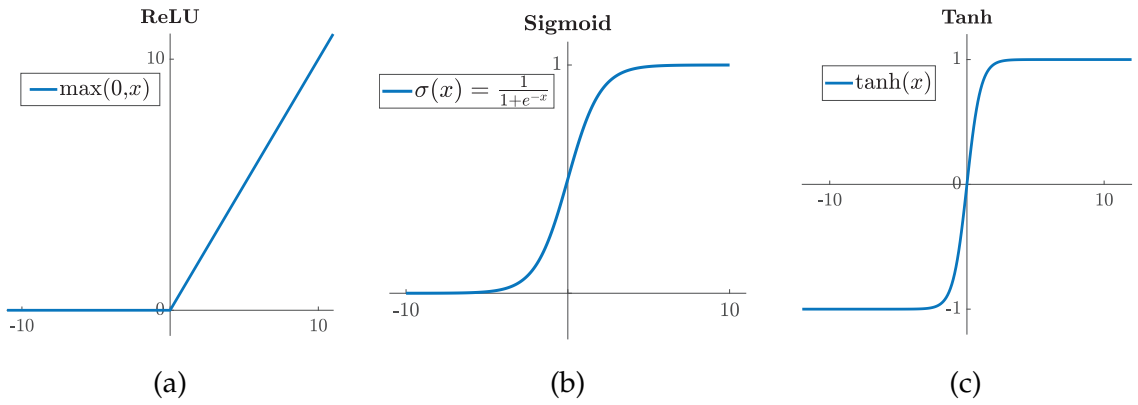


Figure 6.8: The (a) ReLU, (b) Sigmoid, and (c) Tanh functions are popular non-linear activation functions that are used in deep learning architectures.

systems. The deep structure presented in Figure 6.7 (d) is designed to address this issue. The idea is to mediate the knowledge of the present delay to all other nodes. This is accomplished through shortcut injections. This modification, however, led only to a minor improvement. A more intuitive way is to apply deep recurrent neural networks (RNNs), as they are characterized by a feedback loop and thereby establish a way of memorization by sharing weights over time. RNNs typically suffer from the vanishing gradient problem. In some cases, the gradients will become very small, preventing the weights within the network from changing its value. The issue of vanishing gradients is a severe problem for time series prediction applications, as the learning of long data sequences is hampered. A way of mitigating this problem is to use LSTMs [137], which are the most well-known and promising representatives of RNNs that provide a solution for the vanishing gradient problem. LSTMs solve the problem by establishing a link between its forget gate activations and the computation of the gradients. This link allows the information that is supposed to be memorized by the LSTM module to flow through the forget gate. Figure 6.7 (b) and (c) illustrate two different LSTM-based architectures that are investigated for head-motion prediction. In Figure 6.7 (a), a new type of architecture is depicted that is designed as an interleaved architecture of LSTMs and dense FFNs. The motivation is to increase the number of weights to improve the learning behavior and maintain the memorization characteristics by the LSTM layers.

The Adam optimization [138] algorithm is used as an extension to stochastic gradient descent, combining both advantages of RMSProp and AdaGrad [139]. The maximum number of epochs is set to 1000. The early stopping technique (patience=2) is applied to avoid overfitting. In addition, a learning rate decay scheduler is used to decrease the initial learning rate of 0.001 every 30 epochs by 70 %. The batch size is set to 2^{11}. The rectified linear unit (ReLU) is deployed as the activiation function for the FFNs, since ReLUs reduce the likelihood of the gradients to vanish and are computationally more efficient compared to the Sigmoid function, for instance. A comparison of popular activation functions is shown in Figure 6.8. The deep learning approaches are implemented in Keras [140] and Tensorflow [141]. The time needed to predict future orientation values is in the domain of single-digit microseconds (Core i7 (x64), GeForce GTX 1080 Ti) and thus is suitable for realtime applications.

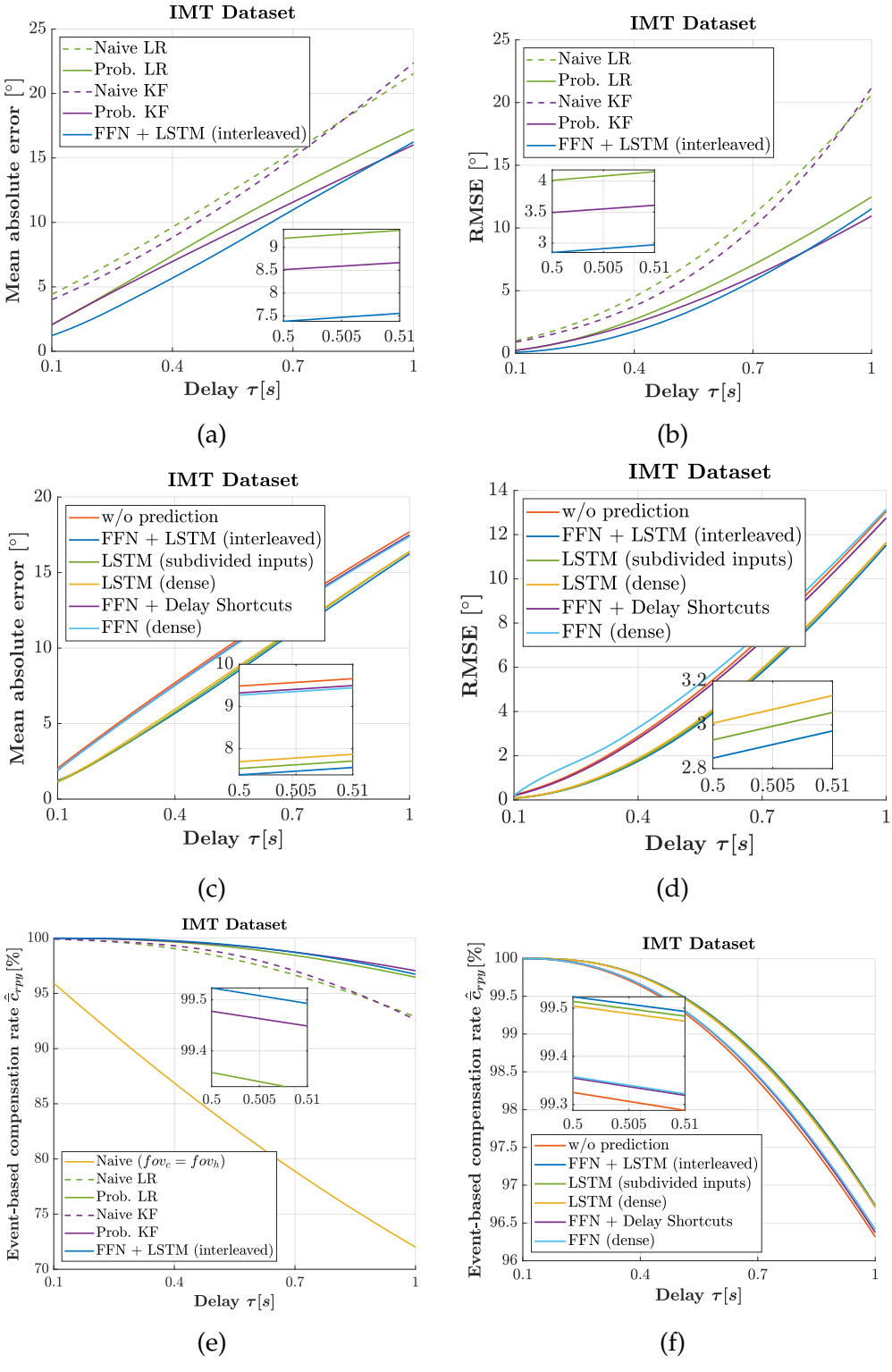


Figure 6.9: The mean absolute error and the root mean square error are considered as valid metrics to fairly compare the investigated added value of the probabilistic weighing technique as well as the performance of the deep architectures shown in (c) and (d). Figure (e) compares the compensation rates of the deep architectures. The interleaved structure of LSTMs and FFNs turns out to be the best-performing deep learning-based prediction approach. Its achievements are set against the state-of-the-art naive prediction methods as well as their probabilistic version. The deep architectures outperform the prior art for latencies up to 0.7 s. Higher delays benefit from using the KF-based probabilistic adaptation technique, which work better for high latencies.

6.4.1 Results

The accuracy and smoothness of the deep learning-based head-motion predictors are again evaluated by means of the MAE and RMSE values to provide common metrics for a fair comparison. The results of the previously mentioned architectures are illustrated in Figure 6.9 (a) and (b). The resulting degree of achievable delay-compensation is depicted in Figure 6.9 (c). The dense FFN exhibits only a slight improvement compared to applying no prediction at all. Mediating the knowledge of the present delay as shortcut injections to the nodes of the dense FFN did not lead to any major performance boost, although a minor tweak is observable. When using RNNs, LSTMs in particular, the performance can be improved in terms of MAE and RMSE values. Their merits are also reflected in the respective event-based compensation rate measures. The best performance is achieved by the interleaved architecture of LSTM and dense FFN blocks (see Figure 6.7 (a)).

The best performing deep learning-based head-motion predictor – the interleaved architecture of LSTMs and dense FFN blocks – is separately compared to state-of-the-art methods presented in Figure 6.9 (d) - (f). It is contrasted with widely employed approaches such as the linear regression technique and the polynomial extrapolation method, which is premised on a constant acceleration motion model and a Kalman filter-based optimal state estimate. They are tagged as *naive LR* and *naive KF*. The probabilistic versions thereof, introduced in Section 6.3, are further included in the figures and are termed *prob. LR* and *prob. KF*. Compared to the state-of-the-art, the deep learning-based approach clearly shows noticeable improvements both for the MAE and RMSE values as well as the achievable level of delay-compensation. The probability-based modification of the state-of-the-art methods performs worse than the deep learning-based approach but still exhibits competitive outcomes. For latency values higher than $0.9\,\mathrm{s}$, the *prob. KF* is even able to excel over the deep learning-based approach. The same behavior can be observed in the event-based compensation rate, where the deep interleaved network performs better than the other approaches.

6.5 Spatio-temporal Viewport Prediction

The interleaved deep architecture of LSTMs and dense FFNs proved to show solid performance. The only source of information that is currently considered is the head orientations from the IMU. Scene information, however, can be another valuable information source that might help to predict the future viewport more accurately. In contrast to related work, where spatial and temporal scene information along with past head trajectories are fed into a deep network, this study proposes a solution that chooses a different way of approaching this issue. Rather than early fusing the input data, a late fusion strategy is presented. The ulterior motive is to provision a stable, reliable base network that mainly relies on past head trajectories and to improve it with additional scene semantics as an optional source. In this way, a solution is provided that is robust and agnostic to noisy, low-quality images, poor lighting conditions, and any other potential source of disturbances that might impact the performance of the viewport predictor.

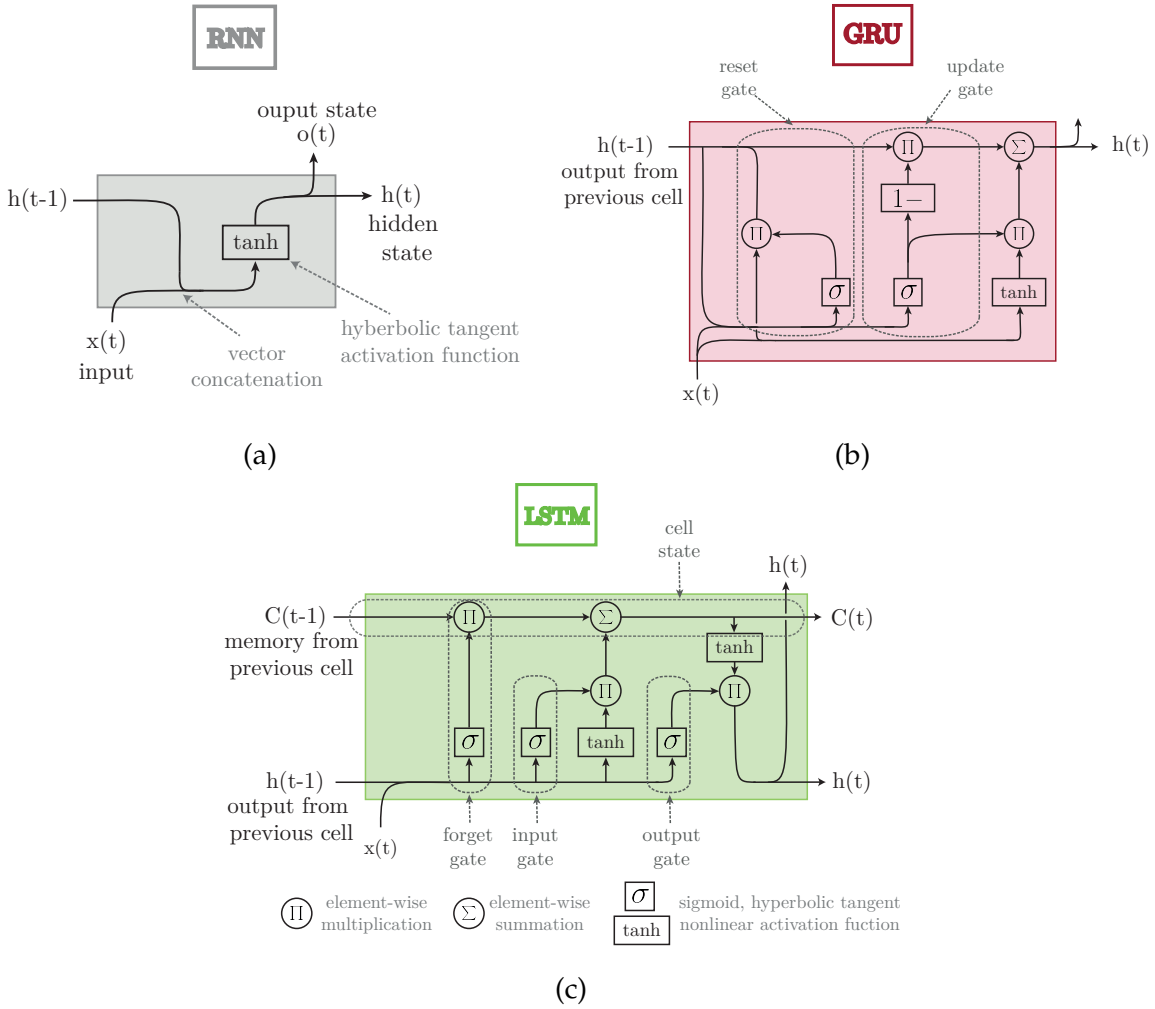


Figure 6.10: Overview of state-of-the-art recurrent neural networks. (a) shows the architecture of a general recurrent neural network (RNN), (b) a gated recurrent unit (GRU) [142], and (c) a long short-term memory (LSTM) cell [137]. All architectures share their weights over time. GRUs are considered to be a modified version of LSTMs. Their forget and input gates are combined into a single update gate, merging the cell state and memory information into one state.

6.5.1 Head Orientation-based Deep Network

The semantic viewport prediction scheme relies on a base network that is able to already deliver highly accurate prediction values without leveraging any scene information. For this reason, further effort is put into advancing the head orientation-based deep network, termed the H-network. A new type of deep network is proposed that is based on an interleaved structure of gated recurrent units (GRUs) [142] and convolution layers. GRUs belong to the group of recurrent neural networks and share weights over time, and thus capture long-term dependencies. GRUs can be seen as modified LSTMs, where the forget and input gates are combined into a single update gate. The cell state and memory information is thereby merged into one state. GRUs are conceptually simpler and thus computationally more efficient. Empirical analyses of sequence modeling tasks revealed no substantial performance difference between GRUs and LSTMs [143]. A structural comparison of RNNs,

LSTMs, and GRUs is depicted in Figure 6.10.

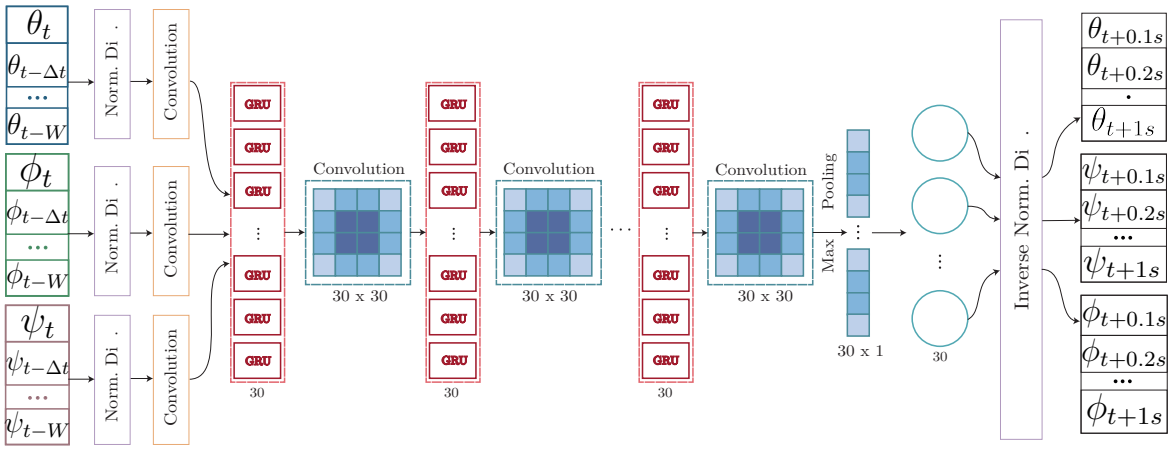


Figure 6.11: Illustration of the novel deep learning-based head-motion prediction model. The proposed deep architecture uses an interleaved structure of five stacked GRU layers with 30 cells followed by a convolutional layer. A max pooling layer extracts the most distinct features and converts the output to the desired output dimensions. A final dense feed-forward layer is incorporated to increase the number of learned parameters for enhanced prediction capabilities (adopted from [2] © 2019 IEEE).

The proposed models' input and output are identical to the processing pipeline introduced in Section 6.4. The deep network is fed with a sequence of current and past pan, tilt, and roll orientation values. Past trajectories within an empirically discovered window of W$=250\,\mathrm{ms}$ (last 20 values) are considered for each orientation direction. The network's generalization capabilities are ensured by transforming the absolute orientation values into their normalized differences according to Equations (6.28) and (6.29). The core of the novel network consists of an interleaved structure of six dense GRU layers each with 30 cells and subsequent convolution layers with a kernel size of 30×30, which extract the most distinct features. A successive *max pooling* layer is incorporated as a discretization process. The output of the last convolution layer is thereby downsampled while conserving the most significant features. After passing the output of the *max pooling* layer through a dense feed-forward network, which aims to increase the amount of trainable weights, the remapping to absolute orientation values takes place as defined in Equation (6.30). A graphical visualization of the deep network is provided in Figure 6.11. The training procedure is configured for 100 epochs, including an early stopping technique (patience=10) to avoid overfitting. The batch size is set to 2^9 time samples. A learning rate decay scheduler is incorporated to decrease the learning rate over time. The initial learning rate of 10^{-3} is decreased every ten epochs by 70 %. The Adam optimizer is used as the gradient descent algorithm of choice. Similar to the previous network, ReLUs are used as the activation function for the FFN blocks. The proposed network is also implemented in Keras and Tensorflow using the Core i7 (x64) machine (GeForce GTX 1080 Ti) for training and inference. The network predicts prospective orientation values in the range of single-digit microseconds, thus proving its realtime inference abilities.

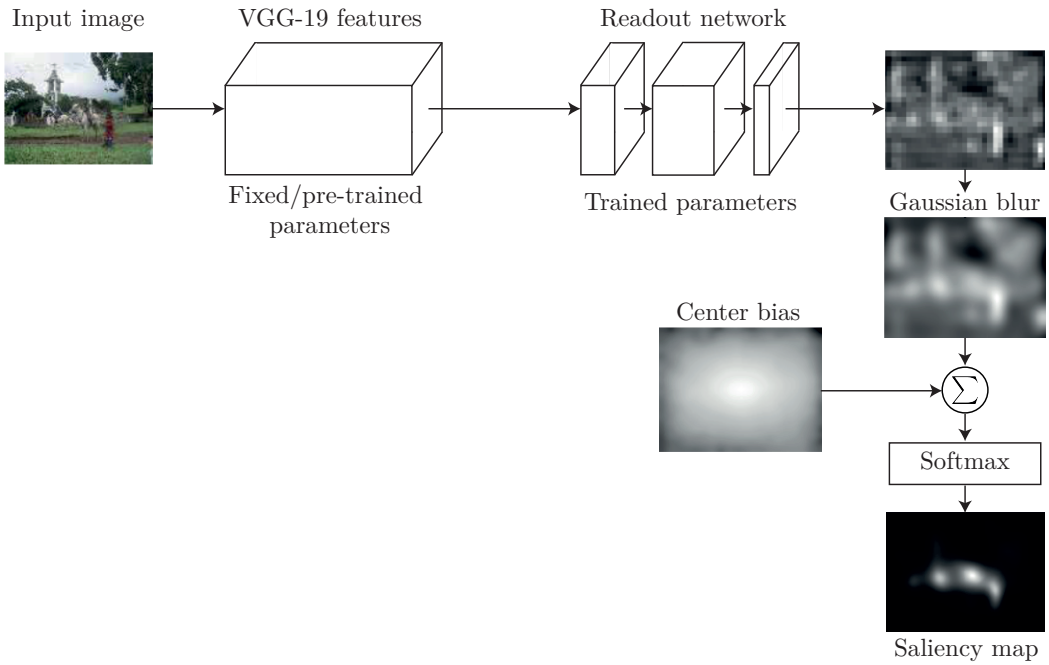


Figure 6.12: Overview of the adopted Deepgaze II model proposed in [145] for saliency creation. The network is built upon high-level features of the VGG-19 network that is initially developed for object recognition [146]. The Deepgaze II model extracts the VGG-19 features and adds a readout network along with a center bias to produce saliency maps (reproduced from [145] © IEEE 207).

6.5.2 Saliency Maps

Saliency maps $S(t)$ convey information about the visual attractiveness of spatial regions within an image frame at time t. They can contain information that might help to improve the accuracy of the prediction policy. A user experiencing a (3D) 360° video with an HMD is likely to navigate to regions within the scene that might appear visually attractive. This assumption is, of course, dependent on the user's task or intended action. However, the prior art showed that head and eye gaze interactions are coupled and that gaze statistics are correlated with saliencies within the scene [144].

Salient regions can be detected within the scene to anticipate the user's prospective gaze direction. Saliency detection is recently gaining more and more attention in research. Attractive applications include salient object detection, visual tracking, and attention and fixation prediction. The accuracy of saliency maps can be quantified by using the widely deployed Area Under the ROC (Receiver Operating Characteristics) Curve (AUC) metric [147]. The AUC interprets saliency maps as classifiers for pixels that are fixated or not [147]. ROC measures the performance of a classifier, and depicts the trade-off between *true positive* and *false positive* rates [148].

The literature contains various methods for saliency map creation, which range from manually selecting low-level and high-level features to data-driven deep learning-based architectures. An in-depth discussion of this can be found in [149]. The success in creating saliency maps increased dramatically with the application of deep neural networks. End-to-end deep networks are either directly trained from scratch [150] or transfer learning is applied by lever-

aging parts of pre-trained deep networks that were initially designed for object recognition tasks. Representative examples are Deepgaze I [151], Deepgaze II [145], SALICON [152], and ML-NET [153]. The MIT saliency benchmark [154] allows the comparison of state-of-the-art saliency map generation approaches. This benchmark scores and reports the performance for latest saliency models [154].

Within the scope of this work, the Deepgaze II model is adopted for saliency map creation due to its high score rating, especially for the AUC metric. At the time of this study, no other model is ranked better. The Deepgaze II model, visualized in Figure 6.12, is based on the features of the VGG-19 network [146], which is a deep convolutional network with 19 layers and designed for image recognition. Input images are subsampled by a factor of two and fed to the normalized VGG-19 network. Rather than taking the output of the VGG-19 network, the feature maps of high-level convolutional layers are extracted for saliency detection [145]. These high-level feature maps are then merged into one 3D tensor with 2560 (5×512) channels, which is used as input for a second neural network that the authors termed the *readout network*. This readout network constitutes four layers of 1×1 convolutions followed by ReLU nonlinearities. The final output of the readout network is then blurred by convolving it with a Gaussian kernel to regularize the predictions. A center bias is added as a prior distribution, since fixations tend to be near the center of the image. A *softmax* layer is utilized as a final step to convert the output into a probability distribution. Figure 6.13 (a) illustrates the ouput of the Deepgaze II model for a sample viewport.

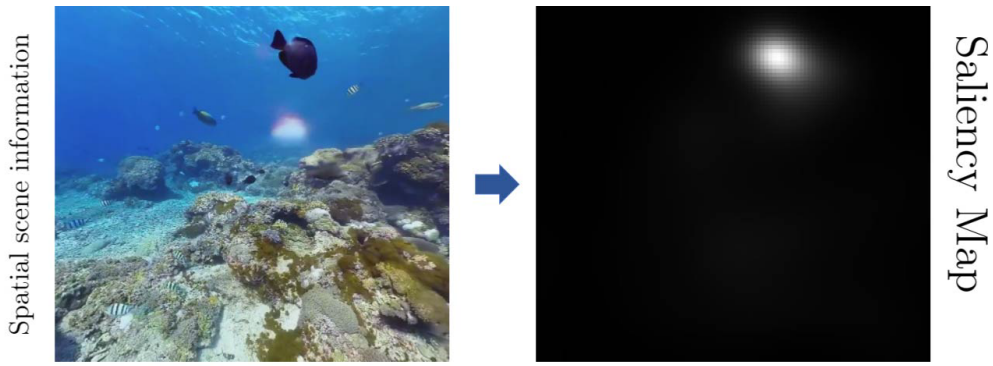
6.5.3 Motion Maps

Not only the spatial but also the temporal component plays a significant role for gaze prediction. Moving objects within a scene might temporarily influence the human's gaze direction. In this study, motion maps $M(t)$ are created to quantify the amount of motion present within the viewport of a scene given an image frame at time t. Optical flow is a valid and widely deployed method to generate motion maps. Optical flow estimates the motion as either instantaneous image velocities or discrete image displacements by computing the 2D displacement of each pixel in successive frames as a 2D vector field. Fundamental for the optical flow to work is the assumption that pixel intensities of an object do not change between consecutive frames and neighboring pixels have similar motion.

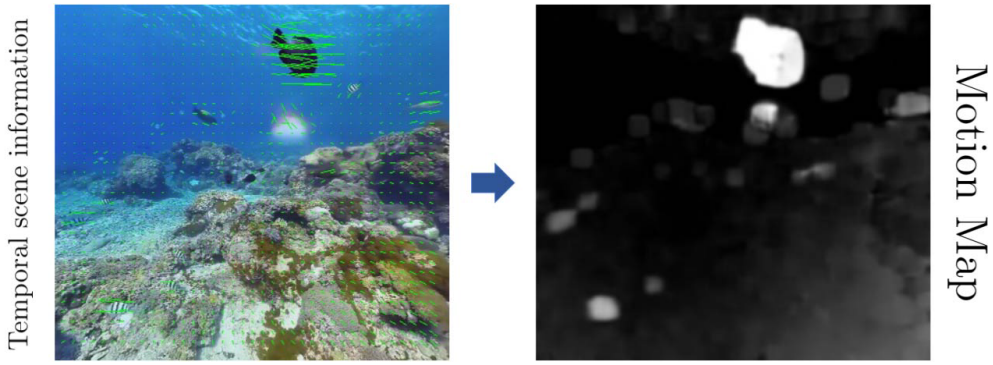
Within the scope of this thesis, Farneback's motion estimation algorithm is selected to calculate the motion maps [155]. A quadratic polynomial is defined to approximate the neighborhood of each pixel [155]:

$$f_1(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{A}_1 \boldsymbol{x} + \boldsymbol{b}_1^T \boldsymbol{x} + c_1, \tag{6.31}$$

with $f_1(\boldsymbol{x})$ being the intensity signal and $\boldsymbol{x}$ the pixel position. The coefficients $\boldsymbol{A}_1$, $\boldsymbol{b}_1$, and c_1 are estimated by means of a weighted least squares method with respect to the signal values within the corresponding neighborhood. A new signal $f_2(\boldsymbol{x})$ can then be derived from $f_1(\boldsymbol{x})$



(a)



(b)

Figure 6.13: The images in (a) show the output of the Deepgaze II network for a random input image. (b) illustrates the extraction of temporal information within a viewport. The motion vectors are shown in the left image. The right picture demonstrates the conversion of motion vectors to a probability map.

and a global displacement $\boldsymbol{d}$:

$$f_2(\boldsymbol{x}) = f_1(\boldsymbol{x}-\boldsymbol{d}) = (\boldsymbol{x}-\boldsymbol{d})^T \boldsymbol{A}_1 (\boldsymbol{x}-\boldsymbol{d}) + \boldsymbol{b}_1^T (\boldsymbol{x}-\boldsymbol{d}) + c_1 \tag{6.32}$$

$$= \boldsymbol{x}^T \boldsymbol{A}_1 \boldsymbol{x} + (\boldsymbol{b}_1 - 2\boldsymbol{A}_1 \boldsymbol{d})^T \boldsymbol{x} + \boldsymbol{d}^T \boldsymbol{A}_1 \boldsymbol{d} - \boldsymbol{b}_1^T \boldsymbol{d} + c_1 \tag{6.33}$$

$$= \boldsymbol{x}^T \boldsymbol{A}_2 \boldsymbol{x} + \boldsymbol{b}_2^T \boldsymbol{x} + c_2. \tag{6.34}$$

The coefficients $\boldsymbol{A}_2$, $\boldsymbol{b}_2$, and c_2 can thus be expressed in terms of $\boldsymbol{A}_1$, $\boldsymbol{b}_1$, and c_1:

$$\boldsymbol{A}_2 = \boldsymbol{A}_1, \tag{6.35}$$

$$\boldsymbol{b}_2 = \boldsymbol{b}_1 - 2\boldsymbol{A}_1 \boldsymbol{d}, \tag{6.36}$$

$$c_2 = \boldsymbol{d}^T \boldsymbol{A}_1 \boldsymbol{d} - \boldsymbol{b}_1^T \boldsymbol{d} + c_1. \tag{6.37}$$

The displacement $\boldsymbol{d}$, which estimates the motion between two consecutive frames, can then be computed as:

$$\boldsymbol{d} = -\frac{1}{2} \boldsymbol{A}_1^{-1} (\boldsymbol{b}_2 - \boldsymbol{b}_1). \tag{6.38}$$

The resulting optical flow field represents the actual motion between two frames. To extract the relative motion within the scene, the ego-motion of the user is removed by subtracting

it from the corresponding motion vectors. The resulting dense vector field is then converted into a grayscale motion map that resembles the probability distribution of saliency maps. To smooth the map and remove noise, a Gaussian filter is applied. Figure 6.13 (b) illustrates the dense vector field and its motion map for an example input image.

6.5.4 Deep Spatio-temporal Fusion

In contrast to the state-of-the-art, a late-fusion strategy is chosen to ensure agnosticism to the deployed camera system as well as to scene conditions. Thus, two further networks are developed that leverage spatial and temporal image information by means of saliency and motion maps, respectively. In the remainder of this study, these architectures are termed as the *saliency network (S-network)* and the *motion network (M-network)*. The networks' architectures are illustrated in Figure 6.14.

6.5.4.1 Saliency Network

The S-network takes all saliency maps into account that are within an empirically determined window size of $W_c = 500\,\mathrm{ms}$ with $\Delta t_c = \frac{1}{f_c}$. For a frame rate of $f_c = 30\,\mathrm{fps}$, the last 15 values are fed into the network. This study aims to apply controlled artificial intelligence by merging deep networks with low level computations. In this regard, the scene information within the saliency maps is manually extracted, rather than training directly on the saliency maps. In the scope of this study, two exploitation approaches are investigated:

1. The location of the most attractive region within the viewport is calculated by extracting the maximum salient points $\max\left(S(t), ..., S(t - W_c)\right)$ for all saliency maps within the window W_c.

2. The location of the saliency map's centroid $(\bar{x}, \bar{y})$, which corresponds to its center of mass, is computed according to:

$$\bar{x} = \frac{m_{10}}{m_{00}}, \quad \bar{y} = \frac{m_{01}}{m_{00}}. \tag{6.39}$$

   by using the spatial image moments with:

$$m_{pq} = \sum_x \sum_y x^p y^q S(x, y). \tag{6.40}$$

Figure 6.15 illustrates saliency maps with a varying amount of saliency-emphasized regions and the locations that are picked with respect to the exploitation policy. Taking the maximum salient point or the centroids of the saliency maps (saliency moments) results in different locations that are passed as inputs for the S-network. The extracted location describes the relative motion of the salient region with respect to the current head position. The reprojecting technique introduced in Chapter 2 is used to infer the location of the desired salient point in the 3D world. Since the images are in an equirectangular format, only the pan and tilt rotations can be deduced. These locations are then normalized to be in the range of -1 to 1 and convolved with a 1D kernel of size 30 acting as a low-pass filter.

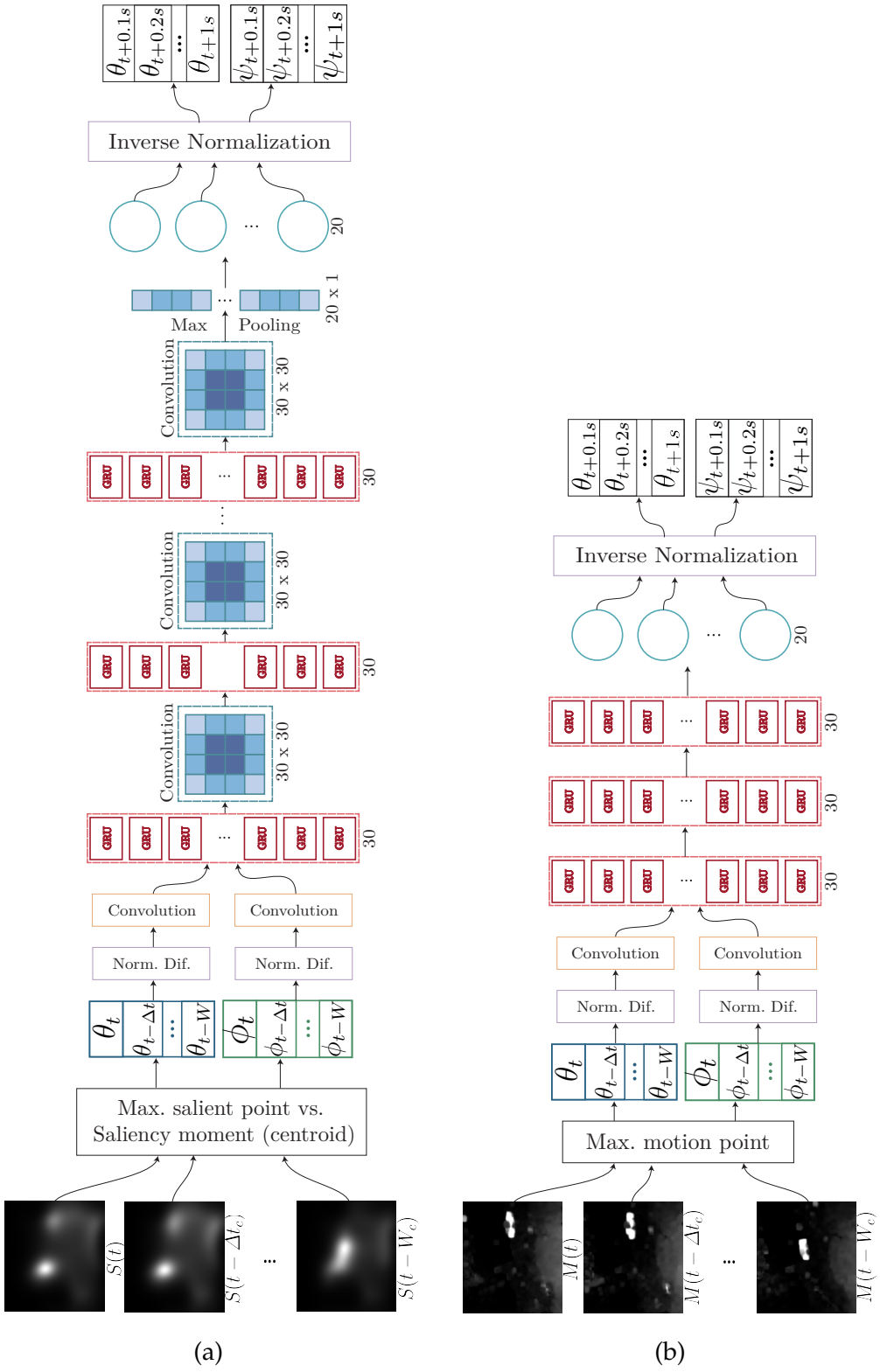
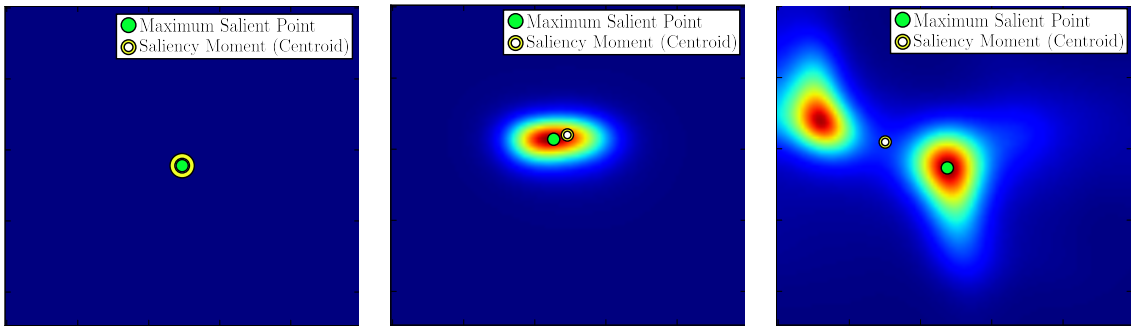


Figure 6.14: Overview of the proposed saliency and motion exploitation deep architectures. (a) The saliency network (S-network) uses low-level techniques to extract the most distinct features for saliency maps within a window W_c and passes them into a network that is based on an interleaved structure of three stacked GRU and convolutional layers. The subsequent max pooling layer ensures the desired output dimension by conserving the most distinct features. (b) The motion network (M-network) takes the strongest motion flows within W_c and passes them into a deep structure based on the three GRU layers. Neither network contains any information about the roll rotation, and thus output prospective pan and tilt orientation values for a course of 1 s.



(a) (b) (c)

Figure 6.15: The max-based and centroid-based exploitation techniques are demonstrated for three saliency maps with varying saliency regions within the viewport. The maximum-based utilization methods grabs the highest salient pixel value as prospective orientation. The centroid-based method uses the image moments to detect the center of mass as potential future head orientation.

The preprocessed input data is then passed through an interleaved structure of five stacked GRU and convolution layers. The GRU layers consist of 30 cells each. The convolution units are designed with a kernel size of 30×30. A max pooling layer is followed to reduce the dimension to the desired output size, conserving the most distinct features. A final dense FFN is added to increase the number of parameters to be learned. The saliency network is visualized in Figure 6.14 (a).

The reprojecting technique introduced in Chapter 2 is used to infer the location of the desired salient point in the 3D world. Since the images are in an equirectangular format, only the pan and tilt rotations can be deduced. These locations are then normalized to be in the range of -1 to 1 and convolved with a 1D kernel of size 30 acting as a low-pass filter. The preprocessed input data is then passed through an interleaved structure of five stacked GRU and convolution layers. The GRU layers consist of 30 cells each. The convolution units are designed with a kernel size of 30×30. A max pooling layer is followed to reduce the dimension to the desired output size, conserving the most distinct features. A final dense FFN is added to increase the number of parameters to be learned. The saliency network is visualized in Figure 6.14 (a).

6.5.4.2 Motion Network

Empirical experiments lead to a slightly different design for the motion network, as is illustrated in Figure 6.14 (b). The input of the motion network is inspired by the saliency network. The core of the proposed architecture consists, however, of three GRU layers, each with 30 cells, followed by a dense FFN with 30 nodes.

6.5.4.3 Dataset Allocation

The IMT dataset is employed to train the networks. Four video sequences are used for training, one for validation, and one for testing. The mean absolute error is chosen as loss function (L1-norm). The Adam optimizer is used to update network weights. The initial learning rate

is set to 10^{-3}. A learning rate decay scheduler is incorporated to decrease the initial learning rate every ten epochs by 10 %. The maximum number of epochs is thereby constrained to be 1000. The early stopping technique is embedded to monitor the validation loss and stop the learning process ahead of schedule with patience of ten epochs.

6.5.4.4 Deep Fusion

After individually training the H-, S-, and M-networks, the goal is to fuse them in a way to optimally exploit useful information. Empirical investigations yielded the *deep fusion network (F-network)*, presented in Figure 6.16. After removing the last layers of the H-, S-, and M-networks, their outputs are fed into a new type of deep fusion network. The F-network consists of an interleaved structure of GRU layers, each with 60 cells, and convolution units with a kernel size of 60×60. Similar to the previous networks, a max pooling layer is followed to reduce the dimensions to the desired output size, while preserving important features. Three dense FFNs, each with 20 nodes, are added before remapping them to absolute orientation values by computing the inverse of their normalized differences. The F-network aims to optimize the pan and tilt orientations in particular. The prediction for roll is adopted from the H-network, as saliency and motion maps do not contribute any relevant information for roll rotations.

The output of the F-network is trained to be a whole course of future orientations from 0.1 s to 1 s with step size 0.1 s. During the training process of the fusion network, the core of the individual H-, S-, and M-network are defined to be non-trainable to leave the previously trained parameters unchanged. The IMT dataset is separated into a training, validation, and testing set as was previously described. The batch size is specified to 2^{11} with a learning rate of 0.001. The learning rate decay scheduler is deployed again to decrease the initial learning rate every ten epochs by 100 % for a maximum number of 1000 epochs. The Adam optimizer computes adaptive learning rates for each parameter and is used to update the network's parameters accordingly. Overfitting is prevented by using an early stopping technique with the patience of two epochs. ReLUs are utilized as activation functions for the dense FFNs. All deep learning architectures and models are designed and trained with Keras and TensorFlow. Despite its deepness, the fusion network is able to infer future orientation data in the domain of microseconds (Core i7 (x64), GeForce GTX 1080 Ti) and is hence still suitable for realtime applications.

6.5.5 Results

The late-fusion strategy necessitates a base network that is robust against external influences and able to deliver stable and reliable prediction values without any scene information. The head orientation-based deep network (H-network) is revised and restructured to precisely meet these demands. The IMT dataset is used to compare the delay-dependent mean absolute errors, and the root mean square errors (for pan rotations) of the H-network to the prior art. Figure 6.17 (a) and (b) contrast the H-network with machine learning-based approaches that were introduced previously.

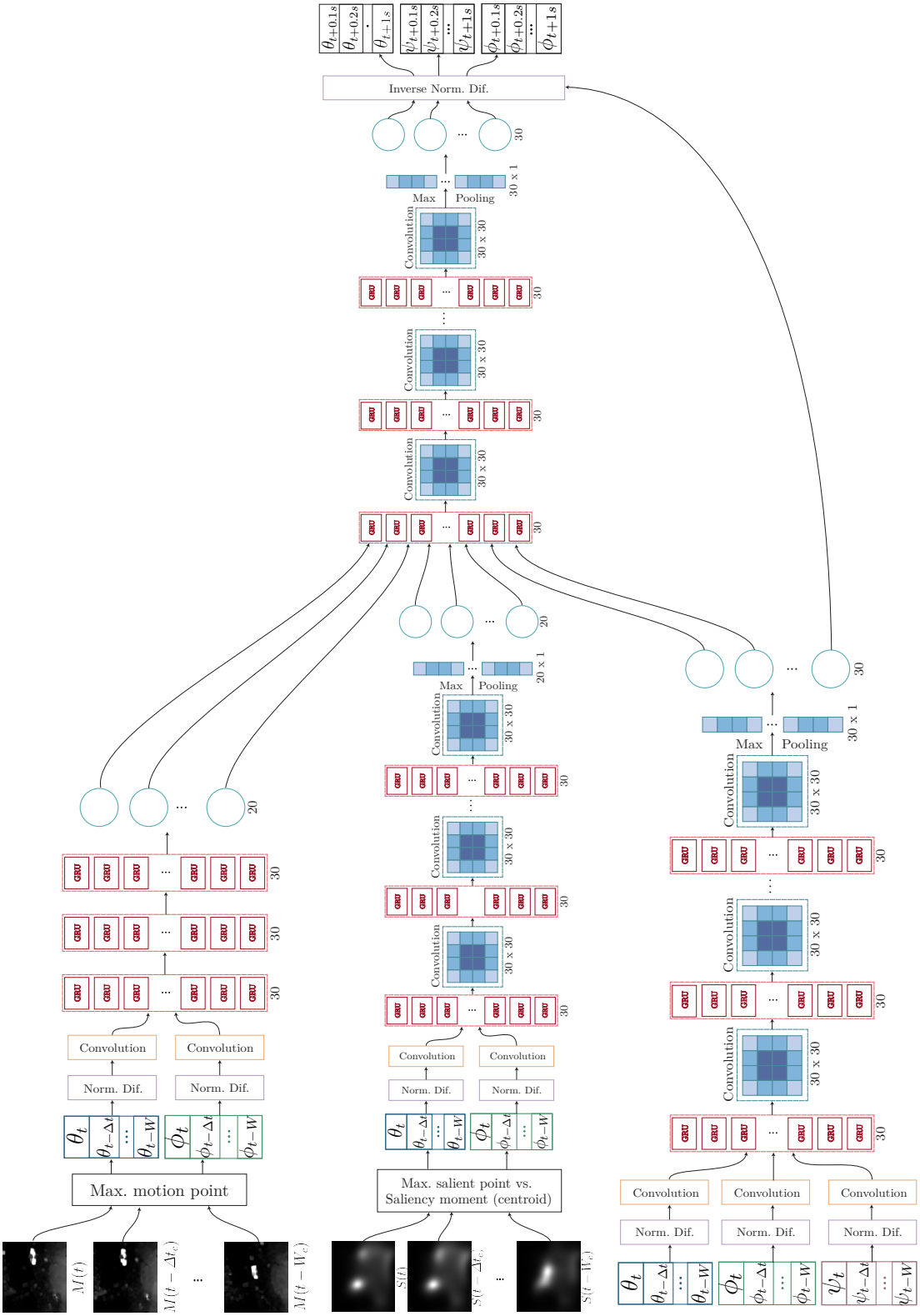


Figure 6.16: This schematic presentation illustrates the final deep late-fusion network (F-network) that merges the outcomes of the saliency and motion map with the outcomes of the orientation data-based deep architectures for the pan and tilt orientations. The outputs of the individual networks are passed into a stacked architecture of GRU and convolutional layers. The prospective roll values are taken from the H-network.

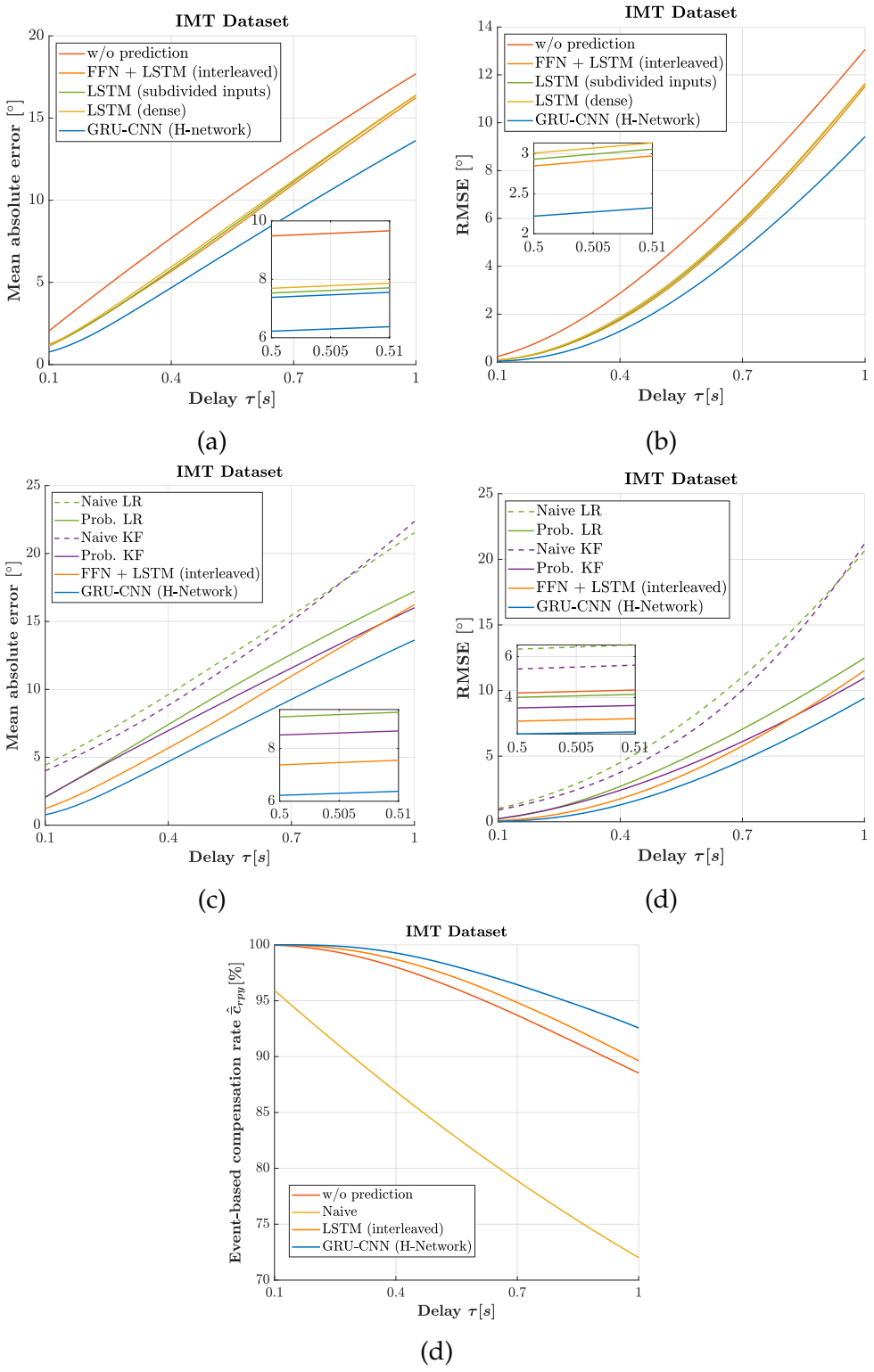


Figure 6.17: The proposed H-network, which uses the deep orientation data-based architecture (GRU-CNN), is compared to previous work and representative state-of-the-art methods. The mean absolute error, the root mean square error, and the compensation rate are selected as metrics for a fair comparison. (a) and (b) compare the H-network to the previously investigated deep architectures. The results clearly show that the H-network outperforms prior work. The plots in (c) and (d) are used to analyze the H-network's achievements in contrast to state-of-the-art representatives. The H-network significantly outperforms prior art and exhibits fewer erroneous prediction values. The compensation rate in (d) is presented to convey the achievable degree of delay-compensation. The results confirm the superiority of the H-network's performance when compared to prior work.

The figures demonstrate that although prior deep learning-based methods are able to perform well, they are not able to keep up with the innovative H-network.

The proposed H-network is further contrasted to state-of-the-art representatives as well as the probabilistic extensions thereof, which are depicted in Figure 6.17 (c) and (d). It is apparent that the H-network is able to outperform the prior art by large margins. These findings are likewise reflected in the achievable degree of delay-compensation, which is depicted in Figure 6.17 (d). The compensation rates of the H-network are compared to the best performing prior work. The achievable level of delay-compensation is substantially better, which is especially evident for larger latencies.

The orientation-based deep architecture is then fused with spatio-temporal information for superior viewport prediction. Saliency maps and optical flow-based motion maps are used, respectively, to describe the spatial and temporal features within the viewport. The key information within these maps is leveraged by calculating their relative motion to the most distinct locations. For saliency maps, the location of the most distinct features is determined to be either the most salient point or the saliency map's centroid, which is based on the image's moment. For motion maps, the location of the maximum flow is assumed to be the most distinct feature. This information is then fused by means of the deep fusion network (F-network) as displayed in Figure 6.16.

Evaluation for pan rotations The IMT dataset is harnessed to visualize the mean absolute and root mean square errors of the proposed predictors with respect to horizontal motions in Figure 6.18 (a) and (b). The two deep fusion networks, which rest on the maximum-based and centroid-based exploitation technique of saliency maps, are compared to the individual H-, S-, and M-networks to investigate their respective isolated contributions. The S-network is inspected both for the maximum-based and moment-based computation policy. The moment-based saliency exploitation evinces higher errors than the maximum-based technique. The maximum-based method performs even better than the previously introduced LSTM and FFN-based deep architecture. Merely exploiting saliency maps based on computing the centroids seems to deliver no relevant information for viewport prediction. Their errors are only slightly lower than the method where no prediction is employed at all. Inspecting the outcomes of the DeepGaze II network disclosed mostly a center-biased, single dominant salient region. Computing the center of mass thereof appears to be an exploitation strategy that is too passive for viewport prediction. This tendency is, however, not discernible for the fused network, where the moment-based saliency retrieval is able to contribute relevant information to the overall performance and even leads to an improved performance compared to the head orientation-based (GRU-CNN) deep architecture that is based on an interleaved structure of GRUs and convolutional neural networks (CNN). As part of the deep fusion network, the H-network is identified as the most dominant and influential one. Higher errors are observed when fusing the H-network with the maximum saliency-based S-network. Relying instead only on the maximum motion flow as potential prospective head orientation shows, however, only minor tweaks, leaving considerable room for improvements with respect to the deployed exploitation technique. Spatial scene infor-

mation appear to be more useful for viewport prediction than the temporal one at pixel level. The MAE and RMSE values for the optical flow-based network are even higher than computing no prediction at all. However, these findings are explicitly observed for the performance of horizontal motions.

Evaluation for tilt rotations Similar tendencies are observed for the mean absolute error with respect to tilt orientations, which are visualized in Figure 6.18 (c). Merely relying on optical flow-based information led to the worst performance. Computing even no prediction at all proved better. The LSTM-based network does not show a major improvement compared to the prediction-less processing and performs especially poorly for high latency values. Improvements are observed for both saliency-based prediction approaches. The moment-based saliency exploitation realizes better prediction for tilt orientations. This phenomenon is also reflected in the fused networks. The centroid-based utilization of saliency maps yielded the best results for the deep fusion network and even beat the orientation-based GRU-CNN (H-) network. The deep fusion network, which gathered maximum salient points, showed worse performance when compared to the H-network.

Evaluation for roll rotations As saliency and motion maps do not contain any awareness of roll rotations, this information needs to be provided by the orientation data-based deep networks. The mean absolute errors of roll rotations are illustrated in Figure 6.18 (d) for the GRU-CNN (H-) and the FFN+LSTM-based networks. Both approaches are compared to the naive version, which employs no prediction. The evaluation for roll rotations shows a surprising outcome. Computing no prediction for roll rotations performs significantly better than any prediction technique. It appears to be of secondary importance that the H-network performs marginally better than the LSTM-based network. This finding is incorporated into the resulting deep fusion framework. The roll rotations are no longer considered for prediction. Instead, the current roll position is leveraged as prospective orientation. This adoption holds true for the individual motion and saliency network as well as the fusion networks, which are accordingly modified and denoted as *Deep Fusion (Moments) - Modified* and *Deep Fusion (Max) - Modified*.

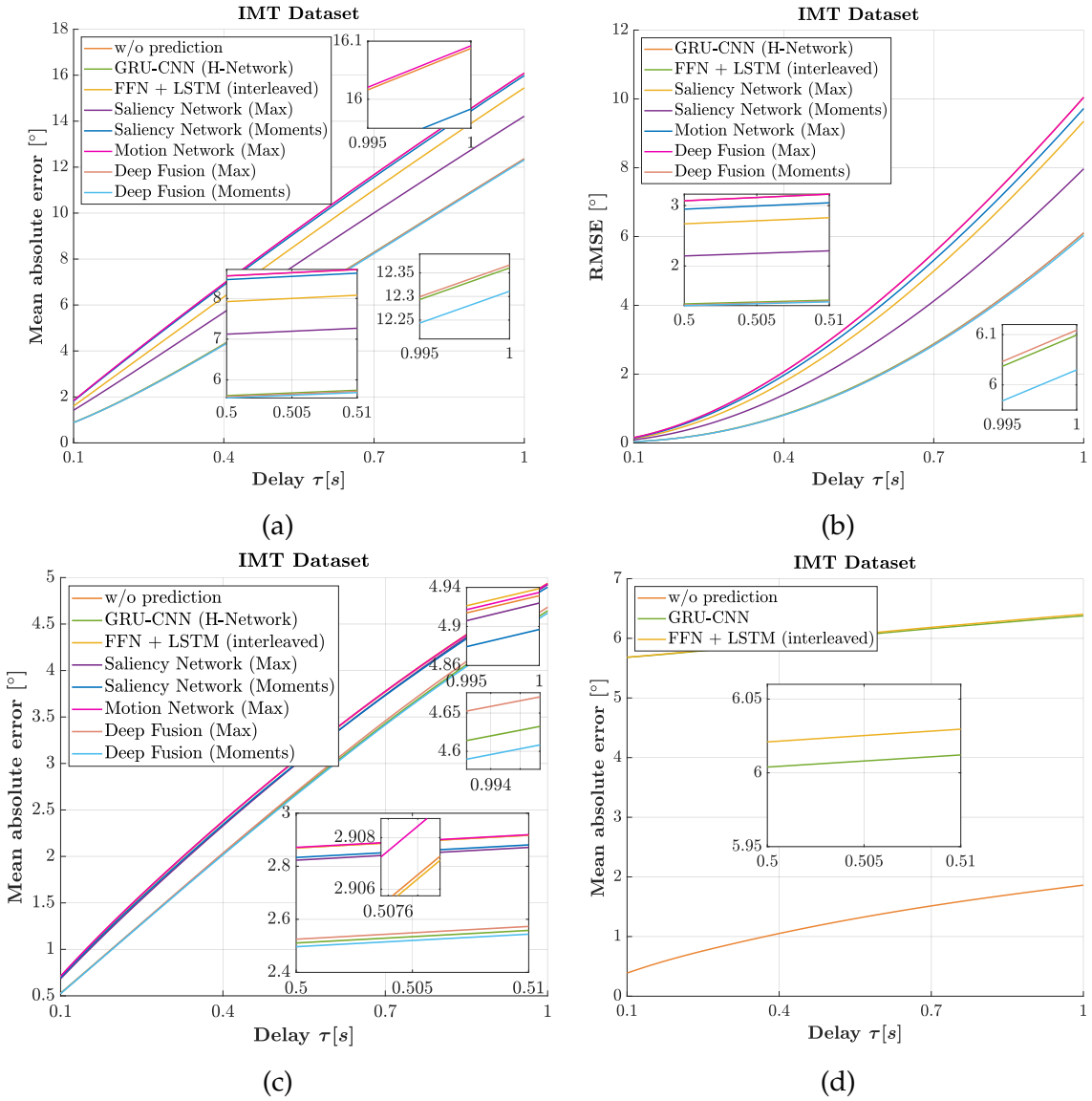


Figure 6.18: The mean absolute error and the root mean square error are used to convey the accuracy of the prediction approaches for horizontal motions. (a) and (b) juxtapose the error values of deep fusion network (each for the max-based and centroid-based exploitation technique), the motion network, the saliency network – both for the max-based and centroid-based exploitation technique –, the head orientation-based network, as well as the best-performing previous deep architecture. The results show that the moment-based deep fusion network outperforms any other prediction strategy. The mean absolute error for tilt and roll rotations are plotted in (c) and (d). Computing the predictions for roll rotations performs significantly worse when compared to employing no prediction at all.

Achieved delay-compensation Figure 6.19 (a) juxtaposes the compensation rate of the proposed approaches next to the prior art. The optical flow-based prediction method is only able to minimally improve the compensation rate with respect to the prediction-less process, despite slightly larger mean absolute errors. Discernible improvements are instead achieved with the LSTM-based deep architecture, which perform better than the moments-based but worse than the maximum saliency-based S-network. It is interesting to note that solely deploying the maximum saliency-based exploitation policy for the S-network

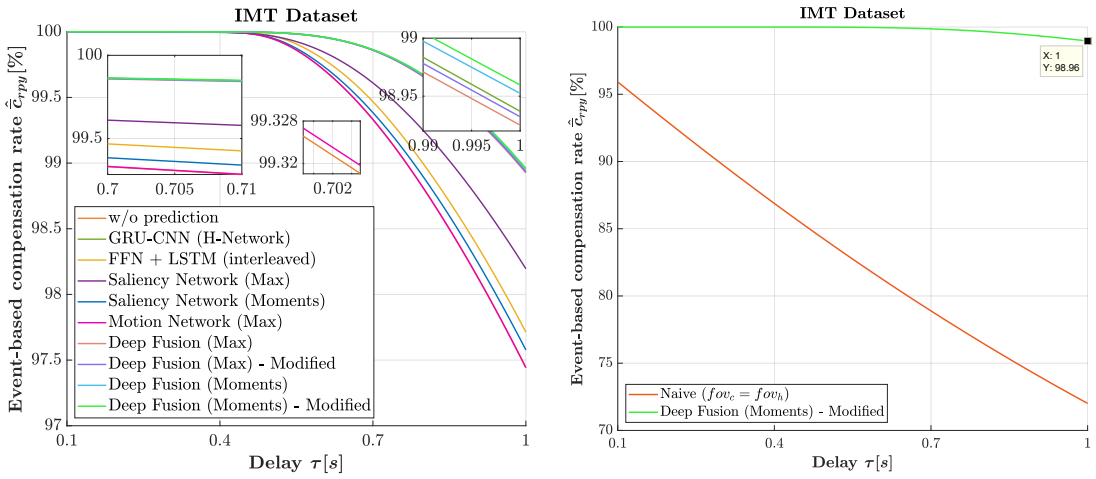


Figure 6.19: (a) The compensation rates of the fusion networks (also for their modified versions) are compared to the individual networks that are based on saliency and motion maps. Head orientation data are presented with respect to the delay in question to convey the achievable degree of delay-compensation. The results verify that the modified deep fusion network that leverage the moments-based exploitation technique, clearly outperforms the prior art. (b) The best-performing deep fusion network along with its buffer-based compensation paradigm is contrasted with the naive version where no compensation and prediction is applied. The results unambiguously confirm the favorable benefits of the delay-compensation strategy over the naive technique. Mean compensation rates of almost 99 % are even achieved for latencies as high as 1 s.

yields remarkable compensation abilities; particularly compared to the centroid-based extraction technique. However, this tendency is not reflected in the fused F-network. Although the maximum saliency-based network individually performed superiorly compared to its centroid-based pendant, the deep fusion network seems to better exploit the information of the moment-based S-network. Fusing the S-network based on maximum salient points revealed more unsatisfactory results than the head orientation-based H-network itself.

The results further prove that the modification of the deep fusion networks is able to deliver enhanced achievements. The deep fusion network that is premised on the moment-based saliency architecture clearly outperforms the state-of-the-art and demonstrates a substantial level of reachable delay-compensation. Figure 6.19 is used to compare the final buffer-based delay-compensation algorithm, including its proper viewport prediction policy, to the naive approach, where compensation and prediction are not deployed. A mean compensation rate of 99.99 % is achieved for the investigated latency values in the range of 0.1 s to 1 s. Even for a delay as high as 1 s, a compensation rate of almost 99 % is feasible. The results unambiguously indicate the superiority of the proposed approach, which ensures instantaneous visual feedback for the user despite the presence of large delays.

6.5.6 Application to On-demand 360° Video Streaming

The overall objective of the proposed semantic viewport prediction algorithms is to optimize the immersiveness and the quality of experience of telepresence applications. However, the approaches presented here are not limited to realtime telepresence. This section aims to prove its applicability for live/offline 360° video streaming applications.

Both on-demand and live streaming of high-resolution 360° videos are characterized by high bandwidth and low latency requirements. A proof-of-concept scenario is implemented to highlight the benefits of highly accurate viewport prediction.

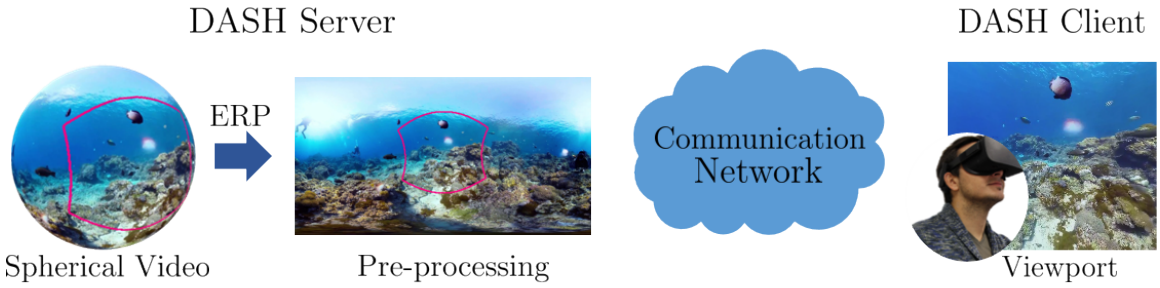


Figure 6.20: Overview of a typical streaming pipeline for immersive 360° videos. The spherical video is first converted into a 2D equirectangular projection format and then tiled for better resource allocation. The visual data is then exchanged between the DASH server and client. The user is equipped with an HMD to select the desired viewport intuitively.

A Dynamic Adaptive Streaming over HTTP (DASH) based communication network is established to stream (offline) 360° video content on-demand between a server and a client as depicted in Figure 6.20. Similar to traditional 2D video streaming, DASH is a method to provide bandwidth adaptability. Chapter 2 introduced streaming techniques that are especially useful for streaming 360° video content. Streaming the complete 360° video footage claims large portions of the transmission capacity although only the viewport can be consumed by the user. Within the scope of the proof-of-concept, a tile-based, viewport adaptive streaming policy is adopted to reduce the massive rate requirements and ensure a high quality of experience by means of proper viewport prediction.

The spherical video is first projected into the 2D equirectangular format (see Chapter 2) to be able to leverage state-of-the-art video encoding standards. The projected 2D video is then spatially divided into tiles and temporally bundled into time segments (chunks of 1 s). Practical experiments proved that a 4×8 tiling scheme is a good trade-off in terms of encoding efficiency and high granularity for superior viewport adaptability. Each chunk is independently encoded (HEVC, main profile) at different bitrates. The quantization parameters (QP) have been varied from 22 to 42 with a step size of 5 to provide different bitrates for each tile. Lower QP values result in less distortion but higher bitrates per tile. In doing so, there are five representations stored at the server, which can be requested from the client with regard to available resources. The DASH client is responsible for the adaptation behavior and requests a suitable representation for each tile at the beginning of the segment download. Prior to the segment download, the client determines the target bitrate based on the esti-

mated throughput of the communication network and the video's buffer level as proposed in [156]. The decision on each tile's bitrate is premised on the proposed viewport prediction algorithm. Higher bitrates are assigned to tiles that lie within the predicted viewport. With respect to the target bitrate and the predicted viewport location, a rate-distortion optimized tile quality selection is performed. The overall objective is to find the optimal bitrate for each tile such that the QoE of the user is maximized.

The QoE is defined as a measure that describes the weighted distortion and the spatial quality variance, which needs to be minimized for the best user experience [157]. The implemented optimization framework subdivides the image into a grid of $N = 4 \times 8 = 32$ tiles. Each tile n has B possible bitrates b. The different bitrates are obtained by varying the QP of the HEVC codec. d_{nb} and R_{nb} term the level of the present distortion and bitrate of tile n at the bth representation, respectively. Tiles that lie within the field of view of the predicted viewport trajectory for the next time segment receive higher bitrates than tiles in the peripheral. A normalized weight $\tilde{w}_n$ is assigned to each tile to quantify its relevance for bitrate allocation:

$$\tilde{w}_n = \frac{w_n}{\sum_{i=1}^{B} w_i}, \quad \text{with} \quad w_n = \frac{1}{\Delta_n^2 + 1}, \tag{6.41}$$

where Δ_n denotes the Euclidean distance of a tile n to the center of the closest tile that belongs to the prospective viewport. Given the equirectangular image in the YUV format, the distortion d_{nb} is computed as the mean squared error between the Y-frames of the original and the reconstructed tiled videos $\hat{Y}$:

$$d_{nb} = \frac{1}{\mathrm{w} \cdot \mathrm{h}} \sum_{i=1}^{\mathrm{w}} \sum_{j=1}^{\mathrm{h}} \left(Y(i,j) - \hat{Y}(i,j) \right)^2, \tag{6.42}$$

with w and h denoting the tiles' width and height within the ERP plane. The distortion that is computed by means of the ERP tile does not accurately reflect the perceived distortion by the user, who is exposed to the spherical video. The actual distortion (mean squared error) of each tile with respect to the viewing sphere can then be computed by weighing the distortion d_{nb} with a value c_n:

$$d'_{nb} = d_{nb} \cdot c_n, \tag{6.43}$$

which accounts for the reprojection effects. c_n is defined as the ratio of the spherical area of tile n to its corresponding tile size on the ERP plane:

$$c_n = \frac{\int_{\phi_n}^{\phi_n + \frac{\pi}{4}} \int_{\theta_n}^{\theta_n + \frac{\pi}{4}} R^2 \cos(\phi)\, d\theta d\phi}{\left(\frac{\pi R}{4}\right)^2}, \tag{6.44}$$

with R being the sphere's radius (see Figure 2.12). To ensure a satisfying QoE, the *weighted distortion* Λ and the *spatial quality variance* Ξ are introduced as functions that are to be opti-

mized [157]:

$$\Lambda = \sum_{i=1}^{N} \sum_{j=1}^{B} \tilde{w}_i \cdot d'_{ij} \cdot a_{ij}, \tag{6.45}$$

$$\Xi = \sum_{i=1}^{N} \sum_{j=1}^{B} \tilde{w}_i \cdot \left| d'_{ij} - \Lambda \right| \cdot a_{ij}, \tag{6.46}$$

where $a_{ij} = 1$ if tile i is selected to be streamed at the jth quality. The spatial quality variance indicates the perceivable spatial smoothness for a given video quality. The objective is to minimize the overall optimization function that is defined as the conical sum (with weight ν) of the weighted distortion and the spatial quality variance:

$$\min \quad \Lambda + \nu \cdot \Xi, \tag{6.47}$$

$$\text{s.t.} \quad \sum_{i=1}^{N} \sum_{j=1}^{B} \mathcal{R}_{ij} \cdot a_{ij} \quad \leq \quad \mathcal{R}_{\text{target}}, \tag{6.48}$$

$$\sum_{j=1}^{B} a_{ij} = 1, \quad \text{with} \;\; a_{ij} \in \{0, 1\}. \tag{6.49}$$

The two constraints ensure that the overall sum of bitrates is less than or equal to the target bitrate $\mathcal{R}_{\text{target}}$ and that only one representation is selected per tile.

To apply fast and efficient optimization solvers [158], the underlying optimization problem, which uses a nonlinear function for the spatial quality variance, is converted into an integer linear programming (ILP) problem with quadratic constraints. First, an auxiliary variable $k_{ij} = d'_{ij} - \Lambda$ is introduced to reformulate the spatial quality variance as:

$$\Xi = \sum_{i=1}^{N} \sum_{j=1}^{B} \tilde{w}_i \cdot k_{ij} \cdot a_{ij}. \tag{6.50}$$

Further constraints are set to ensure the absolute value of k_{ij} according to:

$$k_{ij} \geq d'_{ij} - \Lambda \quad \text{and} \quad k_{ij} \leq -\left(d'_{ij} - \Lambda\right). \tag{6.51}$$

A further auxiliary measure l_{ij} is used to remove the nonlinear multiplication with a_{ij}:

$$\Xi = \sum_{i=1}^{N} \sum_{j=1}^{B} \tilde{w}_i \cdot l_{ij}, \;\; \text{with} \;\; l_{ij} = k_{ij} \cdot a_{ij}, \tag{6.52}$$

obtaining an ILP-based optimization problem with quadratic constraints.

Quality of experience and quality of service are measures that indicate the user's satisfaction while being exposed to a video streaming session. Metrics conveying this information for 360° video streaming comprise bandwidth utilization, buffer stall ratio, perceived video quality, and viewport quality smoothness. Special focus is placed on the perceived video quality during a streaming session to highlight the benefits of the proposed viewport prediction paradigm. One commonly accepted indicator for the perceived video quality is the peak

signal-to-noise ratio (PSNR). Computing the PSNR within the equirectangular video format does not directly reflect the perceived video quality due to projection artifacts. Rather than computing the reprojected viewport for every head orientation and subsequently calculating the PSNR, a computationally more efficient method is given by the weighted-to-spherically-uniform (WS-) PSNR [159]. The WS-PSNR accounts for projection artifacts and computes the viewport PSNR by using the viewport pixels within the ERP plane. This is done by assigning projection-dependent weights to each pixel. The weight distribution for the equirectangular projection format is portrayed in Figure 6.21 (a) and can be computed for a pixel location (x, y) according to [159]:

$$w_{\text{ERP}}(x, y) = \cos\left(\left(y - \frac{h_{\text{ERP}}}{2} + \frac{1}{2}\right) \cdot \frac{\pi}{2}\right), \tag{6.53}$$

where h_{ERP} denotes the ERP plane's height. Due to the oversampling effect of the equirectangular projection towards the pole regions, weights are decreased from the equator to the poles of the ERP plane. The WS-PSNR for an image I_{ERP} in the ERP format can then be computed as [159]:

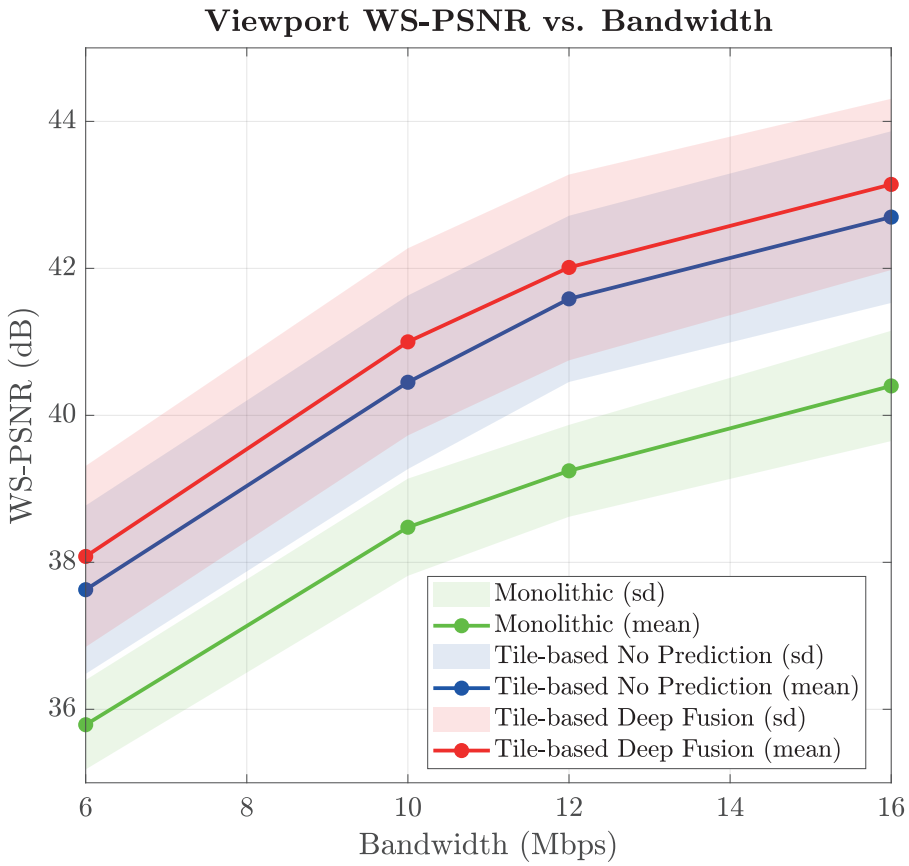
$$\text{WS-PSNR} = 10 \log\left(\frac{(\max(I_{\text{ERP}}(x, y))^2)}{\text{WMSE}}\right), \tag{6.54}$$

by utilizing the weighted mean square error (WMSE) for a viewport V as:

$$\text{WMSE} = \sum_{(x,y) \in V} \left(Y(x, y) - \hat{Y}(x, y)\right)^2 \cdot \frac{w_{\text{ERP}}(x, y)}{\sum\limits_{(x,y) \in V} w_{\text{ERP}}(x, y)}. \tag{6.55}$$

One representative video of the IMT dataset is selected to inspect the WS-PSNR with respect to the consumed transmission rate. The average viewport WS-PSNR, including its mean and standard deviation regarding the available network bandwidth, is visualized for six users in Figure 6.21 (b). The network throughput is kept constant during streaming. Multiple sessions are carried out for different throughput values, which range from 6 Mbps to 16 Mbps. The WS-PSNR is thereby examined for the monolithic streaming, where the whole $360°$ video footage is sent, compared to the tile-based approach and its extension with the proposed viewport prediction paradigm. The results confirm that tile-based streaming yields superior viewport WS-PSNR values compared to the monolithic streaming. Adopting the proposed viewport prediction technique clearly demonstrates further improvements. Significantly higher viewport WS-PSNR values are achieved when deploying tile-based streaming along with proper viewport prediction.

(a)



(b)

Figure 6.21: Pictorial illustration of the weight distribution of the WS-PSNR for equirectangular projection formats. (b) Results showing the perceived video quality in terms of WS-PSNR over the consumed transmission rate. Adopting the modified deep-learning-based late-fusion paradigm helps to greatly improve the perceived video quality while claiming significantly less bandwidth.

6.6 Chapter Summary

This chapter introduced several contributions with respect to viewport prediction. The deep learning-based approach was first applied to head orientation data for viewport prediction and compared to the prior art. Various architectures were proposed and investigated. Next, scene information was incorporated into the forecasting paradigm to optimize the prediction accuracy. Spatio-temporal information was inferred by computing saliency and motion maps. The saliency maps are generated by using a state-of-the-art deep learning framework for saliency detection. The motion maps are produced via optical flow-based low-level computations. To be robust against external influences, such as poor lighting conditions or occlusions, and to be agnostic to the underlying hardware system, a late-fusion strategy was chosen to fuse spatio-temporal information with the head motion data. The core idea is to provide highly precise prediction capabilities using head motion data – provided in high frequency – and augment the predictability with spatio-temporal scene information. In doing so, individual networks are first pre-trained to exploit spatial and temporal scene information and merged in a deep fusion network. The head orientation-based prediction algorithm was greatly advanced by leveraging a structure of stacked GRUs and convolutional layers. The fused network shows remarkable performance and is able to provide a compensation rate of 99.99 % for investigated latencies between 0.1 s to 1 s on the IMT dataset. The proposed concept is not only applicable for realtime telepresence systems but can also be embedded in live and on-demand 360° video streaming applications. Utilizing the presented prediction paradigm helps to significantly improve the WS-PSNR values with respect to the consumed transmission rate as compared to prediction-less processing.

Chapter 7

Conclusion

Omnistereoscopic telepresence is a technology that promises great advances in many areas such as healthcare, education, entertainment, and science. Its way to lasting acceptance and dissemination is limited by the challenges that first need to be overcome. Some of the difficulties for providing immersive telepresence are described and discussed in Chapter 1. This thesis addresses the challenge of mediating visual information of a far off located remote environment in realtime. The overall objective is to provide the user depth-sensing, high acuity vision of the distant scene without perceiving the lag between head-motion and visual response.

7.1 Summary

A server-/client-based telepresence setup is developed that leverages a stereo-camera setup with an electro-mechanical augmentation unit to mirror the user's head motion in three degrees-of-freedom. A delay-compensation scheme is proposed that mitigates the perceived delay and helps to improve the realistic feeling of presence. The overall approach is a combination of three individual solutions. The fundament of the presented technique applies a buffer-based delay-compensation technique that uses wide-angle or fisheye cameras to capture a greater visual field of the remote scene than is actually displayed to the human user. The residual image content around the user's viewport defines the available buffer that is leveraged for local delay-compensation. When the user rotates his/her head, the remaining footage can be used for instantaneous visual feedback until the updated image frame arrives. The bottleneck of this methodology stems from quick head rotations or abrupt orientation changes. In this case, it is highly probable that the viewport is outside of the buffer region and only a subregion of the image content can be used for local delay compensation. The methodology is extended by a psycho-physically-founded velocity-based field-of-view adaptation technique that temporarily constricts the visual field as a function of the current head rotation velocity. This allows for momentarily larger buffer regions resulting in advanced delay-compensation abilities during quick rotations. A subjective pilot-study is conducted to investigate the perceived experience of human users who are exposed to a telepresence scenario by means of a dynamic field-of-view adaptation technique. Results verify that the presented methodology does not affect the feeling of presence and additionally helps

to mitigate the emergence of motion sickness. The adaptive field-of-view modification can be seen as an optional element that can be incorporated into the delay-compensation algorithm and is especially favorable for latencies in the range of 0.1 s to 0.5 s. An additional vital component of the delay-compensation paradigm is given by a proper viewport prediction policy. A late-fusion strategy is proposed that does not only consider past head orientation data that are collected from the orientation sensor embedded in the head-mounted-display but also leverages spatial and temporal scene information within the viewport. Saliency maps and motion maps are created for the corresponding viewport to quantify the spatial and temporal information. Two exploitation policies are investigated to infer the most crucial information within these maps. The maximum salient point exploitation technique is compared to the centroid-based one, which uses the moments of saliency maps to find its center of mass. The temporal information within motion maps is extracted by selecting the maximum flow thereof. The proposed deep fusion network that combines head orientation data with spatio-temporal scene information for proper viewport prediction evinces substantial improvements compared to the prior art. A remarkable high mean compensation rate of 99.99 % is achieved for the tested user profiles.

7.2 Limitations

The proposed delay-compensation scheme is able to positively address the challenges depicted in Figure 1.1. The proposed delay-compensation vision system is realtime-capable, provides depth perception in any desired viewing direction and is characterized by a sizeable stereoscopic budget. Even the time for deep learning-based viewport prediction is in the domain of microseconds. However, this property does not hold true for the saliency map creation. Chapter 6 introduced the DeepGaze II network as a benchmark for saliency map generation. The results show high-accurate estimations of saliencies for a given image content. The DeepGaze II network requires, unfortunately, a lot of processing time to produce such saliency maps. The time needed to create one saliency map with respect to the requested viewport is in the range of seconds per frame. That is why the saliency maps are produced in a lengthy offline pre-processing step prior to the exploitation stage.

It should be further noted that the dataset used for training the fusion network of spatio-temporal scene information is limited and could greatly benefit from more videos and user profiles. The training and inference is expected to become better for a larger dataset.

Another limitation is given by the camera system, which is only supposed to be applied for first-person view telepresence systems. To transfer the conceptual ideas to multi-view streaming necessitates structural modifications of the camera system.

7.3 Future Work

This work aimed to approach significant challenges that hamper the mainstream acceptance of telepresence systems. Although the delay-compensation approach addressed many critical issues and helped to greatly improve the achievable level of delay-compensation and thus the visual comfort of the user, there remain still some subjects that show room for improvement. These subjects will be briefly discussed in the following along with some potential approaches that might help to improve them.

Buffer-based delay-compensation The buffer-based latency-compensation model already managed to greatly decrease the motion-to-photon latency by providing immediate access through a broadened visual scene coverage. It is evident that this approach is highly subject to the provided buffer size. A complete 360° visual representation of the remote scene would infer the maximum amount of buffer. It should be noted that using two panorama cameras in a stereo camera setup for omnistereoscopic vision is not favorable as is pictured in Figure 2.9. The effective baseline between the two cameras decreases with the rotation angle. The smaller the baseline is, the weaker the depth perception becomes. A novel camera solution is highly preferred that is able to account for the reduced depth perception during head rotations. It is conceivable to apply three (or four cameras) instead of two. Deployed in a triangular configuration, they might help to create an even larger visual field while keeping the effective baseline constant.

Dynamic field-of-view adaptation The velocity-based field-of-view adaptation technique is psycho-physically motivated and shows excellent potential. The conducted subjective tests confirmed its positive effect on the retrievable level of delay-compensation and helped to suppress the emergence of motion sickness. This study is, however, a pilot study and needs additional extensive experiments to investigate its long-term implication. The current adaptation technique constricts the visual field symmetrically in both the horizontal and vertical direction. Further research is needed to incorporated further characteristics of the human eye. Gaze behavior, for instance, is an important subject that needs to be addressed in future work. The prior art verified that gaze motion precedes the head movement by around 40 ms to 60 ms [144], [160]. There is also the so-called vestibulo-ocular reflex that demands special attention [161]. The vestibulo-ocular reflex is a mechanism that moves the gaze oppositely to the head-motion to enhance the quality of vision by stabilizing the viewing direction. Incorporating such mechanisms are likely to improve the visual comfort of the user further. The eye's gaze can thereby not only be leveraged for the dynamic field-of-view adaptation but has also great potential for a positive bias of the viewport prediction quality.

Semantic viewport prediction The virtues of the semantic viewport prediction paradigm by means of spatio-temporal viewport exploitation is highly subject to the performance of the DeepGaze II saliency map creation and the optical flow-based motion maps. While the optical flow computations can be computed in realtime, this is yet not the case for the saliency

map creation, which performs in the domain of seconds per frame. Further research is required to either apply pruning techniques to address the high computational costs in the deep convolutional architectures or develop novel saliency creation approaches that are able to run in realtime.

The deep architecture proposed in this work uses mainly the mean absolute error as loss function to train the deep networks. The loss function $\mathcal{L}$ is used to estimate the loss of the current model between the true y and predicted value $\hat{y}$. The weights of the network are updated so that the loss is minimized for the next epoch:

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^{n} \left| y^{(i)} - \hat{y}^{(i)} \right|. \tag{7.1}$$

In doing so, the network is trained to perfectly estimate the prospective head orientation values for a subsequent course of 1 s. When applying the buffer-based delay-compensation approach, however, it is not necessarily required to impeccably predict the exact location as long as the predicted trajectory stays within the available buffer region. A novel loss function that is based on the compensation rate can be incorporated to train a network to compensate for a given delay such as:

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^{n} \left| 1.0 - c\left(y^{(i)}, \hat{y}^{(i)} \right) \right|. \tag{7.2}$$

It should be noted that the compensation does not always have one optimal global value for given pan, tilt, and roll values. It is highly probable that the network will be stuck in local minima without significantly improving the degree of delay-compensation. The delay-compensation model has to be modified in a way to exhibit one dominant global minima for optimal training.

Moreover, providing one single deep neural network approach that is supposed to perform accurately for all individuals in the world is not only challenging but also unrealistic considering the varying anatomy of each person, or their motion habits. It would be of great interest to develop a solution that is able to adapt to each user's motion behavior and to each task at hand by means of artificial intelligence that continues learning at every session and thus optimizes the prediction accuracy substantially. This can be achieved by devising an offline/online prediction policy that uses a pre-trained base model as foundation and customizes the network's weights through an online learning process. Conceivable is a deep reinforcement learning approach that uses a pre-trained model as environment and action model.

Bibliography

Publications by the author

Journal publications

[1] T. Aykut, M. Karimi, C. Burgmair, A. Finkenzeller, C. Bachhuber, and E. Steinbach, "Delay compensation for a telepresence system with 3d 360 degree vision based on deep head motion prediction and dynamic fov adaptation," *Robotics and Automation Letters*, vol. 3, no. 4, pp. 4343–4350, 2018.

[2] T. Aykut, J. Xu, and E. Steinbach, "Realtime 3d 360-degree telepresence with deep-learning-based head-motion prediction," *Journal on Emerging and Selected Topics in Circuits and Systems*, 2019.

Conference publications

[3] M. Karimi, T. Aykut, and E. Steinbach, "Mavi: A research platform for telepresence and teleoperation," *arXiv preprint arXiv:1805.09447*, 2018.

[4] T. Aykut, S. Lochbrunner, M. Karimi, B. Cizmeci, and E. Steinbach, "A stereoscopic vision system with delay compensation for 360° remote reality," in *Proceedings of the on Thematic Workshops of ACM Multimedia*, Mountain View, California, USA: ACM, 2017, pp. 201–209.

[5] T. Aykut, C. Zou, J. Xu, D. Van Opdenbosch, and E. Steinbach, "A delay compensation approach for pan-tilt-unit-based stereoscopic 360 degree telepresence systems using head motion prediction," in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 1–9.

[6] T. Aykut, C. Burgmair, M. Karimi, J. Xu, and E. Steinbach, "Delay compensation for actuated stereoscopic 360 degree telepresence systems with probabilistic head motion prediction," in *Winter Conference on Applications of Computer Vision*, Lake Tahoe, USA: IEEE, 2018, pp. 2010–2018.

[7] D. Van Opdenbosch, M. Oelsch, A. Garcea, T. Aykut, and E. Steinbach, "Selection and compression of local binary features for remote visual slam," in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 7270–7277.

[8] D. Van Opdenbosch, T. Aykut, N. Alt, and E. Steinbach, "Efficient map compression for collaborative visual slam," in *Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2018, pp. 992–1000.

[9] J. Xu, A. Bhardwaj, G. Sun, T. Aykut, N. Alt, M. Karimi, and E. Steinbach, "Learning-based modular task-oriented grasp stability assessment," in *RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 3468–3475.

General publications

[10] M. Slater and M. Usoh, "Presence in immersive virtual environments," in *Virtual Reality Annual International Symposium*, IEEE, 1993, pp. 90–96.

[11] W. Barfield and S. Weghorst, "The sense of presence within virtual environments: A conceptual framework," *Advances in Human Factors Ergonomics*, vol. 19, pp. 699–699, 1993.

[12] M. Slater and M. Usoh, "Representations systems, perceptual position, and presence in immersive virtual environments," *Presence: Teleoperators & Virtual Environments*, vol. 2, no. 3, pp. 221–233, 1993.

[13] C. Hendrix and W. Barfield, "Presence within virtual environments as a function of visual display parameters," *Presence: Teleoperators & Virtual Environments*, vol. 5, no. 3, pp. 274–289, 1996.

[14] S. M. LaValle, A. Yershova, M. Katsev, and M. Antonov, "Head tracking for the oculus rift," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, IEEE, 2014, pp. 187–194.

[15] J. T. Reason and J. J. Brand, *Motion sickness*. Academic press, 1975.

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

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

[18] M. A. Watson and F. Black, "The human balance system: A complex coordination of central and peripheral systems," *Portland, OR: Vestibular Disorders Association*, 2008.

[19] R. H. So, W. Lo, and A. T. Ho, "Effects of navigation speed on motion sickness caused by an immersive virtual environment," *Human Factors*, vol. 43, no. 3, pp. 452–461, 2001.

[20] D. Drascic, "Skill acquisition and task performance in teleoperation using monoscopic and stereoscopic video remote viewing," in *Proceedings of the Human Factors Society Annual Meeting*, SAGE Publications, Los Angeles, CA, vol. 35, 1991, pp. 1367–1371.

[21] J. Y. Chen, E. C. Haas, and M. J. Barnes, "Human performance issues and user interface design for teleoperated robots," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1231–1245, 2007.

[22] R. S. Allison, B. J. Gillam, and E. Vecellio, "Binocular depth discrimination and estimation beyond interaction space," *Journal of Vision*, vol. 9, no. 1, pp. 10–10, 2009.

[23] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An invitation to 3-d vision: from images to geometric models*. Springer Science & Business Media, 2012, vol. 26.

[24] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[25] J. M. Van Verth and L. M. Bishop, *Essential mathematics for games and interactive applications: a programmer's Guide*. CRC Press, 2008.

[26] D. Forsyth and J. Ponce, *Computer vision : a modern approach*. Prentice Hall, 2003.

[27] C. Loop and Z. Zhang, "Computing rectifying homographies for stereo vision," in *Computer Vision and Pattern Recognition*, IEEE, vol. 1, 1999, pp. 125–131.

[28] A. Fusiello, E. Trucco, and A. Verri, "A compact algorithm for rectification of stereo pairs," *Machine Vision and Applications*, vol. 12, no. 1, pp. 16–22, 2000.

[29] P. Monasse, J.-M. Morel, and Z. Tang, "Three-step image rectification," in *British Machine Vision Conference*, BMVA Press, 2010, pp. 89–1.

[30] C. B. Duane, "Close-range camera calibration," *Photogramm. Eng*, vol. 37, no. 8, pp. 855–866, 1971.

[31] J. Weng, P. Cohen, and M. Herniou, "Camera calibration with distortion models and accuracy evaluation," *Transactions on Pattern Analysis & Machine Intelligence*, no. 10, pp. 965–980, 1992.

[32] T. A. Clarke and J. G. Fryer, "The development of camera calibration methods and models," *The Photogrammetric Record*, vol. 16, no. 91, pp. 51–66, 1998.

[33] J. Wang, F. Shi, J. Zhang, and Y. Liu, "A new calibration model of camera lens distortion," *Pattern Recognition*, vol. 41, no. 2, pp. 607–615, 2008.

[34] K. Miyamoto, "Fish eye lens," *J. Opt. Soc. Am.*, vol. 54, no. 8, pp. 1060–1061, 1964.

[35] M. M. Fleck, "The wrong imaging model," Technical Report TR 95-01, University of Iowa, Tech. Rep., 1995.

[36] C. Hughes, P. Denny, E. Jones, and M. Glavin, "Accuracy of fish-eye lens models," *Applied Optics*, vol. 49, no. 17, pp. 3338–3347, 2010.

[37] A. Basu and S. Licardie, "Alternative models for fish-eye lenses," *Pattern Recognition Letters*, vol. 16, no. 4, pp. 433–441, 1995.

[38] S. Shah and J. Aggarwal, "Intrinsic parameter calibration procedure for a (high-distortion) fish-eye lens camera with distortion model and accuracy estimation," *Pattern Recognition*, vol. 29, no. 11, pp. 1775–1788, 1996.

[39] F. Devernay and O. Faugeras, "Straight lines have to be straight," *Machine Vision and Applications*, vol. 13, no. 1, pp. 14–24, 2001.

[40] D. Gonzalez-Aguilera, J. Gomez-Lahoz, and P. Rodríguez-Gonzálvez, "An automatic approach for radial lens distortion correction from a single image," *Sensors journal*, vol. 11, no. 4, pp. 956–965, 2011.

[41] S. Chan, H.-Y. Shum, and K.-T. Ng, "Image-based rendering and synthesis," *Signal Processing Magazine*, vol. 24, no. 6, pp. 22–33, 2007.

[42] E. H. Adelson, J. R. Bergen, *et al.*, "The plenoptic function and the elements of early vision," 1991.

[43] L. McMillan and G. Bishop, "Plenoptic modeling: An image-based rendering system," in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, 1995, pp. 39–46.

[44] H.-Y. Shum, S.-C. Chan, and S. B. Kang, *Image-based rendering*. Springer Science & Business Media, 2008.

[45] S. E. Chen, "Quicktime vr: An image-based approach to virtual environment navigation," in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, 1995, pp. 29–38.

[46] R. Szeliski, H.-Y. Shum, H.-Y. Shum, and H.-Y. Shum, "Creating full view panoramic image mosaics and environment maps," in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 1997, pp. 251–258.

[47] H. Shum and S. B. Kang, "Review of image-based rendering techniques," in *Visual Communications and Image Processing*, International Society for Optics and Photonics, vol. 4067, 2000, pp. 2–14.

[48] R. Szeliski, "Image alignment and stitching: A tutorial," *Foundations and Trends in Computer Graphics and Vision*, vol. 2, no. 1, pp. 1–104, 2007.

[49] H.-Y. Shum and R. Szeliski, "Construction of panoramic image mosaics with global and local alignment," in *Panoramic Vision*, Springer, 2001, pp. 227–268.

[50] M. Levoy, "Light fields and computational imaging," *Computer*, vol. 39, no. 8, pp. 46–55, 2006.

[51] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, 1996, pp. 31–42.

[52] H.-Y. Shum and L.-W. He, "Rendering with concentric mosaics," in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., 1999, pp. 299–306.

[53] S. E. Chen and L. Williams, "View interpolation for image synthesis," in *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, 1993, pp. 279–288.

[54] M. Lhuillier and L. Quan, "Image interpolation by joint view triangulation," in *Computer Vision and Pattern Recognition*, IEEE, vol. 2, 1999, pp. 139–145.

[55] S. M. Seitz and C. R. Dyer, "View morphing," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, 1996, pp. 21–30.

[56] S. Laveau and O. D. Faugeras, "3-d scene representation as a collection of images," in *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, IEEE, vol. 1, 1994, pp. 689–691.

[57] S. Avidan and A. Shashua, "Novel view synthesis in tensor space," in *Computer Vision and Pattern Recognition, Proceedings, Computer Society Conference on*, IEEE, 1997, pp. 1034–1040.

[58] L. McMillan, "An image-based approach to three-dimensional computer graphics," PhD thesis, Citeseer, 1997.

[59] Y. Mori, N. Fukushima, T. Yendo, T. Fujii, and M. Tanimoto, "View generation with 3d warping using depth information for ftv," *Signal Processing: Image Communication*, vol. 24, no. 1-2, pp. 65–72, 2009.

[60] S. Xiang, H. Deng, L. Zhu, J. Wu, and L. Yu, "Exemplar-based depth inpainting with arbitrary-shape patches and cross-modal matching," *Signal Processing: Image Communication*, vol. 71, pp. 56–65, 2019.

[61] K. Muller, P. Merkle, and T. Wiegand, "3-d video representation using depth maps," *Proceedings*, vol. 99, no. 4, pp. 643–656, 2011.

[62] M. Tanimoto, M. P. Tehrani, T. Fujii, and T. Yendo, "Free-viewpoint tv," *Signal Processing Magazine*, vol. 28, no. 1, pp. 67–76, 2011.

[63] P. Ndjiki-Nya, M. Koppel, D. Doshkov, H. Lakshman, P. Merkle, K. Muller, and T. Wiegand, "Depth image-based rendering with advanced texture synthesis for 3-d video," *Transactions on Multimedia*, vol. 13, no. 3, pp. 453–465, 2011.

[64] L.-w. He, J. Shade, S. Gortler, and R. Szeliski, "Layered depth images," 1998.

[65] P. E. Debevec, C. J. Taylor, and J. Malik, "Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, 1996, pp. 11–20.

[66] C. Demonceaux, P. Vasseur, and Y. Fougerolle, "Central catadioptric image processing with geodesic metric," *Image and Vision Computing*, vol. 29, no. 12, pp. 840–849, 2011.

[67] A. A. Kostrzewski, S. Ro, I. Agurok, and M. Bennahmias, *Panoramic video system with real-time distortion-free imaging*, US Patent 7,336,299, 2008.

[68] J. Borden, *Panoramic indexing camera mount*, US Patent 5,752,113, 1998.

[69] L. E. Gurrieri and E. Dubois, "Acquisition of omnidirectional stereoscopic images and videos of dynamic scenes: A review," *Journal of Electronic Imaging*, vol. 22, no. 3, pp. 030 902–030 902, 2013.

[70] C. Birklbauer and O. Bimber, "Panorama light-field imaging," in *Computer Graphics Forum*, Wiley Online Library, vol. 33, 2014, pp. 43–52.

[71] J. Foote, S. Ahmad, and J. Boreczky, *Automatic video system using multiple cameras*, US Patent 7,015,954, 2006.

[72] M. Brown and D. G. Lowe, "Automatic panoramic image stitching using invariant features," *International Journal of Computer Vision*, vol. 74, no. 1, pp. 59–73, 2007.

[73] Z. Zhu, "Omnidirectional stereo vision," in *International Conference on Robotics and Automation*, 2001, pp. 22–25.

[74] S. Peleg and M. Ben-Ezra, "Stereo panorama with a single camera," in *Computer Vision and Pattern Recognition, Computer Society Conference on.*, IEEE, vol. 1, 1999, pp. 395–401.

[75] C. Richardt, Y. Pritch, H. Zimmer, and A. Sorkine-Hornung, "Megastereo: Constructing high-resolution stereo panoramas," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1256–1263.

[76] C. Weissig, O. Schreer, P. Eisert, and P. Kauff, "The ultimate immersive experience: Panoramic 3d video acquisition," in *International Conference on Multimedia Modeling*, Springer, 2012, pp. 671–681.

[77] R. Aggarwal, A. Vohra, and A. M. Namboodiri, "Panoramic stereo videos with a single camera," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3755–3763.

[78] O. Schreer, P. Kauff, P. Eisert, C. Weissig, and J.-C. Rosenthal, "Geometrical design concept for panoramic 3d video acquisition," in *Signal Processing Conference (EUSIPCO), Proceedings of the 20th European*, IEEE, 2012, pp. 2757–2761.

[79] R. Anderson, D. Gallup, J. T. Barron, J. Kontkanen, N. Snavely, C. Hernández, S. Agarwal, and S. M. Seitz, "Jump: Virtual reality video," *Transactions on Graphics*, vol. 35, no. 6, p. 198, 2016.

[80] *Facebook surround 360*, https://facebook360.fb.com/ facebook-surround-360/, last accessed: March 1, 2019.

[81] *Nokia ozo*, https://ozo.nokia.com/eu/, last accessed: March 1, 2019.

[82] *Samsung - project beyond*, http://thinktankteam.info/beyond/, last accessed: March 1, 2019.

[83] K. Matzen, M. F. Cohen, B. Evans, J. Kopf, and R. Szeliski, "Low-cost 360 stereo photography and video capture," *Transactions on Graphics*, vol. 36, no. 4, 148:1–148:12, 2017.

[84] R. Konrad, D. G. Dansereau, A. Masood, and G. Wetzstein, "Spinvr: Towards live-streaming 3d virtual reality video," *Transactions on Graphics*, vol. 36, no. 6, 209:1–209:12, Nov. 2017.

[85] *Jaunt - cinematic vr*, https://www.jauntvr.com/ last accessed: February 27, 2019.

[86] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[87] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[88] J. P. Snyder, *Map projections–A working manual*. US Government Printing Office, 1987, vol. 1395.

[89] M. Yu, H. Lakshman, and B. Girod, "A framework to evaluate omnidirectional video coding schemes," in *International Symposium on Mixed and Augmented Reality*, IEEE, 2015, pp. 31–36.

[90] C.-W. Fu, L. Wan, T.-T. Wong, and C.-S. Leung, "The rhombic dodecahedron map: An efficient scheme for encoding panoramic video," *Transactions on Multimedia*, vol. 11, no. 4, pp. 634–644, 2009.

[91] J. Li, Z. Wen, S. Li, Y. Zhao, B. Guo, and J. Wen, "Novel tile segmentation scheme for omnidirectional video," in *International Conference on Image Processing*, IEEE, 2016, pp. 370–374.

[92] J.-R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (hevc)," *Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1669–1684, 2012.

[93] *Oculus*, https://www.oculus.com/, last accessed: March 2, 2019.

[94] *Htc vive*, https://www.vive.com/de/, last accessed: March 2, 2019.

[95] *Samsung gear vr*, https://www.samsung.com/global/galaxy/gear-vr/ last accessed: March 2, 2019.

[96] X. Corbillon, A. Devlic, G. Simon, and J. Chakareski, "Optimal set of 360-degree videos for viewport-adaptive streaming," in *Proceedings of the 2017 ACM on Multimedia Conference*, ACM, 2017, pp. 943–951.

[97] E. Kuzyakov and D. Pio, *Next-generation video encoding techniques for 360 video and vr*, https://code.fb.com/virtual-reality/next-generation-video-encoding-techniques-for-360-video-and-vr/, last accessed: March 2, 2019.

[98] X. Corbillon, G. Simon, A. Devlic, and J. Chakareski, "Viewport-adaptive navigable 360-degree video delivery," in *International Conference on Communications*, IEEE, 2017, pp. 1–7.

[99] T. Stockhammer, "Dynamic adaptive streaming over http–: Standards and design principles," in *Proceedings of the Second Annual Conference on Multimedia Systems*, ACM, 2011, pp. 133–144.

[100] K. Misra, A. Segall, M. Horowitz, S. Xu, A. Fuldseth, and M. Zhou, "An overview of tiles in hevc," *Journal of Selected tTopics in Signal Processing*, vol. 7, no. 6, pp. 969–977, 2013.

[101] A. Zare, A. Aminlou, M. M. Hannuksela, and M. Gabbouj, "Hevc-compliant tile-based streaming of panoramic video for virtual reality applications," in *Proceedings of the 24th International Conference on Multimedia*, ACM, 2016, pp. 601–605.

[102] V. R. Gaddam, M. Riegler, R. Eg, C. Griwodz, and P. Halvorsen, "Tiling in interactive panoramic video: Approaches and evaluation," *Transactions on Multimedia*, vol. 18, no. 9, pp. 1819–1831, 2016.

[103] J. Chakareski, R. Aksu, X. Corbillon, G. Simon, and V. Swaminathan, "Viewport-driven rate-distortion optimized 360º video streaming," in *International Conference on Communications*, IEEE, 2018, pp. 1–7.

[104] R. H. So and M. J. Griffin, "Experimental studies of the use of phase lead filters to compensate lags in head-coupled visual displays," *Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 26, pp. 445–454, 1996.

[105] J. Y. Jung, B. D. Adelstein, and S. R. Ellis, "Discriminability of prediction artifacts in a time-delayed virtual environment," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, SAGE Publications, vol. 44, 2000, pp. 499–502.

[106] R. Azuma and G. Bishop, "A frequency-domain analysis of head-motion prediction," in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, 1995, pp. 401–408.

[107] H. Himberg and Y. Motai, "Head orientation prediction: Delta quaternions versus quaternions," *Transactions on Systems, Man, and Cybernetics*, vol. 39, no. 6, pp. 1382–1392, 2009.

[108] C. Bachhuber and E. Steinbach, "Are today's video communication solutions ready for the tactile internet?" In *Wireless Communications and Networking Conference Workshops*, IEEE, 2017, pp. 1–6.

[109] G. A. Seber and A. J. Lee, *Linear regression analysis*. John Wiley & Sons, 2012, vol. 329.

[110] G. Welch, G. Bishop, *et al.*, "An introduction to the kalman filter," 1995.

[111] P. M. Djuric, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez, "Particle filtering," *Signal Processing Magazine*, vol. 20, no. 5, pp. 19–38, 2003.

[112] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, ACM, 2016, pp. 1–6.

[113] Y. Bao, H. Wu, T. Zhang, A. A. Ramli, and X. Liu, "Shooting a moving target: Motion-prediction-based transmission for 360-degree videos," in *International Conference on Big Data*, IEEE, 2016, pp. 1161–1170.

[114] A. Mavlankar and B. Girod, "Video streaming with interactive pan/tilt/zoom," in *High-Quality Visual Experience*, Springer, 2010, pp. 431–455.

[115] C.-L. Fan, J. Lee, W.-C. Lo, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, "Fixation prediction for 360 video streaming in head-mounted virtual reality," in *Workshop on Network and Operating Systems Support for Digital Audio and Video*, ACM, 2017, pp. 67–72.

[116] T. Alshawi, Z. Long, and G. AlRegib, "Understanding spatial correlation in eye-fixation maps for visual attention in videos," in *International Conference on Multimedia and Expo*, IEEE, 2016, pp. 1–6.

[117] S. Chaabouni, J. Benois-Pineau, and C. B. Amar, "Transfer learning with deep networks for saliency prediction in natural video," in *International Conference on Image Processing*, IEEE, 2016, pp. 1604–1608.

[118] T. V. Nguyen, M. Xu, G. Gao, M. Kankanhalli, Q. Tian, and S. Yan, "Static saliency vs. dynamic saliency: A comparative study," in *International Conference on Multimedia*, ACM, 2013, pp. 987–996.

[119] Y. Xu, Y. Dong, J. Wu, Z. Sun, Z. Shi, J. Yu, and S. Gao, "Gaze prediction in dynamic 360° immersive videos," in *Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5333–5342.

[120] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[121] X. Corbillon, F. De Simone, and G. Simon, "360-degree video head movement dataset," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, 2017.

[122] *Fraunhofer heinrich hertz institut*, https://www.hhi.fraunhofer.de/ last accessed: March 12, 2019.

[123] L. Merritt and R. Vanam, "X264: A high performance h. 264/avc encoder," *http://neuron2.net/library/avc/overview_x264_v8_5. pdf*, 2006.

[124] M. Carbone and L. Rizzo, "Dummynet revisited," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 2, pp. 12–20, 2010.

[125] J. M. Van Verth and L. M. Bishop, *Essential mathematics for games and interactive applications*. CRC Press, 2015.

[126] G. A. Korn and T. M. Korn, *Mathematical Handbook for Scientists and Engineers: Definitions, Theorems, and Formulas for Reference and Review*, ser. Dover Civil and Mechanical Engineering Series. Dover Publications, 2000.

[127] K. W. Arthur and F. P. Brooks Jr, "Effects of field of view on performance with head-mounted displays," PhD thesis, University of North Carolina at Chapel Hill, 2000.

[128] C. Jerome, R. Darnell, B. Oakley, and A. Pepe, "The effects of presence and time of exposure on simulator sickness," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, SAGE Publications Sage CA, vol. 49, 2005, pp. 2258–2262.

[129] A. F. Seay, D. M. Krum, L. Hodges, and W. Ribarsky, "Simulator sickness and presence in a high field-of-view virtual environment," in *Extended Abstracts on Human Factors in Computing Systems*, ACM, 2002, pp. 784–785.

[130] W. IJsselsteijn, H. d. Ridder, J. Freeman, S. E. Avons, and D. Bouwhuis, "Effects of stereoscopic presentation, image motion, and screen size on subjective and objective corroborative measures of presence," *Presence: Teleoperators & Virtual Environments*, vol. 10, no. 3, pp. 298–311, 2001.

[131] P. DiZio and J. R. Lackner, "Circumventing side effects of immersive virtual environments," in *HCI (2)*, 1997, pp. 893–896.

[132] A. S. Fernandes and S. K. Feiner, "Combating vr sickness through subtle dynamic field-of-view modification," in *3D User Interfaces*, IEEE, 2016, pp. 201–210.

[133] N. Kala, K. Lim, K. Won, J. Lee, T. Lee, S. Kim, and W. Choe, "P-218: An approach to reduce vr sickness by content based field of biew processing," in *SID Symposium Digest of Technical Papers*, Wiley Online Library, vol. 48, 2017, pp. 1645–1648.

[134] T. Schubert, F. Friedmann, and H. Regenbrecht, "The experience of presence: Factor analytic insights," *Presence: Teleoperators & Virtual Environments*, vol. 10, no. 3, pp. 266–281, 2001.

[135] R. S. Kennedy, N. E. Lane, K. S. Berbaum, and M. G. Lilienthal, "Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness," *The international journal of aviation psychology*, vol. 3, no. 3, pp. 203–220, 1993.

[136] J. O. Rawlings, S. G. Pantula, and D. A. Dickey, *Applied regression analysis: a research tool*. Springer Science & Business Media, 2001.

[137] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[138] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[139] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[140] F. Chollet *et al.*, *Keras*, https://keras.io, 2015.

[141] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015. [Online]. Available: https://www.tensorflow.org/.

[142] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[143] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[144] V. Sitzmann, A. Serrano, A. Pavel, M. Agrawala, D. Gutierrez, B. Masia, and G. Wetzstein, "Saliency in vr: How do people explore virtual environments?" *Transactions on visualization and computer graphics*, vol. 24, no. 4, pp. 1633–1642, 2018.

[145] M. Kummerer, T. S. Wallis, L. A. Gatys, and M. Bethge, "Understanding low-and high-level contributions to fixation prediction," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4789–4798.

[146] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[147] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand, "What do different evaluation metrics tell us about saliency models?" *CoRR*, 2016. [Online]. Available: http://arxiv.org/abs/1604.03605.

[148] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[149] A. Borji, M.-M. Cheng, Q. Hou, H. Jiang, and J. Li, "Salient object detection: A survey," *arXiv preprint arXiv:1411.5878*, 2014.

[150] E. Vig, M. Dorr, and D. Cox, "Large-scale optimization of hierarchical features for saliency prediction in natural images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2798–2805.

[151] M. Kümmerer, L. Theis, and M. Bethge, "Deep gaze i: Boosting saliency prediction with feature maps trained on imagenet," *arXiv preprint arXiv:1411.1045*, 2014.

[152] X. Huang, C. Shen, X. Boix, and Q. Zhao, "Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 262–270.

[153] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "A deep multi-level network for saliency prediction," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, IEEE, 2016, pp. 3488–3493.

[154] Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, and A. Torralba, *Mit saliency benchmark*, 2015.

[155] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Scandinavian conference on Image analysis*, Springer, 2003, pp. 363–370.

[156] G. Tian and Y. Liu, "Towards agile and smooth video adaptation in dynamic http streaming," in *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, ACM, 2012, pp. 109–120.

[157] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming," in *Proceedings of the 25th ACM international conference on Multimedia*, ACM, 2017, pp. 315–323.

[158] *Gurobi optimization*, http://gurobi.com/, last accessed: May 4, 2019.

[159] Y. Sun, A. Lu, and L. Yu, "Weighted-to-spherically-uniform quality evaluation for omnidirectional video," *IEEE signal processing letters*, vol. 24, no. 9, pp. 1408–1412, 2017.

[160] E. G. Freedman, "Coordination of the eyes and head during visual orienting," *Experimental brain research*, vol. 190, no. 4, p. 369, 2008.

[161] V. Laurutis and D. Robinson, "The vestibulo-ocular reflex during human saccadic eye movements.," *The Journal of Physiology*, vol. 373, no. 1, pp. 209–233, 1986.

List of Figures

List of Tables